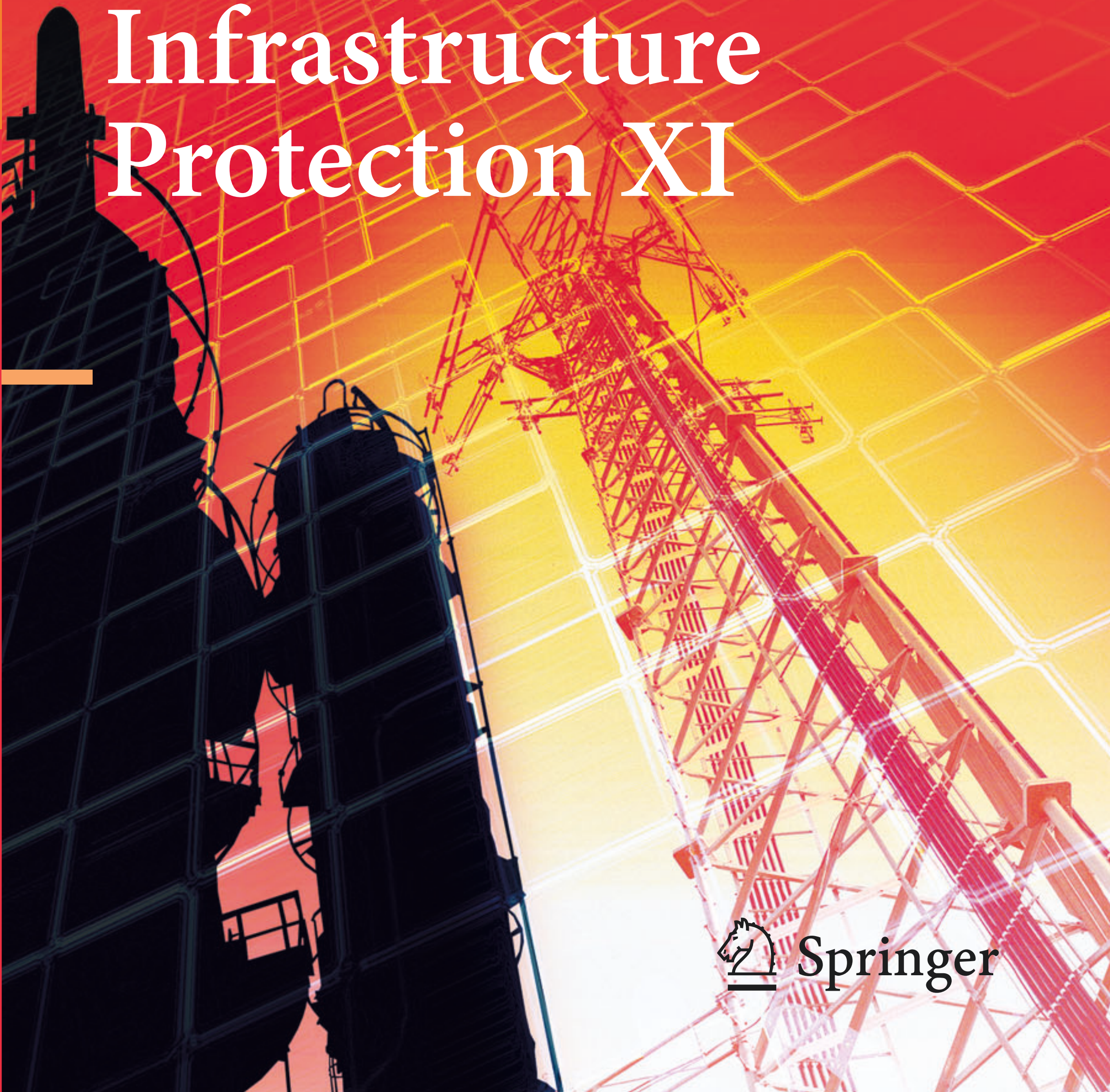
IFIP AICT 512

Mason Rice
Sujeet Shenoi
(Eds.)

Critical Infrastructure Protection XI



Springer

Mason Rice · Sujeet Shenoi (Eds.)

Critical Infrastructure Protection XI

11th IFIP WG 11.10 International Conference, ICCIP 2017
Arlington, VA, USA, March 13–15, 2017
Revised Selected Papers

Springer

Editors
Mason Rice
Air Force Institute of Technology
Wright-Patterson Air Force Base, OH
USA

Sujeet Shenoi
University of Tulsa
Tulsa, OK
USA

ISSN 1868-4238 ISSN 1868-422X (electronic)
IFIP Advances in Information and Communication Technology
ISBN 978-3-319-70394-7 ISBN 978-3-319-70395-4 (eBook)
https://doi.org/10.1007/978-3-319-70395-4

Library of Congress Control Number: 2017959615

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Contents

Contributing Authors

Sofia Belikovetsky is a Ph.D. student in Information Systems Engineering at Ben-Gurion University of the Negev, Beer-Sheva, Israel. Her research focuses on the security of additive manufacturing processes and systems.

Jason Bindewald is an Assistant Professor of Computer Science at the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. His research interests include autonomous systems, machine learning, multi-agent system design and human-machine teaming.

Raymond Chan is a Ph.D. student in Computer Science at the University of Hong Kong, Hong Kong, China. His research interests include digital forensics and critical infrastructure protection.

Kam-Pui Chow is an Associate Professor of Computer Science at the University of Hong Kong, Hong Kong, China. His research interests include information security, digital forensics, live system forensics and digital surveillance.

Edward Colbert is a Cyber Security Researcher at the U.S. Army Research Laboratory in Adelphi, Maryland. His research interests include cyber-physical system security and Internet of Things security, especially in tactical environments.

Bogdan Crainicu is an Assistant Professor of Computer Science at Petru Maior University of Tirgu-Mures, Mures, Romania. His research interests include network and computer security, cryptography, software-defined networking and cloud computing architectures.

Joseph Daoud is an M.S. student in Computer Science at the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. His research interests include cyber operations and critical infrastructure protection.

Antonio Di Pietro is a Staff Scientist at the Laboratory for the Analysis and Protection of Critical Infrastructures, ENEA, Rome, Italy. His research interests include decision support systems for emergency management, geographical information systems and infrastructure modeling.

Adrian-Vasile Duka is an Assistant Professor of Engineering at Petru Maior University of Tirgu-Mures, Mures, Romania. His research interests include control systems engineering and cyber-physical system protection.

Stephen Dunlap is a Cyber Security Research Engineer at the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. His research interests include embedded system security, cyber-physical system security and critical infrastructure protection.

Yuval Elovici is a Professor of Information Systems Engineering, Director of the Telekom Innovation Laboratories and Head of the Cyber Security Research Center at Ben-Gurion University of the Negev, Beer-Sheva, Israel. His research interests include computer security and network security.

Luca Galbusera is a Scientific/Technical Support Officer at the Joint Research Centre of the European Commission, Ispra, Italy. His research interests include optimal and robust control, networked and multi-agent control systems, and critical infrastructure modeling and analysis.

Andrea Gasparri is an Associate Professor of Engineering at the University of Roma Tre, Rome, Italy. His research interests include robotics, sensor networks and networked multi-agent systems.

Bela Genge is an Associate Professor of Computer Science and a Marie Curie Fellow at Petru Maior University of Tirgu-Mures, Mures, Romania. His research interests include critical infrastructure protection, secure and resilient design of critical control systems and network security.

Georgios Giannopoulos is a Scientific Officer at the Joint Research Centre of the European Commission, Ispra, Italy. His current research focuses on computational tools for analyzing risk, interdependencies and the economic impacts of critical infrastructure disruptions.

Flavius Graur is an M.Sc. student in Information Technology at Petru Maior University of Tirgu-Mures, Mures, Romania. His research interests include computer and network security, penetration testing and software-defined networking.

Dimitris Gritzalis is the Associate Rector for Research, Professor of Information Security and Director of the Information Security and Critical Infrastructure Protection Laboratory at Athens University of Economics and Business, Athens, Greece. His research interests include critical infrastructure protection, social media intelligence and smartphone security and privacy.

Piroska Haller is an Associate Professor of Computer Science at Petru Maior University of Tirgu-Mures, Mures, Romania. Her research interests include industrial control system security and distributed systems.

Uday Kanteti is an M.S. student in Computer Science at the Missouri University of Science and Technology, Rolla, Missouri. His research interests include information assurance, critical infrastructure protection and formal methods.

Wayne King is a Project Leader at Lawrence Livermore National Laboratory, Livermore, California. His research focuses on the physics, material science, engineering and control aspects of additive manufacturing.

Megan Leierzapf is a System Validation Engineer at Intel Corporation, San Jose, California. Her research interests include cyber security and systems engineering.

Htein Lin is an M.S. student in Cyber Operations at the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. His research interests include network security and critical infrastructure protection.

Joshua Lubell is a Computer Scientist in the Systems Integration Division at the National Institute of Standards and Technology, Gaithersburg, Maryland. His research interests include model-based engineering, cyber security, cyber-physical systems, information modeling and markup technologies.

Georgia Lykou is a Ph.D. candidate in Informatics and a Researcher in the Information Security and Critical Infrastructure Protection Laboratory at Athens University of Economics and Business, Athens, Greece. Her research interests include critical infrastructure protection, risk assessment and environmental threats.

Caleb Mays is an M.S. student in Cyber Operations at the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. His research interests include wireless network security and critical infrastructure protection.

Bruce McMillin is a Professor of Computer Science at the Missouri University of Science and Technology, Rolla, Missouri. His research interests include critical infrastructure protection, computer security, formal methods and distributed systems.

Barry Mullins is a Professor of Computer Engineering at the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. His research interests include cyber operations, critical infrastructure protection and computer, network and embedded system security.

Stefano Panzieri is an Associate Professor of Automatic Control and Head of the Models for Critical Infrastructure Protection Laboratory at the University of Roma Tre, Rome, Italy. His research interests include industrial control systems, robotics and sensor fusion.

Antonios Papachrysanthou is an Assistant Researcher in the Information Security and Critical Infrastructure Protection Laboratory at Athens University of Economics and Business, Athens, Greece. His research interests include critical infrastructure protection and information security.

John Pecarina is an Assistant Professor of Computer Science at the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. His research interests include distributed systems, cryptographic protocols, cyber-physical system security and critical infrastructure protection.

Evan Plumley is an M.S. student in Computer Science at the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. His research interests include network security, cyber security training and critical infrastructure protection.

Gregory Pope is a Group Leader at Lawrence Livermore National Laboratory, Livermore, California. His research interests include vulnerability analyses of additive manufacturing and Internet of Things software control systems.

Benjamin Ramsey is a Cyberspace Operations Officer in the U.S. Air Force, Washington, DC. His research interests include wireless network security and critical infrastructure protection.

Mason Rice is an Assistant Professor of Computer Science at the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. His research interests include network and telecommunications security, cyber-physical system security and critical infrastructure protection.

Anthony Rose is an M.S. student in Electrical Engineering at the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. His research interests include wireless network security and Internet of Things security.

Julian Rrushi is an Assistant Professor of Computer Science at Western Washington University, Bellingham, Washington. His research interests include industrial control system security and defensive cyber deception.

Hunor Sandor is a Ph.D. student in Computer Science at the Technical University of Cluj-Napoca, Cluj-Napoca, Romania; and a Researcher in the Department of Computer Science at Petru Maior University of Tirgu-Mures, Mures, Romania. His research interests include cyber security and cyber-physical security, and designing response and reconfiguration techniques to mitigate threats and vulnerabilities in large-scale systems.

George Stergiopoulos is a Senior Researcher and Postdoctoral Fellow in the Information Security and Critical Infrastructure Protection Laboratory at Athens University of Economics and Business, Athens, Greece. His research interests include critical infrastructure protection, applications security and cryptography.

Eniye Tebekaemi is a Ph.D. candidate in Information Technology and a Researcher in the Radio and Radar Engineering Laboratory at George Mason University, Fairfax, Virginia. His research interests include cyber security, cyber-physical systems and intrusion detection systems.

Duminda Wijesekera is a Professor of Computer Science at George Mason University, Fairfax, Virginia; and a Visiting Research Scientist at the National Institute of Standards and Technology, Gaithersburg, Maryland. His research interests include cyber security, digital forensics and transportation systems.

Mark Yampolskiy is an Assistant Professor of Computer Science at the University of South Alabama, Mobile, Alabama. His research focuses on the security aspects of additive manufacturing, cyber-physical systems and the Internet of Things.

Timothy Zimmerman is a Computer Engineer in the Intelligent Systems Division at the National Institute of Standards and Technology, Gaithersburg, Maryland. His research focuses on cyber security in the manufacturing sector with an emphasis on industrial control systems and robotics.

Preface

The information infrastructure – comprising computers, embedded devices, networks and software systems – is vital to operations in every sector: chemicals, commercial facilities, communications, critical manufacturing, dams, defense industrial base, emergency services, energy, financial services, food and agriculture, government facilities, healthcare and public health, information technology, nuclear reactors, materials and waste, transportation systems, and water and wastewater systems. Global business and industry, governments, indeed society itself, cannot function if major components of the critical information infrastructure are degraded, disabled or destroyed.

This book, *Critical Infrastructure Protection XI*, is the eleventh volume in the annual series produced by IFIP Working Group 11.10 on Critical Infrastructure Protection, an active international community of scientists, engineers, practitioners and policy makers dedicated to advancing research, development and implementation efforts related to critical infrastructure protection. The book presents original research results and innovative applications in the area of infrastructure protection. Also, it highlights the importance of weaving science, technology and policy in crafting sophisticated, yet practical, solutions that will help secure information, computer and network assets in the various critical infrastructure sectors.

This volume contains sixteen revised and edited papers from the Eleventh Annual IFIP Working Group 11.10 International Conference on Critical Infrastructure Protection, held at SRI International in Arlington, Virginia, USA on March 13–15, 2017. The papers were refereed by members of IFIP Working Group 11.10 and other internationally-recognized experts in critical infrastructure protection. The post-conference manuscripts submitted by the authors were rewritten to accommodate the suggestions provided by the conference attendees. They were subsequently revised by the editors to produce the final chapters published in this volume.

The chapters are organized into four sections: (i) infrastructure protection; (ii) infrastructure modeling and simulation; (iii) industrial control system security; and (iv) Internet of Things security. The coverage of topics showcases the richness and vitality of the discipline, and offers promising avenues for future research in critical infrastructure protection.

This book is the result of the combined efforts of several individuals and organizations. In particular, we thank David Balenson, Molly Keane and Zachary Tudor for their tireless work on behalf of IFIP Working Group 11.10. We gratefully acknowledge Idaho National Laboratory and the Institute for Information Infrastructure Protection (I3P), managed by George Washington University, for their sponsorship of IFIP Working Group 11.10. We also thank the U.S. Department of Homeland Security, National Security Agency and SRI International for their support of IFIP Working Group 11.10 and its activities. Finally, we wish to note that all opinions, findings, conclusions and recommendations in the chapters of this book are those of the authors and do not necessarily reflect the views of their employers or funding agencies.

MASON RICE AND SUJEET SHENOI

I

INFRASTRUCTURE PROTECTION

Chapter 1

PROTECTING THE TRANSPORTATION SECTOR FROM THE NEGATIVE IMPACTS OF CLIMATE CHANGE

Georgia Lykou, George Stergiopoulos, Antonios Papachrysanthou and Dimitris Gritzalis

Abstract Observed and projected climate changes, such as temperature increase, sea-level rise and increases in the frequency and intensity of extreme weather events, are posing challenges to critical infrastructure operations, especially in the transportation sector, which is a pillar of economy and society. Like the other critical infrastructure sectors, the transportation sector comprises complex systems with responsibilities distributed across many stakeholders. The challenges to implementing integrated climate change adaptation approaches in the transportation sector require appropriate governance and coordinated action.

Adaptation to climate change demands resilient and sustainable infrastructures. However, despite the importance of the transportation sector and the huge challenges posed by climate change, adaptation to climate change as a means to reduce risk in the sector is relatively low. Adaptation actions require climate vulnerability analyses and impact knowledge that would help ensure that adaptation options are properly identified, evaluated and monitored. This research attempts to identify and analyze global adaptation initiatives in order to classify adaptation options while also focusing on emerging adaptation challenges and opportunities in the transportation sector. This research should assist the various stakeholders in improving the effectiveness and future sustainability of transportation while stimulating actions for adapting to climate change.

Keywords: Climate change, transportation sector, adaptation options

1. Introduction

Modeling studies indicate that global average temperatures will increase more than two degrees Celsius over the next century [3]. The United Nations

Intergovernmental Panel on Climate Change (IPCC) [1] has provided evidence that these changes will have significant implications on extreme weather events, economic development and stability, population and environmental health. Climate change concerns with regard to the critical infrastructure focus on climate variations and extreme weather events that will increase in magnitude, frequency and/or duration [3]. Critical infrastructure assets are typically designed to withstand weather-related stressors common in specific localities, but shifts in climate patterns increase the ranges and types of potential risks. Most infrastructure assets that are being constructed today are expected to last for decades or even centuries [13]. Investing in infrastructures that are not designed to cope with climate change will almost certainly result in significant increases in future costs and amplify the potential for unplanned outages and failures.

Transportation is a critical infrastructure that supports the smooth functioning of society and the prosperity and viability of local, regional and national economies [4]. It facilitates access to services that are vital to commerce and the quality of human life. Gradual climate changes such as increases in temperature, sea level and rainfall along with the projected growth in the frequency and intensity of extreme weather events will seriously challenge the transportation sector. While mitigation efforts are of great importance, critical infrastructures such as transportation must be adapted to reduce the emission of anthropogenic gases and their contributions to climate change.

This research examines a number of climate change adaptation approaches, strategies and action plans with the goal of understanding adaptation measures that are applicable to the transportation sector. Also, it provides a detailed classification and analysis of each adaptation measure according to established methodologies. It focuses on the impacts on the transportation sector and examines national and sectoral adaptation plans for best practices and efficient adaptation options.

2. Transportation Sector

Transportation involves the movement of people and goods from one location to another [6]. The transportation sector comprises: (i) transportation infrastructure – fixed installations such as roads, railroads, bridges, canals, pipelines and terminals (e.g., airports, railroad stations, bus stations and seaports); (ii) vehicles – cars, buses, trucks, railroad cars and locomotives, ships, aircraft, drones, etc.; and (iii) operations – people, institutions, laws, policies and information systems that transform infrastructure and vehicles into working transportation networks [2]. Modes of transportation include road, rail, pipelines, water, air and space.

Transportation activities are the result of bringing together diverse resources. Service providers put together these resources to provide different modes of transportation that collectively offer a variety of transportation services to customers. Regulators at various administrative levels provide the basic rules to enable transportation operations to run smoothly, efficiently and with minimal impacts [6]. Finally, the numerous users of the transportation modes make their choices and shape the demand for transportation.

Disruptions to transportation systems can cause large economic and human losses. For this reason, the transportation sector is characterized as a critical infrastructure – an important pillar of economy and society [9]. Since most stakeholders may only have a partial perspective of the transportation systems they manage or use [4], without a national protection strategy, the stakeholders would react autonomously and in an uncoordinated manner to address the challenges imposed by climate change. Given the broad impacts of climate variations and the strong interconnectivity within the transportation sector, such a fragmented approach would lead to great inefficiencies and negative impacts on the sustainability and resilience of the transportation infrastructure.

2.1 Climate Change Impacts on Transportation

Rising temperatures and extended periods of heatwaves increase rail buckling, road pavement deterioration and thermal discomfort for vehicular passengers [9]. Weather extremes cause floods and landslides that result in short-term delays and interruptions in multiple transportation modes as well as long-term interruptions and detours when land-side infrastructures are damaged or destroyed. Sea-level rise threatens harbors and other transportation infrastructure assets and services in coastal areas. Air transport is impacted by changing wind patterns, floods and other extreme weather events. Additionally, climate impacts can trigger changes to society (e.g., different tourist destinations) and the economy (e.g., reduced agricultural production). Tables 1 and 2 present the climatic events and risks of climate change for various transportation modes based on a literature analysis [4, 9, 16].

The effects of malfunctions, disturbances and broken transportation links may stretch far beyond the originally-affected areas [17]. The transportation system has a transboundary character and is highly interconnected within and across the various transportation modes; hence, disturbances in one portion of a transportation network could have domino effects in other parts of the network [13]. The effects typically involve service disruptions that prevent the movement of passengers and goods. The interdependencies could result in losses that are many times higher than the direct costs to the transportation sector itself [4, 12].

The European Joint Research Centre [10] has presented a comprehensive quantitative assessment of the impacts of current and future climate extremes on critical infrastructures in Europe. The dynamics of climate hazards were analyzed throughout the 21st century using physical models and adaptation tools. The assessment reveals that damage in Europe as a result of climate extremes could triple by the 2020s and amount to more than ten times the current damage of 3.4 billion euros per year by 2100. According to the study, heat waves will cause the most damage, primarily to roads and railroad tracks. These transportation modes will also suffer losses from coastal flooding, which will increase drastically over time. Inland waterway transport will be impacted

Table 1. Climate change events and risks on transportation (Part 1).

Type	Climatic Events	Risks
Land Transport – Rail	Summer heat	Rail buckling, material fatigue, increased instability of embankments, overheating of equipment, increase in wildfires and infrastructure damage
	Winter cold and ice	Ice on trains and catenaries, damage to infrastructure due to low temperatures
	Extreme precipitation	Damage to infrastructure due to floods and landslides, scouring of structures, destabilization of embankments
	Extreme storms	Damage to infrastructure such as signals and power cables (e.g., due to falling trees)
	In general:	Reduced safety, increased repair and maintenance costs, disruption of just-in-time delivery of goods and passengers
Land Transport – Roads	Summer heat	Pavement deterioration, melted tarmacs, reduced life of asphalt road surfaces, increase in wildfires and infrastructure damage, expansion and buckling of bridges
	Extreme precipitation and floods	Damage to infrastructure, road submersion, underpass flooding, over-strained drainage systems, landslides, destabilization of embankments
	Extreme storms	Damage to infrastructure, fallen trees block roads
	In general:	Reduced speed, road closures and road safety hazards, disruption of goods delivery, welfare losses, higher repair and maintenance costs

by droughts while sea-level rise and increased storm surges will significantly increase the damage to ports. These projections create an urgent need to develop and implement an adaptation approach with a long-term, systemic perspective.

3. Approaches for Climate Change Adaptation

Adaptation involves actions that respond to current and future climate change impacts and vulnerabilities [8]. This involves protecting against the negative impacts of climate change as well as building resilience and leveraging any benefits it may provide. This section discusses adaptation assessment, adaptation options and classification, and global adaptation initiatives.

Table 2. Climate change events and risks on transportation (Part 2).

Type	Climatic Events	Risks
Air Transport – Airports	Summer heat	Greater need for ground cooling, degradation of runways and foundations, reduced engine efficiency due to higher air density at high altitudes, decreased airport lift and increased runway lengths
	Extreme precipitation	Flood damage to runways and other infrastructures, water run-off exceeds capacity of drainage systems
	Sea-level rise	Flooding of runways, outbuildings and access roads
	In general:	Interruption and disruption of services and ground access, delays and passenger loss of confidence, periodic airport closures and higher maintenance costs
Maritime Transport	Changes in sea conditions	More severe storms and extreme waves, restrictions on loading capacities and routes, navigation problems
	Sea-level changes	Navigation affected by changes in sedimentation rates, inland shipping disruptions
	Fewer days below freezing	Reduced problems due to ice accumulation on vessels, decks, riggings and docks, occurrence of dangerous ice fog
	Reduced sea ice	Improved access, longer shipping seasons, new routes
	In general:	Disruption of just-in-time delivery of goods, inland shipping limitations, welfare losses
Ports	Extreme storms, sea-level rise, floods, landslides	Devastation of infrastructure, increased maintenance and restoration costs, flow bottlenecks and interruptions, welfare losses

3.1 Adaptation Assessment

Adaptation assessment is the practice of identifying options to adjust to climate change and evaluating them based on criteria such as availability, benefits, costs, effectiveness, efficiency and feasibility [3]. Two main types of analyses are used to assess potential adaptation: (i) identification of adaptation options; and (ii) evaluation of adaptation options [14]:

- **Identification of Adaptation Options:** Adaptation options are intended to address climate vulnerabilities and build resilience by reducing the negative impacts to acceptable levels. Adaptation options may also leverage positive opportunities that arise from climate change [19]. Adaptation options range from actions that build adaptive capacity (e.g., sharing information and creating supportive institutional frameworks) to concrete adaptation measures (e.g., technical solutions, warning systems and insurance mechanisms). Limits to adaptation options are imposed by the specific times when the actions can be implemented and the geographical locations where the actions may be executed. Additionally, there exist inherent limits and uncertainty about the extent to which the actions can enhance the adaptive capacity and protect regions, economic sectors and communities. It is also challenging to decide on the most appropriate protection levels to implement based on the knowledge of current and projected climate change impacts and damage costs.

- **Evaluation of Adaptation Options:** After a set of adaptation options has been identified, the next step is to evaluate the options in order to guide decisions regarding their selection and implementation. World Resources Institute and World Bank guidance documents [21] list several criteria for evaluating the suitability of an adaptation option in contributing to an objective: (i) cost analysis, including total costs and cost effectiveness; (ii) environmental implications; (iii) secondary or cross-sectoral impacts, externalities or co-benefits; (iv) social implications, including implications for sensitive groups; (v) short-, medium- and long-term efficacy; (vi) effectiveness at reducing the impacts of extreme events; (vii) effectiveness under different climate scenarios; (viii) limiting factors for implementation and sustainability (e.g., resource constraints); (ix) consultation with a broad set of stakeholders; (x) provision for reviewing options based on changing assessments of risk; and (xi) transparency in the process and justification of option selection.

3.2 Classification of Adaptation Options

The European Environment Agency [8] classifies adaptation actions into three broad categories:

- **Green Actions:** These actions involve ecosystem-based approaches that leverage nature-provided services to achieve cost effective and possibly more feasible adaptation solutions. Examples include green earthworks to combat landslides, renewable energy sources and low energy demand infrastructures.

- **Soft Actions:** These actions involve managerial, legal and policy approaches that alter human behavior and governance styles such as new policies and procedures, land-use controls, information dissemination and economic incentives that reduce or prevent disaster vulnerability. Examples include legislation, standards and best practices, and public information campaigns.

- **Gray Actions:** These actions involve various technological and engineering solutions for infrastructure, corresponding to physical interventions, construction measures and using engineering services to enable infrastructures to withstand extreme events. Examples include coastal and river flood defenses, early warning systems, redundant equipment and networks.

Green and soft actions specifically focus on decreasing the sensitivity and increasing the adaptive capacity of human and natural systems to foster resilience; these actions are often less resource intensive and provide multiple benefits. Gray actions involve innovative technological solutions that typically cost more and require more research, experience and training to be implemented.

Adaptation has an extremely important role in reducing the economic costs of climate change. While adaptation has a cost, it significantly reduces the financial consequences of inaction and, in many cases, provides benefits that dramatically outweigh the costs [9]. Since it is important to enable cost-effective and proportionate adaptation to climate change, options that can achieve adaptation while minimizing the associated risks and uncertainties are especially appropriate.

In addressing adaptation risk and uncertainty, the United Kingdom Climate Impacts Programme (UKCIP) [19], categorizes options as no regrets, low regrets, win-win and adaptive/flexible management options that target incremental adaptation:

- **No Regret Adaptation Options:** These adaptation actions are worthwhile regardless of the extent of future climate change. They include justified and cost-effective measures under current climate conditions that are further justified when their introduction is consistent with addressing risks associated with projected climate changes.

- **Low Regret Adaptation Options:** These adaptation actions have relatively low costs and their benefits, although primarily realized under projected climate changes, may be relatively large.

- **Win-Win Adaptation Options:** These adaptation actions minimize climate risks and exploit potential opportunities; additionally, they support mitigation and other social and environmental objectives. Some of these actions may have been introduced for reasons other than addressing climate risks, but they also deliver the desired adaptation benefits.

- **Adaptive Management Options:** These adaptation actions are incremental and flexible, and do not effect large-scale adaptation. The actions are based on assessments of the current environment, but are designed for incremental change (including changing the direction) as knowledge, experience and technology evolve.

In contrast, maladaptation options hinder or reduce climate change adaptation and must be avoided to the extent possible:

- **Maladaptation Options:** These maladaptation actions: (i) do not increase resilience and adaptive capacity or do not reduce vulnerabilities; (ii) are not sustainable from an environmental, economic or social perspective (e.g., over-exploitation of water resources); or (iii) conflict with other long-term policy objectives.

Maladaptation can be prevented by considering the climatic and socioeconomic elements that constitute vulnerabilities to climate change.

3.3 Global Adaptation Initiatives

A variety of adaptation initiatives are underway around the world. The U.S. Department of Homeland Security has developed a climate change adaptation roadmap and climate action plan that align with the President's climate action plan, preparing the United States for the impacts of climate change [2]. As of 2016, fifteen states have published climate adaptation plans.

The European strategy for adaptation to climate change sets out a framework and mechanisms to prepare for current and future impacts [7]. The strategy encourages and supports actions by European Union member states and creates a basis for informed decision-making on adaptation in the years to come. The majority of the member states have adopted national adaptation plans and strategies that outline their implemented and planned actions to facilitate the adaptation of transportation and other sectors to climate change.

The Australian Government has developed the ACT Climate Change Adaptation Strategy that coordinates efforts focused on adapting to climate change. This strategy identifies key adaptation policies to enable communities to become more resilient by communicating the risks and impacts of climate change and incorporating climate change risk considerations and adaptation actions in government policies.

4. Adaptation in the Transportation Sector

This section, which constitutes the core of the chapter, identifies and analyzes climate change adaptation actions in the transportation sector. The actions are drawn from national adaptation strategies and relevant publications such as U.S. Government reports, European Union directives and publicly-available adaptation action plans [2, 5, 9, 15, 20].

This section also introduces and categorizes the adaptation options for the transportation sector using the classification categories discussed above. Seven types of options are analyzed, each dealing with a different aspect of adaptation: (i) effective governance; (ii) infrastructure planning; (iii) redundancies within and between transportation modes; (iv) operational contingencies; (v) early warning systems; (vi) building adaptive capacity; and (vii) collaboration. In the following, each adaptation option is presented along with a table that summarizes the classification of the adaptation options. The presentation and classification of adaptation actions are intended to stimulate further research and discussion by transportation sector stakeholders.

4.1 Adaptation to Climate Change

The transportation sector routinely deals with extreme events that cause disruptions, whether they stem from natural hazards or human-initiated events such as accidents and power outages, and has developed resilience strategies. Therefore, the adaptation of transportation systems to climate change requires a wide perspective that embeds adaptation in broader transition strategies [9] instead of leaving it to be implemented by individual stakeholders such as infrastructure owners/operators or regulatory authorities.

Transportation is specifically addressed in most of the national strategies and plans studied in this research. The national strategies and plans primarily focus on the transportation infrastructure and aspects of transportation services, such as the development of alternative routes and means of transportation, traffic management, reviews of technical conditions of vehicles and vehicular operations, and support to transportation infrastructure owners/operators in developing their adaptation assessments and actions. The literature (see, e.g., [20]) reveals that stakeholders such as railroad companies [15], airports and port authorities, and air traffic control operators [5] are well aware of the potential climate change impacts and the need to adapt, and have started taking action. The prospects of significant reconstruction costs, lengthy recovery processes and severe disruptions have influenced transportation infrastructure owners/operators to undertake comprehensive assessments of their assets.

4.2 Effective Governance for Adaptation

The primary role of government is to enable adaptation actions at the local and regional levels by creating an appropriate framework [6, 21]. This includes effective institutions, knowledge, supportive policy, laws and regulations, and funding. As such, transportation should also be a part of national adaptation strategies and action plans.

Since stakeholders acting at the local, regional or company levels implement actions such as climate-proofing infrastructure and operations, it is vital that authorities create synergies and engage all the stakeholders in the transportation sector. Enhancing legislation with national standards for earth and public works and requiring climate risk assessments as a prerequisite for new asset construction can ensure the integrity and future protection of the transportation infrastructure.

Public funding for new or existing infrastructure reinforcement should incorporate adaptation assessments to ensure infrastructure sustainability. Table 3 presents the effective governance actions categorized according to adaptation option type. Note that most of actions proposed are soft measures, except for funding issues. This is not surprising because the roles of government are mostly managerial and policy setting, which create the appropriate framework for executing adaptation actions at the local, regional and national levels.

Table 3. Effective governance actions proposed in adaptation plans.

Adaptation Action	**Actions**			**Risk Aspect**			
	Green	**Soft**	**Gray**	**No Regrets**	**Low Regrets**	**Win-Win**	**Adaptive**
Develop strategic plan for sustainable transportation		X		X			
Develop national adaptation strategy and action plan		X				X	
Create an adaptation framework that engages stakeholders in the transportation sector		X			X		
Incorporate adaptation requirements in legislation and regulatory norms		X			X		
Enhance standards and national/regional requirements		X				X	
Require climate risk and environmental assessments for all new infrastructure		X				X	
Ensure funding for new infrastructure and existing infrastructure reinforcement			X				X
Coordinate future infrastructure plans		X				X	

4.3 Infrastructure Design and Planning

Smooth and effective operations of transportation systems rely heavily on infrastructures that are intended to last for many decades, possibly more than a century. Investments in the transportation infrastructure are usually costly. Therefore, an anticipatory approach is necessary for planning new infrastructure assets, and for existing infrastructure renovation, improvements and maintenance. Considering future climate trends now will help keep adaptation costs at manageable levels and avoid future unsustainable development paths for transportation systems.

Climate change adaptation must be considered at all relevant levels from network planning to project management. Concrete methodological guidance must be provided on how the integration can be implemented effectively [9]. Certain soft actions require relatively low investments. However, further mainstreaming of adaptation into transportation infrastructure investments can have substantial implications for infrastructure resilience and adaptation costs over the long term. In general, adaptation integrated in new and upgraded infrastructures comes at a lower cost than future integration.

Green actions include earthwork projects that combat landslides and green transportation networks that enhance energy efficiency, engage renewable en-

Table 4. Infrastructure design and planning actions proposed in adaptation plans.

Adaptation Action	**Actions**			**Risk Aspect**			
	Green	**Soft**	**Gray**	**No Regrets**	**Low Regrets**	**Win-Win**	**Adaptive**
Revise obsolete designs and infrastructure standards		X		X			
Create new standards and recommended practices for infrastructure resilience		X				X	
Improve green design earthworks to combat landslides, subsidence, heaves and wind damage	X					X	
Perform regular civil engineering checks of infrastructure foundations and actions to combat erosion			X				X
Review piping and networks to identify vulnerabilities			X		X		
Strengthen drainage elements and improve storm drain capacity			X				X
Proactively inspect and maintain guidance for infrastructure assets			X			X	
Use design limits to explore if heating, cooling, insulation or drying measures are required			X				X
Design green transportation networks by improving energy efficiency, use renewable energy sources and reduce the carbon footprint of transportation	X					X	

ergy sources and reduce carbon footprints. Table 4 presents the infrastructure design and planning actions categorized according to adaptation option type. The majority of the actions are classified as win-win – they reduce climate risk while providing other benefits and incremental actions for adaptive management.

4.4 Redundancies in Transportation Modes

Designing, building and using redundant infrastructure such as alternative roadways and rail links can enhance the resilience of transportation operations. Procuring and maintaining ready-to-use backup equipment and vehicles for emergencies and adding backup power generation capacity for critical facilities are important redundancy measures. Such strategies involve extra costs to establish and maintain redundancy that may not be needed during normal conditions. Nevertheless, redundancy strategies will become more important and

Table 5. Redundancy planning actions proposed in adaptation plans.

Adaptation Action	**Actions**			**Risk Aspect**			
	Green	**Soft**	**Gray**	**No Regrets**	**Low Regrets**	**Win-Win**	**Adaptive**
Design and build redundant infrastructure in areas vulnerable to climate change			X				X
Design and construct resilient vehicles for all transportation modes			X			X	
Explore multi-modal opportunities (e.g., multi-modal stations and flexible ticketing options)			X			X	
Build and maintain ready-to-use backup equipment and vehicles for emergencies			X				X
Add backup power generation capacity for critical facilities			X				X

more common as the number of extreme weather events and their magnitudes increase as a result of climate change.

Multi-modal transportation offers redundancies at multiple levels [9]. If different transportation modes are available, passengers would be able to select the best modes that meet their needs and switch from one mode to another as necessary. Smart and flexible ticketing enables passengers to switch operators and modes in the event of disruptions. Building and maintaining ready-to-use backup equipment and vehicles for emergencies and adding backup power generation capacity for critical facilities also support adaptation efficiency.

Table 5 presents the redundancy planning actions categorized according to adaptation option type. As expected, redundancy actions correspond to gray measures because they require more funding and additional resources. The actions are categorized as win-win because redundancy options offer social benefits to transportation users and enhance adaptive management.

4.5 Operational Contingency

The transportation sector has traditionally implemented approaches to cope with the impacts of extreme weather events; these approaches also serve as valuable options for adapting to climate change. Preparation for risk situations is accomplished via contingency planning, business continuity planning and disaster recovery planning for extreme weather events. Emergency reporting, emergency equipment preparedness, surveillance and maintenance plans can enhance the integrity of critical facilities during adverse events. It is also important to position transportation assets away from vulnerable areas and

Table 6. Operational contingency actions proposed in adaptation plans.

Adaptation Action	**Actions**			**Risk Aspect**			
	Green	**Soft**	**Gray**	**No Regrets**	**Low Regrets**	**Win-Win**	**Adaptive**
Establish preparedness and prevention plans		X		X			
Establish business continuity and disaster recovery plans		X		X			
Establish emergency reporting and emergency equipment preparedness activities			X		X		
Position records, materials and assets away from vulnerable areas		X			X		
Provide staff with personal protective equipment for use in weather events			X			X	
Relocate critical movable assets before events to reduce damage and impact			X		X		
Establish surveillance and maintenance plans		X				X	
Establish insurance policies for infrastructure assets in vulnerable areas			X		X		

provide staff with personal protective equipment for use in operations during extreme events. Insurance schemes are also important because they can support key infrastructure funding and restoration in vulnerable areas.

Table 6 presents the operational contingency actions categorized according to adaptation option type. The actions are classified as either gray or soft. Green actions are missing because it is difficult to find green solutions that enhance operational contingencies for transportation assets. The listed options range from cost-effective measures to proportionate adaptive measures.

4.6 Early Warning Systems

Early warning systems enable transportation personnel to prepare for extreme weather events induced by climate change or climate variability. For example, Eurocontrol [5] has developed a natural hazards and weather resilience tool that provides information about the potential vulnerabilities of airports and en route sectors in Europe. Warning systems should leverage information and communications technologies to enhance transportation management. These include sensors and other devices that provide real-time information on traffic conditions such as temperature, vehicle speed, obstacles, deformations and other surface characteristics [11]. Information and communications tech-

Table 7. Early warning systems proposed in adaptation plans.

Adaptation Action	**Actions**			**Risk Aspect**			
	Green	**Soft**	**Gray**	**No Regrets**	**Low Regrets**	**Win-Win**	**Adaptive**
Fixed warning systems with GPS technology, meteorological instruments and other sensors that detect extreme weather events			X			X	
Vehicle sensors and devices that transmit real-time information			X			X	
User devices that transmit real-time information		X		X			
Weather warning and incident warning networks			X			X	
Warnings and awareness efforts for staff about the increased risks during inclement weather events		X				X	

nologies enable this information to be accessed in real time by transportation infrastructure managers, service operators and users.

Vehicles and users are increasingly serving as data collectors, enabling infrastructure managers and operators to gain unprecedented real-time information about transportation systems. Handling these enormous flows of information requires advanced technologies that link vehicles to other vehicles and infrastructure, and process and disseminate the information. Transportation system operators and users in their vehicles can receive vital information about infrastructure conditions, infrastructure managers can obtain detailed descriptions of traffic conditions from users and disseminate the information. This greatly facilitates traffic management and enables passengers to adapt their plans and find alternative transportation options. It also improves the quality of transportation services and provides benefits to all the stakeholders, especially users and operators.

Table 7 presents the early warning systems categorized according to adaptation option type. Note that the early warning systems primarily correspond to gray and soft actions that require technological innovation and engineering support. They are also classified as win-win actions because they offer social and other benefits to stakeholders.

4.7 Building Adaptive Capacity

Global initiatives and transnational and national adaptation platforms are collecting relevant information pertaining to all stages of the policy process and making it easily accessible. For example, the European Climate Adaptation Platform (`climate-adapt.eea.europa.eu`) enables stakeholders to access and

Table 8. Building adaptive capacity actions proposed in adaptation plans.

Adaptation Action	**Actions**			**Risk Aspect**			
	Green	**Soft**	**Gray**	**No Regrets**	**Low Regrets**	**Win-Win**	**Adaptive**
Create a public adaptation platform		X		X			
Share information about adaptation best practices		X		X			
Benchmark implementations of adaptation actions		X	X				
Document and share institutional knowledge		X		X			
Build datasets to understand territorial and sectoral vulnerabilities to climate change		X		X			
Increase awareness, education and training on climate change vulnerabilities and impacts		X				X	

share data and information on climate change, current and future vulnerabilities of regions and sectors, national and transnational adaptation strategies and actions, adaptation case studies and tools that enhance adaptation performance and planning.

Information collected on past weather events and their impacts is a valuable starting point for assessing vulnerabilities and developing strategies to adapt to climate change. Sharing knowledge about adaptation best practices and benchmarking implementation case studies help accelerate new initiatives. Additionally, education, training and public awareness efforts on climate change vulnerabilities and impacts also improve adaptation performance.

Table 8 presents the building adaptive capacity actions categorized according to adaptation option type. The (mainly) soft actions increase transportation resilience to climate change and are often cost-effective. However, this research has revealed that public platforms primarily provide information of a general nature; detailed technical and event data are scarce. These platforms should be upgraded and expanded to systematically collect and disseminate data about transportation disruption events at the national level.

4.8 Comprehensive Collaboration

Climate experts should make the various transportation stakeholders aware of the fact that climate-related topics cannot be addressed through traditional, deterministic methods and that alternative approaches for managing the risks brought upon by climate change should be explored. Transportation experts should collaborate to specify their climate forecasting needs in scientific terms,

Table 9. Collaboration actions in adaptation plans.

Adaptation Action	**Actions**			**Risk Aspect**			
	Green	**Soft**	**Gray**	**No Regrets**	**Low Regrets**	**Win-Win**	**Adaptive**
Communicate plans and information to stakeholders and the public		X				X	
Cooperate with stakeholders to expand the sharing of knowledge and best practices		X				X	
Interact with experts and scientists to expand research on adaptation issues		X				X	
Cooperate with experts in other fields to increase knowledge about climate science and adaptation		X				X	

enabling meteorologists to better understand their needs and implement the appropriate solutions. Cooperative efforts by experts from diverse fields would enable transportation stakeholders to increase their management and decision-making flexibility.

Table 9 presents the collaboration actions categorized according to adaptation option type. The collaboration actions are primarily of the soft and win-win types. Cooperation among experts from different domains is a fruitful means to further adaptation efficiency and transportation system resilience.

5. Conclusions

Transportation systems are complex; they play a fundamental role in the economy and society and are characterized by long lifespans and high costs. These characteristics suggest the need for an adaptation approach with a long-term, systemic perspective that also prevents unsustainable development paths and maladaptation.

Several countries have begun to develop adaptation strategies and action plans that focus on the implementation of climate change adaptation measures in the critical infrastructure sectors, including transportation. The actions include information dissemination, capacity building, reviews of technical standards and the incorporation of new information and communications technologies. The engagement of all the principal stakeholders in the transportation sector is very important from the perspective of equity and efficiency. Therefore, regulatory authorities, policymakers and researchers should make strong efforts to engage transportation sector stakeholders in their research and information dissemination activities.

Most climate change adaptation measures for the transportation infrastructure are soft actions (60%). Gray actions are also quite popular (35%) in existing adaptation plans. However, only 5% are green measures – this is a concern because the transportation sector is environmentally-invasive and consumes significant natural resources. It is imperative that climate change adaptation measures for the transportation infrastructure focus on improving energy efficiency and leveraging renewable energy. In general, low regret and win-win actions increase the resilience of transportation systems while providing advantages such as smooth operation, quality of service and efficiency.

Tools and measures for managing risks and impacts, such as early warning systems and contingency plans, can be used to enhance the resilience of the transportation infrastructure to climate change. Adaptation actions that focus on climate-proofing the transportation infrastructure should integrate adaptation requirements into the design of new and upgraded infrastructure – now as opposed to the future — in order to reduce costs over the long term. Another key adaptation action is to provide functionally-redundant options that build capacity and enable flexibility in the event of disasters and other interruptions.

Adaptation actions in the transportation sector should be continually monitored and analyzed. This would enable the principal stakeholders to improve the effectiveness and efficiency of climate change adaptation policy and its implementation. Stakeholders within and outside the transportation sector should also collaborate to ensure that the knowledge gained in other critical infrastructure sectors is leveraged to develop and implement innovative and effective solutions for adapting the transportation infrastructure to cope with the highly disruptive impacts of future climate change.

References

[1] S. Allen, V. Barros, I. Burton, D. Campbell-Lendrum, O. Cardona, S. Cutter, O. Dube, K. Ebi, C. Field, J. Handmer, P. Lal, A. Lavell, K. Mach, M. Mastrandrea, G. McBean, R. Mechler, T. Mitchell, N. Nicholls, K. O'Brien, T. Oki, M. Oppenheimer, M. Pelling, G. Plattner, R. Pulwarty, S. Seneviratne, T. Stocker, M. van Aalst, C. Vera and T. Wilbanks, Summary for policymakers, in *Managing the Risks of Extreme Events and Disasters to Advance Climate Change Adaptation, Special Report of the Intergovernmental Panel on Climate Change*, C. Field, V. Barros, T. Stocker, Q. Dahe, D. Dokken, K. Ebi, M. Mastrandrea, K. Mach, G. Plattner, S. Allen, M. Tignor and P. Midgley (Eds.), Cambridge University Press, Cambridge, United Kingdom, pp. 3–21, 2012.

[2] R. Bierbaum, A. Lee, J. Smith, M. Blair, L. Carter, F. Chapin, P. Fleming, S. Ruffo, S. McNeeley, M. Stults, L. Verduzco and E. Seyller, Adaptation, in *Climate Change Impacts in the United States: The Third National Climate Assessment*, J. Melillo, T. Richmond and G. Yohe (Eds.), U.S. Government Printing Office, Washington, DC, pp. 670–706, 2014.

[3] V. Burkett, A. Suarez, M. Bindi, C. Conde, R. Mukerji, M. Prather, A. St. Clair and G. Yohe, 2014: Point of departure, in *Climate Change 2014: Impacts, Adaptation and Vulnerability, Part A: Global and Sectoral Aspects, Contribution of Working Group II to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*, C. Field, V. Barros, D. Dokken, K. Mach, M. Mastrandrea, T. Bilir, M. Chatterjee, K. Ebi, Y. Estrada, R. Genova, B. Girma, E. Kissel, A. Levy, S. MacCracken, P. Mastrandrea and L. White (Eds.), Cambridge University Press, Cambridge, United Kingdom, pp. 169–194, 2014.

[4] Committee on Climate Change and U.S. Transportation, Potential Impacts of Climate Change on U.S. Transportation, Transportation Research Board Special Report 290, Transportation Research Board, Washington, DC, 2008.

[5] Eurocontrol, Challenges of Growth 2013, Task 8: Climate Change Risk and Resilience, Brussels, Belgium 2013.

[6] European Commission, Roadmap to a Single European Transport Area – Towards a Competitive and Resource-Efficient Transport System, Directorate-General for Mobility and Transport, COM(2011) 144 Final, Brussels, Belgium, 2011.

[7] European Commission, An EU Strategy on Adaptation to Climate Change, Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions, COM(2013) 216 Final, Brussels, Belgium, 2013.

[8] European Environment Agency, Adaptation in Europe – Addressing Risks and Opportunities from Climate Change in the Context of Socio-Economic Developments, EEA Report No. 3/2013, Copenhagen, Denmark, 2013.

[9] European Environment Agency, Adaptation of Transport to Climate Change in Europe – Challenges and Options Across Transport Modes and Stakeholders, EEA Report No. 8/2014, Copenhagen, Denmark, 2014.

[10] G. Forzieri, A. Bianchi, M. Marin Herrera, F. Batista e Silva, L. Feyen and C. Lavalle (Eds.), Resilience of Large Investments and Critical Infrastructures in Europe to Climate Change, Final Report for DG CLIMA, JRC Technical Report AA 071303/2012/630715//CLIMA.C.3 – JRC 32971-2012 NFP, European Commission, Brussels, Belgium, 2015.

[11] S. Grant-Muller and M. Usher, Intelligent transport systems: The propensity for environmental and economic benefits, *Technological Forecasting and Social Change*, vol. 82, pp. 149–166, 2014.

[12] P. Kotzanikolaou, M. Theocharidou and D. Gritzalis, Accessing n-order dependencies between critical infrastructures, *International Journal of Critical Infrastructures*, vol. 9(1-2), pp. 93–110, 2013.

[13] C. McLean, Y. Lee, S. Jain and C. Hutchings, Modeling and Simulation of Critical Infrastructure Systems for Homeland Security Applications, NISTIR 7785, National Institute of Standards and Technology, Gaithersburg, Maryland, 2011.

[14] I. Niang-Diop and H. Bosch, Formulating an adaptation strategy, in *Adaptation Policy Frameworks for Climate Change: Developing Strategies, Policies and Measures*, B. Lim and E. Spanger-Siegfried (Eds.), Cambridge University Press, Cambridge, United Kingdom, pp. 183–204, 2004.

[15] Railway Safety and Standards Board, Tomorrow's Railway and Climate Change Adaptation (T1009), London, United Kingdom (`www.rssb.co.uk/research-development-and-innovation/research-and-development/research-project-catalogue/t1009`), 2016.

[16] G. Stergiopoulos, P. Kotzanikolaou, M. Theocharidou and D. Gritzalis, Risk mitigation strategies for critical infrastructures based on graph centrality analysis, *International Journal of Critical Infrastructure Protection*, vol. 10, pp. 34–44, 2015.

[17] G. Stergiopoulos, P. Kotzanikolaou, M. Theocharidou, G. Lykou and D. Gritzalis, Time-based critical infrastructure dependency analysis for large-scale and cross-sectoral failures, *International Journal of Critical Infrastructure Protection*, vol. 12, pp. 46–60, 2016.

[18] G. Stergiopoulos, E. Vasilellis, G. Lykou, P. Kotzanikolaou and D. Gritzalis, Classification and comparison of critical infrastructure protection tools, in *Critical Infrastructure Protection X*, M. Rice and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 293–255, 2016.

[19] United Kingdom Climate Impacts Programme, Identifying Adaptation Options, Oxford, United Kingdom (`www.ukcip.org.uk/wp-content/PDFs/ID_Adapt_options.pdf`), 2008.

[20] T. Wall and M. Meyer, Risk-Based Adaptation Frameworks for Climate Change Planning in the Transportation Sector: A Synthesis of Practice, Transportation Research Circular E-C181, Transportation Research Board, Washington, DC, 2013.

[21] World Resources Institute, The National Adaptive Capacity Framework: Key Institutional Functions for a Changing Climate Institute, Pilot Draft, Washington, DC (`pdf.wri.org/working_papers/NAC_framework_2009-12.pdf`), 2009.

Chapter 2

EVALUATION OF ADDITIVE AND SUBTRACTIVE MANUFACTURING FROM THE SECURITY PERSPECTIVE

Mark Yampolskiy, Wayne King, Gregory Pope, Sofia Belikovetsky and Yuval Elovici

Abstract Additive manufacturing involves a new class of cyber-physical systems that manufacture 3D objects incrementally by depositing and fusing together thin layers of source material. In 2015, the global additive manufacturing industry had \$5.165 billion in revenue, with 32.5% of all manufactured objects used as functional parts. Because of their reliance on computerization, additive manufacturing devices (or 3D printers) are susceptible to a broad range of attacks. The rapid adoption of additive manufacturing in aerospace, automotive and other industries makes it an attractive attack target and a critical asset to be protected.

This chapter compares emerging additive manufacturing and traditional subtractive manufacturing from the security perspective. While the discussion compares the two manufacturing technologies, the emphasis is on additive manufacturing due to its expected dominance as the manufacturing technology of the future. The chapter outlines the additive and subtractive manufacturing workflows, proposes a framework for analyzing attacks on or using additive manufacturing systems and presents the major threat categories. In order to compare the two manufacturing paradigms from the security perspective, the differences between the two workflows are identified and the attack analysis framework is applied to demonstrate how the differences translate into threats. The analysis reveals that, while there is significant overlap with regard to security, fundamental differences in the two manufacturing paradigms require a separate investigation of additive manufacturing security.

Keywords: Additive manufacturing, subtractive manufacturing, attack framework

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 23–44, 2017.
https://doi.org/10.1007/978-3-319-70395-4_2

1. Introduction

The U.S. Department of Homeland Security has designated manufacturing as one of the sixteen critical infrastructure sectors [40]. The critical infrastructure sectors are not isolated, but are entangled in a complex web of dependencies and interdependencies [32]. A large-scale critical infrastructure disruption poses technical as well as economic, geopolitical, environmental and societal risks [17]. As a result, maintaining the security of manufacturing systems is of paramount importance.

Since the introduction of computer numeric control (CNC) machines in the 1940s, manufacturing processes have become increasingly computerized. Computer numeric control machines enable software control of cutting tools – they reduce a solid block of source material to a desired shape in a process commonly referred to as subtractive manufacturing (SM). In the 1980s, additive manufacturing (AM) was introduced – it is an alternative process in which thin layers of source material are deposited and fused together to form a 3D object. Additive manufacturing machines are often referred to as 3D printers. Subtractive and additive manufacturing machines, which are examples of cyber-physical systems (CPSs), are employed in computer-aided manufacturing (CAM), a process in which manufacturing as well as source material loading and physical transportation are computerized.

Subtractive manufacturing has dominated the manufacturing industry for decades and, until recently, additive manufacturing was predominantly used to produce low-quality plastic parts. However, additive manufacturing technologies have now matured to produce high-quality plastic and metal 3D-printed objects that are usable as functional parts, even in safety-critical systems such as jet engines. Additive manufacturing with other source materials, including ceramics, glass and composites, is rapidly approaching the maturity needed for industrial applications.

According to the 2015 Wohlers Report [41], the additive manufacturing industry had \$5.165 billion in revenue with 32.5% of all manufactured objects used as functional parts. An Ernst & Young study [30] reports that additive manufacturing technologies are being rapidly adopted around the world. In the United States, 16% of the surveyed companies had experience with additive manufacturing and another 16% were considering adopting the technology. The current world leader in additive manufacturing adoption is Germany with 37% of the surveyed companies already employing additive manufacturing and another 12% considering the technology. Numerous studies indicate that the adoption of additive manufacturing will continue to rise, potentially leading to its dominance as the manufacturing technology of the future.

The growing importance of additive manufacturing and its reliance on computerization have led several researchers to voice security concerns [2, 9, 39, 44, 45, 48–50]. However, it is not clear whether and how additive manufacturing security differs from the security of other cyber-physical systems, especially as traditional subtractive manufacturing serves the same purpose and employs similar processes and computerized components. This chapter attempts to answer these questions and identify the similarities and differences between additive and subtractive manufacturing technologies from the security perspective, but the emphasis remains on additive manufacturing.

2. Related Work

Additive and subtractive manufacturing systems are both cyber-physical systems. Therefore, the fundamental aspects of cyber-physical system attacks are applicable to both types of systems. Industrial control systems are also computer-controlled systems, but their mission and implementation differ significantly from additive and subtractive manufacturing systems.

No information is available about attacks that have specifically targeted subtractive manufacturing systems. However, there is a growing body of literature on potential attacks on or using additive manufacturing systems. Therefore, this topic is discussed in this section.

2.1 Cyber-Physical System Security

Cyber-physical systems generally employ closed or open control loops. A closed-loop system directly uses feedback information from sensors for decision making while an open-loop system makes decisions based on a model of a controlled physical process or the input of a human operator who makes decisions based on sensor readings. In a control system, information from sensors is fed to a computing unit that chooses the necessary actions; the signals from the computing unit are sent to actuators. As discussed in [10], a cyber-physical system can be attacked by interrupting one of these communications links or by injecting incorrect sensor information or control commands. Of course, cyber-physical systems may also be attacked by compromising their computing units.

Many cyber-physical systems operate under tight real-time constraints [24]. This is especially the case with safety-critical systems such as automobiles and airplanes. In these systems, disrupting the timing can significantly impact system behavior, even when all the commands and sensor information are correct.

Manipulations performed on cyber-physical systems and the effects of the manipulations are not necessarily in the same domain [46, 47]. While most cyber-physical system attacks focus on manipulations in the cyber domain that lead to effects in the physical domain, this does not have to be the case. Manipulations in the physical domain can cause effects in the physical and/or cyber domains.

Stuxnet is the most famous example of attacks on a production cyber-physical system [15]. The attacks were performed by malware installed on programmable logic controllers that managed centrifuges at a uranium enrichment facility in Natanz, Iran. When activated, the malware caused the centrifuges to rotate at speeds much higher and much lower than the normal operational speed. The attacks reportedly damaged more than 1,000 centrifuges at the uranium enrichment facility.

2.2 Additive Manufacturing Security

Several researchers have analyzed 3D printers and 3D printing processes for vulnerabilities. Networking and communications systems have been found to lack integrity checks when receiving design files [39]. Furthermore, communications protocols employed by desktop 3D printers can be exploited, enabling the retrieval of current and previously-printed 3D models, the termination of active printing jobs and the submission of unauthorized (new) jobs [13]. Software and firmware commonly used in desktop 3D printers contain numerous vulnerabilities [28]. A phishing attack can be used to install a backdoor that enables arbitrary, targeted manipulations of design files by a remote adversary [6]. Malicious software installed on a computer can be used to automate manipulations of design files [35]. Firmware installed on a 3D printer can be malicious [29] or compromised [42], enabling a range of manipulations of the manufacturing process. A range of physical attacks are also possible; manipulations of source materials can have far-reaching impacts on manufactured objects as well as on 3D printers and their manufacturing environments [48].

One of the broadly discussed topics in additive manufacturing security is the impact on intellectual property. Numerous problems have been identified by researchers who have analyzed the legal aspects of intellectual property protection in additive manufacturing environments [9, 20, 38] . For example, a 3D scan of a manufactured object is not considered to be an original technical drawing (blueprint) and, therefore, does not have the same legal protections [9]; thus, it can be used to circumvent copyright protection.

On the technical side, several attack schemes have been discussed and partially evaluated. In a scenario where additive manufacturing is outsourced, a malicious actor can assume the role of a manufacturing service provider and gain unrestricted access to design files and related specifications [44]. Side-channel emanations can be recorded and analyzed in order to steal designs, even when a 3D printer is air-gapped. Analyses of the acoustic emanations of desktop 3D printers have enabled the reconstruction of the geometries of printed objects [2, 33]. Researchers have also demonstrated that similar attacks are possible using infrared imaging [3] and magnetic side channel analysis [19].

It is important to note that, in the context of additive manufacturing, intellectual property is not limited to the specifications of the 3D object geometry – it can also include the specifications of the properties of the manufactured object and the manufacturing process parameters that ensure the fulfillment of the physical requirements [44]. Physical watermarking techniques have been discussed as a means to protect intellectual property in additive manufacturing environments [26].

Another broadly discussed threat category is the ability to inflict physical damage, especially when the quality of a manufactured part is sabotaged. Part quality can be degraded by introducing defects such as voids (internal cavities) [35] or printing portions of objects with the wrong or contaminated materials [50]. The sizes of the defects and their geometries and locations define the extent of manufactured part degradation [6].

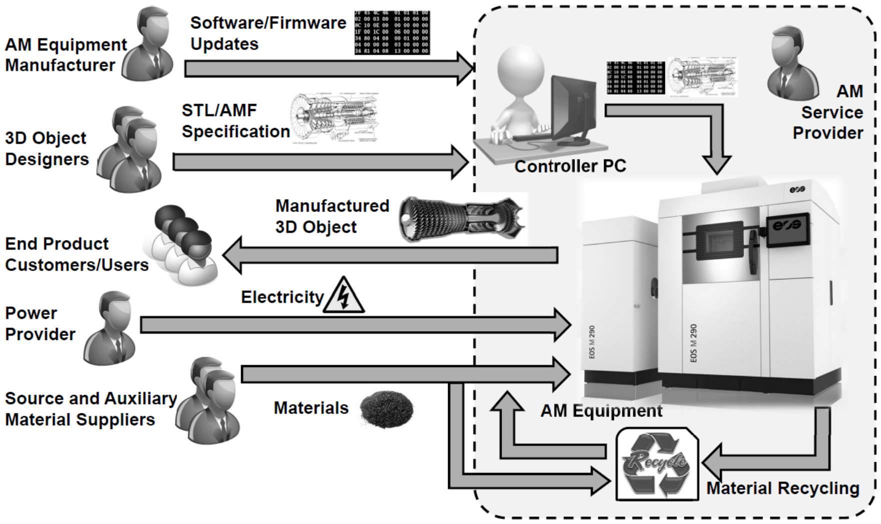


Figure 1. Additive manufacturing workflow.

Other manipulations involve changing the orientation of a printed part [48, 50], introducing additional skew along one of the built axes [12], changing the thickness of layers [29] and varying manufacturing parameters such as the energy of the heat source and scanning strategy in the case of the powder bed fusion process [48]. Manipulations of network command timing [31], energy supply [31] and source material composition [48] have also been identified as potential means of sabotaging parts. Indeed, in the case of additive manufacturing of metal parts, manipulations of manufacturing parameters can damage the additive manufacturing machinery itself and even contaminate the manufacturing environment [48, 49]. Researchers have proposed game theory [43] and side-channel emanation monitoring [12] as approaches for combating sabotage in additive manufacturing environments.

3. Manufacturing Workflows

This section describes the additive and subtractive manufacturing workflows.

3.1 Additive Manufacturing Workflow

Figure 1 presents an additive manufacturing workflow. This workflow is increasingly common when additive manufacturing is offered as a service. As of January 2017, the 3D Printing Businesses Directory lists 892 companies as offering 3D printing services (`3dprintingbusiness.directory`).

Additive manufacturing equipment (which includes but is not limited to 3D printers) is usually developed and provided by an original equipment manufacturer (OEM). In 2015, 62 system manufacturers in 20 countries produced and sold industrial-grade additive manufacturing equipment and hundreds of small companies offered desktop 3D printers [41]. Firmware and software updates for additive manufacturing equipment and controller workstations that extend functionality and fix bugs are provided by third-party companies. Open-source software is also commonly used by desktop 3D printers. Various mechanical, electrical and electronic components (motors, filters, etc.) are required for replacement purposes; these are sold by original equipment manufacturers or third-party companies and shipped directly to customers.

3D object blueprints are provided in the stereolithography [18] (STL) format, additive manufacturing file (AMF) format [4, 25] or 3D manufacturing format (3MF) [1], each of which specifies the computer-aided design (CAD) model of the 3D object to be manufactured. Object blueprints specified in STL/AMF/3MF files are often provided by external 3D object designers directly to additive manufacturing service providers. Another scenario involves a design being provided by the end-product customer who created the design (common for enterprise customers) or purchased it from a designer (common for individual customers).

At the additive manufacturing service provider's facility, an STL/AMF/3MF file can be directly transferred to a 3D printer (via a computer network or USB stick) or interpreted by the controller workstation. In the latter case, the workstation sends the 3D printer individual control commands (often in G-code [14], a language commonly used in computer-aided manufacturing) or as a tool path file containing a sequence of (often proprietary, 3D printer-specific) commands [35].

Additive manufacturing requires electricity and a variety of source and auxiliary materials. While source materials are included in the end-product, auxiliary materials support or enable production in some way. For example, support material structures enable the printing of complex geometries and inert gas (e.g., argon) is often employed when a laser is the heat source. Source materials for plastic printers are usually supplied by original equipment manufacturers; the source material market for metal printers is more open [41].

Depending on the additive manufacturing process, source material and part geometry, the production workflow can include several post-processing steps (not shown in Figure 1). The removal of support structures is a common step in the case of plastic objects; metal parts require hot isostatic pressing, finish machining and surface finishing. For functional parts, non-destructive testing is usually the final step. The Wohlers Report [41] lists several non-destructive testing methods commonly used in traditional subtractive manufacturing that are inadequate to validate the quality of additively-manufactured parts; these include fluorescent penetrant inspection (FPI), radiographic inspection and computed tomography (CT). Also, the inability to detect small defects in additively-manufactured parts using an ultrasonic c-scan has been reported [50]. After all the required production and post-production steps are completed, the manufactured objects are delivered to customers via physical carriers.

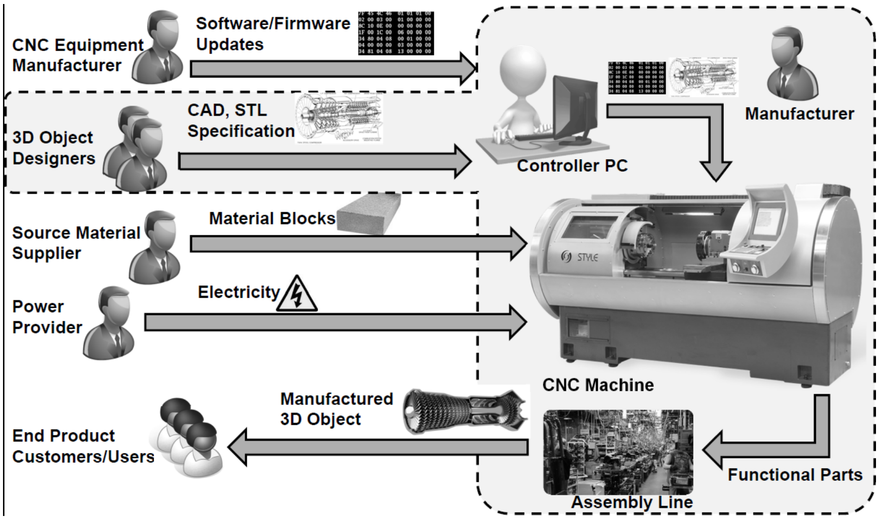


Figure 2. Subtractive manufacturing workflow.

In order to reduce the environmental impact and manufacturing costs, some of the source materials that remain after an additive manufacturing process may be recycled. This is especially true for power bed fusion in which thin layers of powdered source material (usually metal or polymer) are distributed in a bed and fused by a heat source (laser or electron beam). The exposure of unused powder to high temperatures causes the properties of particles to change (in the case of plastic) and/or the particles to agglomerate into large clusters. Both these changes can have negative impacts on the final product quality. Therefore, the remaining powder is often sieved and mixed with "virgin" powder in proportions that minimize the negative impact on part quality.

Table 1 summarizes the main aspects of additive and subtractive manufacturing and compares their workflows.

3.2 Subtractive Manufacturing Workflow

The subtractive manufacturing workflow presented in Figure 2 is broadly similar to the additive manufacturing workflow, but some notable differences exist. While a subtractive manufacturing facility may provide manufacturing services to external customers, it is significantly less common than in the case of additive manufacturing. Instead, a subtractive manufacturing provider typically supplies complete products as a commodity and the part designers are usually located "in house."

As with additive manufacturing, software and firmware updates and manufacturing jobs are initiated from a workstation. However, unlike additive manufacturing, software and firmware updates are usually provided by origi-

Table 1. Comparison of additive and subtractive manufacturing.

	Additive Manufacturing	**Subtractive Manufacturing**
Actors		
Software Provider	• OEMs and commercial software developers	
	• Open-source/community	• –
3D Object Designer	• Increasingly individuals • External to enterprise	• Predominantly enterprises • Internal to enterprise
Customer Relationship	• Increasingly short term • Low volume	• Long term • High volume
Actors/Roles	• High	• Low
Materials		
Source	• Often wire or powder	• Solid blocks
Auxiliary	• Extensively used • Influence product quality	• Barely used/relevant • –
Manufacturing Process		
G-Code Command Defines	• Deposited/fused material • Exterior/interior geometry • Object physical properties	• Material to be removed • Exterior geometry • –
Power Outage	• Impacts manufacturing speed	
	• Impacts part quality	• –
Timing	• Impacts manufacturing speed	
	• Impacts part quality	• –
Maturity Level		
Workflows	• Mature and well-established (manual and CAM)	
Software and Firmware	• New and immature with many bugs	• Very mature with few bugs
Quality Control	• SM tools/approaches conditionally applicable • Immature tools/approaches	• Tools/approaches well-established and understood • Mature tools/approaches
Availability/Accessibility		
Equipment	• Enterprises/employees	
	• Private individuals	• –
Blueprints	• Restricted access (enterprise-guarded intellectual property)	
	• Third-party commercial and non-commercial websites	• –

nal equipment manufacturers or commercial software developers; community involvement in open-source efforts are negligible. Furthermore, control com-

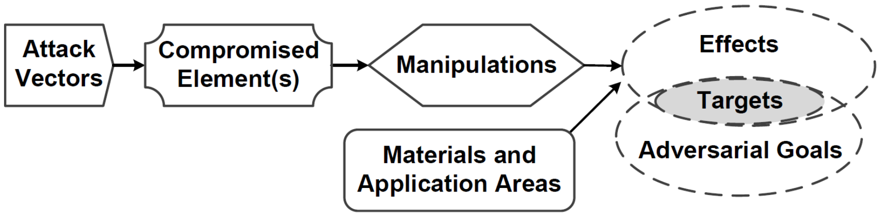


Figure 3. Attacks on or using computer-aided manufacturing (based on [49]).

mands (in G-code) sent from a controller workstation to a computer numeric control machine define the movements of tools that remove the extraneous material to create an object.

The source material used in subtractive manufacturing also differs. In additive manufacturing, the source material is usually in wire or powder form; in the case of subtractive manufacturing, solid blocks of material are used. Auxiliary material is less important in subtractive manufacturing; for example, a stream of water can be used to remove shavings or cool a manufactured part. While a computer numeric control machine also uses electrical power, its importance to the process is different (this is discussed in more detail below).

Probably the most obvious difference between the additive and subtractive manufacturing workflows is that subtractive manufacturing has no material recycling. Instead, subtractive manufacturing requires assembly; this is because subtractive manufacturing is limited to defining the external shape of an object – if internal structure is needed, the object is produced by assembling multiple components, each of which is manufactured separately.

Finally, the two workflows have different numbers of actors and different actor relationships. Subtractive manufacturing usually has long-term, high-volume customer-provider relationships; in additive manufacturing, these relationships are short-term with small production runs. Additionally, the number of additive manufacturing service providers is growing rapidly. This is because the same additive manufacturing equipment can be used to produce a variety of 3D objects. In contrast, subtractive manufacturing often requires highly specialized equipment.

4. Attack Analysis Framework

This section presents a framework for analyzing attacks on or using computer-aided manufacturing. The security threat categories for additive manufacturing are also presented. The threats are also relevant to subtractive manufacturing.

4.1 Attacks

Figure 3 presents key attacks on or using computer-aided manufacturing [49]. Several attack vectors can be used to compromise one or more elements of the manufacturing workflows shown in Figures 1 and 2 (for additive manufacturing

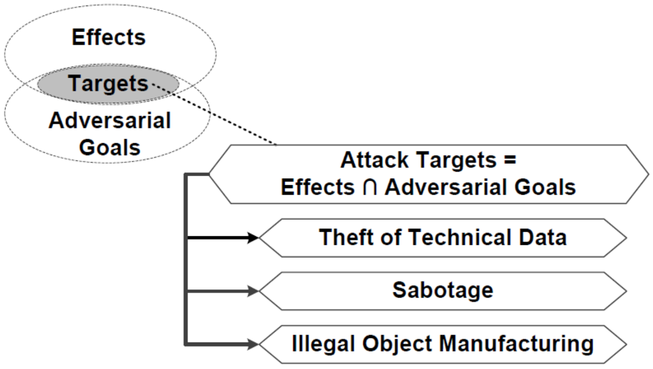

Effects
Targets
Adversarial Goals
Attack Targets = Effects ∩ Adversarial Goals
Theft of Technical Data
Sabotage
Illegal Object Manufacturing

Figure 4. Major threat categories.

and subtractive manufacturing, respectively). For example, social engineering can be used to trick users into installing malicious software or firmware updates. The compromised element(s), their roles in a workflow and the degree to which an adversary can control the element(s) determine the specific manipulations that the adversary can perform. In conjunction with the manufacturing equipment, source materials and application area of the manufactured parts, the manipulations determine the achievable effects. In the case of a functional part of a device (e.g., jet engine blade), slight changes to the size or shape can render the entire device less efficient. Other changes may degrade the mechanical properties of a part so that it breaks during use [35, 48, 50] or may cause material fatigue to develop much faster than expected [6].

4.2 Security Threat Categories

Only a fraction of the effects that can be produced by attacks intersects with the goals of an adversary. For example, not all changes to the internal geometry of an object (e.g., positions and sizes of internal cavities) would compromise the mechanical properties of the object. Furthermore, the goals and objectives (i.e., "stepping stones" for achieving the adversary's goals) differ across adversaries. For instance, a hostile nation state may be interested in compromising safety whereas a malicious competitor may be interested in increasing manufacturing costs. The intersection of the attack effects and adversarial goals are referred to as attack targets or threats because they are both achievable by an adversary and are of interest to the adversary.

Figure 4 presents the three major security threat categories (or attack targets) that have been identified for additive manufacturing. Two of the categories, theft of technical data and sabotage of additive manufacturing, are discussed in the research literature (see Section 2). A theft of technical data attack seeks to illegally replicate 3D objects or the manufacturing process itself. Sabotage attacks seek to inflict physical damage, e.g., by compromising the quality of manufactured parts or physically damaging additive manufacturing equipment.

Several articles discuss the misuse of 3D printers for manufacturing illegal items such as firearms and components of explosive devices [7, 34, 37]. With the exception of discussing the legal aspects [8, 22, 27, 38], the research literature has largely ignored this last category.

5. Security Analysis

This section compares the additive and subtractive manufacturing paradigms from the security perspective. The analysis focuses on the similarities and differences in the manufacturing workflows discussed in Section 3. The analysis is structured according to the semantically-distinct elements of attacks on or using cyber manufacturing. The results are summarized in Tables 2 and 3.

5.1 Attack Vectors

Additive and subtractive manufacturing equipment are cyber-physical systems. Both types of manufacturing have similar workflows with almost identical categories of actors and information, software and material flows. Cyber and physical attack vectors can be exploited to compromise various elements of the manufacturing systems. Therefore, the attack vectors that can be used to compromise the two types of systems are almost identical.

Many attack vectors considered in cyber security studies also apply to both workflows, enabling the compromise of cyber components. These include spear phishing, hacking, source file worms, etc. Because of the novelty and relative immaturity of additive manufacturing, additive manufacturing software and firmware are significantly more vulnerable to cyber attacks than their subtractive manufacturing counterparts.

Malicious insiders are a classical attack vector that can target both manufacturing workflows and compromise cyber and physical supply chains. Also, social engineering is applicable to both workflows.

However, certain differences between the attack vectors for the two workflows arise from the differences existing between the workflows themselves (Table 1). The following are the most notable differences that enable the attack vectors unique to additive manufacturing:

- Supplier-consumer relationships in additive manufacturing are significantly more dynamic and flexible than in subtractive manufacturing.

- The number of potential suppliers in additive manufacturing is much larger than in subtractive manufacturing; this also includes third-party software and firmware providers.

- The number of additive manufacturing service providers is much higher than in subtractive manufacturing. More importantly, they often provide manufacturing services instead of specific products.

Table 2. Comparison of additive and subtractive manufacturing security.

	Additive Manufacturing	**Subtractive Manufacturing**
Attack Vectors		
External Adversary Compromises Benign Workflow	• Hacking • Source file worms • Physical attack on physical supply chain • Social engineering • Et cetera	
	• Specially-crafted blueprint files • Malicious software and/or firmware	• – • –
Malicious Actors Assume Existing Roles	• 3D object designer • Service provider • Software developer • Source/auxiliary materials provider	• – • – • – • –
Compromised Elements		
Source Material	• Often wire or powder	• Solid blocks
General Categories	• Roles/actors • Software, firmware and hardware • Network communications • Physical supply chain • Power supply	
Workflow-Specific	• Post-processing • Material recycling	• Assembly line • –

- Object designers in additive manufacturing are primarily external actors while designers in subtractive manufacturing are internal actors (e.g., a division in an enterprise).

These differences induce attack vectors in additive manufacturing environments; the attack vectors are either new or have lower probability and limited impact in subtractive manufacturing environments. First, in an additive manufacturing environment, an adversary can assume one of two roles: (i) 3D object designer; or (ii) additive manufacturing service provider. These two roles are potentially malicious and do not require hacking, social engineering or some other means to compromise the actor. A malicious 3D object designer can create special files (e.g., with embedded worms) while a malicious additive manufacturing service provider could obtain access to and compromise STL/AMF/3MF 3D object blueprint files. Additionally, the involvement of third parties in software and firmware development provides opportunities for

Table 3. Comparison of additive and subtractive manufacturing security.

	Additive Manufacturing	**Subtractive Manufacturing**
Manipulations		
General Categories	• Compromise another workflow element • Information exfiltration • Control loop attacks • False sensor reading is provided • False control action is provided • Sensor reading is not provided or is not followed • Control action is not provided or is not followed • Control action is provided too late or is out of sequence • Control action is stopped too soon or is applied too long • Sensor reading is provided too late or is out of sequence • Sensor reading is stopped too soon or is applied too long • Power supply spikes • Source material chemical composition	
Workflow-Specific	• Source material properties • Auxiliary materials • Power supply interruption • Power supply properties • Operation duration/speed	• Source block quality • – • – • – • –
Effects/Attack Targets		
Theft of Technical Data	• 3D object geometry	
	• Required properties • Manufacturing process specifications	• – • –
Sabotage	• Integration ability • Equipment damage • Environmental contamination	
	• Physical properties of degraded part • Weight • Weight distribution • Implosion/explosion • Environmental damage	• – • – • – • – • –
Illegal Part Manufacturing	• Access to illegal or illegally-manufactured items	
	• Required equipment is increasingly accessible • Blueprints are increasingly available on the Internet	• – • –

adversaries to develop and distribute malicious software and firmware, compromising equipment at additive manufacturing service provider sites.

5.2 Compromised Elements

According to Yampolskiy et al. [49], elements that can be compromised in the additive manufacturing workflow belong to four general categories: (i) actors or workflow roles assumed by the actors; (ii) software, firmware and/or hardware; (iii) network communications; and (iv) physical supply chain. An additional fifth category is power supply, whose impact on additive manufacturing is discussed in [31].

All five categories of elements can be compromised in additive and subtractive manufacturing environments. There are, however, noticeable differences at a fine level – based on where the elements belonging to the five categories are employed in a workflow and their purpose in the workflow. A major difference arises from the multiplicity and diversity of the auxiliary materials, and their significance on the quality of a manufactured part compared with traditional subtractive manufacturing. Another is that additive manufacturing employs several post-processing steps to improve part quality, some of which have limited or no importance in subtractive manufacturing (e.g., hot isostatic pressing). Furthermore, in some additive manufacturing processes, unused source material can be recycled and reused whereas in subtractive manufacturing, removed material is not directly reusable. Indeed, the recycling process in additive manufacturing itself can impact part quality because it affects the overall quality of the source material used. Additionally, the subtractive manufacturing workflow usually incorporates a step that is largely irrelevant in additive manufacturing – the assembly phase during which individually-manufactured pieces are assembled to create a functional part. Last, but not least, non-destructive testing equipment is commonly employed to verify the quality of manufactured functional parts; this equipment is largely computerized and, therefore, can be compromised by cyber attacks. Note that this is independent of reports that various non-destructive testing approaches used in subtractive manufacturing fail to detect defects in additively-manufactured parts [41, 50].

5.3 Manipulations

Every compromised element of an additive or subtractive manufacturing workflow can be used as a staging point to compromise other workflow elements. The role of the compromised element(s) in a manufacturing workflow and degree to which an adversary exercises control over it/them determine the manipulations that are possible.

If the controller workstation or other computing equipment is compromised and connected to the Internet, classical cyber attacks that exfiltrate information or enable remote access via backdoors are possible. The attacks can be used to gain access to technical data, manipulate STL/AMF/3MF 3D object design files and modify key manufacturing process parameters. Illegal access to technical data or its use by an adversary who impersonates an additive manufacturing service provider are plausible attack vectors. An adversary who has direct access to equipment can also create restricted objects such as firearms; of course, this applies to both additive and subtractive manufacturing equipment that can produce the restricted objects or their components. Principal differences between additive and subtractive manufacturing technologies with regard to malicious manipulations are the broad availability of additive manufacturing equipment and access to object blueprints on the Internet.

A number of manipulations are possible if the controller workstation, additive manufacturing equipment or network communications between the controller and equipment are compromised. Since these three components comprise a cyber-physical system, most cyber-physical attacks are applicable to additive manufacturing systems. At the fundamental level, communications between a sensor and controller or between a controller and actuator can be interrupted or corrupted [10]. Furthermore, in the case of real-time processes – characterized by cyber-physical systems in general and additive/subtractive manufacturing systems in particular – the correctness of operations and their timing are of paramount importance [24]. Therefore, manipulations of sensor readings and control commands involve disruptions of their timing and order [31]. The physical process that is controlled by the particular control loop ultimately determines the effects of any manipulations; this applies to additive and subtractive manufacturing alike. Furthermore, the timing of manipulations in the manufacturing cycle influences the effects and their extent [23].

Additive and subtractive manufacturing equipment require power. Power spikes can affect both types of equipment in a similar manner (e.g., damage electric motors). However, power supply interruptions have different impacts on the manufactured objects. In the case of subtractive manufacturing, power interruptions only affect the speed of production. However, interruptions can have severe impacts on additively-manufactured part quality (this is discussed this in more detail below); therefore, these manipulations are categorized as additive-manufacturing-specific. Similarly, manipulations (e.g., control loop attacks) that affect the durations of particular operations impact additive and subtractive manufacturing differently – manufacturing speed for both technologies and part quality, in addition, for additive manufacturing.

Manipulations of source materials are possible in additive and subtractive manufacturing, but the available manipulations can be quite different. In both cases, the chemical compositions of source materials can be changed. In subtractive manufacturing, the quality of the source material blocks can be manipulated (e.g., they can have different microstructures). In the case of additive manufacturing, properties such as the form factor can be manipulated (e.g., size and shape of the powder particles or diameter of the wire). Additionally, the chemical compositions of the auxiliary materials used in additive manufacturing can be manipulated. In general, the attack vectors that enable source and auxiliary material manipulations in additive manufacturing are manifold and are much easier to exploit than in the case of subtractive manufacturing.

5.4 Effects

This section discusses the effects of attacks on additive and subtractive manufacturing. The discussion of effects is structured in terms of the three security threat categories (Figure 4).

Theft of Technical Data. Two scenarios in which intellectual property violations are possible in additive and subtractive manufacturing are: (i) compromise of the cyber infrastructure of a benign manufacturer (e.g., when a controller workstation is connected to the Internet); and (ii) manufacturer is a malicious actor. In both cases, the adversary is interested in gaining access to the victim's intellectual property and eventually commits an infringement.

The differences between the two manufacturing technologies arise from what is considered to be intellectual property. Clearly, 3D object geometry corresponds to intellectual property in both manufacturing paradigms. However, in the case of subtractive manufacturing, the material properties of the manufactured part depend directly on the properties of the source material block. In contract, in additive manufacturing, the part material is created (and, thus, its properties are defined) during the manufacturing process itself. Therefore, in additive manufacturing, the required properties of the manufactured 3D object as well as the manufacturing process specifications correspond to intellectual property [44].

Sabotage. The following scenarios enable the quality of a manufactured part and/or manufacturing equipment to be sabotaged:

- Manipulation of the object specifications regardless of its representation, which could be a STL/AMF/3MF file, individual G-code commands, toolpath file, etc.
- Compromise of the cyber infrastructure of the manufacturing process.
- Compromise of the physical supply chain of source and auxiliary materials.
- Manipulation of the power supply.

As discussed in [42] and demonstrated experimentally in [35], manufactured parts can be sabotaged by modifying their exterior shapes and dimensions, thereby affecting their integration; this is clearly applicable to both manufacturing paradigms. In particular, this is achieved by modifying object specification files or compromising the cyber infrastructure of the manufacturing process. For example, compromised firmware could manipulate the thickness of the printed layers in additive manufacturing [29].

However, several sabotage attacks are specific to additive manufacturing and are not possible in subtractive manufacturing. Introducing voids in a manufactured object [35] or replacing portions of a manufactured object with a different material [50] can degrade the physical properties of the object; this can also change the weight and weight distribution of the object [49]. Furthermore, various additive manufacturing parameters can be manipulated to affect the microstructure of the manufactured object and, thus, its physical properties; the parameters include build direction, heat source energy and scanning strategy [48]. The ability to sabotage the quality of a 3D-printed part by manipulating some of these parameters has been proven experimentally [29, 42, 50].

As Stuxnet [15] and the Aurora experiment [36] have demonstrated, a cyber-physical attack that forces equipment to operate outside its designated operational ranges can induce physical damage. Such cyber-physical attacks, which exploit the fact that cyber components control physical processes, are applicable to both manufacturing paradigms. An attack that damages a process chamber used in additive manufacturing could release hazardous materials to the environment. This is a manufacturing-process-specific case of sabotage. An example is the release of the metal powder used in selective laser melting, which is hazardous because of the fine particle size (0.1 to 5 μm) [21].

Last, but not least, in the case of additive manufacturing with metals, damage to the process chamber can lead to an explosion or implosion with subsequent damage to the manufacturing equipment environment and a likely fire [48, 49]. For example, when the heat source is a laser, the production chamber is commonly filled with inert gas to prevent an exothermic reaction; increasing the oxygen pressure can cause the combustible fine metal powder to explode. A vacuum environment is maintained to minimize the deflection of electrons when an electron beam is used as the heat source. Therefore, a slow leak is a more likely outcome of process chamber damage, but an implosion caused by a specially-crafted attack cannot be ruled out [49]. While safety mechanisms are implemented to shut down the heat sources used in additive manufacturing during safety-critical events, these mechanisms can be disabled by malicious or compromised 3D printer firmware.

Illegal Object Manufacturing. Two possible scenarios for this threat category are: (i) an adversary owns the blueprint of a potentially illegal object and the requisite manufacturing equipment; and (ii) an adversary has access to the blueprint and the manufacturing equipment.

In these scenarios, there are no technical differences in using additive or subtractive manufacturing to produce illegal items. The only practical differences arise from two factors. First, high quality additive manufacturing equipment is increasingly accessible to private owners, unlike industrial-grade subtractive manufacturing equipment that is predominantly owned by enterprises. Second, STL/AMF/3MF blueprint files for 3D printing are widely available on the Internet (e.g., from makers' forums); in contrast, blueprints used in subtractive manufacturing are generally well protected because of their intellectual property value to subtractive manufacturing enterprises.

6. Conclusions

The rapid proliferation of additive manufacturing has raised concerns about its security. In this context, it is important to understand the extent to which additive manufacturing differs from traditional subtractive manufacturing. This chapter has evaluated the additive and subtractive manufacturing workflows, and has presented a framework for analyzing attacks on or using computer-aided manufacturing systems along with the major threat categories that target additive manufacturing. In particular, the framework was applied to identify how the differences in the workflows translate to the security domain.

The analysis concludes that, while there are overlaps in the security of additive and subtractive manufacturing, significant differences exist. This is true for all the components in the analysis framework – attack vectors, compromised elements, manipulations and attack targets. Two of the three major threat categories for additive manufacturing – theft of technical data and sabotage – differ considerably from subtractive manufacturing. However, in the case of the third category – illegal object manufacturing – no notable differences exist between additive and subtractive manufacturing. The results of this investigation coupled with the increasing importance of additive manufacturing in all the critical infrastructure sectors emphasize the need to address the security aspects in a comprehensive and timely manner.

References

[1] 3MF Consortium, 3D Manufacturing Format, Core Specification and Reference Guide, Version 1.1, Wakefield, Massachusetts (`3mf.io/wp-content/uploads/2016/03/3MFcoreSpec_1.1.pdf`), 2015.

[2] M. Al Faruque, S. Chhetri, A. Canedo and J. Wan, Acoustic side-channel attacks on additive manufacturing systems, *Proceedings of the Seventh International Conference on Cyber-Physical Systems*, article 19, 2016.

[3] M. Al Faruque, S. Chhetri, S. Faezi and A. Canedo, Forensics of Thermal Side-Channels in Additive Manufacturing Systems, CECS Technical Report #16–01, Center for Embedded and Cyber-Physical Systems, University of California, Irvine, Irvine, California, 2016.

[4] American Society for Testing and Materials, ISO/ASTM52915-16: Standard Specification for Additive Manufacturing File Format (AMF), Version 1.2, West Conshohocken, Pennsylvania, 2016.

[5] H. Atkinson and S. Davies, Fundamental aspects of hot isostatic pressing: An overview, *Metallurgical and Materials Transactions A*, vol. 31(12), pp. 2981–3000, 2000.

[6] S. Belikovetsky, M. Yampolskiy, J. Toh, J. Gatlin and Y. Elovici, dr0wned – Cyber-physical attack with additive manufacturing, *Proceedings of the Eleventh USENIX Workshop on Offensive Technologies*, 2017.

[7] N. Bilton, The rise of 3-D printed guns, *The New York Times*, August 13, 2014.

[8] J. Blackman, The 1st Amendment, 2nd Amendment and 3D printed guns, *Tennessee Law Review*, vol. 81(3), pp. 479–538, 2014.

[9] A. Brown, M. Yampolskiy, J. Gatlin and T. Andel, Legal aspects of protecting intellectual property in additive manufacturing, in *Critical Infrastructure Protection X*, M. Rice and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 63–79, 2016.

[10] A. Cardenas, S. Amin and S. Sastry, Secure control: Towards survivable cyber-physical systems, *Proceedings of the Twenty–Eighth International Conference on Distributed Computing Systems Workshops*, pp. 495–500, 2008.

[11] K. Chan, M. Koike, R. Mason and T. Okabe, Fatigue life of titanium alloys fabricated by additive layer manufacturing techniques for dental implants, *Metallurgical and Materials Transactions A*, vol. 44(2), pp. 1010–1022, 2013.

[12] S. Chhetri, A. Canedo and M. Al Faruque, KCAD: Kinetic cyber-attack detection method for cyber-physical additive manufacturing systems, *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, 2016.

[13] Q. Do, B. Martini and K. Choo, A data exfiltration and remote exploitation attack on consumer 3D printers, *IEEE Transactions on Information Forensics and Security*, vol. 11(10), pp. 2174–2186, 2016.

[14] Electronic Industries Association, ANSI/EIA RS-274-D-1980: Interchangeable Variable Block Data Format for Positioning, Contouring and Contouring/Positioning Numerically Controlled Machines, Washington, DC, 1980.

[15] N. Falliere, L. O'Murchu and E. Chien, W32.Stuxnet Dossier, Version 1.4, Symantec, Mountain View, California, 2011.

[16] W. Frazier, Metal additive manufacturing: A review, *Journal of Materials Engineering and Performance*, vol. 23(6), pp. 1917–1928, 2014.

[17] D. Helbing, Globally networked risks and how to respond, *Nature*, vol. 497(7447), pp. 51–59, 2013.

[18] J. Hiller and H. Lipson, STL 2.0: A proposal for a universal multi-material additive manufacturing file format, *Proceedings of the Solid Freeform Fabrication Symposium*, pp. 266–278, 2009.

[19] A. Hojjati, A. Adhikari, K. Struckmann, E. Chou, T. Nguyen, K. Madan, M. Winslett, C. Gunter and W. King, Leave your phone at the door: Side channels that reveal factory floor secrets, *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 883–894, 2016.

[20] T. Holbrook and L. Osborn, Digital patent infringement in an era of 3D printing, *University of California Davis Law Review*, vol. 48(4), pp. 1319–1385, 2015.

[21] Inside Metal Additive Manufacturing, The Role of (Super) Powders in SLM (`www.insidemetaladditivemanufacturing.com/blog/the-role-of-super-powders-in-slm`), April 10, 2014.

[22] J. Johnson, Print, lock and load: 3-D printers, creation of guns and the potential threat to Fourth Amendment rights, *Journal of Law, Technology and Policy*, vol. 2013(2), pp. 337–361, 2013.

[23] M. Krotofil, A. Cardenas, J. Larsen and D. Gollmann, Vulnerabilities of cyber-physical systems to stale data – Determining the optimal time to launch attacks, *International Journal of Critical Infrastructure Protection*, vol. 7(4), pp. 213–232, 2014.

[24] E. Lee, Cyber physical systems: Design challenges, *Proceedings of the Eleventh IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, pp. 363–369, 2008.

[25] H. Lipson, AMF tutorial: The basics (Part 1), *3D Printing and Additive Manufacturing*, vol. 1(2), pp. 85–87, 2014.

[26] B. Macq, P. Alface and M. Montanola, Applicability of watermarking for intellectual property rights protection in a 3D printing scenario, *Proceedings of the Twentieth International Conference on 3D Web Technology*, pp. 89–95, 2015.

[27] K. McMullen, Worlds collide when 3D printers reach the public: Modeling a digital gun control law after the Digital Millennium Copyright Act, *Michigan State Law Review*, vol. 2044(1), pp. 187–225, 2014.

[28] S. Moore, P. Armstrong, T. McDonald and M. Yampolskiy, Vulnerability analysis of desktop 3D printer software, *Proceedings of the 2016 Resilience Week*, pp. 46–51, 2016.

[29] S. Moore, W. Glisson and M. Yampolskiy, Implications of malicious 3D printer firmware, *Proceedings of the Fiftieth Hawaii International Conference on System Sciences*, pp. 6089–6098, 2017.

[30] A. Muller and S. Karevska, How Will 3D Printing Make Your Company the Strongest Link in the Value Chain? EY's Global 3D Printing Report 2016, Ernst & Young, Mannheim, Germany, 2016.

[31] G. Pope, STPA for additive manufacturing, presented at the *Systems Theoretic Accident Model and Processes Workshop*, 2016.

[32] S. Rinaldi, J. Peerenboom and T. Kelly, Identifying, understanding and analyzing critical infrastructure interdependencies, *IEEE Control Systems*, vol. 21(6), pp. 11–25, 2001.

[33] C. Song, F. Lin, Z. Ba, K. Ren, C. Zhou and W. Xu, My smartphone knows what you print: Exploring smartphone-based side-channel attacks against 3D printers, *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 895–907, 2016.

[34] A. Sternstein, Things can go kaboom when a defense contractor's 3-D printer gets hacked, *Nextgov*, September 11, 2014.

[35] L. Sturm, C. Williams, J. Camelio, J. White and R. Parker, Cyber-physical vulnerabilities in additive manufacturing systems, *Proceedings of the Twenty-Fifth International Solid Freeform Fabrication Symposium*, pp. 951–963, 2014.

