SANS

www.sans.org

**SECURITY 560**

NETWORK PENETRATION TESTING AND ETHICAL HACKING

560.1

# Planning, Scoping, and Recon

*The right security training for your staff, at the right time, in the right location.*

IMPORTANT-READ CAREFULLY:



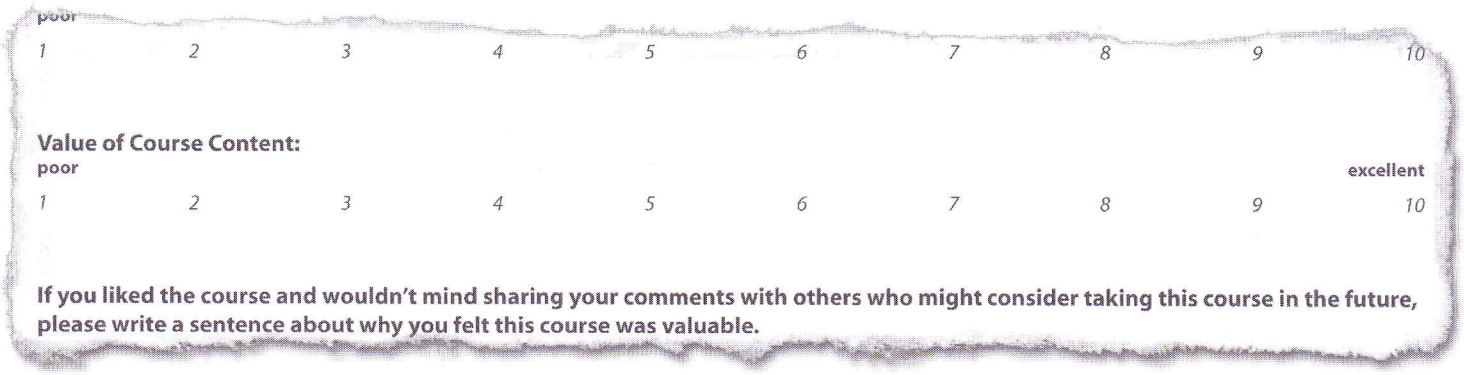
V2011_0418

# Note to Students
# About SANS Institute Evaluations

SANS Institute works extremely hard to continuously improve both the quality of courseware and instruction. The evaluation forms that we provide daily are one of the most effective methods to help improve the course and to give feedback to the instructors. The numerical score helps us track trends, and the written comments give us the context to make immediate improvements. We truly appreciate the time you take to provide this feedback.

Some things to remember: 10 = very good, 1 = very bad!

***Your scores and comments should be given based on your experience, the quality of the instruction, and the quality of the course material. Instructors should NOT be soliciting scores or in any way trying to influence your evaluations.***




Please make every effort to score each of these areas individually and write comments which you feel are relevant or important in the comments section. If a lab does not work on your computer system, please reflect that in the course content score, and please give us any information you have as to what went wrong so we make appropriate corrections to our courseware. Rest assured that all eval comments are read by the senior management at SANS. Evaluations are used to adapt the course and make it even better, so please do fill them in! If you have specific comments regarding the venue, snacks, etc., we have provided a separate evaluation specifically for those areas.

Some people do not believe in giving 10s, and that is fine. However, please keep in mind that delivering a course is a performance, and performers work harder when they are motivated. If and only if your course is on par with the best instruction you have ever received a 10 is appropriate.

If you give the overall course and/or course content a low score, please explain why in the comments field so we can adapt. You can also be assured if you explain your concerns to an event manager we will be discreet. If you are willing, please sign your name at the bottom of the evaluation. If you are having difficulties, SANS can quickly come to your aid if we know who you are. In addition, we also routinely use quotes from previous students in upcoming brochures. If you do not want to be quoted, please let us know.

Thank you for your thoughtful consideration,

Eric Bassel, Mason Brown, Stephen Northcutt, and Alan Paller

This page intentionally left blank.

# The SANS Institute

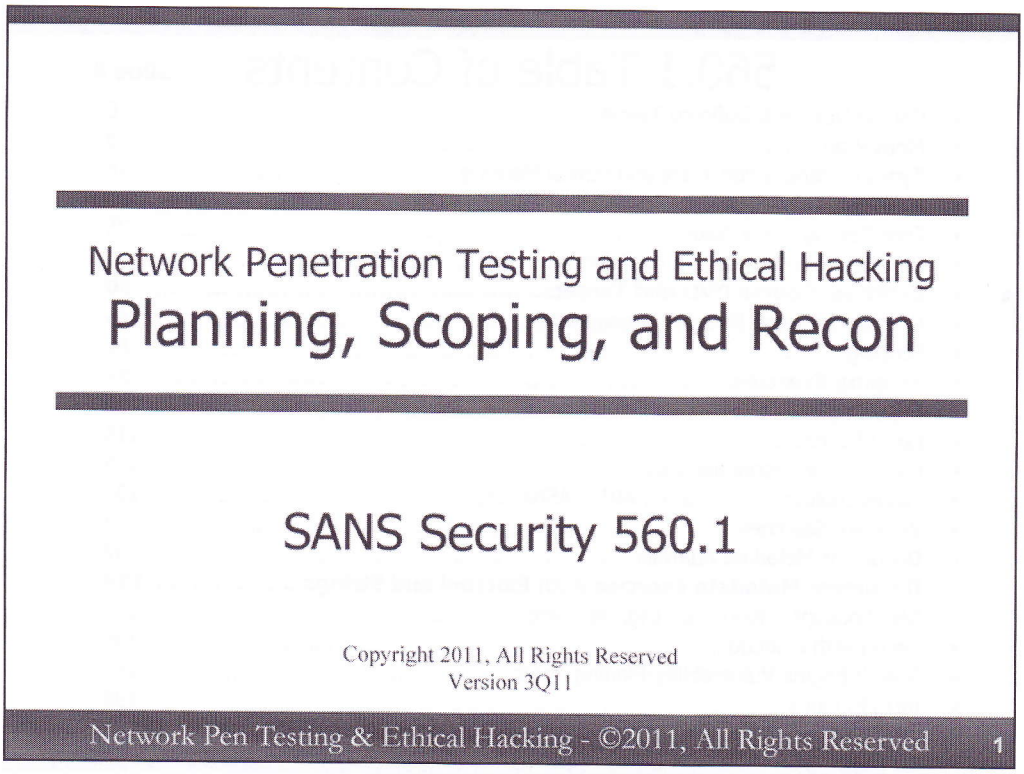
## Code of Ethics

I certify that by having access to tools and programs that can be used to break or "hack" into systems, that I will only use them in an ethical, professional and legal manner. This means that I will only use them to test the current strength of security network so that proper improvements can be made. I will always get permission before running any of these tools on a network. If for some reason I do not use these tools in a proper manner, I do not hold SANS or the presenter liable and accept full responsibility for my actions.

Name_____ Signature_____

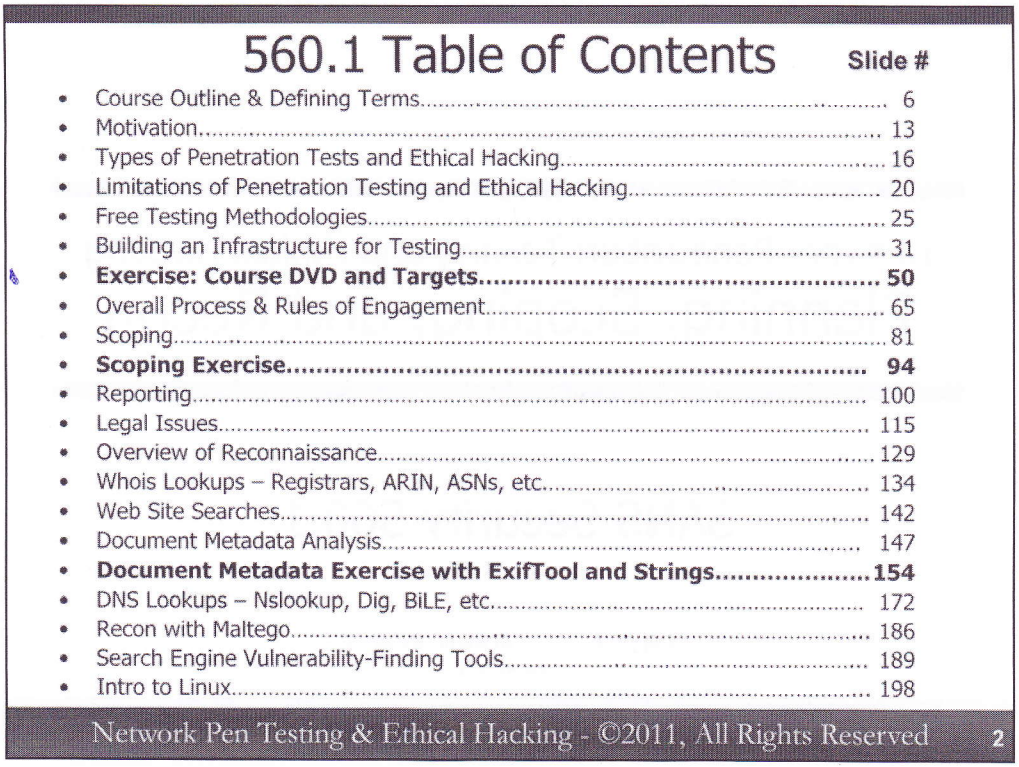
Company_____ Date_____

This page intentionally left blank.

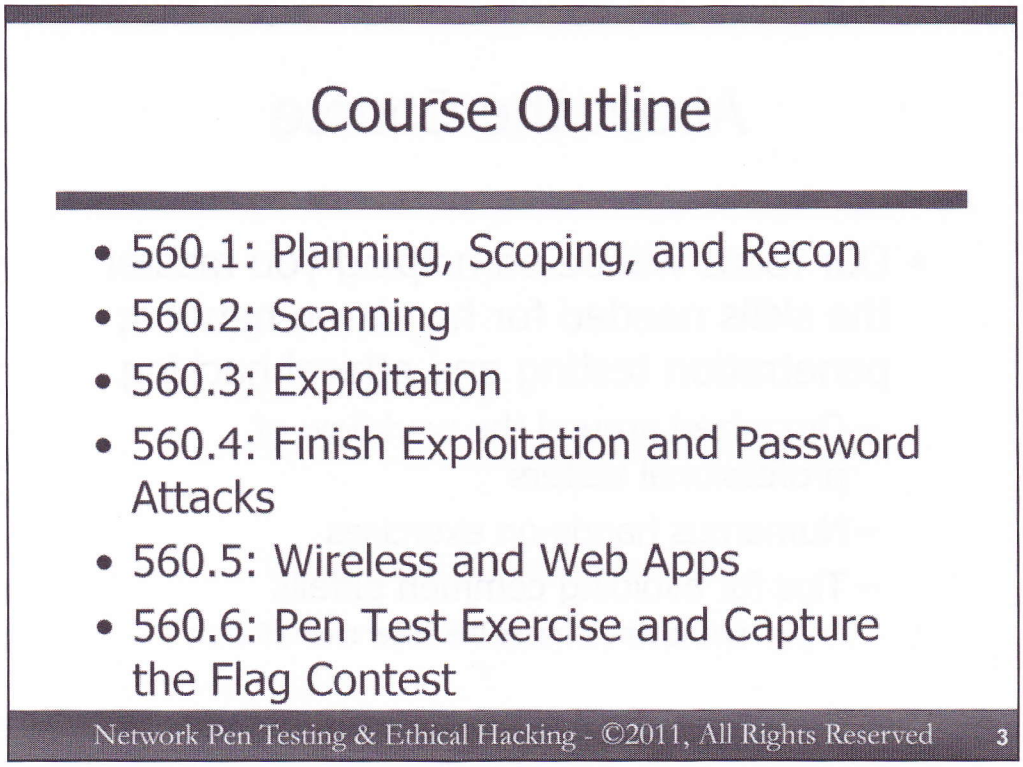



Hello and welcome to this course! The purpose of this course is to prepare you to perform ethical hacking and penetration testing exercises in a professional, safe, and repeatable manner for your organization. Throughout this course, we'll cover hundreds of tools and techniques, detailing how you can use them to find vulnerabilities in your organization to help improve your security stance. We'll have hands-on exercises throughout, culminating in a Capture-the-Flag penetration test exercise for the entirety of 560.6.

Let's keep this session interactive. If you have a question, please let the instructor know. Discussions about relevant topics are incredibly important in a class like this, as we have numerous attendees with various levels of skill coming into the class. Share your insights, and ask questions. The instructor does reserve the right, however, to take a conversation off-line during a break or outside of class in the interest of time and applicability of the topic.




This slide is a table of contents, and also acts as an overview of what we will be discussing throughout 560.1.




This course is divided into several sections, each designed to prepare you in a vital aspect of network penetration testing and ethical hacking.

560.1 sets the stage, defining terms and providing a detailed discussion of the planning process, including building a penetration testing and ethical hacking infrastructure, establishing ground rules for testing, and scoping projects. This section also covers the Reconnaissance phase of a test.