[36] M. Swearingen, S. Brunasso, J. Weiss and D. Huber, What you need to know (and don't) about the Aurora vulnerability, *POWER Magazine*, September 1, 2013.

[37] D. Tirone and J. Gilley, 3D printing: A new threat to gun control and security policy? *The Conversation* (`theconversation.com/3d-printing-a-new-threat-to-gun-control-and-security-policy-61416`), July 19, 2016.

[38] J. Tran, The law and 3D printing, *John Marshall Journal of Information Technology and Privacy Law*, vol. 31(4), pp. 505–520, 2015.

[39] H. Turner, J. White, J. Camelio, C. Williams, B. Amos and R. Parker, Bad parts: Are our manufacturing systems at risk of silent cyberattacks? *IEEE Security and Privacy*, vol. 13(3), pp. 40–47, 2015.

[40] U.S. Department of Homeland Security, Critical Infrastructure Sectors, Washington, DC (`www.dhs.gov/critical-infrastructure-sectors`), 2017.

[41] Wohlers Associates, Wohlers Report 2016, Fort Collins, Colorado, 2016.

[42] C. Xiao, Security attack on 3D printing, presented at the *xFocus Information Security Conference*, 2013.

[43] Z. Xu and Q. Zhu, Cross-layer secure cyber-physical control system design for networked 3D printers, *Proceedings of the American Control Conference*, pp. 1191–1196, 2016.

[44] M. Yampolskiy, T. Andel, J. McDonald, W. Glisson and A. Yasinsac, Intellectual property protection in additive layer manufacturing: Requirements for secure outsourcing, *Proceedings of the Fourth Program Protection and Reverse Engineering Workshop*, article 7, 2014.

[45] M. Yampolskiy, T. Andel, J. McDonald, W. Glisson and A. Yasinsac, Towards security of additive layer manufacturing, presented at the *Thirtieth Annual Computer Security Applications Conference*, 2014.

[46] M. Yampolskiy, P. Horvath, X. Koutsoukos, Y. Xue and J. Sztipanovits, Taxonomy for descriptions of cross-domain attacks on CPSs, *Proceedings of the Second ACM International Conference on High Confidence Networked Systems*, pp. 135–142, 2013.

[47] M. Yampolskiy, P. Horvath, X. Koutsoukos, Y. Xue and J. Sztipanovits, A language for describing attacks on cyber-physical systems, *International Journal of Critical Infrastructure Protection*, vol. 8, pp. 40–52, 2015.

[48] M. Yampolskiy, L. Schutzle, U. Vaidya and A. Yasinsac, Security challenges of additive manufacturing with metals and alloys, in *Critical Infrastructure Protection IX*, M. Rice and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 169–183, 2015.

[49] M. Yampolskiy, A. Skjellum, M. Kretzschmar, R. Overfelt, K. Sloan and A. Yasinsac, Using 3D printers as weapons, *International Journal of Critical Infrastructure Protection*, vol. 14, pp. 58–71, 2016.

[50] S. Zeltmann, N. Gupta, N. Tsoutsos, M. Maniatakos, J. Rajendran and R. Karri, Manufacturing and security challenges in 3D printing, *Journal of the Minerals, Metals and Materials Society*, vol. 68(7), pp. 1872–1881, 2016.

Chapter 3

DETECTING DATA MANIPULATION ATTACKS ON THE SUBSTATION INTERLOCKING FUNCTION USING DIRECT POWER FEEDBACK

Eniye Tebekaemi, Edward Colbert and Duminda Wijesekera

Abstract Any form of deliberate physical or cyber activity that attempts to undermine the control mechanisms that maintain the key goals of reliability, efficiency and safety of a physical system can be considered to be an attack on the system. Indeed, an attack can be as subtle as a configuration change that prevents the optimal operation of a power system.

This chapter describes an approach that enhances the security of the interlocking function in a power distribution substation by using the power flow behavior of the physical system during switching events as direct power feedback. The approach detects potential over-the-network data modification attacks on the interlocking function using out-of-bounds sensor measurements. The direct power feedback adds an extra layer of security and redundancy to existing power substation interlocking function protection mechanisms.

Keywords: Smart grid, substation, interlocking function, attack detection

1. Introduction

When smart grids become operational, power substations will be expected to support bidirectional power flows between distributed energy sources, storage facilities and power consumers. Substations use switchgear to maintain the appropriate flow of power, protect equipment and provide redundancy during power source and equipment failures. Interlocking functions in substations prevent improper operations of switchgear by maintaining information about their operational states and permissible state transitions from their current states to the next states. This ensures correct switching sequences and prevents switch operations that could violate the integrity of the substations. Due to the significant role played by the interlocking function in the safe and reliable

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 45–62, 2017.
https://doi.org/10.1007/978-3-319-70395-4_3

operation of power systems, any attack that compromises the state information and state transition integrity of an interlocking function can have disastrous consequences.

The interlocking function implemented in intelligent electronic devices (IEDs) in an IEC 61850 based power substation relies exclusively on Generic Object Oriented Substation Event (GOOSE) status messages between switchgear controllers in order to maintain the state information of all the switchgear in the substation, This reliance on GOOSE status messages is potentially a single point of failure for the interlocking function. Moreover, it fails to provide the substation with the required resilience to cyber-physical attacks.

This chapter examines the unique physical system behavior characteristics in response to switchgear events and extracts useful consequent system behavior attributes in order to specify a method that uniquely identifies switchgear events and to provide a cyber-physical security solution that integrates these observations into traditional cyber security controls. The physical system behavior is an important, but often neglected, part of cyber-physical security research. Indeed, understanding and considering the physical behavior of a power substation plays a key role in designing a resilient security solution.

2. Related Work

Peripheral information from sensor measurements is often used to monitor the state and behavior of cyber-physical systems. However, little work has been done on integrating this information in intrusion detection systems. Colbert et al. [2] have developed a process-oriented method for intrusion detection in industrial control systems. Data from critical elements in a physical system are collected by sensors and used to estimate the system state. Control operations sent over the network are intercepted by the intrusion detection system and evaluated using the estimated system state and system guard conditions. An alert is raised when a control operation violates the guard conditions based on the estimated system state.

Koustandria et al. [5] have developed a hybrid control network intrusion detection system. Expected communications patterns and the limitations of a physical system are leveraged to detect a wide range of attacks. The system, which is designed for protective digital relays in power transmission grids, detects attacks using packet sequences, time gaps between packets and the measured current in relays. Every packet and communications flow are evaluated against the expected packet sequence, maximum allowed time delay and measured current in the relay. An attack is detected when any of these constraints are violated or a circuit breaker activation request is received when the measured current is less than the cut-off current.

Mitchell et al. [7] have created a behavioral-rule-based intrusion detection system for unmanned air vehicles. A set of system (physical) behavior rules and system state transformation rules are employed to identify attacks. The detection system comprises monitor nodes (sensors and actuators) that monitor other nodes (sensors and actuators) or a neighboring system (unmanned air vehicle) that monitors another trusted system (unmanned air vehicle). The monitoring system evaluates the behavior of the monitored system against a set of predefined system behavior and transition rules, and identifies a violation as a potential attack.

Sawada et al. [12] and Harshe et al. [3] have proposed cyber-physical system security solutions that use local (backup) controllers, which kick in when remote (central) controllers are compromised or are unavailable. The central controllers usually optimize the networked control system to yield high performance while the local controller guarantees the minimum performance requirements of the logical subsystem. The security solutions continuously evaluate control signals received from the central controller against the physical system and switches to the backup controller when a violation is observed.

A security solution for a cyber-physical system must be designed to comprehend and respond to the unique process behavior of the system. The solutions discussed above do not directly address data manipulation attacks on a power substation interlocking process, but they do provide a useful starting point for reasoning about the security of cyber-physical systems.

3. Substation Interlocking

Switchgear implement protection and control functions that are triggered in response to system guard conditions, automation and optimization functions or by human intervention. Substations are equipped with switchgear devices that are independently controlled and perform functions such as fault isolation, sectionalization, and over-current and over-voltage protection. The types of switchgear used in substations include isolator switches, contactor switches, earthing switches and circuit breakers.

3.1 Substation Switching

The IEC 61850 Standard recommends that switchgear be triggered by intelligent electronic devices that implement circuit breaker (XCBR) or circuit switch (XSWI) logical nodes at the process level. In turn, the circuit breaker and circuit switch logical nodes are controlled by intelligent electronic devices that implement protection and control functions such as time over-voltage protection (PTOV), instantaneous over-current protection (PIOC) and switch controller (CSWI). The first letter of a logical node name is used as a group identifier for logical nodes with similar functions. For example, the first "I" in "IHMI" (human-machine interface) identifies IHMI as belonging to interface group I.

Figure 1 shows a typical example of the operation sequence of an IEC 61850 substation interlocking function discussed in [8]. Human experts create interlocking rules and feed them to the system via the human-machine interface (IHMI). In Message 1, the interlocking function (CILO) imports the rules, validates the state of all the switchgear devices (via Messages 3, 4 and 8), and waits for a request from the switch controller (CSWI). In Message 2, the human controller issues a switch OPEN command to the switch controller and, in turn,

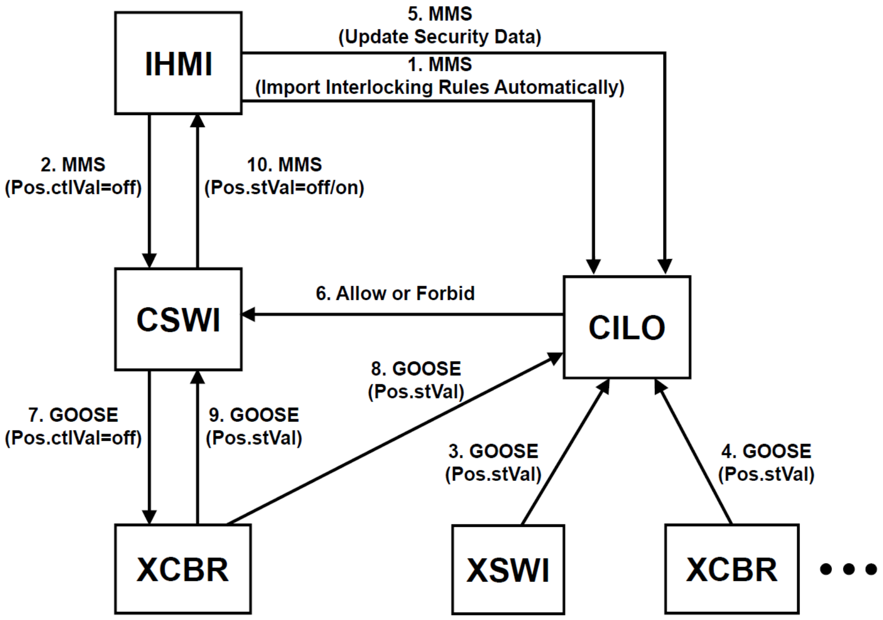


Figure 1. IEC 61850 CILO-controlled switchgear operation [8].

the switch controller requests the interlocking function to check if the execution of the command violates an interlocking rule. In Message 6, the interlocking function responds with an allow if no rule is violated and a forbid otherwise. In Message 7, the switch controller proceeds with a switch OPEN command if an allow response was received by instructing the circuit breaker/circuit switch (XCBR/XSWI) to OPEN. In Message 9, the circuit breaker/circuit switch notifies the switch controller about the failure or success of the operation and, in turn, the switch controller notifies the human-machine interface of success or failure. Finally, in Message 8, the circuit breaker notifies the interlocking function of the state change if any. As described in [8], GOOSE update messages are protected with a keyed-hash message authentication code (HMAC). From time to time, the circuit breaker and circuit switch are expected to send status messages to the interlocking function to ensure that the state information maintained by the interlocking function correctly reflects that of the physical switchgear devices.

3.2 Interlocking Function Operation

The IEC 61850 Standard refers to substation automation functions as logical nodes. The interlocking function logical nodes (LNs), which are implemented at the station level or bay level, contain the rules that govern all valid switchgear

Table 1. Valid configurations of the switchgear devices in the testbed.

Configuration	**CS1**	**CS2**	**CB**	**IS**	**ES**
1	0	0	0	0	0
2	0	1	0	0	0
3	0	1	1	0	0
4	0	1	1	1	0
5	0	0	0	0	1
6	0	0	0	1	1
7	0	0	1	0	1
8	0	0	1	1	1
9	1	0	0	0	0
10	1	0	1	0	0
11	1	0	1	1	0

configurations, the current states of switchgear devices and the transition sequences. Based on the interlocking rules imported from the human-machine interface, the interlocking function generates the valid configuration table and transition sequences.

In the testbed used in this research, a single bay substation was implemented using two power sources. The testbed consisted of five switchgear devices, one earthing switch (ES), two contactor switches (CS1 and CS2), one isolator switch (IS) and one circuit breaker (CB). The interlocking function had the eleven valid switchgear configurations shown in Table 1. A zero indicates that a switchgear is in the OPEN position and a one indicates that the switchgear is the CLOSE position.

Algorithm 1 : Validate switch controller request.

```
1: procedure VALIDATECSWIREQUEST(request)
2:     temp = FALSE
3:     if request $\neq$ NULL then
4:         n = getNoSwitch(request)
5:         curConfig = getCurConfig()
6:         newConfig = getNewConfig(request)
7:         temp = isValid(newConfig, validConfigTable)
8:         if n == 1 then
9:             RETURN temp
10:        end if
11:        CALL transSeqFn(request,curConfig)
12:    end if
13:    RETURN temp
14: end procedure
```

The behavior of the interlocking function is described using the validate switch controller request algorithm (Algorithm 1). The algorithm is executed whenever a new request is received. In Line 4 of the algorithm, the interlocking function checks for the number of switchgear devices affected by the request, Line 5 obtains the current switchgear configuration and Line 6 computes the new configuration based on the change request. In Line 7, the interlocking function checks that the request does not violate any interlocking rules and returns either TRUE or FALSE. If the number of switchgear devices that would be affected by the request is no more than one and the new configuration is valid, then the interlocking function returns TRUE to the switch controller, implying that the change is allowed. If multiple switchgear devices are affected by the request, then the algorithm proceeds to Line 11 and invokes the transition sequence function. The transition sequence specifies the order in which the switchgear affected by the change request should be implemented. An execution interval of 1 ms to 10 ms is typically allowed for concurrent switchgear operations.

3.3 Substation Communication Protocols

IEC 61850 specifies the use of the generic object oriented substation event (GOOSE) and sampled value (SV) protocols for power substation communications. GOOSE and SV are fast data transfer protocols that execute in the data link layer and are used in the substation local-area network to control, report events and transmit measured values.

GOOSE Protocol. The GOOSE protocol specified in the IEC 61850-8-1 Standard is a multicast/broadcast protocol that uses a publisher-subscriber communications model for sending and receiving data between intelligent electronic devices. Bay-level intelligent electronic devices use the GOOSE protocol to report switch state changes (ON and OFF). The GOOSE protocol uses the status number (StNum) and sequence number (SqNum) to distinguish between state change events and re-transmissions. StNum starts from one and is incremented for every state change (OPEN or CLOSE) event. SqNum, which starts from zero, indicates re-transmissions of a previous notification. For example, the first status change in the switchgear has StNum = 1 and SqNum = 0. The switchgear keeps broadcasting its state information at time intervals less than 60 s until a new state is recorded. For each re-transmission, StNum remains the same but SqNum is incremented.

SV Protocol. The SV communications protocol defined in IEC 61850-9-2 is a multicast/broadcast protocol that uses a publisher-subscriber communications model to receive data streams of sampled values from sensors in a substation. The SV protocol is primarily used to send voltage and current measurements obtained from current and voltage sensors to all the subscribing intelligent electronic devices. The protocol uses the sample count (SmpCnt) field in the SV protocol data unit to indicate every new sample and the sample rate (SmpRate) to specify the number of samples per second. SmpCnt is incremented for every new sample and there are no re-transmissions.

4. Attack Description

The interlocking function translates switchgear configuration rules into a valid configuration table as shown in Figure 1. A valid configuration is a vector that indicates the permitted state of all the switchgear devices at a given instant. The valid configuration table is the collection of all possible valid configurations.

Let s be the number of switchgear devices in a substation and let $C \in \{0,1\}^s$ correspond to all possible switchgear configurations. Let $\vec{C}'$ be a valid configuration and n be the total number of valid configurations. Then, the valid configuration table T is the set:

$$T = \{\vec{C}'_1, \vec{C}'_2, \cdots, \vec{C}'_n\} \tag{1}$$

A state change request τ_{i+1} is allowed to change the interlocking function configuration state from $\vec{C}'_i$ to $\vec{C}'_j$ if and only if:

$$F : \vec{C}'_i \times \tau_{i+1} \Rightarrow \vec{C}'_j \in T \tag{2}$$

where F is the transition mapping function and $1 \leq i, j \leq n, i \neq j$. Whenever a change request is successfully executed by a circuit breaker or circuit switch, a status update message is sent to the interlocking function, which updates its current configuration state from $\vec{C}'_i$ to $\vec{C}'_j$.

Process level communications are time-critical because IEC 61850 requires a delay of no more than 4 ms in the transmission of GOOSE and SV messages. This requirement hinders the implementation of an encryption-based security solution. IEC 61850 does not recommend the encryption of GOOSE and SV messages and mandates that encryption-based message integrity checks may be used for GOOSE only if they meet the 4 ms time requirement. Intelligent electronic devices in the process local-area network depend on the timestamps, StNum and SqNum for GOOSE messages, and SmpCnt for CV messages to detect data manipulations. Tebekaemi and Wijesekera [13] have demonstrated a successful GOOSE attack when the attacker has physical access to the process local-area network. Attacks on SV messages are more difficult to detect when the SmpRate has a high value because it is harder to predict the next SmpCnt value.

- **Scenario 1: Dropped Update Message:** This scenario assumes that the attacker has access to the process local-area network at the substation and can block GOOSE update messages to the interlocking function. When a status change request is received by the switch controller, it queries the interlocking function to validate the request. The interlocking function validates the request against the current system state $\vec{C}'_i$ and instructs the switch controller to execute the request. The switch controller executes the request and broadcasts its new status, which is blocked by the attack. Since no update message is received by the interlocking function, it still believes that the system is in state $\vec{C}'_i$ instead of the new state $\vec{C}'_j$. The current state of the interlocking function no longer reflects the actual state of the physical system. Although the interlocking function and physical system may still have valid configurations, any new change request results in F using the wrong input $\vec{C}'_i$ instead of $\vec{C}'_j$.

- **Scenario 2: Corrupt Update Message:** This scenario assumes that the attacker has access to the process local-area network and can modify GOOSE update messages, inject new GOOSE packets or arbitrarily send GOOSE update messages. The attacker may be able to deceive the interlocking function to believing that an update has occurred and that its current state should be updated, causing the interlocking function to update its current state to $\vec{C}'_j$ while the system remains in $\vec{C}'_i$.

Scenarios 1 and 2 poison the configuration state of the interlocking function. If the malicious update is a valid configuration state, then no flag is raised and the attack goes unnoticed by the intelligent electronic device. The result could be disastrous if an attacker can successfully place the interlocking function in an invalid state. For example, according to Table 1, CS1 and CS2 cannot be in the CLOSE position at the same time. Assuming that the bay is disconnected autonomously for maintenance purposes, both CS1 and CS2 must be OPEN before ES can be in the CLOSE position. The interlocking function configuration table is poisoned to show that both CS1 and CS2 are OPEN and, thus, the interlocking function proceeds to validate an ES CLOSE request when either CS1 or CS2 is in the CLOSE position. Executing the request increases the current astronomically because the voltage is suddenly reduced to approximately zero – this could damage equipment and cause a fatal accident. Row 14 in Table 2 shows that such a request increases the current to 850 times its nominal value.

5. Proposed Solution

Electrical equipment exhibits unique physical attribute properties when triggered by ON/OFF commands; the properties manifest themselves as transients, steady state changes, and amplitude and frequency changes in the voltage and current waveforms. Observations of disturbances in the voltage and current waveforms can provide direct power feedback pertaining to physical- and cyber-controlled events. It is possible to monitor and detect ON/OFF events involving electrical equipment and trace the events to the originating equipment using their transient states, steady states or frequency changes of the measured voltages and currents [4, 10]. Similar techniques have been used to detect and locate faults in power systems [1, 9, 11].

Current and voltage sensors are used in substations to provide information about the voltage and current of the supplied electric power, which is used to drive substation functions such as voltage/voltage-ampere reactive (VAR) control, frequency control, power quality control, and over-voltage and over-current protection. Current and voltage sensors give information about the portions

Table 2. Voltage and current measurements during switchgear ON/OFF operations.

Device	**Position**	**Type**	**Sensor 1**	**Sensor 2**	**Sensor 3**
CS	ON	V	1.001	1.001	0.465
		A	0.528	0.525	1.229
	OFF	V	0.195	0.195	0.09
		A	0.103	0.102	0.24
CB	ON	V	1.001	1.001	0.465
		A	0.529	0.525	1.229
	OFF	V	1	0.102	0.047
		A	0.107	0.053	0.125
IS	ON	V	1.001	1.001	0.465
		A	0.528	0.525	1.229
	OFF	V	1	1	0.009
		A	0.066	0.012	0.023
ES	ON	V	0	0	0
		A	850	0	0
	OFF	V	1.001	1.001	0.465
		A	0.528	0.525	1.229

of the system that are energized. Intelligent electronic devices use this information to determine switchgear positions (OPEN or CLOSE) at any instant. Switchgear events are also observable via the electrical waveforms they generate; switching a device ON or OFF generates transients that appear as spikes in its waveform and steady state amplitude changes as shown in Figure 2 (note that p.u. = measured value/nominal value). Monitoring switchgear events provides useful information about the times when events occur and the originating switchgear devices, which help detect illegal switchgear manipulations.

5.1 Switchgear Event Detection

Event detection algorithms compare the measured value of a signal to a reference value; when a significant difference is detected, an event of interest is declared to have occurred. In order to increase the accuracy of event detection in a power signal, a change event is computed based on the properties of the signal over a time frame called the event detection window. This helps reduce the effects of noise in the signal and the false event detection ratio.

In the initial simulated testbed, the electrical noise was normally distributed, which may not be the case in an actual substation. The detection algorithm employed was a simple mean change detector that compared the detection window w_i against the pre-event window w_{i-1}. If $n = |w|$, $w_i = x_1, x_2, \cdots, x_n$ and $w_{i-1} = y_1, y_2, \cdots, y_n$, then:

$$|\frac{\sum_{i=1}^{n} x_i - \sum_{i=1}^{n} y_i}{n}| > \xi \tag{3}$$

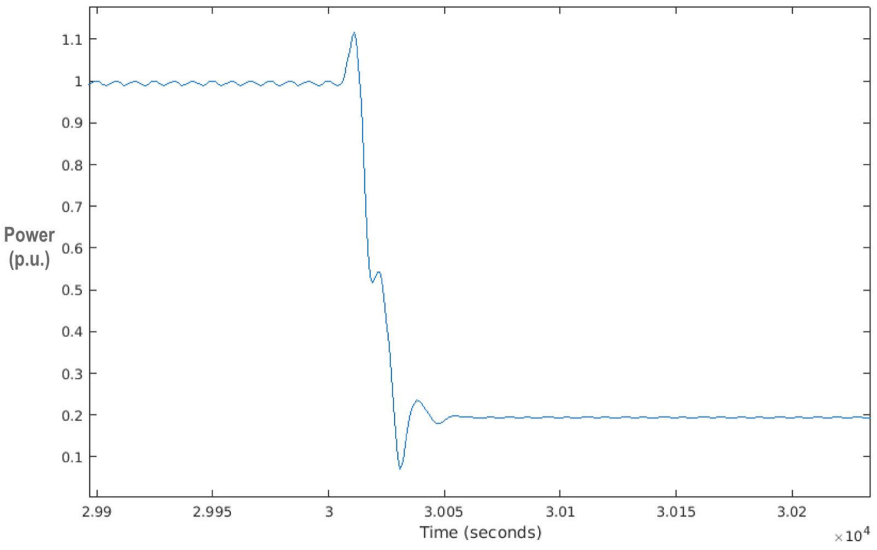


Figure 2. Transient and steady state voltage during switch CLOSE operations.

indicates the occurrence of an event, where μ is the mean value, x_i and y_i are sample points of the DC component of the signal and ξ is a predetermined threshold value.

Voltage and current signals usually contain noise caused by imperfections in electrical equipment and devices, thermal conditions, electrostatic interference, electromagnetic interference, radio frequency interference and cross-talk. Noise in measured signals can cause detection systems to increase the number of false positives or completely misdetect events. To address the effects of noise, the sensitivity of the detection system (threshold) must be set to achieve a high detection rate (e.g., 100%) given the noise level and the lowest possible false positive rate within an acceptable response time. A more sensitive threshold enables the system to detect minor events and respond quickly, but with less accuracy; in contrast, a less sensitive threshold causes the system to miss minor events and respond slowly, but with better accuracy.

This research assumes that the measured voltage and current signals contain noise, and employs the change detection method described in [4]. Specifically, the noise e_i is assumed to be a continuous white Gaussian process so that $x'_i = x_i + e_i$ and $y'_i = y_i + e_i$. The detection threshold $\xi = \chi^2_{\alpha,k-1}$ is a chi-square goodness of fit test with a confidence interval of $(100-\alpha)\%$ and detection sensitivity factor k. An event is detected when:

$$\sum_{i=1}^{n} \frac{(x'_i - y'_i)^2}{y'_i} > \xi \tag{4}$$

Table 3. Switchgear event truth table.

Close	**Open**	**Type**	**Sensor 1**	**Sensor 2**	**Sensor 3**
	CS	V	0	0	0
		A	0	0	0
CS	CB	V	1	0	0
		A	0	0	0
CB	IS	V	1	1	0
		A	0	0	0
IS		V	1	1	1
		A	1	1	1
ES		V	0	0	0
		A	1	0	0

The detection threshold can be pre-computed and fixed if the noise level is expected to be the same; alternatively, it can be computed dynamically during system operation if the noise level is expected to change.

5.2 Switchgear State Identification

The switchgear state detection process involves the determination of sections of the bay that are energized based on the sensor measurements. The sensor measurements are mapped using the switchgear state truth table (Table 3) to identify which switchgear devices are in the CLOSE and OPEN positions. The switchgear truth table is preconfigured and contains the combination of high and low voltage and current values measured by all the sensors in the testbed that map to the ON or OFF states of switchgear in the substation. The switchgear state identification serves two purposes: (i) to attribute a detected event to the originating switchgear; and (ii) to validate the state of the physical system during the interlocking function request validation operation. Table 2 shows the measured values of each switch when it is the CLOSE and OPEN positions. The information in this table is used to generate the switchgear event truth table shown in Table 3. The event truth table is used to predict which switchgear devices are in the CLOSE and OPEN positions based on the sensor measurements. In the event truth table, a zero indicates that the measured value from a given sensor is low while a one indicates that the value is high.

5.3 Interlocking Function Security Controller

Switchgear status update information is sent from the circuit breaker or circuit switch to the interlocking function in the form of GOOSE packets over the process local-area network. The IEC 61850 Standard also allows sampled voltage and current measurements to be sent from the merging units to intelligent electronic devices via the process local-area network in the form of SV packets. The interlocking function security controller uses the SV messages to detect

Algorithm 2 : Check for modified GOOSE updates.

```
 1: procedure isMessageModified(gooseUpdate)
 2:     if stNumChange(updateMsg) then
 3:         powFeedback == getPowFeedback()
 4:         if updateMsg.stVal == powFeedback.val then
 5:             if updateMsg.time ≈ powFeedback.time then
 6:                 return FALSE
 7:             end if
 8:         end if
 9:         return TRUE
10:     end if
11: end procedure
```

changes in the waveforms and obtains the direct power feedback for switchgear events. The security controller uses GOOSE and SV messages, which function as two independent sources, to validate the correct states of switchgear devices.

Algorithm 2 describes the high-level behavior of the proposed interlocking function security controller. The algorithm is invoked whenever a GOOSE update message (updateMessage) is received from a switchgear (circuit breaker or circuit switch). In Line 2, the security controller checks if the update message is a retransmission or a new event notification. If the update message is a new event notification, then the security controller obtains the power feedback information from the SV messages in Line 2. In Line 3, the most recent measurements from the sensors are obtained and are used to estimate the current states of the switchgear devices. In Lines 4 and 5, the GOOSE update message and the power feedback information are compared to check if the reported event is consistent and is within the same time frame. The GOOSE update and SV feedback messages arrive at the interlocking function at slightly different times, so the time values are approximate and a check is performed to see if both messages arrive within an acceptable time frame. An inconsistency in the reported event or time frame implies that there is a high probability that the GOOSE update message has been modified.

Algorithm 3, which runs continuously as a background process, checks for changes in voltage and current waveforms indicated by the SV messages. In Line 3, if a significant change is detected, the security controller proceeds to obtain the change information in Line 4. The reported change is checked in Line 5 to ascertain if the event is the result of a state change and returns TRUE if the event is caused by a switchgear. If the event is the result of a switchgear operation and no GOOSE update message is received, then there is a high probability that the update message has been blocked.

6. Implementation and Results

Power substations have bays that connect feeders to power sources. Each bay has switchgear that implement the bay-level protection and control func-

Algorithm 3 : Check for missing GOOSE updates.

```
procedure isUpdateMissing
    while TRUE do
        if eventDetected() then
            powFeedback == getPowFeedback()
            if stChange(powFeedback) == TRUE then
                return TRUE
            end if
        end if
    end while
end procedure
```

tions. The IEC 61850 Standard does not have a preference regarding where the interlocking function should be implemented (i.e., station level or bay level); instead, it leaves the decision to substation designers. At the station level, the interlocking function has to maintain the state and configuration information of switchgear in all the bays in the substation. Thus, for a substation with n bays and x switchgear devices per bay, the interlocking function maintains $n \times x$ switchgear states with 2^{nx} possible switchgear configurations. The configuration table grows rapidly as n and x increase, and can easily overwhelm the intelligent electronic device.

The proposed solution relies on SV messages received by the interlocking function from merging units. SV messages transmitted as multicast packets provide a continuous stream of currents and voltages sampled at high rates. In the case of a substation with multiple bays, the interlocking function has to process the continuous streams of multicast packets from all the merging units distributed across the bays in the substation. This can cause congestion in the station local-area network and may also lead to the failure of the network interface controller of the intelligent electronic device with the interlocking function. For these reasons, it is recommended that the interlocking function be implemented at the bay level and, in fact, the proposed solution is designed for a bay-level interlocking function.

6.1 Implementation

Tebekaemi and Wijesekera [13] have designed and implemented a substation simulation testbed. Certain modifications were made to this testbed to support the substation interlocking function discussed in this chapter. Figure 3 shows a schematic diagram of the modified testbed with three virtual machines (VMs) running on a VMware ESXi server and a MacBook Pro computer.

Power System (VM1). The substation was simulated on the Intel core i7 MacBook Pro computer with a 2.5 GHz processor, 16 GB RAM and 512 GB SSD. The substation was a single-bay step-down station created with Matlab/Simulink that incorporated two contactor switches (CS1 and CS2), a groun-

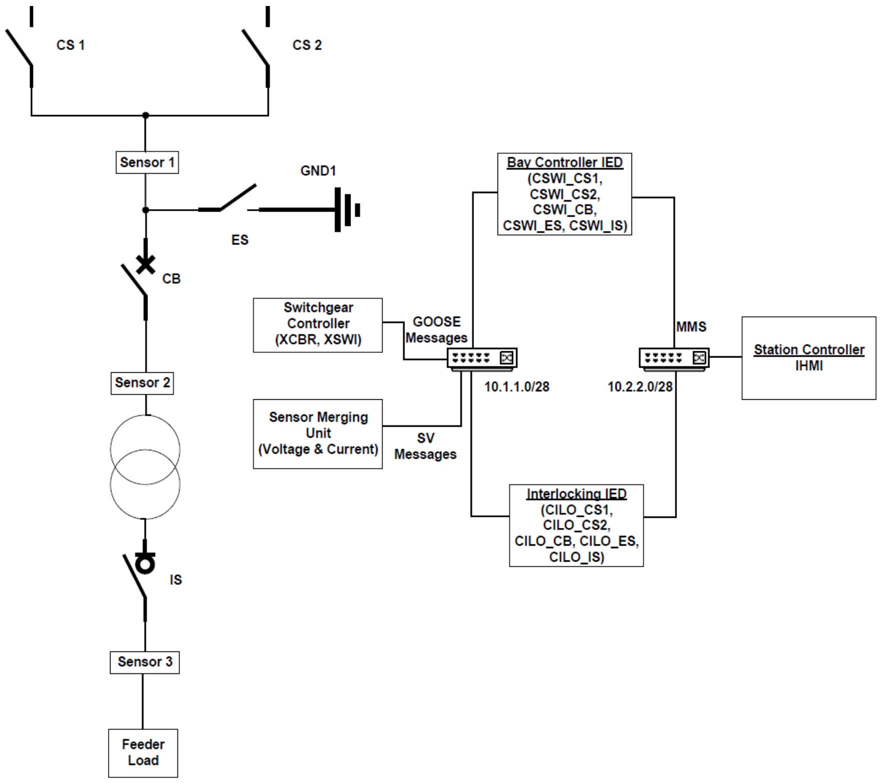

CS 1
CS 2
Sensor 1
GND1
ES
CB
Sensor 2
IS
Sensor 3
Feeder Load
Bay Controller IED (CSWI_CS1, CSWI_CS2, CSWI_CB, CSWI_ES, CSWI_IS)
Switchgear Controller (XCBR, XSWI)
GOOSE Messages
MMS
Station Controller IHMI
10.1.1.0/28
10.2.2.0/28
Sensor Merging Unit (Voltage & Current)
SV Messages
Interlocking IED (CILO_CS1, CILO_CS2, CILO_CB, CILO_ES, CILO_IS)

Figure 3. Implementation schematics of the substation testbed.

ding/earthing switch (ES), an isolator switch (IS) and a circuit breaker (CB). Voltage and current measurements were obtained from three sensors installed at different locations in the bay.

Virtual Intelligent Electronic Devices. The following virtual intelligent electronic devices were incorporated in the modified testbed:

- **Merging Unit and Switchgear Controller (VM1):** The merging unit and switchgear controller were implemented as standalone C/C++ applications based on the IEC 61850 Standard. These applications also ran on VM1 (Ubuntu 14.04.4LTS with two core processors, 2 GB RAM and 20 GB HDD). The merging unit and switchgear controller communicated with the simulated substation using UDP ports. The merging unit collected sampled measurements from all three sensors, timestamped them and broadcasted the values using the SV protocol. The switchgear

Algorithm 4 : Interlocking rules.

```
1: if CS2==CLOSE then DENY CS1 CLOSE
2: end if
3: if CS1==CLOSE then DENY CS2 CLOSE
4: end if
5: if ES==CLOSE then DENY CS1 CLOSE
6: end if
7: if ES==CLOSE then DENY CS2 CLOSE
8: end if
9: if CS1==CLOSE then DENY ES CLOSE
10: end if
11: if CS2==CLOSE then DENY ES CLOSE
12: end if
```

controller relayed the OPEN/CLOSE GOOSE commands from the bay controller to the appropriate switchgear devices.

- **Bay Controller IED (VM2):** The bay controller intelligent electronic device was implemented as a C/C++ application based on the IEC 61850 Standard that executed on VM2 (Ubuntu 14.04.4LTS with two core processors, 2 GB RAM and 20 GB HDD). The bay controller intelligent electronic device comprised five switch controller logical nodes (CSWI_CS1, CSWI_CS2, CSWI_ES, CSWI_CB and CSWI_IS), each corresponding to a switchgear device in the substation.

- **Interlocking IED (VM2):** The interlocking intelligent electronic device comprised five interlocking function logical nodes (CILO_CS1, CILO_CS2, CILO_ES, CILO_CB and CILO_IS), each of which maintained the state information of the corresponding switchgear device in the testbed. The interlocking intelligent electronic device executed the data manipulation detection algorithms and maintained the switchgear configurations and transition rules. The interlocking rules specified in Algorithm 4 were implemented in the interlocking intelligent electronic device based on Table 1.

Attacks. The following attacks were executed on the testbed:

- **Blocked GOOSE Update:** This attack requires access to the process local-area network and blocks GOOSE update messages from reaching their destinations. The attack was simulated by configuring the controllers not to send update messages after a state change operation.

- **Modified GOOSE Update:** This attack requires access to the process local-area network. GOOSE update messages are broadcast in plaintext to all the subscribing intelligent electronic devices. The TCPDump tool was used to capture network traffic and replay it unmodified using the

Table 4. Performance of the interlocking function with and without security.

	No Security	**Security (No Noise)**	**Security (Noise)**
Replay	✓	✓	✓
Modified Replay	×	✓	✓
Missing Update	×	✓	✓
Time (ms)	1.351	1.446	57.955

TCPReplay tool. Modified network traffic was transmitted using the Scapy traffic manipulation tool.

6.2 Results

The simulation was first executed with the interlocking function security controller deactivated. The interlocking intelligent electronic device used the GOOSE StNum, SqNum and timestamp fields to detect replay attacks. However, if StNum, SqNum and timestamp were modified to mimic a new update message, it was possible to successfully modify the configuration state of the interlocking intelligent electronic device. In the case of missing and blocked update messages, the interlocking intelligent electronic device had no way of detecting the events and easily entered an inconsistent state. When the security controller was activated, the modified replay attacks and the missing update messages were detected. The security controller always validated the GOOSE update messages with the power feedback SV messages to ensure that the GOOSE update messages were valid. Also, by continuously listening to changes in the physical system, the security controller was able to detect configuration changes observed by the power feedback SV messages but not reported by the GOOSE update messages.

Table 4 summarizes the performance of the interlocking function with and without the security controller. The times (in ms) were measured from the instant the control operation was initiated by the switch controller to the instant the interlocking intelligent electronic device updated its configuration state.

7. Conclusions

Interlocking is a critical substation automation function that ensures the safety of human lives and power equipment, and the reliability and resilience of power systems. As a result, interlocking functions are high value targets for malicious entities. However, power systems have tight timing requirements that prevent the use of cryptographic techniques and tools to protect network traffic and data. This requires the design and implementation of other protection mechanisms for power systems.

This chapter has presented a novel method for detecting data manipulation attacks on interlocking functions in power distribution substations. The method relies on knowledge about the behavior of the physical system and integrates it in conventional intrusion detection mechanisms. The method is applicable to other power system components and functions that involve automated switching functions, including distribution bus networks and ship power systems. The research also demonstrates that integrating knowledge about the physical behavior of a cyber-physical system in cyber security controls is vital to enhancing system reliability and resilience.

References

[1] A. Al-Mohammed and M. Abido, An adaptive fault location algorithm for power system networks based on synchrophasor measurements, *Electric Power Systems Research*, vol. 108, pp. 153–163, 2014.

[2] E. Colbert, D. Sullivan, S. Hutchinson, K. Renard and S. Smith, A process-oriented intrusion detection method for industrial control systems, *Proceedings of the Eleventh International Conference on Cyber Warfare and Security*, pp. 497–500, 2016.

[3] O. Harshe, N. Chiluvuri, C. Patterson and W. Baumann, Design and implementation of a security framework for industrial control systems, *Proceedings of the International Conference on Industrial Instrumentation and Control*, pp. 127–132, 2015.

[4] Y. Jin, E. Tebekaemi, M. Berges and L. Soibelman, Robust adaptive event detection in non-intrusive load monitoring for energy aware smart facilities, *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4340–4343, 2011.

[5] G. Koutsandria, V. Muthukumar, M. Parvania, S. Peisert, C. McParland and A. Scaglione, A hybrid network IDS for protective digital relays in the power transmission grid, *Proceedings of the IEEE International Conference on Smart Grid Communications*, pp. 908–913, 2014.

[6] R. Liu, C. Vellaithurai, S. Biswas, T. Gamage and A. Srivastava, Analyzing the cyber-physical impact of cyber events on the power grid, *IEEE Transactions on Smart Grid*, vol. 6(5), pp. 2444–2453, 2015.

[7] R. Mitchell and I. Chen, Adaptive intrusion detection of malicious unmanned air vehicles using behavior rule specifications, *IEEE Transactions on Systems, Man and Cybernetics: Systems*, vol. 44(5), pp. 593–604, 2014.

[8] J. Pan, B. Duan, C. Qiu and G. Li, Research on interlocking CILO based on IEC 61499/62351, *Proceedings of the Asia-Pacific Power and Energy Engineering Conference*, 2012.

[9] P. Nayak, A. Pradhan and P. Bajpai, A fault detection technique for a series-compensated line during power swing, *IEEE Transactions on Power Delivery*, vol. 28(2), pp. 714–722, 2013.

[10] A. Rababaah and E. Tebekaemi, Electric load monitoring of residential buildings using goodness of fit and multi-layer perceptron neural networks, *Proceedings of the IEEE International Conference on Computer Science and Automation Engineering*, vol. 2, pp. 733–737, 2012.

[11] M. Riera-Guasp, J. Antonino-Daviu and G. Capolino, Advances in electrical machine, power electronics, and drive condition monitoring and fault detection: State of the art, *IEEE Transactions on Industrial Electronics*, vol. 62(3), pp. 1746–1759, 2015.

[12] K. Sawada, T. Sasaki, S. Shin and S. Hosokawa, A fallback control study of networked control systems for cybersecurity, *Proceedings of the Tenth Asian Control Conference*, 2015.

[13] E. Tebekaemi and D. Wijesekera, Designing an IEC 61850 based power distribution substation simulation/emulation testbed for cyber-physical security studies, *Proceedings of the First International Conference on Cyber-Technologies and Cyber-Systems*, pp. 41–49, 2016.

Chapter 4

NETWORK FORENSIC ANALYSIS OF ELECTRICAL SUBSTATION AUTOMATION TRAFFIC

Megan Leierzapf and Julian Rrushi

Abstract The computations and input/output values of intelligent electronic devices that monitor and operate an electrical substation depend strongly on the state of the power system. This chapter presents an approach that correlates the physical parameters of an electrical substation with the network traffic that intelligent electronic devices send over a substation automation network. Normal network traffic in a substation automation network is modeled as a directed, weighted graph, yielding what is referred to as a model graph. Similar graph modeling is performed on unknown network traffic. The research problem of determining whether or not unknown network traffic is normal involves a subgraph isomorphism search algorithm. Normal network packets in unknown network traffic form a graph that is a subgraph of the model graph. In contrast, malware-generated network packets present in unknown network traffic produce a graph that is not a subgraph of the model graph. Time series analysis of network traffic is performed to estimate the weights of the edges in the graphs. This analysis enables the subgraph isomorphism search algorithm to find structural matches with portions of the model graph as well matches with the timing characteristics of normal network traffic. The approach is validated using samples drawn from recent industrial control system malware campaigns.

Keywords: Industrial control systems, malware, digital forensics

1. Introduction

As the recent BlackEnergy and Dragonfly malware campaigns have demonstrated, industrial control systems (ICSs) are continually targeted by malware that spies on or disrupts industrial processes, including those in the electric power grid. Digital forensic investigations of incidents involving industrial control systems are, therefore, of considerable importance [1, 8]. This chapter

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 63–78, 2017.
https://doi.org/10.1007/978-3-319-70395-4_4

focuses on investigations of intrusions in intelligent electronic devices (IEDs), which are control devices deployed in electrical substations.

The principal contribution is a network forensic approach that leverages graph theory and time series analysis to identify malicious packets in substation automation network traffic. A model graph is used to comprehensively model intelligent electronic devices, their monitoring and protection functions, state of the electrical substation and network traffic that the intelligent electronic devices exchange with each other while performing their functions. The time series analysis helps compute the weights of the edges in the model graph, which characterize the relationships of network traffic with time. The same graph modeling approach is employed to construct graphs that model real network traffic in a substation automation network. Custom Python scripts are used to analyze packet capture (`pcap`) files and help construct adjacency matrices – specifically, concrete graphs – based on captured network packets.

Concrete graphs developed from normal network traffic are subgraphs of the model graph, assuming, of course, that the vertices, edges and weights in all the graphs are computed accurately. Experiments with emulated exploits and malware, including samples used in the Dragonfly industrial control system malware campaign (Havex), demonstrate that concrete graphs developed from their network traffic do not correspond to subgraphs of the model graph. This feature is leveraged to discern malicious packets in large traffic captures from substation automation networks. The novel contribution is a network forensic approach that considers the topological and timing characteristics of intelligent electronic devices induced by the power system.

2. Problem Statement

The research problem involves the dissection and forensic analysis of network traffic from an electrical substation network in order to reliably separate network packets generated by malware from packets generated by intelligent electronic devices during their normal monitoring and protection functions.

The inputs comprise the following data:

- A log file containing the physical parameter values of an electrical substation of interest over time as measured by various sensors. Examples of physical parameters include voltages, currents, temperatures, phases, frequencies and amplitudes.

- A network capture file in `pcap` format that contains traffic captured from the electrical substation network. Each packet is timestamped, as it is commonly the case when network traffic is captured with a tool such as Wireshark.

The outputs comprise the following data:

- A file in `pcap` format that contains all the network packets that the forensic analysis has determined to have been generated by malware.

- A second `pcap` file that contains only the network packets that the forensic analysis has determined to have been generated by intelligent electronic devices as part of their normal operation.

The proposed network forensic approach leverages the logical coupling between intelligent electronic device functions and the state of the power system. This coupling is inherent in an electrical substation. The analysis is conducted based on the IEC 61850 network communications standard.

Intelligent electronic devices do not exercise a directional comparison protection function based on so-called PDIR logical nodes if the forward and reverse-looking instantaneous directional overcurrent or distance elements at each line terminal do not activate the directional comparison scheme in use. If the directional comparison protection function is not exercised, then no network packets are exchanged, except for packets that carry periodic parameter measurements. Otherwise, if the parameters cause the activation of the directional comparison scheme, then the intelligent electronic devices generate network traffic that include commands that trip one or more circuit breakers.

A similar observation applies to the distance protection function for so-called PDIS logical nodes, directional overpower for so-called PDOP logical nodes and, in fact, all the other protection functions performed by intelligent electronic devices in an electrical substation.

Srivastav et al. [6] have discussed the regularities existing in industrial control network traffic. Unlike conventional information technology network traffic that fluctuates at high values, industrial control network traffic is relatively stable and somewhat predictable. The proposed forensic approach deciphers the relationships between the power system state and the network traffic generated by intelligent electronic devices. It engages these relationships to distinguish between normal and malicious traffic.

3. Graph Construction

This section describes the construction of the model and concrete graphs.

3.1 Model Graph Construction

The model graph formally describes normal network traffic in a substation automation network. Each vertex in the graph represents a pair comprising an intelligent electronic device and its monitoring or protection function. The model graph is directed and weighted. An edge from vertex A to vertex B represents network traffic between the intelligent electronic devices represented by vertices A and B, respectively. An intelligent electronic device can be the source or destination of network packets. However, the intelligent electronic device represented by vertex A is the initiator of the network communications; specifically, this device should have initiated the TCP three-way handshake by transmitting a packet with the TCP SYN flag set. In another scenario, the intelligent electronic device represented by vertex B could initiate network communications with the intelligent electronic device represented by vertex A.

However, all the network traffic in the associated TCP session is represented by an edge from vertex B to vertex A, regardless of which devices are the sources and destinations of specific packets in the session.

When the device represented by vertex A sends a UDP packet to the device represented by vertex B as it begins to exercise a function and, thus, possibly trigger the exchange of other UDP packets, then the edge goes from vertex A to vertex B. Each edge in the model graph has a weight that is determined via time series analysis. Every intelligent electronic device in an electrical substation has a set of monitoring and protection functions that it can perform; this increases the number of vertices present in the model graph. For example, if an electrical substation has just three intelligent electronic devices, each with the ability to execute 200 functions, there would be a total of 600 vertices in the model graph corresponding to this rather small electrical substation. As discussed later, knowledge of the specific monitoring and protection functions performed by an intelligent electronic device is of utmost importance.

Exploits and malware that affect an intelligent electronic device can only send or receive network traffic associated with the monitoring and control functions performed by the device. Otherwise, their malicious network packets would stand out and be trivial to spot.

Consider a situation where an intelligent electronic device is supposed to exercise a transformer protection function (and no other function) at time t_x for 4 ms. To conceal malicious network traffic, malware running on or targeting the intelligent electronic device could only send or receive traffic at time t_x, and typically not longer than 4 ms. If the intelligent electronic device in question is not supposed to exercise any functions at time t_x, then it would not send or receive any traffic at time t_x. As a result, the malicious network traffic would stand out.

An intelligent electronic device is a real-time machine, meaning that its reaction time is almost instantaneous. However, there is a minuscule delay in the transmission of its response to an anomalous event. The time taken for the intelligent electronic device to exercise its monitoring and protection functions is known; the expected time may be expressed as a range (e.g., 1 to 4 ms). However, if the network traffic continues to flow for 5 ms or longer, there is a high likelihood that some of the traffic was generated by malware. The malware traffic may not exhibit the typical patterns of electrical substation traffic. Thus, small differences in network activity could be very useful in identifying malicious network traffic.

Monitoring and protection functions are exercised by intelligent electronic devices under a specific set of physical parameter values in order to protect electrical substation equipment. The reporting of the parameter values by sensors is time critical because the intelligent electronic devices must perform their tasks in a timely manner [2]. An example protection function is the redirection of power. If a tree were to fall on a power line and cause a power outage, the protection function would redirect the power around the fallen line to deliver power to the affected customers. The power transformer protection function switches off a power transformer in an electrical substation when an internal fault is detected. Switching off the transformer has the effect of protecting it; if the power transformer were to be unprotected, it would explode in the worst-case scenario.

An internal fault is detected by comparing the primary and secondary currents. If the currents exceed the computed thresholds, then a fault is assumed and the transformer is switched off. Rahman and Jeyasurya [5] have experimented with several power transformer protection function algorithms; the curve fitting, rectangular transform, Fourier and finite impulse response algorithms were determined to be feasible for implementation in microprocessor-based transformer differential relays. For the power redirection and power transformer protection functions mentioned above to be exercised by intelligent electronic devices, it is necessary for the relevant sensors to report anomalous power system states, namely certain ranges of physical parameter values that characterize the physical processes of the substation.

The two protection functions are exercised by intelligent electronic devices only when the sensor values are in the stipulated ranges. This consideration applies to all monitoring and protection functions. Clearly the physical parameter values are not important to malware; as a result, malware does not adhere to the same structural and time restrictions that the substation protection functions must satisfy. Knowledge of some essential characteristics of the electrical substation of interest helps identify information that is used to create the model graph. Two pieces of critical information are the intelligent electronic devices and the specific functions they can exercise. These enable the determination of the vertices and edges in the model graph.

Two methods may be used to determine the model graph vertices and edges. The first method is to examine the configuration files of the intelligent electronic devices. The configurations indicate the functions that are exercised by the devices and the other intelligent electronic devices with which the devices exchange network traffic while exercising their functions. The drawback of this method is that significant manual effort is required to develop the vertices and edges corresponding to all possible functions exercised by all the intelligent electronic devices in a substation.

The second method is to write code that observes normal substation automation network traffic and learns the vertices and edges. The drawback of this method is that some functions are exercised very infrequently; thus, observing normal network traffic even for long periods of time may not be enough to discern the rarely-implemented functions. The proposed approach combines the two methods; it applies machine learning to network traffic only to compute the weights of the edges in the model graph.

Fiigure 1 shows a small portion of a model graph associated with an electrical substation. The edge weights are integer values to simplify the presentation. The substation has three intelligent electronic devices, 1, 2 and 3, each with two functions, A and B. Decisions about when to exercise the functions are made based on the reported values of ten physical parameters. Six vertices are

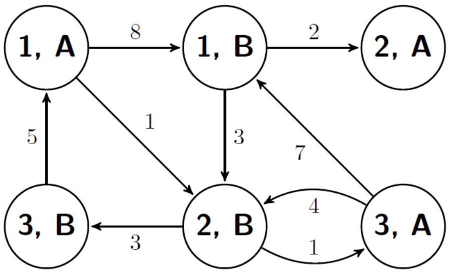


Figure 1. Sample model graph.

required to develop this portion of the model graph, one for each (intelligent electronic device, function) pair.

It was decided to construct the model graph in this manner because the intelligent electronic devices and functions had to be expressed explicitly. Using a single vertex for each intelligent electronic device was not an option because there can only be one directed edge from an origin vertex to a destination vertex. If this approach had been employed, the time series analysis would have represented all the network packets from an origin device to a destination device relative to all the monitoring and protection functions implemented by the two devices. As a result, it would not be possible to discern which functions were exercised and when they were exercised, making it exceedingly difficult to identify malicious network traffic.

In the model graph, physical parameters are directly used when defining each vertex; in fact, the monitoring and protection function modeled by a vertex depends entirely on the physical parameters. The reasoning is as follows: if the physical parameter values of a power system warrant a specific monitoring and control function being exercised by an intelligent electronic device, then a vertex exists and this vertex has an edge with a weight that connects it with another vertex. The model graph covers all the possible states of a power system where the states are expressed in terms of physical parameter value ranges. This is one more reason why the model graph can be extremely large.

3.2 Concrete Graph Construction

A concrete graph is constructed from the network packets contained in a `pcap` file. It is not known *a priori* whether the network packets are normal or malicious. Furthermore, if the `pcap` file contains malicious network packets, it is not known which packets are malicious and which packets are normal. The forensic analysis involves the construction of the concrete graph and the use of subgraph isomorphism search to determine the malicious and normal packets.

The vertices of the concrete graph are determined by examining the headers of the packets for the source and destination IP addresses. The IP addresses help identify the intelligent electronic devices that were senders or receivers

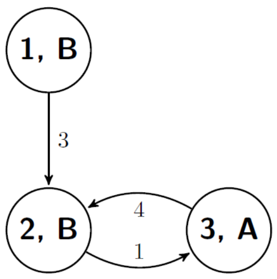


Figure 2. Sample concrete graph of normal activity.

of network traffic. The `pcap` files are timestamped, which enables the determination of the times when network traffic began to flow in the substation automation network.

The concrete graph is constructed by referring to a log file that contains the physical parameter values over time as measured by the sensors. The `pcap` file containing network traffic is then examined. The physical parameter values are used to determine the monitoring and protection functions that the addressed intelligent electronic devices exercised at specific times that the network traffic was recorded. The vertices in the concrete graph are marked based on the identified monitoring and protection functions and the IP addresses observed in the network traffic. The source and destination IP addresses identify the two intelligent electronic devices involved in a TCP session and the specific device that started the TCP session or initiated UDP communications. This information is sufficient to define all the edges of the concrete graph. The weights of the edges are computed via time series analysis as in the case of the model graph.

A concrete graph is compared against the model graph using subgraph isomorphism search. Checking for isomorphism between the concrete graph and a subgraph of the model graph relies on a graph theoretic technique (see, e.g., [7]). Graph isomorphism is a topic that has been researched for decades; a number of algorithms and library implementations for subgraph isomorphism search have been developed.

Figures 2 and 3 show two concrete graphs constructed from traffic in the same network as the partial model graph shown in Figure 1. The edge weights were again chosen to be integer values to simplify the presentation. The concrete graph in Figure 2 was determined to be a subgraph of the model graph; therefore, the captured network traffic represented by the concrete graph was not malicious.

On the other hand, in the case of the concrete graph in Figure 3, the weight of the edge from the origin vertex (1, A) to the destination vertex (2, B) is 5, which is not the same as in the partial model graph in Figure 1. This implies that the concrete graph is not a subgraph of the model graph and the network traffic is, therefore, determined to be malicious. The physical parameter values when

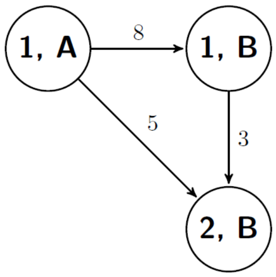


Figure 3. Sample concrete graph of malicious activity.

intelligent electronic devices exercise their monitoring and control functions provide additional clues to whether or not traffic is malicious. Specifically, if the physical parameter values are within the normal ranges and monitoring and control functions are still exercised (i.e., unnecessarily), then malicious network activity has occurred.

4. Time Series Analysis

This section demonstrates how time series analysis is used to compute the weights of the edges in the model and concrete graphs.

Intelligent electronic devices in an electrical substation perform computations and send/receive network traffic in a manner that exhibits strong dependencies on time and the power system state. The logical nodes in intelligent electronic devices do not act randomly. Each possible state of the power system demands that specific intelligent electronic devices exercise specific functions (i.e., manifest specific logical nodes) and these functions are carried out within specified time thresholds. The time constraint (threshold) imposed on a function demands that a logical node in an intelligent electronic device distributes all its function computations and I/O processing (and the traffic it transmits to other intelligent electronic devices) over time such that it satisfies the time threshold.

Experimental research in a laboratory environment has determined that the distribution of computations and I/O processing of a logical node in an intelligent electronic device over time is predictable. Certain fluctuations are observed in the times that network traffic sent by an intelligent electronic device on behalf of a logical node or logical device; however, the fluctuations are within predictable ranges. If this were not the case, intelligent electronic devices would send network traffic to each other too late, failing to meet the time thresholds of the overall monitoring and protection functions represented by the logical nodes. The relationship between network traffic and time for each monitoring and protection function exercised by an intelligent electronic device helps increase the granularity of the graph representation of substation automation network traffic. Thus, the crucial knowledge includes not only the direction and structure of the communications between one intelligent electronic device and another, but also the time characteristics of the network communications between the devices involved in a monitoring and protection function.

The weight of an edge from an intelligent electronic device A to another intelligent electronic device B is computed by developing a time series of all the network traffic that A sends to B and all the network traffic that B sends to A in response when a monitoring and protection function is exercised. If B is the initiator of the traffic that it sends to A, then this traffic and the traffic that A sends B in response are associated with a different time series. This is because, there is an edge from vertex A to vertex B and another edge from vertex B to vertex A.

The time series comprises the observations of network payloads sent at time t and excludes the packet headers. In other words, the observed data corresponds to a set of data of size x_t sent at time t. Note that these observations are made of network traffic sent by intelligent electronic devices over a substation automation network that is not under attack.

Thus, the proposed network forensic approach has anomaly detection characteristics in that there is a learning phase involving normal network communications between intelligent electronic devices. This learning phase requires a network capture file in the `pcap` format that is obtained by passively – and thus safely – sniffing substation automation network traffic (e.g., using Wireshark). The choice of timestamp precision is critical to the construction of the time series. Since the time window of most monitoring and protection functions ranges from zero to the time thresholds set for the functions (e.g., 4 ms), the timestamp precision in the proposed approach was set in the order of microseconds and the time unit was set to 100 ms. Thus, the time series of a monitoring and protection function would typically go from $t = 0$ to at most $t = 40$.

Time series with longer time windows may be constructed if there is enough network traffic. This is rarely the case for normal substation automation network traffic, but it is more likely for malware traffic. The time series constructed are discrete in that the sets of times t at which observations are made of packet payloads x_t are discrete.

The observations of network traffic over time are cumulative. If device A sends 500 bytes of data to device B at time a and device B responds with 300 bytes of data at time b, then $x_a = 500$ and $x_b = 800$. The time series is viewed according to the classical decomposition model:

$$x_t = w_t + s_t + y_t \tag{1}$$

where the w_t component of the observed x_t is the trend in the time series terms and is the factor that is leveraged by the proposed approach; s_t corresponds to the seasonality, which was noticeably absent in the time series encountered in the experiments; and y_t is the residual component in the time series terms.

The trend w_t is deterministic and is characteristic of a specific time series. This makes the trend a desirable differentiator. Given that the trend is specific to the time series representing normal network traffic for a monitoring and

Table 1. Testbed device functions and communications protocols.

Machine	**Function**	**Protocol**
SEL-487E-3	Transformer protection relay	IEC 61850
SEL-421-4	Automation, protection	IEC 61850
SEL-3555	Automation, data, HMI	IEC 61850
Windows Server	OPC server	OPC, IEC 61850
Windows Client	OPC client	OPC
Windows Machine	Development, testing	OPC, IEC 61850

protection function, its value over time would differ from the trend in a time series representing malicious network traffic. Note that the trend w_t of the time series representing normal network traffic is computed during the learning phase. The weight of the edge that goes from device A to device B is set to be a vector of trend values w_t, namely $[w_1, w_2, ..., w_n]$ where n is the end of the time window. During actual network forensic analysis, the time series of the unknown traffic to be analyzed is computed, the trends of each of the time series are calculated and the weight of the corresponding edge is then determined.

The trend of a time series is computed using the least-squares estimation method. The model trend is first expressed as a quadratic function of time:

$$w_t = \beta_0 + \beta_1 t + \beta_2 t^2 \tag{2}$$

Next, the quadratic function of time is fit to the network traffic and the values of the parameters β_0, β_1 and β_2 are determined by minimizing the following equation:

$$\sum_{t=1}^{n}(x_t - (\beta_0 + \beta_1 t + \beta_2 t^2))^2 \tag{3}$$

The optimal values of β_0, β_1 and β_2 are used to compute the trend w_t at each t according to Equation (2), yielding the final edge weight.

Note that perfect matches between edge weights in the model and concrete graphs are neither sought nor desired. This is because fluctuations are normal and are, indeed, expected. In fact, the subgraph isomorphism search algorithm looks for weights that deviate substantially from those in the model graph.

5. Experimental Evaluation

The experiments leveraged a testbed with protective relays that emulated a real-world electrical substation. Table 1 summarizes the main components of the testbed along with their functions and communications protocols. The SEL-487E-3 device is a protective relay designed to monitor and protect a power transformer from electrical faults; it runs intelligent algorithms that detect various types of faults and take actions in a timely manner by operating electrical circuit breakers and disconnect switches. The SEL-421-4 device performs industrial automation functions; it comprises 32 programmable elements for local control, remote control, automation latching and protection latching. It also performs various functions that protect overhead electrical transmission lines and underground cables. The SEL-3355 device performs multiple substation functions; it has an integrated human-machine interface (HMI) with a local display port. In the experiments, the SEL-3355 device was used as a real-time automation controller. The SEL-3355 polled the SEL-487E-3 and SEL-421-4 for substation data. The network communications between these industrial control devices employed the IEC 61850 protocol.

The experimental testbed incorporated a Windows platform hosting an Object Linking and Embedding (OLE) for Process Control (OPC) server [3], which executed an IEC 61850 protocol driver to obtain substation data (including decoy data) from the protective relays. The IEC 61850 protocol driver subsequently stored the substation data in the OPC server. The testbed also incorporated an OPC client running under Windows; the OPC client application provided a graphical user interface for a human operator to enter OPC commands. Another Windows machine was used for development and testing. All the machines and devices in the experimental testbed were connected via a local area network.

The experiments validated the proposed approach using malware samples involved in the Dragonfly cyber espionage campaign. Several versions of the Dragonfly malware exist; the malware samples used in the experiments were obtained from public malware research repositories. Versions of the Havex industrial control system malware were also used; these were of particular interest because they use the OPC protocol.

The experiments emulated malware propagation and execution in a substation automation network. The experiments used the `nmap` tool to scan for the IP addresses of intelligent electronic devices and their network services. The vulnerability exploitation phase involved code that targeted several memory vulnerabilities using shellcode injection and heap spraying on intelligent electronic devices. Two exploitation scenarios were implemented, one where the malware had no prior knowledge of the power system state and the other where the malware was able to obtain complete knowledge about the power system state before launching its attacks.

The experiments also emulated the malware installation phase, which involved the injection and execution of a dropper on a compromised intelligent electronic device. As with traditional malware, the dropper was responsible for installing malware modules on the compromised intelligent electronic device. Single-stage and dual-stage dropper were emulated. The single-stage dropper incorporated the emulated malware modules that were installed on the compromised intelligent electronic device whereas the dual-stage dropper downloaded the modules from another compromised machine over the network.

Traffic associated with the network scans and test exploit that operated without any prior knowledge of the power system state produced visible struc-

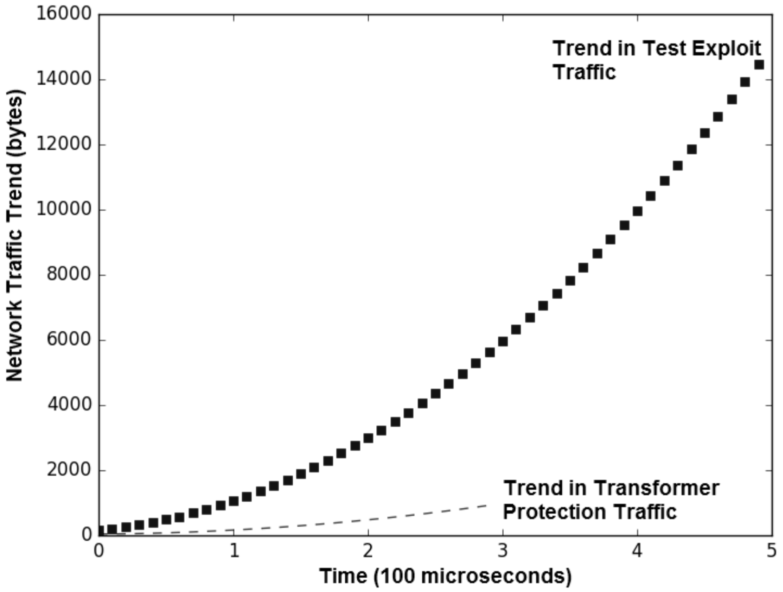


Figure 4. Time series trend comparison results for a test exploit.

tural deviations from the model graph. The target selection was random as were the times when the network probes and test exploit were launched. Even when the launch time coincided (by chance) with the time that a protection function was exercised by an intelligent electronic device, the structural deviations from the model graph caused the subgraph isomorphism search algorithm to fail. Thus, no further time series analysis was necessary in the subgraph isomorphism search.

The time series analysis became necessary when the network scans and test exploit operated with full knowledge of the power system state. The network probes and test exploit were launched at the time that a power transformer protection function was exercised by the intelligent electronic devices. Furthermore, the probes and exploit were launched from a compromised intelligent electronic device to another intelligent electronic device, both of which were exchanging network traffic when the power transformer protection function was exercised. The graph structure of the network traffic matched that of the model graph. However, as shown in Figure 4, time series analysis detected deviations in the trend. This was due to the test exploit generating large quantities of network traffic when performing its heap sprays. Its time window also extended beyond the maximum 4 ms time window of the power transformer protection function. The subgraph isomorphism search failed because the edge weights

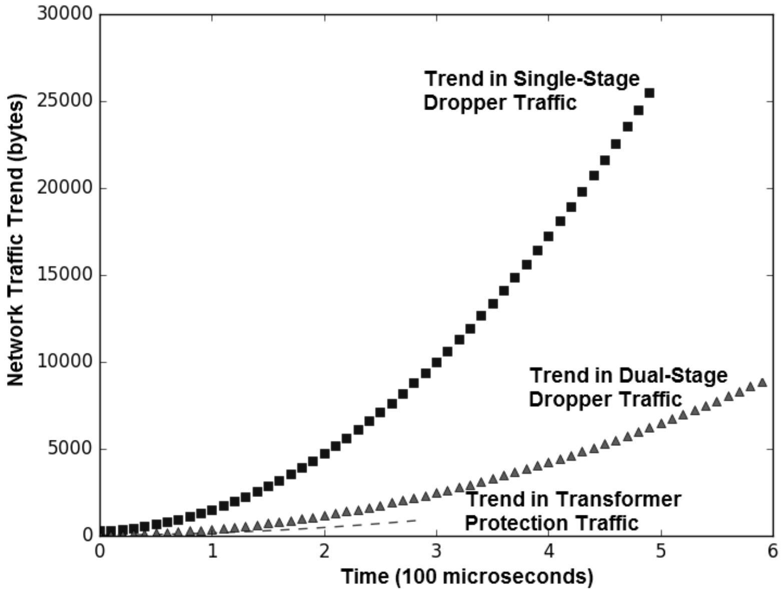


Figure 5. Time series trend comparison results for a test malware installation.

in the concrete network traffic graph deviated substantially from those in the model graph.

When malware installation was performed without any knowledge of the power system state, the network traffic caused by the dropper did not match a valid power system protection function (unless a coincidence occurred). The mismatch of the network traffic and substation protection function caused the subgraph isomorphism search to fail, resulting in the immediate detection of the malicious network packets.

A challenging situation for a defender occurs when the network traffic caused by the dropper is interwoven with the network traffic of a valid protection function. If the dropper were to be downloaded on a compromised intelligent electronic device with a high degree of mimicry, then the structural properties of the concrete graph would match those in the model graph. Since the network traffic would appear to be consistent with the substation protection function, the subgraph isomorphism search would have to rely on the time series analysis of the network traffic.

The weights in the concrete graph corresponding to the single-stage dropper deviated considerably from the weights in the model graph. The single-stage dropper incorporated several malware modules and, thus, its downloading on the compromised intelligent electronic device was characterized by heavy network traffic and the higher trend shown in Figure 5. In contrast, because the

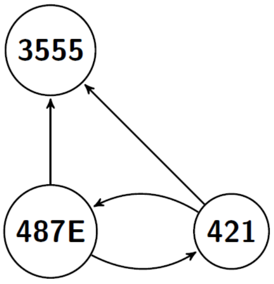


Figure 6. Excerpt topology of a concrete graph of normal network activity.

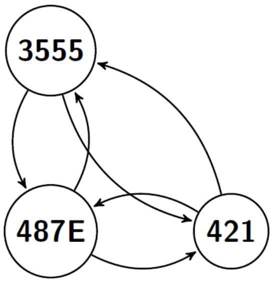


Figure 7. Excerpt topology of a concrete graph of malicious network activity.

dual-stage dropper was a thin client module that downloaded the malware modules over the network at a later time, much less traffic was observed than in the case of the single-stage dropper. Nevertheless, the time series in Figure 5 shows large deviations for the single-stage dropper and dual-stage dropper network traffic from the network traffic corresponding to a power transformer protection function.

The `nmap` tool and Havex malware generated network traffic with strong deviations in the trends, which caused the subgraph isomorphism searches to fail. The deviations in the trends were similar to the trends shown in Figures 4 and 5. The concrete graphs constructed from `nmap` and Havex network traffic as well as those constructed with the majority of the emulated malicious activities also demonstrated structural deviations from the model graph, which again caused the corresponding subgraph isomorphism searches to fail. Typical failures resulted from network traffic being sent to intelligent electronic devices that were not supposed to participate in the exercising of a function; this created additional vertices and edges in the concrete graphs, causing the subgraph isomorphism searches to fail. Figures 6 and 7 present portions of concrete graphs corresponding to normal traffic and malicious traffic, respectively. In general, the concrete graphs constructed from normal network traffic were easily determined to be subgraphs of the model graph.

6. Conclusions

The proposed network forensic approach treats an electrical substation as a cyber-physical system and explores its strong interrelations to develop a model graph that deeply characterizes the structural and timing properties of normal network traffic in the substation automation network. A similar concrete graph is constructed for unknown (potentially) network traffic. Unlike authorized code that executes on intelligent electronic devices, exploit code and malware generate network traffic that deviates from the monitoring and control tasks performed in an electric power system. In particular, the traffic generated by malicious code has structural and timing properties that differ from those of normal traffic. Thus, the problem of determining whether or not unknown network traffic is malicious involves a subgraph isomorphism comparison of the concrete graph against the model graph. Normal network packets yield a concrete graph that is a subgraph of the model graph whereas malware-generated network packets produce a graph that is not a subgraph of the model graph. Experiments involving real industrial control system malware demonstrate that the resulting approach can reliably identify malicious packets in large traffic captures from a substation automation network.

Acknowledgement

This research was supported in part by Defence R&D Canada and by the U.S. Office of Naval Research under DURIP Contract No. N00014-15-1-2891.

References

[1] I. Ahmed, S. Obermeier, M. Naedele and G. Richard, SCADA systems: Challenges for forensic investigators, *IEEE Computer*, vol. 45(12), pp. 44–51, 2012.

[2] K. Brand, C. Brunner and I. de Mesmaeker, How to use IEC 61850 in protection and automation, *Electra*, no. 222, pp. 11–21, 2005.

[3] J. Lange, F. Iwanitz and T. Burke, *OPC: From Data Access to Unified Architecture*, VDE-Verlag, Berlin, Germany, 2010.

[4] Y. Liang and R. Campbell, Understanding and Simulating the IEC 61850 Standard, Technical Report UIUCDCS-R-2008-2967, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, 2008.

[5] M. Rahman and B. Jeyasurya, A state-of-the-art review of transformer protection algorithms, *IEEE Transactions on Power Delivery*, vol. 3(2), pp. 534–544, 1988.

[6] A. Srivastav, C. Ortega, P. Ahuja, M. Christian and A. Cardenas, Exploratory analysis of Modbus and general IT network flows in a water SCADA system, presented at the *Industrial Control System Security Workshop*, 2015.

[7] J. Ullman, An algorithm for subgraph isomorphism, *Journal of the ACM*, vol. 23(1), pp. 31–42, 1976.

[8] T. Wu, J. Pagna Disso, K. Jones and A. Campos, Towards a SCADA forensics architecture, *Proceedings of the First International Symposium on ICS and SCADA Cyber Security Research*, pp. 12–21, 2013.

II

INFRASTRUCTURE MODELING AND SIMULATION

Chapter 5

MULTIPLE SECURITY DOMAIN MODEL OF A VEHICLE IN AN AUTOMATED PLATOON

Uday Kanteti and Bruce McMillin

Abstract This chapter focuses on the security of automated vehicle platoons. Specifically, it examines the vulnerabilities that occur via disruptions of the information flows among the different types of sensors, the communications network and the control unit in each vehicle of a platoon. Multiple security domain nondeducibility is employed to determine whether or not the system can detect attacks. The information flows among the various domains provide insights into the vulnerabilities that exist in the system and whether the model is nondeducible. If nondeducibility is found to be true, then an attacker can create an undetectable attack. Defeating nondeducibility requires additional information sources, including invariants pertaining to vehicle platoon operation. A platoon is examined from the control unit perspective to determine if the vulnerabilities are associated with preventing situational awareness, which could lead to vehicle crashes.

Keywords: Automated vehicle platoons, multiple security domain nondeducibility

1. Introduction

Automated vehicle systems are likely to be the future of transportation. The concept of a vehicle platoon where 8-25 vehicles follow each other and each vehicle mimics the actions performed by the vehicle in front of it is compelling. The Partners for Advanced Transit and Highways (PART) Project was introduced in 1986 to make this concept a reality. It was believed that introducing platoons would increase road capacity, reduce trip delays and limit energy consumption. A major reason for the PATH Program was to reduce accidents and breakdowns [13]. However, many people are hesitant to trust the decision making of driverless vehicles. As such, the approach taken in this research is to

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 81–97, 2017.
https://doi.org/10.1007/978-3-319-70395-4_5

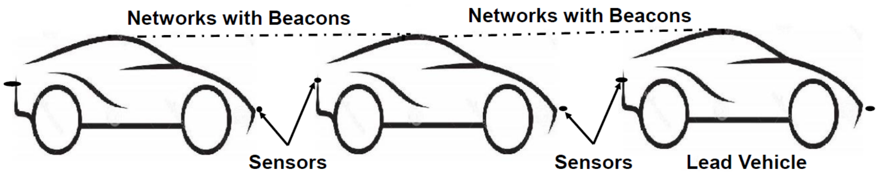


Figure 1. A vehicle platoon and its information transfer paths.

reduce the security threats that may impede vehicle decision making, reducing the barrier to the adoption of driverless vehicle technologies.

An automated vehicle is a cyber-physical system in which significant cyber, communications and physical components work together. The information present in the system may be cyber in nature or it may pertain to the physical properties of the system. Preserving correct information flows treats the security issues in the cyber-physical system uniformly. Disruption of information leads to the lack of deducibility of the system state.

The main contribution of this research is the development of a cyber-physical platoon model in which attacks are deducible. The goal is to create security domains such that, if an attack occurs in one domain, then the compromised domain can be detected with the help of information paths from the other domains. This research and its case scenarios demonstrate the potential to detect security problems in a cyber-physical system.

2. System Model

Figure 1 presents a vehicle platoon and its information transfer paths. The vehicle platoon works in the following manner [16]:

- The vehicle in front of the platoon is assigned the role of the lead vehicle. The lead vehicle decides the movements of the platoon. The vehicles behind the lead vehicle follow.
- Each vehicle in the platoon (except for the lead vehicle) receives information from the previous vehicle and maneuvers accordingly.
- If the lead vehicle wishes to slow down or take a turn, it indicates this via a beacon message and the other vehicles follow suit.
- Each vehicle checks and compares the information it receives from sensors and from the communications network. If the information is consistent, the vehicle proceeds; otherwise, it raises a flag.
- Other vehicles check the flagged vehicle information and decide whether to keep the vehicle in the platoon or remove it from the platoon.

Communications between autonomous vehicles create a vehicular ad hoc network. In platooning, each vehicle is connected directly to the vehicle in front, to the vehicle behind it and also to the lead vehicle (since each vehicle mimics the lead vehicle). This results in a constant number of connections (i.e., three connections) for each vehicle. All the vehicles in the platoon stay connected and the scalability problem is reduced.

This cyber-physical system consists of embedded computers, control units, a physical system, the vehicles and their sensors, and a message-passing communications network, each residing in its own security domain. Information flows among the various security domains.

The proposed model includes the following components:

- **Communications Network:** A platoon may use any wireless networking technology, the most prominent being a short range radio technology such as WLAN (standard Wi-Fi or ZigBee). Cellular technologies or LTE can also be used. In the United States, the IEEE 1609 WAVE (wireless access in vehicular environments) protocol stack builds on IEEE 802.11p WLAN that operates on seven reserved channels in the 5.9 GHz frequency band. The WAVE protocol stack is designed to provide multi-channel operation (even for vehicles equipped with a single radio), security and lightweight application layer protocols.

- **Sensors:** Velodyne and HDL-64E LiDAR sensors are designed for obstacle detection and navigation by autonomous vehicles. Their durability, $360°$ field of view and very high data rates render the sensors ideal for the most demanding perception applications as well as for 3D mobile data collection and mapping applications. The information received from a LiDAR sensor is sent to the vehicle control unit. RADAR sensors are currently used in advanced cruise control systems to measure vital parameters such as range, angle and Doppler velocity. This information is used to assess the driving situation and signal the control unit about potentially dangerous events.

- **Control Unit:** The control unit is the brain of a vehicle and gives it directions. The control unit makes decisions based on the inputs it receives from the sources mentioned above. If discrepancies in the information paths are detected, the control unit alerts other vehicles by sending special-purpose messages. If enough information paths are available, the control unit can take corrective actions itself.

- **Monitor:** Invariant equations (e.g., distance $=$ speed $\times$ time (t)) are evaluated by an embedded monitor to compute where the vehicle would be at time t. The assumption is that the vehicle would have its own speed calculation mechanism such as a speedometer and a clock to tell the time. At each instant, the monitor computes the distance information and sends it to the control unit. The monitor evaluates the invariant using each information source to check for consistency.

Messages with information about speed and distance are passed from the lead vehicle to other vehicles in the platoon in the form of beacons (messages transmitted in the network from one vehicle to another) as shown in Figure 1. Beacons contain information about the speeds and distances of all the vehicles in front of a given vehicle. A control unit gathers information directly from the sensors and from the communications network that exists between the vehicles.

3. Related Work

Various security threats target confidentiality (e.g., an attacker tracks a vehicle), integrity (e.g., an attacker changes beacon information) and availability (e.g., an attacker jams the network that communicates speed and distance values). The attacker is defined as someone/something that is not authorized to access or modify the vehicle or system components, but is able to do so. The attacker may intend to cause a traffic jam or even a crash.

3.1 Confidentiality

Much research on confidentiality or privacy has focused on securing the entity itself. Separate access control [9] can prevent changes to vehicle control commands. Dividing each section on the control board restricts information flow based on the type of control implemented by the section. This preserves the privacy of the vehicle.

A virtual trip lane (VTL) with multiple zones (e.g., VTL_1 and VTL_2) [14] can preserve privacy by using the lane to regulate location and speed reports. The estimated time for VTL_1 is computed when the vehicle enters the next zone VTL_2 and is compared against the actual arrival time in VTL_2. Releasing trajectory data only within a single virtual trip lane zone helps protect privacy.

3.2 Integrity

Research has focused on securing information using encryption [7], but this only provides conditional privacy (i.e., only the entities involved in the communications know about the communications). Several integrity attacks have targeted vehicles via direct access to the hardware or via a wireless channel. The main goal of an attacker is to access the controller area network (CAN) of a vehicle and control the vehicle.

Koscher et al. [8] discuss the vulnerabilities of controller area networks. Controller area network packets contain no authentication fields or even source identifier fields, meaning that any component can “invisibly” send packets to any other component in the network. This means that just one compromised component can be used to control all the other components on the controller area network bus. Koscher et al. report that broadcasting malicious data from an infected controller area network can enable an attacker to seize control of a vehicle.

A widely-reported wireless attack [11] took control of the engine control unit (ECU) of a Jeep that sends commands to the other components in the vehicle. The attack leveraged a cellular network connected to the vehicle entertainment system in order to compromise the controller area network. The attack was able to kill the brakes, steer the vehicle and control the horn and parking lights at will.

Koscher et al. [8] have also worked on wireless access, which is discussed in [3]. Access to the engine control unit is gained via Bluetooth and reverse engineering. Once access is gained, the attacker can make changes to the braking system and modify the speed of the vehicle.

Another recent attack targeted a Tesla S model by gaining wireless access to the controller area network of the vehicle [1]. The attack was launched when the driver connected to a malicious Wi-Fi hotspot. The researchers were able to devise an alternate authentication scheme using ECDSA with omission techniques and TESLA++. However, the problem of scalability still exists because each new vehicle has to be authenticated with every other vehicle in a group by a roadside unit (despite the fact that the authentication overhead is reduced when two groups intend to communicate). A special message is sent when the information is not authentic; the verification is performed by the engine control unit using information from another vehicle.

3.3 Availability

An attack that impacts the availability of information can cause serious problems to connected vehicles. Traditional techniques such as beamforming (i.e., actively steering wireless transmission and reception beams to maximize useful signal reception while minimizing interfering signal reception) can combat jamming attacks on sensors [12].

A Sybil attack can be used to target connected vehicles. In this attack, the reputation system of a peer-to-peer network system is subverted maliciously by creating a large number of pseudonymous identities. Using these identities, the attacker can gain a disproportionately large influence on the functioning of the system. This attack can be detected efficiently using a less complex cryptographic technique and obtaining pseudonyms from roadside units at continuous intervals from a trusted source [2]. Distinct roadside units that periodically collect reports from communicating vehicles in the neighborhood reduce the vulnerability.

In the long term, it is believed that vehicles involved in a Sybil attack would pass similar information as the benign vehicles. A trust-based system for detecting malicious vehicles can employ an iterative filtering algorithm to detect malicious vehicles and address the problem of collusion [6], where vehicles attempt to improve the trust ratings of false vehicles.

Denial-of-service (DoS) attacks can be detected easily. Lyamin et al. [10] use time intervals to listen on a channel and determine whether or not all the beacons have been received. If there is a beacon loss, two nodes may be involved in a collision within the same group. To prevent collisions, the nodes must have an initialization phase. The initialization phase ensures that the nodes start from safe states (i.e., no attacks) and do not collide with each other.

Some attacks are difficult to detect and mitigate using only one vehicle [11]. When multiple vehicles are present, the vehicles could examine the information they receive and determine that a vehicle is under attack. This is possible because the attacked vehicle would send information that would not match the information available to the other vehicles.

Sun et al. [14] have shown that the use of virtual trip lane zones can address privacy concerns. Their approach relies on real-time data. Therefore, if an attacker were to know the position of a vehicle in a virtual trip zone, the attacker would be track the movement of the vehicle in real time.

Certain attacks spoof data originating from a communications network and beacons [3, 8]. Cryptography can be used to secure messages between vehicles [1, 7]. Using a strong encryption method can secure communications, but the assumption is that the cryptographic keys are exchanged securely before communications are initiated. Another underlying assumption is that the vehicle itself is not compromised. In real-time scenarios involving vehicles it may not be possible to securely share information while the vehicles are in motion. The denial-of-service attacks discussed in [10, 12] are based on reducing interference and listening to the channel periodically, but hazardous weather conditions could make it difficult to listen to the channel. Instead of treating all these issues separately, the proposed methodology adopts a scientific approach that uniformly models the interactions of distinct security domains.

3.4 Multiple Security Domain Nondeducibility

Nondeducibility was introduced by Sutherland [15] in an attempt to model infrastructures that secure information using a partitioned model. The partitions are grouped into two or more sets. These sets are usually labeled as high and low with all the information restricted to one side of the partition or the other. Information that cannot be determined from the other side of the domain is said to be nondeducibility secure. However, the partition must be absolute and simple. Overlapping security domains present severe difficulties for nondeducibility as do information flows that cannot be evaluated because the model lacks the required valuation functions.

V is a set of valuation functions such that $V^i_{s_x}(w)$ returns the value of a state variable s_x as seen by an entity i in world w. For example, if a vehicle control unit c obtains distance information from sensor d_s, then $V^c_{d_s}(w)$ returns true; otherwise, it returns false.

Definition. A system is multiple security domain nondeducible (MSDND) secure if there exists a world with a pair of states where one state must be true and the other false (exclusive or), but an entity i has no valuation function for the states. An entity i in security domain SD^i cannot know which state is true and which state is false [4]. In particular:

$$\begin{aligned} \text{MSDND(ES)} = \exists w \in W \vdash \Box\, [\,(s_x \vee s_y)\,] \wedge \sim(s_x \wedge s_y) \wedge \\ [\,w \models (\,\nexists V^i_{s_x}(w) \wedge \nexists V^i_{s_y}(w)\,)\,] \end{aligned}$$

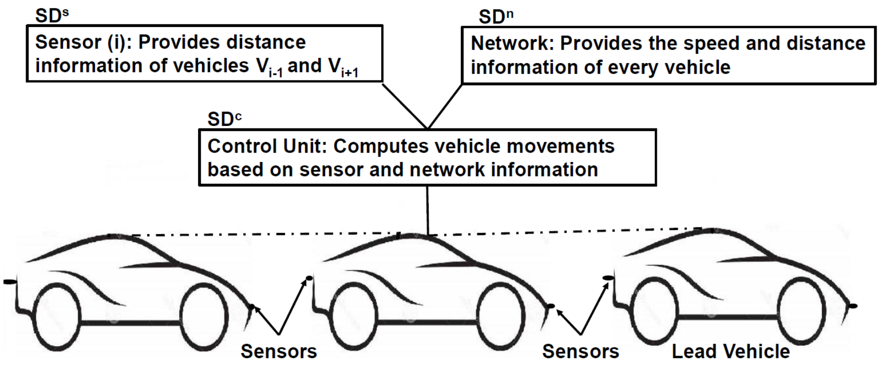

SD^s
Sensor (i): Provides distance information of vehicles V_{i-1} and V_{i+1}
SD^n
Network: Provides the speed and distance information of every vehicle
SD^c
Control Unit: Computes vehicle movements based on sensor and network information
Sensors
Sensors
Lead Vehicle

Figure 2. Security domains of information flow.

where s_x and s_y are states, V_x^i and V_y^i are the valuation functions of s_x in domain i and s_y in domain i, respectively, and w is a world.

4. Problem Statement

It is possible to devise an attack – like Stuxnet [5] – that changes the speed and distance values. However, it is important to be able to tell if the attack (i.e., changing sensor or network information) is MSDND. If the attack is MSDND, then the attacker has an advantage because the target would not know which component is malfunctioning. Therefore, the model should be designed to eliminate attacks that are MSDND secure.

Figure 2 shows the security domain partitions and the interactions between vehicle control units and communications points. A monitor positioned in a vehicle evaluates the invariants pertaining to the vehicle state. If a discrepancy exists in the distance information sent by one of the paths to the control unit, then the control unit would know that something is wrong.

The following entities can be evaluated to determine the interactions between a vehicle and the communications system:

- **c** : The vehicle control unit c obtains data and computes the movement.
- **s** : Each LiDAR sensor s provides a distance value $d(s)$.
- **n** : The communications network n between the vehicles provides the network value $d(n)$.
- **ISV** : The information source validator ISV is executed by the monitor to check if the information received from the information paths is valid. Table 1 shows that ISV sequentially checks sources against other sources and invariants that involve multiple sources.

Table 1. Information source validation performed by ISV.

Check	Source
ch_1	$d(s) = d(n)$
ch_2	$d(s) = d(invariant_1)$
ch_3	$d(n) = d(invariant_1)$
ch_4	$d(s) = d(beacon)$
ch_5	$d(s) = d(invariant_2)$
ch_6	$d(n) = d(beacon)$
$\vdots$	$\vdots$

The term $d(c)$ denotes the distance that the control unit accepts based on information received from the paths. For a given state, the valuation functions return the values of the corresponding state variables (c, s, n) as seen by the entity in control.

The control unit detects a compromised information path if the information it receives from the corresponding sensor is not equal to the value it receives from the communications network. In other words, it uses the result of a check ch_i. If the check value is false, then the control unit knows that one of the information paths has been compromised.

Five cases and their sub-cases are discussed in the following sections.

4.1 Case 1

Figure 1 shows that there are two information paths, sensor and network. Case 1 assumes that a vehicle has exactly one information path – either sensor or network – that provides information about the distance between the vehicle and the vehicle in front of it. In particular, the goal is to show that one information path can make the system MSDND secure as well as not MSDND secure. Four sub-cases are considered.

Case 1(a): Assume that the vehicles are connected only to network n and that the network provides correct information (i.e., $d(n)$ = true):

- $w \models (\exists V^c_{d(n)}(w))$: Control unit receives distance information from the network.

It follows that:

$$\begin{aligned}\text{MSDND(ES)} = {} & \exists w \in W : w \vdash \Box(d(n) \oplus \neg d(n)) \wedge \\ & [w \models (\exists V^c_{d(n)}(w) \vee \nexists V^c_{\neg d(n)}(w))]\end{aligned}$$

The system is not nondeducible secure to the control unit according to the definition because $\exists V^c_{d(n)}(w)$.

Case 1(b): Assume that the vehicles connected to network n receive incorrect information (i.e., $\neg d(n)$ = true):

- $w \models (\nexists V^c_{\neg d(n)}(w))$: Control unit receives distance information from the network.

It follows that:

$$\begin{aligned} \text{MSDND(ES)} = {} & \exists w \in W : w \vdash \Box(d(n) \oplus \neg d(n)) \wedge \\ & [w \models (\nexists V^c_{d(n)}(w) \vee \nexists V^c_{\neg d(n)}(w))] \end{aligned}$$

The system is nondeducible secure to the control unit according to the definition.

Case 1(c): Assume that the vehicles are connected only to sensor s, which provides correct information (i.e., $d(s)$ = true). This case is similar to Case 1(a).

Case 1(d): Assume that the vehicles are connected to sensor s and receive incorrect information (i.e., $\neg d(s)$ = true). This case is similar to Case 1(b).

Although the goal is to create a model that is not nondeducible secure, the control unit in Cases 1(b) and 1(d) would believe the data and direct the vehicle accordingly. Having no other information path for verification is potentially hazardous because it is nondeducible if the information received is incorrect.

4.2 Case 2

In Case 2, the sensor and communications network provide distance information to the control unit. Upon receiving the distance information, the control unit attempts to determine whether or not the information provided is correct. The control unit would know that an information path is corrupted if there is a discrepancy in the information provided by the two paths. In the following, two sub-cases are examined.

Case 2(a): If n is faulty, then the system has been compromised. The attacker can manipulate the information and, thus, incorrect information is received by the control unit from the network (i.e., $\neg d(n)$ = true and $d(s)$ = true).

- **S1:** $w \models (\exists V^c_{d(s)}(w))$: Control unit receives information from the sensor.
- **S2:** $w \models (\nexists V^c_{\neg d(n)}(w))$: Control unit receives information from the network.
- **S3:** $w \models (\exists V^c_{ch_1})$: Information received from the sensor and network do not match.

From statement S3, the control unit would know that an information path has been compromised.

Combining statements S1 and S2 yields the following expression:

$$\begin{aligned}\text{MSDND(ES)} = {} & \exists w \in W : w \vdash \Box(\neg d(n) \oplus ch) \wedge \\ & [w \models (\exists\, V^c_{d(ch_1)}(w) \vee \nexists\, V^c_{\neg d(ch)}(w))]\end{aligned}$$

The system is not nondeducible to the control unit because the unit can deduce that something is wrong. But it cannot determine which information path is responsible. An additional information path is needed to determine the path that transmits incorrect data.

Case 2(b): If s is faulty, then the system has been compromised. The attacker can manipulate the information and, thus, incorrect information is received by the control unit from the network (i.e., $\neg d(s)$ = true and $d(n)$ = true). The following statements hold:

- **S1:** $w \models (\nexists V^c_{\neg d(s)}(w))$: Control unit receives information from the sensor.
- **S2:** $w \models (\exists\, V^c_{d(n)}(w))$: Control unit receives information from the network.
- **S3:** $w \models (\exists V^c_{ch_1})$: Information received from the sensor and network do not match.

From statement S3, the control unit would know that an information path has been compromised. Combining statements S1 and S2 yields the following expression:

$$\begin{aligned}\text{MSDND(ES)} = {} & \exists w \in W : w \vdash \Box(\neg d(s) \oplus ch) \\ & \wedge [w \models (\exists V^c_{d(ch_1)}(w) \vee \nexists V^c_{\neg d(s)}(w))]\end{aligned}$$

The system is not nondeducible to the control unit because the unit can deduce that something is wrong. But it cannot determine which information path is responsible for transmitting the incorrect data.

Invariants relate the properties of a vehicle in a manner that, if one portion of the model is compromised, then the invariant is falsified. The MSDND model is extended as follows:

- $invariant_{dist}(d(i))$: distance = speed × time. Each vehicle has its own speedometer that enables it to compute the distance that it has traveled during a period of time.

Based on the model, the three sources that provide distance information are:

$$d_{t2} = \begin{cases} \text{range calculated by LiDAR/RADAR} - d_{t1}(s) \\ \text{speed} \times \text{time} - d_{t1}(i) \\ \text{distance calculated via network} - d_{t1}(n) \end{cases}$$

The three sources, sensor, network and invariant, are shown in Figure 3.

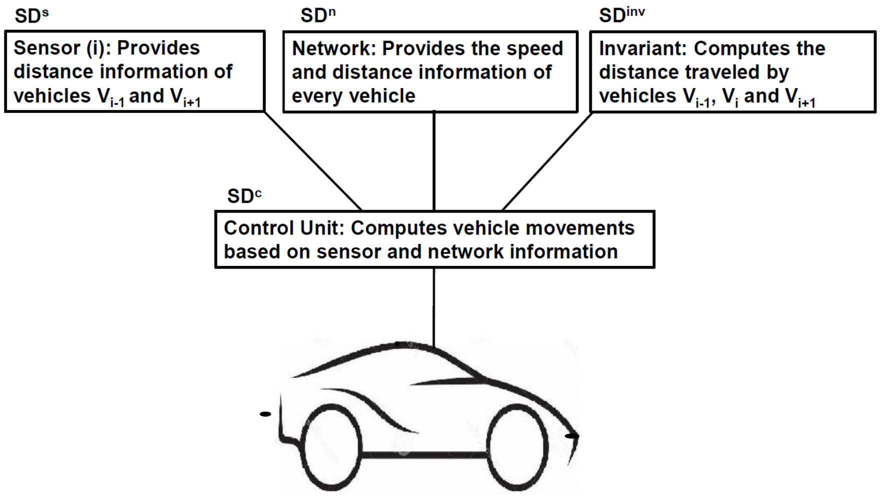

SD^s
Sensor (i): Provides distance information of vehicles V_{i-1} and V_{i+1}
SD^n
Network: Provides the speed and distance information of every vehicle
SD^inv
Invariant: Computes the distance traveled by vehicles V_{i-1}, V_i and V_{i+1}
SD^c
Control Unit: Computes vehicle movements based on sensor and network information

Figure 3. Invariant model with three information paths.

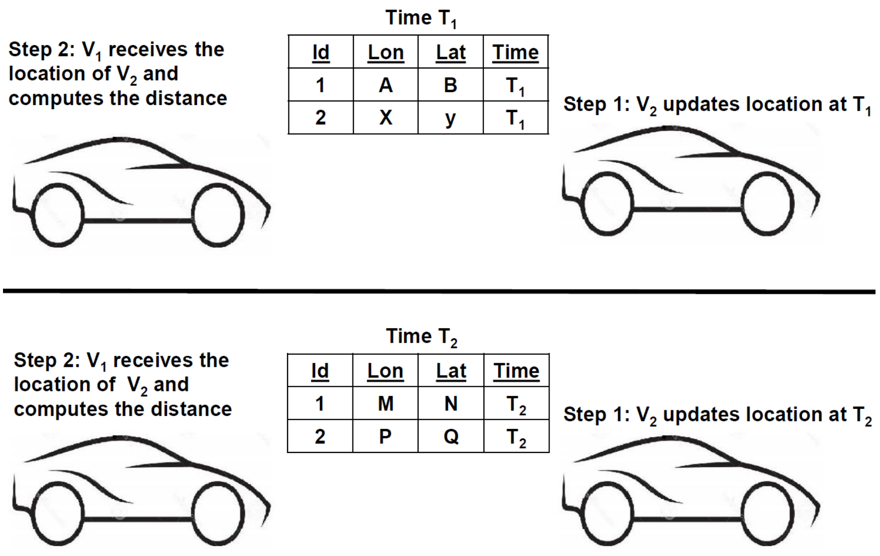
Sensor vs. Invariant: It is assumed that the vehicles are moving at constant velocity. Assume that, at time = 1 s, a sensor reports $d(s) = 5$ m and the vehicle speed is 30 m/sec. At time = 5 s, since there is no change in the speed of the vehicle, the sensor should report $d(s) = 5$ m. If any other information is provided, then it can be determined that the sensor has been compromised.

Network vs. Invariant: The network periodically updates the speed, location and other relevant information about a vehicle as shown in Figure 4. Specifically, the vehicle can calculate the distance covered in the $t_2 - t_1$ interval using the latitude (lat) and longitude (lon) and by computing: distance = speed × time. If at time t_1, the network gives a certain location for vehicle V_{i+1}, and at t_2, the network gives another location for V_{i+1}. Then, vehicle V_i can compute the distance moved by V_{i+1} because it has its speed and the duration. If there is a discrepancy in the information, then the network has been compromised.

The control unit would have the correct distance information if a valuation function exists for any one of ch_1, ch_2 or ch_3.

4.3 Case 3

In Case 3, another information path (invariant path) is added to the model. This additional information path helps make the model not MSDND secure. The control unit would also have the correct distance information if a valuation function exists for any one of the checks ch_1, ch_2 or ch_3, which can indicate


Time T_1
Step 2: V_1 receives the location of V_2 and computes the distance
Step 1: V_2 updates location at T_1
Time T_2
Step 2: V_1 receives the location of V_2 and computes the distance
Step 1: V_2 updates location at T_2

Time T_1

Id	Lon	Lat	Time
1	A	B	T_1
2	X	y	T_1

Time T_2

Id	Lon	Lat	Time
1	M	N	T_2
2	P	Q	T_2

Figure 4. Network vs. invariant.

which path is compromised. Thus, the control unit can sense that something is wrong.

Case 3(a): n is faulty (i.e., $\neg d(n)$ = true). The following statements hold:

- **S1:** $w \models (\nexists V^c_{\neg d(n)}(w))$: Control unit receives information from the network.
- **S2:** $w \models (\exists V^c_{d(s)}(w))$: Control unit receives information from the distance estimators.
- **S3:** $w \models (\exists V^c_{d(i)}(w))$: Control unit receives information from *invariant*$_{dist}$.
- **S4:** $w \models \nexists V^c_{ch_1}$.
- **S5:** $w \models \exists V^c_{ch_2}$.
- **S6:** $w \models \nexists V^c_{ch_3}$.
- **S7:** $w \models \exists V^c_{d(c)}(w)$: From statement S5.

It follows that:

$$\begin{aligned}\text{MSDND(ES)} = \exists w \in W : w \vdash \Box(\neg d(n) \oplus ch_2) \wedge \\ [w \models (\exists V^c_{d(ch_2)}(w) \vee \nexists V^c_{\neg d(n)}(w))]\end{aligned}$$

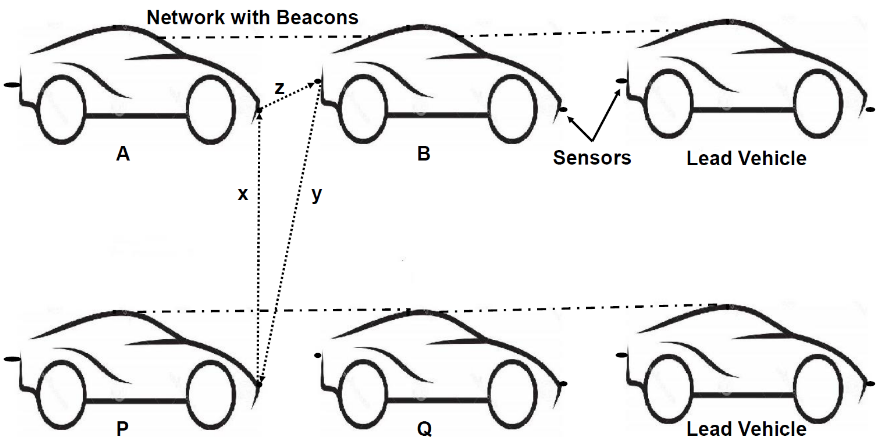


Figure 5. Multiple platoons.

This is not nondeducible secure because the control unit can deduce that something is wrong and can determine that the network is responsible for the incorrect data being transmitted.

Case 3(b): s is faulty (i.e., $\neg d(s)$ = true). This case is similar to Case 3(a).

It is assumed that the invariant is always true and that the information about the speed is unaltered. However, as discussed in [11], the attacker can also change the information received by the control unit. In this situation, the other two components, sensor and network, can provide the correct information.

Assume that there are multiple vehicles in the platoon and multiple platoons are nearby. This situation is shown in Figure 5.

The sources of information are:

- **s** : Each LiDAR sensor s provides a distance value $d(s)$.
- **n** : The communications network n between the vehicles provides the network value $d(n)$.
- **Invariant$_1$** : distance = speed $\times$ time gives the value of $d(invariant_i)$. Every vehicle has a speedometer with which it can compute the distance it has traveled during a period of time.
- **ISV** : The information source validator ISV is essentially an array data type that stores the true/false results after comparing the distance reported by each information source with another as shown in Table 1. If more true values are reported than false values, then a valuation function for ISV exists.
- **invariant$_2$** : As shown in Figure 5, communications can occur between two platoons. In this case, vehicles A, B and P form an information path

because each vehicle is equipped with proximity sensors. Vehicle P has the information $\{x, y\}$, vehicle A has the information $\{x, z_A\}$ and vehicle P has the information $\{y, z_B\}$. Sending this information back to vehicles A, B and C yields the information $\{x, z_A, z_B\}$. If z_A is not equal to z_B, then no valuation function exists for $invariant_2$; otherwise, a valuation function exists.

- **beacon$_\mathbf{i}$** : This provides information about $speed_i$ and $velocity_i$ of vehicle V_i in the platoon.

4.4 Case 4

In Case 4, multiple vehicles are in a platoon and no other platoons are nearby. Vehicles communicate information between each other using beacons. A beacon is an additional information path to the control unit that can help detect an incorrect information path.

Consider the situation where a vehicle provides incorrect information (i.e., $\neg beacon$ = true). The following statements hold:

- **S1:** $w \models (\exists V^c_{d(n)})$: Control unit receives information from the network.
- **S2:** $w \models (\exists V^c_{d(s)})$: Control unit receives information from the sensor.
- **S3:** $w \models (\exists V^c_{inv_1})$: Control unit receives information from $invariant_1$.
- **S4:** $w \models (\nexists V^c_{beacon_i})$: Control unit receives information from the beacon.
- **S5:** $w \models \exists V^c_{ISV}$: From statements S1, S2 and S3.
- **S6:** $w \models \exists V^c_{d(c)}(w)$: From statement S5.

It follows that:

$$\begin{aligned} \text{MSDND(ES)} = \exists w \in W : w \vdash \Box(\neg beacon_i \oplus ISV) \wedge \\ [w \models (\exists V^c_{d(c)}(w) \vee \nexists V^c_{\neg d(c)}(w))] \end{aligned}$$

This is not nondeducible secure because the control unit can deduce that something is wrong and can determine that $beacon_i$ is responsible for the incorrect data being transmitted. Thus, it is possible to detect if a vehicle in the platoon has been compromised.

4.5 Case 5

In Case 5, there are multiple platoons as shown in Figure 5. Information paths exist between adjacent platoons. The information paths help detect an incorrect data path when multiple information paths are compromised.

Consider the situation where $\neg beacon_i$ = true. The following statements hold:

- **S1:** $w \models (\exists V^c_{d(n)})$: Control unit receives information from the network.
- **S2:** $w \models (\exists V^c_{d(s)})$: Control unit receives information from the sensor.
- **S3:** $w \models (\exists V^c_{inv_1})$: Control unit receives information from $invariant_1$.
- **S4:** $w \models (\nexists V^c_{beacon_i})$: Control unit receives information from $beacon_1$.
- **S5:** $w \models (\exists V^c_{inv_2})$: Control unit receives information from $invariant_2$.
- **S6:** $w \models \exists V^c_{inv_2}$: From statements S1, S2, S3 and S5.
- **S7:** $w \models \exists V^c_{d(c)}(w)$: From statement S5.

It follows that:

$$\begin{aligned}\text{MSDND(ES)} = {} & \exists w \in W : w \vdash \Box(\neg beacon_i \oplus invariant_2) \wedge \\ & [w \models (\exists V^c_{d(c)}(w) \vee \nexists V^c_{\neg d(c)}(w))]\end{aligned}$$

This is not nondeducible secure because the control unit can detect that something is wrong and can determine that $beacon_i$ is responsible for the incorrect data being transmitted.

As demonstrated above, information from other platoons can be used to detect the compromised information path and, ultimately, the compromised vehicle. The case where platoons can detect if a vehicle has been compromised is considered because Case 4 demonstrates that it is possible to detect a compromised vehicle without a platoon. However, if other information sources are compromised, then having the additional source assists in attack detection.

5. Conclusions

MSDND is useful to model attacks where the goal is to hide critical information from an attacker. MSDND secure is a major disadvantage to a vehicle platoon and a boon to an attacker because information can be hidden by making it impossible to detect an attack or the valuation function could be falsified to produce an invalid valuation, rendering the information MSDND secure and undetectable. As demonstrated in the case study, the vehicle control units in a platoon may be unable to determine which vehicle has been compromised.

This research has focused on securing vehicles in an automated platoon. Minimizing the number of assumptions made in the model is a topic of future research. It is also necessary to handle situations where more than half of the information sources are compromised. Another problem is to handle cases where two sets of information sources provide the same incorrect information. Other vehicular scenarios that will be considered include platoon joining, lane changing and platoon splitting.

Acknowledgement

This research was supported, in part, by the Future Renewable Electric Energy Distribution Management Center, a National Science Foundation supported Engineering Research Center, under Grant EEC 0812121; by the National Science Foundation under Grant CNS 1505610; and by the National Institute of Standards and Technology under Grant 60NANB15D236.

References

[1] Y. Abueh and H. Liu, Message authentication in driverless cars, *Proceedings of the IEEE Symposium on Technologies for Homeland Security*, 2016.

[2] M. Al Mutaz, L. Malott and S. Chellappan, Leveraging platoon dispersion for Sybil detection in vehicular networks, *Proceedings of the Eleventh International Conference on Privacy, Security and Trust*, pp. 340–347, 2013.

[3] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner and T. Kohno, Comprehensive experimental analyses of automotive attack surfaces, *Proceedings of the Twentieth USENIX Conference on Security*, 2011.

[4] G. Howser and B. McMillin, A multiple security domain model of a drive-by-wire system, *Proceedings of the Thirty-Seventh IEEE Computer Software and Applications Conference*, pp. 369–374, 2013.

[5] G. Howser and B. McMillin, A modal model of Stuxnet attacks on cyber-physical systems: A matter of trust, *Proceedings of the Eighth International Conference on Software Security and Reliability*, pp. 225–234, 2014.

[6] H. Hu, R. Lu, Z. Zhang and J. Shao, REPLACE: A reliable trust-based platoon service recommendation scheme in VANET, *IEEE Transactions on Vehicular Technology*, vol. 66(2), pp. 1786–1797, 2017.

[7] D. Huang, S. Misra, M. Verma and G. Xue, PACP: An efficient pseudonymous authentication-based conditional privacy protocol for VANETs, *IEEE Transactions on Intelligent Transportation Systems*, vol. 12(3), pp. 736–746, 2011.

[8] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham and S. Savage, Experimental security analysis of a modern automobile, *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 447–462, 2010.

[9] U. Lang and R. Schreiner, Managing security in intelligent transport systems, *Proceedings of the Eighteenth IEEE International Conference on Intelligent Transportation Systems*, pp. 48–53, 2015.

[10] N. Lyamin, A. Vinel, M. Jonsson and J. Loo, Real-time detection of denial-of-service attacks on IEEE 802.11p vehicular networks, *IEEE Communications Letters*, vol. 18(1), pp. 110–113, 2014.

[11] C. Miller and C. Valasek, Remote exploitation of an unaltered passenger vehicle, presented at *DEF CON 23*, 2015.

[12] G. Patounas, Y. Zhang and S. Gjessing, Evaluating defense schemes against jamming in vehicle platoon networks, *Proceedings of the Eighteenth IEEE International Conference on Intelligent Transportation Systems*, pp. 2153–2158, 2015.

[13] S. Shladover, The California PATH Program of IVHS research and its approach to vehicle-highway automation, *Proceedings of the Intelligent Vehicles Symposium*, pp. 347–352, 1992.

[14] Z. Sun, B. Zan, J. Ban, M. Gruteser and P. Hao, Evaluation of privacy preserving algorithms using traffic knowledge based adversary models, *Proceedings of the Fourteenth IEEE International Conference on Intelligent Transportation Systems*, pp. 1075–1082, 2011.

[15] D. Sutherland, A model of information, *Proceedings of the Ninth National Computer Security Conference*, pp. 175–183, 1986.

[16] A. Wasef and X. Shen, PPGCV: Privacy preserving group communications protocol for vehicular ad hoc networks, *Proceedings of the IEEE International Conference on Communications*, pp. 1458–1463, 2008.

Chapter 6

DISTRIBUTED DATA FUSION FOR SITUATIONAL AWARENESS IN CRITICAL INFRASTRUCTURES WITH LINK FAILURES

Antonio Di Pietro, Stefano Panzieri and Andrea Gasparri

Abstract This chapter presents a distributed data fusion algorithm for situational awareness in critical infrastructures whose link failures are based on the transferable belief model. The algorithm is applied to a case study involving a class of critical infrastructures that exchange the possible causes of the faults or threats that affect them. The algorithm is robust to communications link failures caused by natural disasters, cyber attacks or physical security breaches. Theoretical results show that algorithm convergence only requires the connectedness of the network topology over a certain time window, providing resilience in the face of temporary disruptions in the infrastructure communications layer.

Keywords: Distributed data fusion, information sharing, situational awareness

1. Introduction

Data fusion provides a means for combining pieces of information that come from diverse sources and sensors. Fusing this information can help bolster the security of critical infrastructure assets such as power grids and water distribution networks by providing improved situational awareness that can significantly enhance decision making. Critical infrastructure assets typically aggregate information coming from their sensors for their own use and do not share information about their operational states with other infrastructures. This is because disseminating sensitive information to other infrastructures can pose security problems, a topic that has been investigated by several researchers (see, e.g., [18]). However, the lack of information sharing about disaster mitigation procedures negatively affects disaster recovery, as in the case of the Sri Lanka tsunami of 2004 [13].

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 99–117, 2017.
https://doi.org/10.1007/978-3-319-70395-4_6

The U.S. Department of Homeland Security has spearheaded the creation of information sharing and analysis centers (ISACs) to facilitate information sharing efforts in the various critical infrastructure sectors. In fact, during real-time situations, aggregating information about the operations of multiple infrastructures can be very effective. For example, physical damage to a system can be assessed and then repaired in different ways depending on its cause (e.g., flooding due to heavy rainfall versus the opening of a valve by a computer virus). An accurate assessment of a situation obtained via data fusion can lead to successful incident response. Such an assessment requires all the relevant data to be available at the same time.

Ducourthial et al. [8] have proposed a distributed algorithm that implements data fusion in an unknown network topology. The algorithm computes the confidence of each node by combining all the data coming from its neighboring nodes using a discounted cautious operator and without relying on a central node for data collection. The algorithm converges in finite time for any initial configuration and any unknown network topology. However, the algorithm requires the network topology to become stable (i.e., nodes and links are fixed, and agents do not perform any dynamic observations) in order to reach convergence.

Considerable research has been conducted on applying data fusion techniques to enhance the security of critical infrastructure assets. Flammini et al. [9] have proposed a theoretical centralized framework for correlating events detected by a wireless sensor network in the context of critical infrastructure protection. This framework was leveraged in an early warning system that enhances decision making on combating security threats by collecting data from various sources. However, the centralized nature of this and other approaches – where all the data must be collected by a single node that performs data aggregation – reduces their robustness to node failure.

In contrast, this research advances the state of the art by eliminating the common assumption of a static network topology in order to accommodate scenarios where link failures may occur due to natural disasters, cyber attacks or physical security breaches. Specifically, a distributed data aggregation algorithm for situational awareness in critical infrastructures is presented, where the network topology that describes the communications layer is unknown and may change with time. The algorithm converges in finite time without requiring a stable network topology by engaging the cautious rule of combination [7] to aggregate data. This combination rule does not require the information sources to be independent or distinct and is, therefore, preferred to other rules (e.g., transferable belief model conjunctive rule [6] and Dempster combination rule [16] that lack robustness when information with equal credibility is combined several times). Because the cautious rule of combination is appropriate when all the sources are considered to be reliable, the convergence response is defined when all the sources are non-distinct and reliable. The effectiveness of the proposed algorithm is demonstrated using a case study involving several interconnected and interdependent critical infrastructures that are subjected to physical failures. Addressing this challenging critical infrastructure protection scenario requires a novel distributed data fusion framework that can reduce the risk of cascading failures by dynamically sharing information between the infrastructures.

2. Preliminaries

The theory of evidence is a formalism for modeling imprecision and uncertainty without resorting to classical probability. This theory, introduced by Dempster [5] and Shafer [16], also referred to as the Dempster-Shafer theory, associates a number between zero and one to model the degree of confidence in a proposition based on partial (uncertain or imprecise) evidence.

Let $\Omega = \{\omega_1, ..., \omega_n\}$ be a finite set of possible values of a variable ω whose elements ω_i are assumed to be mutually exclusive. Let $\Gamma(\Omega) \triangleq 2^\Omega$ be the power set of Ω. The goal is to quantify the confidence of a proposition of the form: "The true value of ω is in γ" where $\gamma \in 2^\Omega$. The set Ω is referred to as the frame of discernment.

Basic Belief Assignment. A function $m : 2^\Omega \to [0, 1]$ is called a basic belief assignment (BBA) m if $\sum_{\gamma_a \in 2^\Omega} m(\gamma_a) = 1$ with $m(\emptyset) = 0$.

A basic belief assignment m can be equivalently represented by its associated commonality $q : 2^\Omega \to [0, 1]$ defined as:

$$q(\gamma_a) = \sum_{\gamma_b \supseteq \gamma_a} m(\gamma_b), \qquad \gamma_a \in 2^\Omega \tag{1}$$

Thus, for $\gamma_a \in 2^\Omega$, $m(\gamma_a)$ is the portion of the confidence that supports exactly γ_a. In other words, the true value of ω is in γ_a but, due to the lack of further information, it does not support any strict subset of γ_a.

A limitation of the Dempster-Shafer formulation is that the application of the Dempster combination rule [16] produces counterintuitive results when strong conflicts exist among the sources to be combined [19]. Smets [17] attempted to address this problem by proposing the transferable belief model (TBM), which relies on the concept of the basic belief assignment, but removes the assumption $m(\emptyset) = 0$. The removal of this assumption applies when the frame of reference is not exhaustive, so that it is reasonable to believe that another event, not modeled in the considered frame, will occur. This leads to the definition of the TBM conjunctive rule [6], which is more robust than the Dempster combination rule in the presence of conflicting evidence.

TBM Conjunctive Rule. The combination rule used in the transferable belief model removes the normalization constant in the Dempster combination rule. The new TBM conjunctive rule is defined as:

$$m_{ij}(\gamma_a) = \sum_{\gamma_b, \gamma_c; \gamma_b \cap \gamma_c = \gamma_a} m_i(\gamma_b) m_j(\gamma_c) \qquad \gamma_a \in 2^\Omega \tag{2}$$

The TBM conjunctive rule is associative and its use is appropriate when conflicts arise due to the low reliability of some of the data sources. However, this rule and the Dempster combination rule rely on the distinctness assumption of the sources; in other words, the information sources are independent. This limitation can be avoided using a combination rule with the idempotence property. Denoeux [7] defines an associative, commutative and idempotent operator called the cautious rule of combination, which is appropriate when all the sources are considered to be reliable. This rule does not require the assumption of independence.

Weight Function. Let m be a generic basic belief assignment. Then, the weight function $w : 2^\Omega \setminus \Omega \rightarrow \mathbb{R}^+$ is defined as:

$$w(\gamma_a) = \prod_{\gamma_b \supseteq \gamma_a} q(\gamma_b)^{(-1)^{|\gamma_b|-|\gamma_a|+1}} \qquad \forall \gamma_a \in 2^\Omega \setminus \Omega \tag{3}$$

$$= \begin{cases} \dfrac{\prod_{\gamma_b \supseteq \gamma_a, |\gamma_b| \notin 2\mathbb{N}} q(\gamma_b)}{\prod_{\gamma_b \supseteq \gamma_a, |\gamma_b| \in 2\mathbb{N}} q(\gamma_b)} & \text{if } |\gamma_a| \in 2\mathbb{N} \\ \\ \dfrac{\prod_{\gamma_b \supseteq \gamma_a, |\gamma_b| \in 2\mathbb{N}} q(\gamma_b)}{\prod_{\gamma_b \supseteq \gamma_a, |\gamma_b| \notin 2\mathbb{N}} q(\gamma_b)} & \text{otherwise} \end{cases}$$

Cautious Rule of Combination. Let m_i and m_j be two generic basic belief assignments in the transferable belief model with weight functions w_i and w_j, respectively. Then, their combination using the cautious conjunctive rule, denoted by $w_{i \ominus j} = w_i \ominus w_j$, is defined by the weight function:

$$w_{i \ominus j}(\gamma_a) = w_i(\gamma_a) \ominus w_j(\gamma_a) = \min(w_i(\gamma_a), w_j(\gamma_a)) \qquad \forall\, \gamma_a \in 2^\Omega \setminus \Omega \tag{4}$$

The proposed data aggregation technique works with the weight function w, which is obtained from masses using the commonality function q that is derived from the initial set of basic belief assignments. The next two sections demonstrate how these functions are generated from an initial set of basic belief assignments and are used to test the convergence of the algorithm.

3. Distributed Data Fusion

Consider a network described by an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}(t)\}$ where $\mathcal{V} = \{v_i : i = 1, .., N\}$ is a set of nodes (agents) and $\mathcal{E}(t) = \{e_{ij}(t) = (v_i, v_j)\}$ is a set of edges that represent the available point-to-point communications channels. The term t_k denotes the instant when the k^{th} communication occurs in the network.

Four assumptions are made about the network of agents: (i) the network is described by a connected undirected graph; (ii) every node produces a local basic belief assignment expressed as a weight function called the direct con-

Algorithm 1 : Gossip algorithm.

```
$t = 0$, $s_i(0) = C_i(0)\ \ \forall\ i \in 1, ..., N$
$s_i(t_{stop})\ \ \forall\ i \in 1, ..., N$
while stop_condition == False do
    Select an edge $e_{ij}$ in $\mathcal{E}(t)$ according to $\mathfrak{e}$
    Update the state of the selected agents according to $\mathcal{R}$
    $s_i(t+1) = s_i(t) \ominus s_j(t)$
    $s_i(t+1) = s_i(t) \ominus s_j(t)$
    $t = t + 1$
end while
```

fidence; (iii) agent communications are asynchronous – at every time instant t_k only one pair of agents (i, j) interact; and (iv) each agent i can handle the storage of the current direct confidence and the edge confidence computed via aggregation with a node v_j such that $(v_i, v_j) \in \mathcal{E}(t)$.

In the proposed framework, interactions between agents are modeled using a gossip algorithm [1]. An interaction is defined as a triplet $\{\mathcal{S}, \mathcal{R}, \mathfrak{e}\}$ for which the following conditions hold:

- $\mathcal{S} = \{s_1, ..., s_n\}$ is the set containing the local states $s_i \in \mathbb{R}^q$ of each agent i in the network such that $s_i(t) = (w_i(t, \gamma_1), .., w_i(t, \gamma_q))$ at time t with $q = |2^\Omega \setminus \Omega|$.
- $\mathcal{R}$ is the interaction rule based on the $\ominus$ operator that, for any two agents (i, j) with $e_{ij} \in \mathcal{E}(t)$, yields $\mathcal{R} : \mathbb{R}^q \times \mathbb{R}^q \to \mathbb{R}^q$ such that:

$$s_i(t) \ominus s_j(t) = (w_{i\ominus j}(t, \gamma_1), ..., w_{i\ominus j}(t, \gamma_q)) \tag{5}$$

- $\mathfrak{e}$ is the edge selection process, which specifies the edge $e_{ij} \in \mathcal{E}(t)$ that is selected at time t.

Algorithm 1 specifies the gossip algorithm used in this work. The term $C_i(0)$ in the algorithm denotes the initial direct confidence of agent i. Note that the algorithm does not require agents to have unique identifiers. In other words, agents are not required to know the identities of the neighbors with which they exchange information. This assumption is not cosmetic because security and confidentiality are common requirements in interdependent critical infrastructures [2].

Thus far, the gossip algorithm based on the $\mathcal{R}$ interaction rule has been presented. This update rule is similar to the min-consensus algorithm described by Cortes [3]. The major difference is that the proposed framework considers a gossip scheme and, thus, the interactions are asynchronous while Cortes [3] considers a consensus scheme where the interactions are synchronous.

Lemma 1: Consider a gossip algorithm $\{\mathcal{S}, \mathcal{R}, \mathfrak{e}\}$ over a time-varying graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}(t)\}$ with $\mathcal{S}$ and $\mathcal{R}$ as defined previously. Assume that each agent i at time $t = 0$ provides an independent set of direct confidences described by the weight function values $s_i(0) = \{w_i(0, \gamma_a);\ \gamma_a \in 2^\Omega \setminus \Omega\}$ obtained from the basic belief assignments and commonality functions $m_i(0)$ and $q_i(0)$, respectively, via Equations (1) and (2). If $\mathfrak{e}$ is such that $\forall t$ there exists a $\Delta t \in \mathbb{R}$ where $\mathcal{G}(t, t + \Delta t)$ is connected, then there exists a time $t = \bar{t}$ such that $\forall\ t' > \bar{t},\ \ \forall\ \gamma_a \in 2^\Omega \setminus \Omega$ the following relation holds:

$$s(t') = s_1(0) \ominus s_2(0)\ \ominus \ldots \ominus s_n(0) \tag{6}$$

In other words, each agent i converges to the same weight function.

Proof: In order to prove the convergence of the algorithm to steady state, consider a generic network topology where $|V| = N$ is the number of agents (i.e., critical infrastructures). Consider the network at different time intervals $[t_0, t_0 + \Delta t_0]$, $[t_1, t_1 + \Delta t_1]$, ..., $[t_h, t_h + \Delta t_h]$ with $t_1 = t_0 + \Delta t_0 + 1$, $t_h = t_{h-1} + \Delta t_h + 1$. During each time interval Δt_i, the agents interact using the interaction rule $\mathcal{R}$ to form a connected graph. In particular, for any pair v_i and v_j such that $(v_i, v_j) \in \mathcal{E}(t)$ at time t, the cautious rule of combination is applied so that the two agents agree on the minimum of the weight function set values:

$$s_i(t) \ominus s_j(t) = (w_{1\ominus 2}(t, \gamma_1), .., w_{1\ominus 2}(t, \gamma_z)) \tag{7}$$

where $z = |2^\Omega \setminus \Omega|$.

Without any loss of generality, consider γ_a and assume that there exists an agent v_q such that $w_q(0, \gamma_a) = \overline{w}(0, \gamma_a) \leq w_m(0, \gamma_a)$ for $v_m \in\ \mathcal{V}$. Note that in each iteration, all the $w(t, \gamma_a)$ values with $\gamma_a \in 2^\Omega \setminus \Omega$ are compared between two agents to find the minimum value. For the sake of simplicity, consider a generic $w(t, \gamma_a)$ (the reasoning below will hold for any $w(t, \gamma_a)$). For each time interval Δt_k such that the graph $\mathcal{G}'(t_k + \Delta t_k)$ is connected, a particular edge selection policy is used that updates only one agent to $\overline{w}(t_k, \gamma_a)$. In addition, consider a partition of $P = \{U, W\}$ of $\mathcal{V}$ with $U, W \subseteq \mathcal{V},\ U\ \cap\ W = \emptyset,\ U\ \cup\ W = V$ where U contains the agents that have reached the $\overline{w}(t_k, \gamma_a)$ and $W = V \setminus U$ is the set of remaining agents. The worst-case scenario for the edge selection policy $\mathfrak{e}$, in terms of the number of interactions required to update the state of only one agent, is when it verifies that the connection between two agents $v_u \in U$ and $v_w \in W$ for which $e_{uv} = (v_u, v_w) \in \mathcal{E}'(t_k, t_k + \Delta t_k)$ occurs as the last interaction and the graph $\mathcal{G}'(t_k + \Delta t_k) = \{U \cup W,\ \mathcal{E}'(t_k, t_k + \Delta t_k)\}$ with $e_{uv} \in \mathcal{E}'(t_k, t_k + \Delta t_k)$ is connected.

Next, consider the worst-case topology of a network with N agents; this is the topology for which a larger number of updates is required to reach convergence when the worst-case edge selection policy is considered. Clearly, the worst-case scenario is the topology with the largest diameter d (i.e., line topology with $d = N - 1$).

Figure 1 shows the convergence to steady state at different time intervals for the worst-case network topology with $N = 5$, where the agent with $\overline{w}(t_i, \gamma_a)$ is placed on the extreme right so that the longest diameter is obtained. Note that

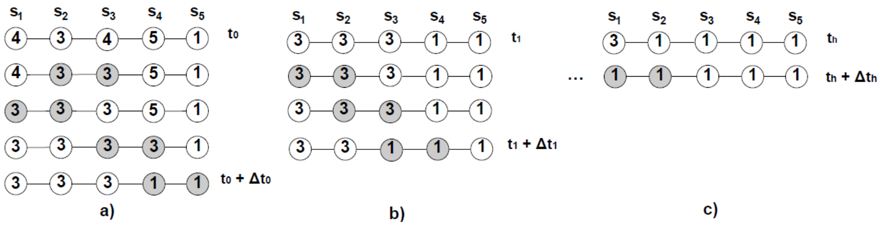


Figure 1. Steady-state convergence at different time intervals.

the number inside each circle represents the weight function set value $w_i(t_i, \gamma_a)$ of a generic set $\gamma_a \in 2^\Omega \setminus \Omega$ associated with agent i.

Now consider a time interval $[t_0, t_0 + \Delta t_0]$. Without any loss of generality, consider $|W| = N - 1$ and $|U| = 1$ at time t_0. At the exact time $t_0 + \Delta t_0$, two agents $v_u \in U$ and $v_w \in W$ communicate during the last iteration so that $|W| = N - 2$ and $|U| = 2$, which renders the graph $\mathcal{G}'(t_0 + \Delta t_0) = \{U \cup W,\ \mathcal{E}'(t_0, t_0 + \Delta t_0)\}$ connected.

Now consider a new time interval $[t_1, t_1 + \Delta t_1]$. At time t_1, $|W| = N - 2$ and $|U| = 2$. Now consider a new time interval $[t_h, t_h + \Delta t_h]$. At time t_h, $|W| = N-(h+1)$ and $|U| = h+1$. By iterating using the same reasoning for the edge selection policy, at the exact time $t_{N-1} + \Delta t_{N-1}$, two agents $v_u \in U$ and $v_w \in W$ communicate during the last iteration so that $|W| = 0$ and $|U| = N$, which renders the graph $\mathcal{G}'(t_{N-1}+\Delta t_{N-1}) = \{U \cup W,\ \mathcal{E}'(t_{N-1}, t_{N-1}+\Delta t_{N-1})\}$ connected. At this point, all the agents v_i for $i = 1..N$ have reached the same state $s(t') = \{\overline{w}(t', \gamma_a);\ \gamma_a \in 2^\Omega \setminus \Omega\}$. Therefore, $s(t')$ is at steady state in the multi-agent system. Since only one agent is updated during each time interval Δt_i, $d \cdot \Delta t_i$ (where d is the diameter of the network) is the number of time intervals $[t_i, t_i + \Delta t_i]$ after which all the nodes are updated. □

Lemma 2: Consider an edge selection policy $\mathfrak{e}$ such that $\forall\ t$ there exists a $\Delta t \in \mathbb{N}$ where $\mathcal{G}(t, t + \Delta\, t)$ is connected. If $\forall\ t$ there exists a time $M \in \mathbb{N}\ :\ \Delta\ t < M$, then any agent converges by $t = d \cdot M$ where d is the diameter of the network.

Proof: The proof follows directly from Lemma 1. In particular, recall that, in the worst-case scenario involving the topology, the network exhibits the largest diameter d so that $d = N - 1$ and $d \cdot \Delta t_i$ is the number of time intervals $[t_i, t_i + \Delta t_i]$ after which all the nodes are updated. Assuming that an upper bound M is available on the time required for the network to be connected during each time interval $[t_i, t_i + \Delta t_i]$, then the time required to update one agent is $t_i = M$. By iterating using the same reasoning, the process takes $t = d \cdot M$ to update all the agents. Therefore, the overall time required for the algorithm to converge in the worst-case scenario for the topology is linear with respect to the diameter of the network topology $\mathcal{G}$. □

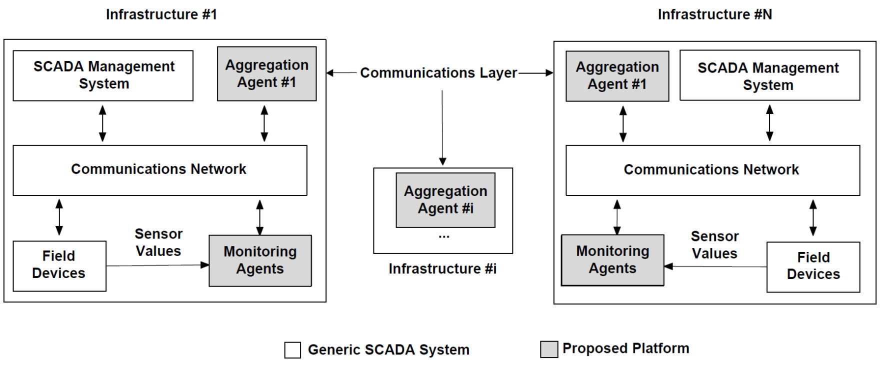


Figure 2. Communications framework between infrastructures in the case study.

4. Case Study

The case study involves a set of critical infrastructures in which the occurrence of certain conditions are monitored by a set of n agents. The infrastructures, which are geographically distributed, generically represent the infrastructure assets of a city. Each infrastructure is monitored by a SCADA system which includes: (i) a SCADA management system (e.g., human-machine interface and historian) located in a SCADA control center for monitoring and controlling field equipment; (ii) field devices (i.e., sensors, programmable logic controllers, remote terminal units and intelligent electronic devices) that acquire and transmit process parameter values and implement control actions; and (iii) a communications network that supports message exchange between field devices and the SCADA management system. Although critical infrastructures exhibit different kinds of dependencies, they generally do not share information. In contrast, this case study assumes that each infrastructure is able to produce information about the possible causes of faults in the infrastructure. This information, which is produced by monitoring agents deployed on the field devices, is exchanged among the agents (i.e., infrastructures).

The information shared among the infrastructures is provided by two kinds of agents as shown in Figure 2: (i) monitoring agents that acquire measurements from the field devices (physical events, cyber events and physical security events) and; (ii) aggregation agents that assist a SCADA management system by collecting the information from neighboring monitoring agents and distributing the information to peer agents in the other infrastructures. The resulting system is modeled as a multi-agent platform for distributed data aggregation in which each agent produces a basic belief assignment associated with the cause of a critical event that can affect the functionality of an infrastructure.

The monitoring agents, which detect physical and cyber events, are connected to the aggregation agents via the SCADA communications network of the associated infrastructure. Monitoring agents detect events by leveraging a security patrol, which is shared by the infrastructures to identify wireless intruders that are proximal to the monitored infrastructures. A monitoring agent sends alarm condition notifications via wireless communications using the nearest infrastructure communications network. Every agent executes the algorithm presented in Section 3 in order to evaluate the direct confidence and to communicate with its neighboring nodes. Communications between the aggregation agents use virtual private network (VPN) links. Note that aggregation agents only aggregate information; their direct confidence values depend only on their neighboring nodes. The convergence of monitoring and aggregation agents occurs in finite time steps, providing the most credible cause(s) of a fault. According to Lemmas 1 and 2, algorithm convergence in finite time is ensured for any edge selection policy $\mathfrak{e}$ that produces a connected graph. This property is especially important in a disaster environment (such as the considered scenario), where communications paths between pairs of nodes may be unavailable.

Since the security patrol is mobile, its links with peer aggregation agents change over time. For simplicity, it is assumed that the security patrol is connected to only one aggregation agent during each time step – in any case, according to Lemma 1, a violation of this assumption does not affect the convergence of the algorithm.

The proposed approach is distributed because it can be implemented in a network without a central node. It differs from the centralized approaches that are typically employed in traditional SCADA system architectures. In a centralized approach, the SCADA management system gathers and correlates events and security information originating from field equipment in order to detect malicious activities that are perpetrated in a distributed manner – such a centralized node is able to produce more accurate information about the state of the monitored system. In contrast, the proposed approach is distributed because the SCADA management system nodes (manifested by aggregation agents) act as neutral nodes with initial basic belief assignments such that $m(\Omega) = 1$. These nodes provide valuable information when they perform aggregations in conjunction with other information agents.

4.1 Problem Formulation

The case study considers four interdependent critical infrastructures that can be affected by failures and/or threats. Each infrastructure is able to produce one or more basic belief assignments from physical, cyber and physical security events detected by the monitoring agents. The frame of discernment is $\Omega = \{a, b, c, d\}$ where a denotes a possible physical failure, b a possible cyber intrusion or attack, c a possible physical security threat and d a normal functioning level.

Figure 3 presents the case study scenario derived from [10], which involves a dam that feeds a hydroelectric power station that, in turn, feeds a power distribution substation through a transmission network (not modeled for simplicity). A base transceiver station (BTS) provides telecommunications services and receives electricity from the power distribution station. The dam provides

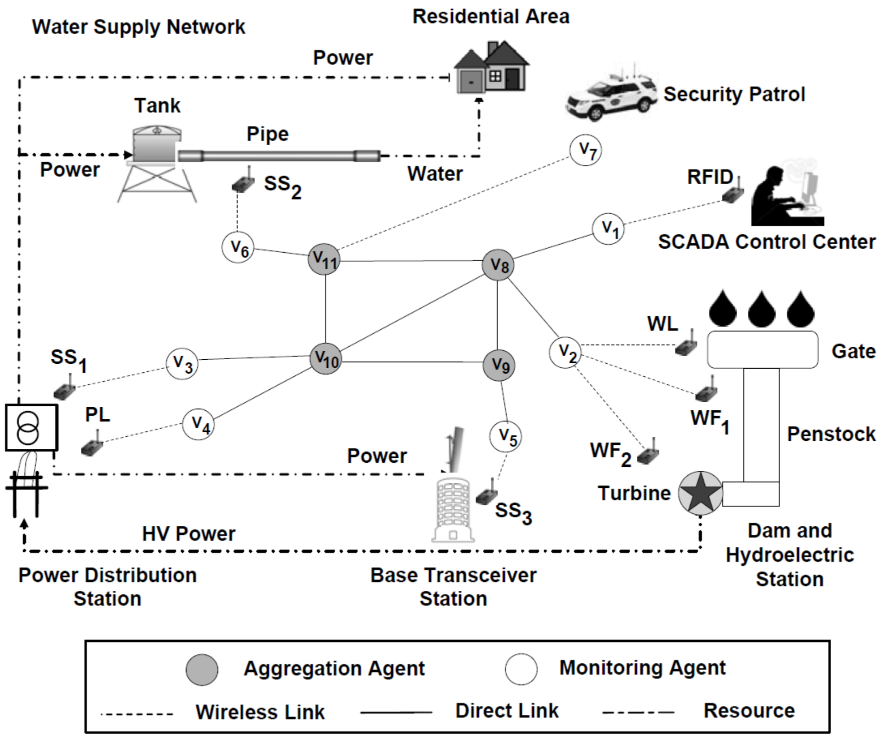


Figure 3. Case study scenario.

water to the hydroelectric power station through a gate that is remotely controlled to release basin water and activate the power plant turbine. The water supply network feeds the water pumps and automation devices, all of which receive electricity from the power distribution station. Infrastructure failures potentially cause societal and economic disruptions in the city district.

Table 1 shows the monitoring agents considered in each infrastructure. The following sections present practical methods that can be implemented by the agents to generate the relative basic belief assignments from sensor measurements. Also, the convergence of the algorithm starting with an initial set of basic belief assignments is demonstrated.

4.2 Dam and Hydroelectric Power Station

The dam and hydroelectric power station are controlled by a SCADA system that utilizes a wireless sensor network. Water fed to the hydroelectric power station is conveyed through pipes called penstocks. Agent v_1 is configured to monitor unauthorized physical access to the SCADA control room of the dam. In particular, the agent receives information wirelessly from a radio frequency identification (RFID) door sensor installed in the SCADA control room and

Table 1. Monitoring agents considered in each infrastructure.

Infrastructure	**Agents**
Dam and Hydroelectric Station	v_1, v_2
Power Distribution Station	v_3, v_4
Base Transceiver Station	v_5
Water Supply Network	v_6
Security Patrol	v_7

sends alerts about possible intrusions that could impact the proper functioning of the dam.

Table 2. Basic belief assignment generated by agent v_1.

A	$\mathbf{m_1(A)}$	
	Door Closed	**Door Open**
d	0.9	–
ac	–	0.3
bc	–	0.4
abc	–	0.2
Ω	0.1	0.1

When modeling the basic belief assignment of agent v_1, it is necessary to consider the possibility that an intruder with access to the SCADA control room may be able to launch a cyber attack (Table 2).

Agent v_2 periodically monitors the water flow rates and water levels measured by the sensors and uses them to check for security violations that could cause a turbine control malfunction. As explained in [10], two conditions hold in a generic dam under normal conditions: (C1) the difference between the water flow rates as measured by two water flow sensors located at the extremes of the penstock (WF_1 and WF_2 in the scenario) should disappear within about three seconds; and (C2) the variation in the water level in the basin of the dam (WL in the scenario) should be consistent with the variations in the incoming and outgoing water flows. Although the violation of each individual condition cannot be considered to be a consequence of a cyber attack, but rather a physical failure, violations of both conditions can increase the credibility of a cyber attack. In fact, a possible attack scenario involves compromises of the water flow sensors in order to hide changes in the water flow rates in the penstock. Thus, the basic belief assignment generated by agent v_2 combines the verification/violations of the two security conditions (Table 3).

Table 3. Basic belief assignment generated by agent v_2.

A	$\mathbf{m_2(A)}$			
	C1, C2	**¬C1, C2**	**C1, ¬C2**	**¬C1, ¬C2**
d	0.9	–	–	–
ab	–	–	0.2	0.3
ac	–	0.3	0.1	–
ad	–	0.1	0.5	–
bc	–	0.2	–	0.5
abc	–	0.1	–	–
Ω	0.1	0.3	0.2	0.2

4.3 Power Distribution Station

Earthquakes and hurricanes are known to have devastating effects on power distribution systems. Thus, reinforced concrete, fire- and explosion-resistant walls or barriers are installed between major pieces of equipment such as transformers, circuit breakers and regulators housed in power distribution facilities. Torrential rains caused by hurricanes can affect distribution systems more severely than generation and transmission systems. Floods caused by heavy rainfall can damage low-hanging lines in a power distribution system and cause power disruptions.

Agents v_3 and v_4 provide early warnings about possible physical faults induced by seismic events and floods, respectively. Agent v_3 obtains peak ground acceleration (PGA) data from a seismic sensor SS_1 installed in the substation building and estimates the credibility of a physical fault on the power distribution station based on the structural properties of the building. The building is assumed to have one story and to be a recent reinforced concrete construction. These properties are associated with a seismic vulnerability index $I_v = 0$ from a value range of –6 to 60. Agent v_3 transforms the peak ground acceleration of a seismic event to a microseismic intensity index I_{MCS} using the following equation for a building with the properties mentioned above [4]:

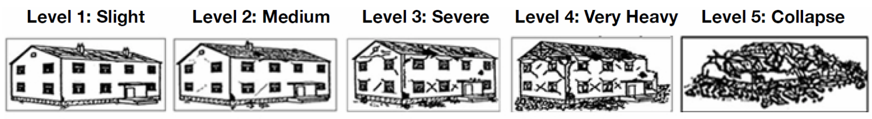
$$log(PGA) = 0.594 + 0.197 I_{MCS} \tag{8}$$

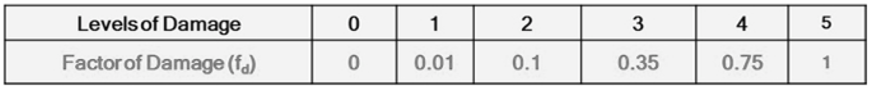
The following equations from [11] relate I_{MCS} and I_v to the mean damage d to the building (value range of 0 to 5) and the corresponding damage factor f_d (value range of 0 to 1):

$$d = 0.5 + 0.45(\arctan(0.55(I_{MCS} - 10.2 + 0.05 I_v)) \tag{9}$$

$$f_d = d^{1.75} \tag{10}$$

Based on these equations and the computed I_{MCS} and I_v values, agent v_3 can calculate the damage factor f_d (Figure 4) and estimate the credibility of a





Levels of Damage	0	1	2	3	4	5
Factor of Damage (f_d)	0	0.01	0.1	0.35	0.75	1

Figure 4. Expected damage [11, 15].

Table 4. Basic belief assignment generated by agents v_3 and v_5.

A	$\mathbf{m_3(A)}$					
	$\mathbf{f_d = 0}$	$\mathbf{f_d = 1}$	$\mathbf{f_d = 2}$	$\mathbf{f_d = 3}$	$\mathbf{f_d = 4}$	$\mathbf{f_d = 5}$
a	–	–	–	–	0.3	0.7
d	0.9	0.4	–	–	–	–
ab	–	0.3	0.4	0.5	0.6	0.2
ac	–	–	0.1	0.2	–	–
ad	–	–	0.2	–	–	–
Ω	0.1	0.3	0.3	0.3	0.1	0.1

physical fault affecting the power distribution station. Table 4 shows the basic belief assignment generated by agent v_3 based on the damage factor f_d.

Agent v_4 evaluates the rain precipitation in real time using a pluviometer sensor located in the substation; this data is used to assess the possible effects of flooding on the functionality of the substation. To accomplish this, a hot-spot analysis was conducted over a two-year period for a residential urban area. Linear regression analysis was then performed between the average frequency of disconnections at a specific electrical substation and the amount of rain precipitation in the area of the substation. Data relative to the rain precipitation was provided by a pluviometer installed near the substation.

The linear regression results in Figure 5 reveal a high correlation between the average frequency of disconnections and the amount of rain precipitation. This suggests that the amount of rain precipitation is a reliable predictor of the disconnection frequency and, therefore, provides a metric for specifying the basic belief assignment for agent v_4. Table 5 presents the credibility levels of physical faults in the substation of interest based on the daily quantity of rain precipitation denoted by Q (in mm).

4.4 Base Transceiver Station

A large number of base transceiver stations are installed in cities, often on the rooftops of buildings. This makes a base station vulnerable because an earthquake could damage the building housing the station, disrupting telecom-

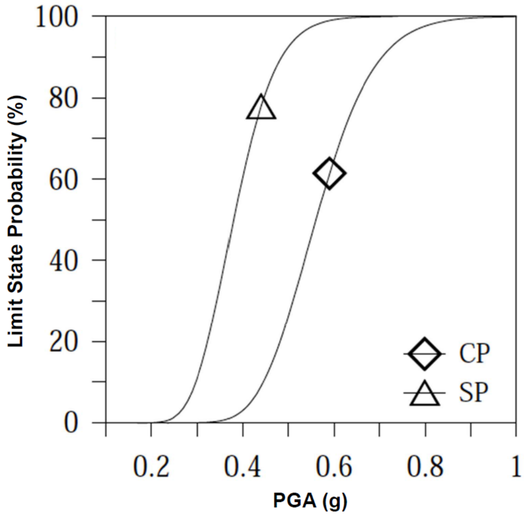


Figure 5. Correlation between the disconnection frequency and rain precipitation.

Table 5. Basic belief assignment generated by agent v_4.

A	$\mathbf{m_4(A)}$			
	$\mathbf{Q \leq 20}$	$\mathbf{20 < Q \leq 35}$	$\mathbf{35 < Q \leq 50}$	$\mathbf{Q > 50}$
a	–	–	–	0.6
d	0.9	0.3	–	–
ab	–	0.5	0.4	0.2
ac	–	–	0.1	–
ad	–	–	0.2	–
Ω	0.1	0.2	0.3	0.2

munications services in the area. Thus, agent v_5 could use the peak ground acceleration from the seismic sensor SS_2 installed in the building housing the base station to estimate the possible damage to the station based on the structural properties of the building. A building with the same structural properties as in Section 4.3, but with five stories (consistent with common base station installations) is considered. These properties can be associated with a seismic vulnerability index $I_v = 20$ according to Equations (9) and (10). Thus, agent v_5 is assigned the same basic belief assignment as agent v_3 in order to relate the damage level of the building to the occurrence of a physical failure to the base transceiver station.

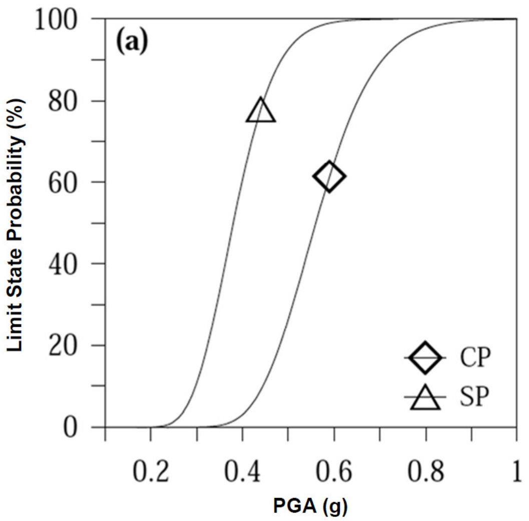


Figure 6. Fragility curve of a segmented pipeline [12].

4.5 Water Supply Network

Earthquakes are the most serious natural threat to a water supply network. They cause multiple types of damage to pipelines (e.g., longitudinal cracks, circumferential cracks and compression joint breaks) that result in severe water supply disruptions. To detect the effects of seismic events on the water supply network, agent v_6 acquires peak ground acceleration data from seismic sensor SS_3 installed in a pipeline that serves a residential area. The monitored segmented pipeline is assumed to have brittle iron pipes with bell-and-spigot joints, which is typical in water supply networks. The basic belief assignment generated by agent v_6 is determined by using the fragility curve for the specified pipeline as defined in [12] and diagrammed in Figure 6.

Table 6 shows the basic belief assignment generated by agent v_6. The value is proportional to the severity of the peak ground acceleration detected during an earthquake.

4.6 Security Patrol

The security patrol manifested by agent v_7 is based on the security vehicle prototype presented in [14]. The vehicle has equipment that detects wireless threats using the standard war-driving technique. The vehicle drives around a facility to collect and analyze wireless network traffic in order to detect potential intruders that are proximal to the facility. Predetermined security procedures are performed when threats are detected.

The security patrol vehicle can detect cyber and physical security threats. However, defining the basic belief assignment policy for this agent is application-

Table 6. Basic belief assignment generated by agent v_6.

A	$\mathbf{m_6(A)}$				
	$\mathbf{PGA \leq 0.2}$	$\mathbf{0.2 < PGA \leq 35}$	$\mathbf{35 < PGA \leq 50}$	$\mathbf{35 < PGA \leq 50}$	$\mathbf{PGA > 50}$
a	–	–	–	0.3	0.7
b	–	–	–	–	–
d	0.9	0.5	0.1	–	–
ab	–	0.3	0.5	0.5	0.2
ac	–	–	0.2	0.1	–
bc	–	–	–	–	–
Ω	0.1	0.2	0.2	0.1	0.1

dependent because it requires a deep analysis of the environment monitored by the vehicle and the wireless traffic data. Indeed, to quantify the credibility of cyber and physical security threats in the form of a basic belief assignment, several properties should be considered, including the signal strength of the emitter and the number of packets collected. The stronger the signal, the more accurate the location of a potential intruder. Moreover, the greater the number of wireless packets collected, the more likely it is that a cyber attack would be discovered.

In this scenario, it is assumed that the security patrol can, at each time step, specify the credibility of cyber and physical security threats in terms of a basic belief assignment. This information is communicated periodically via a wireless connection to the aggregation agent of the infrastructure of interest.

4.7 Numerical Example

This section presents the results of executing the distributed data fusion algorithm for a specific set of basic belief assignments and a random topology generated at each time step. In order to establish algorithm convergence based on Lemma 1, the edge selection policy $\mathfrak{e}$ generated a random connected graph at each time step, where the edges between the agents may or may not have existed and the security patrol was connected to an aggregation agent that changed over time. This policy complies with the assumptions underlying the application of the algorithm in a disaster scenario where the communications network may undergo temporary or permanent disconnections.

Table 7 shows the set of basic belief assignments at time $t = 0$ corresponding to the network topology shown in Figure 3. It is assumed that the security patrol provides the same alarm conditions over time as agent m_7 in Table 7 when it monitors the water supply network. However, the security patrol provides no information (i.e., $m(\Omega) = 1$) when it monitors other infrastructures. The last row in Table 7 shows the convergent basic belief assignment $\overline{m}(\gamma_a)$ at time $\overline{t} = 106$, which was obtained using the weight function $\overline{w}(\gamma_a)$ as described in Section 2. The basic belief assignments $\overline{m}(\gamma_a)$ exhibit the highest credible values corresponding to the occurrence of a physical fault that affects the con-

Table 7. Example of initial basic belief assignments for agents v_1 to v_{11}.

BBA	$\emptyset$	**a**	**b**	**c**	**ab**	**ac**	**ad**	**bc**	**abc**	$\mathbf{\Omega}$
$m_1(0)$	–	–	–	–	–	–	–	–	–	1
$m_2(0)$	–	–	–	–	–	0.3	0.1	0.2	0.1	0.3
$m_3(0)$	–	–	–	–	0.5	0.2	–	–	–	0.3
$m_4(0)$	–	–	–	–	0.4	0.1	0.2	–	–	0.3
$m_5(0)$	–	–	–	–	0.5	0.2	–	–	–	0.3
$m_6(0)$	–	–	–	–	0.5	0.3	–	–	–	0.2
$m_7(0)$	–	–	–	–	–	–	–	0.3	0.4	0.3
$m_{8-11}(0)$	–	–	–	–	–	–	–	–	–	1
$\overline{m}(\bar{t})$	0.22	0.41	0.06	0.03	0.12	0.08	–	–	0.04	0.04

sidered infrastructures. It is worth noting that the same convergent basic belief assignments are obtained using a centralized approach where all the basic belief functions, expressed as weight functions, are aggregated using the cautious operator.

5. Conclusions

The proposed distributed data fusion algorithm based on the transferable belief model is designed to provide situational awareness in critical infrastructures with link failures. A key result is that the algorithm converges in finite time for any connected network topology when the cautious rule of combination is used for data aggregation. The application of the algorithm to a realistic scenario involving interdependent critical infrastructures demonstrates its utility as an information sharing methodology. Information sharing among infrastructures is extremely useful for decision making during emergency situations, enabling the understanding of the most credible causes of service degradation and the implementation of timely countermeasures.

Future research will focus on integrating additional agents to address a broader range of events that affect infrastructure assets (e.g., lightning and landslides).

References

[1] S. Boyd, A. Ghosh, B. Prabhakar and D. Shah, Randomized gossip algorithms, *IEEE Transactions on Information Theory*, vol. 52(6), pp. 2508–2530, 2006.

[2] F. Caldeira, M. Castrucci, M. Aubigny, D. Macone, E. Monteiro, F. Rente, P. Simoes and V. Suraci, Secure mediation gateway architecture enabling communications among critical infrastructures, *Proceedings of the Future Network and Mobile Summit*, 2010.

[3] J. Cortes, Finite-time convergent gradient flows with applications to network consensus, *Automatica*, vol. 42(11), pp. 1993–2000, 2006.

[4] L. Decanini and F. Mollaioli, Formulation of elastic earthquake input energy spectra, *Earthquake Engineering and Structural Dynamics*, vol. 27(12), pp. 1503–1522, 1998.

[5] A. Dempster, A generalization of Bayesian inference, *Journal of the Royal Statistical Society, Series B (Methodological)*, vol. 30(2), pp. 205–247, 1968.

[6] A. Dempster, Upper and lower probabilities induced by a multivalued mapping, in *Classic Works of the Dempster-Shafer Theory of Belief Functions*, R. Yager and L. Liu (Eds.), Springer, Berlin-Heidelberg, Germany, pp. 57–72, 2008.

[7] T. Denoeux, Conjunctive and disjunctive combination of belief functions induced by nondistinct bodies of evidence, *Artificial Intelligence*, vol. 172(2–3), pp. 234–264, 2008.

[8] B. Ducourthial, V. Cherfaoui and T. Denoeux, Self-stabilizing distributed data fusion, in *Stabilization, Safety and Security of Distributed Systems*, A. Richa and C. Scheideler (Eds.), Springer, Berlin, Germany, pp. 148–162, 2012.

[9] F. Flammini, A. Gaglione, N. Mazzocca, V. Moscato and C. Pragliola, Wireless sensor data fusion for critical infrastructure security, *Proceedings of the International Workshop on Computational Intelligence in Security for Information Systems*, pp. 92–99, 2008.

[10] V. Formicola, A. Di Pietro, A. Alsubaie, S. D'Antonio and J. Marti, Assessing the impact of cyber attacks on wireless sensor nodes that monitor interdependent physical systems, in *Critical Infrastructure Protection VIII*, J. Butts and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 213–229, 2014.

[11] S. Lagomarsino and S. Giovinazzi, Macroseismic and mechanical models for the vulnerability and damage assessment of current buildings, *Bulletin of Earthquake Engineering*, vol. 4(4), pp. 415–443, 2006.

[12] G. Lanzano, E. Salzano, F. Santucci de Magistris and G. Fabbrocino, Vulnerability of pipelines subjected to permanent deformation due to geotechnical co-seismic effects, *Chemical Engineering Transactions*, vol. 32, pp. 415–420, 2013.

[13] C. Pathirage, D. Amaratunga, R. Haigh and D. Baldry, Knowledge sharing in disaster management strategies: Sri Lankan post-tsunami context, *Proceedings of the CIB World Building Congress*, pp. 2981–2993, 2007.

[14] I. Patterson, J. Nutaro, G. Allgood, T. Kuruganti and D. Fugate, Optimizing investments in cyber-security for critical infrastructure, *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, article 20, 2013.

[15] M. Pollino, G. Fattoruso, L. La Porta, A. Della Rocca and V. James, Collaborative open source geospatial tools and maps supporting response planning for disastrous earthquake events, *Future Internet*, vol. 4(2), pp. 451–468, 2012.

[16] G. Shafer, *A Mathematical Theory of Evidence*, Princeton University Press, Princeton, New Jersey, 1976.

[17] P. Smets, The combination of evidence in the transferable belief model, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12(5), pp. 447–458, 1990.

[18] D. Sutton, J. Harrison, S. Bologna and V. Rosato, The contribution of NEISAS to EP3R, in *Critical Information Infrastructure Security*, S. Bologna, B. Hammerli, D. Gritzalis and S. Wolthusen (Eds.), Springer-Verlag, Berlin Heidelberg, Germany, pp. 175–186, 2013.

[19] L. Zadeh, On the Validity of Dempster's Rule of Combination of Evidence, Memorandum UCB/ERL-M, Electronics Research Laboratory, University of California, Berkeley, Berkeley, California, 1979.

Chapter 7

EXPLOITING WEB ONTOLOGIES FOR AUTOMATED CRITICAL INFRASTRUCTURE DATA RETRIEVAL

Luca Galbusera and Georgios Giannopoulos

Abstract Semantic web technologies play a significant role in many open data initiatives, including geo-mapping projects and platforms. At the same time, semantic principles are also promoted as a key enabling factor for multi-domain analyses of critical infrastructures as well as for improved emergency response. This chapter reviews the recent literature on ontology-based analysis and management of critical infrastructures, and proposes the use of ontology processing techniques to bridge the gap between infrastructure knowledge representation and available (often general-purpose) open data sources. In particular, it discusses an approach for matching a given critical infrastructure ontology to an ontology built on OpenStreetMap (OSM) tags that enables structured access to the associated geographical dataset.

Keywords: Critical infrastructures, web ontologies, open data, OpenStreetMap

1. Introduction

Directive 2003/98/EC [19] and revised Directive 2013/37/EU [20] are fundamental policy references for the implementation of European Union initiatives on public data sharing. Furthermore, Commission Decision 2011/833/EU [18] enforces the principles of extended accessibility, economy of access to data, reusability and expanded data reach. Accordingly, the European Union Open Data Portal [22] was established as "the single point of access to a growing range of data from the institutions and other bodies of the European Union (EU)." The portal offers searching, exploration and downloading functionalities, and supports semantic technologies (e.g., SPARQL queries) based on linked data principles. Data is free for use and reuse for commercial and non-commercial purposes, and users can provide suggestions and feedback.

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 119–136, 2017.
https://doi.org/10.1007/978-3-319-70395-4_7

Directive 2007/2/EC [21] established the Infrastructure for Spatial Information in the European Community (INSPIRE), which promotes extensive geographic data sharing. The data infrastructure under development [30] has the objective to support European Union environmental and environment-related policies and activities by enabling data sharing and access by public organizations and the community. A series of implementation stages is planned for completion by 2021 [32]. Related legislation [31] focuses on data specifications, metadata, network services, data and service sharing, monitoring and reporting.

Several other governmental and non-governmental open data initiatives are also in place [44]. An interesting phenomenon is represented by the surge of voluntary open data projects, many of which focus on geographic knowledge. Ballatore et al. [3] provide an overview of a number of these efforts, mainly focusing on global-scoped and mostly crowdsourced projects based on popular semantic web formats specific to geographic data and to broader knowledge realms.

Over time, the original concepts informing the creation of the first geographic portals have been evolving towards a next-generation vision involving "multiple connected infrastructures based on open access and participation across multiple technological platforms that will address the needs of different audiences" [26]. Additionally, Digital Earth [27] is expected to evolve into "a digital nervous system of the globe, actively informing about events happening on (or close to) the Earth's surface by connecting to sensor networks and situation-aware systems" [26]. Semantic heterogeneity is acknowledged as one of the key challenges to achieving this objective. Semantic web principles are being applied to address this issue and the Semantic Geospatial Web initiative, supported by the W3C Geospatial Semantic Web Community Group [78], promotes the use of geospatial ontologies, semantic gazetteers and geographic vocabularies.

Semantic-web-oriented effort are accompanying the development of OpenStreetMap (OSM) [46], a collaborative mapping project that aggregates geographic information collected on a voluntary basis. OpenStreetMap data is organized into different element types and users complement them with freely-chosen tags in order to associate meaning and supporting information with geographic items. Each tag is defined by a (key, value) pair and, while the use of existing tags is encouraged, contributors are allowed to introduce new tags.

As a result of this intrinsic flexibility, the set of tags in use dynamically evolves in time along with geographic entries. Therefore, in addition to the tag reference pages provided by the OpenStreetMap Wiki [49], projects such as Taginfo [48] exist to keep track of the tags currently represented in OpenStreetMap and to provide statistics about their usage. Methods and tools based on semantic web principles have been proposed to overcome tag heterogeneity and enable structured access to the OpenStreetMap dataset and related resources. The LinkedGeoData portal [38] accommodates OpenStreetMap-sourced dataset information and links to third-party projects in a semantic web format [2, 60].

A resource description framework (RDF) graph representation based on the OpenStreetMap Wiki contents is provided by the OpenStreetMap Semantic Network [47] that also maps OpenStreetMap tags to corresponding concepts in the WordNet lexical database [77] and LinkedGeoData [3]. An ontology constructed over the set of OpenStreetMap tags is provided by the OSMonto Project [50], enabling hierarchically structured access to tags and providing a baseline reference to interface with other types of ontologies in order to perform, for instance, semantic analysis [8].

These efforts are part of a wide landscape of general ontology application scenarios. Uschold and Gruninger [68] identify the following four categories:

- **Neutral Authoring:** Information artifacts are authored in a unified, ontology-based language, supporting conversions to multiple target formats and ultimately overcoming interoperability constraints imposed by *ad hoc* approaches.

- **Common Access to Information:** Ontologies are exploited to translate information between various formats and representations.

- **Ontology-Based Specification:** Ontologies provide a basis for specifying and developing applications.

- **Ontology-Based Search:** Ontologies are used to support structured access to information repositories, enabling their organization and classification at appropriate levels of abstraction.

Ontologies can be expressed using formal languages, often based on first-order logic or descriptive logic. Researchers [13, 25] have examined the development of ontology languages, in general, and web ontology languages, in particular. The OWL Web Ontology Language is one of the most common standards in use today [1].

As discussed in this chapter, ontologies and semantic (web) technologies are also being promoted in the literature to support the analysis and management of critical infrastructures. For instance, they help provide a systematic representation of heterogeneous systems in terms of entities and their interconnections for study and simulation purposes. They can also portray threat types, targets and actors involved in disaster response, as well as enhance emergency management and information sharing [39]. The emergence of semantic-oriented approaches in the geospatial information and critical infrastructure protection communities provide opportunities to create enhanced critical infrastructure analysis and management solutions. Indeed, geo-information sources such as OpenStreetMap contain valuable critical infrastructure information and they stimulate community efforts to overcome crises [29, 58].

This chapter considers all the aspects discussed above and relates the use of ontological representations of critical infrastructure concepts (e.g., assets, threats and interdependencies) to information collection from data sources. Several ontology-based methods for critical infrastructure analysis and management are reviewed, and a method for critical infrastructure information retrieval from open data sources with an emphasis on OpenStreetMap is proposed. The method exploits ontology mapping as a means to interface an assumed ontology describing a critical infrastructure system to a second ontology constructed on the OpenStreetMap tag system.

2. Ontological Approaches

Ontologies play a significant role in the development of models of critical infrastructures and their management during and after adverse events. This section reviews recent literature in the areas of conceptual modeling of critical infrastructures, critical infrastructure simulation and information sharing.

2.1 Critical Infrastructure Modeling

Systematizing knowledge about critical infrastructures requires the establishment of a consistent semantics and a means for addressing the diversity that stems from an inherently multi-disciplinary investigation area. At the same time, it opens many opportunities for analysis, including automated reasoning and decision support. The construction of taxonomies is a fundamental step in this direction. Drawing on the work by Perrow [53], Rinaldi et al. [54] have proposed a seminal taxonomy of critical infrastructure elements and their interdependencies. Another notable work is the Infrastructure Data Taxonomy of the U.S. Department of Homeland Security [69] that has been used to guide and structure analyses [64]. Taxonomies have also been employed to categorize critical infrastructure threats and attacks (see, e.g., [35] for cyber-related threats).

Wolthusen [76] has proposed a method for representing critical infrastructure systems that is oriented towards data collection and exchange as well as modeling and simulation. The approach starts with a high level description of entities and dependencies, and exploits multigraphs to handle different types of dependencies. An ontological model and exchange mechanism data format are introduced based on RDF and OWL, and a multi-domain critical infrastructure representation sourced from expert knowledge is formalized.

Lee and Gandhi [36] have introduced an ontology-based active requirements engineering framework for software-intensive systems analysis based on a hierarchical representation that includes top-level generic requirements, mid-level domain spanning requirements and leaf-node subdomain requirements.

Sotoodeh and Kruchten [59] present a conceptual modeling framework for disaster management that comprises three ontologies: an emergency operation center ontology related to disaster response components and two disaster affecting infrastructure ontologies that represent infrastructures and their reference communities along with their relationships at a high level of abstraction. Concepts such as regions and people with associated wellness characterizations are included together with infrastructure and resource characterizations. Interdependencies are described at the physical and social levels.

Sicilia and Santos [57] have introduced an infrastructure incident assessment ontology for the high-level representation of infrastructures, incidents and their causes. It involves a service-dependent semantics of connections. Interdependencies (physical, connectivity-based, policy-based and procedure-based) can be specified *a priori* or inferred using Semantic Web Rule Language (SWRL) rules. The use of reasoning techniques to support emergency response is also demonstrated.

A network security framework has been developed by the INTERSECTION Project [11] for identifying and classifying vulnerabilities in heterogeneous networks [7]. The framework comprises an ontology that extends beyond single-domain networks and focuses on resources and vulnerabilities as the key components. Four resource subclasses are identified: (i) physical resources; (ii) logical resources; (iii) software; and (iv) services. In addition, three vulnerability subclasses are identified: (i) physical resource vulnerabilities; (ii) logical resource vulnerabilities; and (iii) software vulnerabilities. An OWL-based decision support architecture is presented as well.

An ontology handling tool created by the INSPIRE Project [10] and described in [4] is a standards-based instrument for enabling automatic audits of the security and criticality levels associated with information systems. Its infrastructure discovery component, ontology repository and expert visualization tool combine to facilitate analyses of critical infrastructure vulnerabilities while considering the associated information and communications technology components.

Creese et al. [14] have used automated reasoning for critical infrastructure resilience assessment problems based on a top-down, layered conceptual mapping of assets, controls, vulnerabilities and risk. An *ad hoc* dependency modeling language that exploits a natural-language-like semantics is used to enable automated reasoning and what-if analyses with a focus on organizational aspects.

El-Diraby and Osman [17] have used domain ontologies to depict urban infrastructures in terms of processes, actors and products. Physical products are classified into generic and sector-specific products. Infrastructure products are characterized in terms of the functions they perform (i.e., conveyance, control, protection, access, measuring, storage and locating products). The composition of products is allowed and an extended set of attributes (i.e., dimensional, spatial, material, shape, cost, performance, surrounding soil, dependency, redundancy and state of operation attributes) is specified for each product. The construction of the ontology is subject to formal consistency checks and an expert elicitation-based assessment.

De Nicola et al. [15] have presented an ontology and semantic rules for emergency management in smart cities. CEML is used as the reference modeling language. The knowledge modeling strategy employs an upper-level ontology that extends the domain ontology with CEML concepts, along with an emergency ontology that enables automated knowledge management and reasoning. A tool exploiting SPARQL provides automated support for defining emergency management plans.

Xu et al. [79] employ geo-ontologies for earthquake emergency response. Knowledge is organized into four classes: (i) factual knowledge; (ii) rule-based knowledge; (iii) procedural knowledge; and (iv) meta-knowledge. The content types include emergency response and rescue knowledge, disaster information estimation, emergency information and terms, and emergency foundation data.

Takahashi and Kadobayashi [62] provide a list of industry specifications and a reference ontology for cyber security operational information. Conceptual models that target cyber dependencies are considered in [45], where a human factors ontology is employed to specify a cyber security framework. Other researchers [66, 67] discuss an ontology-based approach for vulnerability and interdependency representation as well as disruption scenario generation for critical infrastructures. Luo et al. [40] have developed a knowledge modeling formalism for emergency situations and planning in metropolitan areas, and have implemented it in a training tool.

2.2 Critical Infrastructure Simulation

Ontologies are widely used in critical infrastructure simulation architectures. In particular, they provide model specifications and enable the integration and combination of multiple analysis techniques and tools. Conceptual interoperability has been formally studied in the context of simulation theory. Wang et al. [75] have introduced a conceptual interoperability model for determining the degree of interoperability of a system. Various aspects of integration, interoperability, composability are discussed in [52].

Critical infrastructure taxonomies are leveraged in creating complex critical infrastructure simulators. For example, Tolk et al. [65] have presented a modeling and simulation development framework, and have used it in a case study involving the Infrastructure Data Taxonomy of the U.S. Department of Homeland Security [69].

Van Dam and Lukszo [71] have employed agent-based models for the energy and transportation infrastructures. Their bottom-up methodology, which is motivated by the presence of multiple decision makers, distributed nature of the problem and dynamic operational environments, involves a generic ontology that is customized to the domains of interest based on expert opinion.

The IRRIIS Project [12] has developed an ontological information model for vulnerability analyses of large and complex critical infrastructures [34]. It is implemented in a federated simulation environment and supports the development of a risk estimator for determining if specific conditions in an infrastructure are critical singly or in combination.

The DIESIS Project [9] has adopted a layered approach for federated critical infrastructure simulation. It employs ontologies to describe the dynamic bindings between subsystems [55]. An ontology component is used to express meta-knowledge (abstract representations of basic system concepts and relationships); this is accompanied by an infrastructure ontology (domain-specific critical infrastructure ontology) and a federation ontology (for specifying semantically-coherent interconnections and rules).

Masucci et al. [42] discuss the derivation of ontology components and their relationships using OWL and SWRL. Castorini et al. [5] describe an application involving the power grid, railway and telecommunication domains, and their mutual relationships. Interested readers are referred to [63, 70] for more details.

The I2Sim interdependency simulator is a key contribution to ontology-based simulations of critical infrastructures [41]. To support I2Sim development, an ontology is presented for modeling temporal dependencies between infrastructures based on tokens (goods and services provided by one entity to another), cells (entities that perform functions), nodes (token generators) and transportation channels (flows of tokens subject to capacity and time delay constraints).

Ventura et al. [73] expand on this approach and introduce a taxonomy for classifying infrastructure interdependencies based on criteria such as the nature of the involved entities (human-object, object-object or human-human), directions of relationships (unidirectional or bidirectional), nature of relationships (information, physical, geographical or organizational/human/societal), states of relationships (static or dynamic) and type of failure if disrupted (cascading failure in associated entities, escalating failure or common origin failure).

Grolinger et al. [28] explore ontologies associated with a water distribution simulator and power system simulator, and map them to the I2Sim ontology. According to Grolinger et al., while federated simulation approaches as used in the DIESIS Project attempt to "integrate existing domain simulators by enabling their coordination and collaboration," I2Sim belongs to the architectural type that "includes simulation frameworks that enable the modeling of different infrastructures and their interdependencies."

2.3 Information Sharing

Another research topic covers information sharing and the related interoperability concerns, especially the need to manage the diversity of data sources and formats in emergency response procedures that often involve a number of actors. The literature in this field is extensive and has strong relationships with the conceptual modeling and simulation of critical infrastructures. Some of the literature discussed above addresses this aspect as well. However, certain recent contributions related to information sharing remain to be discussed.

Kim et al. [33] present an information sharing mechanism based on ontologies that addresses cyber dependencies between infrastructures. Di Maio [16] has proposed an open ontology approach that improves the performance of emergency response systems based on the principle of collaboration. Di Maio also discusses different levels of conceptual interoperability and the important notion of resilience in emergency response systems.

The interoperability gap affecting emergency planning systems is addressed in [74] using an emergency planning ontology. This ontology, which is based on the suggested upper merged ontology, is formally specified in terms of concepts, relations, functions, axioms and instances.

Galton and Worboys [24] have proposed architectural specifications for interoperability that take into account sensor networks and crowdsourced information collection procedures, together with related spatio-temporal data distribution considerations. Galton and Worboys also note that open-source geospatial information can be very useful in an emergency management framework.

Li et al. [37] describe a cloud computing platform for emergency management that relies on crowdsourced information. Drawing on existing emergency management ontologies, Li et al. introduce a novel ontology that considers additional information such as the types of hazards and emergencies, as well as meteorological factors. Castrucci et al. [6] have developed a mediation system that enables secure communications between critical infrastructures, implements fault mitigation strategies and supports information discovery while overcoming information exchange and data heterogeneity problems.

3. Ontology-Based Information Retrieval

The literature review in the previous section covers conceptualizations of various aspects of critical infrastructure protection and emergency management. An important point is that structured descriptions of critical infrastructures can also enhance information retrieval processes. As a consequence, this section focuses on an information retrieval approach for critical infrastructures based on ontology matching or alignment [23, 56].

Ontology matching techniques seek to find relationships between elements of different ontologies under analysis and provide similarity measures. An automated procedure for performing the alignment is important when dealing with large ontologies, multi-step information exchange chains and real-time processing. The complexity of multiple ontology matching applications can be managed by combining the alignment criteria and the resulting similarity measures using expert judgment or artificial intelligence [72].

The proposed matching-based approach has three sub-tasks: (i) ontology population; (ii) ontology matching; and (iii) ontology-driven data retrieval. These sub-tasks are described below.

3.1 Ontology Population

In the first step, two starting ontologies CI_Ont and T_Ont are created and populated. CI_Ont is a domain ontology that describes a set of critical infrastructure components, threats and their relationships. T_Ont is a target ontology built on a reference dataset from which information is to be retrieved. The application focuses on the OpenStreetMap geographic dataset.

The following are the key aspects involved in constructing the two ontologies:

- **CI_Ont:** An OWL ontology CI_Ont is constructed to include classes that describe the set of critical infrastructure sectors, sub-sectors and asset types (i.e., critical infrastructure classes) relevant to the geographic information retrieval problem and classes that describe the set of threats (threat classes). Subclass relationships are established between the elements of the critical infrastructure classes based on general-purpose and specialized glossaries and taxonomies. Interdependency relationships are also specified between critical infrastructure classes based on information collected from the technical literature. Furthermore, threat-to-critical-infrastructure class relationships are specified to express the significance of threats to the various infrastructure elements.

- **T_Ont:** The T_Ont ontology is populated with tag information via a semi-automated process based on the Taginfo system. A set of significant reference keys (e.g., building, highway, natural, land use, surface, power, waterway, wall, amenity, leisure, railway) is first identified for the analysis domain of interest and consistent with CI_Ont. A set of values is extracted for each key from the same source based on a filtering criterion. As in the case of the OSMonto ontology, the filtering criterion only includes values used a sufficiently high number of times according to the statistics provided by Taginfo. The resulting keys and values are arranged into the OWL ontology T_Ont to express the hierarchical relationships between keys and their associated values.

3.2 Ontology Matching

An alignment is computed based on the CI_Ont and T_Ont ontologies and additional lexical resources. A preprocessing step and a multi-step alignment procedure are involved.

Preprocessing. In this step, the CI_Ont and T_Ont ontologies are processed to adhere to established orthographic rules and standards (e.g., hyphenation and capitalization of labels). Furthermore, CI_Ont undergoes an ontology enrichment stage. This is accomplished by defining a lexicon L_{CI_Ont} by collecting the labels associated with the CI_Ont classes. The lexicon is partitioned into $L^{CI}_{CI\_Ont}$ and $L^{th}_{CI\_Ont}$ by collecting the labels of the critical infrastructure and threat classes, respectively.

For each element in $L^{CI}_{CI\_Ont}$, a set of relevant synonyms is fetched from the WordNet database via an automated routine that uses the MIT Java Wordnet Interface [43]. Correspondingly, the enriched critical infrastructure ontology CI_Ont,e is constructed by extending CI_Ont with the synonym entries and establishing consistent equivalence relationships. The associated enriched lexicon $L_{CI\_Ont,e}$ collects the labels of the extended set of classes so that $L_{CI\_Ont} \subseteq L_{CI\_Ont,e}$. The lexicon $L_{CI\_Ont,e}$ is partitioned into $L^{CI}_{CI\_Ont,e}$ and $L^{th}_{CI\_Ont}$, where $L^{CI}_{CI\_Ont,e}$ collects the extended set of labels of the critical infrastructure classes in the enriched ontology. Finally, in the case of T_Ont, a lexicon L_{T_Ont} is created based on OpenStreetMap keys and values expressed in the ontology.

Multi-Step Alignment. The alignment procedure involves the following steps that are performed in sequence:

- **Lexical Matching:** Exact element matches between $L^{CI}_{CI\_Ont,e}$ and L_{T_Ont} are determined.

- **String Matching:** String similarity metrics are employed to compare terms from the two lexicons. One of the metrics used is the Levenshtein distance (e.g., [61]).

The two matching methods are applied sequentially so that the alignments found via lexical matching are included directly in the final result while string matching is used to search for additional relevant bindings. The final alignment is produced using a similarity aggregation criterion encoded in a matching matrix $\mathrm{M}(\mathrm{L}^{CI}_{CI\_Ont,e}, \mathrm{L}_{T\_Ont}) \in [0,1]^{(|L^{CI}_{CI\_Ont,e}|,|L_{T\_Ont}|)}$ based on similarity thresholding. To increase the confidence levels when applying string matching, it is possible to consider the presence of multiple matches among CI_Ont synonym classes and T_Ont components that are associated with OpenStreetMap tag values that refer to the same keys.

3.3 Data Retrieval

The inclusion of threat classes, threat-to-critical-infrastructure class relationships and interdependency relationships in CI_Ont enables the discovered alignment to be used for targeted data extraction. This is accomplished by defining a relationship map in terms of the following adjacency matrices:

- $A^{th\to CI} \in \{0,1\}^{(|L^{th}_{CI\_Ont}|,|L^{CI}_{CI\_Ont}|)}$ for all threat-to-critical-infrastructure class relationships, where $(i,j) = 1$ means that threat i affects critical infrastructure component j based on lexical indexing.

- $A^{CI\to CI} \in \{0,1\}^{(|L^{CI}_{CI\_Ont}|,|L^{CI}_{CI\_Ont}|)}$ for all critical infrastructure interdependency relationships.

- $A^{CI\to CI,e} \in \{0,1\}^{(|L^{CI}_{CI\_Ont}|,|L^{CI}_{CI\_Ont,e}|)}$ for all synonym relationships.

The three adjacency matrices are then combined with the matching matrix described above. The set of T_Ont items of interest is then obtained starting from each considered threat component. Thus, starting from a specified threat scenario and a geographical area of interest, it is possible to use the composition and query OpenStreetMap based on the significant (key, value) items that are found. The operation can be performed, for instance, by using the Overpass API [51] via the Overpass Turbo interface (see `overpass-turbo.eu`).

As an example, consider a scenario where waterways are affected by a specified threat and the road infrastructure may be affected due to the interdependencies. For both components, the matcher identifies a set of relevant (key, value) pairs to include in a query, whose output is presented in Figure 1. The identified pairs include several facilities that are related to the waterway and road sectors.

4. Conclusions

This research has focused on the use of semantic technologies in critical infrastructure analysis with an emphasis on information retrieval. Of particular interest has been the use of ontologies for critical infrastructure modeling

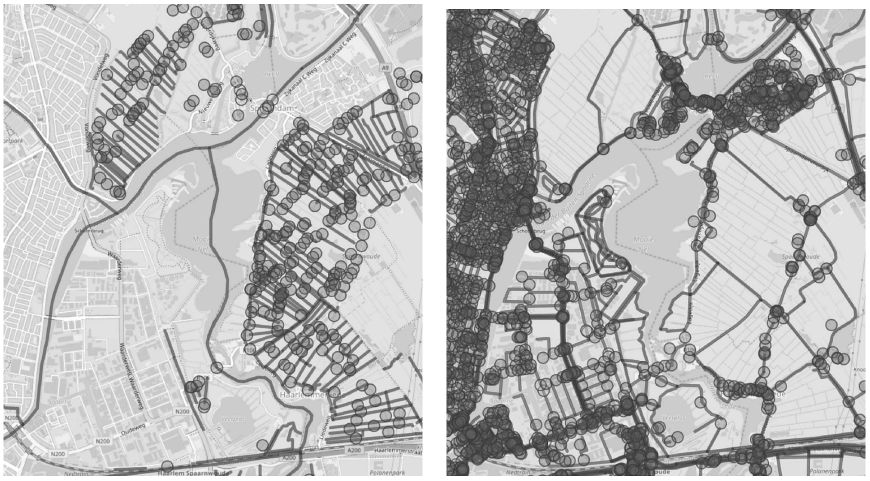


Figure 1. Query outputs for waterway (left) and road infrastructures (right).

and simulation as well as for emergency response. The proposed approach for ontology-based information retrieval from open geographic data sources (especially, OpenStreetMap) is applicable to critical infrastructure protection. The target ontology describing the OpenStreetMap content is constructed based on statistics about the actual use of tags, which evolves over time. An ontology specified for critical infrastructures incorporates threat and interdependency information and is enriched during processing. The matching procedure used for querying information is layered in terms of alignment methods and comprises lexical and string matching components. The principal contribution of this research is its ability to foster critical infrastructure tool integration, interoperability and composability. Moreover, the structured access to open-source information facilitated by the proposed approach can enhance multi-sector knowledge advancement, especially in conjunction with expertise from various technical fields.

Future research will incorporate interactive matching and special taxonomies and dictionaries devoted to critical infrastructures for ontology enrichment and improved similarity aggregation. Additionally, efforts will focus on inference mechanisms for incrementally improving the reference critical infrastructure ontology and on incorporating data quality checking mechanisms.

References

[1] G. Antoniou and F. van Harmelen, *A Semantic Web Primer*, MIT Press, Cambridge, Massachusetts, 2004.

[2] S. Auer, J. Lehmann and S. Hellmann, LinkedGeoData: Adding a spatial dimension to the web of data, *Proceedings of the Eighth International Semantic Web Conference*, pp. 731–746, 2009.

[3] A. Ballatore, D. Wilson and M. Bertolotto, A survey of volunteered open geo-knowledge bases in the semantic web, in *Quality Issues in the Management of Web Information*, G. Pasi, G. Bordogna and L. Jain (Eds.), Springer-Verlag, Berlin Heidelberg, Germany, pp. 93–120, 2013.

[4] M. Bouet and M. Israel, INSPIRE Ontology Handler: Automatically building and managing a knowledge base for critical information infrastructure protection, *Proceedings of the Twelfth IFIP/IEEE International Symposium on Integrated Network Management*, pp. 694–697, 2011.

[5] E. Castorini, P. Palazzari, A. Tofani and P. Servillo, Ontological framework to model critical infrastructures and their interdependencies, *Proceedings of the Complexity in Engineering Conference*, pp. 91–93, 2010.

[6] M. Castrucci, A. Neri, F. Caldeira, J. Aubert, D. Khadraoui, M. Aubigny, C. Harpes, P. Simoes, V. Suraci and P. Capodieci, Design and implementation of a mediation system enabling secure communications among critical infrastructures, *International Journal of Critical Infrastructure Protection*, vol. 5(2), pp. 86–97, 2012.

[7] M. Choras, R. Kozik, A. Flizikowski, R. Renk and W. Holubowicz, Ontology-based decision support for security management in heterogeneous networks, *Proceedings of the Fifth International Conference on Intelligent Computing*, vol. 2, pp. 920–927, 2009.

[8] M. Codescu, G. Horsinka, O. Kutz, T. Mossakowski and R. Rau, OSMonto – An ontology of OpenStreetMap tags, *Proceedings of the First European State of the Map Conference*, pp. 55–64, 2011.

[9] Community Research and Development Information Service, DIESIS Project, European Union Publications Office, Luxembourg City, Luxembourg (`cordis.europa.eu/project/rcn/92603_en.html`), 2017.

[10] Community Research and Development Information Service, INSPIRE Project, European Union Publications Office, Luxembourg City, Luxembourg (`cordis.europa.eu/project/rcn/87757_en.html`), 2017.

[11] Community Research and Development Information Service, INTERSECTION Project, European Union Publications Office, Luxembourg City, Luxembourg (`cordis.europa.eu/project/rcn/85347_en.html`), 2017.

[12] Community Research and Development Information Service, IRRIIS Project, European Union Publications Office, Luxembourg City, Luxembourg (`cordis.europa.eu/project/rcn/79319_en.html`), 2017.

[13] O. Corcho and A. Gomez-Perez, A roadmap for ontology specification languages, *Proceedings of the Twelfth European Workshop on Knowledge Acquisition, Modeling and Management*, pp. 80–96, 2000.

[14] S. Creese, M. Goldsmith and A. Adetoye, A logical high-level framework for critical infrastructure resilience and risk assessment, *Proceedings of the Third International Workshop on Cyberspace Safety and Security*, pp. 7–14, 2011.

[15] A. De Nicola, M. Melchiori and M. Villani, A lateral thinking framework for semantic modeling of emergencies in smart cities, *Proceedings of the International Conference on Database and Expert Systems Applications*, part 2, pp. 334–348, 2014.

[16] P. Di Maio, An Open Ontology for Open Source Emergency Response System, Mae Fa Luang University, Chiang Rai, Thailand (`www.academia.edu/610005/An_Open_Ontology_for_Open_Source_Emergency_Response_System`), 2007.

[17] T. El-Diraby and H. Osman, A domain ontology for construction concepts in urban infrastructure products, *Automation in Construction*, vol. 20(8), pp. 1120–1132, 2011.

[18] European Commission, 2011/833/EU: Commission Decision of 12 December 2011 on the Reuse of Commission Documents, Brussels, Belgium (`eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32011D0833`), 2011.

[19] European Parliament and Council of the European Union, Directive 2003/98/EC of the European Parliament and of the Council of 17 November 2003 on the Reuse of Public Sector Information, Brussels, Belgium (`eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32003L0098`), 2003.

[20] European Parliament and Council of the European Union, Directive 2003/98/EC of the European Parliament and of the Council of 17 November 2003 on the Reuse of Public Sector Information, Brussels, Belgium (`eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:02003L0098-20130717`), 2003.

[21] European Parliament and Council of the European Union, Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 Establishing an Infrastructure for Spatial Information in the European Community (INSPIRE), Brussels, Belgium (`eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:32007L0002`), 2007.

[22] European Union Publications Office, EU Open Data Portal, Luxembourg City, Luxembourg (`data.europa.eu/euodp`), 2017.

[23] J. Euzenat and P. Shvaiko, *Ontology Matching*, Springer-Verlag, Berlin Heidelberg, Germany, 2007.

[24] A. Galton and M. Worboys, An ontology of information for emergency management, *Proceedings of the Eighth International Conference on Information Systems for Crisis Response and Management*, 2011.

[25] A. Gomez-Perez and O. Corcho, Ontology languages for the semantic web, *IEEE Intelligent Systems*, vol. 17(1), pp. 54–60, 2002.

[26] M. Goodchild, H. Guo, A. Annoni, L. Bian, K. de Bie, F. Campbell, M. Craglia, M. Ehlers, J. van Genderen, D. Jackson, A. Lewis, M. Pesaresi, G. Remetey-Fulopp, R. Simpson, A. Skidmore, C. Wang and P. Woodgate, Next-generation Digital Earth, *Proceedings of the National Academy of Sciences*, vol. 109(28), pp. 11088–11094, 2012.

[27] A. Gore, *Earth in the Balance: Ecology and the Human Spirit*, Rodale Books, New York, 2006.

[28] K. Grolinger, M. Capretz, A. Shypanski and G. Gill, Federated critical infrastructure simulators: Towards ontologies for support of collaboration, *Proceedings of the Twenty-Fourth Canadian Conference on Electrical and Computer Engineering*, pp. 1503–1506, 2011.

[29] B. Herfort, M. Eckle, J. Porto de Albuquerque and A. Zipf, Towards assessing the quality of volunteered geographic information from OpenStreetMap for identifying critical infrastructures, *Proceedings of the Twelfth International Conference on Information Systems for Crisis Response and Management*, pp. 24–27, 2015.

[30] Infrastructure for Spatial Information in Europe, INSPIRE, Brussels, Belgium (`inspire.ec.europa.eu`), 2017.

[31] Infrastructure for Spatial Information in Europe, Legislation, Brussels, Belgium (`inspire.ec.europa.eu/inspire-legislation`), 2017.

[32] Infrastructure for Spatial Information in Europe, Roadmap, Brussels, Belgium (`inspire.ec.europa.eu/inspire-roadmap`), 2017.

[33] H. Kim, M. Biehl and J. Buzacott, M-CI^2: Modeling cyber interdependencies between critical infrastructures, *Proceedings of the Third IEEE International Conference on Industrial Informatics*, pp. 644–648, 2005.

[34] R. Klein, Information modeling and simulation in large dependent critical infrastructures – An overview of the European Integrated Project IRRIIS, *Proceedings of the International Workshop on Critical Information Infrastructures Security*, pp. 131–143, 2008.

[35] R. Kozik and M. Choras, Current cyber security threats and challenges in critical infrastructure protection, *Proceedings of the Second International Conference on Informatics and Applications*, pp. 93–97, 2013.

[36] S. Lee and R. Gandhi, Ontology-based active requirements engineering framework, *Proceedings of the Twelfth Asia-Pacific Software Enginering Conference*, 2005.

[37] J. Li, Q. Li, S. Khan and N. Ghani, Community-based cloud for emergency management, *Proceedings of the Sixth International Conference on System of Systems Engineering*, pp. 55–60, 2011.

[38] LinkedGeoData, LinkedGeoData – Adding a Spatial Dimension to the Web of Data, Institute for Informatics, University of Leipzig, Leipzig, Germany (`linkedgeodata.org/About`), 2017.