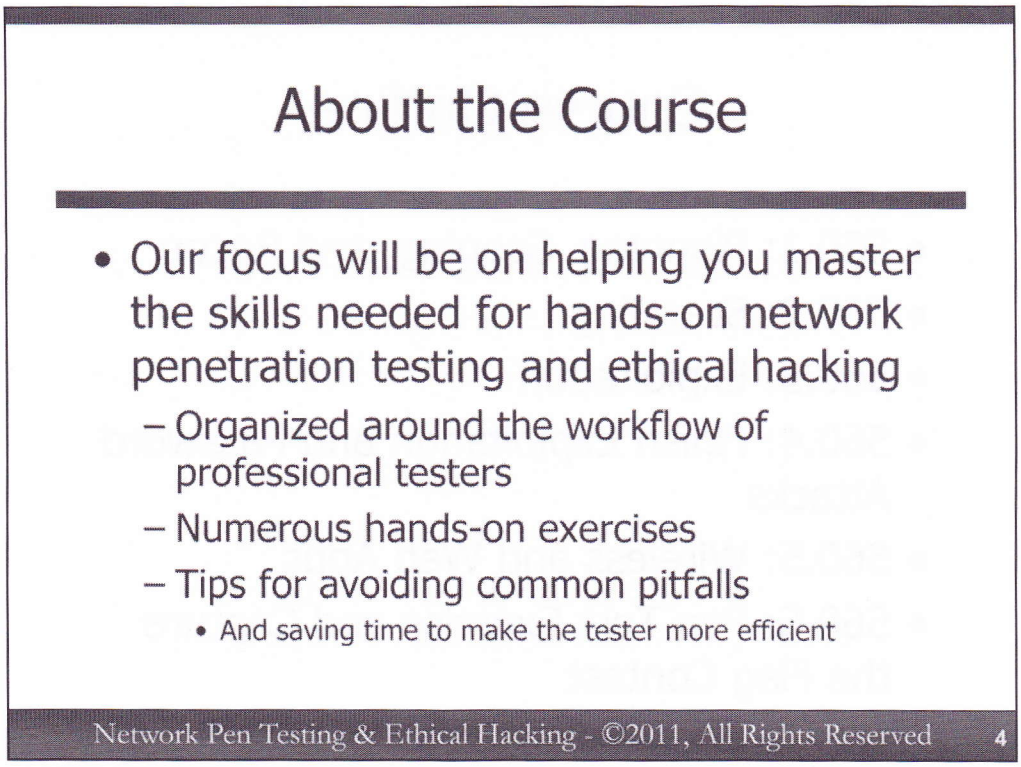
560.2 zooms into scanning, covering the tools and techniques that professional penetration testers and ethical hackers need to master to find target machines, openings on those targets, and vulnerabilities.

560.3 deals with exploits, talking about the different categories of exploits, the manner in which they are packaged, and how to use exploitation tools. This section of the class also covers some of the common pitfalls that come up after exploitation, and how to avoid these problems.

560.4 starts by finishing up with our exploitation discussion. Then, it focuses on passwords, one of the major weaknesses of many target organizations. We'll go over tips for various kinds of password attacks, and then focus on password guessing, password cracking, and pass-the-hash techniques, with in-depth, hands-on exercises for each.

560.5 deals with wireless LAN tests and web application tests. Although this is neither a wireless class nor web app course, network penetration testers and ethical hackers are often called upon to analyze wireless networks and web apps, so we will address each. The SANS Institute has other courses that spend between two and six full days dealing with each of these topics, but we'll cover the vitals for network penetration testers and ethical hackers in 560.5.

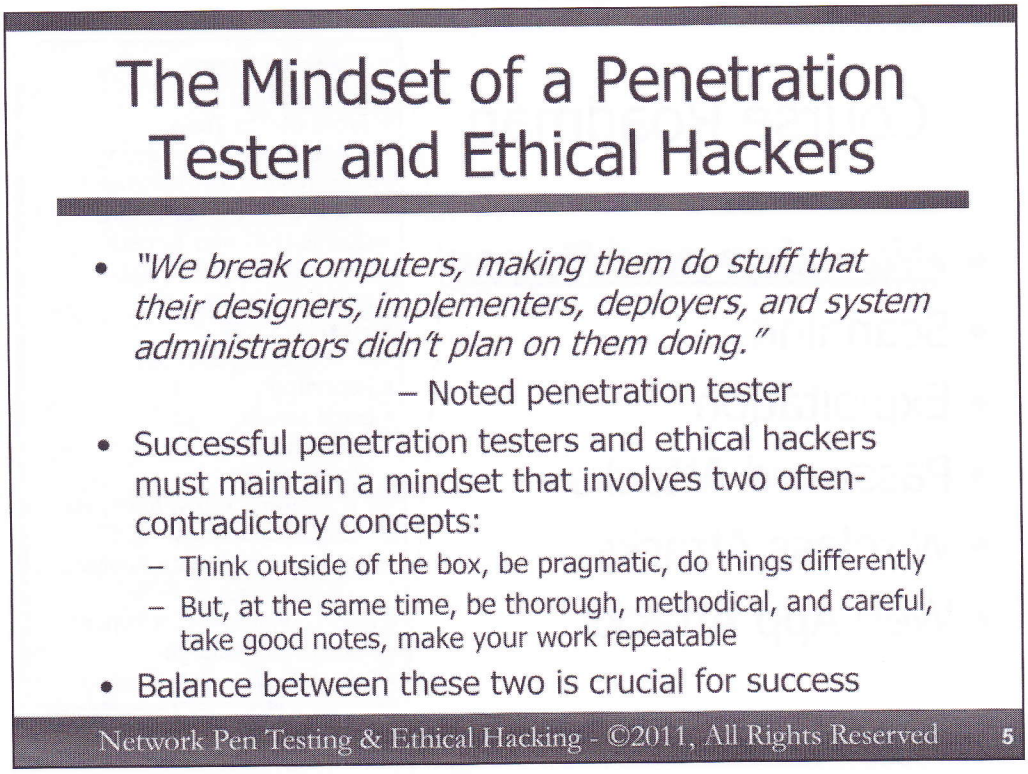
That all leads us to 560.6, in which we'll apply the concepts from throughout the course in a penetration test exercise, which includes a Capture the Flag (CtF) contest. The skills you master from exercises throughout the class will be applied in this lively contest on the last day.




This overall objective for this course is to help prepare you with the skills needed to perform network penetration testing and ethical hacking. Some people who take this class are professional penetration testers already, looking for some extra tips and tools for their arsenal. Others have never hacked a box before in their lives, and are looking to get started. We welcome attendees from across this spectrum. We have strived to develop the materials to help you master the skills of a network penetration tester and ethical hacker regardless of where you sit on that spectrum.

This course is organized around the workflow of a professional penetration tester and ethical hacker, describing the various steps and options a tester has at each step. Note that the general flow of work, however, isn't set in stone. Really good testers are pragmatic, often improvising based on the particulars of a given project when the opportunity arises. The class includes numerous hands-on exercises, each of which is designed to impart an important skill that network penetration testers and ethical hackers require.

The course is also chock full of tips for avoiding common pitfalls that network penetration testers and ethical hackers face. Based on input from numerous professional penetration testers who have learned these lessons the hard way, these tips throughout the course are designed to help you maximize the effectiveness of your own penetration practices. Also, many of these tips are designed to save you a lot of time, making you more efficient. Often, when testing, you will need to achieve some goal. One way of going about that goal may take 3 hours and work only 10% of the time, while another method might take 3 minutes and have a 90% success rate. Following the tips of this class will help you focus your valuable time on the latter.




At the very outset of this class, let's briefly explore the mindset of penetration testers and ethical hackers. A noted penetration tester, someone whose name you would likely recognize but who has requested anonymity, said:

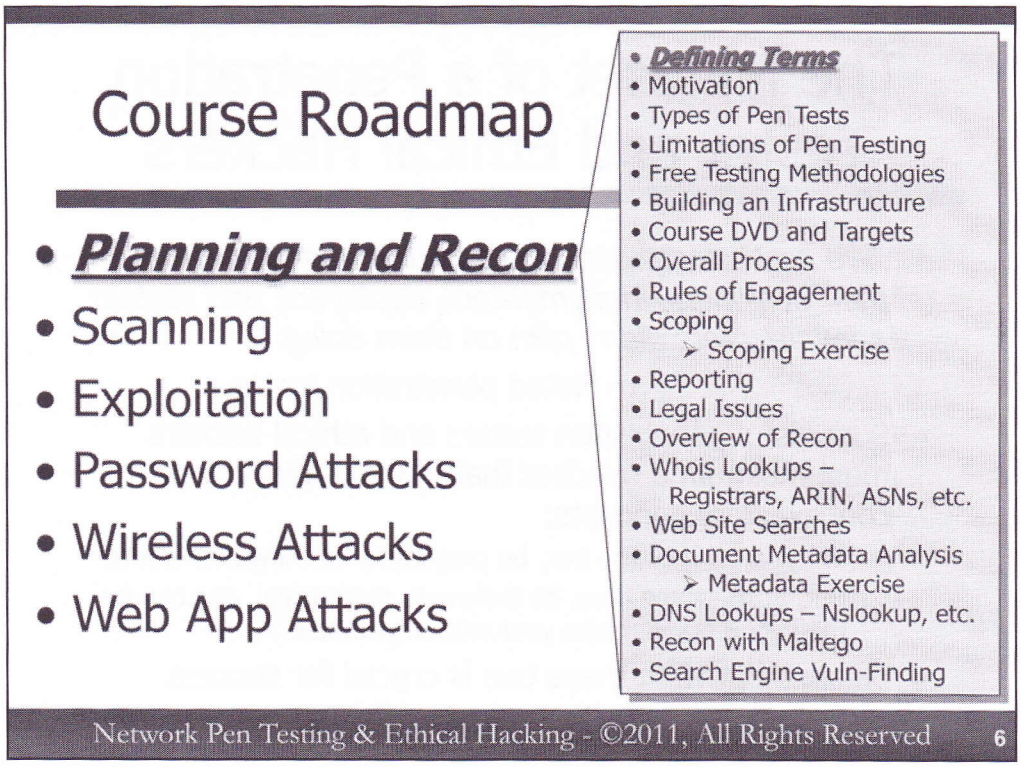
*"We break computers, making them do stuff that their designers, implementers, and system administrators didn't plan on them doing."*

That's what our job is: finding flaws that could allow attackers to do evil on target machines, so that vulnerabilities can be fixed before mayhem ensues. However, to successfully achieve that goal, penetration testers and ethical hackers must maintain a mindset that involves two often-contradictory concepts.

First, a penetration tester or ethical hacker must be flexible and pragmatic, thinking outside of the box. To be successful, you'll need to think differently than most traditional system administrators or network architects, trying to solve problems in often-untraditional ways.

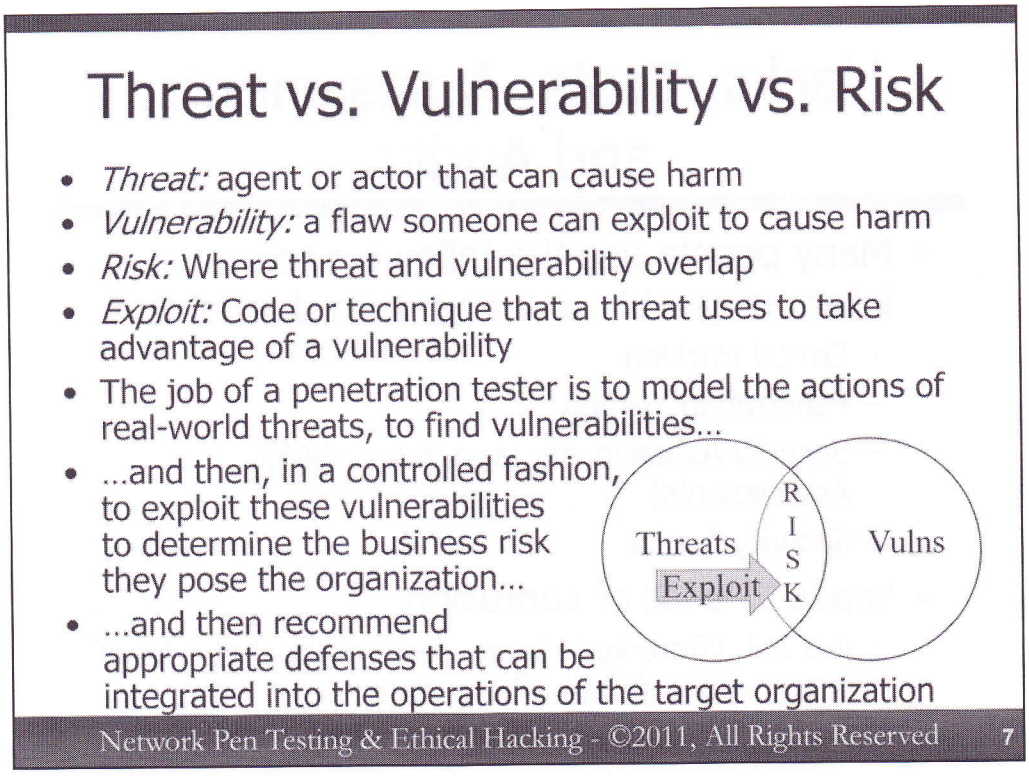
But, at the same time as you wield your pragmatic style, you have to be thorough, methodical, and careful. Your work, to be valuable, must be understandable and reproducible so that the target organization can understand its vulnerabilities and risks and take action to mitigate the flaws. You need to take good notes and produce a high-quality report that presents your findings in a digestible form for people who don't perform penetration testing or ethical hacking professionally -- people who may not share your pragmatic, think differently mindset.

Some people struggle with this mindset, erring by allowing one side to dominate over the other. However, many people are able to resolve this conflict between these two mindsets, balancing them. To be a successful penetration tester, we need to strive for this balance.




To start the session, we need to define some terms so that we can have a consistent terminology to use throughout the rest of the class. What is ethical hacking? How is it associated with penetration testing? How do a vulnerability scan and a penetration test differ? We'll address each of these questions next.

It is important to note that different people use the various terms we'll define in different ways. We are going to present a set of definitions that are very common, but not universal. In other words, we'll introduce this common terminology so that we can be consistent throughout this course and with the most common use of these terms. But, keep in mind that usage may vary for your own organization or with some of the enterprises that you test.




Ethical hacking and penetration testing are tools for dealing with threats, vulnerabilities, risks, and exploits. Many people in the information security business throw around these terms interchangeably, often confusing threats with risk, or vulnerabilities with exploits. Each has a distinct meaning, though, and the terms should be applied carefully.

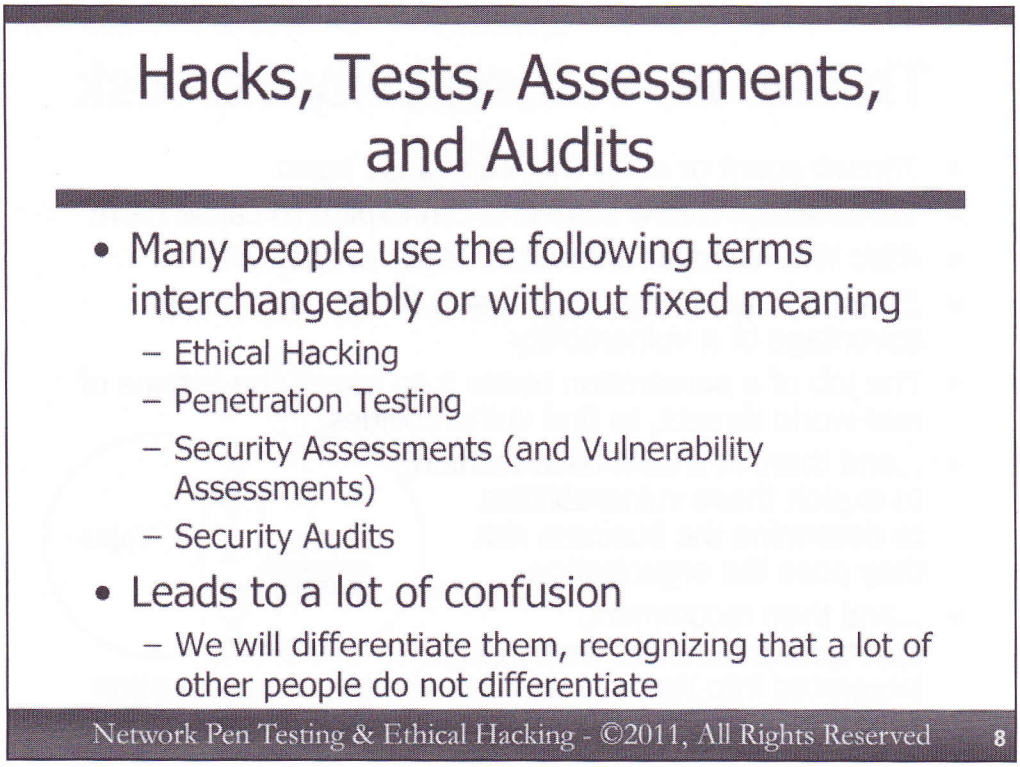
A *threat* is an actor or agent that may want to or actually can cause harm to the target organization. Threats include organized crime, spyware companies, and disgruntled internal employees who start attacking their employer. Worms and viruses also represent a threat, as they could cause harm in your organization even without a human directing them to do so by infecting machines and causing damage automatically. In this course, we will often refer to threats generically as "attackers" or "bad guys".

A *vulnerability* is some flaw in our environment that an attacker could use to cause damage. Vulnerabilities could exist in numerous arenas in our environments, including our architectural design, business processes, deployed software, and system configurations.

*Risk* is where threat and vulnerability overlap. That is, we have a risk when our systems have a vulnerability that a given threat can attack.

An *exploit* is the vehicle by which the attacker uses a vulnerability to cause damage to the target system. The exploit could be a package of code which generates packets that overflow a buffer in software running on the target. Alternatively, the exploit could be a social engineering scheme whereby the bad guy talks a user into revealing sensitive information, such as a password, over the phone.

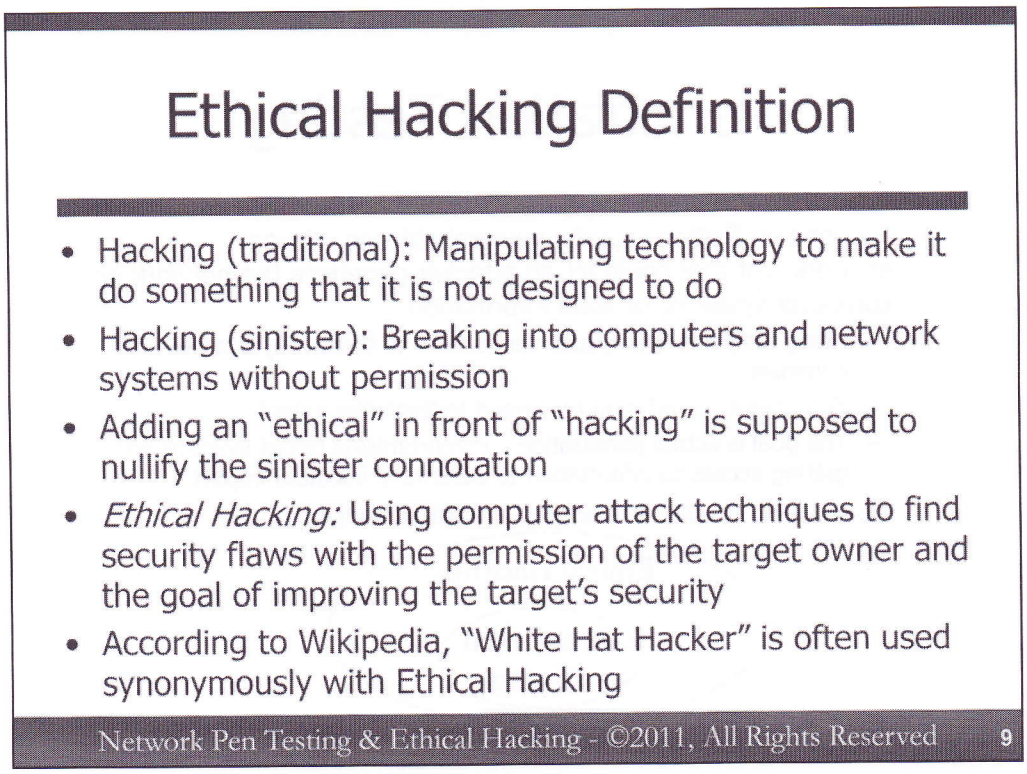
As security professionals, we have to work hard to minimize this risk by minimizing vulnerabilities and blocking threats. That's what penetration testing is all about: We model the activities of real-world threats to discover vulnerabilities. Then, through controlled exploitation, we attempt to determine the business risk associated with these flaws. We then recommend appropriate defenses. These recommendations must apply in light of the operations environment of the target organization. If we do this properly, we stand a significant chance of improving the security stance of the target organization.




There is another set of terms that many information security practitioners use interchangeably, which results in a lot of confusion. The following terms are associated with what an ethical hacker or penetration tester actually does on a day-to-day basis:

- Ethical Hacking
- Penetration Testing
- Security Assessments (and Vulnerability Assessments)
- Security Audits

While these terms are often used interchangeably, they do have subtle distinctions that we should observe.




The term "hacking" means different things to different people. Traditionally, hacking referred to exploration of technology, trying to understand it at a deep level to be able to manipulate it into doing something that it was not designed to do. Early "hackers" of this kind were often hobbyists or academics, with the noble goal of using technology in interesting and innovative ways.

Unfortunately, most people today think of hacking in more sinister terms: breaking into computer systems and accounts without the permission of their owner to make money illicitly or cause damage.

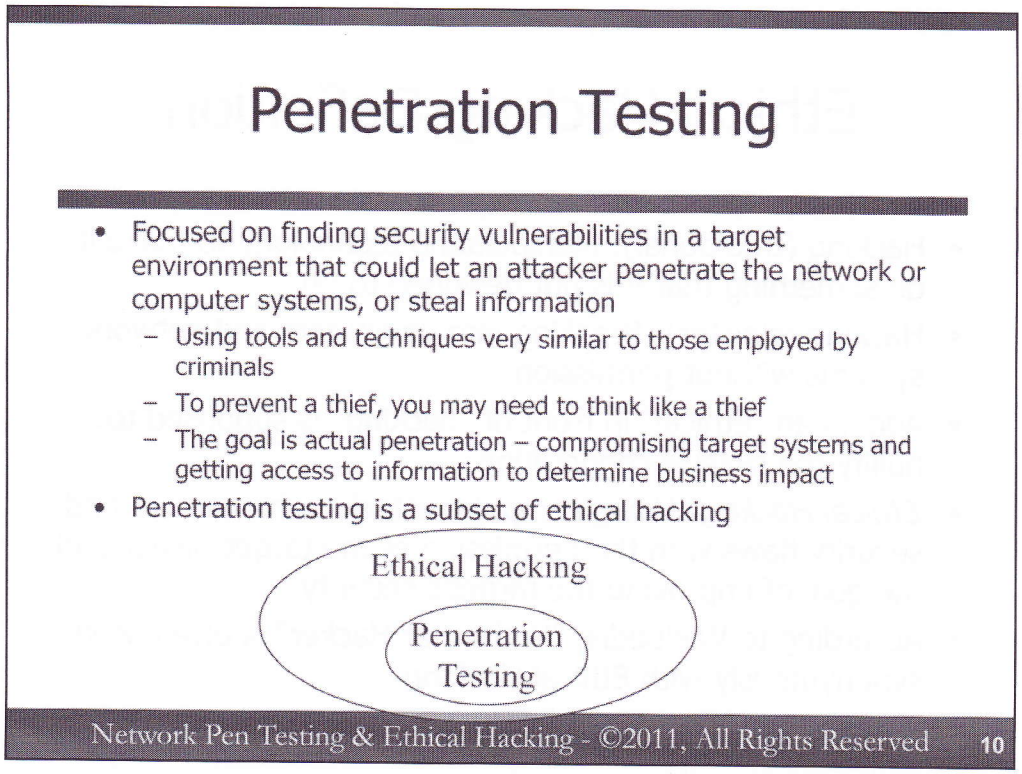
People started to use the term "ethical hacker" to refer to individuals who applied the process of breaking into computer systems, but with the wholesome purpose of finding security vulnerabilities so that they could be fixed. The hope is that the "ethical" prefix will nullify the sinister notions of hacking. Ethical hackers use some of the techniques and processes of the bad guys, but in a professional manner, with permission of the owners of the target systems, to try to improve the security of their targets.

Pulling all this together, we get the following definition of Ethical Hacking:

*Ethical Hacking* is the process of using computer attack techniques to find security flaws with the permission of the target owner and the goal of improving the target's security.

Throughout this class, we'll use the phrase in this non-evil sense.

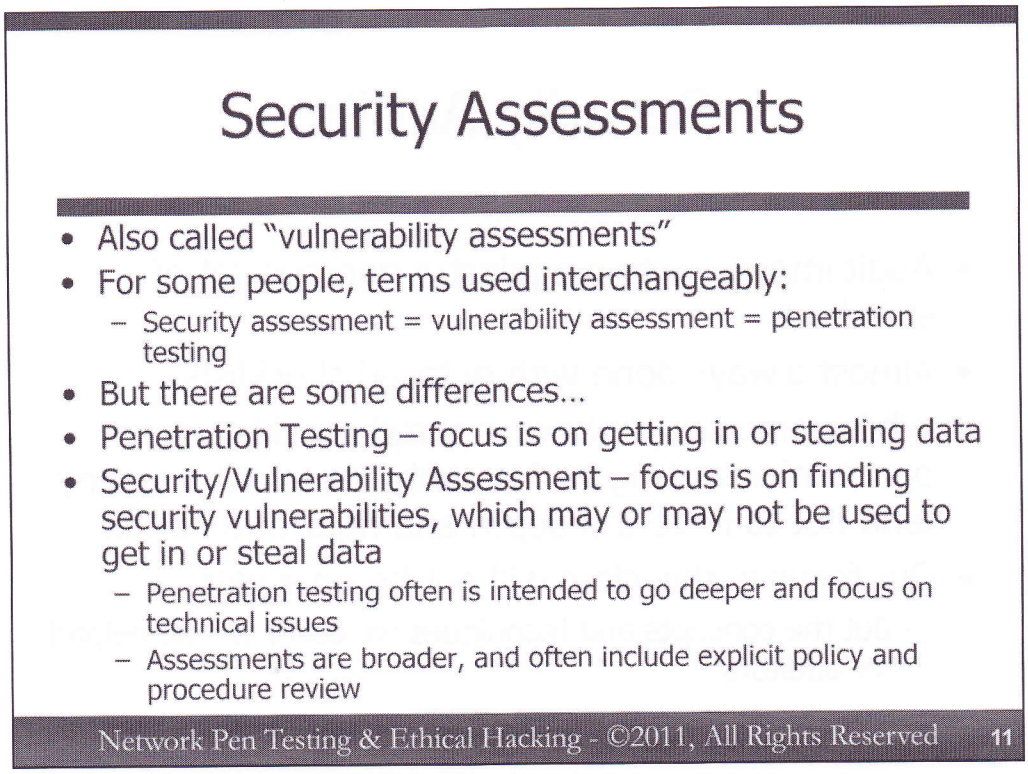
According to Wikipedia, ethical hacking is synonymous with "White Hat Hacking". This is to differentiate it from the "Black Hat Hacking", which again involves the sinister motive.