[39] S. Liu, C. Brewster and D. Shaw, Ontologies for crisis management: A review of the state of the art in ontology design and usability, *Proceedings of the Tenth International Conference on Information Systems for Crisis Response and Management*, pp. 349–359, 2013.

[40] H. Luo, X. Peng and B. Zhong, Application of ontology in emergency plan management of metro operations, *Procedia Engineering*, vol. 164, pp. 158–165, 2016.

[41] J. Marti, J. Hollman, C. Ventura and J. Jatskevich, Dynamic recovery of critical infrastructures: Real-time temporal coordination, *International Journal of Critical Infrastructures*, vol. 4(1-2), pp. 17–31, 2008.

[42] V. Masucci, F. Adinolfi, P. Servillo, G. Dipoppa and A. Tofani, Ontology-based critical infrastructure modeling and simulation, in *Critical Infrastructure Protection III*, C. Palmer and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 229–242, 2009.

[43] MIT Computer Science and Artificial Intelligence Laboratory, JWI 2.4.0, Massachusetts Institute of Technology, Cambridge, Massachusetts (`projects.csail.mit.edu/jwi`), 2015.

[44] A. Nicol, J. Caruso and E. Archambault, Open Data Access Policies and Strategies in the European Research Area and Beyond, Science-Metrix, Montreal, Canada, 2013.

[45] A. Oltramari, D. Henshel, M. Cains and B. Hoffman, Towards a human factors ontology for cyber security, *Proceedings of the Tenth International Conference on Semantic Technology for Intelligence, Defense and Security*, pp. 26–33, 2015.

[46] OpenStreetMap Foundation, OpenStreetMap, Sutton Coldfield, United Kingdom (`www.openstreetmap.org`), 2017.

[47] OpenStreetMap Foundation, OpenStreetMap Semantic Network, Sutton Coldfield, United Kingdom (`wiki.openstreetmap.org/wiki/OSM_Semantic_Network`), 2017.

[48] OpenStreetMap Foundation, OpenStreetMap Taginfo, Sutton Coldfield, United Kingdom (`taginfo.openstreetmap.org`), 2017.

[49] OpenStreetMap Foundation, OpenStreetMap Wiki, Sutton Coldfield, United Kingdom (`wiki.openstreetmap.org/wiki/Main_Page`), 2017.

[50] OpenStreetMap Foundation, OSMonto, Sutton Coldfield, United Kingdom (`wiki.openstreetmap.org/wiki/OSMonto`), 2017.

[51] OpenStreetMap Foundation, Overpass API, Sutton Coldfield, United Kingdom (`wiki.openstreetmap.org/wiki/Overpass_API`), 2017.

[52] E. Page, R. Briggs and J. Tufarolo, Toward a family of maturity models for the simulation interconnection problem, *Proceedings of the Spring Simulation Interoperability Workshop*, vol. 1, pp. 1059–1069, 2004.

[53] C. Perrow, *Normal Accidents: Living with High-Risk Technologies*, Princeton University Press, Princeton, New Jersey, 1999.

[54] S. Rinaldi, J. Peerenboom and T. Kelly, Identifying, understanding and analyzing critical infrastructure interdependencies, *IEEE Control Systems*, vol. 21(6), pp. 11–25, 2001.

[55] E. Rome, S. Bologna, E. Gelenbe and E. Luiijf, DIESIS: An interoperable European federated simulation network for critical infrastructures, *Proceedings of the European Simulation Interoperability Workshop*, pp. 139–146, 2009.

[56] P. Shvaiko and J. Euzenat, Ontology matching: State of the art and future challenges, *IEEE Transactions on Knowledge and Data Engineering*, vol. 25(1), pp. 158–176, 2013.

[57] M. Sicilia and L. Santos, Main elements of a basic ontology of infrastructure interdependency for the assessment of incidents, *Proceedings of the Second World Summit on the Knowledge Society: Visioning and Engineering the Knowledge Society*, pp. 533–542, 2009.

[58] R. Soden and L. Palen, From crowdsourced mapping to community mapping: The post-earthquake work of OpenStreetMap Haiti, *Proceedings of the Eleventh International Conference on the Design of Cooperative Systems*, pp. 311–326, 2014.

[59] M. Sotoodeh and P. Kruchten, An ontological approach to conceptual modeling of disaster management, *Proceedings of the Second Annual IEEE Systems Conference*, 2008.

[60] C. Stadler, J. Lehmann, K. Hoffner and S. Auer, LinkedGeoData: A core for a web of spatial open data, *Semantic Web*, vol. 3(4), pp. 333–354, 2012.

[61] G. Stoilos, G. Stamou and S. Kollias, A string metric for ontology alignment, *Proceedings of the Fourth International Conference on the Semantic Web*, pp. 624–637, 2005.

[62] T. Takahashi and Y. Kadobayashi, Reference ontology for cybersecurity operational information, *The Computer Journal*, vol. 58(10), pp. 2297–2312, 2015.

[63] A. Tofani, E. Castorini, P. Palazzari, A. Usov, C. Beyel, E. Rome and P. Servillo, An ontological approach to simulate critical infrastructures, *Journal of Computational Science*, vol. 1(4), pp. 221–228, 2010.

[64] A. Tolk, The next generation of modeling and simulation: Integrating big data and deep learning, *Proceedings of the Summer Computer Simulation Conference*, 2015.

[65] A. Tolk, S. Diallo, J. Padilla and H. Herencia-Zapana, Reference modeling in support of M&S – Foundations and applications, *Journal of Simulation*, vol. 7(2), pp. 69–82, 2013.

[66] P. Trucco and B. Petrenj, An ontology-based approach to vulnerability and interdependency modeling for critical infrastructure systems, in *Safety and Reliability: Methodology and Applications*, T. Nowakowski, M. Mlynczak, A. Jodejko-Pietruczuk and S. Werbinska-Wojciechowska (Eds.), CRC Press, Boca Raton, Florida, pp. 49–56, 2015.

[67] P. Trucco, B. Petrenj, S. Bouchon and C. Di Mauro, Ontology-based approach to disruption scenario generation for critical infrastructure systems, *International Journal of Critical Infrastructures*, vol. 12(3), pp. 248–272, 2016.

[68] M. Uschold and M. Gruninger, Ontologies and semantics for seamless connectivity, *ACM SIGMOD Record*, vol. 33(4), pp. 58–64, 2004.

[69] U.S. Department of Homeland Security, Infrastructure Data Taxonomy, Washington, DC (`www.dhs.gov/infrastructure-data-taxonomy`), 2017.

[70] A. Usov, C. Beyel, E. Rome, U. Beyer, E. Castorini, P. Palazzari and A. Tofani, The DIESIS approach to semantically interoperable federated critical infrastructure simulation, *Proceedings of the Second International Conference on Advances in System Simulation*, pp. 121–128, 2010.

[71] K. van Dam and Z. Lukszo, Modeling energy and transport infrastructures as a multi-agent system using a generic ontology, *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pp. 890–895, 2006.

[72] J. Vazquez Naya, M. Martinez Romero, J. Pereira Loureiro, C. Munteanu and A. Pazos Sierra, Improving ontology alignment through genetic algorithms, in *Soft Computing Methods for Practical Environment Solutions: Techniques and Studies*, M. Gestal Pose and D. Rivero Cebrian (Eds.), IGI Global, Hershey, Pennsylvania, pp. 240–259, 2010.

[73] C. Ventura, H. Juarez Garcia and J. Marti, Understanding interdependencies among critical infrastructures, *Proceedings of the Ninth U.S. National and Tenth Canadian Conference on Earthquake Engineering*, paper no. 1899, 2010.

[74] W. Wang, C. Dong and P. Yang, Ontology modeling of emergency plan systems, *Proceedings of the Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, pp. 290–294, 2009.

[75] W. Wang, A. Tolk and W. Wang, The levels of conceptual interoperability model: Applying systems engineering principles to M&S, *Proceedings of the Spring Simulation Multiconference*, article no. 168, 2009.

[76] S. Wolthusen, Modeling critical infrastructure requirements, *Proceedings of the Fifth Annual IEEE SMC Information Assurance Workshop*, pp. 101–108, 2004.

[77] WordNet Team, WordNet: A Lexical Database for English, Department of Computer Science, Princeton University, Princeton, New Jersey (`wordnet.princeton.edu`), 2017.

[78] World Wide Web Consortium, Geospatial Semantic Web Community Group, Massachusetts Institute of Technology, Cambridge, Massachusetts (`www.w3.org/community/geosemweb`), 2017.

[79] J. Xu, T. Nyerges and G. Nie, Modeling and representation for earthquake emergency response knowledge: Perspective for working with geo-ontology, *International Journal of Geographical Information Science*, vol. 28(1), pp. 185–205, 2014.

III

INDUSTRIAL CONTROL SYSTEM SECURITY

Chapter 8

ENFORCING END-TO-END SECURITY IN SCADA SYSTEMS VIA APPLICATION-LEVEL CRYPTOGRAPHY

Adrian-Vasile Duka, Bela Genge, Piroska Haller and Bogdan Crainicu

Abstract Recent technological advances have had a strong impact on performance optimization and the provisioning of flexible supervisory control and data acquisition (SCADA) systems. However, most SCADA communications protocols, as currently implemented, are extremely vulnerable to cyber attacks. Several international organizations have been developing security standards to alleviate these threats. Nevertheless, investigations reveal that the vast majority of high-end control hardware devices do not incorporate security features (i.e., security protocols). Therefore, the enforcement of data security in end-to-end communications flows must be addressed at the application layer. This chapter evaluates the feasibility of performing cryptographic computations at the application layer of a programmable logic controller. It shows that, despite the modest computational resources of modern programmable logic controllers, it is possible to develop efficient cryptographic applications that enforce several data security properties in the application layer. The experimental evaluations compare the performance of AES, SHA1 and HMAC-SHA1 against the performance of the new Speck and Simon lightweight block cipher algorithms executing on a Phoenix Contact ILC 350 PN controller with the control logic of a real SCADA system used in the Romanian gas transportation network.

Keywords: SCADA systems, cryptography, embedded systems

1. Introduction

The development and integration of complex software modules in industrial equipment are key factors in performance optimization and the provisioning of flexible supervisory control and data acquisition (SCADA) systems. However, most SCADA communications protocols, as currently implemented, are extremely vulnerable to cyber attacks. Since traditional computer system attacks can impact SCADA systems in a significant manner [5, 6], encryption and authentication mechanisms have become mandatory requirements for protecting SCADA communications.

In an attempt to promote encryption and authentication in SCADA system communications, several organizations, including the National Institute of Standards and Technology (NIST) [17], International Electrotechnical Commission (IEC) [9] and American Gas Association (AGA) [7], have been developing security standards. A recent effort by the Object Linking and Embedding (OLE) for Process Control (OPC) Foundation has resulted in the specification of the OPC Unified Architecture (OPC UA) [13]. OPC UA provides a built-in security model, including the establishment of secure communications channels and application-layer client-server sessions.

Despite these efforts, researchers have observed that implementing computationally-intensive cryptographic algorithms in control hardware is not feasible (see, e.g., [8]). This is mainly due to the critical time constraints imposed on control logic and the limited computational resources of control hardware. Studies also suggest that symmetric cryptography (e.g., AES) combined with keyed hash-based message authentication codes (MAC) (i.e., HMAC) can enhance the security of SCADA systems in situations where the provisioning of separate, external cryptographic modules are not feasible. Additionally, despite technological progress and the roll-out of high-end devices with OPC UA support, the vast majority of high-end control hardware devices do not incorporate security features (i.e., security protocols). As a result, over the next 10 to 15 years, numerous industrial control devices with limited security features will continue to be provisioned and deployed in the field. Therefore, data security for end-to-end communications flows must be addressed at the application layer. This would enable the enforcement of security features directly on the exchanged data structures.

This chapter evaluates the feasibility of performing cryptographic computations at the application layer of programmable logic controllers (PLCs). It examines the software architecture of a traditional control application, including the structuring of variables and the planning of execution tasks. It demonstrates that, despite the modest computational resources provided by a programmable logic controller, it is possible to deploy efficient cryptographic applications that enforce various data security properties at the application layer. The chapter also demonstrates that lightweight block ciphers released recently by the National Security Agency (NSA) [1, 2] support the development of practical solutions even for time-constrained applications. To this end, the performance of the AES, SHA1 and HMAC-SHA1 algorithms are compared against the Simon and Speck families of lightweight block ciphers, and insights are provided on the possible integration of cryptographic operations in software applications of programmable logic controllers. Experimental evaluations of the performance of the cryptographic algorithms are performed using an ILC 350 PN controller from Phoenix Contact and a real control application that is deployed in the Romanian gas transportation system.

2. Related Work

Knapp and Langill [10] describe the potential risks and consequences of cyber attacks against industrial control systems; they discuss the protocols and applications underlying industrial control systems and provide general rules for their protection. Stouffer et al. [17] provide guidelines for securing industrial control systems, SCADA systems and distributed control systems, as well as other control devices such as programmable logic controllers. Saxena and Choi [14] review authentication protocols for smart grids; they specifically analyze the mutual authentication, privacy, trust, integrity and confidentiality of communications in smart grid networks.

Nai Fovino et al. [12] have presented an extension of the Modbus protocol for legacy SCADA systems that supports authentication, non-repudiation and replay protection. Although the extended protocol protects against several attacks, performance and packet size overhead may impact real-time operations. Shahzad et al. [15] have presented a methodology for deploying and testing secure communications using the Modbus/TCP protocol. A cryptographic construction deployed in Modbus/TCP provides an inclusive security solution; the algorithms integrated in industrial control network communications include AES, RSA and SHA-2. According to the authors, the approach provides efficient and inclusive security, but no details are provided about the hardware (programmable logic controllers) on which the secure Modbus protocol was implemented. In any case, according to Hohlbaum et al. [8] and confirmed by the experiments presented in this chapter, it is unlikely that the vast majority of modern control hardware can support RSA and SHA-2 computations. Therefore, it is necessary to analyze the practical integration of security solutions very carefully before deploying them in production environments.

Finally, Mohan et al. [11] have proposed the SNAPE cyber security architecture for microgrids. The architecture isolates the control network from the secure SCADA network and other external networks, provisions bump-in-the-wire devices for integrating legacy equipment that cannot perform cryptographic tasks and uses OPC UA as the communications backbone. The SNAPE architecture also leverages transport layer security (TLS) and end-device authentication. However, Mohan et al. emphasize that trade-offs have to be performed to balance security versus performance, cost and usability.

3. Problem Statement

Several organizations such as the National Institute of Standards and Technology [17], International Electrotechnical Commission [9] and American Gas Association [7] have developed security standards for industrial control systems. These efforts have identified mandatory cipher suites, including modern security protocols such as transport layer security with digital signatures, Diffie-Hellman key exchange and AES with 256-bit keys and SHA. In addition, it is recommended to implement X.509 encoded certificates with certificate management systems and associated protocols.

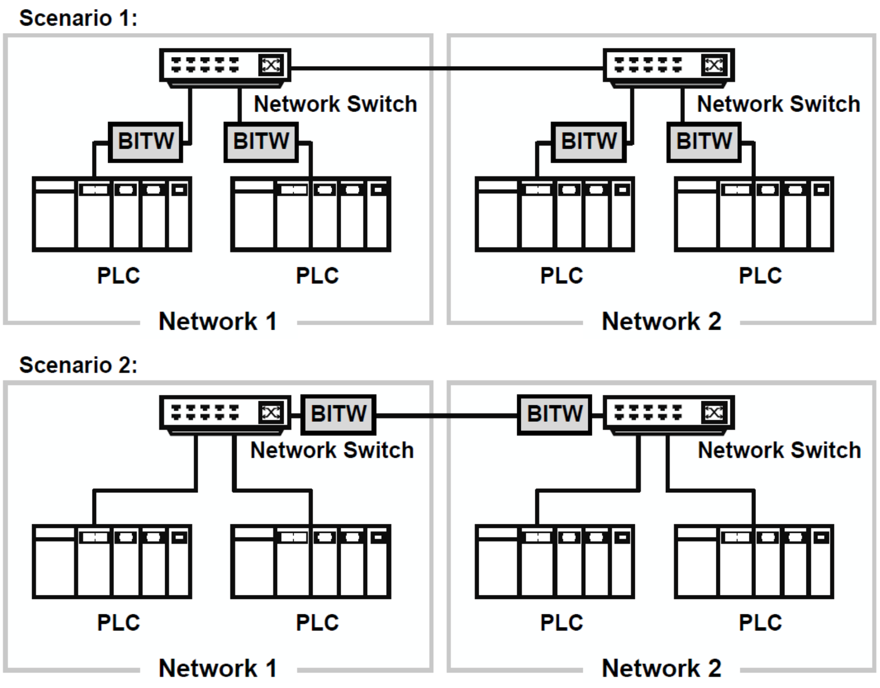


Figure 1. Bump-in-the-wire security scenarios.

While traditional cryptographic systems can indeed enhance the security of SCADA systems, careful analysis is needed to address applications of the recommended cipher suites in deployed control devices. To this end, two main approaches are available: (i) bump-in-the-wire (BITW) devices that are placed separately and in front of control hardware [11, 16]; and (ii) security mechanisms with hardware-accelerated cryptographic support [7–9] that are incorporated in communicating end-devices (e.g., programmable logic controllers).

Bump-in-the-wire solutions are typically used in end-devices with insufficient computing power [11]. This involves the positioning of a separate bump-in-the-wire device in front of each protected end-device. The bump-in-the-wire device establishes a secure communications channel between end-devices that provides a wide range of properties, including confidentiality, integrity, message authenticity and non-repudiation. Figure 1 illustrates two bump-in-the-wire security scenarios. In Scenario 1, a bump-in-the-wire device is positioned in front of each control device. In Scenario 2, a bump-in-the-wire device is positioned at each network entry-point.

From the security perspective, an obvious advantage of Scenario 1 is that, if the network is compromised and the attacker can capture/inject packets, then the internal bump-in-the-wire device would preserve the security of communications. The scenario also leverages the advantages of multiple security devices that collectively provide defense-in-depth. Another layer of security is provided by deploying hardware and software from multiple manufacturers [17]. However, despite the advantages, Scenario 1 raises significant implementation issues. First, secure multi-peer communications require a separate channel to be established between every pair of end-devices. Unfortunately, this requires more complex routing protocols (e.g., multiprotocol label switching (MPLS)) that have to be aware of the presence of more than two end-devices in order to facilitate the routing and encryption of packets between multiple peers. Second, a significant downside is the cost of purchasing and deploying a large number of bump-in-the-wire devices. This is a major aspect of the analysis because a two-end-point virtual private network (VPN) enabler can cost as much as a programmable logic controller. Therefore, while Scenario 1 yields a highly secure network, the cost of the security devices could well exceed the cost of the SCADA infrastructure.

In contrast, in Scenario 2, a bump-in-the-wire device is positioned at the edge of each network. A secure channel is thus established between two bump-in-the-wire devices that control the access points to the two networks. While this scenario significantly reduces infrastructure costs, it does not provide end-to-end security. For example, an attacker who can inject valid packets into one of the networks can easily reach the other network by forging packets, despite the bump-in-the-wire devices positioned at the network edges.

Conversely, integrating security mechanisms in end-devices is clearly superior to using bump-in-the-wire devices because it can enforce end-to-end security requirements on each device. However, this only holds when the end-devices have adequate computing power to establish a secure communications channel. If this is the case, then application-layer authentication and data security can be achieved using a number of protocols, including OPC UA. Still, despite technological advancements and the roll-out of high-end devices with OPC UA support, investigations performed by the authors of this chapter in cooperation with Romanian automation companies reveal that deployed end-devices such as programmable logic controllers do not support security protocols. Furthermore, programmable logic controller vendors, at least those based in Romania, are only just beginning to integrate OPC UA support in their products – this means that, in the short-term, only the next generation of programmable logic controllers will implement security protocols. Unfortunately, this also means that the available high-end controllers will not support security protocols, including OPC UA. As a result, the adoption of OPC UA, at least in Romania, will only occur in the long term.

Based on these observations, it can be concluded that, although technological advancements have enhanced controller performance, high-end devices that are being integrated in SCADA systems do not support modern security protocols such as the highly-recommended OPC UA. As a result, research suggests that bump-in-the-wire solutions should be used to implement security measures. Nevertheless, the authors of this chapter believe that an increase in the computing power of programmable logic controllers would enable the integration of cryptographic algorithms and security protocols in programmable logic controller applications. However, despite improvements in the performance of modern programmable logic controllers, the devices still have limited capabilities. Therefore, it is necessary to carefully examine the computations that can be implemented in time-critical applications. The remainder of this chapter describes the research involved and the insights gained in integrating cryptographic computations in programmable logic controller applications. The analysis is based on experiments performed on a real programmable logic controller with control logic that is used in the Romanian gas transportation network.

4. Cryptographic Applications

This section discusses cryptographic applications for programmable logic controllers.

4.1 PLC Architecture

The main unit of execution in a controller (e.g., programmable logic controller) is a task. Tasks are scheduled to run periodically and may trigger the execution of blocks of code called program units; these are simply referred to as "programs." Tasks that run programs are usually called user tasks. Another key task is communications (e.g., a task that runs the communications with an OPC server/client). Each user task can be assigned a priority level and can run several programs.

Data exchanges between end-points involve variables that can be selected for reading and/or writing. The types of variables vary according to the supported data types. However, typical values include scalar variables (one to four bytes of data) along with arrays, strings, structures and user-defined data types. User-defined data structures are commonly employed to group data. Variables of this type can be selected for reading and/or writing via protocols such as OPC. A user-defined data structure is a common element of a programmable logic controller program and an important feature that is exploited in the proposed security architecture. When a different protocol is used for data transfer (e.g., Modbus/TCP), the variable view is transformed to a memory register map. In this case, a dedicated program copies variables from/to Modbus packets.

4.2 Secure Application Architecture

The proposed application architecture is designed to enforce certain security properties for data exchanges between end-points. The architecture is founded on three main assumptions: (i) symmetric key cryptography is used due to the limited computing resources; (ii) each end-point (e.g., programmable logic controller, OPC client/server or Modbus client/server) is bootstrapped via a unique secret key K_i that is used to establish session keys; and (iii) end-points i and j share the session key K_{ij}.

When the three assumptions hold, it is possible to enforce the security properties of confidentiality, data integrity and authentication on data exchanges. The confidentiality property is achieved by encrypting data using the session key K_{ij}. The data integrity property is achieved by applying a hashing algorithm. The data authentication property is achieved via authenticated hashing, where a hash function is applied to the data and secret key; this operation yields the message authentication code (MAC).

The aforementioned security properties can be enforced on data exchanges (i.e., program variables) in a number of ways. As in any security architecture, it is imperative to apply cryptographic operations before data is transmitted and after it is received. As mentioned earlier, the data exchanges are performed by the communications tasks (e.g., via the OPC protocol). Unfortunately, programmable logic controller applications usually do not have access to the transmission/reception routines of communications modules such as OPC. These vendor-specific modules are usually inaccessible to applications even when a programming license has been purchased. Therefore, cryptographic operations on the exchanged variables must be performed by the programs running in user tasks. Obviously, if the moment of transfer is deterministic and controllable by user code (as in the case of the Modbus protocol), then the cryptographic operations can be performed by the programs dedicated to these communications.

Because the unit of execution is a program, the proposed application architecture is designed to ensure that, although programs are unaware of the exact times when variable exchanges are performed, they can still enforce the security properties. Accordingly, the architecture involves two aspects. First, programs are extended with cryptographic operations on the variables that are exchanged between end-points. Second, the user-defined data structures that are exchanged are modified to include the computed security data. Thus, programs perform decryption and integrity/authenticity verification operations at the very beginning and terminate their execution with encryption and integrity/authenticity enforcement operations. The results of these computations are stored in data structures and are included in the data transfer.

Obviously, a principal concern regarding these changes to the architecture of a program is the impact on execution time. The following notation is introduced to evaluate the possible limitations of the approach. Let t_b^P and t_e^P be the computational times of the cryptographic operations performed at the beginning and at the end of the execution of a program P, respectively. Let t_u^P be the execution time of the program code.

As mentioned above, tasks are scheduled to run periodically. When a task is scheduled to run, it executes all its associated programs. Let T_α be the task scheduling time. Then, for the proposed scheme to work, it is necessary to ensure that the total execution time of all the programs P associated with task α does not exceed T_α:

$$\sum_P (t_b^P + t_u^P + t_e^P) < T_\alpha \tag{1}$$

Equation (1) ensures that the total execution time of all the programs, including the cryptographic operations, do not exceed the task scheduling period. Note that a programmable logic controller also incorporates a fail-safe mechanism implemented using a watchdog timer. Specifically, an error is signaled when the program execution exceeds a preset watchdog time T_α^W. A watchdog timer is usually configured on a per-task basis. Let T_α^W denote the watchdog timer configured for task α. Then, in addition to enforcing Equation (1), the following equation must hold:

$$\sum_P (t_b^P + t_u^P + t_e^P) < T_\alpha^W \tag{2}$$

Equations (1) and (2) demonstrate that it is necessary to reduce the values of t_b^P and t_e^P as much as possible in order to ensure the feasible application of the proposed methodology. A significant issue is that the value of T_α in a time-critical application can be very small. In fact, T_α can be a few milliseconds, which means that all the cryptographic operations must be completed in less than 1 ms.

The following recommendations help ensure practical applications of the proposed architecture:

- **Recommendation 1:** Applications should use efficient cryptographic algorithms to reduce their computational times as much as possible. Standard cryptographic algorithms such as AES and SHA-1 are recommended. Novel lightweight cryptographic algorithms designed for devices with limited computational resources have been released recently. As shown later in this chapter, block ciphers from the Speck and Simon family [1] can execute several times faster than AES while offering a similar level of security. Furthermore, these ciphers can be used to construct lightweight MAC functions that significantly outperform the HMAC standard.

- **Recommendation 2:** Applications should select variables that are secured (i.e., encrypted or hashed) to reduce their computational times. This step requires knowledge about the specific application in order to identify sensitive variables and the required security properties. A careful examination, for example, may show that it is necessary to ensure only data authenticity for a reduced set of variables that are deemed to be the most critical (e.g., variables used in control loop computations).

5. Use Case Assessment and Results

The use case assessment employed a Phoenix Contact ILC 350 PN controller running the control logic from a real SCADA system. The ILC 350 PN is a high-end controller that runs ProConOS (Programmable Controller Operating System) and is based on Windows CE technology and the .NET 4.2 framework. The control logic was provided by a local automation company that deploys SCADA systems for gas transportation system operators.

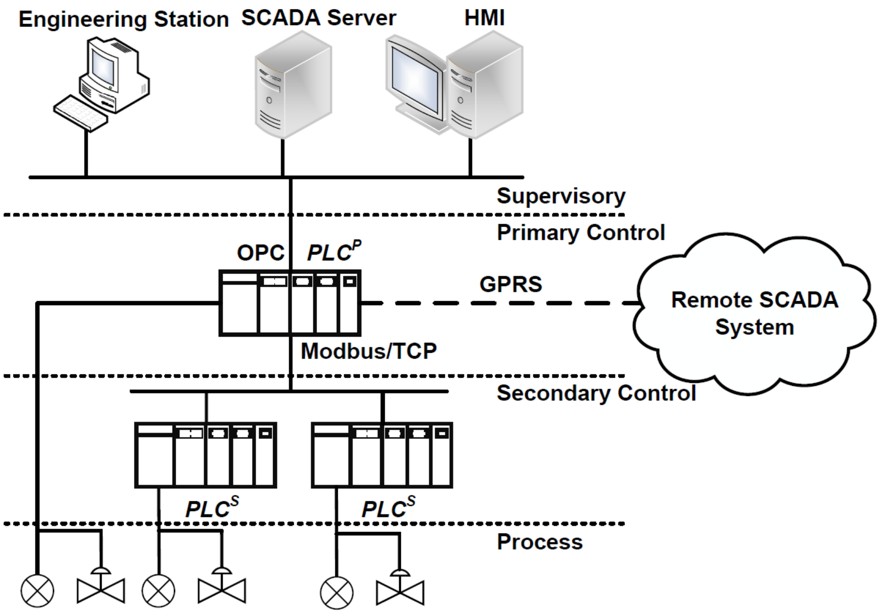


Figure 2. Simplified representation of the analyzed system.

5.1 System Analysis

The programmable logic controller application examined in this work executed the primary control logic used to automate a gas transportation node in the Romanian gas network. Figure 2 shows a simplified view of the network architecture. The control system is structured into two layers: (i) primary control layer; and (ii) secondary control layer. The primary controller PLC^P communicates with secondary controllers PLC^S via Modbus/TCP and is connected to the rest of the SCADA system (that manages information from other gas transportation nodes) via GPRS. PLC^P performs two main functions: (i) implementation of the process control loops based on measurements from PLC^S and information from transducers directly connected to it; and (ii) information exchange with the supervisory layer via OPC.

The remainder of the analysis focuses on the application architecture implemented as part of PLC^P. PLC^P executes five main tasks:

- **Task 1 ($\alpha = 1$):** This task runs two main control loop programs, P_1 and P_2, with a cycle time of $T_1 = 100\,\text{ms}$ and watchdog timer $T_1^W = 200\,\text{ms}$.

- **Task 2 ($\alpha = 2$):** This task reads the digital inputs (alarm and sensor limits) and sets the digital outputs (to open or close control relays according to the sensor alarms) every 50 ms ($T_2 = 50\,\text{ms}$) with watchdog timer $T_2^W = 100\,\text{ms}$.

- **Task 3 ($\alpha = 3$):** This task reads the analog sensor values every 50 ms (T_3 = 50 ms) and sets the alarm variables if the limits are exceeded (T_3^W = 100 ms).

- **Task 4 ($\alpha = 4$):** This task ensures communications with the secondary programmable logic controllers via Modbus/TCP. It is invoked every 10 ms (T_4 = 10 ms) with watchdog timer T_4^W = 100 ms.

- **Task 5 ($\alpha = 5$):** This task implements communications with the remote SCADA system. It runs a program unit that periodically verifies the status of the communications line every 10 ms (T_5 = 10 ms) with watchdog timer T_5^W = 100 ms.

The following data structures are transferred between the components:

- **DATAS1:** This data structure, which is 288 bytes long, is read and written by the remote SCADA system (Task 5), by Task 1 and via OPC.

- **DATAS2:** This data structure, which is 198 bytes long, is received from the secondary controllers (Task 3) and is forwarded via OPC.

- **DATAS3:** This data structure, which is 8 bytes long, is written by Task 2 and is read via OPC.

- **DATAS4:** This data structure, which is 546 bytes long, is written by Task 3 and is forwarded via OPC.

- **DATAS5:** This data structure, which is 202 bytes long, is received from the remote SCADA system and is forwarded via OPC.

Certain observations can be made based on this analysis. `DATAS1` includes measurement variables and setpoints that are changed by three sources. Therefore, the application of the proposed scheme on `DATAS1` requires careful reorganization to separate the values modified by the remote SCADA system from the values modified by PLC^P so that each end-point enforces the security properties on the data it generates. On the other hand, the security measures for `DATAS2` must be applied by the secondary controllers; this is because these controllers take the measurements. Similarly, the security properties for `DATAS5` must be enforced by the sender, which is the remote SCADA system. The security properties for `DATAS3` and `DATAS4` must be enforced by PLC^P.

5.2 Cryptographic Algorithms

This section evaluates the performance of the AES block cipher, SHA1 hash function, HMAC-SHA1 message authentication code and Speck and Simon family of block ciphers. The `IT_Security_1_00` library from Phoenix Contact, which includes AES implementations with 128-, 192- and 256-bit keys, SHA1 and HMAC-SHA1 implementations, was employed. The Speck and Simon family of block ciphers was also implemented on the ILC 350 PN controller. Speck

Table 1. Execution times of the Phoenix Contact cryptographic algorithms.

Algorithm	**Number of Calls**	**Total Time (ms)**	**Time per Call (ms)**
AES-128	500	391	0.782
AES-192	500	463	0.926
AES-256	250	266	1.064
SHA1/512 bytes	100	168	1.68
SHA1/256 bytes	100	108	1.08
SHA1/128 bytes	100	78	0.78
SHA1/64 bytes	100	63	0.63
SHA1/32 bytes	100	47	0.47
HMAC-SHA1/512 bytes	44	115.28	2.62
HMAC-SHA1/256 bytes	34	67.32	1.98
HMAC-SHA1/128 bytes	28	45.92	1.64
HMAC-SHA1/64 bytes	25	36.75	1.47
HMAC-SHA1/32 bytes	22	28.60	1.30

and Simon offer great flexibility and a range of configurations in terms of block and key sizes. Versions are available with block sizes ranging from 32 bits to 128 bits and key sizes ranging from 64 bits to 256 bits. The complete set of configurations for the two ciphers was implemented and tested.

Performance differences seen in the implemented variants of Speck and Simon are due to the data types supported by the controller. The programmable logic controller natively supports data types of 8, 16 and 32 bits as part of the IEC 61131-3 elementary data types, as well as user-defined data types that can be derived from them in the form of data structures and arrays. The programmable logic controller can perform the operations required by Simon and Speck for data sizes of n = 8, 16 and 32 bits. For the remaining cases (i.e., n = 24, 48 and 64 bits), the arithmetic for left/right rotation and modulo 2^n addition was implemented.

5.3 Computational Time

The execution times of the cryptographic algorithms implemented in the ILC 350 PN controller were evaluated by recording the number of elapsed system ticks, where 1 tick = 1 ms. The initial evaluations revealed that single calls to most of the assessed algorithms were executed in less than 1 ms. Since the controller resolution was 1 ms, all the measurements were made using multiple calls to each algorithm. The performance evaluations were performed by a dedicated task and program; the watchdog timer was set to 500 ms.

Table 1 shows the execution times of the cryptographic algorithms implemented as part of the `IT_Security_1_00` library. The results reveal that 500 calls to the AES algorithm with a 128-bit key length were executed in 391 ms.

Table 2. Execution times of the Speck and Simon cryptographic algorithms.

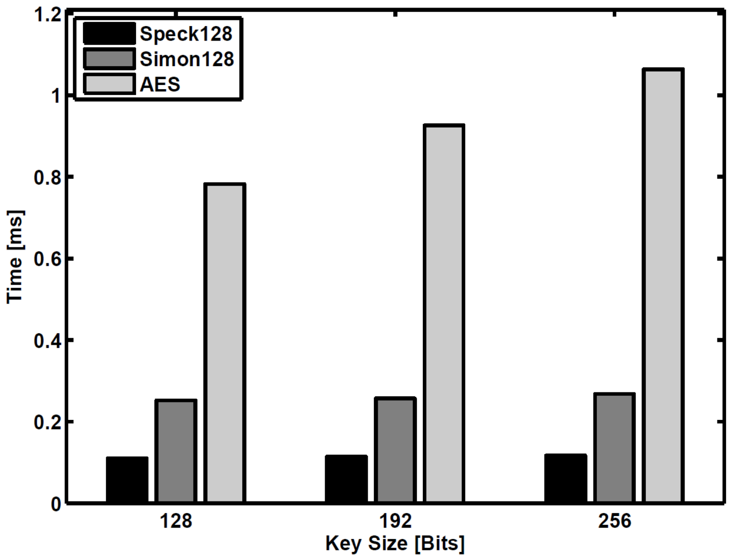
Algorithm	Total Time (ms)	Time per Call (ms)	Algorithm	Total Time (ms)	Time per Call (ms)
Speck32/64	17	0.017	Simon32/64	34	0.034
Speck48/72	27	0.027	Simon48/72	68	0.068
Speck48/96	29	0.029	Simon48/96	68	0.068
Speck64/96	19	0.019	Simon64/96	42	0.042
Speck64/128	20	0.020	Simon64/128	44	0.044
Speck96/96	92	0.092	Simon96/96	197	0.197
Speck96/144	95	0.095	Simon96/144	205	0.205
Speck128/128	110	0.110	Simon128/128	252	0.252
Speck128/196	114	0.114	Simon128/196	257	0.257
Speck128/256	117	0.117	Simon128/256	268	0.268

Note that, in this case, it can be inferred that a single call to the AES algorithm with a 128-bit key completes in 0.782 ms. Given a block size of 128 bits, a single call to AES-128 would encrypt 16 bytes of data. However, when the key size was increased to 256 bits, the program execution time exceeded the value of the watchdog timer. Hence, the number of calls was reduced to 250, for which the algorithm completed its computations in 266 ms.

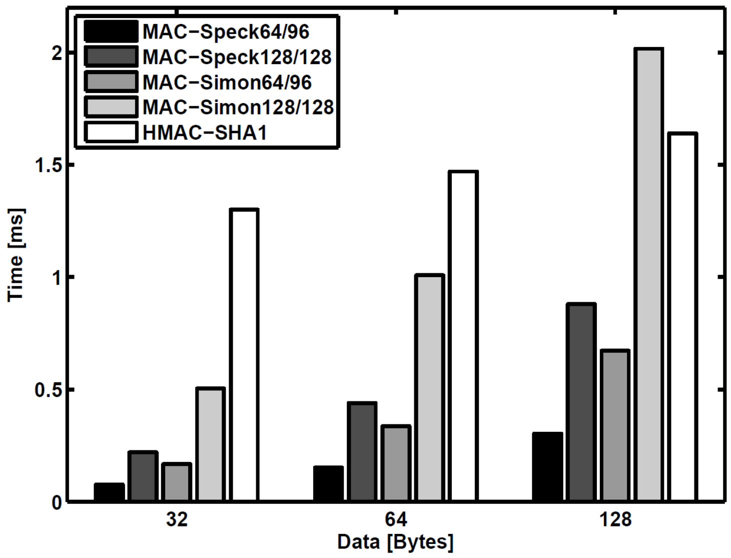
The results reveal that the SHA1 algorithm hashed 512 bytes of data in just 1.68 ms and 32 bytes of data in 0.47 ms. However, the computational times of HMAC-SHA1 increased to 2.62 ms for 512 bytes of data and to 1.30 ms for 32 bytes of data. This is because the HMAC standard includes multiple calls to the SHA1 function as well as several concatenation and arithmetic operations that affect its overall performance.

The Speck and Simon block ciphers were implemented and the correctness of all their variants was verified as documented in [1, 2]. Table 2 shows the execution times of the Speck and Simon algorithms where the total time corresponds to 1,000 calls of each algorithm. The results reveal that the Speck and Simon algorithms provide significant computational advantages over AES. For example, while AES-128 encrypted 16 bytes in 0.782 ms, Speck with 128-bit blocks and a 128-bit key (denoted by Speck128/128) encrypted the same number of bytes in just 0.110 ms. The Simon algorithm with the same configuration encrypted 16 bytes of data in 0.252 ms. In fact, Speck executed seven times faster than AES and Simon executed three times faster than AES (see Figure 3(a)). Thus, significant reductions in computational times can be obtained by adopting the lightweight cryptographic algorithms with 128-bit keys.

While the Speck and Simon algorithms may be used to enforce the confidentiality property, they can also be employed to implement lightweight MAC functions. Black et al. [3, 4] have shown that certain constructions involving block ciphers and simple XOR operations may yield collision-resistant MAC



(a) Computational times of the Speck, Simon and AES algorithms.



(b) MAC computational times based on the Speck, Simon and AES algorithms.

Figure 3. Comparison of the algorithms implemented on the ILC 350 PN controller.

Table 3. Comparison of the MAC-Speck, MAC-Simon and HMAC-SHA1 functions.

Algorithm	**Execution Time (ms)**			**Improvement over HMAC-SHA1**		
	32 B	**64 B**	**128 B**	**32 B**	**64 B**	**128 B**
MAC-Speck64/96	0.076	0.152	0.304	17.1	9.67	5.39
MAC-Speck128/128	0.220	0.440	0.880	5.9	3.34	1.86
MAC-Simon64/96	0.168	0.336	0.672	7.73	4.37	2.44
MAC-Simon128/128	0.504	1.008	2.016	2.57	1.45	**0.81**
HMAC-SHA1	1.3	1.47	1.64	–	–	–

functions. Therefore, a Merkle-Damgard construction was employed to produce lightweight MAC functions with different variants of Speck and Simon.

Figure 3(b) and Table 3 show the computational times for the MAC-Speck, MAC-Simon and HMAC-SHA1 functions on the ILC 350 PN controller. Note that when using a smaller block size (64 bits) and key size (96 bits), the MAC function based on Speck is more than 17 times faster than HMAC-SHA1 for 32 bytes of data and it is 5.39 times faster for 128 bytes of data. Similarly, the Simon64/96-based MAC function implementation is 7.73 times faster than HMAC-SHA1 for 32 bytes of data and 2.44 times faster for 128 bytes of data.

Smaller block sizes yield smaller hash sizes that decrease the collision resistance [4] of the MAC function. Therefore, a larger block size should be chosen to increase the security of the MAC function. However, in this case as well, Speck128/128 is 5.9 times faster than HMAC-SHA1 for 32 bytes of data, 3.34 times faster for 64 bytes of data and 1.86 times faster for 128 bytes of data. Simon128/128 also outperforms HMAC-SHA1 for 32 and 64 bytes of data. These results indicate that it is better to use variants of the MAC-Speck and HMAC-SHA1 functions for larger data blocks.

5.4 Security Properties in Control Applications

The analysis reveals that good performance can be achieved using lightweight cryptography for confidentiality and data authenticity. For the industrial applications under discussion, it is possible to estimate the computational overhead involved in applying the MAC functions to the data structures. In particular, the authenticity properties may be applied to the data generated by PLC^P.

The analysis above identified five data structures of different sizes that are transferred within the system and that PLC^P needs to enforce the security properties on a subset of these data structures – `DATAS3`, `DATAS4` and partly on `DATAS1`.

Since `DATAS3` is eight bytes long, one call to Speck128/128 or Simon128/128 would be sufficient to compute the MAC function. As such, the computational overhead is 0.110 ms for Speck128/128 and 0.252 ms for Simon128/128. On the other hand, when HMAC-SHA1 is used, the computational overhead increases up to 1.23 ms.

The two larger data structures, `DATAS4` and `DATAS1`, require a different approach. `DATAS4` includes the analog/digital measurements of PLC^P performed by Task 3. In this case, using Speck128/128 would add a computational overhead of 3.73 ms and Simon128/128 would add 8.50 ms. On the other hand, using HMAC-SHA1 would reduce the computational overhead to 2.68 ms.

A similar analysis revealed that HMAC-SHA1 may be more appropriate for `DATAS1`. Nevertheless, despite its better performance, the computational overhead of HMAC-SHA1 may be too high for real-time tasks scheduled at a rate of 10 ms. The computational overhead can be reduced using a different configuration for the Speck cipher. For example, when using Speck64/96, the computational overhead for `DATAS1` is reduced to 0.684 ms, the overhead for `DATAS3` is reduced to 0.019 ms and the overhead for `DATAS4` is reduced to 1.28 ms.

Discussions with industry personnel revealed that, in general, the computational overhead of cryptographic operations should not exceed 10% of the task periodicity. Accordingly, the computational overhead for `DATAS3` (under 1 ms) can be handled by Task 2 because $T_2 = 50$ ms. On the other hand, the computational overhead for `DATAS4` in Task 3 ($T_3 = 50$ ms) would be handled best by HMAC-SHA1 or Speck64/96. Conversely, because `DATAS1` is changed by different tasks, enforcing the authenticity property for a single task (as proposed in this work) is not feasible. Therefore, in this case, the control application requires further restructuring to identify the most appropriate way to rearrange `DATAS1` without impacting the performance of applications. This topic will be investigated in future research.

Finally, according to the recommendations presented in the previous section, the computational overhead can be further reduced by carefully examining the variables for which security properties have to be enforced. This work has assumed the worst case scenario in which the security properties need to be enforced in all data exchanges. However, discussions with industry personnel revealed that the variables can be ranked based on their significance because only a subset of variables is actually used by the control loops. Therefore, the proposed protection scheme can be applied to enforce security properties in control applications. However, it should be done on a per-application basis in order to minimize the computational overhead and satisfy the control requirements.

6. Conclusions

Recent technological advances in cyber security and SCADA systems can facilitate the integration of lightweight cryptographic mechanisms in industrial control applications. This enables the enforcement of security properties such as confidentiality, integrity and data authenticity on program variables exchanged between end-points in SCADA networks. However, the efforts undertaken in this research must be extended to integrate key exchange and key distribution protocols. Note that this research does not advocate the replacement of existing network/transport layer security protocols such as SSL/TLS and IPSec. Instead, the work complements these protocols and other related approaches by ensuring that SCADA system security properties can be enforced and verified by end-points.

Acknowledgement

This research was supported by a grant from the Romanian National Authority for Scientific Research and Innovation: CNCS/CCCDI-UEFISCDI, Project No. PN-III-P2-2.1-BG-2016-0013 within PNCDI III.

References

[1] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks and L. Wingers, The Simon and Speck Families of Lightweight Block Ciphers, National Security Agency, Fort Meade, Maryland (`eprint.iacr.org/2013/404.pdf`), 2013.

[2] R. Beaulieu, S. Treatman-Clark, D. Shors, B. Weeks, J. Smith and L. Wingers, The Simon and Speck lightweight block ciphers, *Proceedings of the Fifty-Second ACM/EDAC/IEEE Design Automation Conference*, 2015.

[3] J. Black, P. Rogaway and T. Shrimpton, Black-box analysis of the block-cipher-based hash-function constructions from PGV, *Proceedings of the Twenty-Second Annual International Cryptography Conference*, pp. 320–335, 2002.

[4] J. Black, P. Rogaway, T. Shrimpton and M. Stam, An analysis of the block-cipher-based hash functions from PGV, *Journal of Cryptology*, vol. 23(4), pp. 519–545, 2010.

[5] T. Chen and S. Abu-Nimeh, Lessons from Stuxnet, *IEEE Computer*, vol. 44(4), pp. 91–93, 2011.

[6] A. Cherepanov, BlackEnergy by the SSHBearDoor: Attacks against Ukrainian news media and electric industry, *WeLiveSecurity*, January 3, 2016.

[7] M. Hadley, K. Huston and T. Edgar, AGA-12, Part 2 Performance Test Results, PNNL-17117, Pacific Northwest National Laboratory, Richland, Washington, 2007.

[8] F. Hohlbaum, M. Braendle and F. Alvarez, Cyber security: Practical considerations for implementing IEC 62351, presented at the *Protection, Automation and Control World Conference*, 2010.

[9] International Electrotechnical Commission, IEC/TS Technical Specifications 62351-1 to 62351-7, Power Systems Management and Associated Information Exchange – Data and Communications Security, Geneva, Switzerland, 2012.

[10] E. Knapp and J. Langill, *Industrial Network Security: Securing Critical Infrastructure Networks for Smart Grid, SCADA and Other Industrial Control Systems*, Syngress, Waltham, Massachusetts, 2015.

[11] A. Mohan, G. Brainard, H. Khurana and S. Fischer, A cyber security architecture for microgrid deployments, in *Critical Infrastructure Protection IX*, M. Rice and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 245–259, 2015.

[12] I. Nai Fovino, A. Carcano, M. Masera and A. Trombetta, Design and implementation of a secure Modbus protocol, in *Critical Infrastructure Protection III*, C. Palmer and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 83–96, 2009.

[13] OPC Foundation, Unified Architecture – The Universal Communication Platform for Standardized Information Models, V1.1 GB, Scottsdale, Arizona (`opcfoundation.org/wp-content/uploads/2014/05/OPC-UA_CollaborationOverview_EN.pdf`), 2015.

[14] N. Saxena and B. Choi, State of the art authentication, access control and secure integration in smart grid, *Energies*, vol. 8(10), pp. 11883–11915, 2015.

[15] A. Shahzad, M. Lee, Y. Lee, S. Kim, N. Xiong, J. Choi and Y. Cho, Real-time Modbus transmissions and cryptography security designs and enhancements of protocol sensitive information, *Symmetry*, vol. 7(3), pp. 1176–1210, 2015.

[16] Siemens, SICAM/SIPROTEC: System Hardening for Substation Automation and Protection, Guideline (Best-Practice Guide), V1.11, Release 12.2012, Nuremberg, Germany, 2012.

[17] K. Stouffer, J. Falco and K. Scarfone, Guide to Industrial Control Systems (ICS) Security, NIST Special Publication 800-82, Revision 1, National Institute of Standards and Technology, Gaithersburg, Maryland, 2011.

Chapter 9

SOFTWARE DEFINED RESPONSE AND NETWORK RECONFIGURATION FOR INDUSTRIAL CONTROL SYSTEMS

Hunor Sandor, Bela Genge, Piroska Haller and Flavius Graur

Abstract The technological shift from isolated industrial control systems to system-of-systems architectures has introduced myriad security challenges. Following popular trends, modern industrial control systems are incorporating technologies such as Industry 4.0, Internet of Things and cloud computing. In these architectures, traditional information and communications hardware and software are glued together with physical components and modern technologies based on IP networks such as software defined networking. The ability of these systems to respond and reconfigure themselves to mitigate faults and attacks is immensely attractive. This chapter proposes a three-tier architecture that implements response and reconfiguration capabilities in an industrial control system. It adopts a software defined network tier for dynamic communications flow (re)configuration and whitelisting, an application tier for the optimal placement of anomaly detection systems and a supervision tier for gluing the three tiers together. The effectiveness and performance of the protection mechanism are demonstrated via use case based qualitative and quantitative assessments.

Keywords: Industrial control systems, security, software defined networking

1. Introduction

Modern industrial control systems are complex and heterogeneous system of systems that offer the advantages of traditional information and communications hardware and software. The pervasive integration of off-the-shelf information and communications technology in the core of industrial control systems has broadened the palette of features and applications, enabled flexible and efficient infrastructures and decreased provisioning and maintenance costs.

Advancements in industrial control systems have also triggered a new technological revolution that has introduced novel applications and services. Industry 4.0 is a most popular initiative that showcases the progress made in the field of manufacturing [13, 18]. These systems integrate Internet of Things technologies and the Internet of Services (IoS) to facilitate remote communications between cyber and physical components, and the interactions of human operators with the underlying infrastructure.

While the shift from isolated industrial control system environments to open and complex technological ecosystems provides numerous benefits, it has had a dramatic impact on security [3]. The integration of traditional information and communications technologies in modern control system architectures has significantly increased the exposure to cyber attacks. In the field of traditional computer systems the effects of cyber attacks are usually limited to the cyber dimension, but in the case of industrial control systems the effects propagate to the physical dimension. As a result, malware infections can impact critical infrastructure assets, causing economic losses and physical damage [7, 12].

The Industry 4.0 initiative has identified three key challenges related to security and resilience: (i) stability; (ii) data privacy; and (iii) cyber security [10]. Therefore, a key requirement in modern industrial control systems is a response and reconfiguration property that can mitigate cyber attacks [8]. The core elements of a response and reconfiguration technique are the detection and isolation of security threats. Implementing these elements requires the deployment of security devices across an industrial control system to support the enforcement of multi-level protection strategies.

This chapter proposes a three-tier security solution that provides semi-automated response and reconfiguration functionality in an industrial control system. The proposed solution comprises: (i) a software defined network (SDN) tier that supports dynamic communications flow (re)configuration and whitelisting; (ii) an application tier that enforces the optimal placement of anomaly detection systems (ADSs); and (iii) a software-assisted human supervision and intervention tier that glues the three tiers together. Several traditional protection mechanisms are integrated and harmonized with the aid of software defined networking, including firewalls, anomaly detection systems and communications flow reconfiguration in response to cyber attacks. To achieve its goal, the proposed solution leverages an optimization strategy that: (i) configures software defined network switches to whitelist permitted communications flows; (ii) minimizes inter-network communications flows to reduce the number of firewall deployments; and (iii) minimizes the number of detection systems that monitor industrial communications. Human operators are an integral part of the solution because they can tune the optimization engine to adjust the reconfiguration results. Indeed, the proposed solution provides comprehensive multi-layer protection via a defense-in-depth strategy.

2. Related Work

Several researchers have investigated the response and reconfiguration problem for industrial control systems. Combita et al. [8] have provided a detailed categorization of the most significant detection and reconfiguration techniques. They discuss possible attacks on sensors and actuators and list reconfiguration strategies based on game theory. The surveyed approaches perform reconfigurations at the physical process level by adapting control loop configurations. In contrast, the novelty of the proposed methodology is that, instead of tuning control loops, it reconfigures communications paths to mitigate the adverse effects of attacks and failures. Furthermore, the proposed methodology delivers an optimal reconfiguration solution that enforces the basic requirements of industrial communications (e.g., shortest routing path to minimize communications delays) and ensures that every data flow is monitored by an anomaly detection system.

The applicability of software defined networking has been proven in diverse areas ranging from disaster preparedness [21] to industrial control (e.g., smart grid [9]) and the Internet of Things [23]. Applications in these areas leverage the technology to enhance communications resilience. Researchers have studied link and node failures in software defined networks [15] and approaches for improving recovery mechanisms [24]. Jin et al. [14] have developed the Dionysus system that performs consistent network updates in a software defined network environment. Dionysus constructs a graph of network update dependencies and schedules updates by taking into account the performance of network switches.

Several researchers have analyzed the adoption of software defined networks in the industrial sector. The benefits have been demonstrated in a test infrastructure comprising an IEC 61850 based power system [20]. Dorsch et al. [9] have highlighted the advantages and disadvantages of using software defined networks in industrial environments. They acknowledge the benefits related to network management, quality of service optimization and system resilience, but raise serious concerns about the increased risks of cyber attacks against the centralized controllers of software defined networks.

Genge et al. [11] have proposed a hierarchical network infrastructure for industrial control systems that leverages software defined networking technology. They engage a network reconfiguration engine that performs traffic optimization to shorten communications paths while maintaining key industrial control communications requirements such as quality of service. The approach has been implemented in a software defined network controller named OptimalFlow. In contrast, this research defines a network optimization problem that focuses on the shortest route objective while considering the industrial control network topology and minimizing the interconnections and communications between segments, thereby minimizing the number of firewall deployments.

Finally, it should be noted that the majority of studies assume complete software defined network deployment scenarios in which all the switches have built-in support for software defined networking. However, such scenarios are rarely, if ever, encountered in real-world industrial control environments. In fact, deploying such infrastructures would require major investments and architectural changes. Therefore, this research focuses on a more realistic hybrid infrastructure in which software defined network components function alongside traditional network equipment [26].

3. Proposed Security Solution

This section describes the proposed three-tier security solution that provides semi-automated response and reconfiguration functionality in an industrial control system to combat cyber attacks.

3.1 Overview

Given the vast number of recent cyber attacks that have targeted the industrial sector, it is imperative that modern industrial control systems be endowed with the ability to semi-automatically or automatically respond to cyber attacks. A number of technologies have contributed to the security and resilience of communications infrastructures. Significant improvements in protection have been achieved by migrating firewall filtering features into networking equipment such as switches. This enables a network switch to provide valuable defensive functionalities such as enforcing the whitelisting of network traffic flows and dynamically reconfiguring whitelists based on commands received from a central controller.

Software defined networking [22] is a promising advancement in the field of IP networking that has been recently categorized as the "Next Big Technology" [1]. It provides the means to create virtual networking devices and services and implement global networking decisions by decoupling the control plane where routing decisions are made from the forwarding plane that sends network packets to their destinations. Software defined networks often rely on OpenFlow to enable communications with remote devices [24]. OpenFlow ensures remote access to the forwarding plane of a network switch via an open protocol.

Industrial control system response and reconfiguration can be achieved by combining sensing, decision and intervention components. Figure 1 presents the proposed architecture for implementing industrial control system response and reconfiguration. The architecture incorporates three tiers: (i) a communications tier that leverages software defined networking to whitelist traffic based on firewall filtering principles; (ii) an application tier that optimally distributes anomaly detection systems to minimize costs while ensuring effective distributed monitoring of communications flows; and (iii) a human supervisory tier that glues all the layers by receiving anomaly alerts, computing optimal network configurations and deploying the optimal configurations.

Figure 2 presents the conceptual sequence of actions in the proposed protection scheme. The first step involves a human operator who collects the required parameters for initializing the system and computing an optimal network configuration that includes the network topology, flow demands, links, switches and

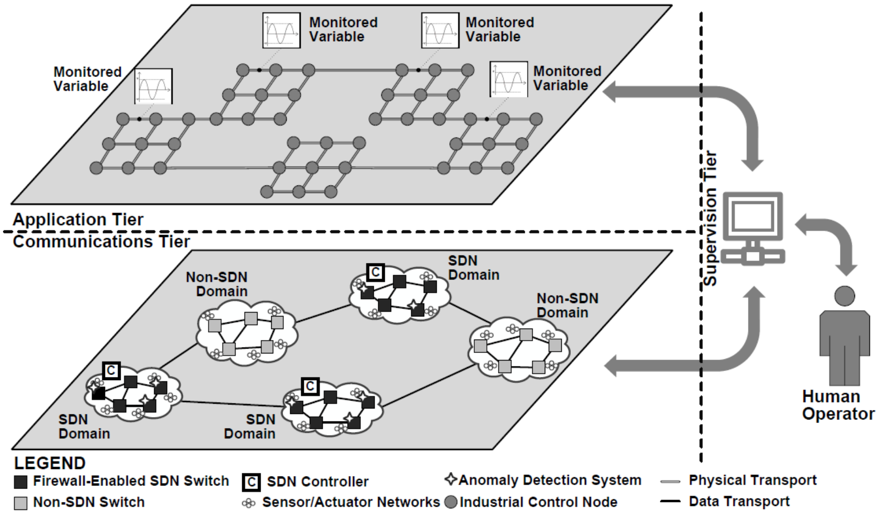


Figure 1. Architecture of the proposed three-tier protection scheme.

anomaly detection systems. In the next step, the operator, with the help of a reconfiguration engine, computes the optimal network configuration and optimal locations for the anomaly detection systems. The optimal whitelisted flow configurations (i.e., forwarding rules) are then applied to the network topology using a software defined network controller; the anomaly detection systems are positioned manually at the prescribed locations. When the network topology has to be changed – in response to a system enhancement or an anomaly alert – the human operator can intervene and change the parameters in order to recompute the optimal network architecture, which is subsequently deployed.

3.2 Communications Tier

The communications tier embodies the basic NIST principles for constructing industrial control system networks [25]. Accordingly, it defines security zones isolated by firewalls, thereby permitting only legitimate (i.e., whitelisted) flows to enter specific zones. The proposed protection scheme assumes a hybrid network infrastructure comprising software defined network and traditional network sub-domains.

Unlike communications flows in traditional information technology networks, communications flows in industrial control networks frequently follow established patterns [4]. In the proposed protection scheme, software defined network switches are used to create security zones and apply flow whitelisting. Thus, the network switches are configured to assume two key roles in the network infrastructure: (i) they function as boundary firewalls that isolate security zones and, thus, implement zone entry conduits (ZECs); and (ii) they function as firewalls by implementing traffic whitelisting.

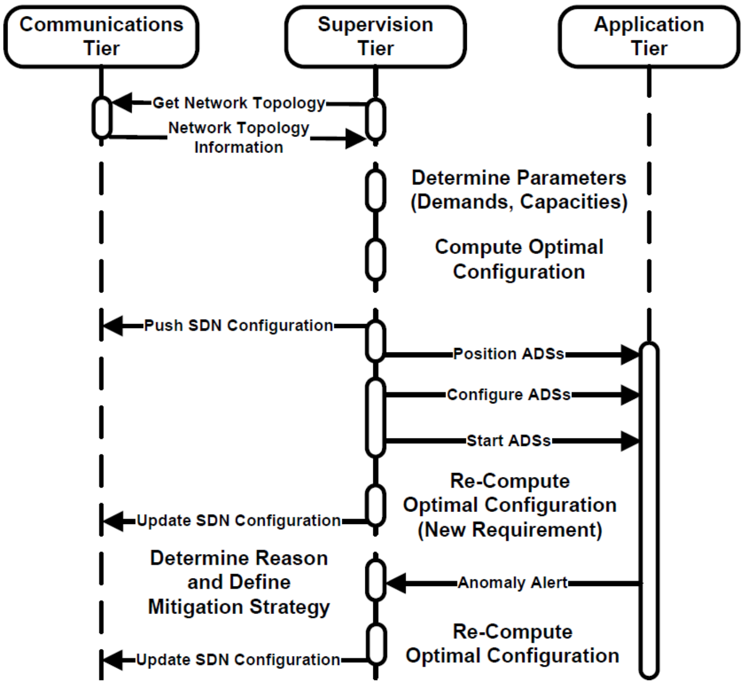

Communications Tier
Supervision Tier
Application Tier
Get Network Topology
Network Topology Information
Determine Parameters (Demands, Capacities)
Compute Optimal Configuration
Push SDN Configuration
Position ADSs
Configure ADSs
Start ADSs
Re-Compute Optimal Configuration (New Requirement)
Update SDN Configuration
Determine Reason and Define Mitigation Strategy
Anomaly Alert
Re-Compute Optimal Configuration
Update SDN Configuration

Figure 2. Conceptual diagram of the action sequences in the proposed architecture.

Figure 3 presents an example software defined network enabled topology with security features. In the topology, Switches 8 and 9 incorporate zone entry conduits for Zones 2 and 3, respectively; Zone 1 does not have a zone entry conduit because it has only exiting flows. Switches 1, 2, 3, 5, 7, 8, 9 and 12 serve as whitelisting firewalls; the remaining switches are present for redundancy purposes and to enable the insertion of additional communications flows.

The proposed scheme permits dynamic network reconfiguration in response to system maintenance, system extensions and upgrades, as well as emergency situations such as failures and cyber attacks. The principal advantage of the software defined network enabled zone entry conduits compared with traditional firewalls is that they are more flexible and are easily reconfigured using standard, open communications protocols.

From a practical deployment point of view, industrial control network switches are placed in two categories: (i) traditional dedicated network switches; and (ii) software based (software defined) network switches. The first category of switches include commercial ready-to-use single-box switches while the second category of switches are realized by running software defined networking software on computers equipped with multiple network cards. Generally, a software based switch is realized using a server running a Linux operating system

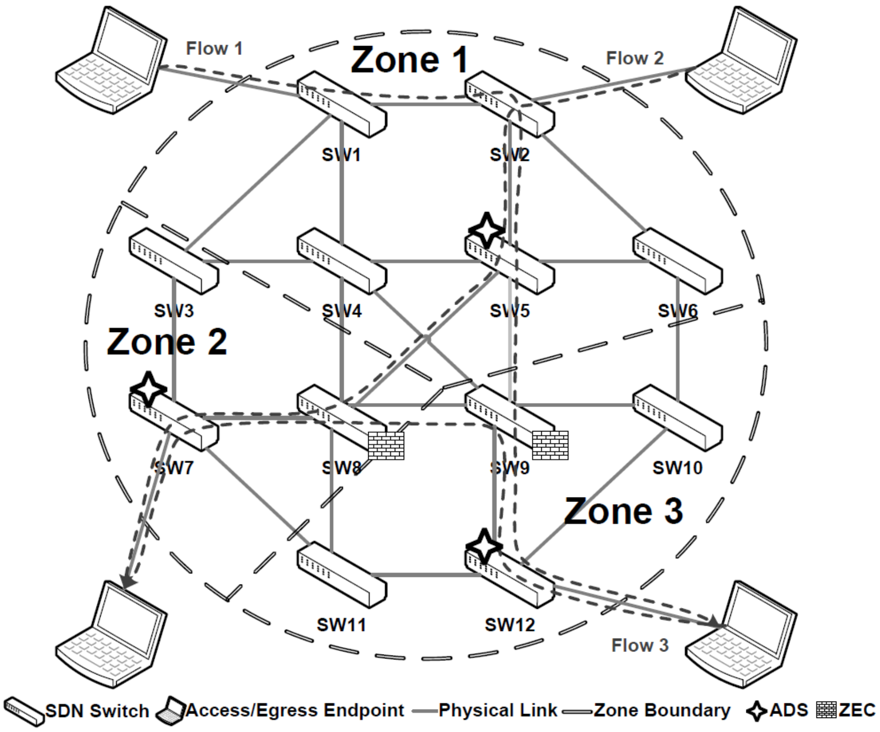


Figure 3. Example software defined network enabled topology with security features.

and dedicated software (e.g., Open vSwitch) that implements the switch features. In addition to its lower cost, a software based switch can run additional software such as an anomaly detection system. Thus, these switches can be transformed into complex protection units with the ability to whitelist flows at the network layer on one hand and the ability to monitor and report abnormal application/traffic behavior on the other hand.

3.3 Application Tier

Protection in the application tier is realized by positioning anomaly detection systems in the network topology, monitoring the behavior of the controlled processes and detecting abnormal events. The ability to implement anomaly detection functionality in software defined network switches greatly increases the flexibility of the security infrastructure by enabling dynamic network reconfiguration – specifically, changing the locations of anomaly detection systems in the network infrastructure. Incorporating an anomaly detection system in a switch enables anomaly-based packet filtering in real time. However, each anomaly detection system can only monitor a selection of flows to avoid congestion and communications delays.

An industrial control system typically has continuous sensor data flows that originate at the network edges. The protection mechanisms in the application tier can readily leverage these sensor data flows to enforce security properties.

Several anomaly detection techniques have been proposed for industrial control systems [6, 16]. A discussion of the properties and applicability of anomaly detection techniques is outside the scope of this research. Instead, an objective of this research is to determine the optimal locations of anomaly detection systems within the network topology in order to enhance communications performance and decrease installation costs.

3.4 Supervision Tier

A human operator resides at the center of the supervision tier. The human operator interacts bidirectionally with the communications and application tiers and can (re)configure the network and anomaly detection infrastructures.

Optimal Network Configuration. The human operator employs an integer linear programming (ILP) tool to obtain the optimal configuration of the industrial control network infrastructure. Note that conventional (i.e., non software defined) segments of the network correspond to static edges between software defined network switches [23].

The optimization problem has three objectives: (i) select the shortest routing paths for whitelisted flows; (ii) minimize the number of anomaly detection systems while satisfying the predefined monitoring requirements; and (iii) minimize the number of boundary switches that realize zone entry conduits in each security zone.

In addition to the three optimization objectives, three requirements are defined in the form of constraints: (i) maximum capacity limits of switches, links and anomaly detection systems cannot be exceeded; (ii) each flow must be monitored in every security zone that it traverses; and (iii) anomaly detection systems and boundary switches can only be positioned at allowed locations.

The remainder of this section formally defines the network optimization problem. Let I be the set of flows, J be the set of software defined switches and Z be the set of security zones. Furthermore, let d_i denote the demand of flow i ($i \in I$) and u_{jl} denote the capacity of link (j, l) ($j, l \in J$).

Assume that, if switches j and l are not connected, then $u_{jl} = 0$. Let u^I be the maximum processing capacity of an anomaly detection system and u^S be the maximum flow processing capacity of a software defined switch. Let g_j^z be a binary parameter with value 1 if switch j belongs to security zone z ($z \in Z$). Let r_z be a binary parameter with value 1 if an anomaly detection system can be positioned in security zone z. Furthermore, let the binary parameters x_{ij}^A and x_{ji}^E have values of 1 if the access and egress end-points of flow i are connected to switch j, respectively.

The optimization problem variables are defined as follows. Let t_{jl}^i be a binary variable with value 1 if flow i is routed on link (j, l). Let p_j be a binary variable with value 1 if switch j can support an anomaly detection system. Let q_j^i be a binary variable with value 1 if flow i is monitored by an anomaly detection system in switch j. Finally, let f_j be a binary variable with value 1 if switch j acts as a boundary firewall that realizes a zone entry conduit.

The objective function to be optimized is defined as follows:

$$Minimize: \ \alpha \sum_{i \in I, j, l \in J} t^i_{jl} + \beta \sum_{j \in J} p_j + \gamma \sum_{j \in J} f_j \tag{1}$$

where α, β and γ are integers that represent the priorities of the three objectives. These parameters can be used by a human operator to tune the optimization problem according to the priorities of the three optimization objectives embodied in Equation (1). Specifically, α expresses the importance of selecting the shortest communications path, β expresses the importance of minimizing the number of anomaly detection systems and γ expresses the importance of minimizing the number of zone entry conduits.

The network design problem specified in Equation (1) is subject to a series of constraints. Due to space constraints, only the most significant constraints are presented.

The following switch capacity constraint ensures that the maximum number of forwarding rules installed in switch j does not exceed the switch capacity:

$$\sum_{i \in I, l \in J} t^i_{jl} + \sum_{i \in I} x^E_{ji} \leq u^S \quad \forall j \in J \tag{2}$$

The following equation expresses the constraint on anomaly detection system capacity:

$$\sum_{i \in I} d_i q^i_j \leq u^I \quad \forall j \in J \tag{3}$$

The following constraint ensures that the link capacity is not exceeded:

$$\sum_{i \in I} d_i t^i_{jl} \leq u_{jl} \quad \forall j, l \in J \tag{4}$$

The following constraint ensures that an anomaly detection system is positioned in every security zone that requires monitoring:

$$\sum_{jl \in J} g^z_j p_j \geq r_z \quad \forall z \in Z \tag{5}$$

The following constraint ensures that each flow is monitored as it traverses a security zone if an anomaly detection system in positioned in the zone:

$$\epsilon \sum_{j \in J} g^z_j q^i_j \geq \sum_{jl \in J} (g^z_j + g^z_l) t^i_{jl} r_z \quad \forall i \in I, z \in Z \tag{6}$$

where ϵ is a large integer such that $\epsilon \geq \sum_{i \in I, j, l \in J} t^i_{jl}$.

Finally, the following constraint ensures that each flow enters a zone via a zone entry conduit:

$$g_j^y g_l^z t_{jl}^i \leq g_l^z f_l \quad \forall i \in I, j, l \in J, y, z \in Z, y \neq z \tag{7}$$

In order to ensure the proper functioning of the optimized infrastructure, the optimization problem incorporates additional constraints such as the selection of continuous flow paths and the selection of a switch for monitoring purposes only if the switch contains an anomaly detection system.

Solving the optimization problem yields: (i) optimal path – list of edges (software defined switch pairs) associated with each flow; (ii) optimal locations – software defined switches for positioning the anomaly detection systems; (iii) anomaly detection system monitoring rules that define the assignment of a flow to an anomaly detection system; and (iv) list of software defined switches with zone entry conduits.

Optimal Network Reconfiguration. In order to avoid complete network reconfigurations when computing a new optimal solution, the network design problem specified in Equation (1) is extended. The extension is designed to ensure: (i) the minimum number of path changes for each flow; (ii) the minimum number of anomaly detection system migrations; and (iii) the minimum number of changes with respect to boundary switches.

Three binary parameters, $\dot{t}_{jl}^i$, $\dot{p}_j$ and $\dot{f}_j$, are defined to express the optimal solution of an already-deployed network topology. Consequently, the objective function in Equation (1) is redefined as follows:

$$\begin{gathered} Minimize: \alpha \sum_{i \in I, j, l \in J} t_{jl}^i + \beta \sum_{j \in J} p_j + \gamma \sum_{j \in J} f_j + \\ \delta\Big(\sum_{i \in I, j, l \in J} (\dot{t}_{jl}^i - \dot{t}_{jl}^i t_{jl}^i) + \sum_{j \in J} (\dot{p}_j - \dot{p}_j p_j) + \sum_{j \in J} (\dot{f}_j - \dot{f}_j f_j)\Big) \end{gathered} \tag{8}$$

where the δ parameter expresses the priority of the objective that seeks to maintain the existing configuration.

The selection of the value for δ compared with the values for α, β and γ distinguishes between two cases: (i) when δ is greater than the other priority parameters, the network configuration yields a solution in which the changes are minimized; and (ii) otherwise, the existing configuration is changed to benefit from a more optimal configuration of paths, anomaly detection systems and/or zone entry conduits. Note that the constraints are the same as in the specification of the original optimization problem.

3.5 Implementation and Scalability

The OptimalFlow hierarchical software defined network controller [11] was employed to implement the proposed protection scheme. OptimalFlow provides a FlowControl unit that: (i) monitors the underlying domain for changes in the network parameter values; (ii) changes the set of installed flows according to the solutions to the optimization problem; and (iii) transparently exposes the edge ports of the software defined network to the upper tiers via a software defined switch that is accessible via the OpenFlow protocol.

The proposed solution was integrated with OptimalFlow by changing the initial optimization problem to the new network optimization problem while retaining the original modules responsible for network parameter monitoring and flow installation. An advantage of OptimalFlow is that it enables the provisioning of hierarchical n-tier systems. Consequently, the solution proposed in this research can be scaled to an n-tier architecture as well. Thus, each security zone in the network topology can be reduced to a software defined network sub-domain that is exposed to an upper level as a software defined network switch. Security zones in the next level are represented by optimally-connected virtual software defined network switches.

4. Experimental Results

This section presents the results of the experiments that were conducted to qualitatively and quantitatively assess the performance of the proposed protection scheme. The protection scheme was evaluated using a realistic scenario on an emulated software defined networking infrastructure and simulated industrial data.

The experiments were conducted on a typical industrial network topology. Software defined switches were distributed in security zones structured according to the control system security guide from Tofino Security [5] (see Figure 4). The topology comprised 35 software defined switches connected via 86 physical links that offered redundant communications between two segments. The network topology had thirteen distinct security zones: External Network (X); Corporate Boundary Management (B); Enterprise Network (E); Process Information Network (I); Process Information Management Network (M); Supervisory Networks (J1, J2); Control Networks (C1, C2); and Process Networks (P1, P2).

First, the applicability of the proposed protection scheme to the scenario described above was evaluated using the OptimalFlow software defined network controller. The experiment was conducted using the Mininet network emulator [17]. The $\delta > \beta > \gamma > \alpha$ priority parameter configuration was employed and IEC 61850 network traffic was generated using the MATLAB Power System Analysis Toolbox (PSAT) running the IEEE 14-bus process model [19].

The analysis focused on three use cases: (i) complete network configuration; (ii) partial network configuration in the case of a new flow; and (iii) partial network configuration in the case of link failure.

Four flows, Flows 1-4, were defined for the first use case. Figure 4 shows the optimal configuration. The optimizer minimized the number of anomaly detection systems and zone entry conduits by reducing the number of communications switches used to route flows. On the other hand, switches were selected to ensure the shortest routing path. Obviously, the solution involved

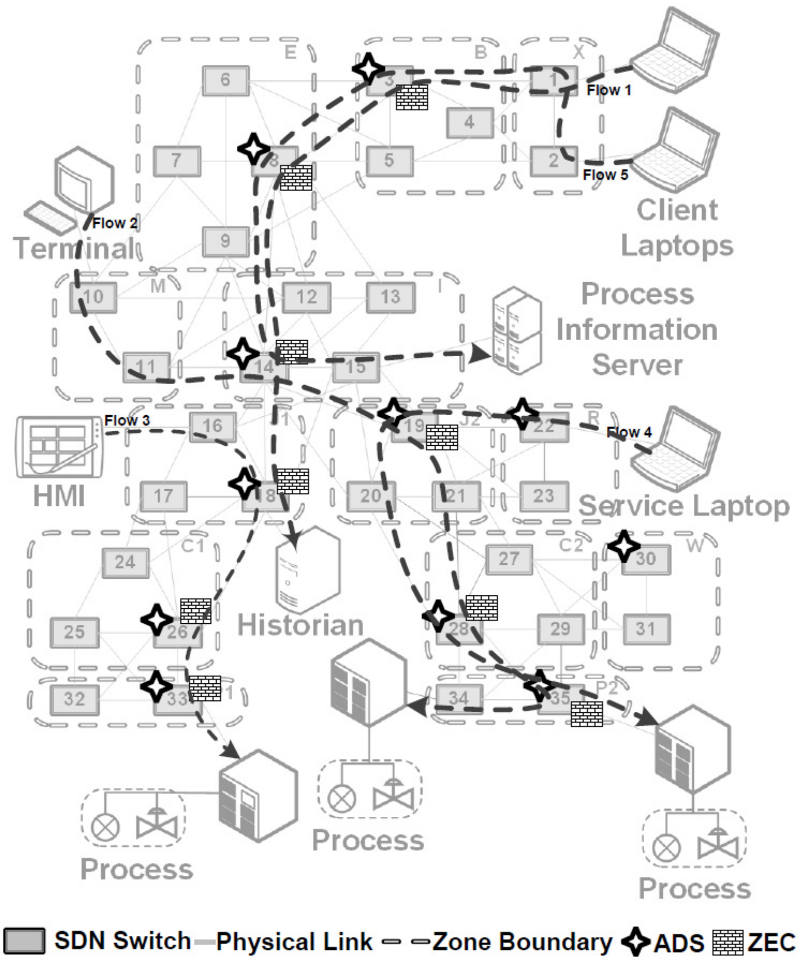


Figure 4. Experimental architecture with an SDN enabled topology.

a trade-off between the shortest routing path and the minimum numbers of anomaly detection systems and zone entry conduits. For example, Flow 2 was routed on a path comprising six links whereas the shortest path only required five links along Switches 10-11-14-20-28-35. This is because the optimizer selected Switch 19 for provisioning an anomaly detection system and as a zone entry conduit. Therefore, the optimizer reduced the number of provisioned anomaly detection systems and zone entry conduits by routing Flow 2 through Switch 19.

In the second use case, a new flow, Flow 5, was added to the network topology and the optimal network configuration was recomputed. Note that the initially-configured flows and the anomaly detection system and zone entry conduit locations were not modified. This was achieved by setting the δ parameter to be larger than the β, γ, α parameters in Equation (8). Figure 4 shows the results of the optimization and the routing of Flow 5.

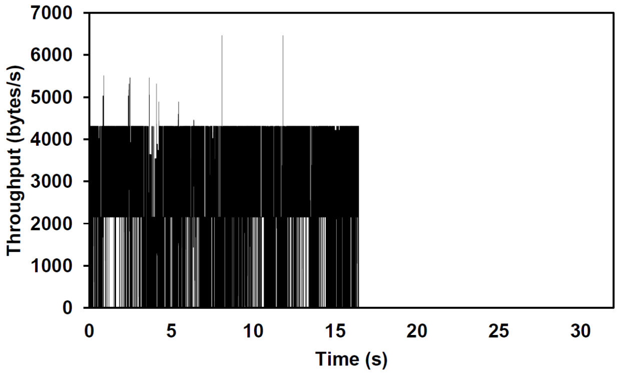
In the third use case, the physical link between Switches 8 and 14 was made to fail by assigning it a capacity of zero. This action automatically triggered the recomputation of the optimal configuration using OptimalFlow. The optimal solution re-routed Flows 1 and 5 on Switches 8, 9 and 14. Subsequently, routing rules were deleted on the link between Switches 8 and 14. The rest of the configuration was not affected because δ was selected to be greater than the other priority parameters.

The optimization and software defined network deployment times were 0.91 s and 1.17 s for the first use case, 1.11 s and 0.76 s for the second use case and 1.18 s and 0.46 s for the third use case, respectively. These results demonstrate that, even if the optimization times are increased slightly when performing network reconfiguration during the second and third use cases, the software defined network times are considerably low. Thus, the total number of reconfiguration operations are also reduced significantly.

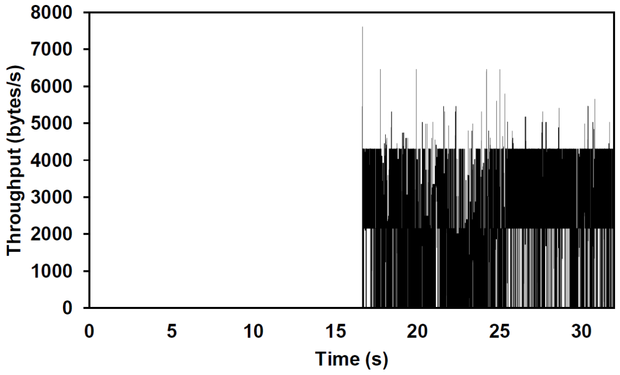
Next, the effects of network reconfiguration on industrial traffic were investigated. The experiment involved the execution of a MATLAB Power System Analysis Toolbox instance and an IEC 61850 server on the host attached to Switch 33. The traffic was monitored using an IEC 61850 client on the machine attached to Switch 16 and by positioning two traffic capturing units, one on the link between Switches 16 and 18 and the other on the link between Switches 16 and 1. In the next step, Flow 3 was re-routed from the link between Switches 16 and 18 to the path involving Switches 16, 17 and 18.

Figure 5 shows the effects of flow re-routing on the traffic monitored at the network segments of interest. Note that the throughput on the link between Switches 16 and 17 was initially around 4.5 KB/s while there was no traffic on the link between Switches 16 and 18. After the re-routing was triggered (after 16 seconds), the traffic throughput exhibited slight changes. During the re-routing, no data was transferred in Flow 3 for a short time period (around 200 ms). This corresponds to the time required to delete the forwarding rules for the failed link and to install new rules for the alternative links. This time period is always shorter than the total software defined network deployment time because it does not cover the communications and processing overhead between OptimalFlow and the software defined network controller. After the re-routing was completed, a temporary increase in the traffic was observed due to the TCP re-transmissions required to recover the lost packets.

Finally, the effects of parameter value changes on the optimization time were investigated. The experiments measured the optimization times for the two topologies presented in Figures 3 and 4 for different numbers of flows. The AIMMS optimization tool [2] was executed on a laptop with an Intel i7 quad core processor, 8 GB RAM and an SSD hard drive. Each configuration was repeated five times. Table 1 presents the optimization times for various parameter values and scenarios for the different configurations.



(a) Traffic between Switches 16 and 17.



(b) Traffic between Switches 16 and 18.

Figure 5. Effects of re-routing network traffic.

Table 1. Optimization times for various parameter values and scenarios.

Flows	**Optimization Time (s)**					
	Topology in Figure 3			**Topology in Figure 4**		
	Min	**Avg**	**Max**	**Min**	**Avg**	**Max**
10	0.09	0.12	0.17	1.59	1.71	1.75
30	0.52	0.66	1.17	5.67	6.51	7.63
50	1.47	1.71	2.05	9.70	10.28	18.89
100	4.34	4.67	6.44	30.47	48.28	56.06

The experimental results demonstrate that the number of flows and the complexity of the topology impact the optimization time. In both the topologies, increasing the number of flows significantly increases the optimization time. Furthermore, increasing the complexity of the topology (i.e., numbers of software defined switches, links and security zones) also increases the optimization time. Nonetheless, the optimization time never exceeded one minute even in the worst-case scenario and was under ten seconds in the majority of cases.

Note that the optimization time can be reduced further by solving the reduced optimization problem specified in Equation (8) and by partitioning the network to leverage the ability of OptimalFlow to solve hierarchical optimization problems. The combination of these strategies enhances the applicability of the proposed network configuration strategy in real industrial control environments. However, current software defined networking technologies – and, thus, the proposed methodology itself – may not be applicable in scenarios involving time-critical communications where network packets need to be delivered within 10 to 20 ms. Further research is needed to speed up the methodology for network reconfiguration in industrial control environments with tight timing constraints.

5. Conclusions

This research has proposed a novel three-tier protection architecture that endows industrial control systems with response and reconfiguration capabilities to cope with failures and cyber attacks. The communications tier, which leverages software defined networking, enables dynamic flow configuration, reconfiguration and whitelisting; the application tier incorporates a distributed anomaly detection infrastructure; and the supervision tier, which incorporates a human in the loop, controls and synchronizes all three tiers. The protection solution is obtained by optimizing industrial control network flows and the numbers and locations of anomaly detection systems and software defined firewalls. The applicability of the proposed architecture is demonstrated by implementing it using the OptimalFlow hierarchical software defined network controller and conducting qualitative and quantitative assessments involving three use cases.

Future research will focus on developing a more efficient optimization algorithm. Additionally, heuristics will be identified and incorporated in the algorithms to reduce the computational time and enhance their application in industrial control environments with large infrastructures and tight timing constraints.

Acknowledgement

This research was supported by a Marie Curie FP7 Integration Grant within the 7th European Union Framework Programme (Grant No. PCIG14-GA-2013-631128).

References

[1] R. Ackerman, Software-defined networking looms as next big technology, *Signal*, May 12, 2014.

[2] Advanced Analytics, Prescriptive Analytics and Supply Chain Management, AIMMS, Bellevue, Washington (`aimms.com`), 2017.

[3] R. Anderson and R. Hundley, The Implications of COTS Vulnerabilities for the DoD and Critical U.S. Infrastructures: What Can/Should the DoD Do? P-8031, RAND Corporation, Santa Monica, California, 1998.

[4] R. Barbosa, R. Sadre and A. Pras, Flow whitelisting in SCADA networks, *International Journal of Critical Infrastructure Protection*, vol. 6(3-4), pp. 150–158, 2013.

[5] E. Byres, Using ANSI/ISA-99 Standards to Improve Control System Security, Version 1.1, White Paper, Tofino Security, Lantzville, Canada, 2012.

[6] M. Caselli, E. Zambon, J. Amann, R. Sommer and F. Kargl, Specification mining for intrusion detection in networked control systems, *Proceedings of the Twenty-Fifth USENIX Security Symposium*, pp. 791–806, 2016.

[7] A. Cherepanov, BlackEnergy by the SSHBearDoor: Attacks against Ukrainian news media and electric industry, *WeLiveSecurity*, January 3, 2016.

[8] L. Combita, J. Giraldo, A. Cardenas and N. Quijano, Response and reconfiguration of cyber-physical control systems: A survey, *Proceedings of the Second IEEE Colombian Conference on Automatic Control*, 2015.

[9] N. Dorsch, F. Kurtz, H. Georg, C. Hagerling and C. Wietfeld, Software-defined networking for smart grid communications: Applications, challenges and advantages, *Proceedings of the IEEE International Conference on Smart Grid Communications*, pp. 422–427, 2014.

[10] R. Drath and A. Horch, Industrie 4.0: Hit or hype? [Industry Forum], *IEEE Industrial Electronics*, vol. 8(2), pp. 56–58, 2014.

[11] B. Genge and P. Haller, A hierarchical control plane for software-defined-network-based industrial control systems, *Proceedings of the IFIP Networking Conference and Workshops*, pp. 73–81, 2016.

[12] M. Hagerott, Stuxnet and the vital role of critical infrastructure operators and engineers, *International Journal of Critical Infrastructure Protection*, vol. 7(4), pp. 244–246, 2014.

[13] M. Hermann, T. Pentek and B. Otto, Design principles for Industrie 4.0 scenarios, *Proceedings of the Forty-Ninth Hawaii International Conference on System Sciences*, pp. 3928–3937, 2016.

[14] X. Jin, H. Liu, R. Gandhi, S. Kandula, R. Mahajan, M. Zhang, J. Rexford and R. Wattenhofer, Dynamic scheduling of network updates, *ACM SIGCOMM Computer Communication Review*, vol. 44(4), pp. 539–550, 2014.

[15] J. Kempf, E. Bellagamba, A. Kern, D. Jocha, A. Takacs and P. Skoldstrom, Scalable fault management for OpenFlow, *Proceedings of the IEEE International Conference on Communications*, pp. 6606–6610, 2012.

[16] I. Kiss, B. Genge and P. Haller, A clustering-based approach to detect cyber attacks in process control systems, *Proceedings of the Thirteenth International Conference on Industrial Informatics*, pp. 142–148, 2015.

[17] B. Lantz, B. Heller and N. McKeown, A network in a laptop: Rapid prototyping for software-defined networks, *Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks*, article 19, 2010.

[18] J. Lee, B. Bagheri and H. Kao, A cyber-physical systems architecture for Industry 4.0 based manufacturing systems, *Manufacturing Letters*, vol. 3, pp. 18–23, 2015.

[19] F. Milano and M. Anghel, Impact of time delays on power system stability, *IEEE Transactions on Circuits and Systems – I: Regular Papers*, vol. 59(4), pp. 889–900, 2012.

[20] E. Molina, E. Jacob, J. Matias, N. Moreira and A. Astarloa, Using software defined networking to manage and control IEC 61850 based systems, *Computers and Electrical Engineering*, vol. 43, pp. 142–154, 2015.

[21] NEC Corporation, Software-Defined Networking (SDN) Solution, West Nippon Expressway Company Limited (NEXCO-West), Case Study, Tokyo, Japan (`www.nec.com/en/case/w-nexco/index.html`), 2016.

[22] Open Networking Foundation, Software-Defined Networking (SDN) Definition, Menlo Park, California (`www.opennetworking.org/sdn-resources/sdn-definition`), 2017.

[23] H. Sandor, B. Genge and G. Sebestyen-Pal, Resilience in the Internet of Things: The software defined networking approach, *Proceedings of the IEEE International Conference on Intelligent Computer Communication and Processing*, pp. 545–552, 2015.

[24] S. Sharma, D. Staessens, D. Colle, M. Pickavet and P. Demeester, OpenFlow: Meeting carrier-grade recovery requirements, *Computer Communications*, vol. 36(6), pp. 656–665, 2013.

[25] K. Stouffer, J. Falco and K. Scarfone, Guide to Industrial Control Systems (ICS) Security, NIST Special Publication 800-82, Revision 1, National Institute of Standards and Technology, Gaithersburg, Maryland, 2011.

[26] S. Vissicchio, L. Vanbever and O. Bonaventure, Opportunities and research challenges of hybrid software defined networks, *ACM SIGCOMM Computer Communication Review*, vol. 44(2), pp. 70–75, 2014.

Chapter 10

THREAT ANALYSIS OF AN ELEVATOR CONTROL SYSTEM

Raymond Chan and Kam-Pui Chow

Abstract Programmable logic controllers are key components of industrial control systems that are used across the critical infrastructure. The infamous Stuxnet malware attacked programmable logic controllers that managed uranium hexafluoride centrifuges in Iran's Natanz facility, causing the centrifuges to operate outside their designed limits while leading plant operators to believe all was well. This attack and others have rendered the task of securing programmable logic controllers an important problem. Most research in the area has focused on network-level intrusion detection and protection mechanisms. Few research efforts have specifically considered threats to the internal networks of industrial control systems, which include connections from the computer platforms that manage programmable logic controllers. This chapter analyzes the threats to the internal environment of an elevator control system that engages a Siemens programmable logic controller. Several approaches for mitigating the threats are presented.

Keywords: Programmable logic controllers, elevator control, threats, mitigation

1. Introduction

Industrial control systems are used across the critical infrastructure to manage physical processes. Industrial control systems include supervisory control and data acquisition (SCADA) systems and distributed control systems (DCSs), both of which incorporate component devices such as programmable logic controllers (PLCs). Programmable logic controllers are connected to human-machine interfaces (HMIs) to enable command and control by human operators and to engineering/development workstations for configuration, programming and diagnostics. Programmable logic controllers commonly execute ladder logic programs to perform their monitoring and control activities.

Traditionally, industrial control systems operated using proprietary protocols in closed (air-gapped) networks. However, many industrial control net-

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 175–192, 2017.
https://doi.org/10.1007/978-3-319-70395-4_10

works are now connected to external networks – even the Internet – to support remote operations, configuration and diagnostics. This exposes industrial control systems and the critical assets they manage to external attacks in addition to attacks by malicious insiders.

Several issues impact the security of industrial control systems. One is that engineers and operators are more concerned about availability than security. Another is the lack of a security mindset. Yet another is the fact that the scale, complexity and diversity of industrial control systems render the implementation of security mechanisms extremely cost-prohibitive. Moreover, adding extra layers of protection can significantly affect the performance and reliability of industrial control systems – asset owners and operators are reluctant to implement security mechanisms that can affect command and control.

Stuxnet has demonstrated that a sophisticated adversary can gain access to an extremely well-protected industrial control network. Once inside the network, the adversary can leverage the fact that programmable logic controllers, because they have limited memory and processing power, are unable to implement security controls such as encryption and intrusion detection. This makes it possible to extract and reprogram the ladder logic to change the behavior of the control system.

This research focuses on programmable logic controllers, arguably the most vulnerable components of industrial control systems. It analyzes the threats to the internal environment of an elevator control system that engages a Siemens programmable logic controller and presents a proof-of-concept program that demonstrates the feasibility of attacks. Also, it describes several approaches for mitigating the threats to the elevator control system.

2. Related Work

Considerable research has focused on industrial control system security (see, e.g., [9]) and managing the risks (see, e.g., [15]). Hadziosmanovic et al. [6] discuss the challenges involved in protecting industrial control system hosts and networks. Several high-level solutions have been developed for protecting industrial control systems (see, e.g., [12, 14]). Wei and Ji [17] have proposed a three-layer architecture for enhancing the security and reliability of industrial control systems. Cohen [4] has specified a reference architecture and guidelines for securing industrial control networks. Jie and Li [7] have analyzed industrial control system security risks and have proposed strategies for protecting control devices. Ghena et al. [5] have leveraged wireless access to maliciously control traffic lights. These research efforts discuss security problems and solutions for industrial control systems, but ignore threats to the internal networks of industrial control systems.

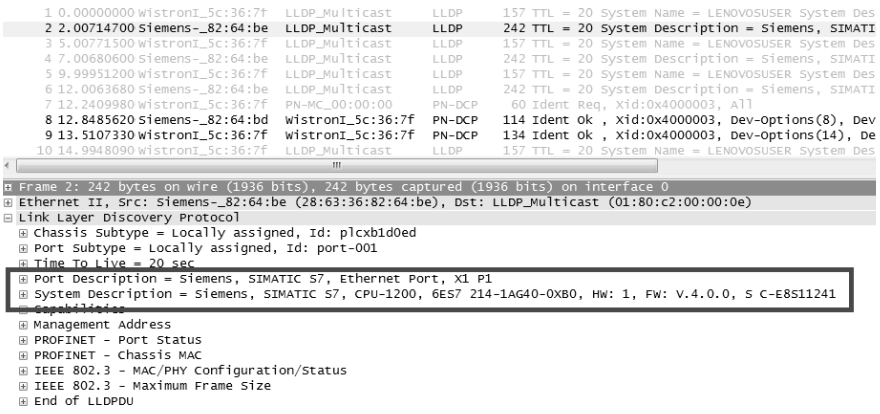
Several researchers have focused on discovering vulnerabilities in industrial control networks. Beresford [2] has analyzed the Siemens S7 protocol and has developed exploits that target Siemens programmable logic controllers. In particular, Beresford demonstrated that it is possible to bypass the authentication protocol and perform memory-read, write-logic and other attacks. Timorin [16] has demonstrated how to capture S7 challenge-response messages and perform replay attacks. Also, Timorin has analyzed the Siemens Total Integrated Automation (TIA) Portal project file, and has shown how to extract the SHA-protected password and change user permissions in the file. Korkmaz et al. [8] have discovered a time delay attack on a control system that could result in the failure of an entire power generation facility. Abe et al. [1] have identified several cyber attacks on Internet-connected control systems; these attacks leverage malware that sends STOP and RESET commands to programmable logic controllers, negatively impacting the industrial control system and the underlying process.

Cardenas et al. [3] have specified a threat model that covers outsider attacks, key-compromise attacks and insider attacks on SCADA systems. Hadziosmanovic et al. [6] have classified industrial control system threats into system-related threats and process-related threats. McLaughlin and Zonouz [11] have introduced a threat model that covers the use of any industrial control system component to upload malicious code to a programmable logic controller. Malchow et al. [10] have proposed a threat model that covers a scenario where an adversary can control an engineering workstation using malware and inject malicious code into a programmable logic controller.