Penetration Testing is very closely related to ethical hacking. Indeed, throughout this course, we often refer to "ethical hacking and penetration testing" to bundle the terms together.

Some people use the term "ethical hacking" to mean the general process of using hacker techniques for good, which includes vulnerability discovery in a target organization's network, software product vulnerability research, and other tasks. In this view, *penetration testing* is a more narrowly focused phrase, dealing with process of finding flaws in a target environment with the goal of penetrating systems, taking control of them. Penetration testing, as its name implies, is focused on penetrating the target organization's defenses, compromising systems and getting access to information.

To summarize, ethical hacking is an expansive term encompassing all hacking techniques used for good, while penetration testing is more focused on the process of finding vulnerabilities in a target environment. In this view, penetration testing is a subset of ethical hacking.

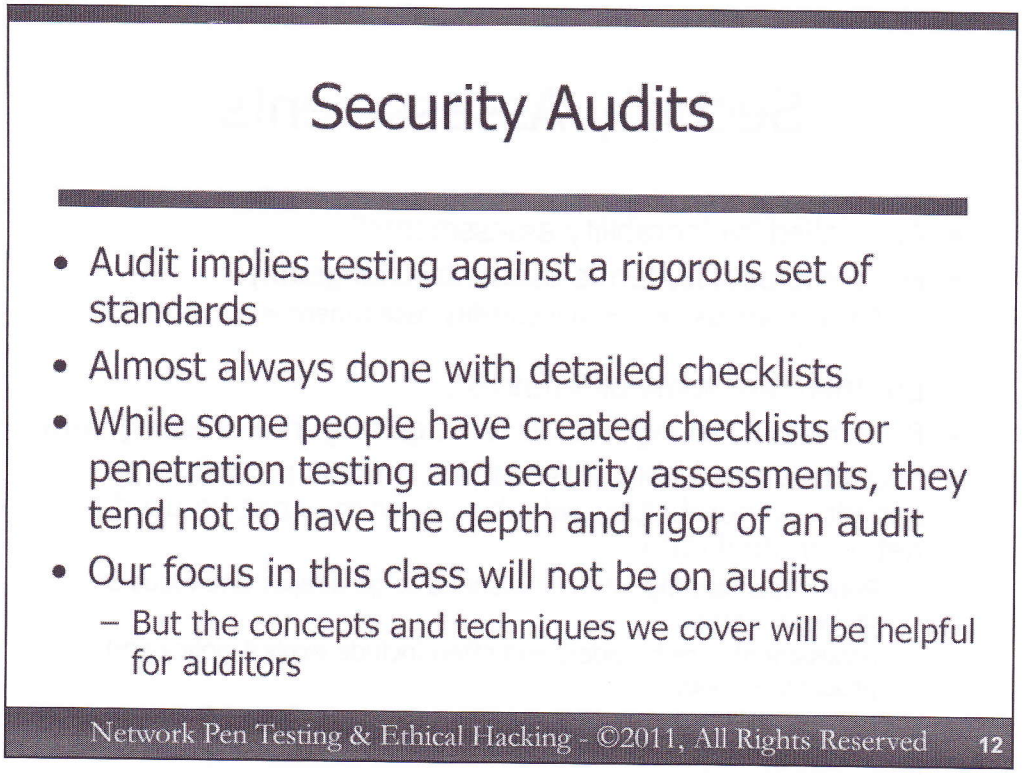



Many people use the phrases "Security Assessment" and "Vulnerability Assessments" to describe the work done by penetration testers and ethical hackers. But, there is a subtle distinction between the ideas of a penetration test and a security assessment.

A penetration test is focused on getting in or stealing data. The emphasis is on penetration of the target environment by exploiting discovered vulnerabilities.

*Security assessments* and *vulnerability assessments* are focused on finding vulnerabilities, often without regard to actually exploiting them and getting in.

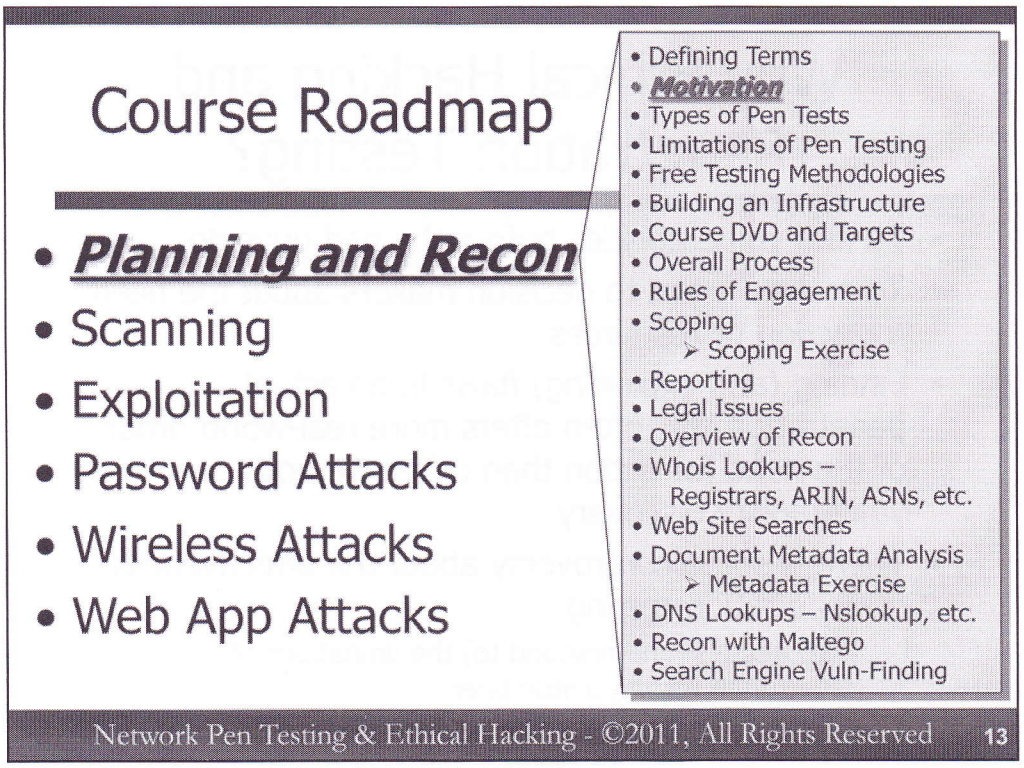
Thus, penetration testing often goes deeper, with its goal of taking over systems and stealing data, while security and vulnerability assessments are broader, involving the process of looking for security flaws. These assessments also often include policy and procedure review, which are usually not included in penetration testing.



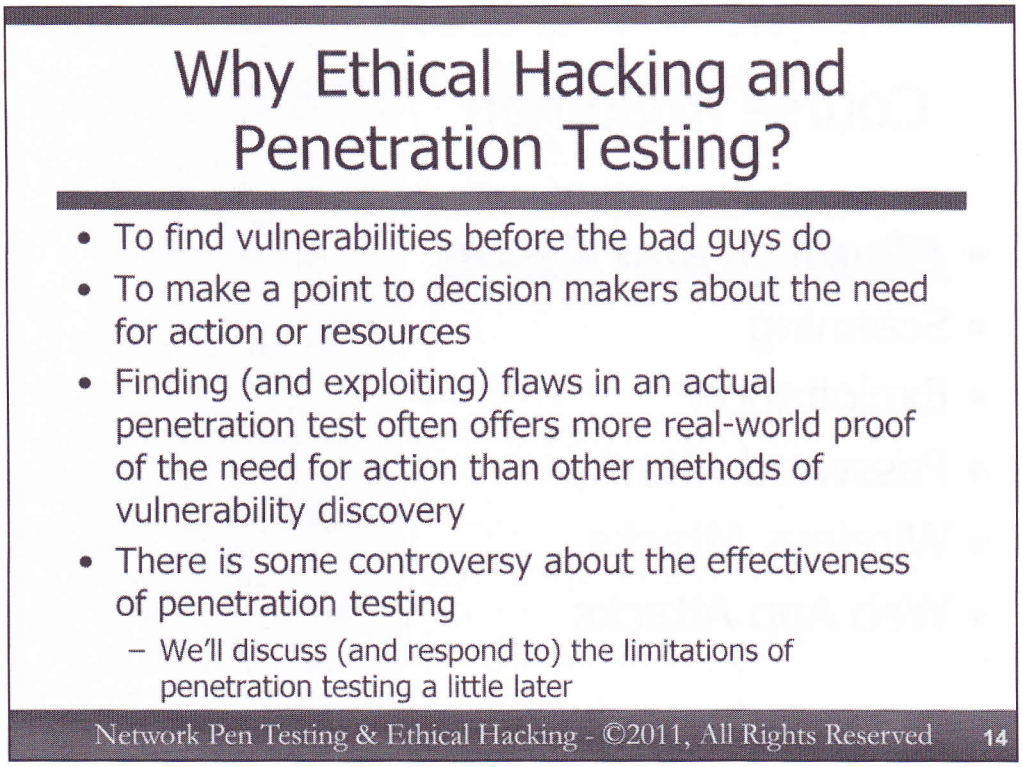

Finally, we have the phrase "Security Audit". Audit implies that we are measuring things against a fixed, pre-determined, rigorous set of standards. These audits are almost always done with detailed checklists.

Some penetration testing and ethical hacking organizations have created their own internal checklists of items that need to be covered in a test, but these checklists aren't as detailed as a comprehensive audit.

Our focus in this class will not be on auditing. SANS has numerous other classes that address security audits in detail. Our focus will be on ethical hacking and penetration testing.



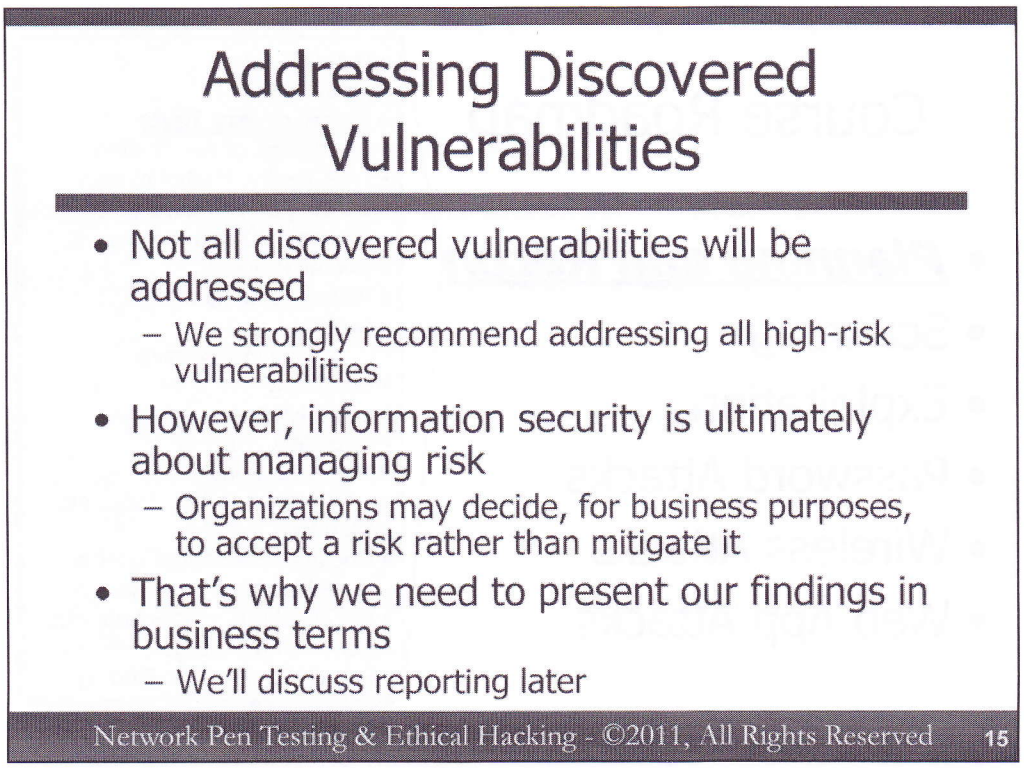

With that terminology covered, let's now turn our attention to the motivation for ethical hacking and penetration testing. What value do they provide to an organization? In short, why are we here?




Many organizations use ethical hacking and penetration testing to find security flaws before the bad guys do. After applying their security policies, procedures, and technology, organizations can use thorough penetration tests to see how effective their security really is in light of an actual attack, albeit by friendly attackers.

An added benefit of ethical hacking and penetration testing is that, because they show real vulnerabilities and indicate what a malicious attacker might be capable of achieving, they can get management's attention. Decision makers, when presented with the carefully formulated results of a test in business terms, are more likely to provide resources and attention to improve the security stance of an organization.

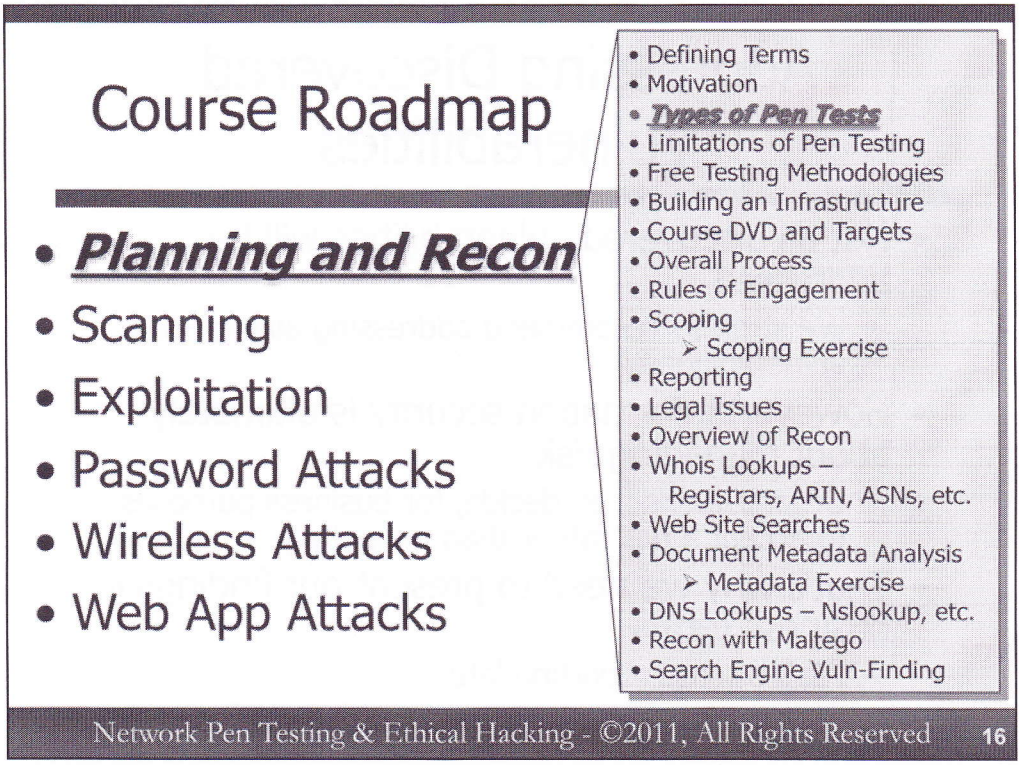
There is some controversy about the value of ethical hacking and penetration testing. Detractors point out that tests are limited, in that they are based on the skill of the tester, the duration of the test, and currently known vulnerabilities. We'll discuss and respond to these limitations a little later in the course.



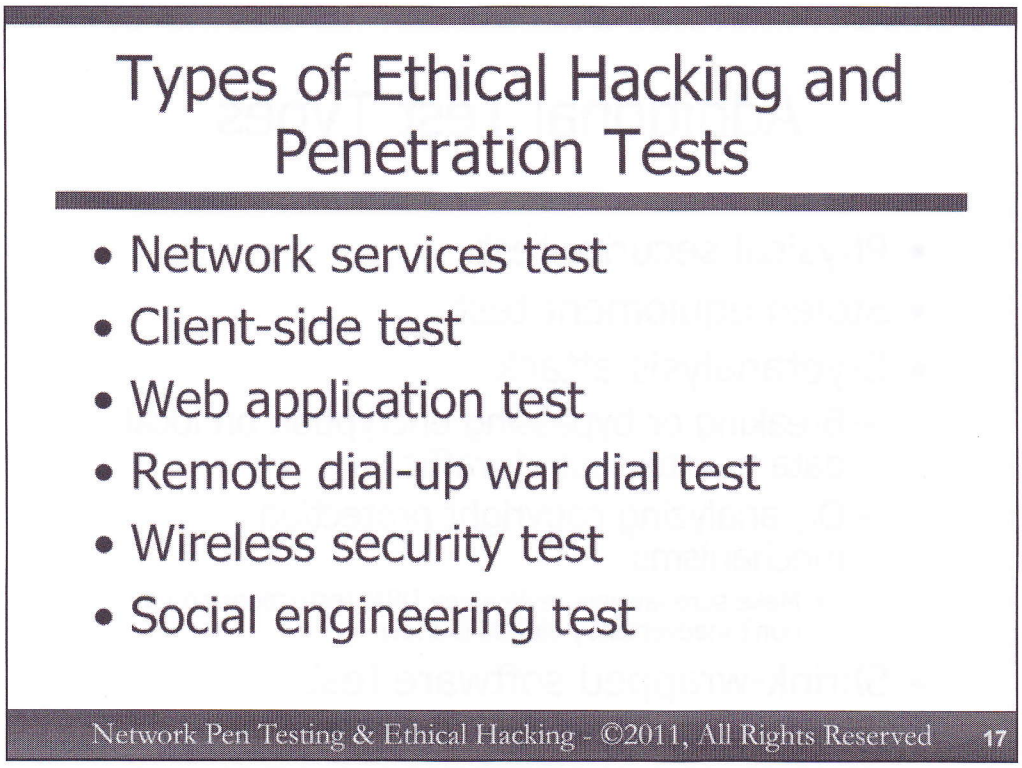

A major goal of penetration testing and ethical hacking is discovering flaws so that they can be remediated (by applying patches, reconfiguring systems, altering the architecture, changing processes, etc.). However, it is important to note that in most tests, not all of the discovered vulnerabilities are actually addressed.

We recommend that all high-risk vulnerabilities be addressed in a timely fashion, but the truth is that some vulnerabilities linger long after a test is complete, even high-risk issues. Remember, information security is all about managing risk, not eliminating it. Decision makers in an organization may conclude that, for business purposes, they will accept a given risk identified during a test, rather than mitigate the associated vulnerability. In the end, it's a business decision, informed by our input.

For this reason, we have to present our findings in both *business* and *technical* terms. That's an important principle to remember throughout this course. We'll get into reporting in more detail later.




There is a large number of different types of ethical hacking and penetration tests. Let's now explore the different types, realizing that many of the tests we'll engage in for our jobs will be a mixture of a subset of these various types.




There are numerous kinds of ethical hacking and penetration tests. They include:

Network services test: This is one of the most common types of tests, and involves finding target systems on the network, looking for openings in their underlying operating systems and available network services, and then exploiting them remotely. Some of these network service tests happen remotely across the Internet, targeting the organization's perimeter networks. Others are launched locally, from the target's own facilities, to evaluate the security of their internal network or the DMZ from within, seeing what kinds of vulnerabilities an internal user could discover.

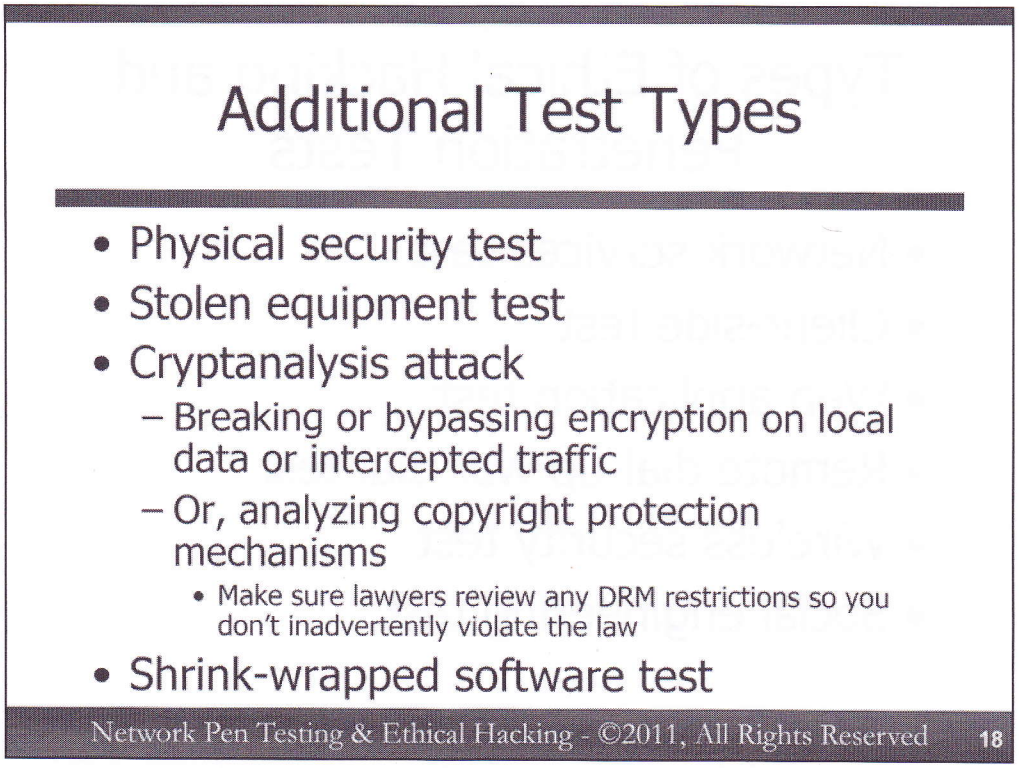
Client-side test: This kind of test is designed to find vulnerabilities in and exploit client-side software, such as browsers, media players, document editing programs, etc.

Web application test: These tests look for security vulnerabilities in the web-based applications deployed on the target environment.

Remote dial-up war dial: These tests look for modems in a target environment, and often involve password guessing to login to systems connected to discovered modems.

Wireless security test: These tests involve exploring a target's physical environment to find unauthorized wireless access points or authorized wireless access points with security weaknesses.

Social engineering test: This type of test involves attempting to dupe a user into revealing sensitive information such as a password. These tests are often conducted over the phone, targeting selected help desks or users, evaluating processes, procedures, and user awareness.



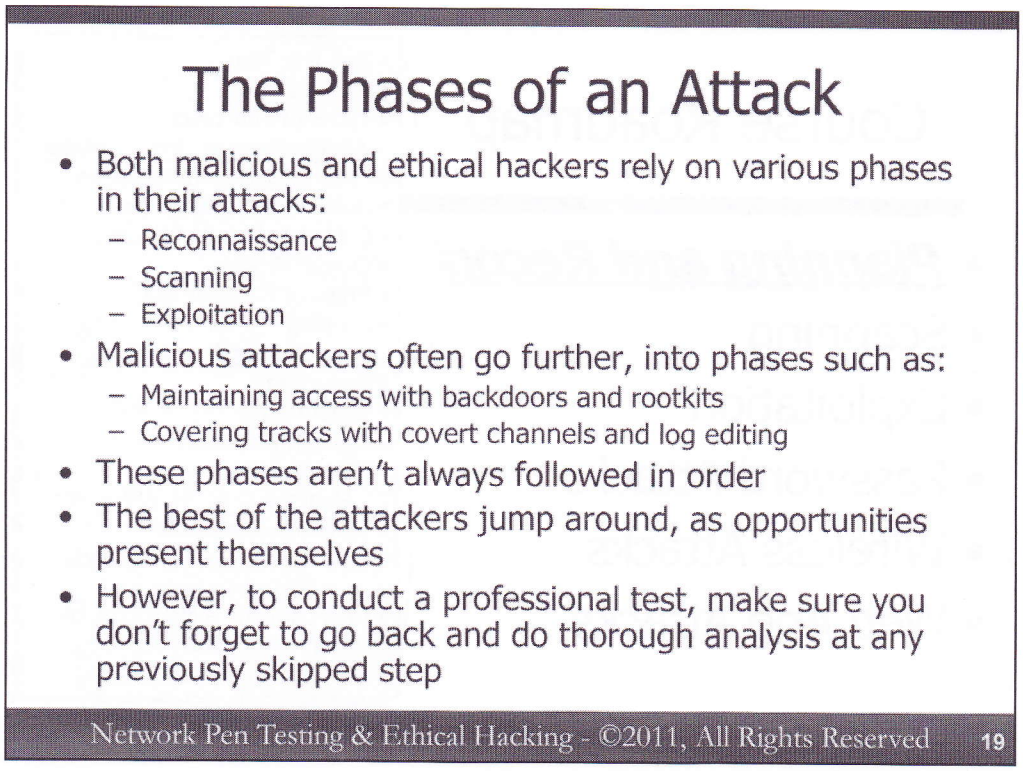

Some additional types of ethical hacking and penetration tests include:

*Physical security test:* These tests look for flaws in the physical security practices of a target organization. Testers may attempt to gain access to buildings and rooms, or to take laptops, desktops, or recycling bins out of target facilities. Dumpster diving tests are a variation of a physical security analysis. Physical testing must be conducted very carefully to ensure that the testers do not get hurt or arrested during their work.