In general, most industrial control system security approaches focus on redesigning the entire architecture or incorporating security mechanisms throughout the architecture. In the case of operational industrial control systems, there are certain latent security problems, especially pertaining to the internal environment, which often has fewer security mechanisms than external networks – examples are the development zone and human-machine control zone. As a result, this research seeks to identify potential vulnerabilities that enable practical and reproducible attacks on the internal networks of industrial control systems. An elevator control system is used as a case study because it is small and ubiquitous, but still a representative, real-world industrial control system. Additionally, the elevator system has several sensors and safety protection mechanisms that can be targeted to cause harm.

3. Threat Model

This research assumes that an adversary can gain access to the internal network of an industrial control system by various means and is able to launch attacks. The attacks are assumed to target: (i) confidentiality; (ii) integrity; or (iii) availability. The proposed threat model focuses on Siemens programmable logic controllers. The model considers the attack capabilities of an adversary upon gaining access to an industrial control network; these capabilities are in addition to implanting malware on a programmable logic controller. In general, it is difficult for an adversary to enter an industrial control network via a phishing email, external USB thumb drive or even as an insider.



```
 1 0.00000000 WistronI_5c:36:7f   LLDP_Multicast     LLDP     157 TTL = 20 System Name = LENOVOSUSER System Des
 2 2.00714700 Siemens-_82:64:be   LLDP_Multicast     LLDP     242 TTL = 20 System Description = Siemens, SIMATI
 3 5.00771500 WistronI_5c:36:7f   LLDP_Multicast     LLDP     157 TTL = 20 System Name = LENOVOSUSER System Des
 4 7.00680600 Siemens-_82:64:be   LLDP_Multicast     LLDP     242 TTL = 20 System Description = Siemens, SIMATI
 5 9.99951200 WistronI_5c:36:7f   LLDP_Multicast     LLDP     157 TTL = 20 System Name = LENOVOSUSER System Des
 6 12.0063680 Siemens-_82:64:be   LLDP_Multicast     LLDP     242 TTL = 20 System Description = Siemens, SIMATI
 7 12.2409980 WistronI_5c:36:7f   PN-MC_00:00:00     PN-DCP    60 Ident Req, Xid:0x4000003, All
 8 12.8485620 Siemens-_82:64:bd   WistronI_5c:36:7f  PN-DCP   114 Ident Ok , Xid:0x4000003, Dev-Options(8), Dev
 9 13.5107330 WistronI_5c:36:7f   WistronI_5c:36:7f  PN-DCP   134 Ident Ok , Xid:0x4000003, Dev-Options(14), De
10 14.9948090 WistronI_5c:36:7f   LLDP_Multicast     LLDP     157 TTL = 20 System Name = LENOVOSUSER System Des

⊞ Frame 2: 242 bytes on wire (1936 bits), 242 bytes captured (1936 bits) on interface 0
⊞ Ethernet II, Src: Siemens-_82:64:be (28:63:36:82:64:be), Dst: LLDP_Multicast (01:80:c2:00:00:0e)
⊟ Link Layer Discovery Protocol
  ⊞ Chassis Subtype = Locally assigned, Id: plcxb1d0ed
  ⊞ Port Subtype = Locally assigned, Id: port-001
  ⊞ Time To Live = 20 sec
  ⊞ Port Description = Siemens, SIMATIC S7, Ethernet Port, X1 P1
  ⊞ System Description = Siemens, SIMATIC S7, CPU-1200, 6ES7 214-1AG40-0XB0, HW: 1, FW: V.4.0.0, S C-E8S11241
  ⊞ Capabilities
  ⊞ Management Address
  ⊞ PROFINET - Port Status
  ⊞ PROFINET - Chassis MAC
  ⊞ IEEE 802.3 - MAC/PHY Configuration/Status
  ⊞ IEEE 802.3 - Maximum Frame Size
  ⊞ End of LLDPDU
```

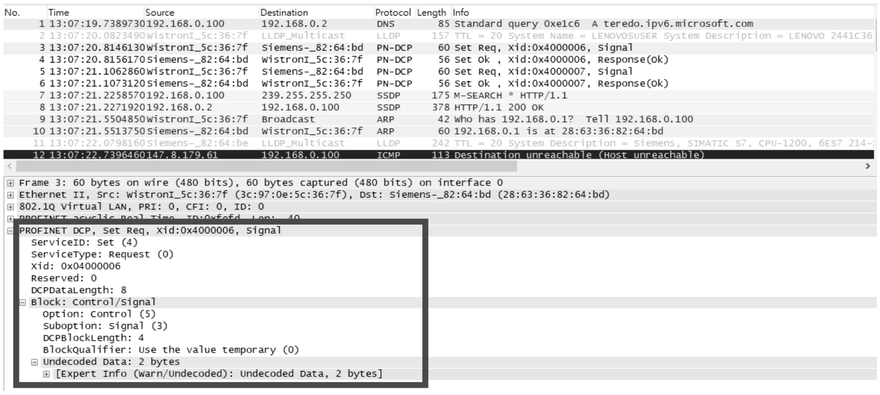
Figure 1. Leveraging LLDP broadcast messages to discover device information.

3.1 Confidentiality Threats

Most industrial control systems do not use encryption and authentication. As a result, an adversary who has access to a network and captures communications data can perform the following attacks:

- **Programmable Logic Controller Discovery Attack:** An adversary can discover all the programmable logic controllers in an internal industrial control network by connecting to the network and capturing network packets using a tool such as Wireshark. Figure 1 illustrates the network packet capture process in the case of a Siemens programmable logic controller that uses the Link Layer Discovery Protocol (LLDP) to broadcast its presence. Important information such as the MAC address, model number, CPU information, hardware information and firmware information are transferred in plaintext. An adversary who captures the internal network traffic can easily obtain detailed information about the programmable logic controllers that could be used to plan specific attacks.

- **False Command or Signal Injection Attack:** Most industrial control system designs assume that all the devices operate in a trusted and closed network. No encryption and authentication mechanisms are implemented between human-machine interfaces and programmable logic controllers. Moreover, many human-machine interfaces and programmable logic controllers are connected to external networks, including the Internet. An adversary who accesses the network interface could inject various control commands. For example, injecting a STOP or RESET command would move a programmable logic controller to the STOP mode and the controller would not operate until it receives a START command.



```
No.   Time             Source              Destination        Protocol Length Info
    1 13:07:19.7389730 192.168.0.100       192.168.0.2        DNS         85 Standard query 0xe1c6  A teredo.ipv6.microsoft.com
    2 13:07:20.0823490 WistronI_5c:36:7f   LLDP_Multicast     LLDP       157 TTL = 20 System Name = LENOVOSUSER System Description = LENOVO 2441C36
    3 13:07:20.8146130 WistronI_5c:36:7f   Siemens-_82:64:bd  PN-DCP      60 Set Req, Xid:0x4000006, Signal
    4 13:07:20.8156170 Siemens-_82:64:bd   WistronI_5c:36:7f  PN-DCP      56 Set Ok , Xid:0x4000006, Response(Ok)
    5 13:07:21.1062860 WistronI_5c:36:7f   Siemens-_82:64:bd  PN-DCP      60 Set Req, Xid:0x4000007, Signal
    6 13:07:21.1073120 Siemens-_82:64:bd   WistronI_5c:36:7f  PN-DCP      56 Set Ok , Xid:0x4000007, Response(Ok)
    7 13:07:21.2258570 192.168.0.100       239.255.255.250    SSDP       175 M-SEARCH * HTTP/1.1
    8 13:07:21.2271920 192.168.0.2         192.168.0.100      SSDP       378 HTTP/1.1 200 OK
    9 13:07:21.5504850 WistronI_5c:36:7f   Broadcast          ARP         42 Who has 192.168.0.1?  Tell 192.168.0.100
   10 13:07:21.5513750 Siemens-_82:64:bd   WistronI_5c:36:7f  ARP         60 192.168.0.1 is at 28:63:36:82:64:bd
   11 13:07:22.0798160 Siemens-_82:64:be   LLDP_Multicast     LLDP       242 TTL = 20 System Description = Siemens, SIMATIC S7, CPU-1200, 6ES7 214-
   12 13:07:22.7396460 147.8.179.61        192.168.0.100      ICMP       113 Destination unreachable (Host unreachable)

⊞ Frame 3: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
⊞ Ethernet II, Src: WistronI_5c:36:7f (3c:97:0e:5c:36:7f), Dst: Siemens-_82:64:bd (28:63:36:82:64:bd)
⊞ 802.1Q Virtual LAN, PRI: 0, CFI: 0, ID: 0
⊞ PROFINET acyclic Real Time, ID:0xfefd, Len: 40
⊟ PROFINET DCP, Set Req, Xid:0x4000006, Signal
    ServiceID: Set (4)
    ServiceType: Request (0)
    Xid: 0x04000006
    Reserved: 0
    DCPDataLength: 8
  ⊟ Block: Control/Signal
      Option: Control (5)
      Suboption: Signal (3)
      DCPBlockLength: 4
      BlockQualifier: Use the value temporary (0)
    ⊟ Undecoded Data: 2 bytes
      ⊞ [Expert Info (Warn/Undecoded): Undecoded Data, 2 bytes]
```

Figure 2. Manipulating the LED of a programmable logic controller.

Many programmable logic controllers in production systems have been operational for several years and often have outdated firmware. McLaughlin and Zonouz [11] have developed the CaFDI tool that sends false data to programmable logic controllers. Abe et al. [1] have demonstrated that sending a STOP or RESET command is adequate to disrupt most programmable logic controllers. An adversary could also send a fake sensor input signal in order to alter an output, potentially causing the entire industrial control system to crash [13].

A personal computer installed with the Siemens Step 7 software can send a discovery command to flash the LED light of a Siemens PLC, leading the operator to believe that the programmable logic controller is malfunctioning. Figure 2 shows the command sent from a personal computer to manipulate the LED of a Siemens programmable logic controller.

- **Ladder Logic Program Leakage Attack:** A ladder logic program specifies how a programmable logic controller should process input signals and generate responses in the form of output signals. A Siemens programmable logic controller implements a control command that enables an engineer to download the ladder logic program from the controller. An adversary who has the requisite access can request the programmable logic controller to send its ladder logic program, enabling the adversary to access and reverse engineer the program.

3.2 Integrity Threats

As mentioned above, programmable logic controllers usually do not implement any authentication checks. An adversary who knows the IP address of a programmable logic controller could seize control of the device and transmit messages. The operator at the human-machine interface would be unable to determine that the messages do not come from an authorized entity.

The following attacks target the integrity of an industrial control system:

- **Response Injection Attack:** Since it is not possible to determine whether or not a message is sent by an authorized programmable logic controller, an adversary can execute a man-in-the-middle attack and inject or alter a response message variable or sensor value sent by a programmable logic controller to display false information on a human-machine interface. Any number of replay attacks are also possible.

- **Ladder Logic Modification Attack:** No authentication checks are performed when uploading a ladder logic program from a development workstation to a programmable logic controller. Thus, an adversary to can create a malicious ladder logic program and upload it to the programmable logic controller so that it replaces the original program. This is possible because no software attestation or protection mechanisms are implemented to verify that the new ladder logic program is authentic. If the new ladder logic program sends valid outputs to the human-machine interface, a behavior-based protection mechanism would be unable to detect the attack at the network level.

3.3 Availability Threats

Programmable logic controllers are devices with low processing power that are designed to operate in real time. It is possible to send malformed packets that delay or disrupt the responses of a programmable logic controller. An example attack on availability is:

- **Denial-of-Service Attack:** One type of denial-of-service attack involves the transmission of malicious commands or packets that delay the response or even crash a programmable logic controller. Another type of attack targets the human-machine interface by installing malware on programmable logic controllers. In this case, an adversary uploads malicious ladder logic programs to all the programmable logic controllers in the industrial control network. When certain attack criteria are satisfied (e.g., time or process conditions), the programmable logic controllers could be made to disrupt the human-machine interface by simultaneously sending it malicious packets.

4. Elevator System Case Study

An elevator system testbed was used to validate the proposed threat model and demonstrate the feasibility of attacks.

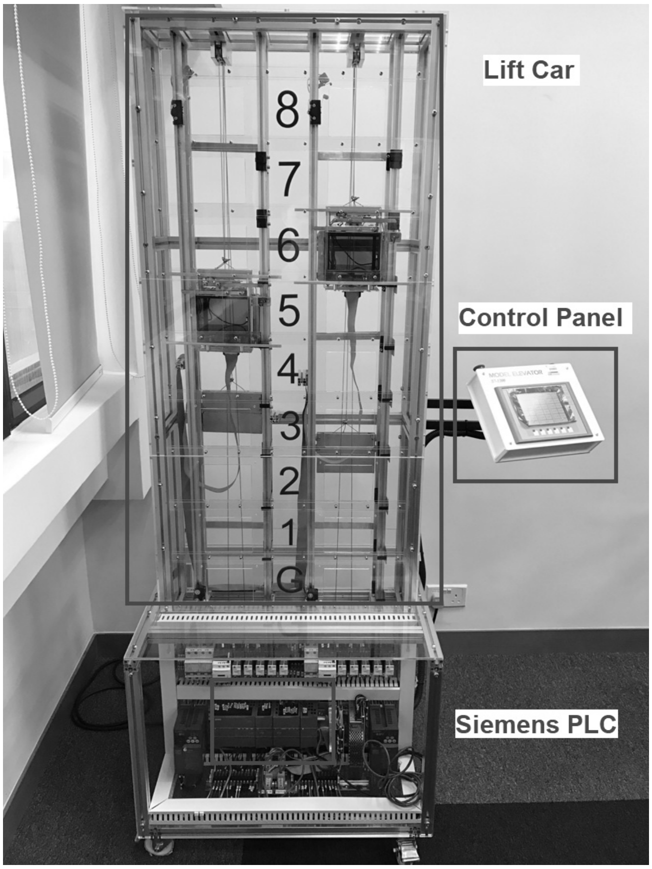


Figure 3. Elevator system testbed.

4.1 Experimental Setup

Figure 3 presents the elevator system testbed used in the experiments. The control circuit of the model elevator system comprised a DC power supply, programmable logic controller, magnetic circuit breaker contactors, relays and variable frequency drives. The control circuit provided an interface for controlling high power devices (e.g., three-phase AC motor) via small control signals transmitted by the programmable logic controller. The KTP600 Control Panel served as the human-machine interface for an operator to obtain status data and control elevator operation. A personal computer installed with the Siemens TIA Portal and connected to the switch of the elevator system (i.e., internal network) was assumed to be exploited by the adversary to launch attacks.

The programmable logic controller was configured to control the model elevator. The model elevator system had two cars that operated over nine floors. Sensors were located on every floor for elevator car positioning. As in a typical modern passenger elevator, the model had two call buttons (car call and hall call) to choose a floor. Door controls were incorporated to close or reopen the doors. An object in the path of the moving doors was detected by sensors or were handled by manually activating a switch that reopened the doors. Otherwise, the elevator doors closed after a preset time.

A driving mechanism moved the elevator car up and down. The elevator control system coordinated the movements of the two elevator cars to provide optimal service by reducing passenger wait times.

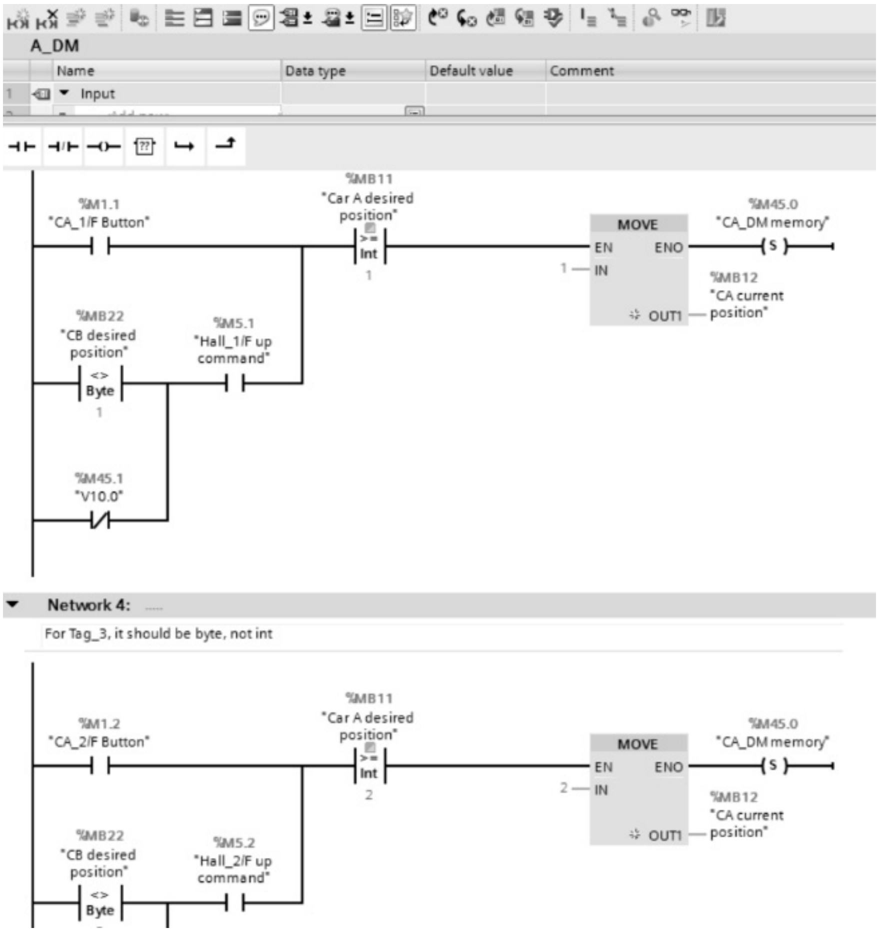
In the experiments, the ladder logic program for elevator control was uploaded to the Siemens S7-1200 programmable logic controller (Firmware v4.0) using the Siemens TIA Portal v13. The Siemens S7-1200 programmable logic controller is one of the popular models available in the market and has been deployed in numerous infrastructure assets.

The experiments assumed that the adversary had installed malware in the personal computer connected to the elevator switch, which enabled him to discover, monitor and control the programmable logic controller in the elevator system. Specifically, the adversary could perform three actions: (i) discover the presence of the programmable logic controller in the network; (ii) query information about the programmable logic controller; and (iii) launch attacks to control and crash the programmable logic controller.

4.2 S7 Base Protocol and Configuration

The S7 base protocol is used by Siemens programmable logic controllers for communications. The protocol has been exploited by Beresford [2] and several third-party software tools are available to control Siemens programmable logic controllers. Starting with Firmware V4.0, Siemens updated the S7 protocol (S7 Plus) for the Siemens S7-1200/1500 programmable logic controller model to provide additional security features. However, this research discovered that the S7 base protocol can still be used to query and command a new programmable logic controller due to the design decision to maintain compatibility with older versions. In particular, the experiments used the S7 base protocol to retrieve information from the programmable logic controller and then change its behavior.

The S7 base protocol incorporates commands for querying and changing the digital inputs (PA) and digital outputs (PE) of the programmable logic controller. No configuration settings were available for protecting access to the PA and PE entries. In fact, this research discovered that the S7 base protocol could be used to communicate with the programmable logic controller without the authentication checks required by the new S7 protocol implemented in the controller. Indeed, the experiments confirmed that the new S7 protocol does not provide any protection to the digital inputs or outputs when the base protocol is used.



For Tag_3, it should be byte, not int

Figure 4. Ladder logic program used by the elevator system.

Based on these findings, a proof-of-concept ladder logic program was developed to send S7 base protocol commands to change the behavior of the elevator system. Figure 4 shows the ladder logic program that manages the elevator system. Figure 5 shows the attack entry point to the elevator system.

4.3 PLC Discovery Attack

As discussed in the section on confidentiality threats, an adversary has to first locate the personal computer with Siemens Step 7 installed and the Siemens programmable logic controller in the internal industrial control network. Information about these devices may be collected in a passive or ac-

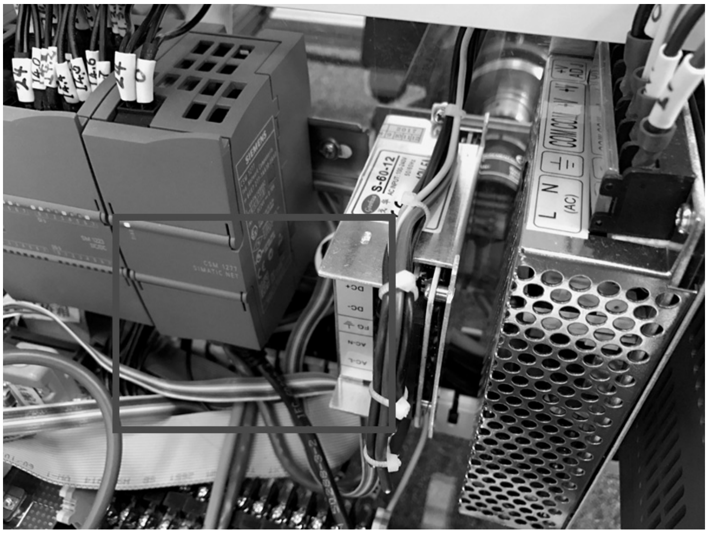


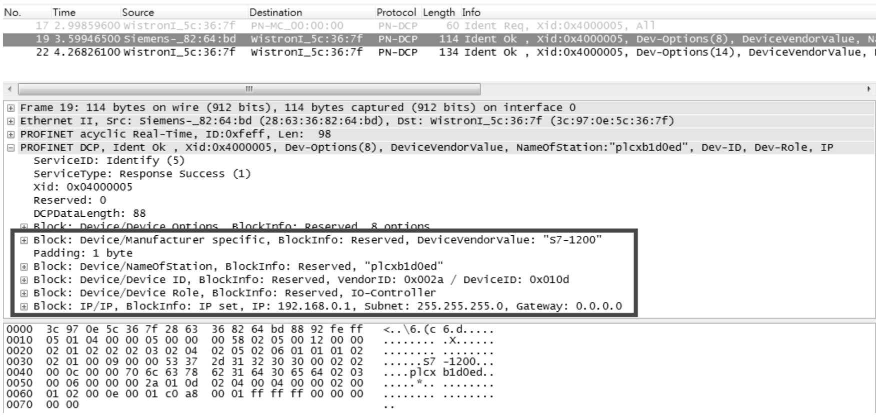
Figure 5. Attack entry point to the elevator system.

tive manner. In the passive collection mode, the personal computer and programmable logic controller broadcast their information to the network periodically using LLDP, which enables the adversary to capture and analyze LLDP packets to identify the devices. In the active collection mode, the adversary uses the PROFINET Discovery and Basic Configuration Protocol (PN-DCP) to query and determine the existence of the personal computer and programmable logic controller [16],

The proof-of-concept program developed in this research sends PN-DCP messages to the network devices and captures and analyzes the response packets to identify the devices of interest. The model information and IP addresses of the devices were obtained by sending simple PN-DCP broadcast packets. Figure 6 shows the response packets from the programmable logic controller that were captured by Wireshark.

4.4 False Command Injection Attack

After the programmable logic controller was discovered, the proof-of-concept program attempted to modify the behavior of the elevator system by sending a write command to the PA entry of the programmable logic controller. After the input value was changed, the programmable logic controller responded to the input according to its ladder logic program. The injected input signal requested the elevator to go to a different floor; this injected signal was the same as the signal that would be sent upon pressing the car call button on the control panel. Figure 7 shows how the proof-of-concept program used the S7 base protocol

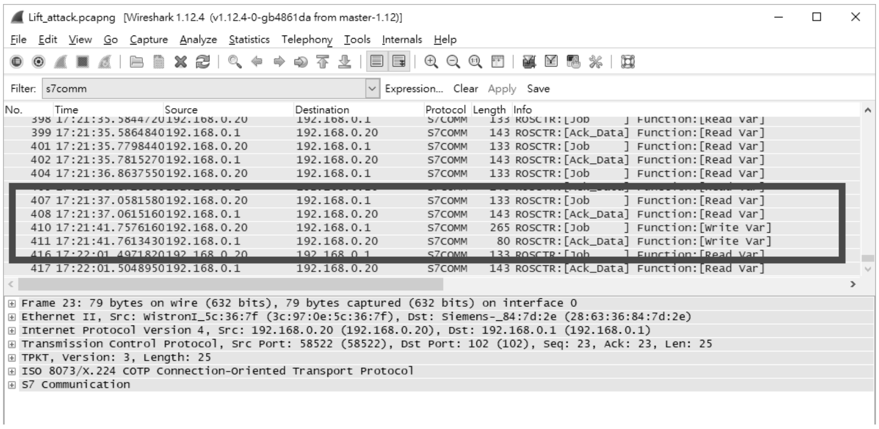


```
No.   Time        Source               Destination         Protocol  Length  Info
   17 2.99859600 WistronI_5c:36:7f   PN-MC_00:00:00      PN-DCP     60 Ident Req, Xid:0x4000005, All
   19 3.59946500 Siemens-_82:64:bd   WistronI_5c:36:7f   PN-DCP    114 Ident Ok , Xid:0x4000005, Dev-Options(8), DeviceVendorValue, N
   22 4.26826100 WistronI_5c:36:7f   WistronI_5c:36:7f   PN-DCP    134 Ident Ok , Xid:0x4000005, Dev-Options(14), DeviceVendorValue, I

⊞ Frame 19: 114 bytes on wire (912 bits), 114 bytes captured (912 bits) on interface 0
⊞ Ethernet II, Src: Siemens-_82:64:bd (28:63:36:82:64:bd), Dst: WistronI_5c:36:7f (3c:97:0e:5c:36:7f)
⊞ PROFINET acyclic Real-Time, ID:0xfeff, Len:  98
⊟ PROFINET DCP, Ident Ok , Xid:0x4000005, Dev-Options(8), DeviceVendorValue, NameOfStation:"plcxb1d0ed", Dev-ID, Dev-Role, IP
    ServiceID: Identify (5)
    ServiceType: Response Success (1)
    Xid: 0x04000005
    Reserved: 0
    DCPDataLength: 88
  ⊞ Block: Device/Device Options, BlockInfo: Reserved, 8 options
  ⊞ Block: Device/Manufacturer specific, BlockInfo: Reserved, DeviceVendorValue: "S7-1200"
    Padding: 1 byte
  ⊞ Block: Device/NameOfStation, BlockInfo: Reserved, "plcxb1d0ed"
  ⊞ Block: Device/Device ID, BlockInfo: Reserved, VendorID: 0x002a / DeviceID: 0x010d
  ⊞ Block: Device/Device Role, BlockInfo: Reserved, IO-Controller
  ⊞ Block: IP/IP, BlockInfo: IP set, IP: 192.168.0.1, Subnet: 255.255.255.0, Gateway: 0.0.0.0

0000  3c 97 0e 5c 36 7f 28 63  36 82 64 bd 88 92 fe ff   <..\6.(c 6.d.....
0010  05 01 04 00 00 05 00 00  00 58 02 05 00 12 00 00   ........ .X......
0020  02 01 02 02 02 03 02 04  02 05 02 06 01 01 01 02   ........ ........
0030  02 01 00 09 00 00 53 37  2d 31 32 30 30 00 02 02   ......S7 -1200...
0040  00 0c 00 00 70 6c 63 78  62 31 64 30 65 64 02 03   ....plcx b1d0ed..
0050  00 06 00 00 00 2a 01 0d  02 04 00 04 00 00 02 00   .....*.. ........
0060  01 02 00 0e 00 01 c0 a8  00 01 ff ff ff 00 00 00   ........ ........
0070  00 00                                              ..
```

Figure 6. Discovering programmable logic controller information via PN-DCP.



Lift_attack.pcapng [Wireshark 1.12.4 (v1.12.4-0-gb4861da from master-1.12)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: s7comm
Expression... Clear Apply Save

```
No.  Time              Source        Destination   Protocol Length Info
  398 17:21:35.5844720192.168.0.20   192.168.0.1    S7COMM  133 ROSCTR:[Job     ] Function:[Read Var]
  399 17:21:35.5864840192.168.0.1    192.168.0.20   S7COMM  143 ROSCTR:[Ack_Data] Function:[Read Var]
  401 17:21:35.7798440192.168.0.20   192.168.0.1    S7COMM  133 ROSCTR:[Job     ] Function:[Read Var]
  402 17:21:35.7815270192.168.0.1    192.168.0.20   S7COMM  143 ROSCTR:[Ack_Data] Function:[Read Var]
  404 17:21:36.8637550192.168.0.20   192.168.0.1    S7COMM  133 ROSCTR:[Job     ] Function:[Read Var]
  407 17:21:37.0581580192.168.0.20   192.168.0.1    S7COMM  133 ROSCTR:[Job     ] Function:[Read Var]
  408 17:21:37.0615160192.168.0.1    192.168.0.20   S7COMM  143 ROSCTR:[Ack_Data] Function:[Read Var]
  410 17:21:41.7576160192.168.0.20   192.168.0.1    S7COMM  265 ROSCTR:[Job     ] Function:[Write Var]
  411 17:21:41.7613430192.168.0.1    192.168.0.20   S7COMM   80 ROSCTR:[Ack_Data] Function:[Write Var]
  416 17:22:01.4971820192.168.0.20   192.168.0.1    S7COMM  133 ROSCTR:[Job     ] Function:[Read Var]
  417 17:22:01.5048950192.168.0.1    192.168.0.20   S7COMM  143 ROSCTR:[Ack_Data] Function:[Read Var]

⊞ Frame 23: 79 bytes on wire (632 bits), 79 bytes captured (632 bits) on interface 0
⊞ Ethernet II, Src: WistronI_5c:36:7f (3c:97:0e:5c:36:7f), Dst: Siemens-_84:7d:2e (28:63:36:84:7d:2e)
⊞ Internet Protocol Version 4, Src: 192.168.0.20 (192.168.0.20), Dst: 192.168.0.1 (192.168.0.1)
⊞ Transmission Control Protocol, Src Port: 58522 (58522), Dst Port: 102 (102), Seq: 23, Ack: 23, Len: 25
⊞ TPKT, Version: 3, Length: 25
⊞ ISO 8073/X.224 COTP Connection-Oriented Transport Protocol
⊞ S7 Communication
```

Figure 7. Using the S7 base protocol to read and write data.

read-variable and write-variable commands to query and manipulate the status of the programmable logic controller.

4.5 Control Signal Injection Attack

The false input command injection attack changed the behavior of the elevator system by sending a write command to the PE entry of the programmable logic controller. An output (control) signal injection attack is more harmful. This attack can change the power output directly, which means that the programmable logic controller would behave differently from the manner specified by its ladder logic program. In the case of the elevator, the false command in-

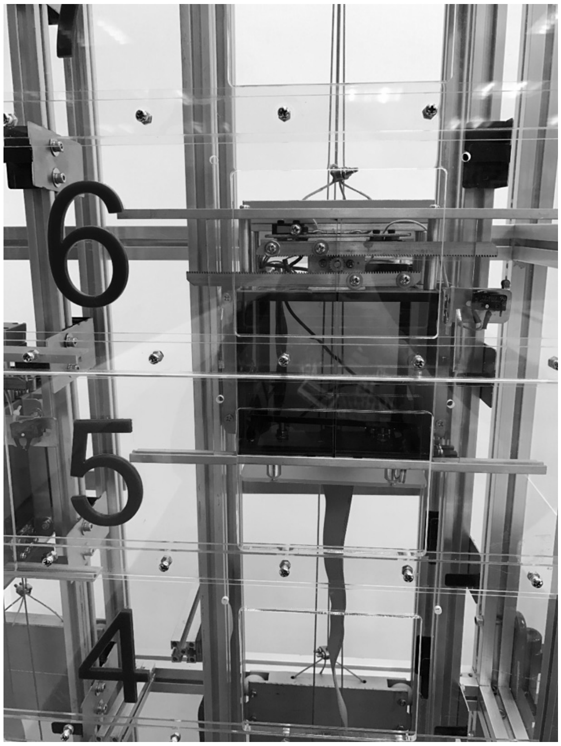


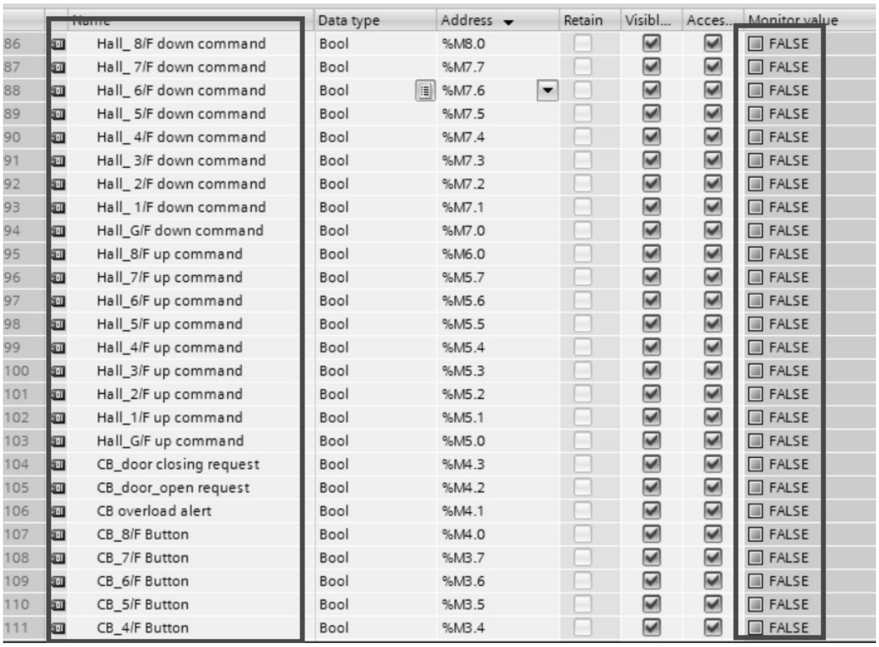
Figure 8. Stopping the elevator car between two floors.

jection attack merely moved the elevator in the same way as when the control panel is used. However, the output control signal injection attack can force the elevator to stop between two floors. Figure 8 shows the result – the elevator car stopped between the fifth and sixth floors after the false control signal was sent to the programmable logic controller.

4.6 Control Variable Injection Attack

A ladder logic program contains a number of variables called programmable logic controller tags that control or perform various operations. An adversary who is able to control the personal computer with the TIA Portal installed would be able to obtain the running ladder logic (including the variables and their addresses) using the Download from PLC function.

In the case of the model elevator system, after the downloaded ladder logic program was modified and uploaded to the programmable logic controller, it was possible to fully control the elevator, including making it function improperly. In the experiments, the buzzer variable was changed from false to true,



	Name	Data type	Address	Retain	Visibl...	Acces...	Monitor value
86	Hall_ 8/F down command	Bool	%M8.0	☐	✓	✓	FALSE
87	Hall_ 7/F down command	Bool	%M7.7	☐	✓	✓	FALSE
88	Hall_ 6/F down command	Bool	%M7.6	☐	✓	✓	FALSE
89	Hall_ 5/F down command	Bool	%M7.5	☐	✓	✓	FALSE
90	Hall_ 4/F down command	Bool	%M7.4	☐	✓	✓	FALSE
91	Hall_ 3/F down command	Bool	%M7.3	☐	✓	✓	FALSE
92	Hall_ 2/F down command	Bool	%M7.2	☐	✓	✓	FALSE
93	Hall_ 1/F down command	Bool	%M7.1	☐	✓	✓	FALSE
94	Hall_G/F down command	Bool	%M7.0	☐	✓	✓	FALSE
95	Hall_8/F up command	Bool	%M6.0	☐	✓	✓	FALSE
96	Hall_7/F up command	Bool	%M5.7	☐	✓	✓	FALSE
97	Hall_6/F up command	Bool	%M5.6	☐	✓	✓	FALSE
98	Hall_5/F up command	Bool	%M5.5	☐	✓	✓	FALSE
99	Hall_4/F up command	Bool	%M5.4	☐	✓	✓	FALSE
100	Hall_3/F up command	Bool	%M5.3	☐	✓	✓	FALSE
101	Hall_2/F up command	Bool	%M5.2	☐	✓	✓	FALSE
102	Hall_1/F up command	Bool	%M5.1	☐	✓	✓	FALSE
103	Hall_G/F up command	Bool	%M5.0	☐	✓	✓	FALSE
104	CB_door closing request	Bool	%M4.3	☐	✓	✓	FALSE
105	CB_door_open request	Bool	%M4.2	☐	✓	✓	FALSE
106	CB overload alert	Bool	%M4.1	☐	✓	✓	FALSE
107	CB_8/F Button	Bool	%M4.0	☐	✓	✓	FALSE
108	CB_7/F Button	Bool	%M3.7	☐	✓	✓	FALSE
109	CB_6/F Button	Bool	%M3.6	☐	✓	✓	FALSE
110	CB_5/F Button	Bool	%M3.5	☐	✓	✓	FALSE
111	CB_4/F Button	Bool	%M3.4	☐	✓	✓	FALSE

Figure 9. Variables used to store commands.

which caused the buzzer to sound forever. The current position of the elevator was also changed while it was moving, which caused the elevator to move to the wrong floor. Note that these variables cannot be configured as read-only as a security measure because the human-machine interface must be able to change the values of variables during normal operations. Figure 9 shows some of the programmable logic controller variables in the TIA Portal that are used to command the elevator. Figure 10 shows the current and desired positions of the elevator while it was moving.

4.7 Sensor Value Response Modification Attack

Sensor values are stored as programmable logic controller tags. These values can be changed to negatively impact the elevator logic. Since the communications between the sensor and programmable logic controller are neither encrypted nor authenticated, the ladder logic program of the programmable logic controller would then execute based on the changed sensor values. In the experiments, the door light sensor value was changed, which prevented the elevator control system from detecting that something was jammed between the doors and the elevator kept trying to close the doors. Figure 11 shows the sensor variables in the TIA Portal that may be modified in sensor value response modification attacks.

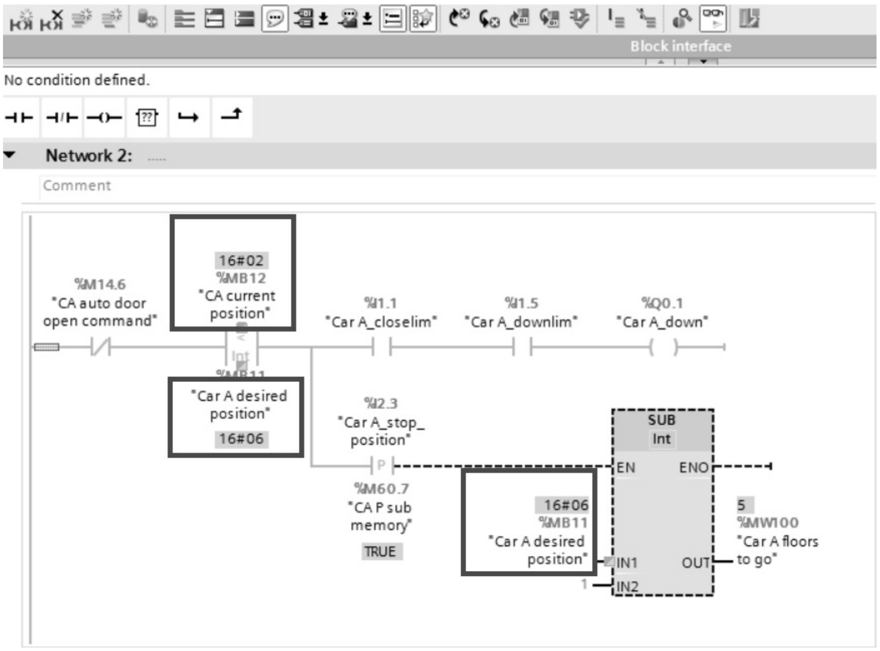
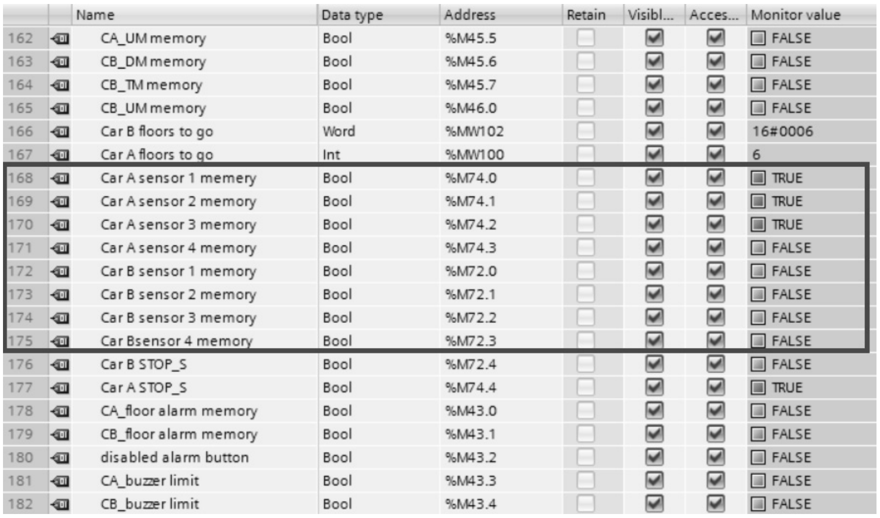


Figure 10. Variables used to store the current and desired elevator positions.



		Name	Data type	Address	Retain	Visibl...	Acces...	Monitor value
162		CA_UM memory	Bool	%M45.5		✓	✓	FALSE
163		CB_DM memory	Bool	%M45.6		✓	✓	FALSE
164		CB_TM memory	Bool	%M45.7		✓	✓	FALSE
165		CB_UM memory	Bool	%M46.0		✓	✓	FALSE
166		Car B floors to go	Word	%MW102		✓	✓	16#0006
167		Car A floors to go	Int	%MW100		✓	✓	6
168		Car A sensor 1 memery	Bool	%M74.0		✓	✓	TRUE
169		Car A sensor 2 memery	Bool	%M74.1		✓	✓	TRUE
170		Car A sensor 3 memery	Bool	%M74.2		✓	✓	TRUE
171		Car A sensor 4 memory	Bool	%M74.3		✓	✓	FALSE
172		Car B sensor 1 memory	Bool	%M72.0		✓	✓	FALSE
173		Car B sensor 2 memory	Bool	%M72.1		✓	✓	FALSE
174		Car B sensor 3 memory	Bool	%M72.2		✓	✓	FALSE
175		Car Bsensor 4 memory	Bool	%M72.3		✓	✓	FALSE
176		Car B STOP_S	Bool	%M72.4		✓	✓	FALSE
177		Car A STOP_S	Bool	%M74.4		✓	✓	TRUE
178		CA_floor alarm memory	Bool	%M43.0		✓	✓	FALSE
179		CB_floor alarm memory	Bool	%M43.1		✓	✓	FALSE
180		disabled alarm button	Bool	%M43.2		✓	✓	FALSE
181		CA_buzzer limit	Bool	%M43.3		✓	✓	FALSE
182		CB_buzzer limit	Bool	%M43.4		✓	✓	FALSE

Figure 11. Sensor variables in the TIA Portal.

4.8 Discussion and Recommendations

Current programmable logic controller security mechanisms are inadequate for combating malicious attacks. All workstations and personal computers installed with the Siemens TIA Portal and located in an internal control network are attractive targets for attacks. As shown in Figure 1, an adversary can discover these computing platforms by capturing and analyzing LLDP packets.

Installing sensors throughout an industrial control network provides situational awareness about attacks and anomalies, but sophisticated adversaries can tamper with the sensor values and send false commands and signals to programmable logic controllers. Because of the complexity of industrial control systems and their underlying physical processes, network intrusion detection and prevention systems have obvious limitations. A promising solution is to encrypt and authenticate all communications involving human-machine interfaces, programmable logic controllers and engineering/development workstations. This is especially important because future industrial control system attacks will go beyond utilizing an engineering workstation to download malicious ladder logic programs and seek to control and attack the programmable logic controllers directly.

Programmable logic controller protection mechanisms also must be enhanced because adversaries can exploit the S7 base protocol and bypass the security configurations to execute attacks. The S7 base protocol should be locked down in modern versions of Siemens programmable logic controllers. In the case of the Siemens TIA Portal, the PE and PA entries in programmable logic controllers should be configured to provide adequate security.

Figure 12 shows that malware can easily discover the development personal computer by capturing and analyzing LLDP packets. Having discovered the personal computer, the adversary could attempt to gain access to it. Firewalls or other network filtering mechanisms must be deployed to block these protocol packets so that it is more difficult to discover devices in an industrial control network.

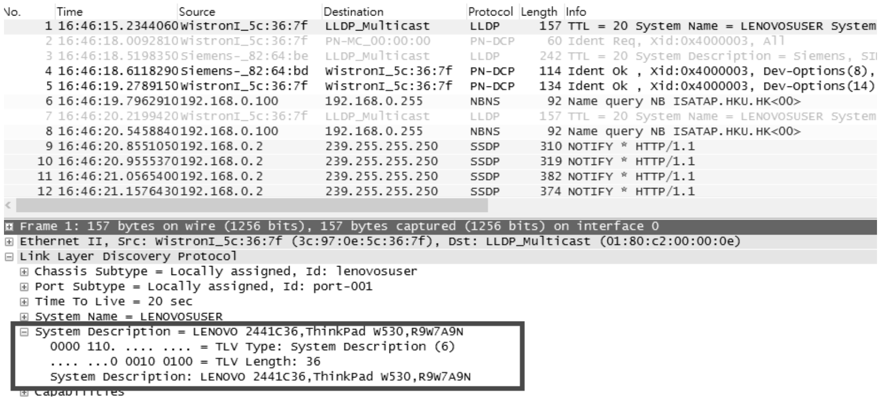
The following recommendations are made to secure the internal network of an industrial control system:

- The development personal computer should be well protected. External device and Internet connections should be disabled to prevent malware attacks.

- Firewalls should be deployed to ensure that discovery and control commands only come from authorized devices.

- Multiple programmable logic controllers from different vendors should be employed in order to survive attacks that target a specific programmable logic controller model or brand.

- Logging and heartbeat mechanisms should be incorporated in ladder logic programs to detect unauthorized modifications.



```
No.   Time             Source             Destination        Protocol Length Info
    1 16:46:15.2344060 WistronI_5c:36:7f  LLDP_Multicast     LLDP     157 TTL = 20 System Name = LENOVOSUSER System
    2 16:46:18.0092810 WistronI_5c:36:7f  PN-MC_00:00:00     PN-DCP    60 Ident Req, Xid:0x4000003, All
    3 16:46:18.5198350 Siemens-_82:64:be  LLDP_Multicast     LLDP     242 TTL = 20 System Description = Siemens, SI
    4 16:46:18.6118290 Siemens-_82:64:bd  WistronI_5c:36:7f  PN-DCP   114 Ident Ok , Xid:0x4000003, Dev-Options(8),
    5 16:46:19.2789150 WistronI_5c:36:7f  WistronI_5c:36:7f  PN-DCP   134 Ident Ok , Xid:0x4000003, Dev-Options(14)
    6 16:46:19.7962910 192.168.0.100      192.168.0.255      NBNS      92 Name query NB ISATAP.HKU.HK<00>
    7 16:46:20.2199420 WistronI_5c:36:7f  LLDP_Multicast     LLDP     157 TTL = 20 System Name = LENOVOSUSER System
    8 16:46:20.5458840 192.168.0.100      192.168.0.255      NBNS      92 Name query NB ISATAP.HKU.HK<00>
    9 16:46:20.8551050 192.168.0.2        239.255.255.250    SSDP     310 NOTIFY * HTTP/1.1
   10 16:46:20.9555370 192.168.0.2        239.255.255.250    SSDP     319 NOTIFY * HTTP/1.1
   11 16:46:21.0565400 192.168.0.2        239.255.255.250    SSDP     382 NOTIFY * HTTP/1.1
   12 16:46:21.1576430 192.168.0.2        239.255.255.250    SSDP     374 NOTIFY * HTTP/1.1

⊞ Frame 1: 157 bytes on wire (1256 bits), 157 bytes captured (1256 bits) on interface 0
⊞ Ethernet II, Src: WistronI_5c:36:7f (3c:97:0e:5c:36:7f), Dst: LLDP_Multicast (01:80:c2:00:00:0e)
⊟ Link Layer Discovery Protocol
  ⊞ Chassis Subtype = Locally assigned, Id: lenovosuser
  ⊞ Port Subtype = Locally assigned, Id: port-001
  ⊞ Time To Live = 20 sec
  ⊞ System Name = LENOVOSUSER
  ⊟ System Description = LENOVO 2441C36,ThinkPad W530,R9W7A9N
      0000 110. .... .... = TLV Type: System Description (6)
      .... ...0 0010 0100 = TLV Length: 36
      System Description: LENOVO 2441C36,ThinkPad W530,R9W7A9N
  ⊞ Capabilities
```

Figure 12. Using the Siemens TIA Portal to obtain information via LLDP.

- Industrial control engineers must be cognizant of programmable logic controller vulnerabilities and potential attacks.

5. Conclusions

The vast majority of research in industrial control system security has focused on network-level intrusion detection and protection mechanisms. In contrast, this research has specifically considered the threats to the internal networks of industrial control systems, which include the connections from the computer platforms that manage programmable logic controllers. The threat model assumes that an adversary can gain access to the internal network of an industrial control system by various means and specifies several attacks on the confidentiality, integrity and availability of programmable logic controllers. Experiments involving a model elevator system with a Siemens programmable logic controller demonstrate the potential attacks and their impacts, and provide guidance for implementing security solutions that can mitigate the attacks.

Future research will concentrate on a larger testbed with programmable logic controllers from different vendors. Additionally, efforts will be directed at developing lightweight security solutions for programmable logic controllers that satisfy the real-time constraints of production industrial control systems.

References

[1] S. Abe, M. Fujimoto, S. Horata, Y. Uchida and T. Mitsunaga, Security threats of Internet-reachable ICSs, *Proceedings of the Fifty-Fifth Annual Conference of the Society of Instrument and Control Engineers of Japan*, pp. 750–755, 2016.

[2] D. Beresford, Exploiting Siemens Simatic S7 PLCs, presented at *Black Hat USA*, 2011.

[3] A. Cardenas, T. Roosta and S. Sastry, Rethinking security properties, threat models and the design space in sensor networks: A case study in SCADA systems, *Ad Hoc Networks*, vol. 7(8), pp. 1434–1447, 2009.

[4] F. Cohen, A reference architecture approach to ICS security, *Proceedings of the Fourth International Symposium on Resilient Control Systems*, pp. 21–25, 2011.

[5] B. Ghena, W. Beyer, A. Hillaker, J. Pevarnek and J. Haldeman, Green lights forever: Analyzing the security of traffic infrastructure, *Proceedings of the Eighth USENIX Workshop on Offensive Technologies*, 2014.

[6] D. Hadziosmanovic, D. Bolzoni, S. Etalle and P. Hartel, Challenges and opportunities in securing industrial control systems, *Proceedings of the Workshop on Complexity in Engineering*, 2012.

[7] P. Jie and L. Li, Industrial control system security, *Proceedings of the International Conference on Intelligent Human-Machine Systems and Cybernetics*, vol. 2, pp. 156–158, 2011.

[8] E. Korkmaz, A. Dolgikh, M. Davis and V. Skormin, ICS security testbed with delay attack case study, *Proceedings of the IEEE Military Communications Conference*, pp. 283–288, 2016.

[9] M. Krotofil and D. Gollmann, Industrial control systems security: What is happening? *Proceedings of the Eleventh IEEE International Conference on Industrial Informatics*, pp. 670–675, 2013.

[10] J. Malchow, D. Marzin, J. Klick, R. Kovacs and V. Roth, PLC Guard: A practical defense against attacks on cyber-physical systems, *Proceedings of the IEEE Conference on Communications and Network Security*, pp. 326–334, 2015.

[11] S. McLaughlin and S. Zonouz, Controller-aware false data injection against programmable logic controllers, *Proceedings of the IEEE International Conference on Smart Grid Communications*, pp. 848–853, 2014.

[12] T. Miyachi and T. Yamada, Current issues and challenges on cyber security for industrial automation and control systems, *Proceedings of the SICE Annual Conference*, pp. 821–826, 2014.

[13] Y. Mo and B. Sinopoli, False data injection attacks in control systems, presented at the *First Workshop on Secure Control Systems*, 2010.

[14] R. Piggin, Emerging good practice for cyber security of industrial control systems and SCADA, *Proceedings of the Seventh IET International Conference on System Safety*, 2012.

[15] T. Spyridopoulos, T. Tryfonas and J. May, Incident analysis and digital forensics in SCADA and industrial control systems, *Proceedings of the Eighth IET International System Safety Conference*, 2013.

[16] A. Timorin, SCADA deep inside: Protocols and security mechanisms, presented at the *Balkan Computer Congress*, 2014.

[17] D. Wei and K. Ji, Resilient industrial control system (RICS): Concepts, formulation, metrics and insights, *Proceedings of the Third International Symposium on Resilient Control Systems*, pp. 15–22, 2010.

Chapter 11

GENERATING HONEYPOT TRAFFIC FOR INDUSTRIAL CONTROL SYSTEMS

Htein Lin, Stephen Dunlap, Mason Rice and Barry Mullins

Abstract Defending critical infrastructure assets is an important, but extremely difficult and expensive task. Historically, decoys have been used very effectively to distract attackers and, in some cases, convince attackers to reveal their attack strategies. Several researchers have proposed the use of honeypots to protect programmable logic controllers, specifically those used in the critical infrastructure. However, most of these honeypots are static systems that wait for would-be attackers. To be effective, honeypot decoys need to be as realistic as possible. This chapter introduces a proof-of-concept honeypot network traffic generator that mimics a genuine control system in operation. Experiments conducted using a Siemens APOGEE building automation system for single and dual subnet instantiations indicate that the proposed traffic generator supports honeypot integration, traffic matching and routing in a decoy building automation network.

Keywords: Honeypots, network traffic generation, industrial control systems

1. Introduction

The United States Ghost Army conducted deception operations in France, Belgium, Luxembourg and Germany during World War II [1]. The mission was to deceive the enemy and lure German units away from Allied combat units. Engineers set up inflatable armored tanks, aircraft, airfields, tents and motor pools. Other tactics such as looping convoy traffic, deploying military police and posting general and staff officers in public places were used to draw Axis resources (e.g., intelligence assets and combat power) away from the real targets. The Ghost Army also played recordings of actual armor and infantry units over loudspeakers. Deceptive radio transmissions were broadcast on fabricated networks called "Spoof Radio." The Ghost Army leveraged the comprehensive deployment of decoys and deception techniques to overload enemy sensors and intelligence gathering capabilities.

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 193–223, 2017.
https://doi.org/10.1007/978-3-319-70395-4_11

Deception techniques and decoy technologies are employed in cyberspace in the form of honeypots. These systems may be simple (e.g., virtual machines) or complex (e.g., full-scale replicas of industrial control systems). Limited-scale industrial control system honeypots do not generate traffic that truly represent control systems. Thus, attackers who passively monitor the honeypots may be able to differentiate them from operational systems. Network traffic generators (NTGs) could increase realism and help deceive potential attackers, but they are geared for traditional information technology network performance testing as opposed to creating industrial control system decoys. This chapter introduces a proof-of-concept honeypot network traffic generator that can mimic genuine industrial control systems.

2. Background

Critical infrastructure systems have long been considered immune to network attacks that have plagued traditional information technology systems. Historically, process control and supervisory control and data acquisition (SCADA) systems have relied on proprietary hardware, software and isolation for security. However, the convergence of information technology and operational technology is pushing towards open standards based on Ethernet, TCP/IP and web-based technologies and protocols. According to Gartner [5], operational technology systems are becoming more like information technology systems, including in terms of their vulnerabilities.

The information technology/operational technology convergence introduces several security challenges. Operational technology systems often run software without updates for 15 to 20 years compared with the three to five year lifecycles of information technology systems [23]. Since the 1960s, SCADA system architecture trends show a drastic decline in the use of proprietary hardware from 60% to 2% and software from 100% to 30% [17]. As a result, security practices (e.g., security through obscurity) used in older-generation operational technology systems are no longer applicable to the newer systems [5].

Information technology systems are capable of handling multiple functions and support the addition of third-party security applications. In contrast, operational technology systems are designed to support specific industrial processes and have resource constraints. Adding resources or features to these systems may not be possible because they often lack the memory and/or computing resources to support the enhancements. Implementing traditional information technology security solutions in industrial control systems may also cause timing disruptions and may negatively impact performance and availability [23].

2.1 Control System Threats

Trend Micro [25] has published a study covering attacks on external-facing industrial control devices and honeypot technologies developed to identify threat actors and their motivations behind attacks. The study highlights five activities conducted by attackers: (i) reconnaissance using free and open-source tools (e.g., ShodanHQ); (ii) port scanning of an intended target with an IP address and surrounding subnets; (iii) fingerprinting of devices for operating system and other identifiable information; (iv) persistence and lateral movement; and (v) data exfiltration.

A report by Idaho National Laboratory [8] highlights the security challenges associated with information technology networks that are applicable to operational technology systems as a result of convergence. Critical cyber security issues that need to be addressed in operational technology systems include: (i) backdoors and holes in network perimeters; (ii) protocol vulnerabilities; (iii) attacks on field devices; (iv) database attacks; and (v) communications hijacking and man-in-the-middle attacks. The report provides guidance for developing defense-in-depth strategies as best practices for control systems in a multi-tier information architecture. In building defense-in-depth for operational technology systems, time-sensitive requirements (e.g., clock cycles of programmable logic controllers) may make proven information technology security technologies inappropriate for control systems. Another recommendation is to use intrusion detection systems that passively analyze traffic and network activity without impacting operations. The systems compare the observed data against predefined rule sets and attack signatures. While contemporary intrusion signatures work for a wide range of attacks, they are inadequate for control networks because they render intrusion detection systems blind to attacks on industrial control systems.

2.2 Honeypots

Honeypots can help mitigate the control system threats discussed above. A honeypot is a decoy-based intrusion detection technology that attracts attackers and supports the analysis of adversary actions and behaviors [4, 11]. Honeypots can be: (i) low-interaction systems; or (ii) high-interaction systems. Low-interaction honeypots emulate services and operating systems, provide limited activity and are generally easy to deploy. High-interaction honeypots use real operating systems, applications and hardware to provide more realistic environments, but are far more difficult to deploy [7].

Hybrid honeypots can be effective at protecting industrial control systems. Winn et al. [26] have shown that honeypots can leverage proxy technology with a programmable logic controller to provide multiple instances of control devices. They also describe an approach for creating low-cost control system honeypots that are authentic and targetable by using data from a programmable logic controller while maintaining authenticity via unique network identities (e.g., IP and media access control (MAC) addresses) that match the corresponding honeypot devices. Warner [24] has proposed a framework that automatically configures emulation behavior by building protocol trees from networked programmable logic controller traces (i.e., captured network data). Girtz et al. [6] have extended Warner's work by forwarding unknown requests to a programmable logic controller proxy to provide responses and updates to a protocol tree. Their research has bolstered the targetability and authenticity of industrial control system honeypots and has demonstrated the ability to emulate the characteristics and interactions of programmable logic controllers.

A successful implementation of honeypot technology can enhance cyber security by incorporating defense-in-depth measures to mitigate vulnerabilities. Complementing conventional security technologies (e.g., firewalls and intrusion detection systems) with honeypots can support early intrusion detection and threat intelligence collection against unknown vectors while providing defenders with valuable knowledge and time to address security concerns [21]. However, by design, honeypots do not manifest authorized activities and are not meant for operational use. As such, any activity in these systems can be considered suspicious [11]. A honeypot functions as a litmus test to detect unauthorized access. The downside to the approach is that honeypots do not actively engage in autonomous network communications. Instead, they rely on interactions with an attacker to generate network activity. This poses a problem because real operational technology networks have non-stop, recurring traffic flows. If an attacker were to target a honeypot, the absence of operational technology network traffic would indicate that it is not part of a real industrial control system. As a result, the honeypot would no longer entice the attacker, who would instead seek a real attack target. This reduces the effectiveness of a honeypot as a security mechanism.

2.3 Network Traffic Generation

An understanding of network architecture is necessary because network traffic visibility depends on the level of compromise. In a switched network, an attacker may only be able to map the network using broadcast messages. The segregation of traffic in a switched network would normally prohibit an attacker from observing control system traffic. Traffic collected would be restricted to the packets originating from or destined to a compromised host.

In the case of a skilled attacker, network device exploitation can offer different vantage points that enable more traffic to be visible. Using Figure 1 as an example, compromising the switch in Subnet 2 could reveal all the traffic in the subnet. Layer 3 traffic can be seen when compromising the router. Exploiting the Subnet 1 switch or the human-machine interface (HMI) would reveal traffic from all the control devices that communicate with the human-machine interface. An attacker may be able to isolate active systems from non-active systems (e.g., honeypots) by passively monitoring the traffic after compromising key nodes in the network.

It is important for an attacker to study network activity because it can increase the odds of a successful attack. If the attacker were to focus resources on the wrong target (e.g., honeypot), the cost of revealing attack information would be detrimental. As such, vigorous network discovery, reconnaissance and network enumeration actions would most likely occur prior to an attack, especially when a zero-day exploit is used. At the system level, an attacker could collect traffic data going to and from a compromised host. This would reveal the identity and function of the machine. At the network level, a compromised

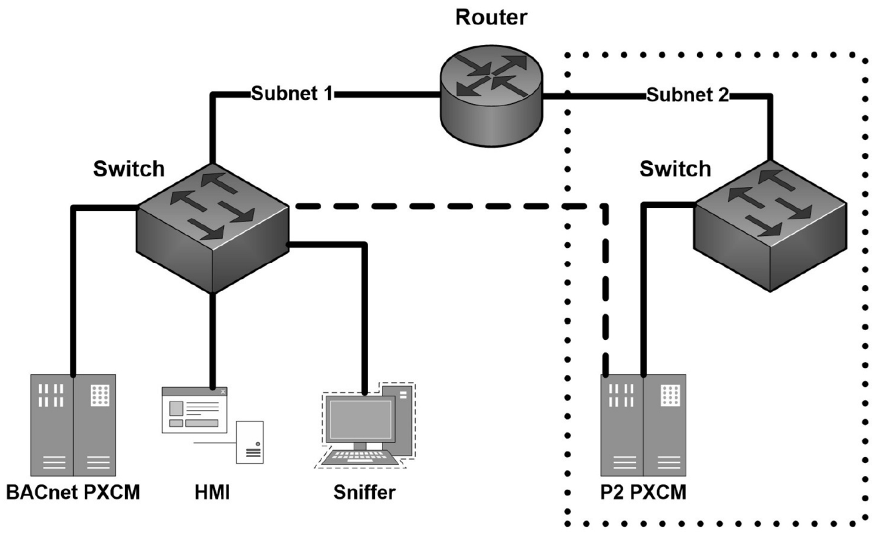


Figure 1. APOGEE platform dual subnet network design.

device provides traffic data from connected systems. With this data, an attacker could identify control devices by analyzing traffic patterns and packet content. This would also identify false targets (i.e., honeypots) due to the absence of traffic originating from the devices.

Honeypots designed for industrial control systems should have the same data traffic and patterns as the systems they are designed to emulate. Full network traffic generation would render the honeypots more effective as decoys. This chapter discusses the use of network traffic generators in conjunction with low-interaction and hybrid honeypots.

The criteria in Table 1 – comprising traffic matching, honeypot integration, network routing and scalability – are specified for assessing viable network traffic generators for industrial control system honetpots.

2.4 Network Traffic Generators

Several commercial off-the-shelf and open-source network testing products were selected as candidates for honeypot network traffic generation. They were evaluated by analyzing product literature reviews against the criteria listed in Table 1.

Commercial Network Traffic Generators. The commercial products considered included: (i) SolarWinds WAN Killer; (ii) NetLoad Stateful Traffic Mix Tester Solution; and (iii) Ixia IxLoad.

Table 1. Honeypot network traffic generator evaluation criteria.

Traffic Matching	**Content Matching:** Generated traffic should match the packet data of the control device that the honeypot is emulating. **Extraneous Packets:** Packets that do not match control system packets must be avoided during generation (e.g., generator control and traffic synchronization). **Packet Ordering:** Generated traffic should not have packets that are out of sequence. **Timing Consistency:** Generated traffic should replicate trace timing as accurately as possible.
Honeypot Integration	**Honeypot Pairing:** Generated traffic must be sent to and from the corresponding honeypot. **Honeypot Header Matching:** IP and MAC addresses in the generated packet headers should reflect the corresponding honeypot systems.
Network Routing	**Distributed Operation:** Generated traffic should originate from multiple points in a network. **Layer 2 Addressing:** Generated traffic that has the same characteristics as the associated honeypots must not cause network addressing problems (e.g., MAC table conflicts). **Layer 3 Routing:** Generated network traffic must be routable in multi-subnet environments.
Scalability	**Cost:** Industrial control devices are distributed physically and logically in a network, which requires a large number of honeypot instances. The high cost of each network traffic generator instance makes it challenging to replicate a complete control system. **Flexibility:** Industrial control systems have diverse components that are distributed across geographic locations. In order to accurately generate control system traffic, network traffic generator instances may have to be installed in multiple locations. The network traffic generators must be configurable to accommodate a large variety of industrial control system applications.

The SolarWinds WAN Killer is one of more than 60 network management tools included in the Engineer's Toolset [22]. WAN Killer enables network administrators to generate random traffic in a wide-area network. The tool can manipulate packet size, circuit bandwidth and bandwidth utilization with randomly-generated data. The tool simulates network traffic primarily for load testing and does not generate traffic corresponding to industrial control system protocols. Due to its inability to generate control network traffic, WAN Killer does not meet the traffic matching criteria.

The NetLoad Stateful Traffic Mix Tester Solution provides off-the-shelf network processing hardware and software as a single package [12, 14]. It supports network testing by generating TCP and user datagram protocol (UDP) network traffic and providing a packet capture (PCAP) file replay feature. The PCAP replay feature generates packets using PCAP data. It can use packet timestamps to mimic the timing of the original PCAP packets. NetLoad reports an inter-packet timing (IPT) of less than one microsecond [13]. The software identifies bidirectional traffic when using PCAP replay and allows directed traffic from two ports. Other features enable the PCAP replay load to be distributed among the four ports. While the appliance can replay captured industrial control network traffic, it was designed for network testing and, therefore, does not implement honeypot integration features. The documentation does not indicate a way to load custom honeypot software on the appliance nor does it feature a way to modify PCAP data to match honeypot characteristics.

Ixia IxLoad is a suite of software and hardware that supports network performance testing [10]. The software can be used in a virtual environment loaded on a server or used in conjunction with proprietary Ixia products. IxLoad delivers a variety of stateful information technology protocols for emulating web, video, voice, storage, virtual private network, wireless, infrastructure and encapsulation/security protocols. For unsupported and propriety protocols, IxLoad provides TCP/UDP replay traffic options through its Application Replay feature [9]. This feature replays network packet captures and can create bidirectional traffic flows through unique ports. Of the three commercial solutions evaluated, IxLoad best meets the network traffic generation requirements based on product literature and vendor contacts. However, IxLoad was not designed for honeypot network traffic generation. Multiple licenses may be required for implementations in large networks, making the solutions cost prohibitive. In any case, further evaluation is needed to determine if IxLoad meets all the criteria for honeypot network traffic generation.

Open-Source Network Traffic Generators. Three open-source traffic generators were considered in this research: (i) Ostinato; (ii) Distributed Internet Traffic Generator; and (iii) Tcpreplay.

Ostinato is a network packet crafter, traffic generator and analyzer with a graphical user interface (GUI) and a Python application programming interface (API). The software operates in a controller-agent architecture, where the controller performs command and control (C2) operations and the agents generate network traffic [15]. The software generates traffic in the network using crafted packets or by replaying data from PCAP files. For the controller-agent architecture to function, Ostinato transmits trace data and device configurations from the controller to the agents during initialization. Timing limitations (i.e., packet transmission based on packets per second in burst modes) were observed during the testing. As a result, the generated traffic did not maintain the original trace timing. This limitation was identified using WireShark analysis tools. A final implementation of a honeypot network traffic generator using Ostinato would require modifications to how timing is handled by the software. Another limitation was found in the handling of command and control operations. These operations (e.g., synchronization and state status) are continuously performed by the controller and agents. The presence of these identifiable packets in a network would likely alert an attacker to the presence of an Ostinato network traffic generator. Ostinato did not meet the traffic matching criteria as a result of these limitations.

The Distributed Internet Traffic Generator (D-ITG) produces IP traffic by replicating the workload of Internet applications [3]. D-ITG generates traffic using stochastic models for packet size and inter-packet timing that mimic supported application-level protocol behavior. The packet size and inter-packet timing data can also be loaded from capture files. Network traffic generation is performed between ITGSend (sender component of the platform) and ITGRecv (receiver component). A command and control connection exists between the two components that controls the traffic generation process for each traffic flow (e.g., port assignments). In the case of large-scale distributed environments, multiple ITGSend instances can be remotely controlled by the D-ITG API. However, the platform does not offer a method for modifying header data to match honeypot characteristics. Traffic is generated one-way from the ITGSend to ITGRecv instances. Two-way traffic generation is implemented using multiple instances of ITGSend and ITGRecv on each pair of honeypots to perform conversational traffic flow. This can be an overwhelming task in a large-scale implementation. Additionally, application layer payload data is ignored and only packet size and inter-packet timing data from traces are used. As such, D-ITG does not meet the honeypot integration and traffic matching criteria.

The Tcpreplay suite of tools enables previously-captured traffic to be replayed through various network devices. Tcpreplay can generate trace data using recorded timestamps. It also supports bidirectional traffic flow through the use of two interfaces. The suite also includes a tool for modifying packet header information. Tcpreplay was selected as a candidate for the pilot study.

3. Test Environment

The test environment incorporated the Siemens APOGEE building automation system (BAS). The environment comprised the following components:

- Siemens Insight software that provides a human-machine interface and engineering workstation functionality.
- Ubuntu VM with a Honeyd honeypot and network traffic generator software (Tcpreplay and a custom network traffic generator).
- Ubiquiti EdgeRouter X router.
- Two Netgear ProSafe Plus switches.
- Two Siemens APOGEE PXC100 programmable controller modular systems with input/output modules.
- Field level network devices (e.g., sensors, lights and fans).

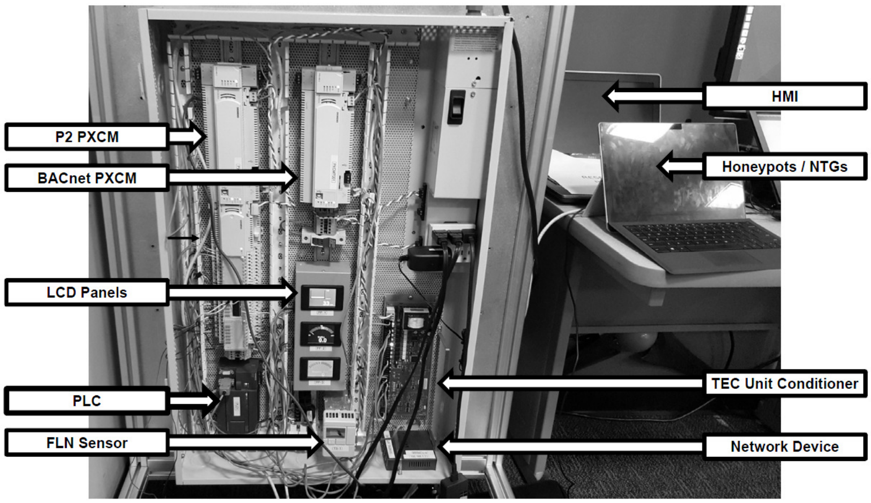


Figure 2. APOGEE platform.

3.1 Design Considerations

The environment was designed to replicate an APOGEE platform that provides building automation system functionality on a single field panel as shown in Figure 2. The two PXC100 programmable controller modular (PXCM) systems were mounted side-by-side with wiring from the input/output module expansions to the field level network (FLN) devices. Serial connections were made to a Siemens Simatic S7-200 PLC and Siemens 550-833 unit conditioner circuit board with the programmable controller modular systems. User feedback and interaction were provided through sensor liquid crystal display panels, physical lights, endpoint devices and the human-machine interface.

3.2 Network Topology

The APOGEE platform incorporates three networks: (i) management level network; (ii) automation level network; and (iii) field level network.

The management level network has servers and client workstations that provide management controls for the APOGEE automation system [19, 20]. The hardware systems at this level include servers and workstations running Siemens Insight or InfoCenter software suites, web accessible terminals, mobile devices and programmable controller modular systems with management level network functionality. Communications integration is provided by proprietary and standard protocols for building automation systems.

The automation level network provides field-panel-to-field-panel communications as well as communications between the management level network and

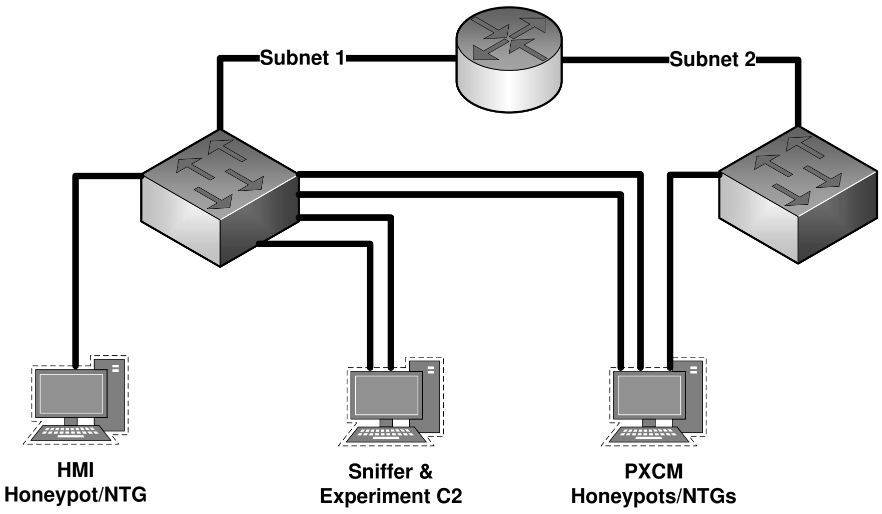


Figure 3. Test platform single and dual subnet network designs.

field level network. The hardware components comprise programmable controller modular system supervisory field panels. These panels can operate in the networked or standalone configurations and provide control, monitoring and energy management functions to field level network devices.

The field level network is the lowest level of the APOGEE building automation network. All the endpoint devices reside at this level and vary depending on the application (e.g., terminal equipment controllers and sensor units).

The network implementation for the test environment platform was based on the recommended Ethernet single subnet and multi-subnet configurations in the Siemens APOGEE technical specifications [18]. The two network topologies considered were: (i) single subnet; and (ii) dual subnet. Due to equipment availability, a dual subnet configuration was used to test the multi-subnet environment. One P2 programmable controller modular system (P2 is a Siemens proprietary protocol) was used in the single subnet and dual subnet configurations with minimal wiring and configuration changes (e.g., IP address reassignment).

Figure 3 shows the APOGEE platform network design. The design incorporates a dual subnet environment with two Siemens programmable controller modular systems: (i) P2 protocol over TCP/IP; and (ii) BACnet protocol over UDP. A network sniffer was added to a mirrored port in Subnet 1 for network capture and analysis functionality. The single subnet design represents a building automation infrastructure for a single building.

The multi-subnet design represents a multi-building infrastructure. The control networks were distributed with network connectivity provided by individual switches. In this setup, a central router connected the multiple buildings using switches to form the building automation network infrastructure.

Honeypots and network traffic generators were employed to replicate the APOGEE platform (see Figure 3). A honeypot and network traffic generator were used to replicate the human-machine interface in Subnet 1. A network sniffer/command and control workstation were incorporated in Subnet 1 to provide: (i) network capture and analysis functionality on one network interface card connected to a mirrored port on the switch; and (ii) experimental trial automation on a second network interface card.

A third workstation, containing the programmable controller modular system honeypots and network traffic generators, was connected to both subnets. Three network interface cards were used to provide: (i) a BACnet programmable controller modular honeypot connection to Subnet 1; (ii) a P2 programmable controller modular system honeypot connection to Subnet 1; and (iii) an additional P2 programmable controller modular system honeypot connection to Subnet 2. Connections to both subnets for the P2 programmable controller modular system honeypot and network traffic generator enabled multiple experimental trials to be conducted without significant changes between runs.

4. Pilot Studies

This section highlights the results of the pilot studies.

4.1 APOGEE Network Traffic Analysis

Network communications between the programmable controller modular systems and the human-machine interface workstation were analyzed to discern what an attacker might see in the network. Traffic collection was performed on a Windows-based client using Wireshark. Multiple collections were made with different settings on each switch (e.g., switched and mirrored port configurations). In addition, network traffic was collected with the control system devices in normal operation and failed states.

The captures represent traffic that can be observed by an attacker at various levels of network compromise. The switched ports represent what an attacker who compromises a network node would see on the host. While the attacker may be able to see traffic traversing the compromised node, most traffic destined for other nodes would not be observed. The mirrored port configuration replicates the access that a skilled attacker may gain through various exploits (e.g., MAC address flooding, administrative credential compromise, switch vulnerability exploitation and switch misconfiguration). A successful network compromise could reveal all the traffic traversing the switch to an attacker.

When the sniffer was connected to a switched port, no control traffic between the programmable controller modular systems and the Insight human-machine interface workstation was collected. This is expected because unicast traffic would normally be restricted between the intended source and destination by the switch. When alternating the programmable controller modular system and the human-machine interface workstation into failed states, address resolution protocol (ARP) and BACnet/IP broadcast management device (BBMD) requests were seen. The analysis confirmed that the system uses unicast traffic for normal operations and broadcast messages to discover nodes. Since the broadcast messages are visible without compromising the network, they provide an attacker with information for mapping the control network.

The next set of traffic collection was conducted using a mirrored port. This resulted in the sniffer capturing all unicast traffic between the programmable controller modular systems and human-machine interface workstation. The capture included routine network traffic for the APOGEE platform. By alternating the programmable controller modular system and human-machine interface workstations in failed states, unicast TCP handshakes and BBMD messages were observed in addition to the broadcast requests that were observed previously.

4.2 Identifying Honeypots

The honeypots were constructed using Honeyd [16]. The Honeyd daemon helps create virtual hosts that appear to run specific operating systems and services. It can simulate multiple addresses from a single host. Building the honeypot configuration profile involved retrieving identity data (e.g., operating system fingerprints, MAC addresses and services) from the control system devices using NMAP. The information gathered was transferred to the honeypot configuration profile for Honeyd. Note that the efficiency of honeypot software was not tested, it was only tasked with providing targetable nodes in the network. The final implementation of the network traffic generator should work with any honeypot platform.

A honeypot system with unique IP and MAC addresses was created for each control system device. Figure 4 shows the network topology obtained using an active NMAP scan. For experimentation purposes, the honeypots were configured to drop packets from the network traffic generator. In the final implementation, non-replay and replay traffic could be segregated and handled by the appropriate honeypot network traffic generator platform.

A pilot study was performed to demonstrate that the lack of control network traffic can give away the identities of the honeypots. NMAP was used to perform an active scan of the test network, which verified that the honeypot systems and control devices, were active and detectable in the network. It was also used to validate that the honeypot characteristics matched those of the corresponding APOGEE system. The network topology in Figure 4 shows three real control system devices (i.e., `10.1.3.2`, `10.1.3.3` and `10.1.3.5`) and three honeypot systems (i.e., `10.1.3.111`, `10.1.3.112` and `10.1.3.104`) for the single subnet configuration. Note that a separate honeypot (i.e., `10.1.4.104`) with the dual subnet configuration was used in Subnet 2. From the standpoint of an active scan, the introduction of the three honeypots potentially decreases an attacker's target selection success by 50% (three real control system devices

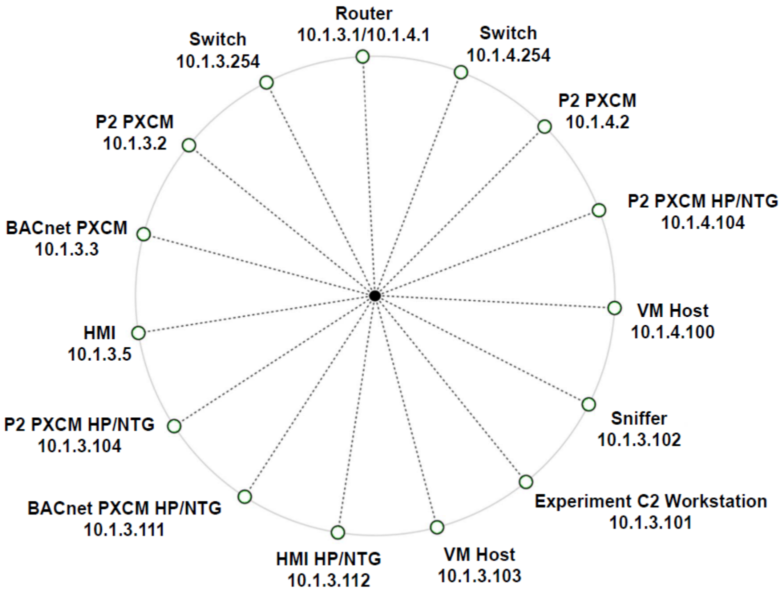


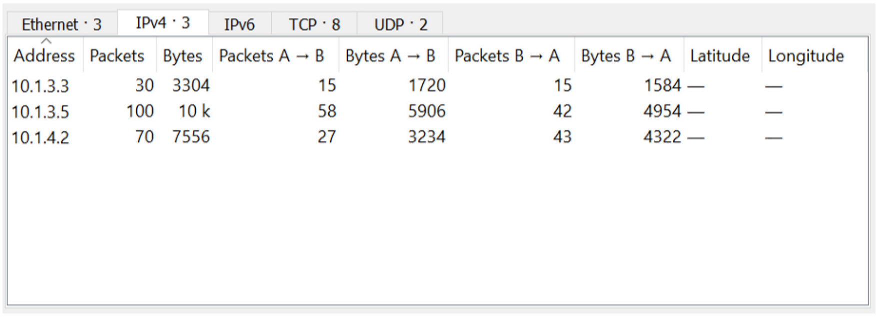
Figure 4. Test network implementation.

out of six control system devices). This target selection success rate can be further decreased by introducing additional honeypots.

An assessment of the traffic data is required because a skilled attack may involve passive network monitoring. To simulate the highest level of compromise to the network, a Wireshark sniffer was placed at a mirrored port of the switch that serviced the human-machine interface. Upon examining only a few minutes of traffic data, the three real control system devices were identified using Wireshark's endpoints statistics tool (see Figure 5). An attacker who can view network traffic would conclude that the targets of interest have the IP addresses `10.1.3.3`, `10.1.3.5` and `10.1.4.2` because only these systems actively transmitted on the network. With this additional reconnaissance, an attacker's target selection success returns to 100%. While the initial implementation of honeypot systems in the control network appeared promising, the pilot study demonstrates that a skilled attacker can easily detect a honeypot via passive monitoring.

4.3 Tcpreplay Network Traffic Generation

In this pilot study, network captures (single and dual subnet configurations) were taken from the APOGEE platform. These PCAP files are referred to as production traces. The Tcpreplay suite contains a variety of tools including: (i) `tcpprep`; (ii) `tcpreplay`; and (iii) `tcprewrite`.



Ethernet · 3 | IPv4 · 3 | IPv6 | TCP · 8 | UDP · 2

Address	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Latitude	Longitude
10.1.3.3	30	3304	15	1720	15	1584	—	—
10.1.3.5	100	10 k	58	5906	42	4954	—	—
10.1.4.2	70	7556	27	3234	43	4322	—	—

Figure 5. Pre-network traffic generation passive monitoring.

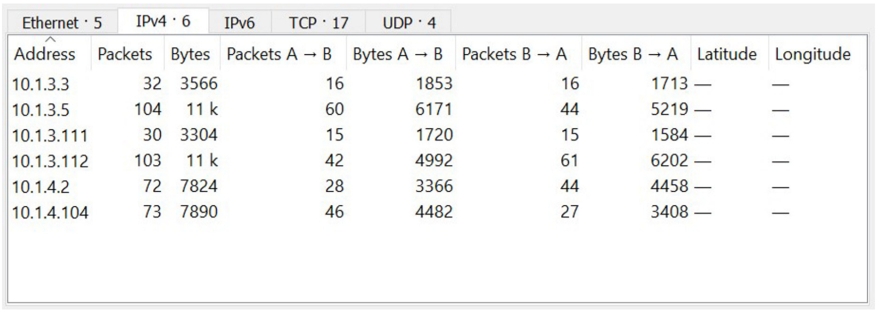
The `tcpprep` tool is a PCAP pre-processor for `tcpreplay` and `tcprewrite`. It identifies and "splits" traffic into two sides of a conversation and assigns a network interface to each packet. It enables the `tcpreplay` tool to generate network traffic through two network interface cards (primary/secondary and client/server). While it can emulate two-way traffic through two unique ports, network traffic generation is still limited to a single workstation and two interfaces. This is a limitation when simulating an industrial control environment that typically incorporates a number of networked devices that are physically and logically distributed.

The `tcprewrite` tool is a PCAP file editor that rewrites TCP/IP and Layer 2 packet headers. It replaces the Layer 2 source and destination addresses of packets so that they can be processed by the correct devices. Operational tests revealed that, while the tool could be used to rewrite packets, it was dependent on `tcpprep` being able to identify conversations properly. In fact, the tool failed to modify the source and destination addresses of all the packets in a sample capture from the APOGEE platform.

The `tcpreplay` tool was used to generate traffic in the network using the original timings recorded in the production trace. A sample capture consisting of one minute of network traffic data was recorded by the sniffer.

Figure 6 presents the results of a capture obtained by the Wireshark endpoints statistics tool. The figure shows that an attacker who can monitor the network would see traffic between all six systems, maintaining 50% target selection success probability corresponding to three real control system devices out of six control system devices. Because the real and honeypot systems generate traffic, an attacker would have to perform a deeper analysis to determine the targets of interest. This requires more steps, which provides defenders with more time to detect and mitigate the attack. This demonstrates that adding decoys with the corresponding network traffic makes it more difficult for an attacker to select a legitimate target.

After the initial test showed that `tcpreplay` was capable of deception in passive scans, a full hour of traffic was generated in Subnet 1. Wireshark was

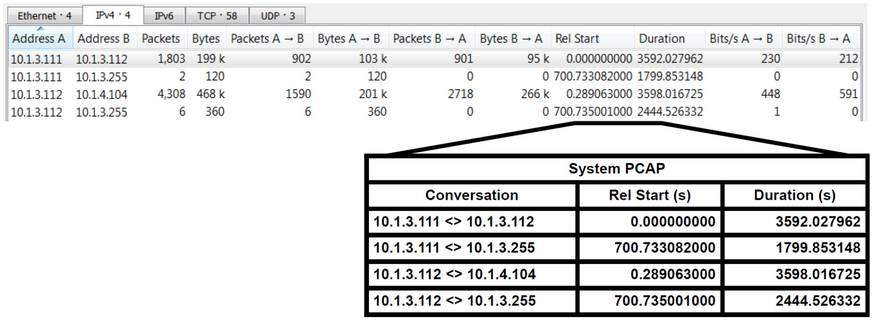


Ethernet · 5 | IPv4 · 6 | IPv6 | TCP · 17 | UDP · 4

Address	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Latitude	Longitude
10.1.3.3	32	3566	16	1853	16	1713	—	—
10.1.3.5	104	11 k	60	6171	44	5219	—	—
10.1.3.111	30	3304	15	1720	15	1584	—	—
10.1.3.112	103	11 k	42	4992	61	6202	—	—
10.1.4.2	72	7824	28	3366	44	4458	—	—
10.1.4.104	73	7890	46	4482	27	3408	—	—

Figure 6. Post-network traffic generation passive monitoring.

used to capture the generated traffic for data collection and analysis in Subnet 1 and Subnet 2. Statistical tools provided by Wireshark were used to compare the production trace packets against the generated packets (referred to as the generated trace).



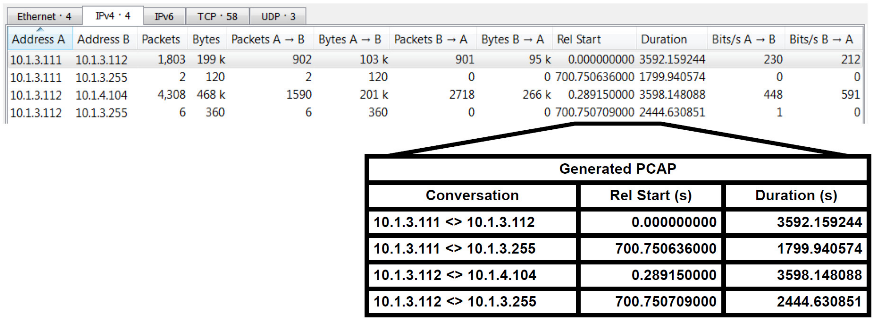
Ethernet · 4 | IPv4 · 4 | IPv6 | TCP · 58 | UDP · 3

Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.3.111	10.1.3.112	1,803	199 k	902	103 k	901	95 k	0.000000000	3592.027962	230	212
10.1.3.111	10.1.3.255	2	120	2	120	0	0	700.733082000	1799.853148	0	0
10.1.3.112	10.1.4.104	4,308	468 k	1590	201 k	2718	266 k	0.289063000	3598.016725	448	591
10.1.3.112	10.1.3.255	6	360	6	360	0	0	700.735001000	2444.526332	1	0

System PCAP		
Conversation	**Rel Start (s)**	**Duration (s)**
10.1.3.111 <> 10.1.3.112	0.000000000	3592.027962
10.1.3.111 <> 10.1.3.255	700.733082000	1799.853148
10.1.3.112 <> 10.1.4.104	0.289063000	3598.016725
10.1.3.112 <> 10.1.3.255	700.735001000	2444.526332

Figure 7. Conversation statistics from `tcpreplay` (production trace).

The traffic capture from Subnet 1 was analyzed first. Wireshark conversation statistics that list conversations (traffic between two endpoints) revealed that there were four pairs of conversations. Conversation statistics for the production trace and generated trace are shown in Figures 7 and 8, respectively. The numbers of packets transmitted and the total sizes of the conversations in the two traces matched. However, there was an average delay of 11.12 ms before the start of conversations and an average increase of 113.65 ms in the durations of conversations. The `tcpreplay` timing measurements (using methods described in Section 4.6) revealed that the difference in the inter-packet timings between the production trace and generated trace had a mean of 0.07 ms. Overall, an increase of 132 ms was observed in the one-hour duration of the replay.



Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.3.111	10.1.3.112	1,803	199 k	902	103 k	901	95 k	0.000000000	3592.159244	230	212
10.1.3.111	10.1.3.255	2	120	2	120	0	0	700.750636000	1799.940574	0	0
10.1.3.112	10.1.4.104	4,308	468 k	1590	201 k	2718	266 k	0.289150000	3598.148088	448	591
10.1.3.112	10.1.3.255	6	360	6	360	0	0	700.750709000	2444.630851	1	0

Generated PCAP		
Conversation	Rel Start (s)	Duration (s)
10.1.3.111 <> 10.1.3.112	0.000000000	3592.159244
10.1.3.111 <> 10.1.3.255	700.750636000	1799.940574
10.1.3.112 <> 10.1.4.104	0.289150000	3598.148088
10.1.3.112 <> 10.1.3.255	700.750709000	2444.630851

Figure 8. Conversation statistics from `tcpreplay` (generated trace).

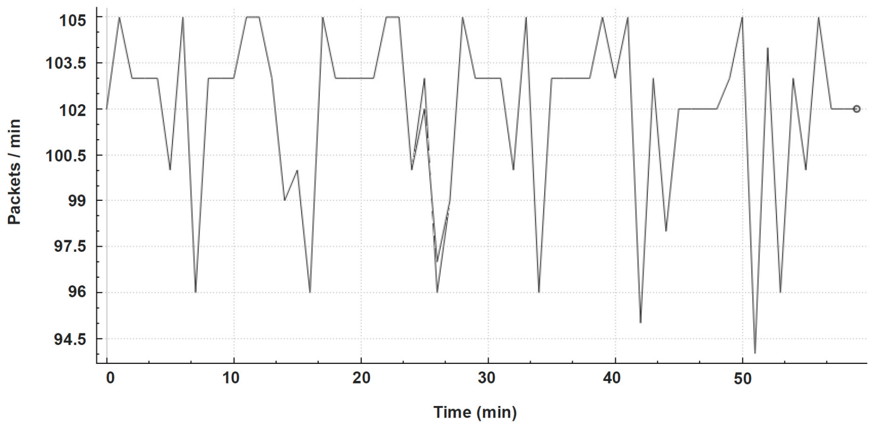


Figure 9. Packet transmission rates for sample production and generated traces.

Some minor variations in the numbers of packets sent was observed at the 22 to 28 minute mark due to processing or networking delays. The Wireshark input/output graph statistics show the numbers of packets transmitted per unit of time over the course of the capture. Figure 9 shows superimposed graphs of the packet transmission rates from a sample production trace and a sample generated trace. The superimposition reveals that the two traces have similar network traffic patterns. However, it would be difficult for an attacker to use this visual representation to identify the traffic pattern that is associated with the real system.

The pilot study revealed some limitations of the `tcpreplay` tool. Sniffing in Subnet 2 revealed that only a limited number of packets destined for the P2 programmable controller modular system honeypot were received. This is attributed to the port assignments made by the switch in its MAC address table and the way in which `tcpreplay` operates. Since `tcpreplay` transmitted the generated traffic via an interface in Subnet 1, all the MAC addresses from the production trace were entered into the address table of the switch and assigned to a particular port in Subnet 1. All network communications (including those outside the generated traffic) destined for the P2 programmable controller modular system honeypot were then sent to this port. Occasionally, the MAC tables refreshed with the actual locations of the honeypot (via the router MAC address). During these periods, packets were forwarded correctly to the router and the correct subnet. This is a limitation in using `tcpreplay` for distributed systems in multi-subnet environments.

If the network topology had comprised a single subnet, then an implementation using the `tcpreplay` tool would be a viable solution. It would also require that the honeypots be collocated on the same workstation as `tcpreplay`; otherwise, multiple associations of non-unique MAC addresses would occur. If any honeypot were to be placed separately from the network traffic generator, an attacker would not see any replayed traffic on the host. This imposes a hardware limitation for designs that require multiple honeypots in the same workstation. Indeed, the pilot study reveals that the `tcpreplay` suite does not meet the honeypot integration, network routing and scalability criteria specified in Table 1.

5. Implementation

While authentic network traffic can be generated by a full-scale system, the cost of such a decoy can be prohibitive [26]. Low-interaction and hybrid honeypot systems can be used to emulate industrial control systems at a much lower cost. However, protocol-specific traffic generation requires significant knowledge, time and resources to replicate the operations of a genuine system with high fidelity.