*Stolen equipment test:* This kind of test involves obtaining a piece of equipment from the target, such as a laptop computer, and then trying to extract sensitive information from it in a laboratory environment.

*Cryptanalysis attack:* This test focuses on bypassing or breaking the encryption of data stored on a local system or across the network. Some of these tests also evaluate the strength of digital rights management (DRM) solutions. Due to legal restrictions regarding reverse engineering copyright protections (such as those imposed by the Digital Millennium Copyrights Act in the US), any contract regarding the analysis of DRM software should be inspected by a lawyer to ensure that proper permission has been derived from the owners of the given DRM solution.

*Shrink-wrapped software test:* In this kind of test, we look for security flaws in software products that can be installed in the tester's own laboratory systems. Such tests look for flaws in the software, such as exploitable buffer overflow conditions, privilege escalation flaws, and exposure of unencrypted sensitive data.




Both malicious attackers and professional penetration testers / ethical hackers apply various phases in their attacks. Attacks are often separated into these phases:

*Reconnaissance* is the process of investigating target organization to gather information about it from publicly available sources, such as domain registration services, websites, and so on. Some people include techniques such as social engineering and dumpster diving in the recon phase.

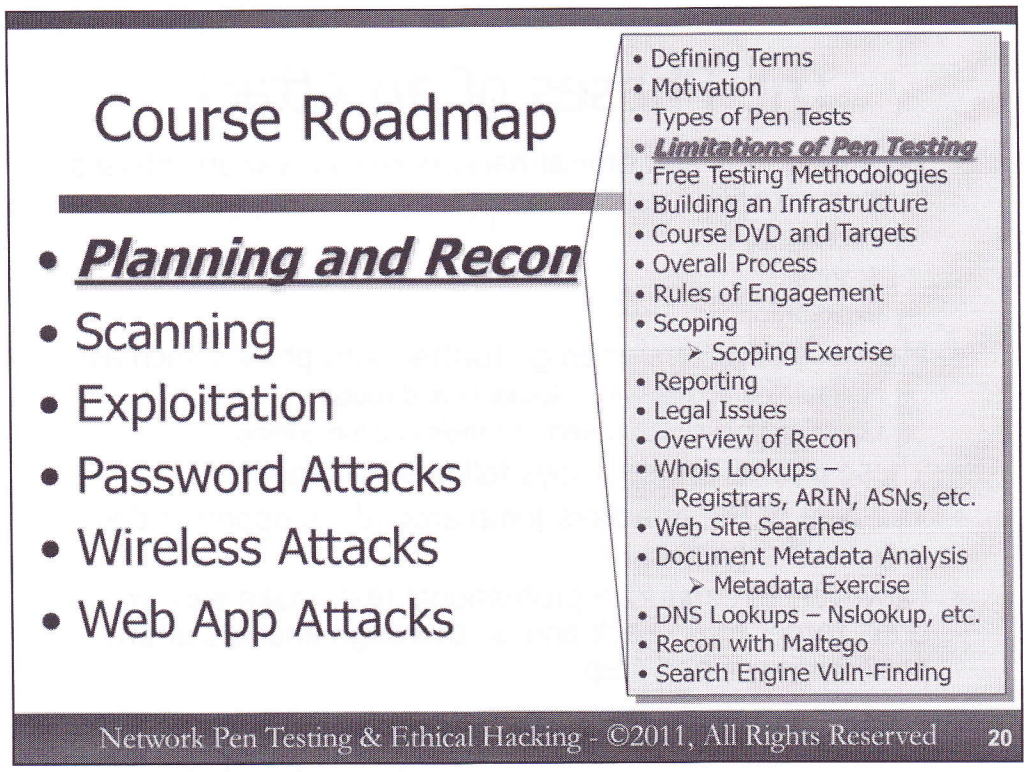
*Scanning* is the process of finding openings in the target organization, such as Internet gateways, wireless access points, available systems, listening ports, and vulnerability lists.

In the *Exploitation* phase, attackers exploit target systems to compromise them, possibly getting control of them or causing a denial of service attack.

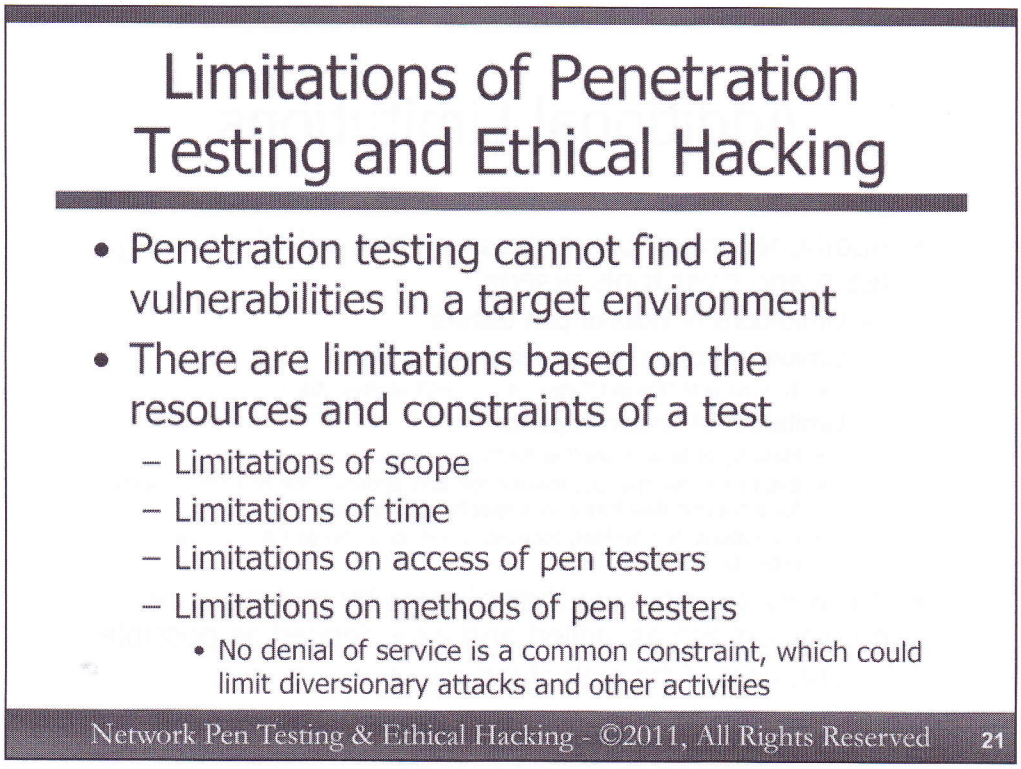
While legitimate tests often include the phases listed above, malicious attackers often go further than the rules of engagement allow for a professional penetration test. The next phase, often used by malicious attacker to maintain access and control of a target machine, involves setting up the compromised machine so the attacker can keep control over it, with techniques such as installing backdoors and planting rootkits. Malicious attackers also often use a final phase, *Covering the Tracks*, in which they employ log editing, file hiding, and covert channels to hide their activities on a system.

Please note that the best of the attackers (both the good guys and the evil ones) are pragmatists. They don't always proceed from reconnaissance to scanning to gaining access and so on. Sure, they use these steps, but they are very likely to jump around between them as events and discoveries warrant. For example, during the recon phase, attackers may discover an exploitable flaw that they will use to gain access directly, temporarily bypassing scanning. Then, once they gain access to one machine, they may go back and start scanning.

From a professional testing perspective, though, be careful when jumping out of order between these steps, making sure that you return to the earlier phases to conduct a comprehensive test.




Although penetration testing and ethical hacking can significantly help organizations find and remedy security flaws, there are some pretty noteworthy limitations of these practices. Information security pros, especially those associated with penetration testing and ethical hacking, need to be familiar with these limitations so that we can put our craft into the proper context of a larger security program. We'll review those limitations next, and consider the proper context for penetration testing and ethical hacking in an overall comprehensive security program.

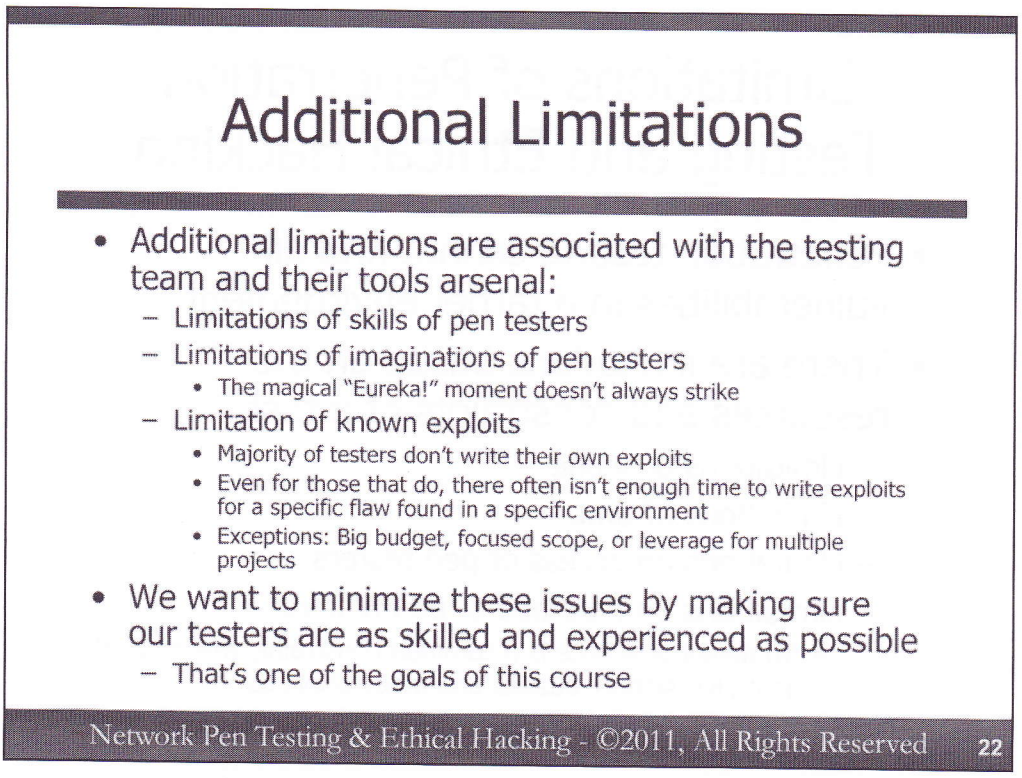



Although penetration testing and ethical hacking are useful practices, they do have some noteworthy limitations worthy of analysis. Many of these limitations are associated with the nature of testing projects themselves, with finite resources and a focused scope.

First off, testing projects by their very nature have a limited scope. Most organizations don't (and can't) test everything, due to resource constraints. We test those elements of our infrastructure that are deemed most vital. But, a real-world attacker may find flaws in other areas that simply weren't part of our testing project's scope. A related limitation is time. Professional penetration testers and ethical hackers are allocated a certain amount of project time for a test. Attackers often have far more time to work on their attack, stretching it out over months or years, when most testing processes last days, weeks, or, at most, a few months.

Furthermore, penetration testers and ethical hackers often have constrained access to the target environment that models where some, but not all, of the bad guys sit. For example, an organization may have a penetration test performed against its DMZ systems from across the Internet, modeling what attackers sitting anywhere in the world would see if they attacked through the normal Internet gateway. However, such a test won't detect vulnerabilities associated with local wireless access points, or attack vectors that could be used by malicious insiders already on the internal network.

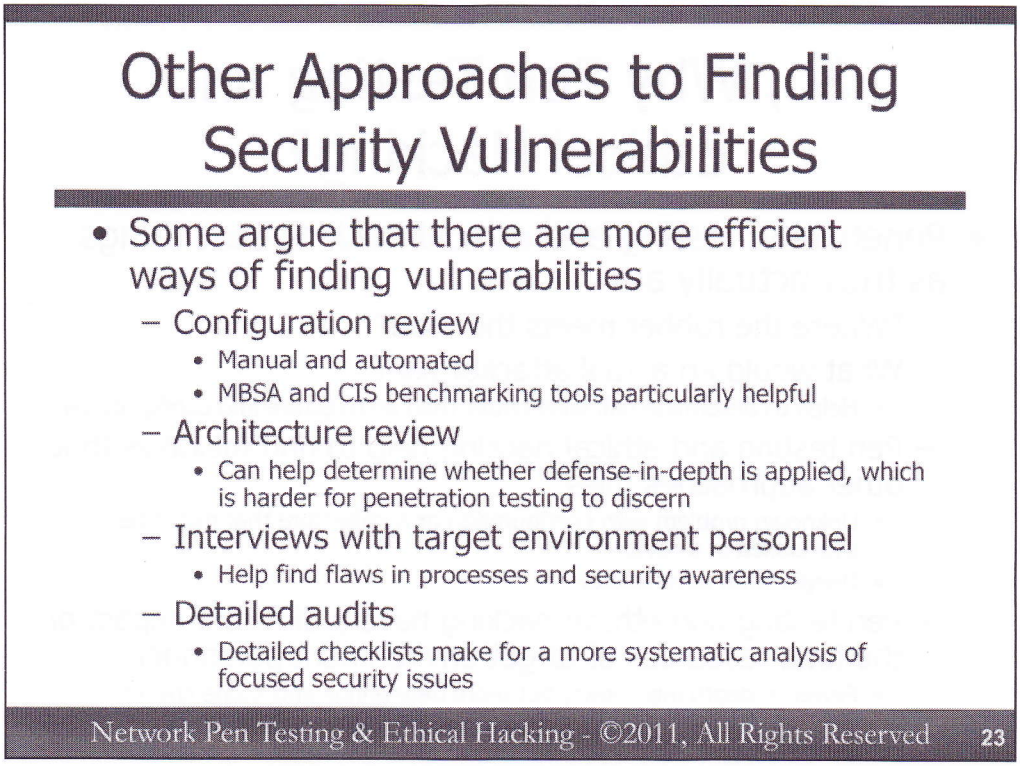
Also, because of the risk of crashing a target system during a test, some particular attack vectors will likely be off the table for a professional penetration tester or ethical hacker. For example, creating a denial of service flood to distract a system administrator from another attack vector would be an ideal tactic for a real bad guy, but will likely fall outside of the rules of engagement for most professional testers.




Beyond the limitations of project-oriented tests, we have limitations associated with the testing team itself. Professional penetration testers and ethical hackers are limited in that they have a finite skill set. Even very skilled testers have their limits, focusing on certain technologies and having less expertise in others. A malicious attacker with a different skill set might hit just the right areas of expertise to find flaws too subtle for testers with a significant but different skill set to find. Additionally, testing regimens are limited by the imagination of the testers themselves. Some attackers are incredibly creative, using vulnerabilities in manners that many penetration testers might not even consider. A major theme of this course is thinking outside of the box in a pragmatic way to bypass defenses in the target organization. But, even the most gifted of professional testers may not have the "Eureka!" moment that a malicious attacker could.

Finally, most professional penetration testing is limited by the current known exploits available publicly. Most professional penetration testers and ethical hackers do not write their own exploits, but instead rely on exploits written by others. Even for those testers who do write exploits, there often isn't enough time to create a custom exploit for a newly discovered flaw in a given target environment. The resources of a test project are finite, and creating custom exploit code could easily consume a great deal of the overall project's budget. Thus, unless the project has a particularly large budget, the client has specified a very narrow focus, or a given exploit for a flaw can be applied to several target organizations in numerous tests, custom exploit development during a penetration test is very unusual.

Of course, we strive to overcome these limitations by having a highly skilled and experienced set of penetration testers and ethical hackers. One of the major goals of this class is supporting professionals in their goal of improving their penetration testing and ethical hacking skills.



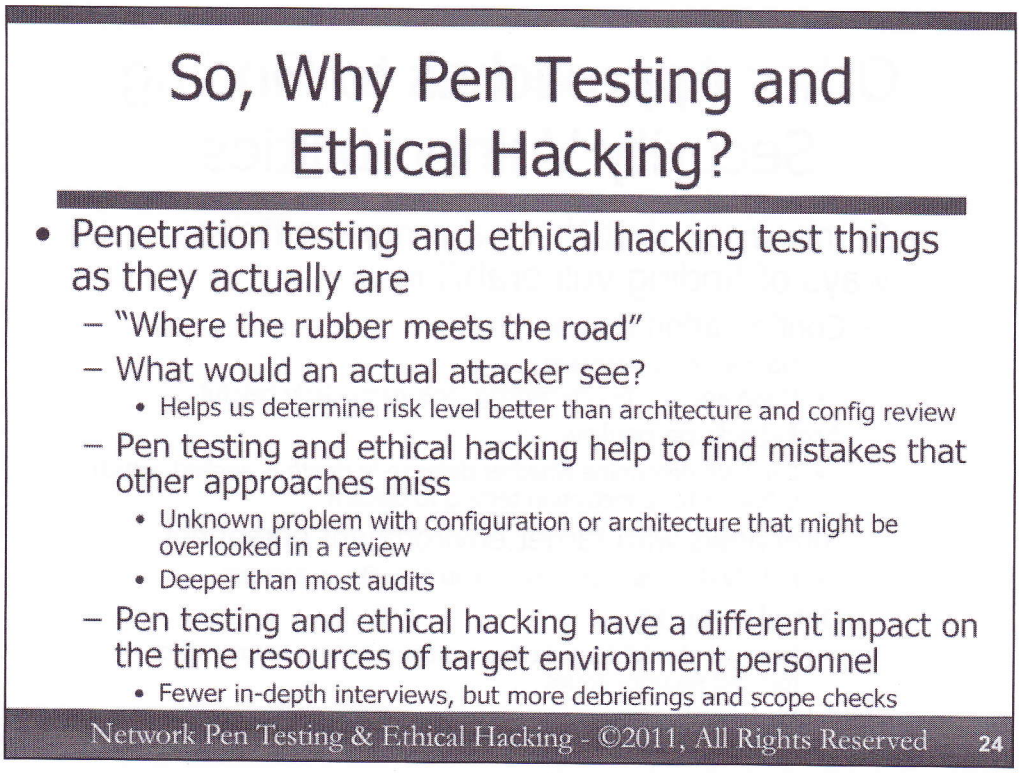

Besides penetration testing, there are other approaches for finding security vulnerabilities in a target environment. Some people argue that these other approaches are more effective (finding flaws more reliably) and/or more efficient (more subtle flaws discovered at lower costs). While debates rage about the relative merits of different approaches, it is true that there is high value in other methods of finding security flaws outside of penetration testing, including:

*Configuration reviews:* By analyzing the configuration settings of network equipment, operating systems, and applications, numerous major security flaws can be discovered. Security personnel can use admin privileges to inspect configurations manually, or to run local configuration-checking tools like Microsoft's Baseline Security Analyzer (MBSA) or the Center for Internet Security (CIS) benchmark tools for Linux, Windows, and other systems. Such tools provide a wealth of information, often finding flaws that cannot be discovered by a penetration test conducted remotely.

*Architecture reviews:* By looking at the overall design of systems and how they relate to each other on the network, significant security flaws can be discovered. For example, careful review of a network diagram might indicate that the organization has failed in its goal of deploying defense in-depth, with multiple layers of filtering between each host and the Internet. A penetration test that doesn't include an overall network topology map review might not be able to find this kind of issue.

*Interviews with target environment personnel:* By discussing security practices with the operations, security, and user-base personnel of a target environment, major security weaknesses often come to light, weaknesses that couldn't be found by a penetration tester because such issues are more associated with process and awareness than with technical specifications and configurations.

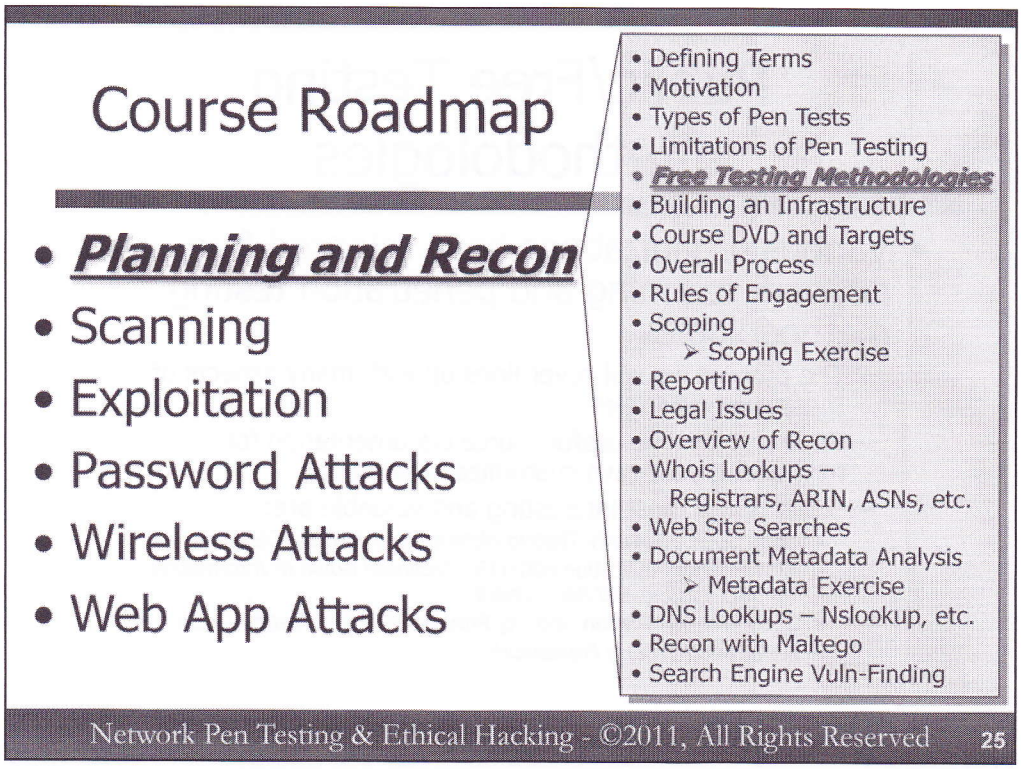
*Detailed audits:* With a carefully refined checklist, auditors also can find security flaws in an environment that would elude a penetration tester or ethical hacker, because the testers usually don't have the access to the target environment that auditors do.



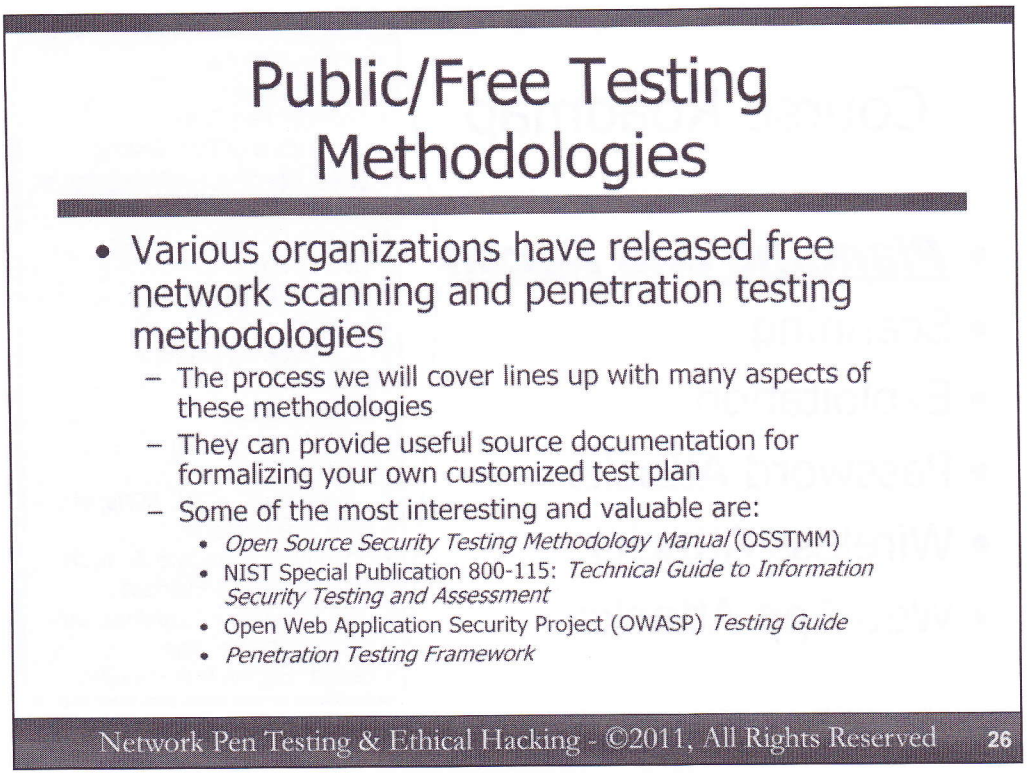

Given the limitations of penetration testing, as well as the benefits of other approaches to finding security vulnerabilities, why should an organization perform penetration tests and ethical hacking exercises? Quite simply, because they provide an excellent view of the actual security state of an environment. They highlight what a real-world bad guy might see if he or she targeted the given organization.

We get to see security in an actual operational context, not merely on paper (like an architecture review) or in discussions (like a set of interviews). We can focus on the most likely exploitable problems and see if an actual attacker could take advantage of them, getting a better feel for the actual risks we face (much more so than is possible with a configuration review). With a better feel for actual risks, management personnel can make better decisions about where to allocate security resources to fix problems. Furthermore, because the goal of many penetration tests and ethical hacking exercises is actual compromise of target machines, they often go deeper than most audits. Penetration tests and ethical hacking engagements also sometimes find subtle flaws that other methods cannot easily discern.

Also, penetration tests and ethical hacking projects have a different impact on the time resources of the target organization. Although initial scoping and periodic debriefs are required for penetration testing, such activities are usually less time consuming for target environment operations personnel than configuration reviews, architecture reviews, detailed interviews, and audits.




Several organizations and individuals have released free ethical hacking and penetration test methodologies. It is important to understand the freely available methodologies for several reasons. First, we want to recognize the people who invested their hard work in creating these methodologies and then provided them on a freely available basis to everyone. Secondly, these methodologies track nicely with the various topics we cover in this course, so reviewing them can help to shore up the topics of this class, often from a slightly different perspective. And, thirdly, as you put together your own penetration testing process, you can utilize concepts and techniques from these documents as well as this course.