An alternate solution is to use packet captures for network traffic generation. This method maintains authenticity because the data and patterns are derived from real systems. However, the confidentiality of the traffic is not maintained because a capable attacker would most likely be able to view the traffic in a real system in the absence of honeypots. One way to mitigate this and maintain some form of confidentiality is to use multiple sets of false captures. The false captures would be generated by real systems that run in a fake operational environment. The traffic would look real, but it would have no operational use. Thus, the real system and the honeypots would have distinct traffic, but the honeypots would still function as decoys and present themselves as attractive targets to an attacker.

The experimental network incoporated five components: (i) production network; (ii) honeypot platform; (iii) honeypot integration; (iv) distributed network traffic generation; and (v) decoy network. Figure 10 shows a block diagram of the design. The production network provided device characteristics (e.g., operating system fingerprints, network addresses and trace data). The honeypot platform provided targetable honeypots in a decoy network with

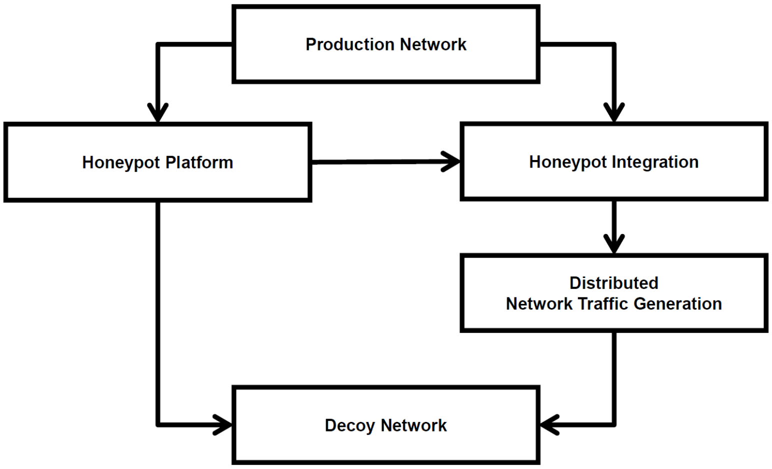


Figure 10. Honeypot-network traffic generation design.

matching characteristics and emulated services associated with the production network. Honeypot integration modified the headers of trace data captured from a production network to match the corresponding honeypot systems. After the honeypot integration was completed, the trace data was used by the distributed network traffic generators to produce and replay industrial control system traffic in the decoy network.

A custom network traffic generator solution was designed due to various limitations in common off-the-shelf and open source products. The distributed network traffic generator (DNTG) was created using Python and Scapy following the design criteria in Table 1 and drawing on the design characteristics of D-ITG [2]. DNTG has two main components: (i) DNTG Prep; and (ii) DNTG Replay. DNTG Prep is a PCAP tool that modifies packet header information to match the honeypot characteristics (i.e., IP and MAC addresses). These characteristics are passed as parameters to DNTG Prep, which modifies the control system packets with the corresponding honeypot information.

DNTG Replay was created to run on each corresponding honeypot device and to operate in a distributed environment. Each DNTG Replay instance functions as a listener and sender. As a listener, DNTG Replay continuously waits and listens for incoming packets. When a packet is received, DNTG Replay searches the production trace to verify if the packet corresponds to generated traffic. If a match is found, DNTG Replay searches for the corresponding responses. The appropriate responses are then placed in a queue for network traffic generation. As a sender, DNTG transmits the queued packets based on the inter-packet timing determined from the production trace. Figure 11 shows an example DNTG architecture containing three DNTG Replay instances in a decoy network.

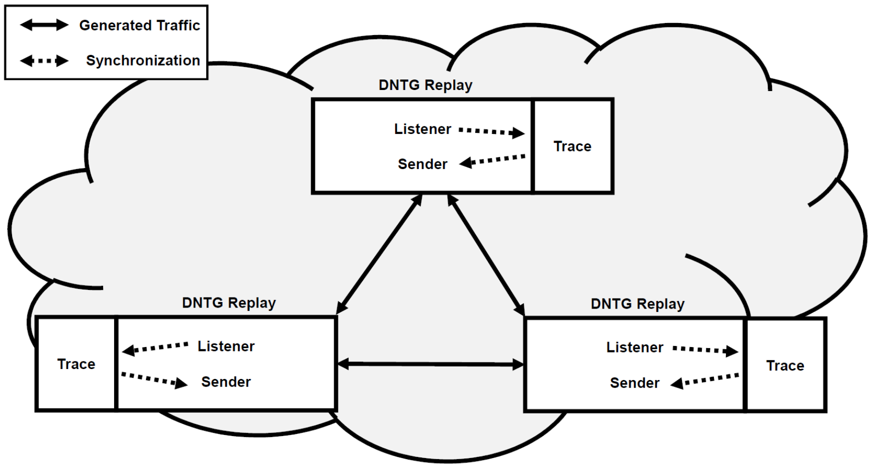
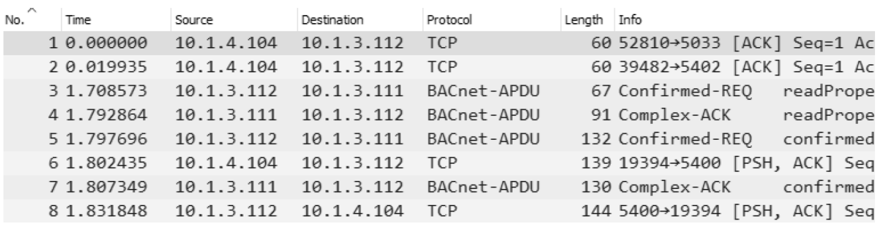


Figure 11. Example DNTG architecture.

5.1 Traffic Matching

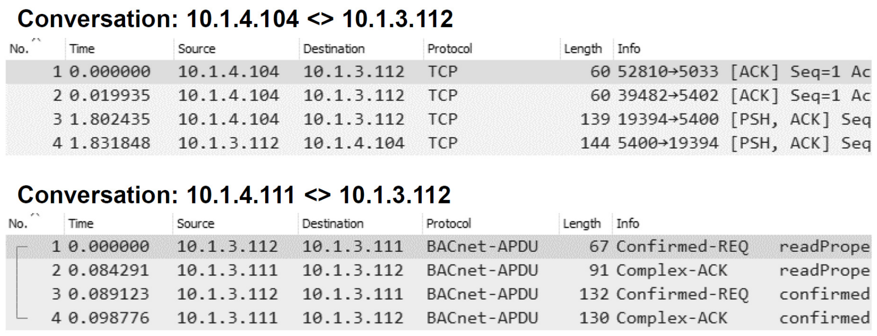
Synchronization is important for multiple DNTG instances to operate in a distributed environment. However, no extraneous packets are allowed during network traffic generation per the criteria in Table 1. Without any command and control traffic, synchronization is dependent on the generated packets. If a network traffic generator is not ready to receive, it may inadvertently miss incoming packets. It is crucial during initialization that each DNTG is ready before traffic generation begins. To perform the initial synchronization, a master DNTG is selected based on the first packet in the production trace. The DNTG instance with the originating IP source address of the packet becomes the master and all other instances listen for network traffic upon initialization. The master DNTG sends a periodic UDP request to the other DNTG instances. When a request is received, a response is sent back to the master. After the responses from all the DNTG instances are received, the master DNTG sends a UDP start packet to each DNTG instance to begin traffic generation. No additional synchronization packets are required after this initial period.

Each DNTG uses an identical production trace and operates asynchronously. By removing command and control traffic during traffic generation, each DNTG is responsible for keeping track of the current location in the production trace and resynchronizing based on the received packets. As each DNTG receives and sends replay packets, it resynchronizes with the production trace. These command and control operations are transparent to an attacker because they are performed locally and not over the network. Only production trace data is transmitted by each DNTG instance during network traffic generation.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.1.4.104	10.1.3.112	TCP	60	52810→5033 [ACK] Seq=1 Ac
2	0.019935	10.1.4.104	10.1.3.112	TCP	60	39482→5402 [ACK] Seq=1 Ac
3	1.708573	10.1.3.112	10.1.3.111	BACnet-APDU	67	Confirmed-REQ readPrope
4	1.792864	10.1.3.111	10.1.3.112	BACnet-APDU	91	Complex-ACK readPrope
5	1.797696	10.1.3.112	10.1.3.111	BACnet-APDU	132	Confirmed-REQ confirmed
6	1.802435	10.1.4.104	10.1.3.112	TCP	139	19394→5400 [PSH, ACK] Seq
7	1.807349	10.1.3.111	10.1.3.112	BACnet-APDU	130	Complex-ACK confirmed
8	1.831848	10.1.3.112	10.1.4.104	TCP	144	5400→19394 [PSH, ACK] Seq

Figure 12. Production trace sample.



Conversation: 10.1.4.104 <> 10.1.3.112

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.1.4.104	10.1.3.112	TCP	60	52810→5033 [ACK] Seq=1 Ac
2	0.019935	10.1.4.104	10.1.3.112	TCP	60	39482→5402 [ACK] Seq=1 Ac
3	1.802435	10.1.4.104	10.1.3.112	TCP	139	19394→5400 [PSH, ACK] Seq
4	1.831848	10.1.3.112	10.1.4.104	TCP	144	5400→19394 [PSH, ACK] Seq

Conversation: 10.1.4.111 <> 10.1.3.112

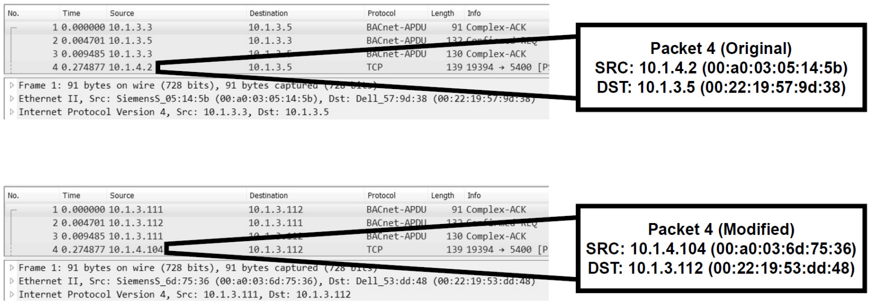
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.1.3.112	10.1.3.111	BACnet-APDU	67	Confirmed-REQ readPrope
2	0.084291	10.1.3.111	10.1.3.112	BACnet-APDU	91	Complex-ACK readPrope
3	0.089123	10.1.3.112	10.1.3.111	BACnet-APDU	132	Confirmed-REQ confirmed
4	0.098776	10.1.3.111	10.1.3.112	BACnet-APDU	130	Complex-ACK confirmed

Figure 13. Sample conversations.

To meet the packet ordering criteria, the DNTG was designed to handle multiple conversations simultaneously. The DNTG parses the production trace and separates the packets into conversations. All traffic to and from an IP address pair are identified as a single conversation to limit the amount of data processed by each DNTG. As such, a conversation is defined as a traffic flow between two unique IP addresses. Using multi-threading, each conversation is processed independently to enable unique conversations to intermingle. This adds variability to the overall capture while maintaining packet order within each conversation. Figure 12 shows packets from a production trace and Figure 13 shows how the packets are segmented into conversations.

Timing consistency is required to replicate the original system as accurately as possible. Autonomous industrial control network traffic consists of routine/polling messages that are sent and received at timed intervals. The DNTG should replicate these intervals (inter-packet timing from the production trace) and not use transmission timing methods such as packets per second or burst modes.

Two methods were considered to handle the timing: (i) compute the time between the current queued packet and the first packet of the conversation; and (ii) use the inter-packet timing. The first method implements a catch up



```
No.  Time     Source     Destination  Protocol     Length Info
  1 0.000000 10.1.3.3   10.1.3.5     BACnet-APDU      91 Complex-ACK
  2 0.004701 10.1.3.5   10.1.3.3     BACnet-APDU     132 Confirmed-REQ
  3 0.009485 10.1.3.3   10.1.3.5     BACnet-APDU     130 Complex-ACK
  4 0.274877 10.1.4.2   10.1.3.5     TCP             139 19394 → 5400 [P
▷ Frame 1: 91 bytes on wire (728 bits), 91 bytes captured (728 bits)
▷ Ethernet II, Src: SiemensS_05:14:5b (00:a0:03:05:14:5b), Dst: Dell_57:9d:38 (00:22:19:57:9d:38)
▷ Internet Protocol Version 4, Src: 10.1.3.3, Dst: 10.1.3.5
```

Packet 4 (Original)
SRC: 10.1.4.2 (00:a0:03:05:14:5b)
DST: 10.1.3.5 (00:22:19:57:9d:38)

```
No.  Time     Source       Destination  Protocol     Length Info
  1 0.000000 10.1.3.111   10.1.3.112   BACnet-APDU      91 Complex-ACK
  2 0.004701 10.1.3.112   10.1.3.111   BACnet-APDU     132 Confirmed-REQ
  3 0.009485 10.1.3.111   10.1.3.112   BACnet-APDU     130 Complex-ACK
  4 0.274877 10.1.4.104   10.1.3.112   TCP             139 19394 → 5400 [P
▷ Frame 1: 91 bytes on wire (728 bits), 91 bytes captured (728 bits)
▷ Ethernet II, Src: SiemensS_6d:75:36 (00:a0:03:6d:75:36), Dst: Dell_53:dd:48 (00:22:19:53:dd:48)
▷ Internet Protocol Version 4, Src: 10.1.3.111, Dst: 10.1.3.112
```

Packet 4 (Modified)
SRC: 10.1.4.104 (00:a0:03:6d:75:36)
DST: 10.1.3.112 (00:22:19:53:dd:48)

Figure 14. Production trace modification.

algorithm to ensure that processing and network delays do not add to the overall length of traffic generation (start to finish); however, significant changes in inter-packet timing were observed. The second method uses inter-packet timing to generate packets without a catch up algorithm. This method was chosen for implementation; although, the overall length of a generated trace is longer than that of a production trace, the inter-packet timing is more consistent with the original intervals.

5.2 Honeypot Integration

As shown in Figure 1, the main network traffic collection occurs in Subnet 1. Since control system devices communicate with the human-machine interface located on this subnet, this is the best traffic collection point for an attacker. To obtain the production trace, network traffic was collected from a mirrored port for ten minutes. IP address filters were used to capture network traffic only from the APOGEE platform. In the single subnet, the IP addresses `10.1.3.2`, `10.1.3.3` and `10.1.3.5` were captured. In the dual subnet setup, the IP addresses `10.1.4.2`, `10.1.3.3` and `10.1.3.5` were captured. A separate PCAP file was created for each subnet configuration.

The production trace used for traffic generation contains characteristics of the real control system devices. Since the generated traffic must match the characteristics of the honeypot systems (i.e., IP and MAC addresses), the trace requires modification. Preparation of the production trace using DNTG Prep was chosen instead of altering the values during traffic generation. This reduces the impact on DNTG Replay performance during runtime. Each packet in the production trace was thus overwritten with the desired replacement IP and MAC addresses of the corresponding honeypot system. Checksum values were also corrected to maintain packet validity. The top portion of Figure 14 shows the original production trace with the addresses reflecting the real control system devices. The bottom portion of the figure shows the modified production trace with the replacement addresses of the honeypot systems.

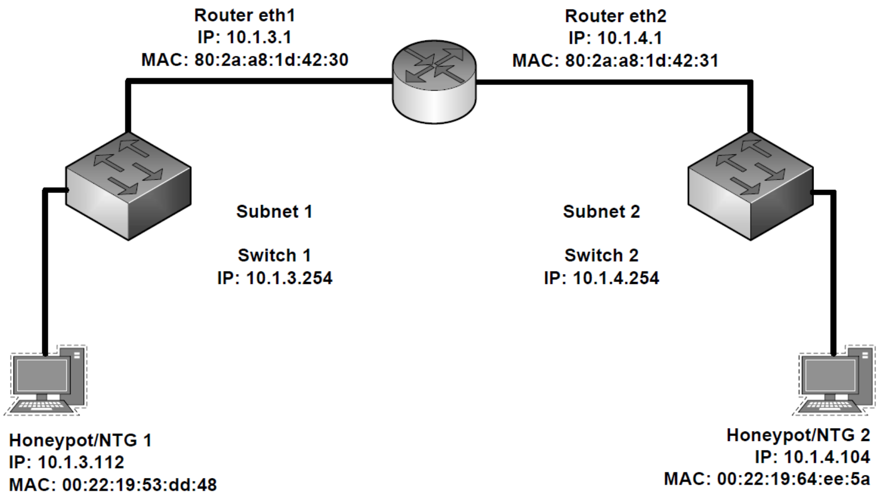


Figure 15. Example dual subnet environment.

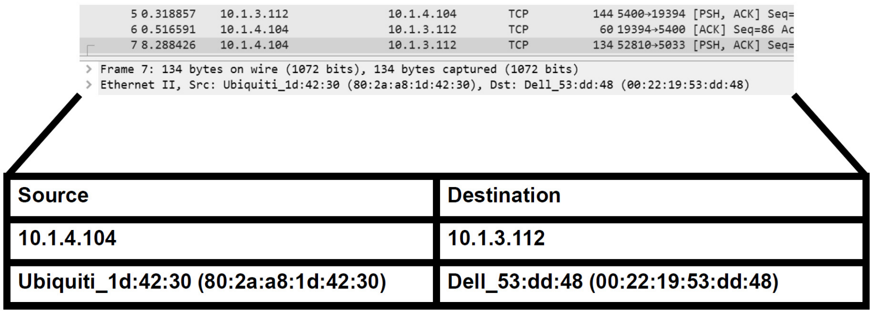
5.3 Network Routing

The network routing criteria focus on the ability of DNTG instances to operate in a distributed environment. To meet this requirement, the generated traffic must have proper Layer 2 and Layer 3 addresses.

Layer 2 addressing requires that generated traffic that shares MAC addresses with the corresponding honeypots must not cause network addressing problems. To eliminate Layer 2 networking problems, network traffic generators that share IP and MAC addresses with a honeypot must be collocated on the same workstation.

Proper Layer 3 routing ensures that DNTG instances can operate in a multi-subnet environment. Figure 15 shows an example dual subnet topology.

Recall that the production trace was captured in Subnet 1 and all traffic from outside subnets would have recorded the MAC address of router `eth1` as the source (shown in Figures 15 and 16). Problems arise when generating traffic from devices in Subnet 2 with trace data captured in Subnet 1. Using packet 7 in Figure 16 as an example, the header contains incorrect address values: (i) the source is the MAC address of router `eth1` (`80:2a:a8:1d:42:30`); and (ii) the destination is the MAC address of NTG 1 (`00:22:19:53:dd:48`). DNTG identifies these instances and modifies the header with the correct values: (i) the new source is the MAC address of NTG 2 (`00:22:19:64:ee:5a`); and (ii) the new destination is the MAC address of router `eth2` (`80:2a:a8:1d:42:31`). These corrections are performed during runtime to avoid customizing the production trace for each DNTG instance.



```
     5 0.318857    10.1.3.112              10.1.4.104       TCP          144 5400→19394 [PSH, ACK] Seq=
     6 0.516591    10.1.4.104              10.1.3.112       TCP           60 19394→5400 [ACK] Seq=86 Ac
     7 8.288426    10.1.4.104              10.1.3.112       TCP          134 52810→5033 [PSH, ACK] Seq=
> Frame 7: 134 bytes on wire (1072 bits), 134 bytes captured (1072 bits)
> Ethernet II, Src: Ubiquiti_1d:42:30 (80:2a:a8:1d:42:30), Dst: Dell_53:dd:48 (00:22:19:53:dd:48)
```

Source	Destination
10.1.4.104	10.1.3.112
Ubiquiti_1d:42:30 (80:2a:a8:1d:42:30)	Dell_53:dd:48 (00:22:19:53:dd:48)

Figure 16. Example MAC addressing problem.

6. Experiments

Multiple experimental trials were conducted to evaluate if the DNTG implementation satisfies the network traffic generation criteria listed in Table 1. Each experimental trial involved generating network traffic from a ten-minute production trace. The trials were conducted using an automated script that alternated the operating environments (single and dual subnets). A total of 177 iterations were conducted for each environment, corresponding to a grand total of 354 experimental trials. The trials were conducted over a 65-hour time period and the outputs provided more than 375,000 sample packets for analysis.

6.1 Metrics

This section outlines the metrics used to validate the DNTG implementation against the criteria listed in Table 1.

Table 2. Traffic matching metrics.

Content Matching	**Packet Bytes:** Generated packets bytes match the production trace bytes.
Extraneous Packets	**Quantity of Packets:** Number of generated packets match the number of production trace packets.
Packet Ordering	**Packet Order:** Generated conversations match the production trace conversations.
Timing Consistency	**Δ Inter-Packet Time:** Generated traffic timing patterns match the production trace timing patterns.

Traffic Matching. Table 2 describes the metrics used to evaluate the DNTG implementation against the traffic matching criteria. Content matching is evaluated by directly comparing corresponding packets between the generated trace and production trace. Two packets match if both packets contain the same bytes. A trial is considered to be successful if every packet in the generated trace matches the corresponding packet in the production trace.

The generated trace is determined to have no extraneous packets if it contains the same number of packets as the production trace. A trial is considered to be successful if the quantity and content of the packets in the generated trace match those in the production trace.

Packet ordering is determined to match if the packet orders of conversations in the generated trace match the packet orders of the corresponding conversations in the production trace. A trial is considered to be successful if every packet in the generated trace is in the correct order.

Timing consistency is measured by comparing the inter-packet timing of packets in the generated trace with the inter-packet timing of packets in the production trace.

The inter-packet time of packet n in the generated trace $GIPT_n$ is computed as:

$$GIPT_n = GT_n - GT_{n-1} \tag{1}$$

The inter-packet time of packet n in the production trace $PIPT_n$ is computed as:

$$PIPT_n = PT_n - PT_{n-1} \tag{2}$$

Finally, the difference in the inter-packet timing in the two traces ΔIPT is computed as:

$$\Delta IPT_n = ABS(GIPT_n - PIPT_n) \tag{3}$$

The Wilcoxon rank-sum non-parametric statistical test was used to compare the distribution of $GIPT$ values to the distribution of $PIPT$ values for each generated trace. A significance level of 0.05 was used to determine if the two distributions are statistically similar. A trial is considered to be a success if the Wilcoxon rank-sum test returned a p-value greater than 0.05.

Honeypot Integration. Honeypot pairing was evaluated based on direct comparisons of corresponding packets from the generated and production traces. The DNTG was designed to only generate traffic if it started a conversation or was responding to a received packet. In addition, the DNTG was hosted on the same honeypot workstation and used matching IP and MAC addresses. Honeypot pairing was determined to be a match if packets were sent and received from each honeypot workstation. To validate the honeypot header matching, an NMAP scan was used to obtain the IP and MAC addresses of each honeypot. This information was then compared against the production trace and generated traces to validate that the generated traffic matched the honeypots. A trial is considered to be successful if every packet header matched the intended honeypot information and every packet satisfied the content matching and extraneous packets criteria.

Table 3. Traffic matching criteria success rates.

	Trials	**Single Subnet**	**Dual Subnet**
Content Matching	354	100%	100%
Extraneous Packets	354	100%	100%
Packet Ordering	354	100%	100%
Timing Consistency	354	0%	0%

Network Routing. The DNTG was designed to run with multiple instances at various locations in a network. Network traffic generation involves each DNTG instance receiving and generating network traffic. A trial is considered to be successful if every packet in the generated trace and the corresponding production trace satisfies the traffic matching criteria. A successful trial implies that the distributed operation, Layer 2 addressing and Layer 3 routing evaluation criteria are satisfied.

Scalability. The scalability of a network traffic generator is based primarily on the design, implementation and pricing of the final implementation. This research did not consider the actual costs (in terms of dollars), because most instantiations would require engineering efforts to determine the proper placement of the traffic generators. While the DNTG is designed to be as flexible as possible, the experiments were limited to the Siemens APOGEE system. Experiments with other implementations are left for future work.

6.2 Experimental Results

A total of 354 experimental trials were conducted and evaluated against the criteria specified in Table 1.

Traffic Matching. All 354 trials achieved 100% matching of the production and generated traces for the metrics: (i) packet bytes; (ii) quantity of packets; and (iii) packet ordering. Table 3 shows the traffic matching success rates for the various criteria.

Timing consistency was evaluated by using a Wilcoxon rank-sum test. This test compared the $GIPT$ distribution in each of the 354 generated traces with the $PIPT$ distribution in the production trace. The Wilcoxon rank-sum test returned a p-value less than 0.05 for all 354 generated trace distributions. Therefore, the generated traffic timing does not match the production traffic timing.

Table 4 shows the ΔIPT summary for all 354 trials. Mean differences of 2.07 ms (single subnet) and 2.26 ms (dual subnet) were observed between the production and generated traces. Further examination using a Wilcoxon rank-sum test revealed that that the generated traces from multiple trials are consistent with each other.

Table 4. ΔIPT (ms) summary.

	Mean	**Min**	**Max**	**SD**
Single Subnet	2.07	0.00	95.78	1.59
Dual Subnet	2.26	0.00	98.93	1.68

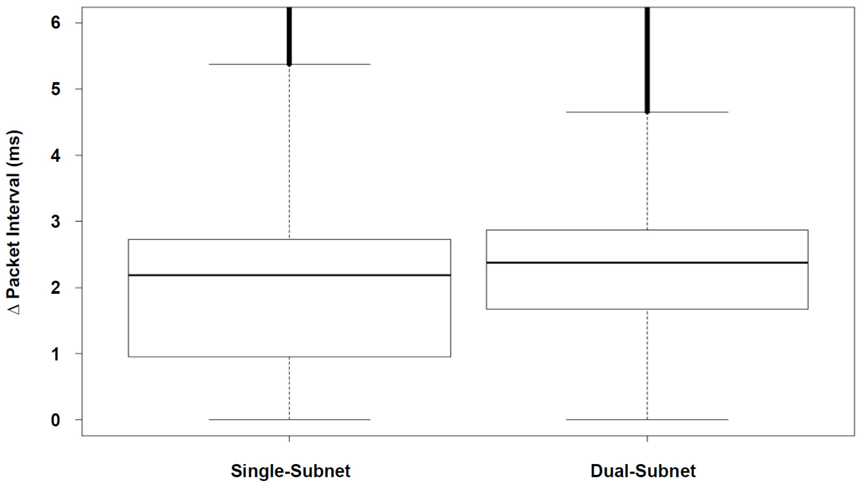


Figure 17. Difference in timing between system and generated intervals.

Figure 17 shows a boxplot of two subnet configurations with similar timing. Figure 18 shows an overlay of the production trace (line) and a sample generated trace (points). Figure 19 shows detailed views of the intervals between packets 800 and 820. It is difficult to visually distinguish the real system from the honeypot system.

The DNTG implementation successfully replicates control system traffic in a distributed environment. While the results indicate that the production and generated traces do not have the same timing, multiple runs demonstrated consistency in the DNTG outputs. This suggests that optimizing the software could reduce the timing differences.

Honeypot Integration. All 354 trials resulted in 100% matches between the production and generated traces for the criteria: (i) content matching; (ii) extraneous packets; and (iii) packet ordering. Satisfaction of these three criteria demonstrates that traffic was generated to (and from) the honeypot network traffic generator pair. In addition, because the generated traffic matches the production trace (validated against the honeypots using NMAP), the honeypot

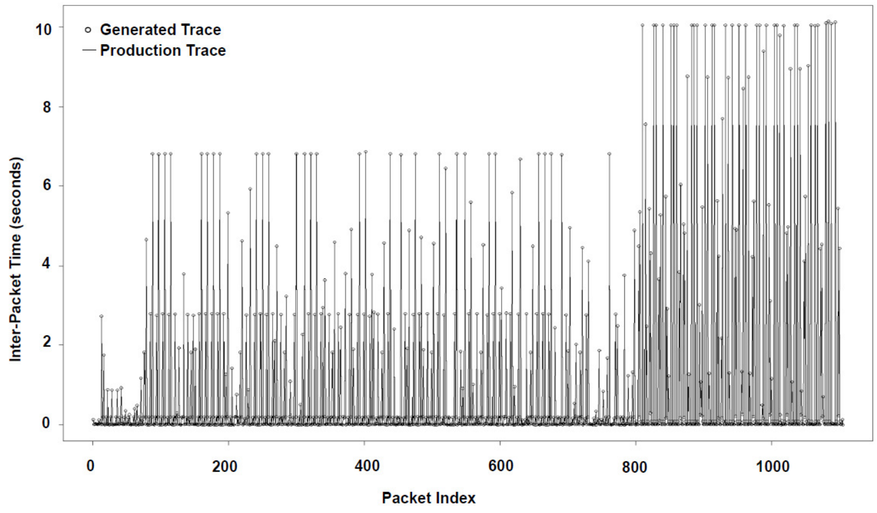


Figure 18. Traffic timing pattern.

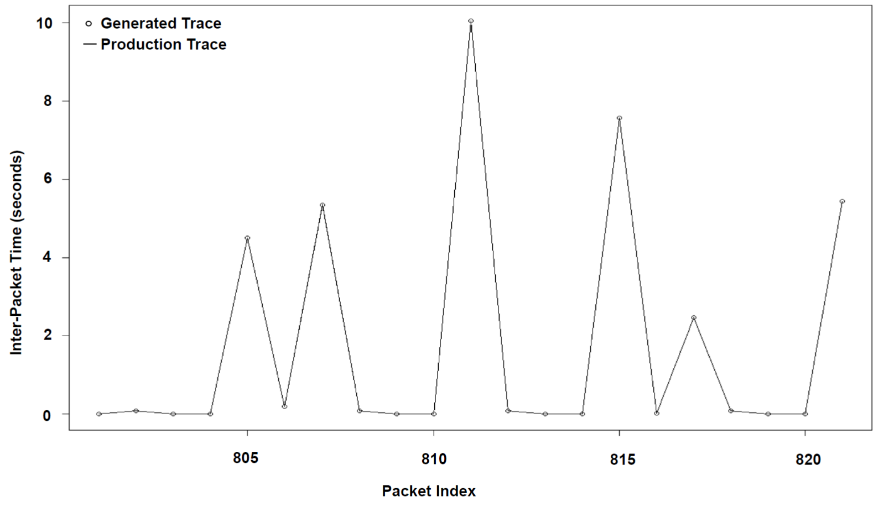
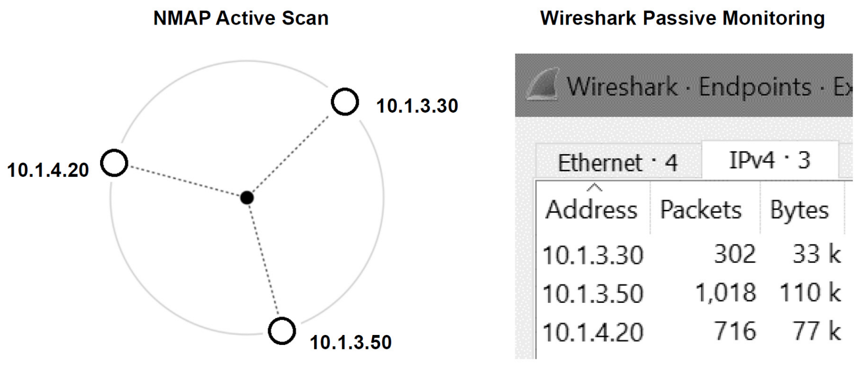


Figure 19. Detailed traffic timing pattern.

header matching criterion is also satisfied. Table 5 shows the honeypot integration criteria pass rates. Figure 20 shows an NMAP active scan and a Wireshark passive mapping of the decoy network used during the experiment. The figure shows that three honeypot systems are detected during an active network scan.

Table 5. Honeypot integration criteria pass rates.

	Trials	**Single Subnet**	**Dual Subnet**
Honeypot Pairing	354	100%	100%
Honeypot Header Matching	354	100%	100%



Address	Packets	Bytes
10.1.3.30	302	33 k
10.1.3.50	1,018	110 k
10.1.4.20	716	77 k

Figure 20. Active and passive network mapping.

The network traffic generated by the DNTG is observed during passive network monitoring. The combination of honeypot systems and network traffic mimic an operating industrial control system.

Table 6. Network routing criteria pass rates.

	Trials	**Single Subnet**	**Dual Subnet**
Distributed Operation	354	100%	100%
Layer 2 Addressing	354	100%	100%
Layer 3 Routing	354	100%	100%

Network Routing. All 354 trials achieved 100% matches between the production trace and generated traces for the criteria: (i) content matching; (ii) extraneous packets; and (iii) packet ordering. Satisfaction of these three criteria demonstrates that traffic was generated to and from multiple instances of the DNTG located in two different network configurations. This shows that Layer 2 addressing and Layer 3 routing were successfully accomplished and validates that the DNTG satisfies the distributed operation criteria. Table 6 shows the network routing criteria pass rates.

Scalability. The DNTG is designed to be cost effective and flexible, but the research did not evaluate these characteristics. The evaluation of cost effectiveness and flexibility is left for future work.

7. Conclusions

Honeypots can be used to protect industrial control systems. The effectiveness of a honeypot is dependent on its ability to entice attackers to select it as a target. Using network traffic generators to create traffic in honeypot networks helps build a realistic decoy industrial control system environment. Indeed, the experimental results demonstrate that network traffic generation can significantly complicate the task of honeypot identification during target selection by an attacker.

Future research will focus on alternating production traces or modifying packet data and values to maintain uniqueness during multiple iterations. This is important because replays of trace data are only as effective as the lengths of the traces. For example, if an hour-long production trace were to be used, the first hour of generated traffic would appear to be authentic. However, after this time period, the same packets would be generated again; this traffic would be automatically highlighted as a retransmission by several tools, including Wireshark.

Another topic for future research is motivated by the fact that the use of trace data does not account for real-time system changes. State changes made during operations may contradict traffic data generated by the DNTG. This would be challenging to implement because it requires detailed knowledge about industrial control system protocols. One approach is to use "live" production data to update the network traffic generator. Future research will attempt to develop and evaluate possible solutions to this problem.

Note that the views expressed in this chapter are those of the authors and do not reflect the official policy or position of the U.S. Air Force, U.S. Army, U.S. Department of Defense or U.S. Government.

Acknowledgement

This research was partially supported by the U.S. Department of Homeland Security Industrial Control Systems Cyber Emergency Response Team (ICS-CERT).

References

[1] Armed Forces History Museum, World War II's U.S. Ghost Army, Largo, Florida (`www.armedforcesmuseum.com/world-war-iis-us-ghost-army`), February 5, 2014.

[2] A. Botta, A. Dainotti and A. Pescape, A tool for the generation of realistic network workload for emerging networking scenarios, *Computer Networks*, vol. 56(15), pp. 3531–3547, 2012.

[3] A. Botta, W. de Donato, A. Dainotti, S. Avallone and A. Pescape, D-ITG 2.8.1 Manual, Computer for Interaction and Communications (COMICS) Group, Department of Electrical Engineering and Information Technologies, University of Naples Federico II, Naples, Italy (www.grid.unina.it/software/ITG/manual), 2013.

[4] L. Even, IDFAQ: What is a Honeypot? SANS Institute, Bethesda, Maryland (www.sans.org/security-resources/idfaq/what-is-a-honeypot/1/9), July 20, 2000.

[5] Gartner, Gartner says the worlds of IT and operational technology are converging, Press Release, Stamford, Connecticut, March 16, 2011.

[6] K. Girtz, B. Mullins, M. Rice and J. Lopez, Practical application layer emulation in industrial control system honeypots, in *Critical Infrastructure Protection X*, M. Rice and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 83–98, 2016.

[7] J. Harrison, Honeypots: The sweet spot in network security, *Computerworld*, November 20, 2003.

[8] Idaho National Laboratory, Control Systems Cyber Security: Defense in Depth Strategies, External Report INL/EXT-06-11478, Idaho Falls, Idaho, 2006.

[9] Ixia, IxLoad Application Replay, Data Sheet, Document No. 915-1744-01 Rev D, Calabasas, California, 2013.

[10] Ixia, IxLoad Overview – Converged Multiplay Service Validation, Data Sheet, Document No. 915-1710-01-2161 Rev S, Calabasas, California, 2016.

[11] I. Mokube and M. Adams, Honeypots: Concepts, approaches and challenges, *Proceedings of the Forty-Fifth Annual ACM Southeast Regional Conference*, pp. 321–326, 2007.

[12] NetLoad, Test Traffic Solution, Danville, California (www.netloadinc.com/manuals/NetLoadInc_Products.pdf), 2014.

[13] NetLoad, Stateful Traffic Mix Tester Solutions, Danville, California (www.netloadinc.com/manuals/NetLoad_Inc_Brief.pdf), 2015.

[14] NetLoad, User Guide for NetLoad Product Series (Revision 8.9), Danville, California (netloadinc.com/manuals/NetLoadInc._Startup_Guide.pdf), 2016.

[15] Ostinato, Ostinato User Guide (www.gitbook.com/book/pstavirs/ostinato-user-guide/details), 2016.

[16] N. Provos, A virtual honeypot framework, *Proceedings of the Thirteenth USENIX Security Symposium*, article no. 1, 2004.

[17] W. Shaw, *Cybersecurity for SCADA Systems*, PennWell Corporation, Tulsa, Oklahoma, 2006.

[18] Siemens, APOGEE Building Level Network on TCP/IP, Technical Specification Sheet (Revision 1), Document No. 149-967, Buffalo Grove, Illinois, 2003.

[19] Siemens, Siemens 2011 APOGEE Brochure (Revision 8), Document No. 153-301 P10, Buffalo Grove, Illinois, 2011.

[20] Siemens, Siemens APOGEE Scalable Brochure, Document No. 611-056, Buffalo Grove, Illinois, 2012.

[21] S. Smith, Catching Flies: A Guide to the Various Flavors of Honeypots, InfoSec Reading Room, SANS Institute, Bethesda, Maryland, 2016.

[22] SolarWinds, WAN Killer: Network Traffic Generator with Engineer's Toolset, Austin, Texas (`www.solarwinds.com/engineers-toolset/wan-killer`), 2016.

[23] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams and A. Hahn, Guide to Industrial Control Systems (ICS) Security, NIST Special Publication 800-82, Revision 2, National Institute of Standards and Technology, Gaithersburg, Maryland, 2015.

[24] P. Warner, Automatic Configuration of Programmable Logic Controller Emulators, M.S. Thesis, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2015.

[25] K. Wilhoit, The SCADA That Didn't Cry Wolf – Who's Really Attacking Your ICS Equipment? (Part 2), Research Paper, Trend Micro, Cupertino, California, 2013.

[26] M. Winn, M. Rice, S. Dunlap, J. Lopez and B. Mullins, Constructing cost-effective and targetable industrial control system honeypots for production networks, *International Journal of Critical Infrastructure Protection*, vol. 10, pp. 47–58, 2015.

Chapter 12

CHALLENGES TO AUTOMATING SECURITY CONFIGURATION CHECKLISTS IN MANUFACTURING ENVIRONMENTS

Joshua Lubell and Timothy Zimmerman

Abstract Information technology is essential to today's manufacturing systems, but it makes them more vulnerable to cyber security threats than ever before. This chapter discusses the challenges to developing automatable configuration checklists for manufacturing environments using the Security Content Automation Protocol (SCAP) family of standards. Increased use of SCAP in manufacturing environments could reduce security vulnerabilities and the likelihood of damaging cyber attacks on manufacturing systems. However, complex relationships and dependencies within and between checklist rules, checking instructions and software result in platform fragmentation. Platform fragmentation makes it difficult to reuse or repurpose existing SCAP-expressed checklist content. Recent research and technological developments can be leveraged to yield potentially promising approaches for mitigating platform fragmentation and improving reuse.

Keywords: Manufacturing environments, control systems, security, checklists

1. Introduction

Information technology is essential to today's manufacturing systems. Microprocessors, software, data repositories, networking protocols and the Internet improve product quality, increase throughput and reduce production costs. But the increased reliance on information technology makes manufacturing environments more vulnerable to cyber security threats than ever before. A cyber attack can cause a loss of confidentiality, data corruption and costly downtime. An additional consequence for manufacturing systems – as with Stuxnet [3] and the German steel mill cyber attacks of 2014 [16] – is extensive physical damage to equipment and the surrounding environment. Physical and environmental

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 225–241, 2017.
https://doi.org/10.1007/978-3-319-70395-4_12

damage can, in turn, result in personal injury or death as well as large financial losses.

Proper security configurations reduce the likelihood of a cyber attack compromising a system, with the added benefit of protecting against incidents caused by (non-malicious) human errors. Configuration settings such as password and remote access policies, authorizing only the minimum access needed by users or processes to accomplish assigned tasks (least privilege principle) and restricting communications between subsystems and external systems (boundary protection principle) help improve the security posture [6]. The least privilege and boundary protection principles are included in the hundreds of security controls specified in National Institute of Standards and Technology (NIST) Special Publication 800-53, Recommended Security Controls for Federal Information Systems and Organizations [22].

The U.S. Department of Homeland Security 2016 Industrial Control Systems Assessment Summary Report [21] underscores the importance of proper configurations of control systems in manufacturing and industrial environments. According to the report, among the most common areas of weakness are the lack of adherence to the least privilege and boundary protection principles, poor authentication mechanisms and weak passwords. The right configuration settings can mitigate these vulnerabilities.

The U.S. National Checklist Program [31] defines a security configuration checklist – also referred to as a hardening guide or security benchmark – as "a series of instructions or procedures for configuring an information technology product to a particular operational environment, for verifying that the product has been configured properly and/or for identifying unauthorized changes to the product." Because such checklists provide guidance that is both concrete and actionable, they are especially useful to smaller companies and organizations that lack in-house cyber security expertise. A checklist may be automatable so that it can be used as digital input to a software tool that reports how well a system is configured with respect to the checklist's guidance. Automated security configuration checking is highly desirable because it reduces the costs of maintaining security and documenting the extent of compliance with a security policy.

To facilitate automation, a checklist should contain structured, computer-interpretable digital data. Furthermore, the tool that interprets the checklist should implement the data models used in the checklist. In other words, in order for a tool to automate a checklist, the tool should parse and process the data structures used in the checklist's digital representation. In this manner, the structured information in an automatable checklist serves as a "digital thread" [10] for cyber security [17] that integrates configuration guidance with system-specific settings. Additionally, the digital thread can associate a configuration setting with a specific security control or requirement, thereby linking the setting to its rationale. An automatable checklist with data structures that provide computer-interpretable configuration instructions and traceability to security requirements is extremely useful. Such a checklist enables a longer and broader cyber security digital thread than a checklist that lacks these items.

This chapter discusses the challenges to developing automatable checklists for manufacturing environments using the Security Content Automation Protocol (SCAP – pronounced "ess-cap") family of standards [30, 32]. The standards specify a means for representing checklist information in a manner that an SCAP-conforming software tool can determine how well the system configuration complies with the checklist. SCAP is widely used in government [24] as well as by private sector enterprises to manage configuration compliance of common operating systems and software applications. Greater use of SCAP in manufacturing environments can reduce security vulnerabilities and the likelihood of damaging cyber attacks on manufacturing systems.

2. Manufacturing Environments

Modern manufacturing environments use industrial control systems to safely and reliably operate industrial processes. At the front line of these operations are programmable logic controllers (PLCs), which are computers developed specifically to monitor and control industrial processes [19]. Early generations of programmable logic controllers were intended to operate in isolated environments with custom operating systems and proprietary communications protocols. In contrast, modern programmable logic controllers incorporate information technologies such as TCP/IP and Ethernet for communications, USB ports for file transfers and commodity operating systems for efficiency, flexibility and convenience. The technologies enable greater visibility of manufacturing processes, the incorporation of real-time manufacturing data in corporate business systems, interoperability with existing networking infrastructures and remote monitoring capabilities [33]. However, the technologies also expose programmable logic controllers to greater cyber security risks.

Programmable logic controllers must be able to audit their configurations in order to detect potential vulnerabilities before they are exploited as attack vectors. Many attack vectors are introduced by enabled-by-default programmable logic controller features and services (e.g., embedded web servers, terminal servers and remote access servers). These features and services may only be detected after a thorough review of cyber-security-hardening guidance from vendors or discovery through active-scanning techniques employed by penetration testers.

The integration of information technologies directly in industrial control systems and manufacturing environments is now a normal occurrence. This is exemplified in the NIST ICS Cybersecurity Testbed [4, 41]. The testbed contains two robotic arms that emulate a material handling application; the robots are integrated into simulated manufacturing machines to perform repetitive tasks normally performed by a human operator (e.g., insert raw parts, remove finished parts, operate machine guarding and start and stop the machining cycle). The manufacturing machines communicate with the robot controllers via TCP/IP

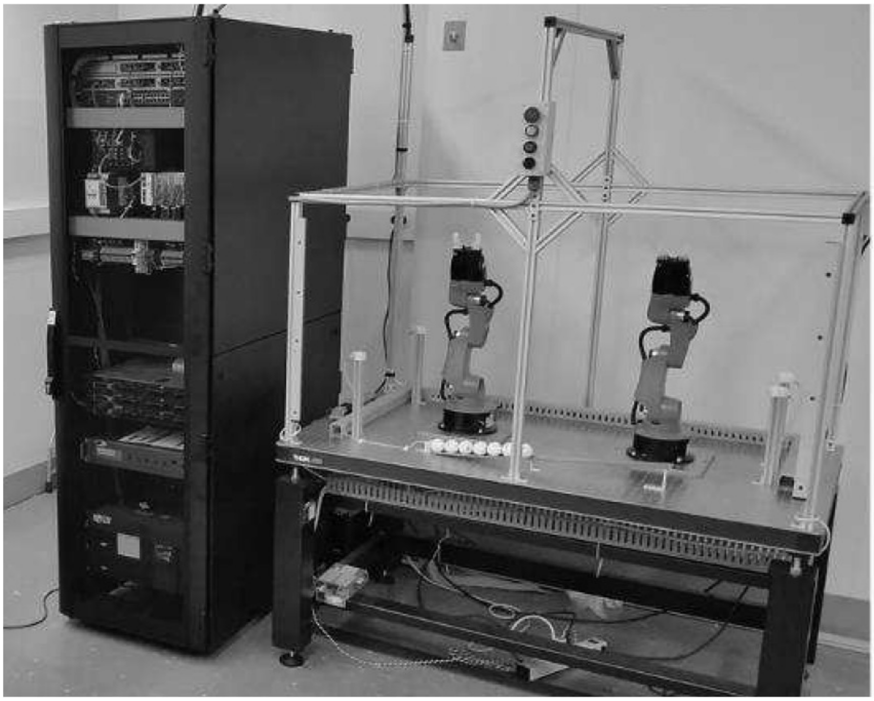
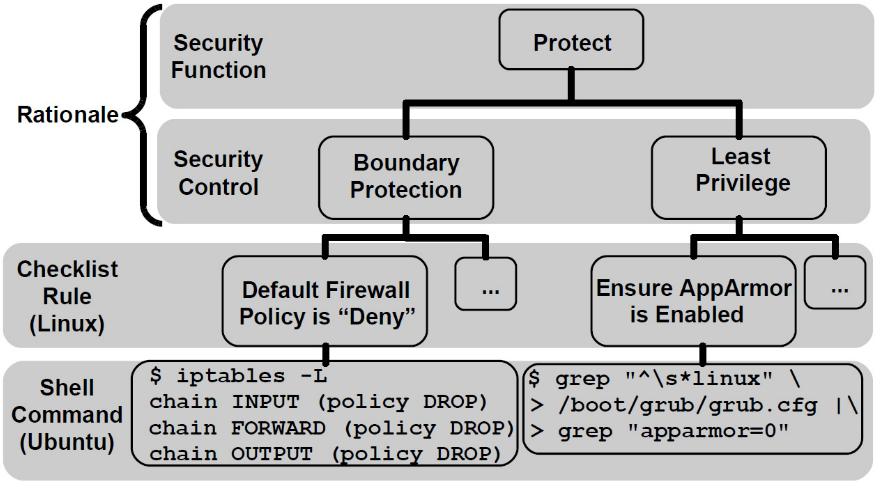


Figure 1. Robotic portion of the NIST ICS Cybersecurity Testbed.

and Ethernet to coordinate the loading and unloading of parts. Figure 1 shows an image of the robotic portion of the NIST testbed.

Software executing in dedicated external controllers controls each robot. The controllers, which are high-performance servers typically found in information technology environments, run the Robot Operating System (ROS) on top of Ubuntu Linux, enabling the robots to collaborate in performing machine tending tasks. Ubuntu is the most common deployment environment for ROS, a software framework that is widely used in robotics research projects and increasingly in commercial robotic applications [7]. ROS-Industrial, a variant of ROS tailored for commercial applications, accelerates ROS deployment by augmenting its advanced manipulation capabilities with better support for reliability and safety [20, 34].

A motivating example is used to demonstrate the relationships between rules, security controls and a high-level security objective for an Ubuntu Linux server running ROS. Figure 2 shows two rules: one rule stipulates that a firewall should be set by default to deny all traffic (with all exceptions explicitly provided) and the other states that an AppArmor Linux kernel enhancement [1] should be enabled to restrict program access to system resources. AppArmor is commonly used in Ubuntu systems with stringent security requirements and



```
$ iptables -L
chain INPUT (policy DROP)
chain FORWARD (policy DROP)
chain OUTPUT (policy DROP)
```

```
$ grep "^\s*linux" \
> /boot/grub/grub.cfg |\
> grep "apparmor=0"
```

Figure 2. Checklist example.

its namespaced access control mechanism is a good match for the ROS data model [39]. The first rule supports boundary protection while the second rule supports least privilege. Both these security controls, in turn, support the high-level "protect" security function specified in the Cybersecurity Framework [23] – a methodology for managing cyber risk, describing an organization's current security posture and target state, and communicating and assessing progress toward meeting goals. Below each rule are shell commands for checking compliance.

Each level in Figure 2 is less general and more implementation-specific than the level above it. The top two levels describe security objectives and are independent of system-specific and implementation details. They provide the rationale for the rules in the checklist rule level. The checklist rule level applies to Linux systems, but it does not assume a particular Linux distribution. The shell command level applies only to Linux systems with the `iptables` package (pre-installed by default in Ubuntu distributions); `iptables` is an application that can be used to configure a Linux kernel firewall.

A corollary to the observation regarding levels and generality is that information in the higher levels is more reusable than information in the lower levels. For example, many rules in a wide variety of checklists support the least privilege security control. This is because least privilege is a universal principle that applies to many deployment situations. The AppArmor rule, however, is more specific. It applies only to Linux systems where the security benefit of AppArmor outweighs the convenience of programs having less-restrictive access to system resources. However, several Linux systems have specialized security limited functionality (SSLF) requirements [31], including the robot controller server that runs ROS. These systems have especially stringent security requirements because of the threats they face and the potential consequences of incidents. Thus, the rule is reusable in many specialized security limited functionality contexts. However, the shell command that implements the rule is less reusable because it assumes the presence of `iptables` instead of an alternative firewall configuration application.

3. SCAP Background

An automated checklist has limited value if it is hardwired to a particular configuration tool or scripting language. Standards for representing rules, system settings, vulnerabilities, platforms and other relevant information enable checklists to be interoperable. Interoperable checklists can be used with any standards-compliant tool. Interoperability standards save checklist developers the trouble of having to learn multiple proprietary formats and lower the barriers to automated configuration checking. To address the need for interoperability, the cyber security research, development and user communities have created several Extensible Markup Language (XML) [40] data representation and exchange standards for software weaknesses and vulnerabilities, naming conventions, system state, configuration checklists, asset identification and severity measurement of software and configuration issues [18]. SCAP provides the recommended practices for using these standards together [30, 32].

SCAP is commonly used to automate security configuration compliance checking, which is the focus of this research. The following SCAP languages and taxonomies are especially relevant to the example in Figure 2 and the upcoming discussion:

- **Extensible Configuration Checklist Description Format (XCCDF):** This language is used to express security checklists, benchmarks and other configuration recommendations. XCCDF can represent the highly structured data needed for automated configuration checking as well as the semi-structured data needed to produce human-readable documentation of a checklist and the results of checking a system configuration.

- **Common Platform Enumeration (CPE):** This naming scheme provides unique identifiers for hardware, operating systems and applications.

- **Common Configuration Enumeration (CCE):** This registry maintains unique identifiers for operating system and software security configurations. SCAP checklists can use mappings from common configuration enumeration values to taxonomies of security principles or business objectives, providing traceability from configuration settings to higher-level requirements.

- **Open Vulnerability and Assessment Language (OVAL):** This language is used to express system configuration information, assess machine state and report assessment results. OVAL is widely used in the cyber security community as part of SCAP as well as with other standards [13].

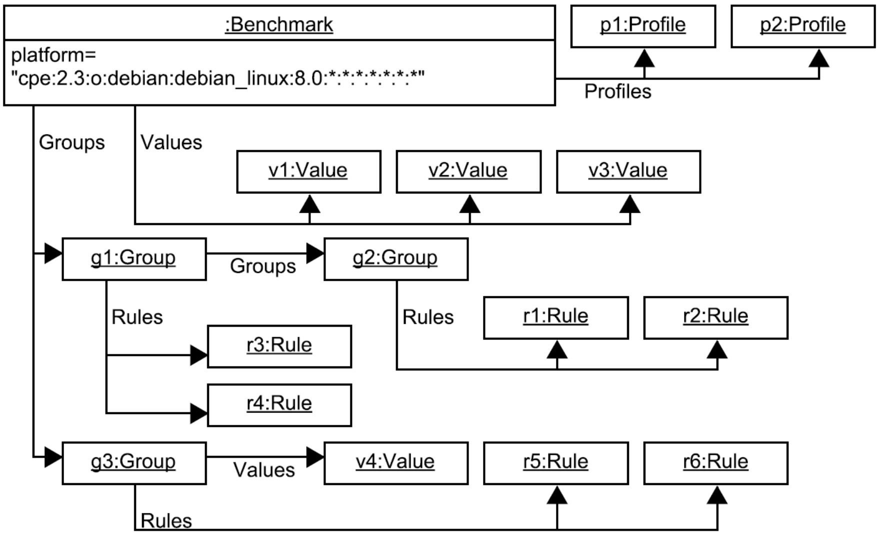


Figure 3. Example of an XCCDF benchmark XML document.

> Many hardware and software vendors produce OVAL content, which they make available to their customers directly and through third-party online repositories. The OVAL data model is versatile and complex; interested readers are referred to [28] for details about the data model

The XCCDF data model [38] represents a checklist document as a `Benchmark` object. A `Benchmark` is a collection of `Rule`, `Value` and `Group` objects. A `Rule` specifies a single item to check, such as the default setting of a firewall. A `Rule` also specifies how the checking should be done, such as with an implementation-specific scripting language or with the OVAL standard. A `Value` represents a named quantity that can be used in a rule and is tailored to a particular configuration scenario. A `Group` collects `Rule`, `Value` and other `Group` objects into an aggregation that is meaningful to a checklist user (e.g., a collection of firewall configuration settings). An XCCDF `Benchmark` also contains one or more `Profile` objects. A `Profile` is a named collection that references `Group`, `Rule` and `Value` objects. For example, an XCCDF Ubuntu checklist may have three profiles: one for single-user desktop systems, another for file servers and a third for specialized security limited functionality systems.

Figure 3 shows an example of an XCCDF checklist, which is based on an example from the XCCDF specification [38]. The Unified Modeling Language (UML) [26] object notation is employed. The checklist applies to the Debian Linux distribution version 8.0, as indicated by the common platform enumeration value assigned to the `platform` attribute of the `Benchmark` object. The checklist has two profiles, p1 and p2, each of which references a subset (omitted from Figure 3 for simplicity) of groups g1, g2 and g3 and values v1, v2 and v3.

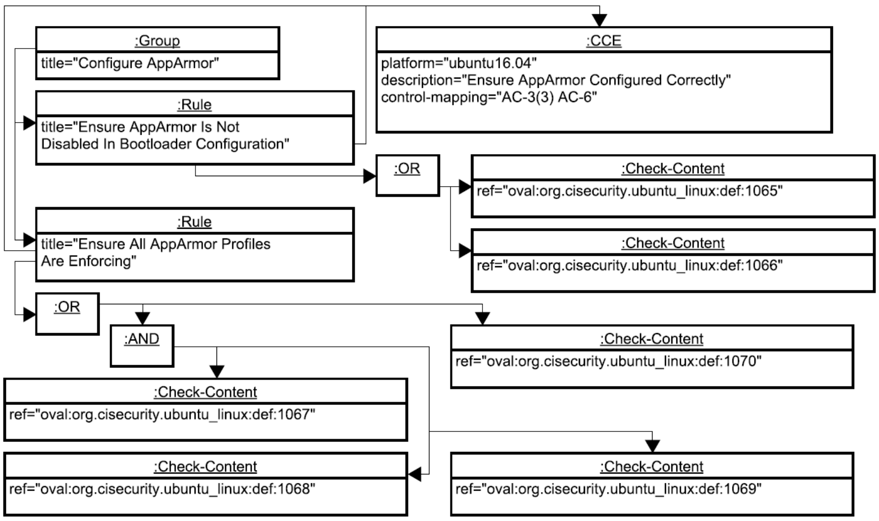

:Group
title="Configure AppArmor"
:Rule
title="Ensure AppArmor Is Not Disabled In Bootloader Configuration"
:CCE
platform="ubuntu16.04"
description="Ensure AppArmor Configured Correctly"
control-mapping="AC-3(3) AC-6"
:OR
:Check-Content
ref="oval:org.cisecurity.ubuntu_linux:def:1065"
:Check-Content
ref="oval:org.cisecurity.ubuntu_linux:def:1066"
:Rule
title="Ensure All AppArmor Profiles Are Enforcing"
:OR
:AND
:Check-Content
ref="oval:org.cisecurity.ubuntu_linux:def:1070"
:Check-Content
ref="oval:org.cisecurity.ubuntu_linux:def:1067"
:Check-Content
ref="oval:org.cisecurity.ubuntu_linux:def:1068"
:Check-Content
ref="oval:org.cisecurity.ubuntu_linux:def:1069"

Figure 4. Example AppArmor rule in XCCDF.

In the figure, g1 aggregates rules r3 and r4; g2 aggregates rules r1 and r2; and g3 aggregates value v4 and rules r5 and r6.

The UML object model in Figure 4 shows how XCCDF is used to express the rules for checking that AppArmor is configured correctly in the Ubuntu scenario shown in Figure 2. Since the actual XCCDF rule representation would have more objects and would be harder to understand, Figure 4 shows a simplified version of the rule representation. Interested readers are referred to [5, 27] for details about an actual XCCDF XML representation. The `CCE` object in the upper-right corner of Figure 4 maps to two NIST Special Publication 800-53 security controls from the Access Control (AC) family, AC-3(3) and AC-6, both of which support the least privilege principle. The two `Rule` objects that support the `CCE` each reference Boolean expressions involving `Check-Content` objects. Each `Check-Content` object references a specific Ubuntu Linux OVAL definition in an external location. SCAP-conforming configuration scanner tools must support OVAL as a checking system. Support for other checking systems, such as those based on Linux shell command languages, is optional. To facilitate interoperability, the use of OVAL in checklists is preferable to non-SCAP checking systems.

The example in Figure 4 also illustrates the complexity of SCAP content. Each rule requires multiple OVAL definitions. Each OVAL definition, in turn, requires additional OVAL objects that are not shown in Figure 4.

4. SCAP Reuse in Manufacturing Environments

A major benefit is that an SCAP-expressed checklist can be used with multiple software tools and can be presented in multiple ways to users depending on

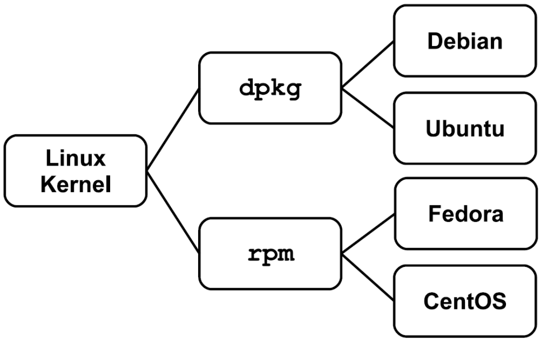


Figure 5. Shared components of Linux distributions.

their desire for details or need to know. Furthermore, XCCDF enables checklist developers to modularize content into groups and profiles, and enables checklist users to create tailoring files based on profiles. These capabilities, along with various online XCCDF and OVAL repositories, benefit SCAP checklist developers and users.

However, SCAP is not as helpful at facilitating the reuse of content applicable to multiple platforms. Consider, for example, the firewall rule in Figure 2, which uses `iptables` to check that the default policy is configured properly. The rule assumes the presence of `iptables` and, therefore, requires another rule to verify that `iptables` is installed. For Ubuntu Linux, the shell command `dpkg -s iptables` could implement such a rule. The `dpkg` package manager `-s` command option determines the status of a package (i.e., whether or not it is installed). However, `dpkg` is not universal across Linux distributions. Linux distributions based on the Debian Linux distribution, such as Ubuntu, use `dpkg`. Other Linux distributions use different package managers. For example, Fedora and CentOS use `rpm` for package management. Figure 5 shows the relationships between the Debian, Ubuntu, Fedora and CentOS Linux distributions in terms of their shared kernel and package manager components.

This example illustrates the problem of platform fragmentation. Platform fragmentation occurs when the same operating system, software application and/or hardware component are bundled by multiple entities, with each bundler providing different customizations [35]. A real-world example that demonstrates how Linux platform fragmentation complicates SCAP checklist development is the SCAP Security Guide (SSG) Project [27]. The project goal is to produce SCAP content (security guides) for a variety of Linux platforms. The SCAP Security Guide developers deal with platform fragmentation by splitting their source code into pieces that can be shared by multiple security guides versus pieces that are specific to a security guide. Building security guides from the source requires running scripts that perform XML transformations, macro substitutions and merging of source files into larger files. The build process is complicated and requires the SCAP Security Guide developers to understand not only SCAP, but also the one-off manner in which the source files are organized and structured [29].

The SCAP platform fragmentation problem is not just limited to Linux. For example, consider the effort needed to develop a checklist for a programmable logic controller. A programmable logic controller may include an operating system (developed by a third party but customized by the vendor) in addition to automation software. Each of these components may be vulnerable to cyber attacks. The NIST ICS Cybersecurity Testbed has a Beckhoff CX9020 programmable logic controller [2] that runs the Windows Embedded Compact operating system (which Beckhoff licenses from Microsoft and has customized) and Beckhoff's TwinCAT automation software. It is actually less challenging to automate the security configuration of a Beckhoff programmable logic controller than that of many other programmable logic controllers with hardware-optimized, vendor-accessible-only operating systems. For these programmable logic controllers, there is no way for even an advanced user to deploy an automated configuration compliance checker. Indeed, only the vendor can alter the operating-system-level configuration settings.

A security professional tasked with developing an SCAP checklist for a Beckhoff CX9020 programmable logic controller might begin by reviewing the ICS-CERT Advisory ICSA-16-278-02 [12]. This document provides guidance for mitigating the vulnerabilities associated with the Windows Embedded Compact operating system and TwinCAT components of the programmable logic controller. Since it is easier to reuse existing SCAP XML content than to create new content from scratch, the security professional could search for XCCDF and OVAL content that is applicable to Windows Embedded Compact and TwinCAT. Unfortunately for the security professional, no registered common platform enumeration identifiers exist for Windows Embedded Compact and TwinCAT. However, SCAP content may exist for a third-party software library used by TwinCAT and Windows Embedded Compact has a number of components in common with Windows 7 (for which a great deal of SCAP content exists). In such a situation, it would be beneficial if the security professional could determine which, if any, of the existing SCAP content could be easily repurposed to automate the ICSA-16-278-02 guidance. Unfortunately, the existing SCAP content lacks the metadata needed to make such a determination.

5. Relevant Research and Standards

In addition to the SCAP Security Guide Project, other recent and ongoing research, implementation and standardization efforts have proposed solutions for coping with fragmentation and promoting reuse. The solutions can be categorized as employing: (i) information modeling; (ii) document-focused; (iii) centralized; or (iv) content-focused approaches.

Fitzgerald and Foley [8] have analyzed SCAP content from several repositories, classified the types of inconsistencies that create ambiguity and impede reuse and provided examples of implicit relationships. Their analysis focused on the OVAL language and common platform enumeration and common configuration enumeration taxonomies, but did not include XCCDF content. They then developed an SCAP ontology employing semantic threat graphs and demonstrated how the SCAP ontology addresses inconsistency challenges by making implicit relationships explicit.

Other researchers have pushed the envelope of SCAP deployment by developing checklists for platforms beyond the usual SCAP realm of desktop operating systems, Internet browsers and office applications. Hlyne et al. [11] have developed a configuration checklist for Cisco routers, which uses OVAL content developed by Cisco and is deployed using the jOval SCAP configuration scanning tool. Kuo and Yang [15] have created an SCAP configuration checklist for Android devices and a configuration management tool using the jOval scanning engine; the tool runs on a server and performs remote scanning and remediation of misconfigurations. Kuo and Yang have used as their information source prose text (without SCAP content) benchmark documentation from the Center for Internet Security (CIS) [9]. CIS benchmarks [5] are available for a variety of operating systems, software environments and network devices. Although many CIS benchmarks are in prose form and not expressed using SCAP, an increasing number are now available to CIS members in the XCCDF and OVAL formats.

Vecchiato et al. [35] have studied configuration assessment data from more than 500 Android smartphones and have found several recurring misconfigurations, the most common being weak passwords and overly-permissive network settings. Android device configuration compliance is an interesting SCAP use case because its platform fragmentation challenges parallel those encountered in manufacturing environments. In comparing Android with common desktop and laptop operating systems, Vecchiato and colleagues characterize Android smartphones as being more personalized to consumer preferences and having lesser capabilities due to their reduced size and other physical constraints. These characteristics are similar to those in manufacturing environments, where hardware and information technology capabilities have to meet requirements unique to the production scenarios and environmental conditions. Furthermore, as in the case of the programmable logic controller example in Section 4, smartphone vendors customize Android devices with their own applications and carrier-specific settings. As a result, Android smartphone vulnerabilities and configuration issues can be hardware-vendor-dependent.

Vecchiato et al. [35] have also proposed an intriguing approach for reducing the fragmentation problem – that vendors and researchers work together to transition some Android device capabilities currently available only through vendors to services available through Google Play. The anticipated result of this approach is that vendors could leverage Google Play's automatic update mechanism, enabling third-party software developers and end users to receive new Android features and patches without having to wait for less-frequent Android updates from their carriers. The approach of Vecchiato and colleagues appears to be appealing in manufacturing environments because it addresses a significant obstacle to automating configuration checking of industrial control devices such as programmable logic controllers – the lack of software interfaces for third-party access to the underlying operating system information. Of course, vendor-provided tools employed to program the devices may enable users to issue queries required to support the auditing of programmable logic controllers, but it is currently not possible to use these tools to conduct automated audits.

An adaptation of the idea of Vecchiato and colleagues could help mitigate the fragmentation problem in manufacturing environments. Specifically, vendors of programmable logic controllers, switches and routers could make their firmware, operating system and other software updates available through centralized trusted digital distribution services. However, unlike Android smartphones, automatically updating industrial control devices raises significant concerns. The availability of the devices and the safety of their operations are paramount. Enabling a third-party developer to update a device automatically without end-user intervention could have catastrophic consequences, especially if the update occurs during production or software bugs are present in the update. However, with the right modifications to address industrial control system availability and safety requirements, an adapted approach might be acceptable. Indeed, the increasing use of commodity operating systems in programmable logic controllers and other devices provides new opportunities for developers to implement automatic software updates as part of centralized services.

Software identification (SWID) tags [37], an international standard for describing software products, offer another potential solution to platform fragmentation. SWID tags use an XML format that allows for better-structured and information-richer content than provided by a common platform enumeration identifier. The common platform enumeration naming scheme is syntactic and does not provide explicit semantics, which leads to ambiguity [8]. In contrast, SWID tags can explicitly represent the unique identifier of a product as well as information about its versioning scheme, patch level, relationships to other software and a host of other useful metadata. Because they enable better software inventory management, SWID tags can also help detect system misconfigurations. Indeed, current SCAP recommended practices provide guidance for replacing common platform enumeration identifiers in SCAP content with SWID tags [36].

Another standard that could promote the reuse of SCAP content is the Darwin Information Typing Architecture (DITA) [25]. As the SCAP Security Guide Project's complicated build system illustrates, producing an SCAP checklist from a collection of components poses XML publishing and content management problems. However, as discussed in Section 4, the source code version control systems that projects such as the SCAP Security Guide typically use are file-based and inadequate for managing relationships between objects within a source XML file or relationships between files. Multiple publications (checklists) can share the same XCCDF or OVAL component. Additionally, a publication can use the same component in more than one context. Kimber [14]

Table 1. Research and development approaches, objectives and proposed solutions.

Approach	**Objective**	**Solutions**
Information Modeling	Mitigate platform fragmentation by improving the ability of SCAP to represent platform dependencies and relationships within and between XCCDF rules and OVAL definitions.	Fitzgerald and Foley [8]; SWID tags [37].
Document-Focused	Use XML-based methods to make SCAP authoring and content reuse easier.	DITA [14, 25]; SSG *ad hoc* build system [29].
Centralized	Reduce the likelihood of misconfigurations by centralizing software distribution and updates. Addresses the root cause of platform fragmentation instead of putting the onus on checklist authors. Requires coordination and trust between stakeholders and tailoring to meet the stringent operational and safety requirements of manufacturing environments.	Vecchiato et al. [35].
Content-Focused	Develop more XCCDF checklists, particularly for platforms where SCAP content is in short supply.	SSG [27]; Hlyne et al. [11]; Kuo and Yang [15]; CIS benchmarks [5].

has implemented a content management and publishing system that combines the capabilities of the Darwin Information Typing Architecture with the Git version control system to generate publications from a collection of interrelated version-controlled XML components. Such a standards-based strategy, if feasible for the SCAP Security Guide Project, would be less *ad hoc* and brittle than the current approach, which uses a mix of scripts, makefiles and manual searches within files [29].

Table 1 summarizes the objectives and the solutions of the research and development approaches discussed above.

6. Conclusions

This chapter has investigated the challenges to reusing existing SCAP content for checking configuration compliance of information technology components in manufacturing environments. An illustrative example using an Ubuntu Linux configuration checklist scenario demonstrates that platform fragmentation complicates the reuse of SCAP content. The same platform fragmentation exists in the industrial control system and Android device domains as well. A review and classification of related research and development approaches and the relevant standards reveal promising solutions that ameliorate platform fragmentation and encourage more SCAP deployments.

The need for cyber security solutions in manufacturing environments is growing. Although platform fragmentation is a barrier to SCAP deployment in the manufacturing sector, other areas – most notably Android mobile devices – share a number of the same issues. Recent and ongoing research and development results could lead to concrete gains in SCAP adoption in the manufacturing sector and in other areas, especially if the stakeholder communities work together and learn from each other.

Disclaimer

This chapter is a contribution of the National Institute of Standards and Technology (NIST). Certain commercial and third-party products and services are identified in this chapter to enhance understanding. Such identification does not imply any recommendation or endorsement by NIST, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

References

[1] M. Bauer, Paranoid Penguin: AppArmor in Ubuntu 9, *Linux Journal*, issue 185, September 1, 2009.

[2] Beckhoff Automation, Manual CX9020, Embedded PC, Version 1.8, Verl, Germany (`download.beckhoff.com/download/document/ipc/embedded-pc/embedded-pc-cx/cx9020_hwen.pdf`), 2017.

[3] E. Byres, A. Ginter and J. Langill, How Stuxnet Spreads – A Study of Infection Paths in Best Practice Systems, Version 1.0, Tofino Security, Lantzville, Canada, 2011.

[4] R. Candell, T. Zimmerman and K. Stouffer, An Industrial Control System Cybersecurity Performance Testbed, NISTIR 8089, National Institute of Standards and Technology, Gaithersburg, Maryland, 2015.

[5] Center for Internet Security, CIS Benchmarks, East Greenbush, New York (`benchmarks.cisecurity.org`), 2017.

[6] A. Creery and E. Byres, Industrial cybersecurity for power system and SCADA networks, *Proceedings of the Industry Applications Society Fifty-Second Annual Petroleum and Chemical Industry Conference*, pp. 303–309, 2005.

[7] C. Fairchild and T. Harman, *ROS Robotics by Example*, Packt Publishing, Birmingham, United Kingdom, 2016.

[8] W. Fitzgerald and S. Foley, Avoiding inconsistencies in the Security Content Automation Protocol, *Proceedings of the IEEE Conference on Communications and Network Security*, pp. 454–461, 2013.

[9] R. Fritz, CIS Google Android, Android 4 Benchmark V1.0.0, Center for Internet Security, East Greenbush, New York (`learn.cisecurity.org/benchmarks`), 2012.

[10] T. Hedberg, J. Lubell, L. Fischer, L. Maggiano and A. Feeney, Testing the digital thread in support of model-based manufacturing and inspection, *Journal of Computing and Information Science in Engineering*, vol. 16(2), 2016.

[11] C. Hlyne, P. Zavarsky and S. Butakov, SCAP benchmark for Cisco router security configuration compliance, *Proceedings of the Tenth International Conference on Internet Technology and Secured Transactions*, pp. 270–276, 2015.

[12] Industrial Control Systems Cyber Emergency Response Team (ICS-CERT), Advisory (ICSA-16-278-02), Beckhoff Embedded PC Images and TwinCAT Components Vulnerabilities, U.S. Department of Homeland Security, Washington, DC (`ics-cert.us-cert.gov/advisories/ICSA-16-278-02`), January 5, 2014.

[13] P. Kampanakis, Security automation and threat information-sharing options, *IEEE Security and Privacy*, vol. 12(5), pp. 42–51, 2014.

[14] E. Kimber, Hyperdocument authoring link management using Git and XQuery in service of an abstract hyperdocument management model applied to DITA hyperdocuments, *Proceedings of Balisage: The Markup Conference*, vol. 15, 2015.

[15] C. Kuo and C. Yang, Security design for configuration management of Android devices, *Proceedings of the Thirty-Ninth Annual Computer Software and Applications Conference*, vol. 3, pp. 249–254, 2015.

[16] R. Lee, M. Assante and T. Conway, German Steel Mill Cyber Attack, ICS CP/PE (Cyber-Physical or Process Effects), Case Study Paper, SANS Institute, Bethesda, Maryland (`ics.sans.org/media/ICS-CPPE-case-Study-2-German-Steelworks_Facility.pdf`), 2014.

[17] J. Lubell, Extending the cybersecurity digital thread with XForms, *Proceedings of Balisage: The Markup Conference*, vol. 15, 2015.

[18] G. McGuire and E. Reid, The State of Security Automation Standards – 2011, A Survey, MP110439, MITRE Corporation, Bedford, Massachusetts (`www.mitre.org/sites/default/files/pdf/11_3822.pdf`), 2011.

[19] S. McLaughlin, C. Konstantinou, X. Wang, L. Davi, A. Sadeghi, M. Maniatakos and R. Karri, The cybersecurity landscape in industrial control systems, *Proceedings of the IEEE*, vol. 104(5), pp. 1039–1057, 2016.

[20] M. Munaro, C. Lewis, D. Chambers, P. Hvass and E. Menegatti, RGB-D human detection and tracking for industrial environments, *Proceedings of the Thirteenth Conference on Intelligent Autonomous Systems*, pp. 1655–1668, 2014.

[21] National Cybersecurity and Communications Integration Center/Industrial Control Systems Cyber Emergency Response Team, NCCIC/ICS-CERT Industrial Control Systems Assessment Summary Report, FY 2015, U.S. Department of Homeland Security, Washington, DC, 2016.

[22] National Institute of Standards and Technology, Security and Privacy Controls for Federal Information Systems and Organizations, NIST Special Publication 800-53, Revision 4, Gaithersburg, Maryland, 2013.

[23] National Institute of Standards and Technology, Framework for Improving Critical Infrastructure Cybersecurity, Version 1.0, Gaithersburg, Maryland, 2014.

[24] National Institute of Standards and Technology, The United States Government Configuration Baseline (USGCB), Gaithersburg, Maryland (`usgcb.nist.gov`), 2017.

[25] OASIS, Darwin Information Typing Architecture (DITA) Version 1.3 Part 0: Overview (Plus Errata 01), OASIS Standard (Incorporating Approved Errata), Burlington, Massachusetts, 2016.

[26] Object Management Group, OMG Unified Modeling Language (OMG UML), Version 2.5, Document No. Formal/2015-03-01, Needham, Massachusetts, 2015.

[27] OpenSCAP, SCAP Security Guide (`www.open-scap.org/security-policies/scap-security-guide`), 2017.

[28] OVAL, OVAL Documentation (`ovalproject.github.io`), 2017.

[29] M. Preisler, Contributing to SCAP Security Guide – Part 1 (`martin.preisler.me/2016/10/contributing-to-scap-security-guide-part-1`), October 28, 2016.

[30] S. Quinn, K. Scarfone and D. Waltermire, Guide to Adopting and Using the Security Content Automation Protocol (SCAP), Version 1.2 (Draft), NIST Special Publication 800-117, Revision 1, National Institute of Standards and Technology, Gaithersburg, Maryland, 2012.

[31] S. Quinn, M. Souppaya, M. Cook and K. Scarfone, National Checklist Program for IT Products – Guidelines for Checklist Users and Developers, NIST Special Publication 800–70, Revision 3, National Institute of Standards and Technology, Gaithersburg, Maryland, 2015.

[32] S. Radack and R. Kuhn, Managing security: The Security Content Automation Protocol, *IT Professional*, vol. 13(1), pp. 9–11, 2011.

[33] A. Sadeghi, C. Wachsmann and M. Waidner, Security and privacy challenges in the industrial Internet of Things, *Proceedings of the Fifty-Second ACM/EDAC/IEEE Design Automation Conference*, 2015.

[34] SwRI Manufacturing Technologies, ROS-Industrial, San Antonio, Texas (`rosindustrial.org`), 2017.

[35] D. Vecchiato, M. Vieira and E. Martins, The perils of Android security configuration, *IEEE Computer*, vol. 49(6), pp. 15–21, 2016.

[36] D. Waltermire and B. Cheikes, Forming Common Platform Enumeration (CPE) Names from Software Identification (SWID) Tags, NISTIR 8085 (Draft), National Institute of Standards and Technology, Gaithersburg, Maryland, 2015.

[37] D. Waltermire, B. Cheikes, L. Feldman and G. Witte, Guidelines for the Creation of Interoperable Software Identification (SWID) Tags, NISTIR 8060, National Institute of Standards and Technology, Gaithersburg, Maryland, 2016.

[38] D. Waltermire, C. Schmidt, K. Scarfone and N. Ziring, Specification for the Extensible Configuration Checklist Description Format (XCCDF), Version 1.2, NISTIR 7275, Revision 4, National Institute of Standards and Technology, Gaithersburg, Maryland, 2012.

[39] R. White, H. Christensen and M. Quigley, SROS: Securing ROS over the wire, in the graph and through the kernel, presented at the *IEEE-RAS International Conference on Humanoid Robots*, 2016.

[40] World Wide Web Consortium, Extensible Markup Language (XML) 1.0 (Fifth Edition), W3C Recommendation, Massachusetts Institute of Technology, Cambridge, Massachusetts (`www.w3.org/TR/REC-xml`), 2008.

[41] T. Zimmerman, Metrics and Key Performance Indicators for Robotic Cybersecurity Performance Analysis, NISTIR 8177, National Institute of Standards and Technology, Gaithersburg, Maryland, 2017.

Chapter 13

CATEGORIZATION OF CYBER TRAINING ENVIRONMENTS FOR INDUSTRIAL CONTROL SYSTEMS

Evan Plumley, Mason Rice, Stephen Dunlap and John Pecarina

Abstract First responders and professionals in hazardous occupations undergo intense training and evaluation to enable them to efficiently and effectively mitigate risk and damage. For example, helicopter pilots train with multiple simulations that increase in complexity before they fly real aircraft. However, in the industrial control systems domain, where incident response professionals help detect, respond and recover from cyber incidents, there is no official categorization of training environments, let alone training regimens. To address this gap, this chapter provides a categorization of industrial control training environments based on realism. Four levels of environments are proposed and mapped to Bloom's Taxonomy. The categorization enables organizations to determine the cyber training environments that best align with their training needs and budgets.

Keywords: Industrial control systems, incident response, training environments

1. Introduction

In the evening of April 17, 2013, an act of arson at a fertilizer plant in West, Texas resulted in an explosion that killed fifteen people, including ten first responders who were fighting the fire [10, 19]. The first responders were not trained to handle a chemical fire and did not fully comprehend the explosive hazards posed by the materials in the plant. The U.S. Emergency Planning and Community Right to Know Act requires all companies to report hazardous chemicals stored in their facilities. However, there are no legal requirements for local first responders to be trained adequately based on the hazard reports.

To avoid disasters like the Texas explosion, it is imperative that incident responders receive training in environments that teach them the required incident response knowledge and skills as well as help assess the extent of the acquired

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 243–271, 2017.
https://doi.org/10.1007/978-3-319-70395-4_13

knowledge and skills. Processes in an industrial environment are managed using industrial control systems that have limited or weak cyber security protections and can pose physical threats to personnel and equipment. The first responders tasked to respond to incidents in industrial control environments must be properly trained and prepared to deal with the complexity and diversity of cyber incidents.

This chapter proposes a framework for identifying and mapping industrial control system cyber incident response knowledge and skills to training environment components. This chapter also proposes a categorization of training environments based on practicality and realism. The categorization assists organizations in determining the cyber training environments that best align with their training needs and budgets.

2. Incident Response Training Environments

The lack of industrial control system defenses is a cause for concern in the community. Contributing factors include cost, system diversity, long lifecycles and organizations that are reluctant to make changes to their operational systems [28]. Generally, the personnel employed at industrial control facilities do not have the skills to properly collect, analyze and examine the command and control traffic in their networks and they find it difficult to differentiate cyber attacks from non-cyber-induced malfunctions [9]. While most organizations are unable to provide high levels of training to their cyber response teams, they can support effective – albeit lower levels of – training that balance organizational goals and budgets. This section discusses the current state of industrial control system training and training environments at U.S. Government, industry and academic entities.

2.1 U.S. Government

The U.S. Department of Homeland Security Department is the U.S. Government entity that provides the vast majority of training programs in the area of industrial control systems. The training efforts primarily focus on the effects of attacks and the development of mitigation strategies as opposed to emergency incident response.

Training courses offered by the Industrial Control System Cyber Emergency Response Team (ICS-CERT) leverage a partial industrial control system to demonstrate exploits and their impacts; the courses culminate in a red and blue team exercise where participants attack and defend an industrial control system [14]. While the training environment is by no means a full-scale system, it incorporates realistic hardware and displays real physical effects. Participants in the advanced course are expected to have prior knowledge of information technology as well as industrial control systems. The advanced course encourages discussion between information technology and industrial control system professionals, which enhances the development of contextual knowledge in both communities. In a real incident response setting, professionals from the two communities must communicate efficiently and effectively to avoid system damage and ensure successful recovery.

However, the exercises and training environment fall short in creating a complete and complex system; moreover, they lean heavily on traditional information technology attacks and defenses instead of focusing intensely on the industrial control domain. Participants must travel to the Idaho National Laboratory facility in Idaho Falls, Idaho for the five-day course. Idaho National Laboratory claims to be able to replicate the control system specifications of any participant, to conduct simultaneous attacks on multiple systems and to perform customized full-scale cyber attacks on an exact replica system [12].

Sandia National Laboratories, which operates as a part of the U.S. Department of Energy National Nuclear Security Administration, is a leader in providing industrial control systems education and training to industry, government and academia [22]. Sandia offers a supervisory control and data acquisition (SCADA) assessment training course that covers the systems and devices in the critical infrastructure and industry. The primary purpose of the course is to provide methodologies and tools for assessing the security of industrial control systems. However, the course is only offered at Sandia's discretion to individuals on an invitation-only basis [22].

Several U.S. research laboratories have leveraged their expertise and capabilities in developing the National SCADA Test Bed [27]. The testbed incorporates full-scale realistic systems and is designed to support research and educational activities. While the emphasis on realism and fidelity ensure that the testbed emulates a real environment, the facility is not currently used to train cyber defenders or incident response teams [27].

2.2 Industry

Industry-provisioned training conducted by vendors is similar to government training; the courses primarily cover industrial control system fundamentals, security audits and assessments of vendor equipment and products. The environments typically provide hands-on laboratory experiences designed to familiarize trainees with programmable logic controllers that are networked to emulate real industrial control environments. The vendors often travel to various locations and offer specific courses to individuals and organizations [3]. The classes are not tailored to train security experts; instead, they are designed for individuals who intend to operate or administer industrial control systems. While these courses are useful, they do not expose potential cyber responders to the complexity of complete systems.

Industry training environments also exist in the form of software simulations of industrial control systems. An example is the LogixPro-500 PLC Simulator that incorporates the RSLogix 500 engineering environment and the ProSim-II programmable process simulation that emulates a programmable logic controller [26]. While this simulation software is not security focused, it provides valuable hands-on experience to novices that advances their understanding of industrial controllers, with the goal of ultimately gaining expertise in industrial control systems security and first response.

2.3 Academia

Academia also uses assorted testbeds for education and research. Mississippi State University maintains a cyber security testbed that emulates a real-world industrial control system with physical processes [16]. Other academic entities have constructed industrial control system environments with fully- or partially-simulated controllers and processes. Reaves et al. [20] have created a testbed that fully simulates an industrial control system environment, including control systems and physical processes. Wertzberger et al. [29] have implemented a training environment that combines real-world control hardware with simulated physical processes and networking that is a step beyond full simulation. While these environments are adequate for introductory training, they lack the complexity of a full-scale system that is required to impart expertise related to emergency response procedures.

The SANS Institute, a cooperative research and education organization, provides training through online courses and in-class and mentored settings around the world. It has created the SANS CyberCity, a scaled model of a small city that incorporates computers, networks, control hardware and embedded devices that emulate infrastructure assets such as a power grid, water system, traffic system and heating, ventilation and air conditioning (HVAC) systems [23]. This model city is used in an on-site course conducted in Austin, Texas, which exposes trainees to industrial control systems and their components, including human-machine interfaces, industrial protocols and data historians. The model city training environment enables trainees to view the physical effects of their cyber actions while operating realistic vendor-supplied technology. The course covers common security flaws and techniques for thwarting attacks on industrial control systems.

3. Bloom's Taxonomy

Educational frameworks have been created to provide insights into the cognitive value acquired from educational activities (e.g., assigned projects and homework). An educational psychologist named Benjamin Bloom (1913-1999) sought to classify educational goals and objectives based on cognitive complexity [11]. The resulting Bloom's Taxonomy, which was revised in 2001, is widely used by teachers and professors for structuring courses that encourage students to learn, apply knowledge and develop an array of cognitive skills.

Bloom's revised taxonomy comprises the six major categories of educational goals listed in Table 1; the categories range from the least complex category (1) to the most complex category (6). The taxonomy illustrates the progression of cognitive complexity from basic understanding to the creation of original ideas and concepts. It provides a means for aligning an educational tool to the level of skill and complexity that the tool is meant to invoke.

Table 1. Bloom's Taxonomy (revised) [11, 15].

1. Remembering	Retrieving, recognizing and recalling relevant knowledge from long-term memory.
2. Understanding	Constructing meaning from oral, written and graphic messages through interpreting, classifying, summarizing, inferring, comparing and explaining.
3. Applying	Carrying out or using a procedure through executing or implementing.
4. Analyzing	Breaking material into constituent parts, determining how the parts relate to one another and their overall purpose through differentiating, organizing and attributing.
5. Evaluating	Making judgments based on criteria and standards through checking and critiquing.
6. Creating	Putting elements together to form a comprehensive view; reorganizing elements into a new pattern or structure through generating, planning or producing.

4. Relating the Taxonomy to Training Platforms

Bloom strongly recommended the acquisition of concrete knowledge before increasing the intricacy of a training environment presented to students. In many fields, especially those with a high risk of incurring damage to property or injury, a form of training simulation is often used to gradually introduce trainees (or students) to additional variables of complexity before attempting an authentic hazardous task.

Simulations have been used in several hazardous and highly technical professions (e.g., military weapons and vehicle operation, aircraft piloting and astronautics) to build a base of knowledge and comfort for trainees. The U.S. Army uses multiple tank simulators to qualify gunnery soldiers and drivers before they operate real tanks [1, 2]. The Army also uses simulations for generic marksmanship training for soldiers called the Engagement Skills Trainer. To take the training a step further, the Army uses a training tool called the Virtual Convoy Operations Trainer, which enables collective training to be practiced in a virtual environment and multiple soldiers to train together with increasing realism [4].

Several categories of simulations, called flight simulation training devices (FSTDs), are available for private helicopter pilot training. The European Aviation Safety Agency (EASA), a certifying authority for flight simulation training environments, has developed specifications that define each environment level. The specifications cover the exact capabilities that each level is required to provide for certification (e.g., form factor of the cockpit and auditory feedback to trainees) [7]. The categories are:

- **Flight and Navigational Procedure Trainer (FNPT):** A fixed-based generic system that is primarily used for initial and refresher helicopter training, including basic and safety procedures, emergencies, navigation, instrument rating and multi-crew cooperation.

- **Flight Training Device (FTD):** A fixed-based system that simulates a specific type of helicopter. In addition to the flight and navigational procedure trainer capabilities, a flight training device is designed for rating pilots on specific helicopter types. This flight simulation training device has limited checking/testing capabilities because it does not include a motion or vibration system.

- **Full Flight Simulator (FFS):** A motion-based system that provides, in addition to a flight training device, motion and vibration cues. It has the highest level of technical complexity and training capability and can be used for proficiency evaluation.

- **Other Training Device (OTD):** A training aid for which a complete cockpit or flight deck is unnecessary. No regulations cover other training devices, which can vary from desktop computers to helicopter dashboards. These training devices are often used to drill pre-flight tasks or familiarize a pilot with a single cockpit instrument.

Each category serves a different purpose by introducing new variables and increased realism. The simulation categories with increasing levels of realism were created to ensure that pilots demonstrate mastery of the equipment and procedures during training to reduce risk during real flights.

NASA houses numerous training simulators that familiarize its trainees with a variety of environments and situations (e.g., launch, landing, payload and rendezvous activities) [17]. The simulations include fixed-based and motion-based simulators. Astronauts train for 300 hours in the simulators to qualify for real operations. NASA training also extends outside the virtual environment to space environments that are recreated using special aircraft and pools. This training is necessary for the astronauts to gain confidence before operating in the hazardous and unpredictable environment of space.

Every training environment provided by these organizations is intended to gradually assimilate trainees into a real, complex and diverse environment. Each environment serves a different purpose and is tailored to the needs of trainees. With every step forward in the training process, a new training platform is introduced that provides new concepts and builds the strong base of knowledge needed to operate in an unpredictable real environment.

5. Training Environment Development

The framework presented in this section identifies industrial control system first response training environment components that facilitate skill acquisition. The skills are divided into overarching phases of a cyber incident response based on the National Institute of Standards and Technology (NIST) Incident Response Lifecycle [5]. The skills presented in each phase of the lifecycle are the result of the analysis and consolidation of multiple sources, including several NIST and U.S. Department of Homeland Security publications.

5.1 Preparation

When considering incident response preparation for an industrial control system, a defense-in-depth strategy is not always appropriate for a response team that, in most cases, will interact with the system only after an incident has occurred. Time-sensitive responses in unfamiliar environments require the preparation phase to focus on the acquisition of general knowledge about industrial control systems that can be used in a variety of environments.

The following skills are deemed to be necessary for incident response preparation:

- **Risk and Recovery Prioritization:** The ability to prioritize components that pose the principal security risks to a system as a whole and determine the components that should be addressed.

- **Attack Vector Assessment:** The ability to understand the attack path that an intruder may take when attempting to compromise a system. A responder must understand how an attacker can gain access and the techniques that could be used to manipulate and pivot in the system.

- **Communication with Asset Owners:** The ability to communicate with an asset owner and employees is essential to gain an understanding of system operation. It enables responders to gain insights into system and network layouts, the scope of the damage and the limitations of a response effort. This skill enables the execution of all other skills required during the preparation phase.

- **Competence with Control System Components:** The ability to understand the functions of control system components and how the components (e.g., engineering software, control hardware and control interfaces) operate in order to be able to identify irregularities, malfunctions and manipulations.

Preparing for cyber responses to an industrial control system requires training components that represent the system in a realistic manner.

The following components are deemed to be necessary for assessing preparation skills:

- **System Familiarization Components:** Examples of system familiarization components include descriptions of the devices that guide risk and recovery prioritization in a response plan and network maps that assess the ability of a trainee to understand the role of operational technology.

- **Control Hardware:** Physical industrial control devices that manage the operation of a physical process include programmable logic controllers and remote terminal units.

- **Engineering Software:** This software is used to program and configure industrial control system hardware. The software is often proprietary and is provided by the control hardware vendor.

- **Human Machine Interface (HMI) Software:** This software supports the monitoring and control of a physical process. It enables a human operator to monitor, analyze and control the operational status of the process.

- **Real Control Process:** A realistic process is one that is encountered in an industrial setting and that provides trainees with opportunities to interact and experiment with the process and understand the physical effects.

- **Varied Industrial Control Vendor Exposure:** It is important to expose trainees to multiple proprietary control technologies in a single training environment. This enables the trainees to grasp the similarities and differences between proprietary control components.

5.2 Detection and Analysis

One of the most challenging aspects of incident response is to accurately detect an attack and determine the scope of the problem [5]. This phase is complicated by the wide range of detection technologies that may provide conflicting, inaccurate and/or incomplete information. Assets may also have malfunctions that were not necessarily caused by malicious activities. While an industrial control system response team is normally not the primary detector of an incident, the response team must be able to identify the potential signs of the problem and confirm, by applying detection and analysis methods, that the problem was caused by malicious activity. The team must be able to engage all sources of incident indicators, including intrusion detection systems, anti-virus systems, security information and event management systems (SIEMs), and network-based and operating-system-based logging systems.

While traditional detection devices are valuable, an industrial control system response team should be able to apply its industrial control knowledge to detect malicious effects that may be physically visible or inherent in control software. This adds a layer of complexity over and above traditional information technology intrusion detection systems.

The following skills are deemed to be necessary for detection and analysis:

- **Anomaly and Event Detection:** The ability to use software and hardware detection systems to pinpoint anomalies and events that impact system operation.

- **System Component Monitoring:** The ability to monitor control components and their logical execution to analyze their functionality and detect abnormalities. This includes monitoring via physical means and software.

- **Traffic Monitoring and Analysis:** The ability to monitor and filter traffic in order to track malicious behavior in an industrial control system and the ability to analyze and understand network traffic in the industrial control system.

- **Log Analysis:** The ability to forensically analyze system logs to trace an incident to its cause and track an attacker.

Assessing detection and analysis skills requires components that implement detection technologies, physical effects and realistic network activity.

The following components are deemed to be necessary for assessing detection and analysis skills:

- **Physical Component Effects:** These include the physical operations involved in a process (e.g., pumping of water in a wastewater treatment plant).

- **Anomaly Detection Tools:** These software and hardware tools enable the detection of anomalies in a control system and network via analyses of the physical process and network traffic (e.g., Grassmarlin and Symantec anomaly detection systems for industrial control systems).

- **Realistic Industrial Network Traffic:** It is important to provide realistic industrial control network traffic corresponding to various industrial protocols (e.g., Modbus, EtherNet/IP and DeviceNet). This provides trainees with practical industrial control protocol exposure to perform analysis and monitoring.

- **System Logging:** Components such as network logging tools and data historians must be available to log interactions and industrial process data. These components enable trainees to conduct forensic analyses of industrial control systems.

5.3 Containment, Eradication and Recovery

The containment, eradication and recovery phase focuses on the ability of a responder to select and apply appropriate strategies for isolation, evidence handling, source identification, threat eradication and restoration [5].

Containment strategies include the complete disconnection of an attacker (or source of activity), sandboxing and network filtration. Implementing a temporary solution that decreases malicious activity and prevents further damage is also included in containment. A strategy for containing a threat must consider the possible consequences (e.g., internal damage and solution duration) [5].

In an industrial control system environment, disconnection can lead to catastrophic effects to the control process, where components may depend on each other for interoperability. The same problem can arise during an attempt to filter traffic. It is important for an industrial control system incident responder to understand system operations before making any isolation or containment decisions.

Evidence must be gathered to document an incident and pursue legal proceedings. The evidence should also include all identifying information, information about the personnel who collected, handled and analyzed the evidence, the times and dates of occurrences and the evidence storage locations [5].

To facilitate recovery, a responder or response team must accurately assess the cause of the problem and apply the proper fixes. After the threat has been completely eradicated and system operations are restored, a series of tests must be conducted to ensure the return to system normality.

The following skills are deemed to be necessary for containment, eradication and recovery:

- **Return of a System to the Operational State:** The ability to rapidly return a system to an operational state, mitigate physical and financial losses, and conduct tests to ensure that the system is restored properly.

- **Attacker Identification:** The ability to identify the source of the incident through a forensic investigation.

- **Attacker Disconnection or Sandboxing:** The ability to isolate an attack source from a network and ensure that no further damage can be done by the attacker.

- **Identification and Mitigation of Exploited Vulnerabilities:** The ability to identify the vulnerabilities that were exploited in an attack and mitigate the security weaknesses.

- **Evidence Gathering and Handling:** The ability to gather and handle evidence in a manner that does not compromise the investigation.

To assess the ability to mitigate, eradicate and document attacks, a training environment must include elements that enable a responder to perform actions that stop attacks while keeping the system functional to the extent possible.

The following components are deemed to be necessary for assessing containment, eradication and recovery skills:

- **Emergency Backup Operation Equipment:** This enables the deployment of manual backup operations that prevent a critical process from failing completely. This enables a trainee to prioritize system operations.

- **Real Malware and Attack Scripts:** These help produce realistic attack scenarios and genuine effects on a system that help trainees to detect and defend against attacks.

- **Physical Disconnection or Isolation Options:** These enable the physical disconnection of portions of a system or the isolation of a portion of the system using some form of sandboxing.
- **Filtering Capabilities:** These involve the deployment of filtering technology (e.g., firewalls) in an effective manner.
- **Acceptance Test Execution:** The execution of acceptance tests helps determine whether or not a system has recovered.

5.4 Post-Incident Activity

The post-incident activity phase involves the synthesis of conclusions from the gathered evidence.

The following skills are deemed to be necessary for post-incident activity:

- **Malware Handling and Analysis:** The ability to understand the effects of malware and the proper way to analyze malware.
- **Incident Documentation:** The ability to synthesize conclusions from evidence and knowledge for accurate documentation and response justification.

In order to assess post-incident activity performance, a training environment must include elements that enable the responder to further analyze and properly document the incident in accordance with organizational standards.

The following components are deemed to be necessary for assessing post-incident activity skills:

- **Malware Analysis Tools:** This software is used to dissect and analyze malware (e.g., IDA Pro, OllyDbg and WinDbg).
- **Documentation Standards:** These formalize the documentation process and ensure that it is performed as required by the organization.

5.5 Training Administration

Every environment should provide effective training as well as feedback to trainees. While it is not part of the incident response lifecycle, proper administration of training is vital to the educational experience of participants. Several components are necessary to ensure the complete monitoring of a training environment.

The following skills are deemed to be necessary for training administration:

- **Real-Time View of Physical Signal Exchange:** The ability of a training administrator to view the input and output signals at system endpoints (e.g., sensors and actuators). This helps ensure that the administrator can assess an accurate representation of an environment even when the integrity of the system monitoring components has been comprised and the components are untrustworthy for assessment purposes [31].

- **Remote Administrative Monitoring:** The ability of a training administrator to assess trainees and control an exercise from a different physical location than the exercise environment.

- **Remote Participation:** The ability of a training administrator to administer exercises to trainees at remote physical locations.

6. Training Environment Levels

This section describes the different levels of industrial control system training environments based on their realism and the fidelity of their components and capabilities. While each level has varying capabilities, the primary delimiter between levels is the increased realism that the environment at the higher level provides in the context of a real industrial control system.

6.1 Level 1 Training Environment

A Level 1 training environment is completely software-based and simulates an industrial control device or control system. This type of environment can provide simplified education and training to inexperienced individuals in the areas of controller operations and process control logic.

Example environments are the LogixPro-500 programmable logic controller simulator [26] and the Honeyd programmable logic controller interaction software [30]. These environments do not provide real physical interactions, just software-defined capabilities. While basic interaction and programming features are supported, the simulation programs may not mimic the exact behavior of real control hardware and software. For example, the Honeyd simulation software supports 2,000 TCP requests per second with 65,536 hosts compared with real programmable logic controllers that support significantly fewer connections [30]. Level 1 environments are also limited by their inability to provide realistic defensive response interactions. Additionally, they do not allow for physical disconnection options that are available to defenders in a real environment.

6.2 Level 2 Training Environment

A Level 2 training environment is an emulated system that manifests real physical effects, but does not incorporate genuine control system hardware and software. This type of environment can be constructed using embedded devices (e.g., Arduino, Raspberry Pi and BeagleBone) or other computing devices that can be programmed to control physical sensors and actuators using common programming languages (e.g., C and Python).

Level 2 environments are used as training platforms and research testbeds by many organizations. Researchers at the Air Force Institute of Technology (AFIT) have created a Level 2 environment that emulates an automobile CAN bus, which is controlled by a BeagleBone Black (Figure 1). The environment, which serves as a research testbed, is used to test the effects of CAN

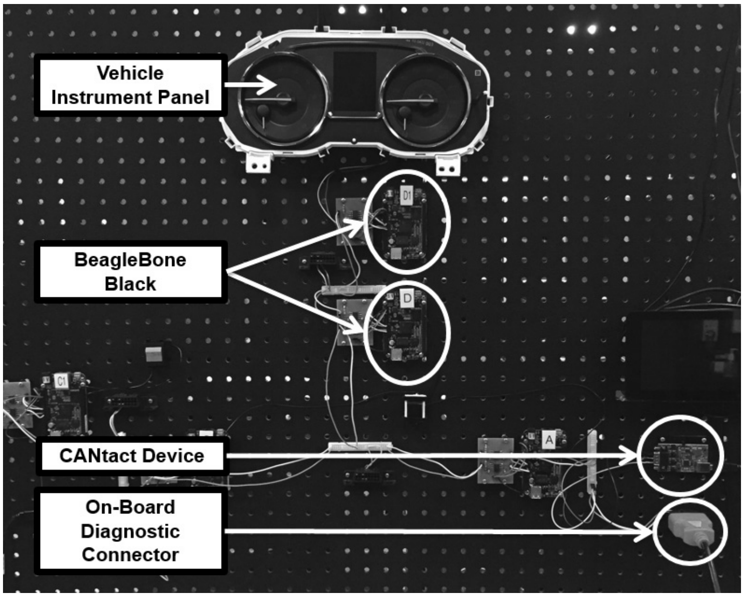


Figure 1. Automobile CAN bus emulation testbed.

bus attacks on vehicular control. The U.S. Industrial Control System Cyber Emergency Response Team (ICS-CERT) uses portable Level 2 training platforms to conduct basic industrial control system classes and exercises. The CybatiWorks Level 2 training kits incorporate Raspberry Pi control emulation devices representing stoplights that use mounted light-emitting diodes (LEDs) as acuators [6]. Siaterlis et al. [24] have created a Level 2 industrial control system simulation testbed for assessing the effects of attacks on networked control systems. While Level 2 environments can manifest physical effects and emulate process control systems, the environments are restricted by the code that executes on the embedded devices. Thus, the environments cannot be guaranteed to mirror the exact behavior of real industrial control systems.

6.3 Level 3 Training Environment

A Level 3 environment comprises genuine process control hardware and software corresponding to a partial industrial control system. In the case of a wastewater treatment plant, an example Level 3 environment comprises the hardware and software that control the lift station portion of the wastewater treatment process. While a Level 3 environment is not fully realistic, it provides a scaled form of realism. Such an environment familiarizes trainees with vendor equipment, industrial networks, process control logic and portability, eliminating the need to construct and maintain a complete and expensive facility.

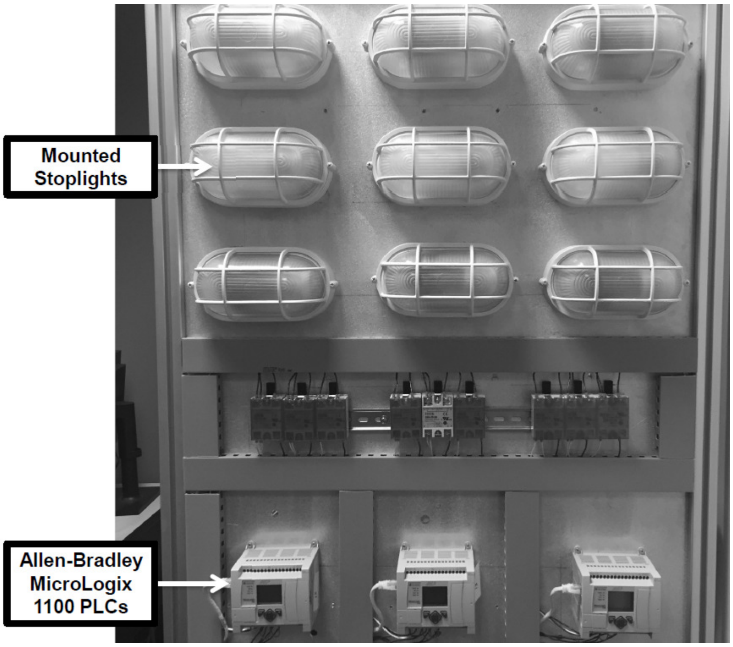


Figure 2. Mounted stoplight control system.

Level 3 environments are used for a variety of security-related activities. The Sandia SCADA Security Development Laboratory, which is considered to be a Level 3 environment, is used to create and evaluate security practices, programs and protocols [21]. Other examples include the Air Force Institute of Technology stoplight system with Allen-Bradley MicroLogix programmable logic controllers that is used to teach industrial control system defense classes (Figure 2). The SANS CyberCity combines Level 2 and Level 3 components in a compact environment that provides robust learning experiences [23].

A Level 3 environment may have genuine hardware and software components, but it still lacks the realism of a production system. This type of environment would not impart an in-depth understanding of a real-world system, especially scenarios where attacks have cascading effects due to interconnections with other systems [3].

6.4 Level 4 Training Environment

A Level 4 training environment is a genuine industrial control facility with functioning processes. Sample Level 4 training environments are located at the Atterbury-Muscatatuck Urban Training Center (MUTC) near Butlerville, Indiana. The training center, which is operated by the Indiana National Guard, is used for military and first responder training. The center houses multiple

Table 2. Mapping of the training environment levels to Bloom's Taxonomy.

Bloom's Taxonomy	**Training Environment Levels**			
	Level 1	**Level 2**	**Level 3**	Level 4
1. Remembering	✓	✓	✓	✓
2. Understanding	✓	✓	✓	✓
3. Applying	–	✓	✓	✓
4. Analyzing	–	✓	✓	✓
5. Evaluating	–	–	✓	✓
6. Creating	–	–	–	✓

industrial facilities, each of which is a separate Level 4 environment. The facilities include a power plant, prison, hospital, subway station, power distribution system and wastewater treatment plant [13]. The cyber portions of some of the Level 4 training environments are still under development.

7. Mapping Training Environment Levels

As the level of a training environment increases, so does the level of cognitive complexity and thinking that can be assessed in the environment. The complexity of training scenarios that can be presented in an industrial control system environment depends on the amount of realism of observations and interactions. Bloom's Taxonomy can be mapped to the different levels of industrial control system training environments for training defenders. The taxonomy is hierarchical in nature and, therefore, an environment that can support exercises at the higher levels of Bloom's Taxonomy can also support exercises at the lower levels. Table 2 shows the mapping of the training environment levels to Bloom's Taxonomy.

A Level 1 fully-simulated industrial control system training environment can present exercises and problems that address the first two cognitive levels of Bloom's Taxonomy, specifically "remembering" and "understanding." The simulated training environment can help evaluate a trainee's ability to recall and retrieve facts that have been programmed into the simulation. The trainees can also interpret meanings from the lessons and make references and comparisons based on the presented facts. A simulation that provides simple programmable logic controller interactions can impart basic programmable logic controller concepts and behavior. However, there is no guarantee that the learnings will directly carry over to a real system. This constrains the ability of a Level 1 environment to support assessments at the higher levels of Bloom's Taxonomy. Another constraint is the limited ability of a trainee to implement realistic security measures. Since a simulated environment can only offer what it is programmed to do, a trainee cannot manipulate a network or make unanticipated configurations to control systems.

A Level 2 training environment in which emulated devices perform physical controller functions can help assess the "applying" and "analyzing" levels of Bloom's Taxonomy. In this type of environment, a trainee can dissect the environment, understand the components and control strategies, and implement external defenses (e.g., firewalls and network isolation). However, since a Level 2 environment does not incorporate real industrial control components, it cannot help evaluate a trainee at Bloom's "evaluating" level.

A Level 3 environment comprises vendor-supplied industrial control hardware and software, but it is not a comprehensive production system. Therefore, it supports training and exercises up to the "evaluating" level of Bloom's Taxonomy. This level of thinking is characterized by making judgments and critiques based on criteria and standards. Evaluative thinking in a Level 3 environment is accomplished by comparing data and observations against the standard operational criteria of control components. The real data enables participants to perform realistic defensive evaluations of the implemented industrial control systems. However, a Level 3 environment struggles to assess trainees at the "creating" level – the highest level of Bloom's Taxonomy.

A Level 4 environment can assess trainees at the highest "creating" level. This level of thinking is characterized by the ability to generate a comprehensive view of a situation. In the context of industrial control system defense and incident response, this type of cognitive complexity cannot be achieved without a complete functioning system. A Level 4 environment provides a trainee with opportunities to view and manipulate every possible element of a real industrial control system. Modifications to existing solutions and new solutions to defense problems can be applied and evaluated. A Level 4 environment also helps trainees discover, analyze and address real-world problems in a creative manner and to observe the ramifications of their actions (e.g., resilience and cascading effects).

8. Example Training Environments

This section presents example training environments at each of the four levels.

8.1 Level 1 Training Environments

The LogixPro-500 PLC simulator enables a trainee to create and manipulate ladder (control) logic, and to view the execution of the logic on simulated sensors and actuators [26]. Consider a scenario where a first responder must be able to analyze a ladder logic program in an industrial controller and determine if it has been tampered with. The trainee would have to understand how to read and write the logic to make these observations. A simulated environment can support the training of basic logic functions and controllers. The training scenarios in the simulated environment provide assessments up to the "understanding" level of Bloom's Taxonomy. They enable trainees to learn facts about control system operation and construct visual meanings and interpretations through the execution of the simulations. The LogixPro-500 PLC simulation training environment is available for user download at `www.thelearningpit.com/lp/logixpro.html`.

Ladder Logic Engineering Scenario.

- **Objective:** Given engineering specifications for controlling a garage door with open and closed sensors in an industrial control environment, develop the logic that enables the control hardware to execute the specifications.
- **Description:** Create a ladder logic program that enables a programmable logic controller to control a garage door with sensors that indicate when the door is opened or closed. The simulation should execute the logic provided by the trainee and visualize the physical response in the garage door simulator.
- **Type:** Understanding the functionality and relationships between the control hardware and logic software.
- **Evaluation Criteria:** Correctly engineer the required functionality within three hours.
- **References:** Engineering specifications for garage door logic, LogixPro-500 software help menu and Rockwell Automation RSLogix user guide.

LogixPro-500 Environment Components. The environment comprises a single computer system with the LogixPro-500 simulation software installed.

- **Control Hardware:** The control hardware comprises a simulated programmable logic controller that executes the ladder logic program developed by a trainee.
- **Engineering Software:** The engineering software is a version of the Allen-Bradley RSLogix500 programming tool.
- **Human-Machine Interface Software:** The human-machine interface is a software simulation that presents an interactive visual representation of the sensors and actuators. The interface displays how the sensors and actuators react to the ladder logic program in the simulated control hardware.
- **Physical Component Effects:** The physical effects of the system are presented on the computer screen.

8.2 Level 2 Training Environments

Raspberry Pi emulation devices can be programmed to emulate a network of stoplights using LEDs. Consider a scenario where a controller is compromised in order to interfere with the timing of the lights. In this scenario, a trainee would have to determine the relationships between the components and identify the cause of the incident. This would correspond to the "analyzing" level of Bloom's Taxonomy. The cost of the training environment with four stoplights is approximately $200.

Stoplight Logic Manipulation Scenario.

- **Objective:** Given a network of emulated stoplights that are out of sync, return the lights to normal functionality and find the compromised device.
- **Description:** Monitor network traffic and analyze to determine which devices were impacted and find the source of the attack.
- **Type:** Understanding the functionality and relationships between the control hardware and logic software, and applying response skills to mitigate the effects of the attack and return the system to normal functionality.
- **Evaluation Criteria:** Return the system to normal functionality within three hours and find the source of the attack within one hour.
- **References:** Network and system logs and code on the Raspberry Pi devices.

Stoplight Network Environment Components. The hardware required is a Raspberry Pi development platform that executes a logic program that controls LED lights.

- **Control Hardware:** The control hardware comprises a network of Raspberry Pi emulation controllers.
- **Human-Machine Interface Software:** The human-machine interface software is programmed to view and communicate with the Raspberry Pis.
- **Physical Component Effects:** The physical effects in the stoplight network are represented by LEDs.
- **System Logging:** Logging is implemented in the network by the Rasberry Pi platforms and passive network monitoring software (e.g., Grassmarlin).
- **Physical Disconnect or Isolation Options:** The physical separation of controllers supports network segmentation and physical network disconnection.
- **Filtering Capabilities:** Firewall filtering capabilities are built into the scenario to isolate the stoplight network from outside connections.

8.3 Level 3 Training Environments

Two examples of Level 3 environments are described: (i) wastewater treatment plant; and (ii) prison facility. All the components for each environment fit in a Pelican 1610 case (62.76 cm × 49.73 cm × 30.3 cm). The environments were created to be as cost effective as possible while incorporating genuine control devices. The scenarios support the assessment of thinking skills up to the "evaluating" level of Bloom's Taxonomy. Note that the descriptions of the environments are more detailed than the other levels to demonstrate that high levels of interaction with genuine industrial control components can be achieved at a relatively low cost while maintaining portability.

1. Wastewater Treatment Plant Environment. The Level 3 wastewater treatment plant environment models a wastewater aeration basin. If the oxygen levels are too high or low, alarms are triggered by two red lights in the exercise environment. The oxygen levels are adjusted by modifying the valve openings and the speed of blower fans. The closed loop control of oxygen level uses an Allen-Bradley programmable logic controller and an Allen-Bradley PowerFlex 40 AC variable frequency drive (VFD). The programmable logic controller controls the valve dilation and computes the oxygen levels while the variable frequency drive adjusts the fan speed based on the programmable logic controller calculations. The cost of this training environment is approximately $16,500.

An example training scenario involves a cyber attack on the wastewater treatment plant, which causes the programmable logic controller in the aeration basin to malfunction. This results in fluctuating oxygen levels.

Wastewater Treatment Aeration Basin Failure Scenario.

- **Objective:** Restore the aeration basin to full functionality and find the source and cause of the incident.
- **Description:** By monitoring network traffic, the human-machine interface and the physical devices, the trainee must recognize when the system fails and effect system recovery by blocking attacker access. Also, the trainee must implement emergency procedures to restore the failed control hardware to its normal functionality.
- **Type:** Evaluating the loss of system control and functionality.
- **Evaluation Criteria:** Return the system to normal functionality within three hours and find the source of the attack within one hour.
- **References:** ControlLogix programmable logic controller manual, PowerFlex 40 AC variable frequency drive manual, control hardware vulnerability reports and control network map.

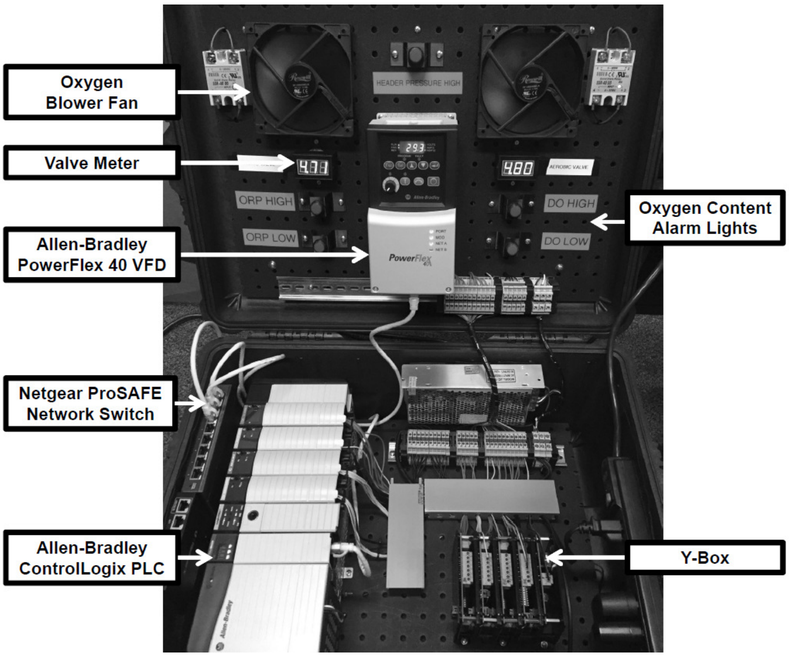

Oxygen Blower Fan
Valve Meter
Allen-Bradley PowerFlex 40 VFD
Netgear ProSAFE Network Switch
Allen-Bradley ControlLogix PLC
Oxygen Content Alarm Lights
Y-Box
HEADER PRESSURE HIGH
ORP HIGH
ORP LOW
DO HIGH
DO LOW
AEROBIC VALVE

Figure 3. Level 3 wastewater treatment plant training environment.

Wastewater Treatment Plant Environment Components.

- **Control Hardware:** The control hardware comprises an Allen-Bradley ControlLogix programmable logic controller and an Allen-Bradley PowerFlex 40 AC variable frequency drive (Figure 3).
- **Engineering Software:** The engineering software used for programming and configuring the Allen-Bradley programmable logic controller is an RSLogix 5000 system (Figure 4).
- **Real Control Process:** The control process for the environment is modeled after a wastewater aeration basin. It controls oxygen diffusion in two zones using two valves and blower fans.
- **Vendor Exposure:** The environment exposes trainees to the use of a programmable logic controller and variable frequency drive.
- **Physical Component Effects:** The physical effects are presented as voltmeter readings that indicate the extent of valve opening (controlled by the programmable logic controller) and the speed of the fans (controlled by the variable frequency drive).
- **Realistic Industrial Network Traffic Generation:** Traffic generated by the human-machine interface workstation, engineering workstation,

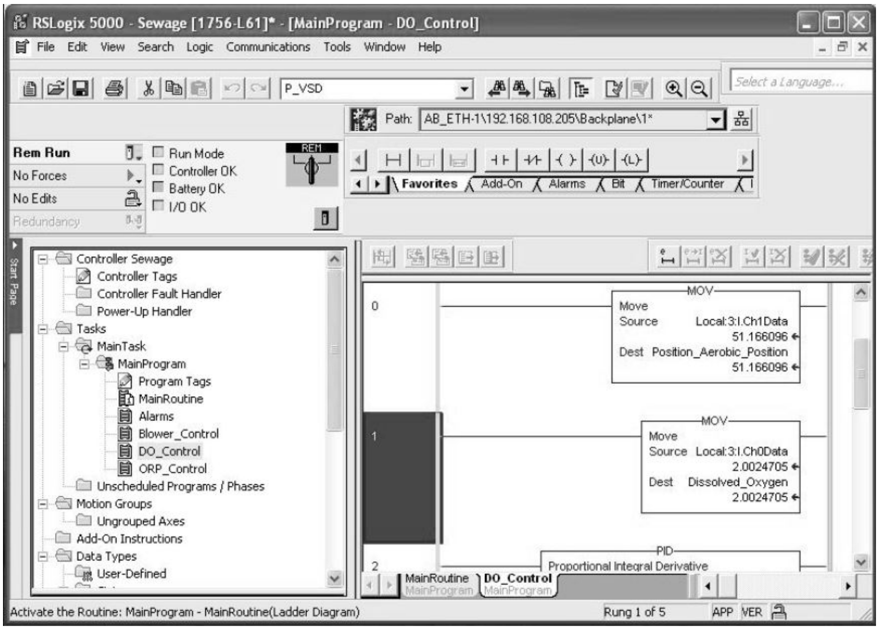


Figure 4. RSLogix 5000 engineering workstation software.

programmable logic controller and variable frequency drive is visible in the network. The realistic network traffic comprises industrial protocol communications, including EtherNet/IP and Common Industrial Protocol (CIP) traffic.

- **Real Malware or Attack Scripts:** Attack scripts that leverage insecure configurations of the control components are incorporated in the training environment.

- **Physical Disconnection or Isolation Options:** All the machines in the network can be physically disconnected from their Ethernet ports and network isolation can be achieved via whitelisting and blacklisting by a Netgear ProSAFE network switch.

- **Filtering Capabilities:** Filtering by an Ubiquiti EdgeRouterX router can be performed using simple firewall rules for network connections in the environment.

- **Real-Time View of Physical Signal Exchange:** Y-Box technology [32] enables an exercise administrator to view the operation of process control endpoints and the human-machine interface to track an ongoing attack (Figure 5).

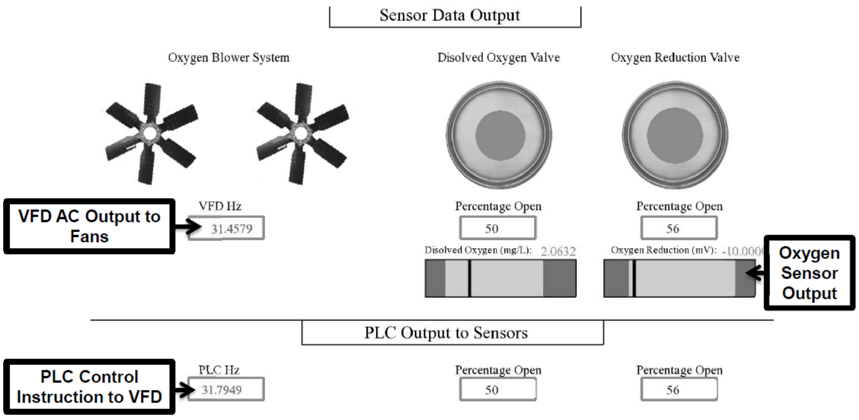


Figure 5. Administrative interface for real-time viewing of attack and defense effects.

- **Remote Administrative Monitoring:** The administration interface for the environment can be viewed using virtual network computing technology; this enables an exercise administrator to evaluate the status of the hardware and software.

- **Remote Participation:** Remote participation is accomplished using remote access tools; this does not hinder the physical manipulation capabilities.

2. Prison Facility Environment The Level 3 prison facility training environment is modeled after a prison cell block containing three prison cells with door lock controls and a mantrap access control system. An Omron programmable logic controller controls the operation of the locks, buttons, security lights and alarm. This equipment costs approximately $1,400.

An example training scenario involves the prison facility experiencing a cyber attack that causes a programmable logic controller to malfunction. This results in the prison door locks being opened.

Prison Control System Failure Scenario.

- **Objective:** Restore the prison to full functionality and find the source and cause of the incident.

- **Description:** By monitoring network traffic, the human-machine interface and the physical devices, the trainee must understand when the system fails and effect system recovery.

- **Type:** Evaluating the loss of system control and functionality.

- **Evaluation Criteria:** Return the system to normal functionality within three hours and find the source of the attack within one hour.

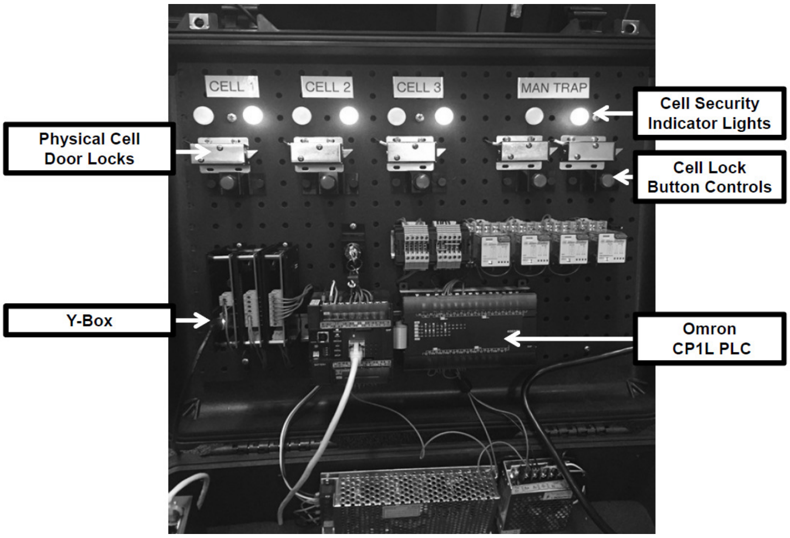


Figure 6. Level 3 prison training environment.

- **References:** Omron CP1L programmable logic control manual, control hardware vulnerability reports and control network map.

Prison Facility Environment Components.

- **Control Hardware:** The control hardware comprises an Omron CP1L programmable controller that controls the prison door locks, buttons, security lights and alarm (Figure 6).

- **Engineering Software:** The engineering software comprises the Omron CX-Programmer.

- **Human-Machine Interface:** The human-machine interface for controlling the prison environment (Figure 7) was created using the Schneider Electric IGSS Free50 software.

- **Real Control Process:** The control process for the environment is modeled after a cell block in a prison in the United States.

- **Physical Component Effects:** The locks and lights are operated using the human-machine interface controls and by physically pressing the lock control buttons in the Pelican case housing the control equipment.

- **Realistic Industrial Network Traffic Generation:** Traffic generated by the human-machine interface workstation, engineering workstation and programmable logic controller is visible in the network. The realistic

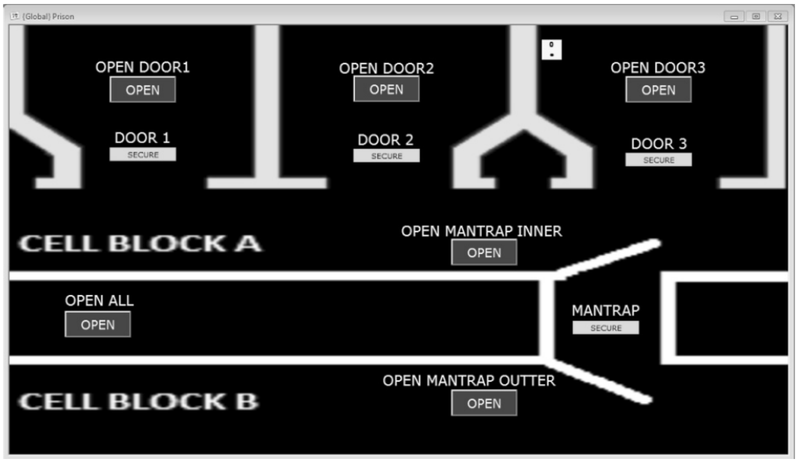


Figure 7. Human-machine interface.

network traffic comprises industrial protocol communications, including EtherNet/IP, Common Industrial Protocol and proprietary Omron protocol traffic.

- **Real Malware or Attack Scripts:** Attack scripts that leverage insecure configurations of the control components are incorporated in the training environment.

- **Physical Disconnection or Isolation Options:** All the machines in the network can be physically disconnected from their Ethernet ports.

- **Filtering Capabilities:** Filtering by an Ubiquiti EdgeRouterX router can be performed using simple firewall rules for network connections in the environment.

- **Real-Time View of Physical Signal Exchange:** Y-Box technology enables an exercise administrator to view the operation of the process control endpoints and the human-machine interface to track an ongoing attack (Figure 8).

- **Remote Administrative Monitoring:** The White Cell interface for the environment can be viewed using virtual network computing technology.

- **Remote Participation:** Remote participation is accomplished using remote access tools; this does not hinder the physical manipulation capabilities.

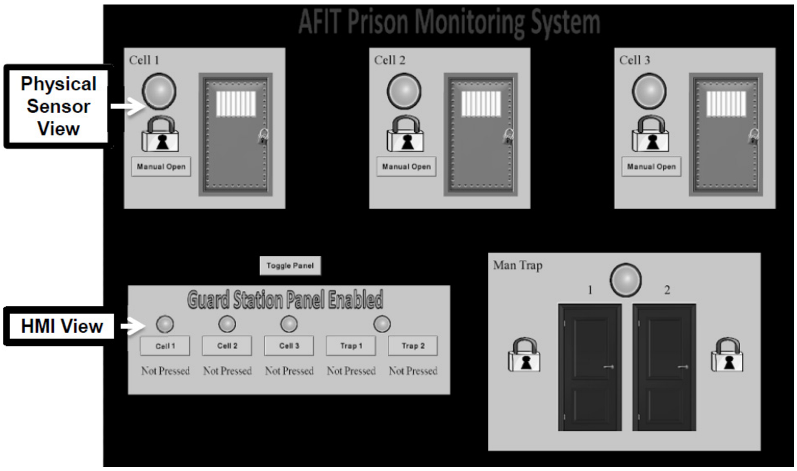


Figure 8. Administrative monitoring view.

8.4 Level 4 Training Environments

A Level 4 training environment functions at the "creating" level of Bloom's Taxonomy. It presents a trainee with a fully-realistic scenario that enables the trainee to use all the available skills and knowledge to arrive at new solutions to complex problems.

In the case of a power distribution plant, a suitable scenario for a Level 4 environment is the appearance of unusual traffic accompanied by unexplained power fluctuations. Given the complexity of the environment with its many components and connections, a trainee would have to appropriately plan a response by prioritizing components in the network, narrow down the root cause of the anomaly and apply fixes to manage the incident and ensure system recovery. If the incident results from cascading effects in a real system, it can be difficult to determine the root cause and craft an appropriate response. This is because most industrial systems are unique environments and the response of a trainee has to be tailored to the specific environment.

The components used to construct a Level 4 power plant environment are similar to those in a real power plant; however, significant additional engineering tasks would be necessary to implement exercise control and monitoring. A Level 4 training environment may not be suitable to train beginners due to the risk of facility damage if an exercise does not go as intended. Instead, a Level 1 or Level 2 environment could be used as a safe sandbox for beginners to make mistakes and learn from their mistakes.

Failsafe plans should also be considered when designing a Level 4 environment for unpredictable situations during exercises that could lead to facility damage. The cost of constructing a Level 4 power plant can be in the millions of dollars or more. While such a Level 4 training environment would certain not be mobile, it would provide remote access as in the case of a real infrastructure asset.

9. Conclusions

This chapter has specified four classes of training environments that are mapped to Bloom's Taxonomy, thus covering the various levels of cognitive complexity required in training programs for industrial control system first responders. Level 1 environments are appropriate for average plant operators while Level 4 environments are needed to prepare industrial control system first responders to handle genuine emergencies. The categorization of environments in terms of progressive complexity is necessary to ensure adequate training and readiness of first responders. The proposed categories also help determine the training environment levels that best align with the training goals and budgets of organizations. Well-designed exercise regimens that properly leverage the appropriate levels of training environments will greatly reduce the likelihood of tragic incidents like the explosion at the fertilizer plant in West, Texas that killed fifteen people, including ten first responders.

Note that the views expressed in this chapter are those of the authors and do not reflect the official policy or position of the U.S. Air Force, U.S. Army, U.S. Department of Defense or U.S. Government.

Acknowledgement

This research was partially supported by the U.S. Department of Homeland Security Industrial Control Systems Cyber Emergency Response Team (ICS-CERT).

References

[1] C. Baltos, Soldiers prepare on tank simulators for Saber Guardian exercise, U.S. Army, Washington, DC (www.army.mil/article/171603/soldiers_prepare_on_tank_simulators_for_saber_guardian_exercise), July 15, 2016.

[2] D. Beckstrom, Tank gunnery simulator: Getting back to basics, U.S. Army, Washington, DC (www.army.mil/article/143185/Tank_Gunnery_Simulator__Getting_Back_to_Basics), February 19, 2015.

[3] J. Butts and M. Glover, How industrial control system security training is falling short, in *Critical Infrastructure Protection IX*, M. Rice and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 135–149, 2015.

[4] H. Chang, Simulators always valuable in military training, U.S. Army, Washington, DC (www.army.mil/article/19599/simulators-always-valuable-in-military-training), April 13, 2009.

[5] P. Cichonski, T. Millar, T. Grance and K. Scarfone, Computer Security Incident Handling Guide, NIST Special Publication 800-61, Revision 2, National Institute of Standards and Technology, Gaithersburg, Maryland, 2012.

[6] Cybati, CybatiWorks, Bloomington, Illinois (`cybati.org/cybatiworks-one`), 2015.

[7] European Aviation Safety Agency, Certification Specifications for Helicopter Flight Simulation Training Devices, CS-FSTD(H), Annex to ED Decision 2012/011/R, Cologne, Germany, 2012.

[8] European Helicopter Safety Team, Teaching and Testing in Flight Simulation Training Devices (FSTD) for Helicopter Pilots, Instructors and Examiners, Training Leaflet, Cologne, Germany, 2016.

[9] M. Fabro and E. Cornelius, Recommended Practice: Creating Cyber Forensics Plans for Control Systems, Idaho National Laboratory, INL/EXT-08-14231, Idaho Falls, Idaho, 2008.

[10] M. Fernandez, Fire that left 15 dead at Texas fertilizer plant is ruled intentional, *New York Times*, May 11, 2016.

[11] M. Forehand, Bloom's Taxonomy, Emerging Perspectives on Learning, Teaching and Technology, University of Georgia, Athens, Georgia (`epltt.coe.uga.edu/index.php?title=Bloom%27s_Taxonomy`), 2005.

[12] Idaho National Laboratory, INL Cyber Security Research: Defending the Network Against Hackers, Fact Sheets: 21st Century Science and Technology, Idaho Falls, Idaho (`www.inl.gov/research/inl-cyber-security-research`), 2014.

[13] Indiana National Guard, Atterbury/Muscatatuck, MUTC Overview, Edinburgh, Indiana (`www.atterburymuscatatuck.in.ng.mil/Ranges/MuscatatuckUrbanTrainingCenter/MUTCOverview.aspx`), 2017.

[14] Industrial Control Systems Cyber Emergency Response Team (ICS-CERT), Training available through ICS-CERT, Idaho Falls, Idaho (`www.ics-cert.us-cert.gov/Training-Available-Through-ICS-CERT`), 2017.

[15] D. Krathwohl, A revision of Bloom's Taxonomy: An overview, *Theory into Practice*, vol. 41(4), pp. 212–218, 2002.

[16] T. Morris, A. Srivastava, B. Reaves, W. Gao, K. Pavurapu and R. Reddi, A control system testbed to validate critical infrastructure protection concepts, *International Journal of Critical Infrastructure Protection*, vol. 4(2), pp. 88–103, 2011.

[17] National Aeronautics and Space Administration, Johnson Space Center: Training for Space, Astronaut Training and Mission Preparation, FS-2006-03-011-JSC, Washington, DC (`www.nasa.gov/centers/johnson/pdf/160410main_space_training_fact_sheet.pdf`), 2006.

[18] National Training and Simulation Association, Air Force Training 2015, Arlington, Virginia (`www.trainingsystems.org/publications/AirForce.pdf`), 2010.

[19] M. Pell, R. McNeill and J. Roberts, Unprepared: Texas blast shows failure of emergency planning law, analysis shows, *NBC News*, May 22, 2013.

[20] B. Reaves and T. Morris, An open virtual testbed for industrial control system security research, *International Journal of Information Security*, vol. 11(4), pp. 215–229, 2012.

[21] Sandia National Laboratories, National Supervisory Control and Data Acquisition (SCADA) Test Bed, Albuquerque, New Mexico (`energy.sandia.gov/energy/ssrei/gridmod/cyber-security-for-electric-infrastructure/scada-systems`), 2015.

[22] Sandia National Laboratories, SCADA Training Courses, Albuquerque, New Mexico (`energy.sandia.gov/energy/ssrei/gridmod/cyber-security-for-electric-infrastructure/scada-systems/education-and-training`), 2015.

[23] SANS Institute, SEC562: CyberCity Hands-on Kinetic Cyber Range Exercise, Bethesda, Maryland (`www.sans.org/course/cybercity-hands-on-kinetic-cyber-range-exercise`), 2017.

[24] C. Siaterlis, B. Genge and M. Hohenadel, EPIC: A testbed for scientifically rigorous cyber-physical security experimentation, *IEEE Transactions on Emerging Topics in Computing*, vol. 1(2), pp. 319–330, 2013.

[25] K. Stouffer, J. Falco and K. Scarfone, Guide to Industrial Control Systems (ICS) Security, NIST Special Publication 800-82, National Institute of Standards and Technology, Gaithersburg, Maryland, 2011.

[26] The Learning Pit, LogixPro 500 PLC Simulator, Whitby, Canada (`www.thelearningpit.com/lp/logixpro.html`), 2016.

[27] U.S. Department of Energy, National SCADA Test Bed, Enhancing Control Systems Security in the Energy Sector, NTSB Fact Sheet, Washington, DC (`www.energy.gov/sites/prod/files/oeprod/DocumentsandMedia/NSTB_Fact_Sheet_FINAL_09-16-09.pdf`), 2009.

[28] U.S. Department of Homeland Security, Recommended Practice: Developing an Industrial Control Systems Cybersecurity Incident Response Capability, Washington, DC, 2009.

[29] N. Wertzberger, C. Glatter, W. Mahoney, R. Gandhi and K. Dick, Towards a low-cost SCADA testbed: An open-source platform for hardware-in-the-loop simulation, *Proceedings of the International Conference on Security and Management*, pp. 555–561, 2011.

[30] M. Winn, M. Rice, S. Dunlap, J. Lopez and B. Mullins, Constructing cost-effective and targetable industrial control system honeypots for production networks, *International Journal of Critical Infrastructure Protection*, vol. 10, pp. 47–58, 2015.

[31] J. Yoon, Framework for Evaluating the Readiness of Cyber First Responders for Industrial Control Systems, M.S. Thesis, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2016.

[32] J. Yoon, S. Dunlap, J. Butts, M. Rice and B. Ramsey, Evaluating the readiness of cyber first responders responsible for critical infrastructure protection, *International Journal of Critical Infrastructure Protection*, vol. 13, pp. 19–27, 2016.

Chapter 14

MULTI-CONTROLLER EXERCISE ENVIRONMENTS FOR TRAINING INDUSTRIAL CONTROL SYSTEM FIRST RESPONDERS

Joseph Daoud, Mason Rice, Stephen Dunlap and John Pecarina

Abstract When systems are targeted by cyber attacks, cyber first responders must be able to react effectively, especially when dealing with critical infrastructure assets. Training for cyber first responders is lacking and most exercise platforms are expensive, inaccessible and/or ineffective. This chapter describes a mobile training platform that incorporates a variety of programmable logic controllers in a single system that helps impart the unique skills required of industrial control system cyber first responders. The platform is modeled after a jail in the United States and was developed to maximize realism. Training scenarios are presented that cover specific cyber first responder skills and techniques. The results demonstrate that the platform is robust and highly effective for conducting sustained training exercises in curricula developed for cyber first responders.

Keywords: Industrial control systems, cyber first responders, training platform

1. Introduction

Diseases can manifest themselves in a number of ways depending on the individual. For health care professionals, this can sometimes make a diagnosis difficult and an accurate prognosis challenging. To prepare themselves to perform these tasks, medical students go through a rigorous curriculum that goes well beyond traditional classroom lectures. The curriculum involves many practical exercises, clinical rotations, internships and residency [5]. The knowledge, skills and experience gained from such a curriculum enhances a physician's ability to analyze a patient's symptoms in the context of the patient's unique medical history in order to arrive at a diagnosis and an accurate prognosis [2].

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 273–294, 2017.
https://doi.org/10.1007/978-3-319-70395-4_14

As in the case of human diseases, cyber threats manifest themselves in many different ways and cyber first responders must be able to diagnose and respond to the threats just like physicians. Given the similarities between the two endeavors, it is reasonable to expect that hands-on training similar to what medical students receive would be very effective for cyber first responders.

This chapter describes a training platform that is specifically designed to provide realistic training for cyber first responders. The mobile training platform incorporates several programmable logic controllers (PLCs) as well as other realistic hardware and software components that maximize the knowledge, skills and experience gained by cyber first responders during their training.

2. Background

Academic institutions, government organizations and businesses offer a variety of certifications, training courses and degree programs in the area of cyber security [9, 12]. These programs provide cyber first responders with valuable skills. However, the vast majority of these programs focus on traditional information technology (IT) systems, often neglecting operational technology (OT) systems. While there is some overlap between the two types of systems with regard to security, the differences are significant enough that cyber first responders need specialized knowledge, skills and experience to handle operational technology incidents involving industrial control systems. Two distinguishing characteristics of industrial control systems are the heavy use of proprietary software and communications protocols and the focus on safety. Almost every vendor has its own proprietary applications for interacting with its control devices (e.g., programmable logic controllers). Additionally, industrial control systems manage physical processes in which anomalies can present significant safety risks. Cyber first responders must be cognizant of these factors when conducting their activities.

Several industrial control system testbeds have been developed, but the vast majority of testbeds are geared toward research and development as opposed to education and training:

- **Sandia National Laboratories:** Sandia National Laboratories operates several facilities, including the Distributed Energy Technology Laboratory, Network Laboratory, Cryptographic Research Facility, Red Team Facility and Advanced Information Systems Laboratory [11]. All these testbeds contain real and simulated supervisory control and data acquisition (SCADA) assets for research and development in various domains. For example, the Distributed Energy Technology Laboratory houses industrial control systems used in electricity generation and distribution; however, control system security is not necessarily the primary focus of research at the laboratory [10].

- **Idaho National Laboratory:** Idaho National Laboratory has several facilities [7]. One of its cyber security facilities is intended to connect to several existing critical infrastructure testbeds, including a SCADA testbed, power grid testbed, mock chemical mixing facility, wireless testbed and physical security testbed. The testbeds comprise a full-scale critical infrastructure test range that covers 890 square miles. Unfortunately, due to the nature of the facility, most learning opportunities are restricted to authorized individuals from government and industry [8].

- **National Institute of Standards and Technology:** The National Institute of Standards and Technology is tasked with publishing guidelines and recommended practices in many disciplines, including cyber security. The NIST industrial control system testbed was developed to enable the evaluation of security guidelines and best practices [4]. The testbed comprises real and emulated industrial control system components that can be evaluated with the appropriate security mechanisms in place.

- **SANS Institute:** The SANS CyberCity is one of a few physical industrial control system platforms that is specifically designed for security training. The CyberCity platform is used in the SANS SEC562 course, which focuses on penetration testing and kinetic cyber effects [13]. It is a 1:87 scale city with hands-on exercises involving railway switching junctions, a water reservoir and power grid. While some of the systems in CyberCity are simulated, real hardware is incorporated in the power grid system, including Allen-Bradley, Siemens and Phoenix Contact programmable logic controllers [14]. CyberCity is an effective training platform, but it is very expensive. Furthermore, it is not a mobile platform. While it can be accessed remotely for training purposes, remote training does not support intense, hands-on interactions with physical components, which is an important aspect of cyber first response training.

Effective training curricula must be in place to enable professionals to assess and react to cyber incidents involving industrial control systems. Butts and Glover [3] propose three core areas that should be covered in an industrial control system training course: (i) industrial control system principles; (ii) cyber manipulation; and (iii) response coordination. Each core area has recommended instructional blocks that cover important areas of proficiency.

The industrial control system principles core provides an introduction to common control system components, cyber-physical interactions involving these components, communications protocols and real-world configurations. The cyber manipulation core covers attack techniques that target industrial control system components and networks. The response coordination core primarily focuses on industrial control system incident response. Butts and Glover [3] recommend that all these concepts be taught via realistic scenarios involving genuine industrial control systems.

Even the best training platforms have very limited value unless they are used appropriately. Developing realistic training scenarios is an important, but difficult, task. Traditional "capture the flag" events, which are focused on gaining access, are fundamentally inadequate for industrial control systems.

The actions taken after gaining access to an industrial control system are far more important.

An effective evaluation of an industrial control system scenario must incorporate the physical process being controlled. Yoon et al. [17] leverage the NFPA 1410 format, which is used by firefighters, to develop an effective framework for evaluating the readiness of cyber first responders. This framework contains specific objectives, descriptions, evaluation criteria and accompanying references for each training scenario. Furthermore, each scenario contains a designator that describes the type of scenario and the skills addressed by the scenario. The framework proposed by Yoon and colleagues is used in this research to develop training scenarios with measurable evaluation criteria.

3. Multi-PLC Training Platform

This section describes the design considerations and implementation details of the multi programmable logic controller training platform.

3.1 Design Considerations

The training platform is designed to incorporate multiple programmable logic controller models, thereby emphasizing the differences between the individual programmable logic controllers.

Requirements. The training platform is intended to be reasonably inexpensive and mobile so that training can be conducted at multiple locations. A replica of a jail was created within a 55.32 cm × 42.39 cm × 26.97 cm Pelican 1610 case using genuine components and realistic programs. To enhance realism, the components were selected based on the design of an actual jail in the United States. Ladder logic programs for the programmable logic controllers were created to implement the same operations as the real jail. Any one of the three programmable logic controllers can be selected by the training administrator to be active at a given time. Figure 1 shows the completed training platform.

The training platform is designed to meet five criteria:

- Incorporate physical components.
- Incorporate cyber manipulation principles.
- Incorporate response coordination techniques.
- Provide hands-on experience.
- Implement effective training scenarios with measurable training evaluation metrics.

Components. Table 1 lists the main components of the platform. The pushbuttons, indicator lights and turnkey replicate components that are found

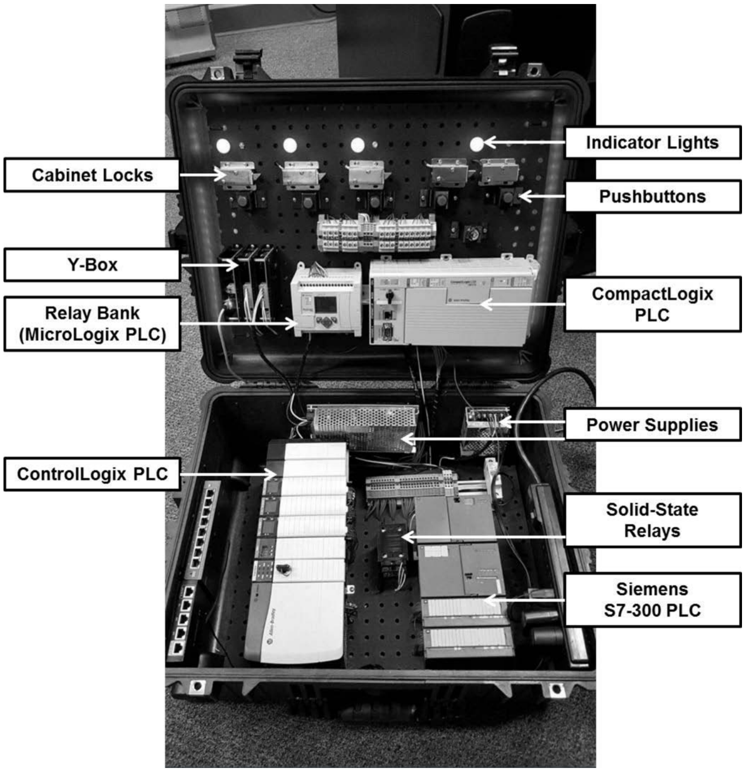


Figure 1. Training platform.

Table 1. Training platform components.

Component	Quantity	Component	Quantity
Cabinet Locks	5	Pushbuttons	5
Relays (Electromechanical)	5	Red Lights	4
Relays (Solid State)	3	Peg Boards	2
Power Supplies (12 V)	1	Power Supplies (24 V)	1
Network Switches	1	Routers	1
Circuit Breakers (10 A)	1	Turnkeys	1
Power Strips	1	CompactLogix PLCs	1
Siemens S7-300 PLCs	1	ControlLogix PLCs	1
Y-Boxes	1		

on the control panel at a guard station in a jail. Indicator lights are controlled based on inputs from a sensor that detects if the cell door is secure. Because the exercise platform does not have actual doors, this sensor is simulated in the

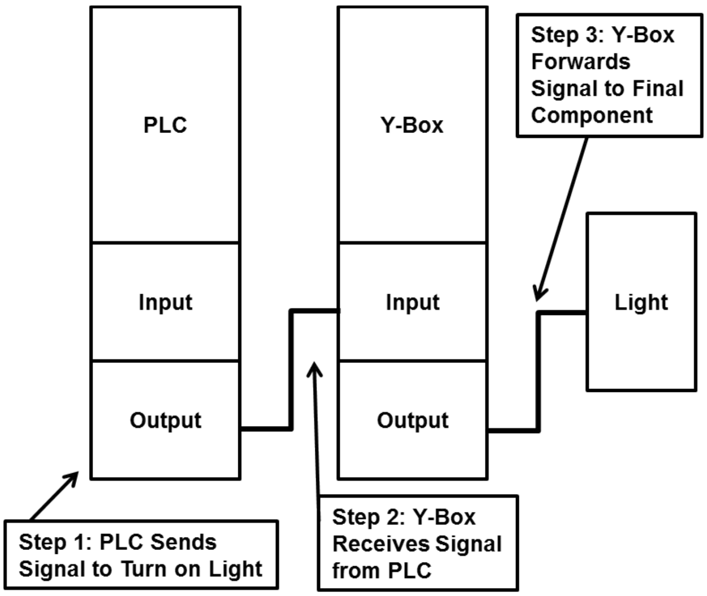

PLC
Input
Output
Y-Box
Input
Output
Light
Step 1: PLC Sends Signal to Turn on Light
Step 2: Y-Box Receives Signal from PLC
Step 3: Y-Box Forwards Signal to Final Component

Figure 2. Wiring diagram for lights.

Y-Box code so that the Y-Box sends the sensor signals to the programmable logic controller. This is the only simulated component in the training platform. Interested readers are referred to [16] for a detailed description of the Y-Box.

The first programmable logic controller is a CompactLogix model L23E. The second is a Siemens S7-300 with one digital input module and one digital output module. The third is a ControlLogix programmable logic controller that also has one digital input module and one digital output module. Additionally, the ControlLogix programmable logic controller does not have a built-in Ethernet or CPU module; therefore, a Logix5555 CPU module and an EWEB Ethernet module are included in the seven-slot chassis. The Y-Box consists of a CPU module with one digital input module and one digital output module. The five electromechanical relays are implemented using a Micrologix programmable logic controller.

Wiring. To take full advantage of the Y-Box technology, the physical components are not wired directly to the programmable logic controller. Instead, different wiring schemes are adopted. For some applications, the Y-Box can be thought of as a "man-in-the-middle" device that receives electrical signals from the programmable logic controllers and other components, and forwards the signals to their destinations. The wiring schemes used for the lights and pushbuttons are shown in Figures 2 and 3, respectively.

The cabinet locks are wired differently because the Y-Box cannot provide sufficient electrical current to disengage the lock. In this case, the Y-Box is used

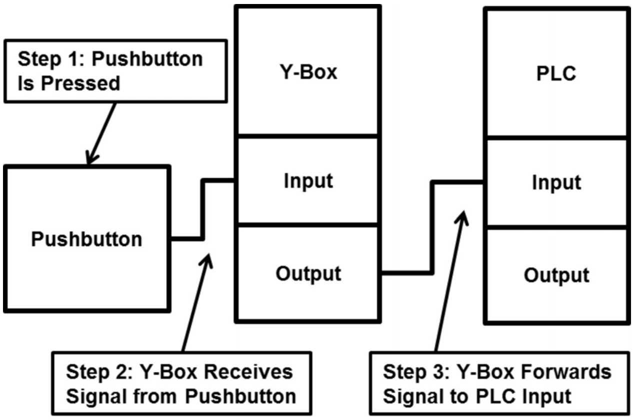

Step 1: Pushbutton Is Pressed
Pushbutton
Y-Box
Input
Output
PLC
Input
Output
Step 2: Y-Box Receives Signal from Pushbutton
Step 3: Y-Box Forwards Signal to PLC Input

Figure 3. Wiring diagram for pushbuttons.

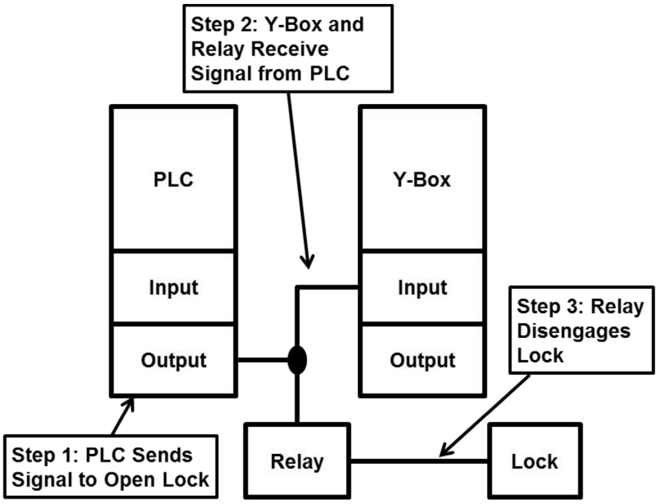

Step 2: Y-Box and Relay Receive Signal from PLC
PLC
Input
Output
Y-Box
Input
Output
Step 3: Relay Disengages Lock
Step 1: PLC Sends Signal to Open Lock
Relay
Lock

Figure 4. Wiring diagram for locks.

to monitor the signal on the line between the programmable logic controller and the relay, which ultimately powers the lock. This is accomplished by daisy chaining the programmable logic controller outputs from the relay to the Y-Box. Figure 4 shows the wiring diagram.

The other wiring challenge involves connecting all three programmable logic controllers as a single set of components. This requires the inputs and outputs of the three programmable logic controllers to be synchronized and wired together. Figure 5 shows the wiring diagram for an indicator light. The outputs of all

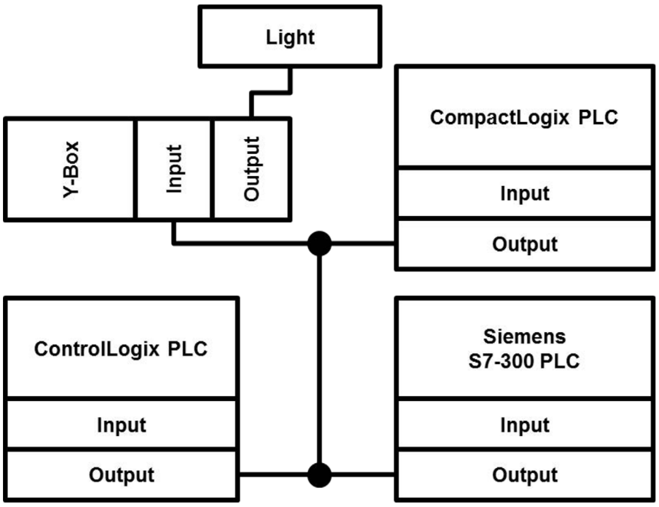


Figure 5. Wiring diagram for programmable logic controller inputs and outputs.

three programmable logic controllers are tied together, ultimately leading to a single wire that is connected to the Y-Box input module.

Programmable Logic Controller Selection. The value of having three programmable logic controllers in a single platform is lost if they cannot all assume full control over the components. Once again, the Y-Box can be leveraged to control the flow of electricity to an individual programmable logic controller while denying power to the other programmable logic controllers. This is accomplished using solid state relays controlled by a Y-Box digital output. When the solid state relay receives the control signal from the Y-Box, power is allowed to flow through the relay to its corresponding programmable logic controller, activating the device. This is the case for the ControlLogix and Siemens S7-300 programmable logic controllers. The CompactLogix programmable logic controller is slightly different from the other two controllers because it operates on 24 V DC. In this case, the relay controls power to a 24 V power supply that, in turn, powers the CompactLogix programmable logic controller. Figure 6 shows the wiring of the relays. Note that the figure is simplified and does not show the 24 V power supply.

3.2 Exercise Layout

Figure 7 shows a possible exercise layout. The following paragraphs describe the functions of each segment of the three tables in the exercise layout.

White Cell Table. An effective white cell should be aware of all the activities performed by the training participants. The simulation terminal is a

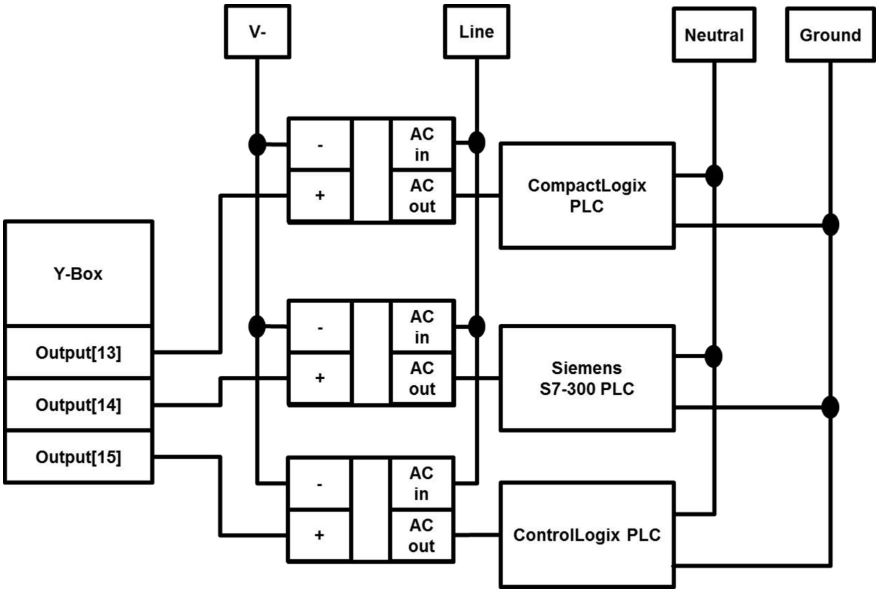


Figure 6. Wiring diagram for programmable logic controller selection.

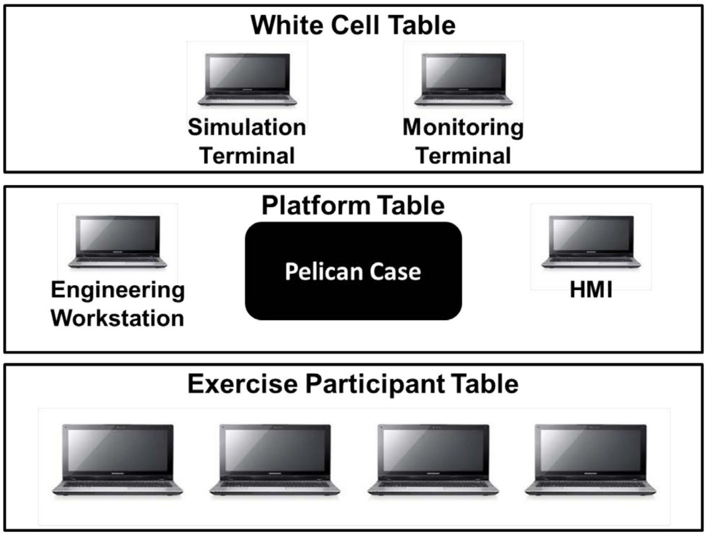


Figure 7. Exercise layout.

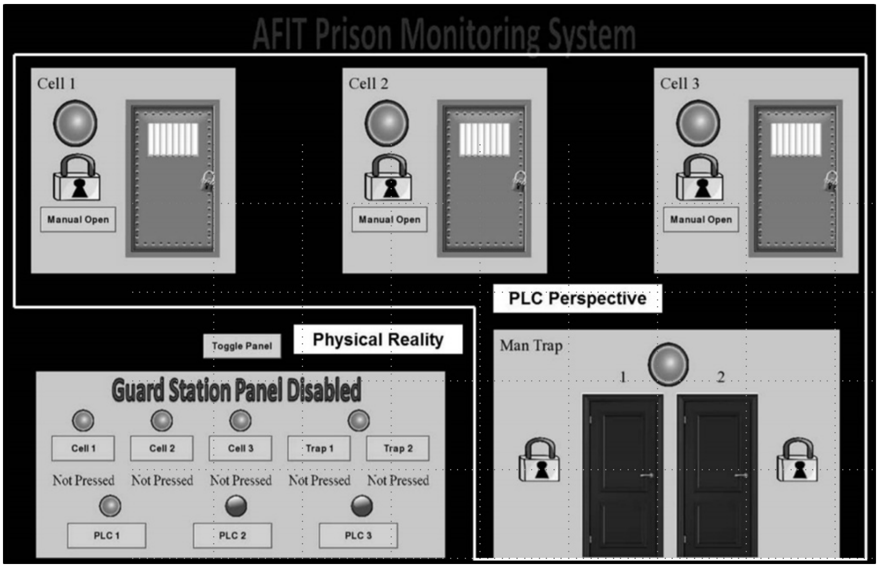


Figure 8. White cell view.

machine that runs the Y-Box software implemented in Python. The monitoring terminal runs network monitoring software and is connected to a mirrored port on the switch to capture all the traffic during the exercise. During the exercise, the white cell should watch the engineering workstation, human-machine interface, network traffic and participants. Furthermore, the white cell should watch the Y-Box software.

If a training scenario involves malware that fools the human-machine interface, it would be difficult for the white cell to maintain awareness of the state of the physical system during the training. The Y-Box overcomes this problem because it is aware of the true states of the locks, lights and pushbuttons. Figure 8 shows the Y-Box software view of the system with the physical reality of the system as well as the system from the perspective of the programmable logic controller. The figure also shows the programmable logic controller selection buttons that dictate which controller is active at any given time. It should be noted that the buttons in the software are capable of overriding the physical components in the Pelican case, enabling the white cell to manage all the aspects of the exercise at all times.

Platform Table. An engineering workstation and a human-machine interface operate beside the platform. The training platform is contained in a Pelican 1610 case. Inside the case is a fully-functioning replica of a guard station panel that closely mimics what would be found in an actual jail. Additionally, the case contains five cabinet locks, three of which represent jail cells and two that serve as mantraps. Each jail cell has a corresponding light that indicates whether or not the cell is secure. The mantrap has only one light that indicates it is secure only when both its doors are closed and locked. The programmable logic controllers are connected to a network switch housed in the Pelican case.

Exercise Participant Table. Training participants are seated at the exercise participant table within sight of the training platform as shown in Figure 7. Laptops are provided with standard security tools (e.g., Kali Linux and Security Onion) as well as virtual machines containing proprietary software applications that interact with the programmable logic controllers. A participant may also bring any tools that he/she feels are appropriate to the exercise. From his/her table, the participant is connected to a network switch in the Pelican case and is free to interact with the platform to complete the assigned tasks. Note that the layout can be adjusted to accommodate different rooms and table sizes, and additional network switches can be added to accommodate more participants.

4. Training Scenario

One of the most important steps in securing an industrial control system is to properly segment the control network [15]. This simple task demonstrates the different implementations of similar features by the three programmable logic controllers. For this reason, a beginner-level scenario was first designed for the multi programmable logic controller training platform.

Because this scenario is intended to demonstrate the differences in programmable logic controller implementations, the scenario is simplified in several ways. First, the initial IP addresses of all three programmable logic controllers are the same (`192.168.108.205`). The new IP addresses that the participants load on the programmable logic controllers are also the same (`10.1.4.205`). Additionally, the participants need not concern themselves about whether or not changing the IP address would impact the functionality of other industrial control system components. In this scenario, it is assumed that all the other issues regarding components that are dependent on the programmable logic controller IP address have already been reconciled. More complicated scenarios can be developed to demonstrate the potential second- and third-order effects that can occur from this process.

The final scenario simplification is that no password protections exist for any of the files. In a real-world environment, it is reasonable to expect that a cyber first responder would be provided the necessary access by an asset owner to perform the tasks. While credentials are required by the ControlLogix administrative web server, they were reset to the factory-default credentials for demonstration purposes. Finally, the training scenario is implemented using the framework proposed by Yoon et al. [17].

The following are the details of the training scenario:

- **Objective:** Isolate a programmable logic controller that is located in an improperly segregated network.

- **Description:** The participant uses the relevant software and appropriate technique to change the programmable logic controller IP address from `192.168.108.205` to the new IP address `10.1.4.205`.
- **Type:** Network reconfiguration.
- **Evaluation Criteria:**
  - Identify the relevant software within five minutes.
  - Identify the appropriate technique for updating the IP address within ten minutes.
  - Update and confirm the new IP address within fifteen minutes.
  - Perform all the activities with minimal programmable logic controller downtime.
- **References:** NIST SP 800-82, Rockwell Automation EWEB module documentation, Siemens S7-300 documentation and Rockwell Automation CompactLogix documentation.

4.1 Segmentation Using a CompactLogix PLC

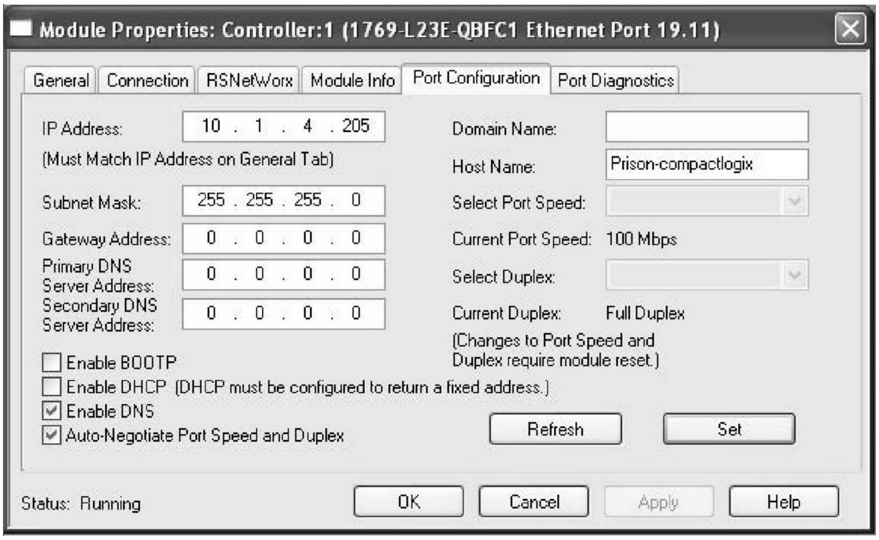
The first task for the participant is to interact with the CompactLogix programmable logic controller. Updating the IP address of this programmable logic controller involves the following steps:

- **Step 1:** Open the appropriate RSLogix5000 project file and access the Ethernet module properties.
- **Step 2:** Under the Port Configuration tab, enter the new IP address in the appropriate field and click Set. Confirm the update in the dialogue windows that appear.
- **Step 3:** Ensure connectivity to the new IP address (this may require routing or changing the IP address of the engineering workstation).

Step 1 requires the identification of the RSLogix5000 software. Step 2 involves the identification and update of the IP address. Step 3 ensures that the programmable logic controller is available. Since the CompactLogix programmable logic controller can have its IP address updated without downtime, the participant should receive a lower evaluation if the programmable logic controller resets or faults. Figure 9 shows the relevant dialogue window for updating the IP address.

4.2 Segmentation Using a Siemens PLC

After the participant has completed the assigned task on the CompactLogix programmable logic controller, the instructor switches control to the Siemens programmable logic controller. After the programmable logic controller has


Module Properties: Controller:1 (1769-L23E-QBFC1 Ethernet Port 19.11)

General | Connection | RSNetWorx | Module Info | Port Configuration | Port Diagnostics

IP Address: 10 . 1 . 4 . 205
(Must Match IP Address on General Tab)
Subnet Mask: 255 . 255 . 255 . 0
Gateway Address: 0 . 0 . 0 . 0
Primary DNS Server Address: 0 . 0 . 0 . 0
Secondary DNS Server Address: 0 . 0 . 0 . 0

☐ Enable BOOTP
☐ Enable DHCP (DHCP must be configured to return a fixed address.)
☑ Enable DNS
☑ Auto-Negotiate Port Speed and Duplex

Domain Name:
Host Name: Prison-compactlogix
Select Port Speed:
Current Port Speed: 100 Mbps
Select Duplex:
Current Duplex: Full Duplex
(Changes to Port Speed and Duplex require module reset.)

Status: Running

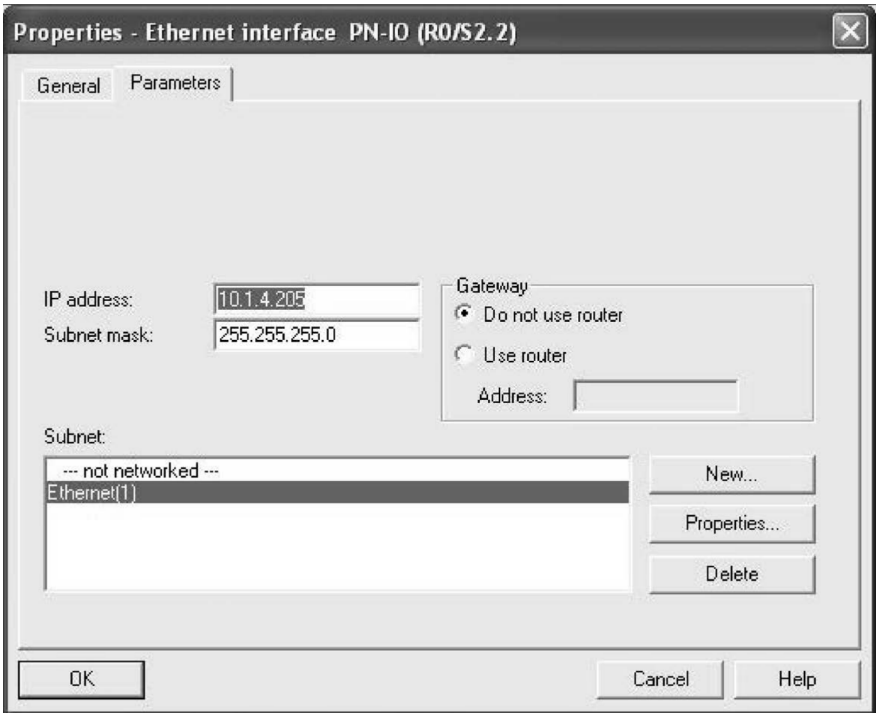
Figure 9. IP address update of the CompactLogix PLC using RSLogix5000 software.

booted, the participant must change the IP address of the Siemens controller to an isolated subnet. This is the first time that the participant is exposed to the differences between the programmable logic controllers. In particular, the programming environment for the Siemens programmable logic controller is different from that of the CompactLogix controller.

The following steps are required to complete the task on the Siemens programmable logic controller:

- **Step 1:** Open the SIMATIC project file.
- **Step 2:** Access the HW Config tab in the SIMATIC software and navigate to the Object Properties of the PN-IO module.
- **Step 3:** Under the General tab, select Properties and enter the new IP address as shown in Figure 10.
- **Step 4:** Download the new configuration to the programmable logic controller using the old IP address as the target station.
- **Step 5:** Ensure connectivity to the new IP address (this may require routing or changing the IP address of the engineering workstation).

Step 1 involves the identification of the SIMATIC software. Steps 2 through 4 involve the identification of the appropriate technique to update the IP address. Step 5 ensures that the programmable logic controller completes the download successfully with minimal downtime.


Properties - Ethernet interface PN-IO (R0/S2.2)

IP address: 10.1.4.205
Subnet mask: 255.255.255.0

Gateway: [✓ Do not use router] Use router
Address:

Subnet:
--- not networked ---
Ethernet(1)

Figure 10. IP address update of the Siemens S7-300 PLC using SIMATIC software.

4.3 Segmentation Using a ControlLogix PLC

The participant now performs the required tasks using an implementation that is unique to the ControlLogix programmable logic controller. The ControlLogix programmable logic controller is equipped with a 1756-EWEB Enhanced Web Server Module that provides an administrative web interface to manage the programmable logic controller.

The following steps are involved:

- **Step 1:** Open a web browser and navigate to the IP address of the programmable logic controller.

- **Step 2:** Open the Network Configuration tab, enter the new IP address in the appropriate field and apply the changes.

- **Step 3:** Confirm that the new address is correct. Upon completion, a message is displayed that notifies the participant of the new IP address.

- **Step 4:** Ensure connectivity to the new IP address (this may require routing or changing the IP address of the engineering workstation).

Step 1 requires the participant to identify the web interface provided by the EWEB module. Step 2 involves the identification and application of the appropriate technique to perform the update. Steps 3 and 4 confirm that the change is successful. There should be no downtime when performing this task on the ControlLogix programmable logic controller.

4.4 Scenario Selection and Alternate Scenarios

The network segmentation scenario was chosen because it effectively demonstrates how different programmable logic controllers often require different techniques to perform the same task. These differences emphasize the value of a cyber first responder having experience on a variety of programmable logic controllers. The scenario incorporates several of the training items proposed by Butts and Glover [3] and implemented in the framework of Yoon et al. [17].

Note that segmenting a network is only one of many tasks that a cyber first responder may need to complete in his/her line of work and it is certainly not a difficult task. Other examples such as modifying ladder logic programs, updating firmware and applying patches are also unique to different programmable logic controllers and have varying levels of difficulty. Because the training platform incorporates real programmable logic controllers, a number of scenarios, including scenarios involving advanced topics, could be implemented with minimal reconfiguration.

Two scenarios that showcase the flexibility of the multi programmable logic controller platform are described below.

Analysis of a Malicious Implant in PLC Firmware

- **Objective:** Reverse engineer the firmware to identify and analyze a malicious implant.
- **Description:** The participant uses the relevant software and appropriate techniques to extract programmable logic controller firmware from the device and identifies malicious code given the correct version of the firmware. The participant then determines the exact functionality and purpose of the malicious code.
- **Type:** Reverse engineering.
- **Evaluation Criteria:**
  - Identify the malicious code within 45 minutes.
  - Restore the programmable logic controller firmware within 20 minutes.
  - Analyze the malicious code within 90 minutes.
- **References:** Rockwell Automation ControlLogix documentation, Siemens S7-300 documentation, Rockwell Automation CompactLogix documentation.

The reverse engineering scenario further emphasizes the differences between programmable logic controllers by requiring a participant to extract and analyze firmware from the devices (see [1] for details about reverse engineering industrial control devices). The scenario also brings up the important point that there are often similarities between programmable logic controllers. Specifically, the CompactLogix and ControlLogix programmable logic controllers have very similar firmware despite being different models. The reverse engineering scenario can be implemented with malware samples of varying complexity to accommodate and/or enhance participant abilities.

Digital Forensics of a Malfunctioning PLC

- **Objective:** Determine the cause of a malfunctioning programmable logic controller.
- **Description:** The participant uses the relevant software and appropriate techniques to identify the root cause of the programmable logic controller behavior.
- **Type:** Digital forensics.
- **Evaluation Criteria:**
  - Collect sufficient data to perform digital forensics within 30 minutes.
  - Identify the cause of the malfunction within 45 minutes.
  - Identify the corrective action within 60 minutes.
- **References:** Rockwell Automation ControlLogix documentation, Siemens S7-300 documentation, Rockwell Automation CompactLogix documentation.

The digital forensic scenario involves similar tasks as the reverse engineering scenario. It also shows that the process for conducting digital forensics on industrial control systems is identical for different programmable logic controllers (see [6] for details about this process). Despite using the same process, the data being analyzed (e.g., ladder logic program, network traffic and log files) would be different because of the operational differences between the programmable logic controllers. These operational differences mean that a cyber first responder in a real-world situation will have to focus on specific, contextualized pieces of information to effectively analyze the root cause of a malfunctioning programmable logic controller. The difficulty of this scenario can be modulated by inducing different types of programmable logic controller malfunctions ranging from simple faults to advanced malware infections. The scenario can be repeated multiple times with different symptoms to increase the participant's exposure to a variety of malfunctions.

5. Results

This section describes the principal results pertaining to training platform development.

5.1 Hardware Verification

The initial debugging of the wiring, Y-Box code and programmable logic controller code involved interacting with the physical components mounted in the Pelican case and confirming that the Y-Box and programmable logic controller behaved as intended. This process revealed that some of the variables had been coded incorrectly in the ladder logic. These variables needed their memory addresses reassigned to correct their mapping to the programmable logic controller inputs and outputs. The Y-Box software was also verified, confirming the behavior of the physical components and that the software could override the physical components to control the case autonomously.

5.2 Reliability Test

After confirming that the components were behaving correctly, an automated Python script tested the reliability of the training platform. The test involved the following steps:

- **Step 1:** Select the programmable logic controller.
- **Step 2:** Power up the selected programmable logic controller.
- **Step 3:** Wait 25 seconds for the programmable logic controller to activate.
- **Step 4:** Test all the buttons, locks and lights for functionality.
- **Step 5:** Shut down the programmable logic controller.
- **Step 6:** Reset the Y-Box parameters.

Initial runs of the reliability test encountered failures because the Python test code sent commands too quickly, which did not provide the Y-Box with adequate time to update its inputs and outputs. This issue was resolved by including "wait" commands of 25 seconds for the programmable logic controller to boot and varying amounts of time between 0.4 and 2.0 seconds for other functions (e.g., button presses, lock status updates and indicator light updates).

Step 4 is the key part of the reliability test. This step starts with the first jail cell and simulates a button press. The script then checks that the programmable logic controller responds appropriately before repeating the process for the other two cells. Next, the test code evaluates the mantrap by testing every possible combination of button presses and confirming the responses. Finally, it simulates a button press on the cell once again with the panel disabled. In this situation, the lock should not disengage and the test is considered to have failed if it does.

Table 2. Reliability test results.

Controller	**Trials**	**Failures**
CompactLogix PLC	50	0
Siemens S7-300 PLC	50	0
ControlLogix PLC	50	0
Total	150	0

The wiring scheme with the Y-Box enables electrical signals to be sent to the programmable logic controller without having to receive signals from the buttons. Furthermore, the state of the panel turnkey can be overridden by the Y-Box itself. These enable each of the functions to be simulated by the Y-Box alone. In the future, the test can be fully automated in a manner that is transparent to the programmable logic controller because the controller receives the same signals as it would under normal operation. To prevent a failure in one iteration from impacting the results of the next iteration, Step 6 resets all the Y-Box values to default values. A total of 150 iterations were performed, with each programmable logic controller tested 50 times. Table 2 shows the reliability test results.

5.3 Timing Test

Incorporating multiple programmable logic controllers in a single platform is useless if the switching between the programmable logic controllers takes a prohibitive amount of time. Ideally, control of the system should be switched from one programmable logic controller to another within the amount of time that it takes for the participant to be prepared for the next task. To evaluate this metric, the time required for each programmable logic controller to fully power up was measured and recorded by an automated Python script, which performed the following steps:

- **Step 1:** Select the programmable logic controller.
- **Step 2:** Send power to the programmable logic controller and start the timer.
- **Step 3:** Send the input command to the programmable logic controller.
- **Step 4:** Wait for the programmable logic controller to react to the input and stop the timer upon completion.
- **Step 5:** Shut down the programmable logic controller.
- **Step 6:** Reset the Y-Box parameters.

In the timing test, it is only necessary to examine the amount of time that it takes for the programmable logic controller to become responsive to an input.

Table 3. Programmable logic controller startup times (seconds).

Controller	Minimum	Maximum	Mean	Std. Dev.
CompactLogix PLC	19.547	19.688	19.629	0.030
Siemens S7-300 PLC	14.782	15.172	15.060	0.062
ControlLogix PLC	4.797	4.843	4.816	0.010

Table 3 presents the results of the timing test, which were also determined over the course of 150 trials (50 trials per programmable logic controller). The results show that the programmable logic controllers have significantly different boot times, but are very consistent across all the trials.

5.4 Functional Analysis Criteria

The multi programmable logic controller training platform is designed to meet the following criteria:

- Incorporate physical components.
- Incorporate cyber manipulation principles.
- Incorporate response coordination techniques.
- Provide hands-on experience.
- Implement effective training scenarios with measurable training evaluation metrics.

The replication of the jail system, including the programmable logic controllers running realistic ladder logic programs and the pushbuttons, locks, lights and turnkey, enables a training participant to experience many of the physical components involved in a real-world system. This addresses the need for cyber first responders to understand the physical processes underlying an industrial control system.

By creating scenarios that incorporate concepts such as reverse engineering and digital forensics, cyber manipulation principles can be effectively taught to cyber first responders. Participants can be exposed to topics like access vectors, vulnerability analysis, implanting malware, manipulating physical processes and defensive mechanisms, all of which are considered to be cyber manipulation principles [3].

Skills involved in response coordination include the ability to prioritize system components, identify attacks and understand the steps required to appropriately defend and restore a system to normal operation. Each of these skills is practiced in some way by the scenarios described in this work and can be enhanced by designing alternate scenarios using the training platform.

The entire training platform is self-contained and all the relevant software is embodied in easily-recoverable virtual machines. This enables training participants to have full access to the system for hands-on exercises without being concerned about potential damage to the system. Cyber first responders can benefit most from hands-on exercises that give them experience with realistic systems that incorporate real hardware.

5.5 Limitations

The multi programmable logic controller platform has certain limitations. First, the platform does not incorporate any analog components. Analog signals are more complicated than digital signals from a programming perspective and should be incorporated to enhance learning experiences.

Another limitation of the platform is the scale of the replica system. In a full-sized jail, there are many more components, including additional doors and alarms. The design choice leads to the trade-off between training platform cost and scale. The final limitation is that the training platform is limited to the use of one programmable logic controller at a time.

6. Conclusions

Effective industrial control system platforms are necessary for cyber first responder training. Unfortunately, most testbeds are designed for research and development activities and are not available for training purposes. Furthermore, testbeds developed for training purposes tend to be very expensive or substitute device simulations in place of genuine components. Ideal testbeds incorporate full-scale, fully-operational industrial control systems with training scenarios that impart the unique skills needed by cyber first responders to operate in real-world environments. The cost of such a testbed is prohibitively high; however, the multi programmable logic controller training platform developed in this research can impart many of the desired skills at a fraction of the cost. Furthermore, the multi programmable logic controller training platform can support scenarios ranging from basic tasks such as changing an IP address to advanced tasks involving reverse engineering and digital forensics.

The multiple programmable logic controllers incorporated in the platform provide opportunities for trainees to experience different devices, protocols and programming environments. Similar to medical students, who go through a rigorous curriculum with hands-on, real-world learning experiences, the multi programmable logic controller training platform enables cyber first responders to gain valuable hands-on experience with real control systems to significantly enhance their cyber defense skills.

Note that the views expressed in this chapter are those of the authors and do not reflect the official policy or position of the U.S. Air Force, U.S. Army, U.S. Department of Defense or U.S. Government.

Acknowledgement

This research was partially supported by the U.S. Department of Homeland Security Industrial Control Systems Cyber Emergency Response Team (ICS-CERT).

References

[1] Z. Basnight, Firmware Counterfeiting and Modification Attacks on Programmable Logic Controllers, M.S. Thesis, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2013.

[2] A. Bauer, Talking with your doctor about prognosis, *Cancer.Net*, August 14, 2014.

[3] J. Butts and M. Glover, How industrial control system security training is falling short, in *Critical Infrastructure Protection IX*, M. Rice and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 135–149, 2015.

[4] R. Candell, T. Zimmerman and K. Stouffer, An Industrial Control System Cybersecurity Performance Testbed, NISTIR 8089, National Institute of Standards and Technology, Gaithersburg, Maryland, 2015.

[5] Department of Psychiatry, New York University School of Medicine, Medical Student Education Program in Psychiatry, New York University, New York (`www.med.nyu.edu/psych/education/medical-student-education`), 2017.

[6] L. Folkerth, Forensic Analysis of Industrial Control Systems, InfoSec Reading Room, SANS Institute, Bethesda, Maryland (`www.sans.org/reading-room/whitepapers/forensics/forensic-analysis-industrial-control-systems-36277`), 2015.

[7] Idaho National Laboratory, INL Cyber Security Research: Defending the Network Against Hackers, Fact Sheets: 21st Century Science and Technology, Idaho Falls, Idaho (`www.inl.gov/research/inl-cyber-security-research`), 2014.

[8] Idaho National Laboratory, University Partnerships, Idaho Falls, Idaho (`www.inl.gov/inl-initiatives/education`), 2016.

[9] International Information System Security Certification Consortium ((ISC)^2), (ISC)^2 Information Security Certification Programs, Clearwater, Florida (`www.isc2.org/credentials/default.aspx`), 2016.

[10] Sandia National Laboratories, Distributed Energy Technology Laboratory, Albuquerque, New Mexico (`energy.sandia.gov/wp-content/gallery/uploads/DETL_Factsheet_SAND2010-3643_Aug2011.pdf`), 2011.

[11] Sandia National Laboratories, SCADA Testbeds, Albuquerque, New Mexico (`energy.sandia.gov/energy/ssrei/gridmod/cyber-security-for-electric-infrastructure/scada-systems/testbeds`), 2016.

[12] SANS Institute, ICS Training Courses, Bethesda, Maryland (`ics.sans.org/training/courses`), 2017.

[13] SANS Institute, SEC562: CyberCity Hands-On Kinetic Cyber Range Exercise, Bethesda, Maryland (`www.sans.org/course/cybercity-hands-on-kinetic-cyber-range-exercise`), 2017.

[14] E. Skoudis, How to build a completely hackable city in five steps: And why you should build your skills in this arena, presented at *SANS Pen Test Hackfest*, 2013.

[15] K. Stouffer, J. Falco and K. Scarfone, Guide to Industrial Control Systems (ICS) Security, NIST Special Publication 800-82, National Institute of Standards and Technology, Gaithersburg, Maryland, 2011.

[16] J. Yoon, Framework for Evaluating the Readiness of Cyber First Responders for Industrial Control Systems, M.S. Thesis, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2016.

[17] J. Yoon, S. Dunlap, J. Butts, M. Rice and B. Ramsey, Evaluating the readiness of cyber first responders responsible for critical infrastructure protection, *International Journal of Critical Infrastructure Protection*, vol. 13, pp. 19–27, 2016.

IV

INTERNET OF THINGS SECURITY

Chapter 15

DEFENDING BUILDING AUTOMATION SYSTEMS USING DECOY NETWORKS

Caleb Mays, Mason Rice, Benjamin Ramsey, John Pecarina and Barry Mullins

Abstract The Internet of Things (IoT) and home and building automation systems are growing fields. Many automation networks use proprietary protocols and few publications have evaluated their security. INSTEON is a leading Internet of Things protocol for home and building automation and, like other proprietary protocols, little research is available relating to its vulnerabilities. This chapter presents techniques for analyzing INSTEON traffic and defending INSTEON networks using virtual decoys. By using a software-defined radio, the packet capture rate for INSTEON traffic is increased from approximately 40% to almost 75% compared with previous research efforts. Additionally, a virtual decoy network has been designed and tested for authenticity and targetability to better protect home and building automation systems.

Keywords: Internet of Things, home and building automation, honeypots

1. Introduction

The family was in the car ready for a great weekend getaway. They had planned a three-day camping trip – fishing, hiking, s'mores – the whole experience. The father was pulling away from the home and pushed the button on his phone to close the garage door. The mother used her smart phone to verify that the external doors were locked and that the thermostat was set. The father maneuvered the family car past parked vehicles on the street. The family started what they all hoped would be a great weekend.

The weekend was great! The weather was perfect; not too hot, not too cold and no rain. When the family returned home and their car approached the driveway, everything looked normal. The garage door was down, the doors were shut, there were no broken windows and no signs of a break-in. The inside, however, was a different story. The home had been burgled. The TV and entertainment systems were gone. Jewelry was missing. Family heirlooms

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 297–317, 2017.
https://doi.org/10.1007/978-3-319-70395-4_15

had been stolen. Somebody was able to open the garage door. The family thought that their garage door could only have been opened and closed via the application on the phone. How did this happen?

Little did the family know that, as they drove away, the intruder was in one of the cars parked on the street. The burglar noticed the car-top carrier, inferring the family would be gone all weekend. The intruder used an inexpensive radio and computer to capture the wireless commands used to open and close the garage door. With a few simple keystrokes, he replayed the command to open the garage door and walked into the home.

Could the family have done anything more to secure their home? Home automation devices are, no doubt, convenient, but automation networks lack proper security. Users are unknowingly making themselves more vulnerable to intruders by using home automation products.

INSTEON is a leading protocol for home and building automation and, like other proprietary protocols, little research has been published about its vulnerabilities. This chapter presents a technique for analyzing INSTEON traffic and defending INSTEON networks using virtual decoys.

2. Background

The Internet of Things (IoT) and home and building automation are growing fields. These technologies are used to connect to smart appliances in homes and buildings from anywhere in the world. Homeowners can control their lights, ensure their front doors are locked and view thermostat settings on their mobile devices. Building managers can control access and industrial HVAC settings with a few mouse clicks. While home and building automation is a relatively new field, a report by Transparency Market Research [19] projects that the global industry will grow to $21.6 billion by 2020. Gartner [6] predicts that Internet of Things devices – for home and business use – will rise from 6 billion to 20 billion by 2020.

2.1 Automation Technologies

The home and building automation market incorporates many technologies and protocols (e.g., Wi-Fi, Bluetooth, ZigBee, Z-Wave and INSTEON). Multiple studies have compared the technical specifications and capabilities of these automation technologies [1, 8, 21], but very few researchers have evaluated the security of proprietary Internet of Things protocols. The lack of security research leaves homes and businesses vulnerable and dangerously accessible to intruders.

Wi-Fi. Wi-Fi is a popular technology that is broadly incorporated in homes and businesses. Amazon's Echo and the Nest line of products use this technology. Wi-Fi attacks such as de-authentication and rogue access point attacks are well-known and tools are incorporated in the Kali version of Linux specifically for evaluating Wi-Fi security. Wi-Fi can be secured through strong encryption techniques. Advances in Wi-Fi security will certainly benefit Wi-Fi-enabled Internet of Things devices.

Bluetooth. Rose and Ramsey [15] have demonstrated that many vendors do not implement optional Bluetooth security features (e.g., encryption by properly pairing devices). Improper Bluetooth pairing leads to confidentiality problems. Rose and Ramsey have passively sniffed passwords for several locks. They have documented several security flaws and have developed proof-of-concept tools that open several Bluetooth-enabled locks. They also note that, when presented with improper commands (e.g., unrecognized characters), many locks fail in a non-recoverable state, resulting in denial-of-service to users.

ZigBee and Z-Wave. Hall and Ramsey [9, 12] have researched the ZigBee and Z-Wave protocols and have developed tools for securing automation networks that use these protocols. The Philips Hue brand of lights is one product that uses ZigBee. Hall and Ramsey capitalized on Z-Wave broadcast commands to enumerate Z-Wave networks. Their research demonstrates the over-availability problem and general lack of confidentiality in the Z-Wave protocol. Additionally, they discovered problems similar to those encountered in Bluetooth systems, where manufacturers did not implement encryption and other security measures in a proper manner.

INSTEON. INSTEON has been producing home and building automation devices for more than ten years and recently announced plans to integrate with Revention's point-of-sale system for building automation control [13]. A wide range of devices, including LED lights, dimmable light switches, open/close sensors and security cameras, are available. INSTEON devices enjoy general acceptance within the Internet of Things community through interoperability with the Amazon Echo and Sonos home surround sound system. Applications on Apple iOS, Google Android and Microsoft Windows 8/10 provide interfaces for monitoring and modifying the devices. Figure 1 presents a schematic diagram of a basic INSTEON network.

2.2 Honeypots for Building Automation Defense

The use of honeypots for network defense is not new. The development of traditional honeypots prior to 2006 was constrained by costly hardware. Virtual technologies have since made traditional honeypots inexpensive, convenient and prevalent. Large research honeypots have been used to capture malicious software in information technology networks [2, 11].

Using honeypots to defend industrial control and supervisory control and data acquisition (SCADA) networks is a relatively new concept. The Honeynet Project [14] has made significant progress in developing an open source honeypot for industrial control networks. Winn et al. [20] have researched industrial control system honeypots and have extended the `honeyd` program to `honeyd+` to make it more authentic. They discuss two levels of interaction displayed by

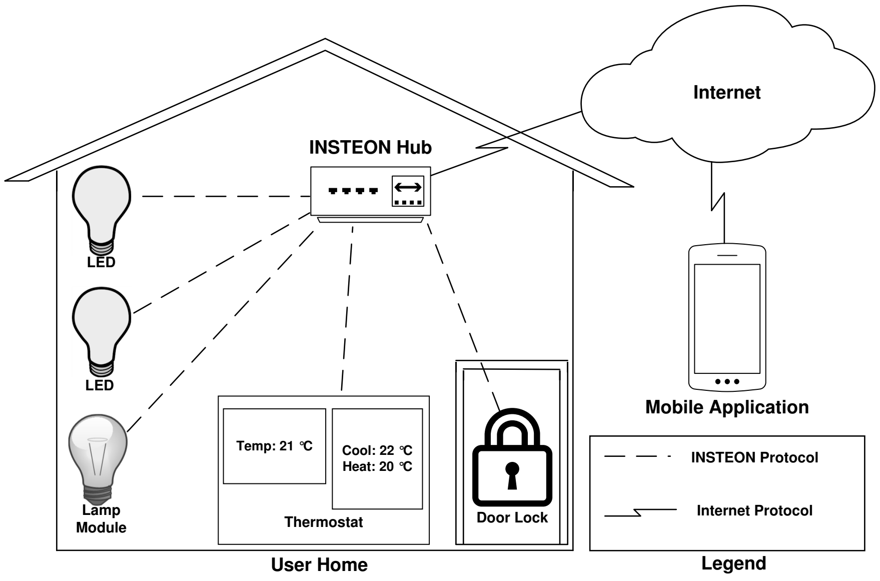


Figure 1. INSTEON network.

a honeypot: (i) high-level interaction; and (ii) low-level interaction. They also argue that the targetability of a honeypot needs to match the intended network and that the decoys must appear authentic enough for an attacker to target the fake devices instead of the real devices. Girtz et al. [7] have enhanced the performance of `honeyd+` by developing an application layer emulator.

Schneier [16] uses the terms Internet of Things devices and cyber-physical systems interchangeably. He correctly writes that the Internet of Things has given the Internet "hands and feet" with the ability to change physical systems through cyber means. Meanwhile, research in the area of Internet of Things security and industrial control system honeypots are converging. This chapter describes the design and implementation of a honeypot with authenticity and targetability characteristics that can protect an INSTEON home automation network.

3. Understanding INSTEON

There are three primary sources of information about the INSTEON protocol. The first two documents are published by INSTEON [4, 18]. The third source is a presentation and software produced by security researchers, Shipley and Gooler [17].

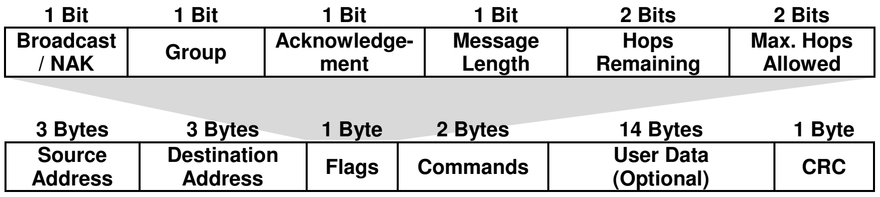


Figure 2. INSTEON packet structure specified in [18].

3.1 INSTEON Documentation

INSTEON created its own communications protocol and published many of the specifications. Two documents, *INSTEON: The Details* [18] and *INSTEON: Developer's Guide* [4], are intended to inform developers, security professionals and users about the protocol.

INSTEON: The Details. *INSTEON: The Details* [18] describes the specifications of INSTEON's wireless and power-line protocol settings, packet structure, hopping mechanism, timing scheme, network diagram and many other details. INSTEON claims to use 915 MHz as the center frequency and frequency shift keying (FSK) with a 64 KHz frequency shift as the radio frequency (RF) modulation method, and encode the signal using the Manchester scheme.

INSTEON defines its own packet structure, which is shown in Figure 2. An INSTEON packet contains five mandatory fields and one optional field. The structure starts with the source and destination device addresses. INSTEON device addresses, which are three bytes long, are set during manufacturing and remain static. The packet structure continues with one byte for flags, two bytes for commands, an optional user-data field and ends with a one-byte cyclic redundancy check (CRC).

INSTEON flags describe the type of message contained in a packet (e.g., broadcast or non-acknowledgment, group, acknowledgment, extended or standard length), the number of hops remaining and the maximum number of hops allowed for the packet. Standard-length packets are ten bytes long while extended-length packets include 14 bytes of user-defined data, making these packets 24 bytes long. INSTEON messages are not routed like typical Internet Protocol messages. Instead, packets traverse the network by hopping across devices. A packet may hop a maximum of three times as it traverses the network to its destination device.

INSTEON devices are statically networked during the setup (pairing) process. A user performs a series of button-pushes and uses the INSTEON mobile application to program the devices. One device is the controller (master) while the other device is the responder (slave). INSTEON devices can be networked to communicate in a mesh network to limit service disruptions due to malfunctioning devices. The INSTEON hub connects to the user's home router for Internet access. The specifications describe the ability to include encrypted

1 Byte	3 Bytes	3 Bytes	2 Bytes	14 Bytes	1 Byte
Flags	Destination Address	Source Address	Commands	User Data (Optional)	CRC

Figure 3. Packet structures specified in [4] and [17].

data using the user-data field of extended length messages. This capability is an add-on that may be implemented by device developers.

INSTEON: Developer's Guide. *INSTEON: Developer's Guide* [4] provides additional information about the INSTEON protocol. While most of the specifications in the document match those provided in the previous document [18], a few details about the radio frequency specifications differ. Specifically, the frequency shift keying deviation and symbol rate are listed as 200 KHz and 9.124 KBaud, respectively. Additionally, as shown in Figure 3, the INSTEON packet structure varies from the structure described in the previous document.

3.2 Previous Research

Shipley and Gooler [17] have reverse engineered the INSTEON protocol and developed a basic transmitter and receiver using the YARD Stick One software-defined radio. They also claim that INSTEON's documentation is incorrect. Specifically, they maintain that INSTEON:

- Does not use the 915 MHz center frequency, but instead 914.975 MHz (914.95 MHz is coded in the software-defined radio receiver).
- Does not use true Manchester encoding.
- Does not use "traditional" frequency shift keying, but instead inverted frequency shift keying.
- Does not use the frequency shift keying value specified in the documentation.
- Does not use the symbol rate specified in the documentation.
- Does not use the cyclic redundancy check algorithm as specified in the documentation.
- Does not use the packet structure specified in the documentation. Figure 3 presents Shipley and Gooler's claimed INSTEON packet structure.

While Shipley and Gooler's transmitter and receiver tool work, they are limited and could be improved; this was the impetus for the research described in this chapter. First, the receiver program outputs packets directly to the terminal. Second, the receiver seemingly drops packets; this chapter describes the pilot studies and experiments used to test and improve the packet capture success rate. Third, Shipley and Gooler have not presented a method for enumerating or investigating INSTEON devices; this limitation provided the motivation to find commands that could be used to query and identify INSTEON devices beyond the device address.

3.3 Integrating Wireshark

Wireshark is the *de facto* network traffic analysis tool. Shipley and Gooler's research did not integrate with Wireshark. Instead, INSTEON network traffic was outputted to a terminal window. This made it difficult to view and analyze the packets or save traffic data.

The integration with Wireshark is a three-step problem: (i) outputting `pcap` data; (ii) developing a dissector to interpret the protocol; and (iii) configuring Wireshark to use the dissector. Over the course of this research, Shipley and Gooler's receiver was enhanced by integrating Wireshark [10].

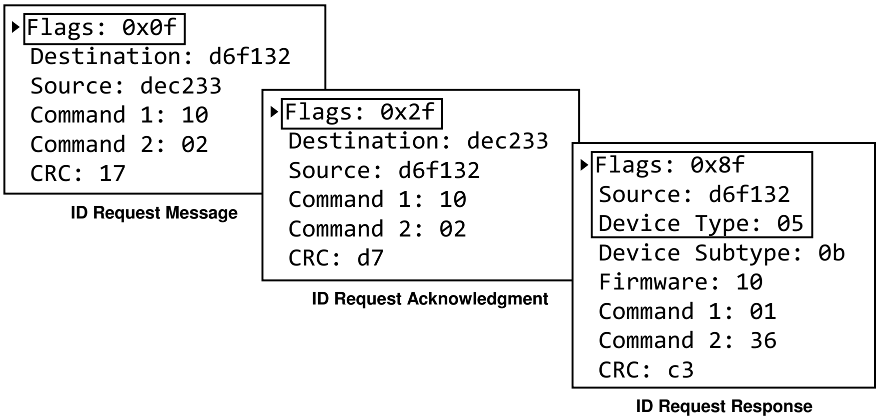
3.4 Pilot Studies

The first pilot study described in this section captured packets between the INSTEON hub and an LED light. The results established the baseline for the packet capture success test. The second pilot study discovered and verified a command for querying a device for its characteristics.

Pilot Study 1: Determining Packet Count. This pilot study employed a similar experimental setup but a significantly more robust reception method than the one used by Shipley and Gooler. The study determined that, for every "on" command, the hub and LED exchanged eight packets. The same was true for every "off" command. Turning the light on and off 20 times produced a total of 320 packets. This established a baseline number for the packet capture success rate experiment. Additional details about this experiment are provided in Section 4.1.

Pilot Study 2: Enumerating Devices. Command tables from 2007 (see [3]) describe a command that can be used to request the identity of a device. Three messages are transmitted as illustrated in Figure 4. First, the controller device issues the command to request the identity of the responding device. The responding device transmits an acknowledgment of the command and then responds by transmitting a broadcast message. In the INSTEON protocol, broadcast messages are typically used in the device pairing process and have a unique structure. INSTEON broadcast messages contain the device category, subcategory and firmware version of the source device. Figure 5 shows the packet structure of a broadcast message.

The documentation [5] identifies device category `0x05` as corresponding to an "access control" device. Tests confirmed that the devices and their respective categories match the documentation. Thus, the device category can be used as


ID Request Message
ID Request Acknowledgment
ID Request Response

```
Flags: 0x0f
Destination: d6f132
Source: dec233
Command 1: 10
Command 2: 02
CRC: 17
```

```
Flags: 0x2f
Destination: dec233
Source: d6f132
Command 1: 10
Command 2: 02
CRC: d7
```

```
Flags: 0x8f
Source: d6f132
Device Type: 05
Device Subtype: 0b
Firmware: 10
Command 1: 01
Command 2: 36
CRC: c3
```

Figure 4. Device identification request, acknowledgment and response messages.

1 Byte	3 Bytes	1 Byte	1 Byte	1 Byte	2 Bytes	1 Byte
Flags	Source Address	Device Category	Device Subcategory	Firmware Version	Commands	CRC

Figure 5. Packet structure for broadcast messages.

the basis for device enumeration. Note that the device category also informs an attacker about the device that is being controlled (e.g., lighting control, climate control or access control).

3.5 INSTEON Protocol Summary

The INSTEON documentation and previous research are inconsistent about the wireless specifications and packet structure of the INSTEON protocol. As mentioned above, one document states that the center frequency is 915 MHz while Shipley and Gooler claim that the true center frequency is 914.975 MHz. Further examination of Shipley and Gooler's code reveal a third potential center frequency value of 914.95 MHz. Neither the INSTEON documentation nor previous research agree on the frequency shift keying value. Additionally, the documentation and previous research disagree on the frequency shift keying symbol rate – one document lists 76.8 KBaud, another lists 9.124 KBaud and Shipley and Gooler list 9.125 KBaud. Table 1 summarizes the discrepancies. These conflicts lead researchers to ask: Which values are correct?

4. Experiments

Two experiments were conducted. The first experiment investigated the packet capture success rate of Shipley and Gooler's receiver and attempted to improve the packet capture rate. The second experiment evaluated the authenticity and targetability of the honeypot developed during this research.

Table 1. INSTEON radio frequency specifications from all sources.

Specification	**The Details**	**Developer's Guide**	**Shipley and Gooler**
Center Frequency	915.000 MHz	915.000 MHz	914.975 MHz 914.950 MHz (in code)
Encoding Method	Manchester	Manchester	"Tokenized" Manchester
Modulation Method	FSK	FSK	Inverted FSK
FSK Shift	64 KHz	200 KHz	150 KHz
FSK Symbol Rate	76.800 KBaud	9.124 KBaud	9.125 KBaud

4.1 Packet Capture Experiment

The conflicting documentation summarized in Section 3.5 provided the control variables for this experiment. The experiment had two objectives. The first objective was to validate (or refute) Shipley and Gooler's claims regarding the INSTEON radio frequency protocol specifications, specifically the center frequency, frequency shift keying value and symbol rate. The second objective was to determine the best settings to obtain the maximum packet capture success rate with the YARD Stick One software-defined radio.

Two INSTEON devices were used in the experiment: (i) INSTEON hub with Internet connectivity; and (ii) INSTEON LED bulb. The INSTEON hub application on a Windows Surface 3 computer was connected via the Internet to the INSTEON hub to control the LED light. The YARD Stick One was connected to a separate laptop running Ubuntu Linux and the software-defined radio configurations. Figure 6 presents the experimental network environment.

The experiment used the pilot study results described in Section 3.4 as the baseline for the packet capture success rate. The light was turned on and off 20 times to generate 320 packet transmissions while the receiver listened for packets. Each received or dropped packet was viewed as a separate, binary trial (successful or failed reception).

The experiment produced data with a binary distribution. The 320 individual results were pooled and viewed as a single trial. The number of received packets was divided by the number of total packets transmitted to compute the mean packet capture success rate.

The experiment involved varying the frequency between the three values in Table 1 (914.95 MHz, 914.975 MHz and 915 MHz). The frequency deviation and symbol rate were held constant at the start according to Shipley and Gooler's specifications. The best frequency was used in the test for the next control variable, and so on. If there was no distinguishable difference between a given variable, then Shipley and Gooler's settings were used for the next trial. Note that the symbol rate of 76.8 KBaud was assumed to be incorrect in

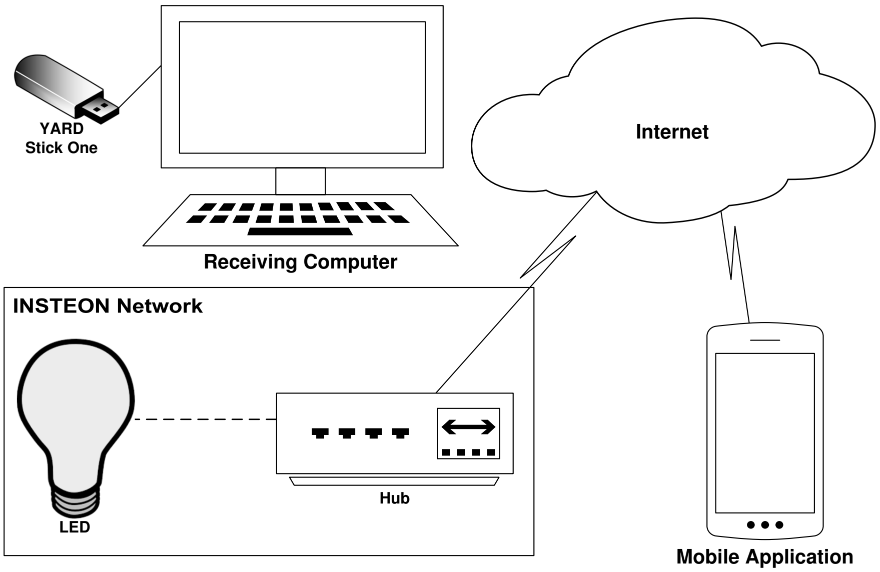


Figure 6. Experimental network environment for packet capture.

Table 2. Packet capture trial runs.

Trial	Frequency	Shift	Symbol Rate
1	914.950 MHz	150 KHz	9.125 KBaud
2	914.975 MHz	150 KHz	9.125 KBaud
3	915.000 MHz	150 KHz	9.125 KBaud
4	915.000 MHz	64 KHz	9.125 KBaud
5	915.000 MHz	200 KHz	9.125 KBaud
6	915.000 MHz	150 KHz	9.000 KBaud
7	915.000 MHz	150 KHz	9.250 KBaud

the experiment because it was very different from the values in the INSTEON documentation [4, 17]. The decision was made to vary the symbol rate between 9 KBaud, 9.125 KBaud and 9.250 KBaud. This resulted in seven radio configurations for the experiment. Table 2 shows the configuration for each experimental trial.

4.2 Functional Testing Experiment

The evaluation of the honeypot involved a functional test for authenticity and targetability. This experiment had two goals. The first goal was to ensure that a single honeypot network enumeration and map matched the genuine INSTEON device enumeration and map, enabling an attacker to believe that

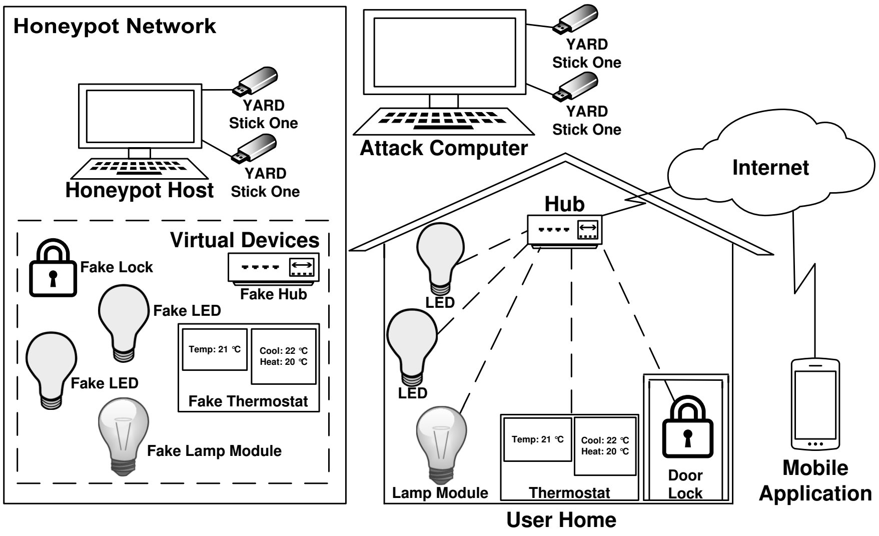


Figure 7. Experimental network environment for functional testing.

the honeypot devices were authentic. The second goal was to ensure that the honeypot network traffic mimics genuine network communications, making the honeypot targetable by an attacker. The simulated network traffic and devices were based on the results of the pilot study in Section 3.4 and other observations made over the course of the research.

The test environment consisted of an INSTEON hub, thermostat, two LED bulbs, a regular light bulb with a lamp dimmer module and a lock controller with door lock. Figure 7 illustrates the experimental environment. The honeypot was hosted on a high-performance computer platform (Dell Precision M4500 laptop running Ubuntu 14.04 LTS). A single honeypot host controlled the distinct honeypot networks.

The honeypot target had to be authentic enough so that an attacker could interact with the virtual devices. A successful implementation would involve the virtual devices responding to commands in a manner identical to genuine devices.

The authenticity portion of the experiment involved the attacker enumerating the devices using the Identity Request command described in Section 3.4. The attacker created a full network map by spoofing this command between all known devices. Figure 8 shows the results of the true INSTEON network scan. An interesting point is that the thermostat (ID: `D6 F1 32`) responded to every other INSTEON device. This is not typical of INSTEON devices and is due to a flaw in the thermostat logic. Based on this information, a decision was made to allow the honeypot thermostat to have the same "flaw."

A targetable honeypot should have multiple decoys to draw the attention of attackers. The decoys should appear to be positioned in the appropriate

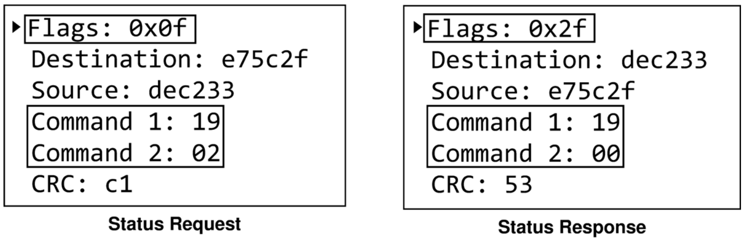
```
------------------------------
D6 F1 32 is a: Climate Control
DE C2 33 is a: Network Bridge
33 D3 32 is a: Dimmable Lighting Control
56 E2 3E is a: Access Control
95 A3 2E is a: Dimmable Lighting Control
E7 5C 2F is a: Dimmable Lighting Control

Controllers are:
DE C2 33 controls: ['D6 F1 32', '33 D3 32', '56 E2 3E', '95 A3 2E', 'E7 5C 2F']
33 D3 32 controls: ['D6 F1 32']
56 E2 3E controls: ['DE C2 33', 'D6 F1 32']
95 A3 2E controls: ['DE C2 33, 'D6 F1 32']
E7 5C 2F controls: ['D6 F1 32']

Responders are:
D6 F1 32 responds to: ['DE C2 33', '33 D3 32', '95 A3 2E', '56 E2 3E', 'E7 C5 2F']
DE C2 33 responds to: ['95 A3 2E', '56 E2 3E']
33 D3 32 responds to: ['DE C2 33']
56 E2 3E responds to: ['DE C2 33']
95 A3 2E responds to: ['DE C2 33']
E7 5C 2F responds to: ['DE C2 33']
------------------------------
```

Figure 8. INSTEON network baseline enumeration.

environments. A single Internet of Things light bulb would be of little interest to an attacker when it is not connected to a larger home automation network. However, the presence of several connected devices could convince an attacker that the decoys are part of a legitimate home automation system. Indeed, an INSTEON hub, thermostat, multiple lights and lock controller would present a network that is reasonably similar to an INSTEON home network, thus posing as an attractive target to an attacker.



```
Flags: 0x0f
Destination: e75c2f
Source: dec233
Command 1: 19
Command 2: 02
CRC: c1
```

Status Request

```
Flags: 0x2f
Destination: dec233
Source: e75c2f
Command 1: 19
Command 2: 00
CRC: 53
```

Status Response

Figure 9. Wireshark output of device status request and response.

The targetability portion of the experiment involved checking for the presence of Android devices connected to the INSTEON application. When a user logs into the mobile application, the hub requests the status of all non-battery-powered devices in the INSTEON network. Figure 9 shows the Wireshark output of the traffic generated when the INSTEON hub requests the status of a device. The honeypot was intended to mimic similar network traffic, thus rendering the honeypot devices targetable by an attacker. The traffic was gen-

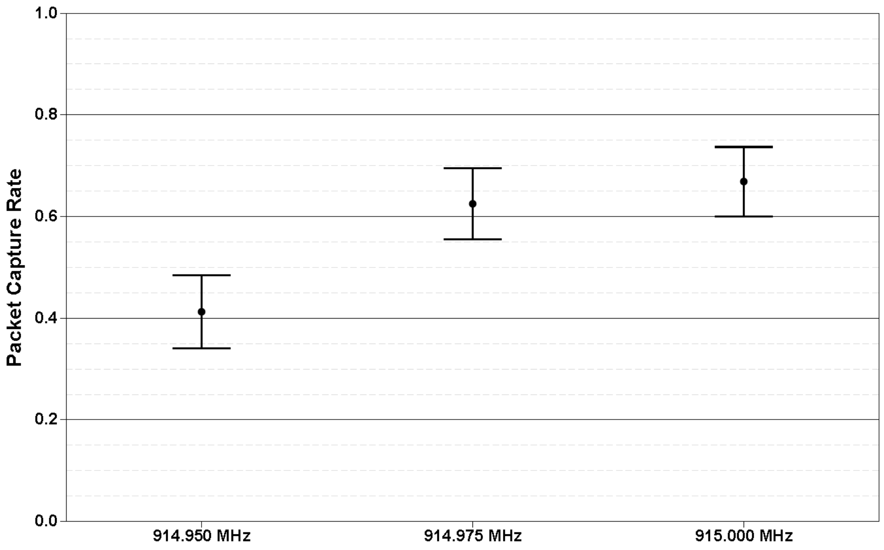


Figure 10. Packet capture results for various center frequencies.

erated using a simple Python program that made use of Shipley and Gooler's INSTEON transmission program.

The honeypot functional testing experiment involved the creation of a honeypot with one virtual network and five distinct virtual networks. If all the honeypot networks mimicked the genuine network, then the honeypot networks would be targetable and would help hide the genuine network from a would-be attacker.

5. Experimental Results

This section analyzes the results of the packet capture and honeypot functional testing experiments.

5.1 Packet Capture Experiment

A 99% confidence interval plot was generated for each pooled trial with comparison plots to present the results. Additionally, a linear model for the binary logistical distribution was produced to verify the results.

Figure 10 shows the impact of varying frequency while keeping the frequency shift keying value and symbol rate constant at 150 KHz and 9.125 KBaud, respectively. The 915 MHz frequency produced a packet capture rate of approximately 65%. The 914.95 MHz frequency, which is in the open-source code published by Shipley and Gooler, produced an inferior packet capture rate (approximately 40%) compared with the other frequencies.

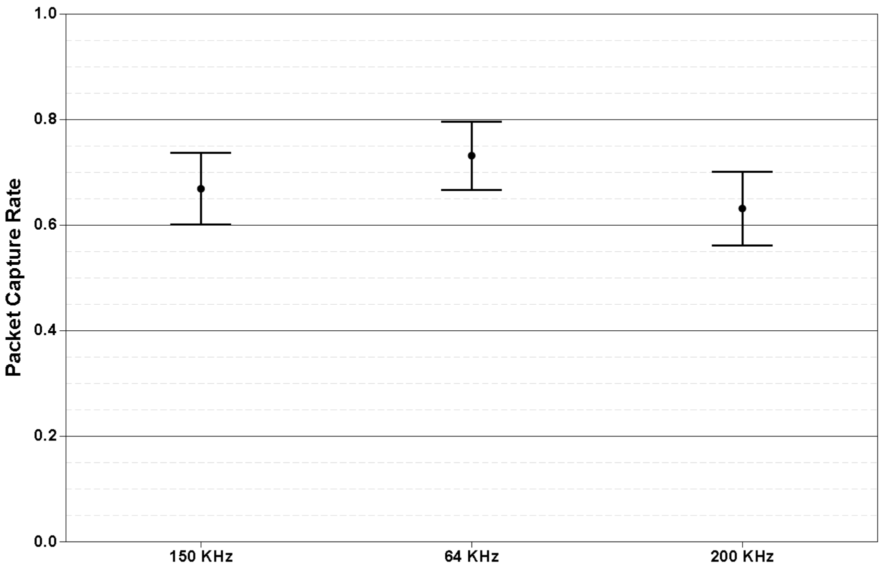


Figure 11. Packet capture results for various frequency shift keying values.

Figure 11 shows the confidence intervals when the frequency was held constant at 915 MHz and symbol rate was 9.125 KBaud, while the frequency shift keying value was varied between 200 KHz, 150 KHz and 64 KHz. No statistical difference in the packet capture rate was observed when varying the frequency shift keying value, although tuning the value to 64 KHz captured approximately 75% of the packets.

The final test varied the symbol rates between 9 KBaud, 9.125 KBaud and 9.25 KBaud while maintaining the center frequency at 915 MHz and the frequency shift keying value at 150 KHz. The confidence intervals revealed that the packet capture rates were not statistically different.

A graph containing the confidence intervals for each experimental trial was constructed and a linear model for the regression was created. Figure 12 shows the confidence interval plots for all the experimental trials. The best reception rate was obtained at approximately 75% when the center frequency was 915 MHz, frequency shift keying value was 64 KHz and symbol rate was 9.125 KBaud. The center frequency of 914.95 MHz encoded in Shipley and Gooler's software clearly yielded an inferior packet capture rate. Modifying the center frequency variable was determined to be statistically significant in improving the packet capture success rate.

Figure 13 shows the linear model. This analysis verifies that the frequency has a statistically significant impact on the packet capture rate and that no other variables are statistically significant.

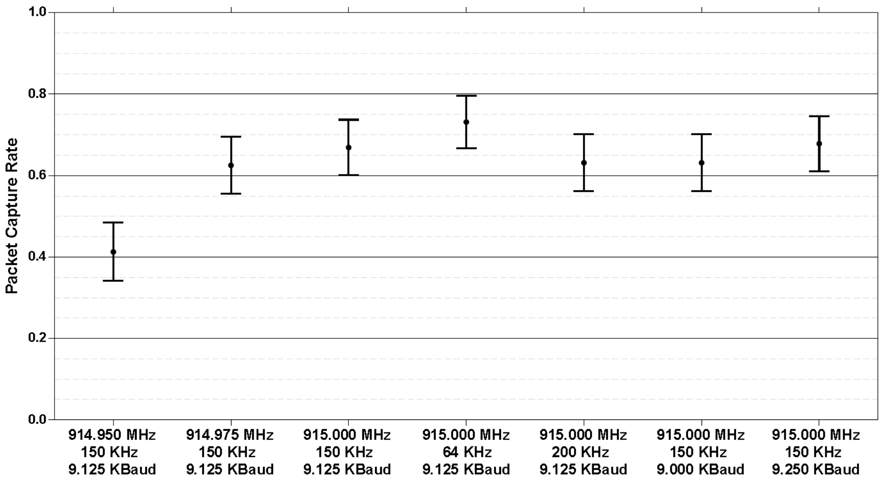


Figure 12. Results of all the experimental trials.

```
Coefficients:
                          Estimate   Std. Error    z value   Pr(>|z|)
(Intercept)               -0.43267      0.31192     -1.387      0.165
Frequency914.975 Mhz       0.86447      0.16195      5.338   9.41e-08  ***
Frequency915 Mhz           1.05618      0.16432      6.427   1.30e-10  ***
Shift150 Khz              -0.08592      0.16927     -1.508      0.612
Shift200 Khz              -0.25087      0.16724     -1.500      0.134
Shift64 Khz                0.21252      0.17449      1.218      0.223
Symbol.Rate9125KBaud       0.16494      0.16593      0.994      0.320
Symbol.Rate9250KBaud       0.20759      0.16656      1.246      0.213
```

Figure 13. Linear model results.

5.2 Functional Testing Experiment

The honeypot functional testing experiment involved two parts. First, the authenticity of the honeypot was determined by mapping and investigating the honeypot network individually and comparing it with the baseline (genuine) INSTEON network. Next, the targetability of the honeypot was measured by determining if the honeypot devices presented themselves in a manner similar to the genuine INSTEON devices.

Figure 14 presents the network enumeration used to determine the authenticity of the honeypot. Table 3 summarizes the genuine INSTEON network enumeration compared with the honeypot enumeration. The honeypot devices match the true INSTEON devices with regard to device category information and when compared against the complete network map in Figure 8. Therefore, the honeypot network accurately mimics the genuine INSTEON network, helping convince an attacker that the virtual devices are authentic.

The targetability of the honeypot was determined by presenting distinct honeypot networks together with the genuine network. Figure 15 compares the

```
------------------------------
C1 C1 C1 is a: Climate Control
B1 B1 B1 is a: Network Bridge
E1 E1 E1 is a: Dimmable Lighting Control
F1 F1 F1 is a: Access Control
A1 A1 A1 is a: Dimmable Lighting Control
D1 D1 D1 is a: Dimmable Lighting Control

Controllers are:
B1 B1 B1 controls: ['C1 C1 C1', 'E1 E1 E1', 'F1 F1 F1', 'A1 A1 A1', 'D1 D1 D1']
E1 E1 E1 controls: ['C1 C1 C1']
F1 F1 F1 controls: ['B1 B1 B1', 'C1 C1 C1']
A1 A1 A1 controls: ['B1 B1 B1, 'C1 C1 C1']
D1 D1 D1 controls: ['C1 C1 C1']

Responders are:
C1 C1 C1 responds to: ['B1 B1 B1', 'E1 E1 E1', 'A1 A1 A1', 'F1 F1 F1', 'D1 D1 D1']
B1 B1 B1 responds to: ['A1 A1 A1', 'F1 F1 F1']
E1 E1 E1 responds to: ['B1 B1 B1']
F1 F1 F1 responds to: ['B1 B1 B1']
A1 A1 A1 responds to: ['B1 B1 B1']
D1 D1 D1 responds to: ['B1 B1 B1']
------------------------------
```

Figure 14. Honeypot baseline scan.

Table 3. Genuine INSTEON network compared with one honeypot network.

INSTEON ID	Honeypot ID	Device Category	Matched
DE C2 33	B1 B1 B1	Network bridge	✓
D6 F1 32	C1 C1 C1	Climate control	✓
E7 5C 2F	D1 D1 D1	Dimmable light control	✓
33 D3 32	E1 E1 E1	Dimmable light control	✓
95 A3 2E	A1 A1 A1	Dimmable light control	✓
56 E2 3E	F1 F1 F1	Access control	✓

two networks – one genuine INSTEON network and one honeypot network – running simultaneously. The honeypot generates network traffic that mimics a user checking the status of the lights, thermostat and other simulated devices. Therefore, the simulated traffic presents a targetable, distinct honeypot network to an attacker. The networks appear to be identical, thus deceiving the attacker that multiple distinct INSTEON networks are present.

Figure 16 shows six networks – one genuine INSTEON network and five honeypot networks – running simultaneously. In each case, the simulated network traffic presented by the honeypot devices is functionally identical to the traffic presented by the genuine INSTEON network devices. This helps "hide" the genuine network among the honeypot networks. Instead of a single genuine network, an attacker is presented with six distinct, targetable networks. This

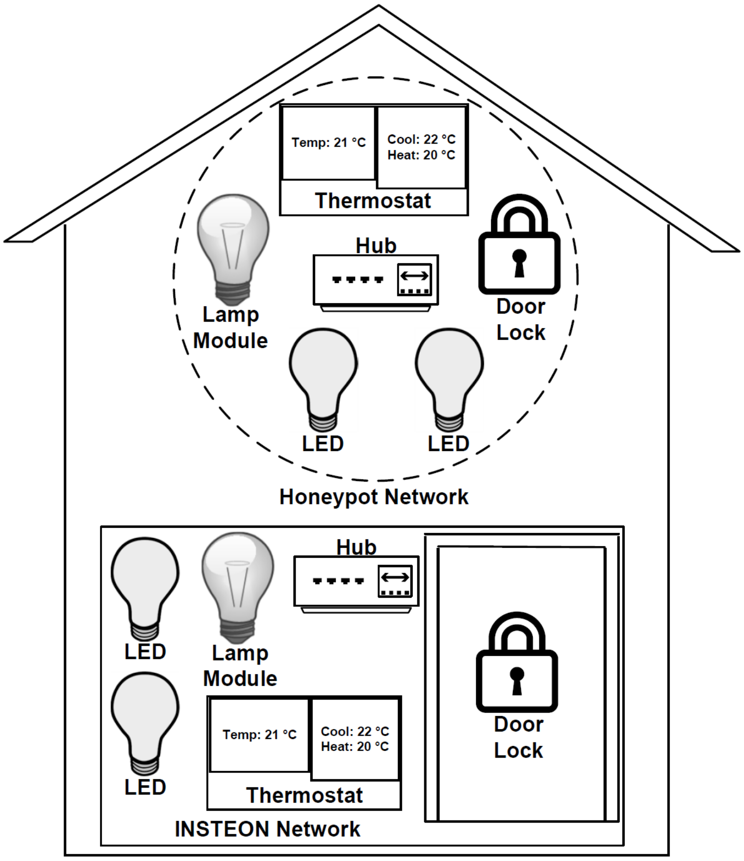


Figure 15. Genuine INSTEON network and a honeypot network.

protects the genuine network by reducing its attack probability to 1/6, unless the attacker performs deeper analysis before targeting the networks.

6. Limitations and Future Work

The honeypot developed in this research is limited in the types of devices it can replicate. Currently, the honeypot is programmed to mimic dimmable lights, access controllers, hubs and thermostats. A typical user would have these devices, but may have other types of devices as well. Other device types include switched lighting controls (e.g., in-wall light switches and dimmers) and security and safety sensors (e.g., door open/close, motion and leak sensors). These capabilities need to be investigated and incorporated in the honeypot.

Additionally, the performance of the honeypot was not measured in the experiments. An experiment that measures packet response times could be

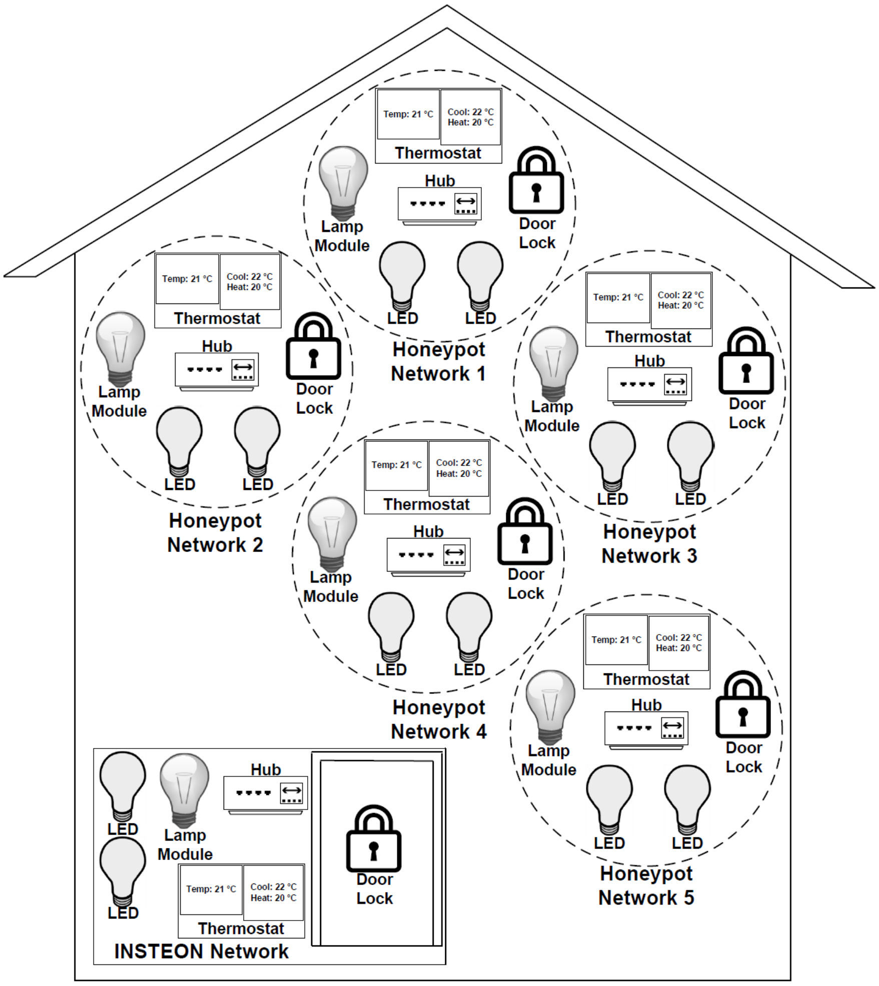


Figure 16. Genuine INSTEON network and five honeypot networks.

performed to identify the number of honeypot devices that could be instantiated before the honeypot experiences noticeable delays.

In its current state, the honeypot developed in this research is purely a decoy. Additional reporting and logging capabilities must be implemented before the honeypot can serve as a robust defensive tool. A relatively simple reporting capability could be implemented by installing an email server in the honeypot host that sends email messages to the user when events of interest occur.

7. Conclusions

Security and safety are becoming priorities as home and building automation technologies and Internet of Things devices proliferate. Device developers and manufacturers need to incorporate sound security engineering principles throughout the design and implementation phases of home and building automation systems. INSTEON is a leading Internet of Things protocol for home and building automation. The proposed technique for analyzing INSTEON traffic using a YARD Stick One software-defined radio improves the packet capture rate from approximately 40% to almost 75% compared with previous efforts. Additionally, the virtual INSTEON decoy networks developed in this research have excellent authenticity and targetability characteristics, which renders them attractive candidates for helping secure home and building automation systems as well as Internet of Things devices in general.

Note that the views expressed in this chapter are those of the authors and do not reflect the official policy or position of the U.S. Air Force, U.S. Army, U.S. Department of Defense or U.S. Government.

Acknowledgement

This research was partially supported by the U.S. Department of Homeland Security Industrial Control Systems Cyber Emergency Response Team (ICS-CERT).

References

[1] A. Baviskar, J. Baviskar, S. Wagh, A. Mulla and P. Dave, Comparative study of communication technologies for power-optimized automation systems: A review and implementation, *Proceedings of the Fifth International Conference on Communication Systems and Network Technologies*, pp. 375–380, 2015.

[2] M. Bowden, *Worm: The First Digital World War*, Atlantic Monthly Press, New York, 2011.

[3] P. Darbee, INSTEON Command Tables, Revision 20070927a, SmartLabs Technology, Irvine, California (`cache.insteon.com/pdf/INSTEON_Command_Tables_20070925a.pdf`), 2007.

[4] P. Darbee, INSTEON Developer's Guide (2nd Edition), SmartLabs Technology, Irvine, California (`cache.insteon.com/pdf/INSTEON_Developers_Guide_20070816a.pdf`), 2007.

[5] P. Darbee, INSTEON Device Categories and Product Keys, Revision 20081008, SmartLabs Technology, Irvine, California (`cache.insteon.com/pdf/INSTEON_DevCats_and_Product_Keys_20081008.pdf`), 2008.

[6] Gartner, Gartner says 6.4 billion connected "things" will be in use in 2016, up 30 percent from 2015, Stamford, Connecticut (`gartner.com/newsroom/id/3165317`), November 10, 2015.

[7] K. Girtz, B. Mullins, M. Rice and J. Lopez, Practical application layer emulation in industrial control system honeypots, in *Critical Infrastructure Protection X*, M. Rice and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 83–98, 2016.

[8] C. Gomez and J. Paradells, Wireless home automation networks: A survey of architectures and technologies, *IEEE Communications*, vol. 48(6), pp. 92–101, 2010.

[9] J. Hall, B. Ramsey, M. Rice and T. Lacey, Z-wave network reconnaissance and transceiver fingerprinting using software-defined radios, *Proceedings of the Eleventh International Conference on Cyber Warfare and Security*, pp. 163–171, 2016.

[10] C. Mays and B. Ramsey, INSTEON, inste-off, inste-open? presented at *DEF CON 24*, 2016.

[11] N. Provos and T. Holz, *Virtual Honeypots: From Botnet Tracking to Intrusion Detection*, Addison-Wesley, Boston, Massachusetts, 2007.

[12] B. Ramsey, Improved Wireless Security through Physical Layer Protocol Manipulation and Radio Frequency Fingerprinting, Ph.D. Dissertation, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2014.

[13] Revention, Revention integrates with Insteon to provide automated light fixtures, Houston, Texas (`revention.com/news/articles/Revention-Integrates-with-Insteon-to-Provide-Automated-Light-Fixtures`), November 29, 2016.

[14] L. Rist, J. Vestergaard, D. Haslinger, A. Pasquale and J. Smith, Conpot ICS/SCADA Honeypot, Honeynet Project (`conpot.org`), 2013.

[15] A. Rose and B. Ramsey, Picking Bluetooth Low Energy locks from a quarter mile away, presented at *DEF CON 24*, 2016.

[16] B. Schneier, Real-world security and the Internet of Things, *Schneier on Security* (`schneier.com/blog/archives/2016/07/real-world_secu.html`), July 28, 2016.

[17] P. Shipley and R. Gooler, Insteon: False security and deceptive documentation, presented at *DEF CON 23*, 2015.

[18] SmartLabs Technology, INSTEON Whitepaper: The Details, Version 2.0, Irvine, California (`cache.insteon.com/documentation/insteon_details.pdf`), 2013.

[19] Transparency Market Research, Global home automation market will be worth US$21.6 billion by 2020, Albany, New York (`www.transparencymarketresearch.com/pressrelease/home-automation-market.htm`), September 3, 2015.

[20] M. Winn, M. Rice, S. Dunlap, J. Lopez and B. Mullins, Constructing cost-effective and targetable industrial control system honeypots for production networks, *International Journal of Critical Infrastructure Protection*, vol. 10, pp. 47–58, 2015.

[21] C. Withanage, R. Ashok, C. Yuen and K. Otto, A comparison of the popular home automation technologies, *Proceedings of the IEEE Conference on Innovative Smart Grid Technologies – Asia*, pp. 600–605, 2014.

Chapter 16

SECURING BLUETOOTH LOW ENERGY LOCKS FROM UNAUTHORIZED ACCESS AND SURVEILLANCE

Anthony Rose, Jason Bindewald, Benjamin Ramsey, Mason Rice and Barry Mullins

Abstract This chapter describes several vulnerabilities that affect commercial and residential Bluetooth Low Energy security devices and outlines methods for exploiting plaintext, obfuscated and hard-coded passwords, brute forcing passwords and hashes, fuzzing commands and performing man-in-the-middle attacks. Evaluations reveal that 75% of the tested security and access control systems have vulnerabilities that grant unauthorized access. In addition to obtaining access, malicious actors can extract sensitive information that can be used to develop patterns of human behavior. This chapter discusses five solutions for preventing or mitigating Bluetooth Low Energy security breaches, most of which involve minimal implementation overhead on the part of developers.

Keywords: Bluetooth Low Energy, access control, locks, vulnerabilities, security

1. Introduction

Bluetooth Low Energy, also marketed as Bluetooth Smart, is a wireless protocol designed for interconnecting Internet of Things (IoT) devices. The Internet of Things is an expanding market that includes linkages from home automation systems to industrial control systems and is expected to grow to more than $19 trillion devices by 2020 [20]. Bluetooth Low Energy devices, with nearly 8.2 billion already in use worldwide, currently constitute more than one-third of all Internet of Things devices [3]. Meanwhile, Internet of Things devices are becoming increasingly intertwined with water, power, emergency services, health care, agriculture, transportation and security systems [15].

Physical security relies on access control to manage admittance to sensitive locations. Typical access control implementations involve the use of personal identification numbers (PINs), radio frequency identification (RFID), public

M. Rice and S. Shenoi (Eds.): Critical Infrastructure Protection XI, IFIP AICT 512, pp. 319–338, 2017.
https://doi.org/10.1007/978-3-319-70395-4_16

key infrastructures and biometrics [5]. These solutions limit the ability of an organization to control credentials (e.g., granting or revoking access on demand). In some cases, access revocation may be impossible or expensive without revoking the credentials of all the users. A major appeal of Bluetooth Low Energy and other wireless systems is that access can be centrally managed. This requires authentication between the user and organization database, which helps eliminate the need to manage devices independently.

Several vendors have released security systems that use Bluetooth Low Energy locks to grant access to server rooms, power plants, water treatment facilities, manufacturing plants and ATMs. Onity [18] offers automation, manufacturing and security products; more than one million of its Bluetooth locking systems are being used in 115 countries.

The growth of Bluetooth within the Internet of Things paradigm has motivated the evaluation of commercial lock systems. Tests conducted as part of this research have revealed vulnerabilities in thirteen of the seventeen evaluated Bluetooth Low Energy locks. This chapter describes the vulnerabilities and proposes implementation guidance for securing the devices.

2. Bluetooth Low Energy

Bluetooth is an umbrella term that covers two completely different protocols: (i) Bluetooth Classic (BTC); and (ii) Bluetooth Low Energy (BLE). Bluetooth Classic focuses on sending the maximum amount of data without regard to power consumption (e.g., music streaming and data storage). Conversely, power saving is a top priority for Bluetooth Low Energy devices. Bluetooth Low Energy offers an interface for low-data-rate devices (e.g., temperature monitors and door locks). Bluetooth Low Energy is also designed to provide a secure and robust wireless communications mechanism that requires minimal energy at data rates up to 1 Mbps [12].

The Bluetooth Low Energy connection process differs from Bluetooth Classic by limiting the device transmission time, thereby minimizing the expended energy. Devices advertise themselves on three channels that are dispersed across the 2.4 GHz band to avoid interference from IEEE 802.11 wireless local area networks (WLANs) [24]. A user connects to a device on an advertising channel to initiate a connection. Bluetooth Low Energy operates under a master/slave model, where the master is typically the user (e.g., phone or tablet) and the slave is the device that awaits a connection (e.g., lock, thermostat or heart rate monitor). Bluetooth Low Energy devices are split into two categories depending on their function: (i) client; and (ii) server. The client is the master in most cases, while the slave is the server. Common Bluetooth Low Energy operations include read, write, notify and indicate, which push or pull data between the client and server through the Generic Attribute Profile (GATT).

The Generic Attribute Profile is constructed as a hierarchy in which the profile is at the top level and is composed of a series of services. Services are collections of characteristics that represent the behavior of a device. For example, a service could be listed as a blood pressure monitor or heart rate monitor

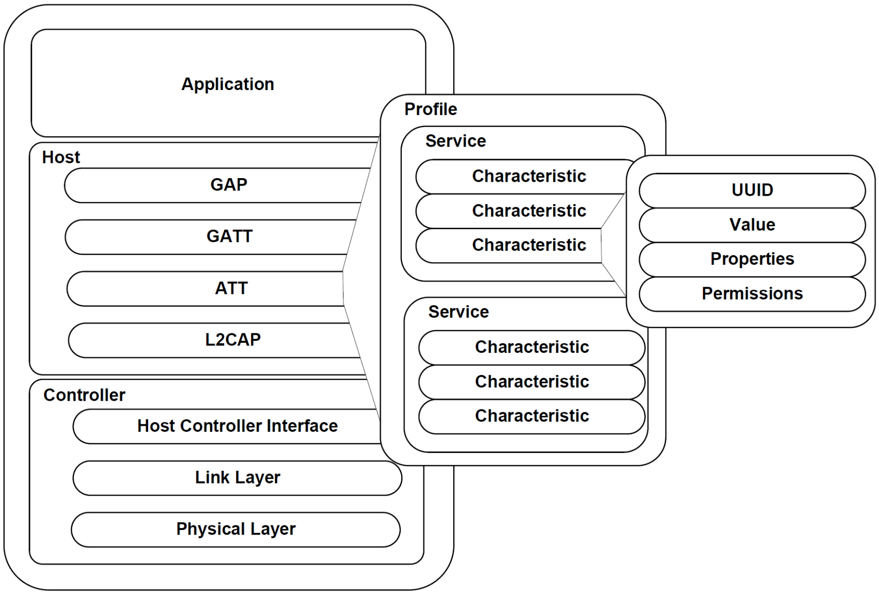


Figure 1. Bluetooth Low Energy stack hierarchy.

for medical devices or temperature readings for thermostats. Characteristics fall into a few different categories below a service. Each service has a universally unique identifier (UUID), value, properties (e.g., read, write, notify and indicate) and permissions. The UUID is a 16-bit or 128-bit identifier used by a manufacturer to specify custom services; however, some UUIDs are universally used across manufacturers. Finally, descriptors fall under characteristics and contain configuration flags and metadata that a manufacturer may desire to share. Figure 1 presents the Bluetooth Low Energy hierarchy.

3. User Behavioral Analytics

User behavior analytics (UBA) detects anomalies that indicate potential insider threats and targeted attacks by tracking and analyzing user behavior. Defensive mechanisms employ user behavior analytics to help prevent attacks. This research proposes an offensive approach that leverages user behavior analytics.

Reconnaissance is one of the most important stages in penetration testing because it helps acquire detailed knowledge of a target prior to an attack [16]. During this phase, a target is continuously monitored for all activity.

Time is critical when it comes to gaining information about a target and acting on it. Minimizing the time spent between target reconnaissance and infiltration greatly enhances the likelihood of an attack being successful [21].

Table 1. Exploited Bluetooth Low Energy devices by type.

Device Name	**Type**	**A**	**B**	**C**	**D**	**E**	**F**
Safetech Quicklock Doorlock	Deadbolt	✓		✓			✓
Vians Doorlock	Deadbolt		✓				✓
Lagute Sciener Doorlock	Deadbolt		✓				✓
Okidokeys	Deadbolt				✓		✓
Poly-Control Danalock	Deadbolt					✓	✓
Ceomate Doorlock	Deadbolt	✓	✓				✓
Safetech Quicklock Padlock	Padlock	✓		✓			✓
Elecycle EL797	Padlock		✓				✓
Elecycle EL797G	Padlock		✓				✓
Mesh Motion Bitlock	Padlock						✓
iBlulock	Padlock	✓		✓			✓
Safetech Gunbox 2.0	Gun Safe	✓		✓			✓
Plantraco Phantomlock	Cabinet Lock	✓		✓			✓

In the past, an attacker would physically monitor a target to gain information; however, attackers can now leverage information present in Bluetooth Low Energy devices in developing attacks.

A number of devices store system logs that contain valuable user behavior analytics information (e.g., user names and timestamps). Applying statistical analysis methods to system logs generates meaningful information that can be leveraged in attacks. For example, user behavior patterns can be inferred, which would provide the ideal times to inject malicious code and avoid detection.

4. Bluetooth Security Vulnerabilities

A wide variety of attacks against Bluetooth Low Energy devices have been developed; most of them exploit vulnerabilities inherent in the protocol or errors in vendor implementations. An analysis of seventeen Bluetooth Low Energy locks reveals that thirteen devices were vulnerable to eight exploits [23]. Table 1 lists the exploited Bluetooth Low Energy devices by type. Note that *A* denotes plaintext passwords, *B* password obfuscation, *C* brute forcing passwords and hashes, *D* command fuzzing, *E* hard-coded passwords and *F* man-in-the-middle attacks. This section discusses the vulnerabilities present in the tested Bluetooth Low Energy devices and how an adversary may exploit them.

The hardware required for Bluetooth Low Energy eavesdropping is affordable. Higher-end devices such as the HackRF One and Ubertooth One are more expensive alternatives that have high power amplifiers and detachable antennas. Replacing an antenna increases the operational range of a device. Increased sniffer range eliminates the need to be near a target to obtain its credentials. Pairing a long-range sniffer with a high power Bluetooth adapter

$$\underbrace{\textbf{0x01}}_{\textbf{Opcode}}\ \underbrace{\textbf{0x12345678}}_{\textbf{Current Password}}\ \underbrace{\textbf{0x66666666}}_{\textbf{New Password}}$$

Figure 2. Reverse engineered Safetech command structure.

(e.g., Sena UD-100) enables commands to be transmitted at distances up to half a mile.

The National Institute of Standards and Technology (NIST) [19] has reported that flawed Bluetooth security implementations are highly susceptible to wireless attacks (e.g., denial-of-service (DoS), eavesdropping, man-in-the-middle attacks, message modification and resource misappropriation). These attacks on Bluetooth systems can be leveraged by adversaries to obtain unauthorized access to sensitive information.

4.1 Plaintext Passwords

A plaintext password is stored or transmitted in a readable format and offers no protection to the user or device. This vulnerability is very common [1, 22] and it enables an adversary to eavesdrop on a conversation through specialized hardware or to embed software that monitors the host controller interface (HCI) traffic. Stolen plaintext passwords can be used to gain access to secure facilities, change administrative privileges or obtain system logs.

Figure 2 illustrates the ease with which a plaintext password can be used in an attack. The Safetech command structure has not been published and was discovered by reverse engineering. The command structure is used in several Safetech Bluetooth Low Energy devices (e.g., door locks, padlocks and safes). The first byte is an opcode that specifies if a device should read the password (`00`) or change the user password (`01`). A user password can be modified merely by changing the first byte and placing the current user password into the next four bytes. The final four bytes are then used to set the new user password. Note that the password for this type of device is limited to numbers.

4.2 Password Obfuscation

Obfuscated passwords provide more protection than plaintext passwords, but they still constitute a major security risk. An obfuscated password uses hashing or encryption to reduce the risk of exposure [4].

The problem with using an obfuscated password is that it can be recorded and replayed to a Bluetooth Low Energy lock. Replaying a password enables an adversary to gain access to the lock without knowing the password. Moreover, if the device uses the same hashing algorithm for the password every time, an adversary can gain access at any time using the sniffed obfuscated password. However, depending on the security implementation, some high-level functions may not be accessible. This could deter an attack that requires a password to gain access to high-level functions.

Table 2. Time expected to brute force passwords.

Available Characters	**Password Length**	**Password Possibilities (millions)**	**Expected Completion Time (years)**
10	8	10^2	0.03
10	16	10^{10}	3.17
128	8	7.2×10^{10}	22.83
128	16	5.1×10^{27}	1.61×10^{18}
256	8	1.8×10^{13}	5,475
256	16	3.4×10^{32}	1.06×10^{23}

Password obfuscation is implemented in the Ceomate door lock. Reverse engineering techniques applied to this product were unable to determine the proprietary hashing process. However, an attacker can still gain access to the device – without knowing the original password and hashing algorithm – merely by replaying the recorded hashed password.

4.3 Brute Forcing

Brute forcing involves submitting repeated guesses of a password or hash with the goal of gaining access [7]. This attack requires a guess of the plaintext password or obfuscated password. Obviously, if the number of password or hash possibilities are massive, a brute force attack is impractical. For example, a device that uses a six-digit PIN could be brute forced in hours, while an eight-character password would require nearly a month.

Table 2 shows the expected amount of time for brute forcing passwords based on the number of available characters and password length. Password length is more important than the number of available characters. Doubling the password length exponentially increases the number of password combinations, while doubling the available characters has a much smaller effect on the number of combinations. Timeouts between successive password attempts significantly reduce the speed of the overall attack.

It is important to use long PINs to protect devices. Assuming a speed of 6,000 attempts/min, a PIN that uses only numbers and has a maximum length of eight could be brute forced within twelve days. Factors that limit an attack include the rate at which the transmitting device sends packets and the speed at which the receiver translates the data. However, the speed of a brute force attack can be greatly increased if a web server is used to store the credentials.

4.4 Command Fuzzing

Command fuzzing occurs when an application accepts an invalid command that has been modified to mimic a valid command, potentially causing the

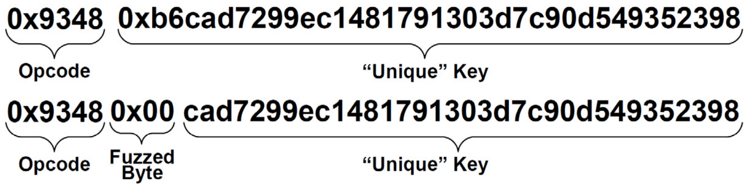

0x9348
0xb6cad7299ec1481791303d7c90d549352398
Opcode
"Unique" Key
0x9348
0x00
cad7299ec1481791303d7c90d549352398
Opcode
Fuzzed Byte
"Unique" Key

Figure 3. Reverse engineered Okidokeys command structure.

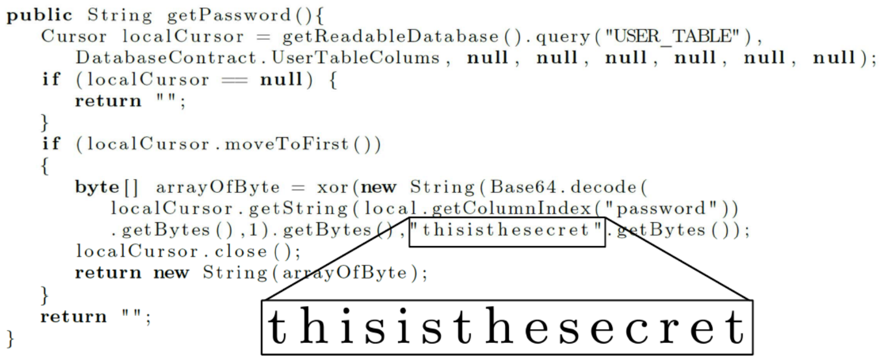
device to enter a new state [17]. This technique involves changing the individual bytes of a packet until the targeted application accepts the invalid command. The goal of fuzzing is to force a device to move into an unstable state in which it behaves in a manner different from what was designed. For example, a lock may go into an error state as a result of a fuzzed command and open without proper authentication. Opening in an error state is usually a design decision made for reasons of fire safety. A device may default to a locked state when entering an error state, but these designs are typically implemented in prison lock systems where locking by default is required. Fuzzing can be problematic when designers implement proprietary encryption. Well-established encryption methods (e.g., Advanced Encryption Standard (AES)) have been proven to offer secure communications channels [14].

Figure 3 shows the reverse engineered Okidokeys command structure (in bytes) at the host controller interface level and the fuzzed packet. Okidokeys describes the implementation as using a highly-secure patented cryptographic solution that offers the same protection as 256-bit AES. However, experiments have revealed that the generated keys are not unique. Specifically, the keys have patterns that are not encountered in AES and other encryption algorithms. This prompted the fuzzing of a previously-valid command that forced the lock to move to an error state in which it opened.

4.5 Hard-Coded Passwords

Hard-coded passwords, where designers leave passwords in applications, are the result of poor programming practices. Such passwords are encountered in more than 40% of Android applications [8]. However, hard-coded passwords are difficult to find because they require applications to be decompiled into readable code. Another method for capturing an administrative password is to implant malware with a keystroke logger on a target device. Hard-coded passwords offer an attacker the ability to gain access to developer options inside of an application and to bypass the built-in security controls.

The easiest method for finding the hard-coded passwords to a Bluetooth Low Energy device is to decompile an Android application package (APK). In general, an attacker would decompile an application after the Android application package has been removed from the device. Programs such as Bytecode Viewer offer a user-friendly environment to reverse engineer Android application pack-



```
public String getPassword(){
   Cursor localCursor = getReadableDatabase().query("USER_TABLE"),
      DatabaseContract.UserTableColums, null, null, null, null, null, null);
   if (localCursor == null) {
      return "";
   }
   if (localCursor.moveToFirst())
   {
      byte[] arrayOfByte = xor(new String(Base64.decode(
         localCursor.getString(local.getColumnIndex("password"))
         .getBytes(),1).getBytes(),"thisisthesecret".getBytes());
      localCursor.close();
      return new String(arrayOfByte);
   }
   return "";
}
```

Figure 4. Hard-coded password found in a Danalock.

ages into readable Java code. This readable code is parsed for keywords to reveal hard-coded passwords, developer comments and other valuable information.

Figure 4 shows a hard-coded password found by decompiling the Danalock application. The plaintext password is stored in a table with the passphrase `thisisthesecret`. Having discovered the password, the adversary can gain access to the lock. Decompilation may also reveal the method of encryption and other information that is hidden. This provides additional opportunities to compromise the system.

4.6 Man-in-the-Middle Attack

A man-in-the-middle attack occurs when two devices are unknowingly connected to a third device that relays information between the two communicating devices [2]. This attack is effective when devices use unauthenticated connections, enabling an attacker to intercept as well as modify and inject fake information or commands. Several tools have been developed for implementing man-in-the-middle attacks (e.g., GATTacker and BTLEjuice). Two attacks that leverage the man-in-the-middle concept are: (i) rogue device attack; and (ii) relay attack.

Rogue Device Attack. In a rogue device attack, an attacker impersonates a target device with the intention of convincing the other communicating device that the rogue device is, in fact, the target device. The majority of applications do not properly authenticate with devices before sending commands, enabling an attacker to clone the target device and send advertisements. The user application initiates a connection after it receives the cloned device advertisement. The user application then sends commands to the cloned device assuming it to be the target device. These commands include passwords and nonces that could be used by an attacker to gain access to the target device.

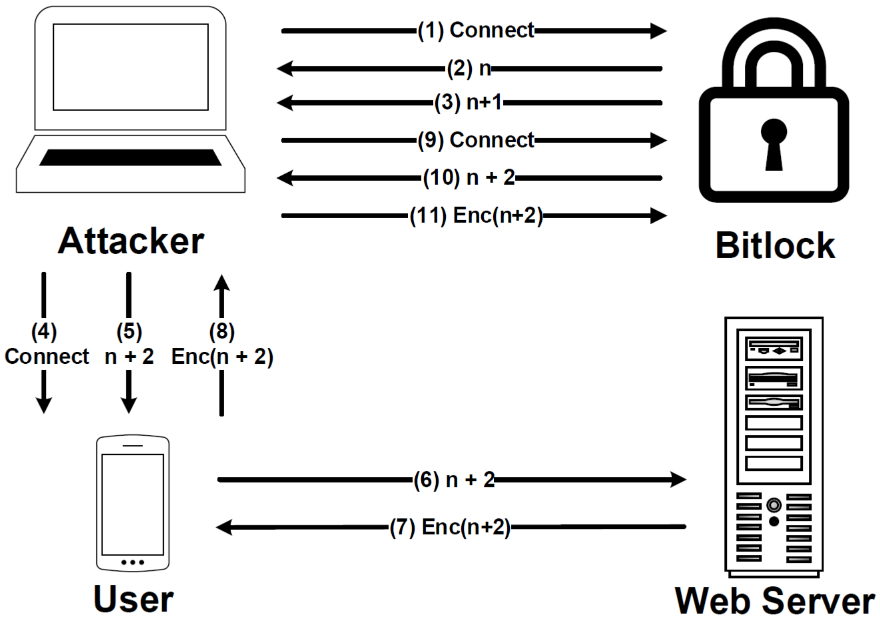


Figure 5. Sequence diagram of a rogue device attack on a Mesh Motion Bitlock.

A nonce is a random number that is used only once and protects a communications connection from a replay attack.

A rogue device attack can be used to exploit a web server that stores user credentials. A user application would be unable to distinguish between the true device and the cloned device, enabling an attacker to steal credentials from the web server. All that the attack requires is for the cloned device to interact with the user application.

Figure 5 shows a rogue device attack on a Mesh Motion Bitlock. This product does not use a plaintext password; however, it has a predictable nonce that enables an adversary to collect credentials and use them to control the lock. The user in this case must have an Internet connection in order to receive credentials from the web server. An attacker connects to the lock and sends invalid credentials with the intention of receiving the current nonce value. The value is sent with the initial connection and is incremented by one when receiving invalid credentials. The user is unaware that his/her device is connected to a spoofed lock when the next nonce is received. During the connection, the user forwards the nonce to the web server and receives the credentials in return. Finally, the credentials are sent from the user to the spoofed lock.

Because the web server trusts the user application, the attacker is not limited to receiving just one set of credentials. In fact, the attacker can flood the user application with nonces with the goal of creating a table to gain permanent

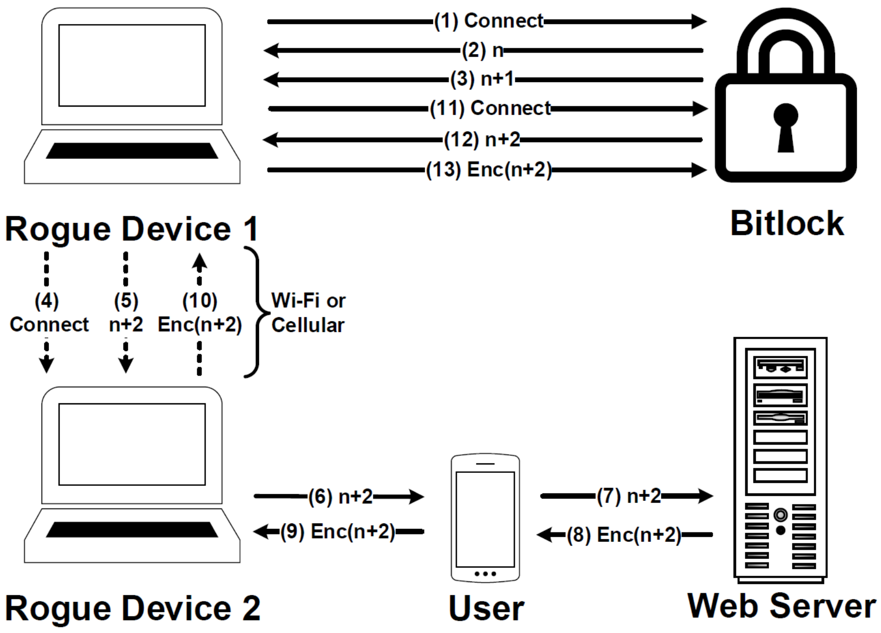


Figure 6. Sequence diagram of a relay attack on a Mesh Motion Bitlock.

access to the device. The attacker can use the table of credentials to open the lock at any time.

Relay Attack. A relay attack is similar to a rogue device attack, but it is designed specifically for scenarios where the nonces are truly random and a rogue device attack is not possible. In this attack, an attacker impersonates a target device and forces the user to communicate via a bridge to the attacker's device. This enables the attacker's device to impersonate the target device and trick the user into communicating with the attacker.

Figure 6 shows how two rogue devices can create a relay attack. Rogue Device 1 connects to the user while Rogue Device 2 connects to the target device. The target device generates a nonce and sends it to the cloned user (Rogue Device 1). Rogue Device 2 connects to the user and impersonates the target device.

After the two rogue devices are in place, a bridge is established using Wi-Fi, cellular or some other means. The bridge supports communications between the rogue devices and facilitates the hand-off of the nonce during the attack. Rogue Device 2 sends the nonce to the user, who then forwards the nonce to the web server to generate the credentials. This phase of the attack mirrors the rogue device attack discussed above. Finally, the credentials that the user unexpectedly generated are passed from Rogue Device 2 back to Rogue Device 1. These credentials are used by the attacker to gain access to the target device.

The danger of a relay attack is that a user can be anywhere as long as a rogue device is nearby to impersonate the target device. This type of attack can be used to target a variety of devices because large organizations require many access points and rely on a central server to handle user credentials.

5. Attack Scenario

Numerous devices store system logs of user activity with information such as user names, permissions and timestamps. An attacker can extract the system logs from locks and analyze the information to construct a profile of activity in a facility. The attacker can use the behavior patterns to gain insight into the organization's inner workings.

This section presents an attack scenario involving a manufacturing facility that uses several Bluetooth Low Energy locks for access control. The Bluetooth Low Energy system has a central server that manages credentials, requiring employees to authenticate via an application installed on their mobile devices. The simulated data in the scenario mimics real data found in employee devices. A proven method for extracting real data is presented, but simulated data is still required to meet the goals of the scenario.

The scenario involves the following steps:

- The attacker connects to a security door lock and scans for all services, characteristics and descriptors.
- The attacker uses the scanned information to construct an identical Bluetooth Low Energy device. The cloned device is used to impersonate the lock and convince the user application to transmit its credentials.
- Concurrently, the attacker uses a second device near the lock to impersonate the user. This setup mirrors the relay attack discussed above.
- The attacker relays information (e.g., nonces and credentials) from the user to the lock via the relay attack. The relay attack provides access to the Bluetooth Low Energy lock for exploitation.
- The attacker accesses developer and administrator privileges to create additional accounts and download system logs.
- The attacker applies user behavior analytics on the system logs to reveal behavior patterns (Figures 7–9). Analysis of the logs provides detailed information about facility operations by highlighting user activity based on the time and/or day of the week.
- The attacker determines the best time to infiltrate the facility based on the analysis.

The analyzed data can provide important information when cross referenced against employee public records. Specifically, user activity may be analyzed in three formats: (i) all user activity by day of week and time of day; (ii)

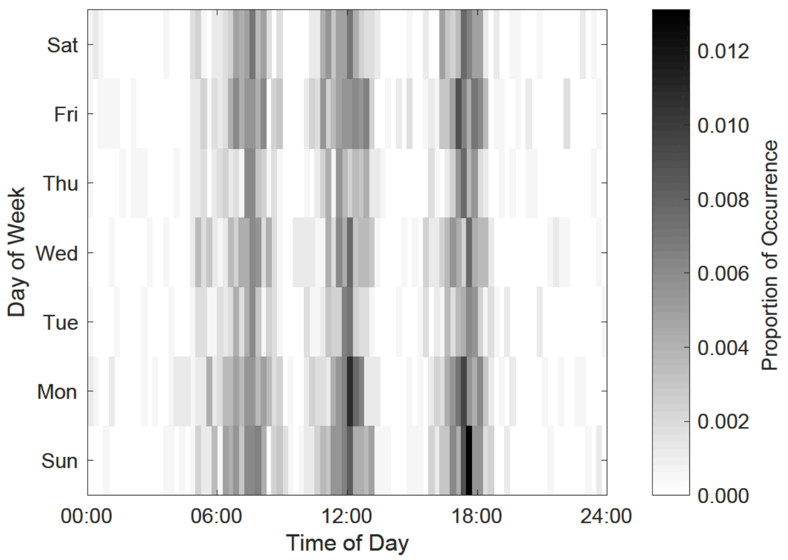


Figure 7. Heat map of historic weekday activity compared with time of day activity.

individual user activity by time of day; and (iii) individual user activity by day of the week.

The heat map in Figure 7 demonstrates that the overall user activity can be determined by comparing user activity to the time of day that users are active. User activities corresponding to entering and leaving the facility are indicated in the heat map, where the darker shades represent higher levels of user activity.

The heat map in Figure 8 highlights the user activity during historical days worked. Finally, the heat map in Figure 9 breaks down user activity on any given day by the time of day. Analyzing this information enables an attacker to determine the ideal day and time to access the facility. Additional information can also be inferred, such as odd activity at specific times of the day for specific users; this may indicate specific tasks (e.g., maintenance).

6. Mitigation Techniques

Many mitigation techniques have been proposed for combating Bluetooth attacks. Table 3 lists several mitigation techniques and the vulnerabilities against which they protect. Note that A denotes plaintext passwords, B password obfuscation, C brute forcing passwords and hashes, D command fuzzing, E hard-coded passwords and F man-in-the-middle attacks. The last two columns of Table 3 rank the implementation and maintenance difficulty of each proposed solution.

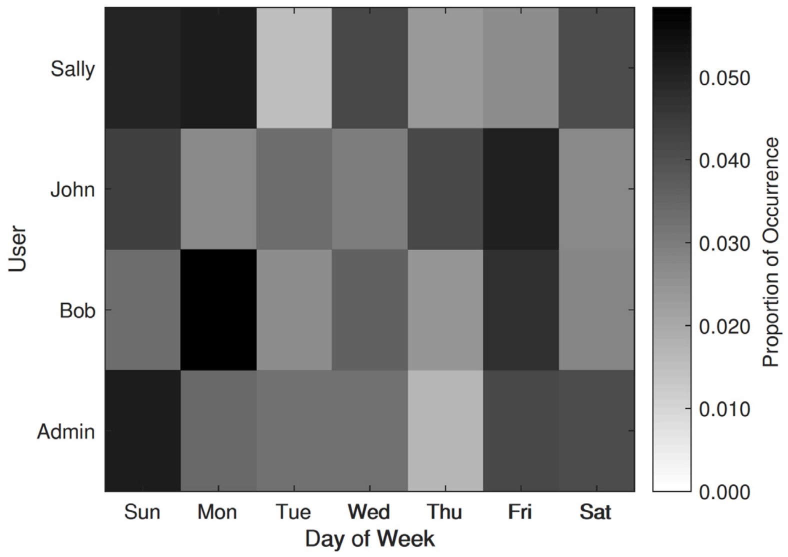


Figure 8. Heat map of historic user activity compared with weekday activity.

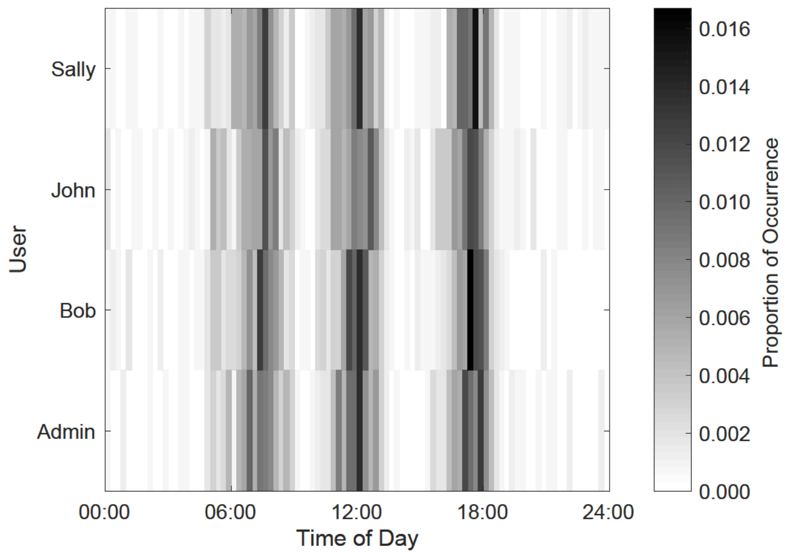


Figure 9. Heat map of historic user activity compared with time of day activity.

6.1 Pairing and Bonding

Pairing and bonding protect against malicious eavesdroppers. Two processes occur during the initial connection. The first step is pairing, which involves an

Table 3. Mitigation techniques and their difficulty.

Mitigation Techniques	**A**	**B**	**C**	**D**	**E**	**F**	**Difficulty of Use**	**Difficulty of Maintenance**
Pairing and Bonding	✓	✓	✓	✓			Low	Low
App Layer Encryption	✓	✓	✓	✓			Medium	Medium
Two-Way Authentication	✓	✓	✓	✓			Medium	Low
Geofencing						✓	Medium	High
BLE Guardian						✓	High	High

exchange of security features and capabilities. This step begins with the client and establishes the types of input and output mechanisms that exist in the device and dictates the type of bonding. Bonding occurs after pairing and the keys have been generated and exchanged. Bonding is a more permanent encryption method that saves the key for use in future connections [25]. When devices are bonded, they can encrypt their connections without having to exchange keys. Bluetooth Low Energy uses AES-CCM encryption after the key exchange process has been completed.

Bluetooth Low Energy uses a secure simple pairing model where devices use one of the following pairing modes:

- **Just Works:** This mode offers little protection. The mode sets the temporary key to all zeroes, enabling any eavesdropper to immediately guess the temporary key. The Bluetooth Special Interest Group documentation [2] notes that Just Works provides no protection against eavesdropping and man-in-the-middle attacks.

- **Passkey Entry:** This mode requires the user and device to use the same six-digit PIN as the temporary key, while the rest of the 128-bit AES key is padded with zeroes. Passkey entry provides only slightly more protection against eavesdropping and man-in-the-middle attacks than Just Works. In fact, previous research has shown that it can be brute forced [24].

- **Numeric Comparison:** This mode is similar to passkey entry, except that both devices input a six-digit PIN independently. This greatly reduces the probability of brute forcing both the PINs.

- **Out-of-Band Communications:** This mode employs the full 128-bit temporary key that is communicated over a non Bluetooth Low Energy channel, typically using near field communications (NFC) technology. Another method is to send the temporary key over Bluetooth Classic because most devices are already equipped to handle both Bluetooth Low Energy and Bluetooth Classic. However, this method is not typical and was only implemented in one of the seventeen devices tested in this research. The use of an out-of-band channel is extremely important and is the best option when using secure simple pairing [10].

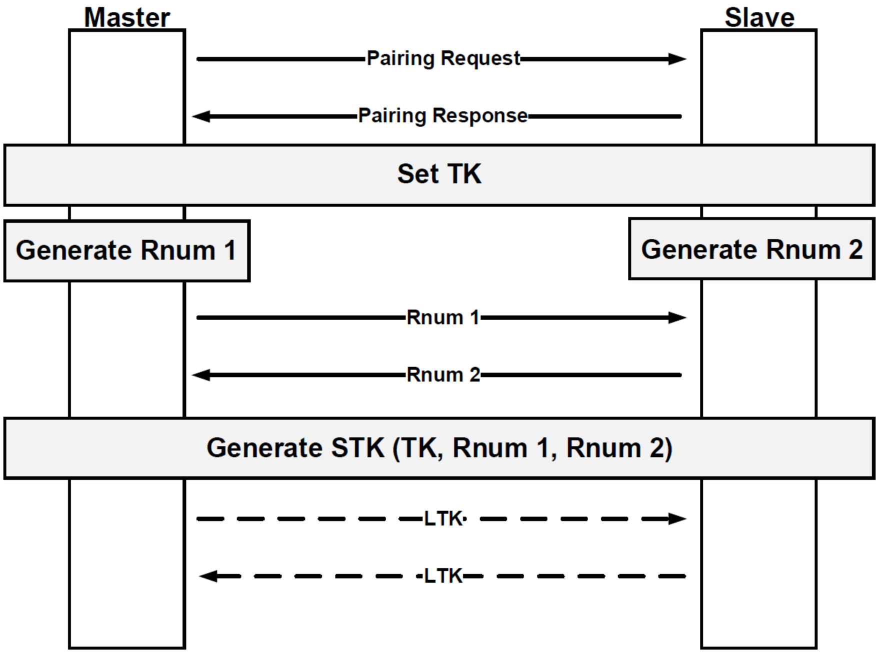


Figure 10. Bluetooth Low Energy Version 4.1 long-term key generation.

The numeric comparison technique is a simple solution for manufacturer implementation because the Bluetooth protocol already supports this type of authentication. However, the technique is only practical if developers use the key exchange improvements specified in Bluetooth Version 4.2. Unfortunately, all the developers whose products were investigated have not used these improvements. Therefore, a new key generation process is incorporated in Bluetooth Version 4.2. This process uses Elliptic Curve Diffie-Hellman (ECDH) key generation and implements new procedures for key generation.

Bluetooth Version 4.1 Link Layer Encryption. Pairing and bonding protect against compromises of plaintext and obfuscated passwords as well as brute forcing and fuzzing attacks. An added feature when using pairing and bonding is the ability to establish link layer encryption. The encryption method used in versions 4.0 and 4.1 is derived from the devices being paired initially and uses the long-term key as shown in Figure 10.

The process for generating the long-term key begins with the temporary key determined through the pairing modes mentioned above (i.e., Just Works, passkey entry, numeric comparison and out-of-band communications). The temporary key is used to encrypt the short-term key, which is generated using the temporary key and two random numbers from the master and slave. Finally,

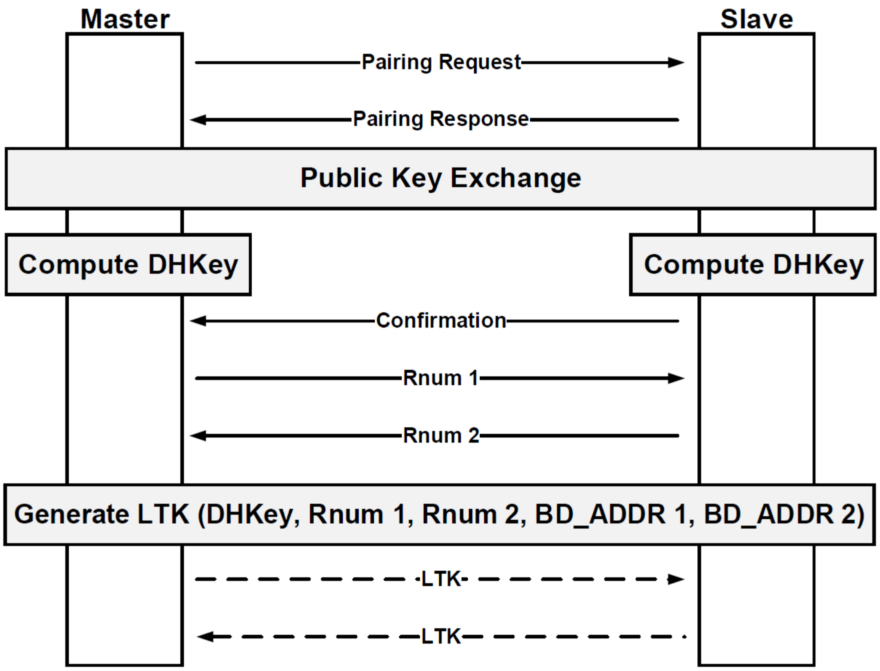


Figure 11. Bluetooth Low Energy Version 4.2 long-term key generation.

the short-term key is used to encrypt the long-term key, which is saved and used for all other communications.

Bluetooth Version 4.2 Link Layer Encryption. Bluetooth Low Energy Version 4.2 no longer uses a short-term key; instead, it uses a key derived from an ECDH key. Figure 11 shows the new key generation method. First, a pairing request occurs, which establishes the key generation method. A public key is exchanged to initiate the long-term key generation process. After the public key is exchanged, each device independently computes an ECDH key using the public key of the other device. Next, the slave computes a confirmation message that the master uses to check against its own key. If the check succeeds, the master sends a random number to the slave. The slave responds with a random number, which initiates the long-term key generation process.

The long-term key requires five parameters: (i) ECDH key; (ii) random number 1 from the master; (iii) random number 2 from the slave; (iv) Bluetooth device address of the master; and (v) Bluetooth device address of the slave. The major change in the protocol is that the ECDH key is never transmitted and is computed independently to protect against eavesdropping. Passive eavesdropping is no longer possible in version 4.2 because of the difficulty of guessing the private key [9]. The other information needed to generate the long-term key in version 4.2 comprises the random numbers sent by each device and their Bluetooth device addresses (BD_ADDR). The connection is encrypted and authenticated after the long-term key is established and this key is used for all future connections.

6.2 Application Layer Encryption

Application layer encryption is one of the most popular methods for securing Bluetooth Low Energy devices. The rationale for application layer encryption is that it does not require new devices to be paired; instead, it relies on the user and device to establish keys to encrypt and decrypt credentials. Application layer encryption can be more complicated than using the standard pairing offered by Bluetooth Low Energy due to the difficulty of managing keys [11]. However, the additional complexity of application layer encryption adds an extra layer of security when it is combined with link layer encryption. Good cryptographic practices (e.g., true random number generation and non-proprietary encryption algorithms) are encouraged for vendor implementations. Application layer encryption protects against attacks on plaintext passwords and obfuscated passwords, as well as brute forcing and fuzzing.

6.3 Two-Way Authentication

Two-way authentication protects against a rogue device attack by forcing the user and device to not (immediately) trust the connection. This method does not use link layer encryption. Instead, public/private keys are used for authentication by devices. For example, the public key of the user is used by the lock to encrypt the nonce $N1$, which is sent to the user. At the same time, the lock sends a plaintext nonce $N2$ to the user. The user decrypts $N1$ with his/her private key and encrypts $N2$ with the public key of the lock. Next, the user replies to the lock with the decrypted $N1$ and encrypted $N2$. Using an asynchronous encryption method ensures that the user and lock have public and private keys, and prevents a rogue device from impersonating a legitimate device. A rogue device would not be able to attack a connection without the private key of one of the devices and the public key of the other device. Thus, two-way authentication protects against attacks on plaintext passwords and obfuscated passwords, as well as brute forcing, fuzzing and man-in-the-middle attacks.

6.4 Geofencing

Geofencing protects against unauthorized access by requiring a user to be within a specific distance of designated GPS coordinates in order to request credentials from a web server. Essentially, a virtual fence is created around a device, where a user must be within a set distance (usually a few feet) to gain access. Geofencing prevents cloned devices from tricking users into providing their credentials. Thus, geofencing protects against rogue device and relay attacks.

The August Lock, a Bluetooth Low Energy lock, implements geofencing. However, geofencing is best combined with other mitigation techniques because it offers no protection against eavesdropping when a user is within the geofence perimeter. Other attacks are also possible in the case of the August Lock. These include replacing the firmware with malicious code or gaining access to developer-only features that should have been removed before the application was released [13].

6.5 Bluetooth Low Energy Guardian

Bluetooth Low Energy Guardian protects user privacy using an administrative program to control which entities can discover, scan and connect to a device [6]. Bluetooth Low Energy Guardian also defends against advertisements. Most man-in-the-middle attacks leverage advertisement packets that do not provide privacy and security protection. However, Bluetooth Low Energy Guardian controls advertisement packets via reactive jamming and by managing the connection request approval process. These protections are required because true advertisement packets are shielded from passive eavesdropping.

The implementation of Bluetooth Low Energy Guardian requires an Ubertooth One in addition to the device being protected. An advantage of this approach is that it actively prevents attacks on a device compared with a standard encryption approach. No additional implementation is required on the part of the device manufacturer and the approach can be used in conjunction with other techniques to enhance security. The downside of the approach is that it requires additional hardware, which may not be practical. Bluetooth Low Energy Guardian protects against man-in-the-middle, rogue device and relay attacks.

7. Conclusions

This research has sought to enhance the security of Bluetooth Low Energy devices. Thirteen of the seventeen devices subjected to testing have vulnerabilities that can be mitigated using the security solutions presented in this chapter. Countermeasures to Bluetooth Low Energy attacks require minimal development and implementation efforts on the part of device manufacturers. The existing pairing and bonding feature of Bluetooth Low Energy provides adequate security to defeat basic attacks. Sophisticated mitigation techniques are expensive to implement, but they offer additional protection against advanced attacks. An alternative protection method is to shield device activity as in the case of Bluetooth Low Energy Guardian, but this requires significant implementation and maintenance efforts.

Note that the views expressed in this chapter are those of the authors and do not reflect the official policy or position of the U.S. Air Force, U.S. Army, U.S. Department of Defense or U.S. Government.

References

[1] R. Baldwin, Researcher finds huge security flaws in Bluetooth locks, *Engadget*, August 10, 2016.

[2] Bluetooth Special Interest Group, Specification of the Bluetooth System, Covered Core Package Version 4.2, Master Table of Contents and Compliance, Specification Volume 0, Kirkland, Washington, 2014.

[3] Bluetooth Special Interest Group, Bluetooth 5 quadruples range, doubles speed, increases data broadcasting capacity by 800%, Kirkland, Washington, June 16, 2016.

[4] S. Boonkrong and C. Somboonpattanakit, Dynamic salt generation and placement for secure password storing, *International Journal of Computer Science*, vol. 43(1), 2016.

[5] Entrust, Advanced Solutions for Critical Infrastructure Protection: Complying with the North American Electric Reliability Corporation Critical Infrastructure Protection Standards, Dallas, Texas, 2012.

[6] K. Fawaz, K. Kim and K. Shin, Protection privacy of BLE device users, *Proceedings of the Twenty-Fifth USENIX Security Symposium*, pp. 1205–1221, 2016.

[7] N. Flor and H. Shannon, Technology corner: Brute force password generation – Basic iterative and recursive algorithms, *Journal of Digital Forensics, Security and Law*, vol. 6(3), pp. 79–85, 2011.

[8] S. Gold, Android, a secure future at last? *Engineering and Technology*, vol. 7(2), pp. 50–54, 2012.

[9] K. Haataja, K. Hypponen, S. Pasanen and P. Toivanen, *Bluetooth Security Attacks: Comparative Analysis, Attacks and Countermeasures*, Springer-Verlag, Berlin Heidelberg, Germany, 2013.

[10] K. Haataja and P. Toivanen, Two practical man-in-the-middle attacks on Bluetooth secure simple pairing and countermeasures, *IEEE Transactions on Wireless Communications*, vol. 9(1), pp. 384–392, 2010.

[11] D. Herrmann, *Complete Guide to Security and Privacy Metrics: Measuring Regulatory Compliance, Operational Resilience and ROI*, Auerbach Publications, Boca Raton, Florida, 2007.

[12] R. Heydon, *Bluetooth Low Energy: The Developer's Handbook*, Pearson Education, Upper Saddle River, New Jersey, 2013.

[13] Jmaxxz, Backdooring the front door: Hacking a "perfectly secure" smart lock, presented at *DEF CON 24*, 2016.

[14] C. Kaufman, R. Perlman and M. Speciner, *Network Security: Private Communication in a Public World*, Prentice Hall, Upper Saddle River, New Jersey, 2002.

[15] National Security Telecommunication Advisory Committee, NSTAC Report to the President on Secure Government Communications, Washington, DC, 2013.

[16] T. O'Connor, *Violent Python: A Cookbook for Hackers, Forensic Analysts, Penetration Testers and Security Engineers*, Syngress, Waltham, Massachusetts, 2013.

[17] P. Oehlert, Violating assumptions with fuzzing, *IEEE Security and Privacy*, vol. 3(2), pp. 58–62, 2005.

[18] Onity, Electronic Locks, Salem, Oregon (`en.onity.com/products/Pages/Electronic-Locks.aspx`), 2016.

[19] J. Padgette, K. Scarfone and L. Chen, Guide to Bluetooth Security, NIST Special Publication 800-121, Revision 1, National Institute of Standards and Technology, Gaithersburg, Maryland, 2012.

[20] G. Press, Internet of Things by the numbers: Markets estimates and forecasts, *Forbes*, August 22, 2014.

[21] B. Ramsey, B. Mullins, W. Lowder and R. Speers, Sharpening the stinger: Tuning KillerBee for critical infrastructure warwalking, *Proceedings of the IEEE Military Communications Conference*, pp. 104–109, 2014.

[22] B. Ramsey, B. Mullins, R. Speers and K. Batterton, Watching for weaknesses in wild WPANs, *Proceedings of the IEEE Military Communications Conference*, pp. 1404–1409, 2013.

[23] A. Rose and B. Ramsey, Picking Bluetooth Low Energy locks from a quarter mile away, presented at *DEF CON 24*, 2016.

[24] M. Ryan, Bluetooth: With low energy comes low security, *Proceedings of the Seventh USENIX Conference on Offensive Technologies*, 2013.

[25] K. Townsend, C. Cufi, Akiba and R. Davidson, *Getting Started with Bluetooth Low Energy: Tools and Techniques for Low-Power Networking*, O'Reilly Media, Sebastopol, California, 2014.