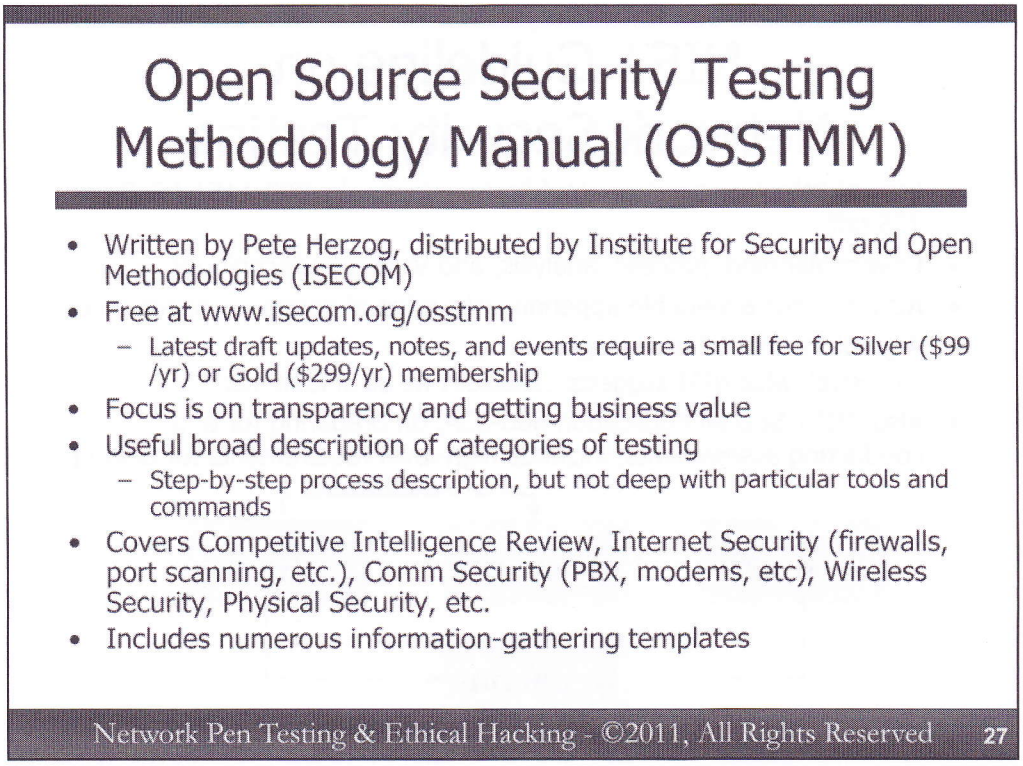

Several organizations have released high-quality, free penetration testing and ethical hacking methodology documents. The process we cover in this course addresses many topics also covered in these methodologies.

We recommend that you review each of these free documents, as they provide useful insights into testing from various different perspectives. Also, when formulating your own customized testing methodology, these document, together with this course, can act as useful sources to pull together wording for your own documentation on your testing processes and findings.

Four of the best free documents on testing methodologies include:

- *Open Source Security Testing Methodology Manual* (OSSTMM)
- NIST Special Publication 800-115: *Technical Guide to Information Security Testing and Assessment*
- Open Web Application Security Project (OWASP) *Testing Guide*
- *Penetration Testing Framework*

Let's briefly explore each one in more detail.

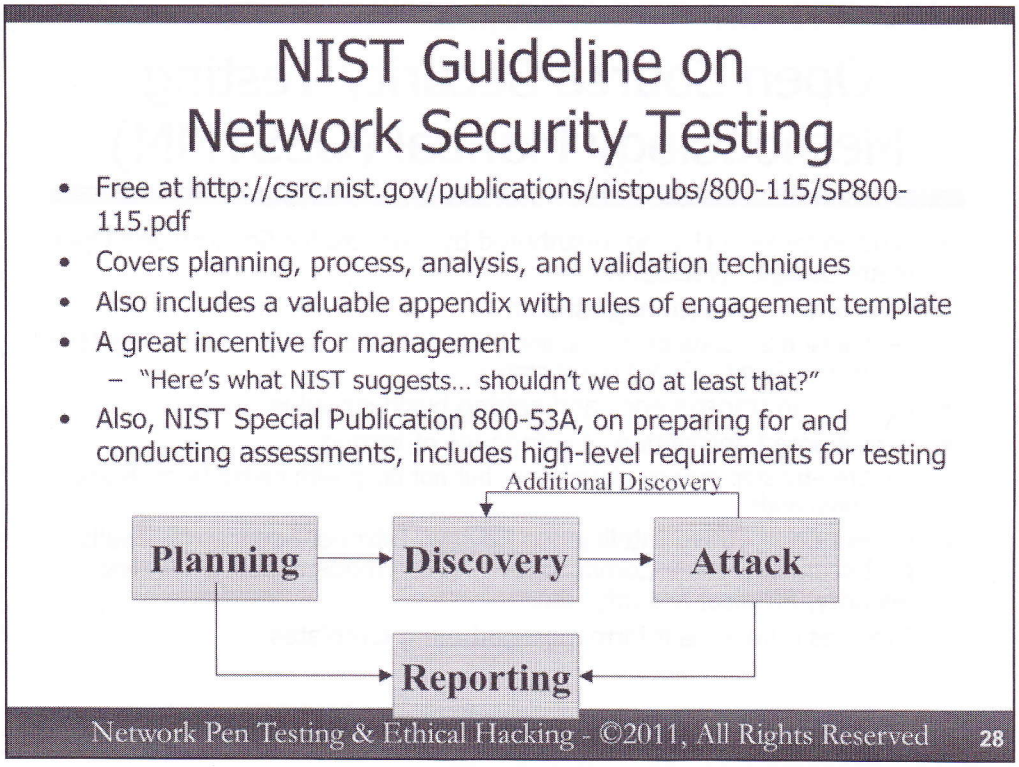



The Open Source Security Testing Methodology Manual (OSSTMM) was released by Pete Herzog and is distributed by the Institute for Security and Open Methodologies (ISECOM). This free document is focused on improving the transparency of enterprise security as well as the methodology of testers. Rather than making security testing a black art of mystery, this very comprehensive document eloquently strives for repeatability, consistency, and high quality in numerous kinds of security tests. The document is written so that organizations and the testers they employ get the maximum business value for their activities. Earlier versions of the OSSTMM are available for free. The latest version and drafts of new updates are available to Silver subscribing members. Gold subscribers get those items as well, plus additional research, mailing lists, and contacts for business questions.

The overall document is very broad, covering numerous kinds of security tests. It does not get into depth with particular commands and tools, but is still immensely useful.

Topics addressed in the OSSTMM include competitive intelligence review (performing reconnaissance against the target enterprise), Internet security analysis (finding open ports and vulnerabilities in Internet-accessible systems), and communications security (addressing vulnerabilities commonly found in PBXs, modems, and fax machines). The process also includes modules for reviewing wireless security (particularly Wireless LANs) and physical security.

One of the best aspects of the OSSTMM is its detailed discussion of scoping a project in advance, as well the report templates that it includes. It has fill-in-the-blank templates for almost every kind of test it describes.

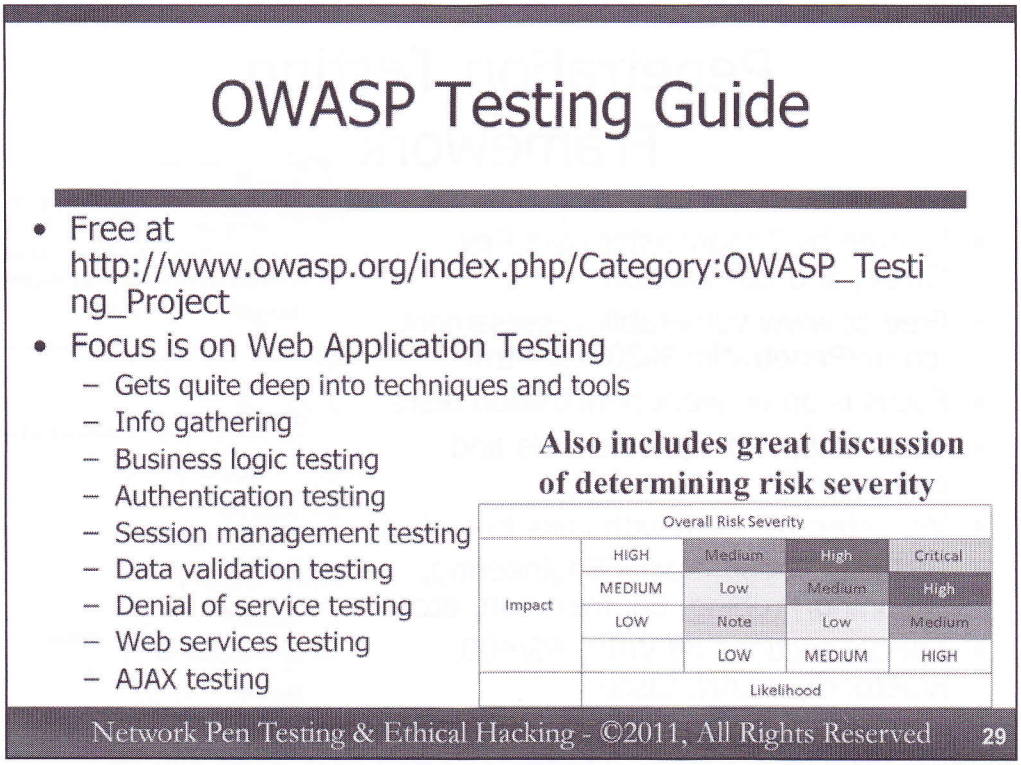



The United States National Institute of Standards and Technology (NIST) has released a document called *Technical Guide to Information Security Testing and Assessment* that covers network penetration testing methodologies at a high level. The document addresses the process commonly applied in testing, planning for tests, conducting detailed analysis, and dealing with validation of discovered issues. It also includes appendices that cover some common tools found in Backtrack.

Another very useful appendix is a template for rules of engagement, helping testers and target system personnel agree upon various vital aspects of how the testing will be conducted. We'll spend some time later in this book discussing rules of engagement.

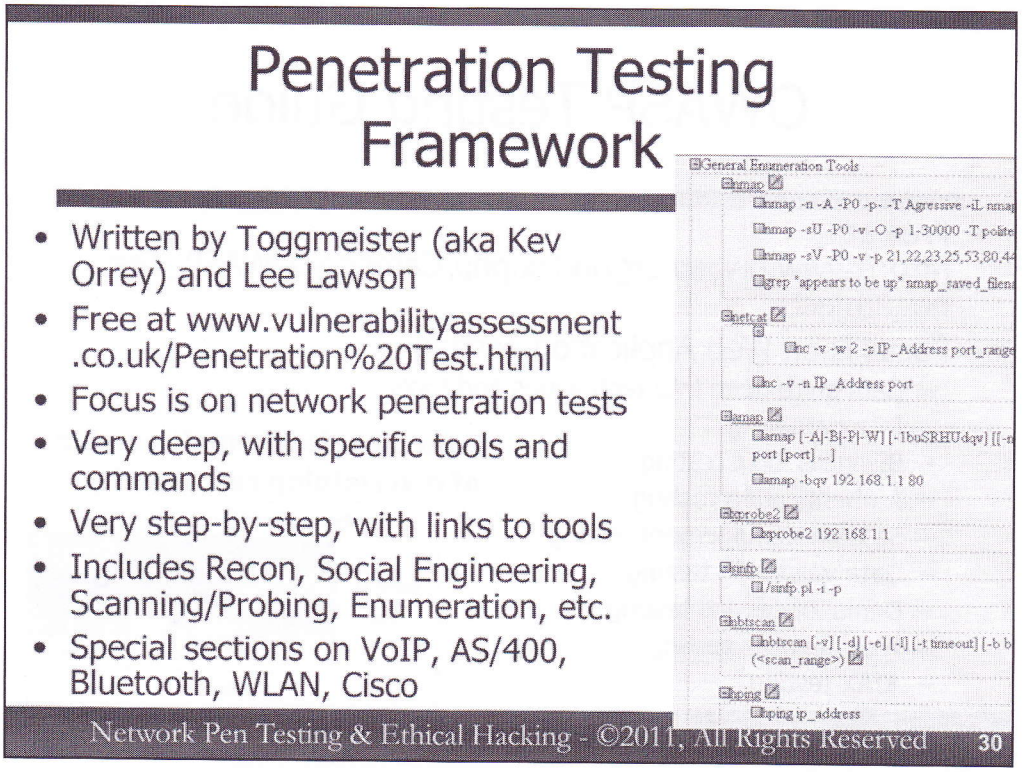
One of the most useful aspects of the NIST guide is the motivation it can help us inspire in management. If management suggests that our testing methodology shouldn't include some vital component that NIST recommends, we can ask our management why they want to deviate from NIST's guidance. Management may then provide business rationale for doing so, or decide that complying with the NIST document is a better practice than they originally anticipated. Either way, we get a better test more attuned to the business needs of our enterprises.

Another document from NIST also addresses measuring security in an organization. The *Guide for Assessing the Security Controls in Federal Information Systems,* Special Publication 800-53A, is more high-level than SP 800-115, but still provides some useful tips for planning assessments.




Next we get to the Open Web Application Security Project (OWASP) Testing Guide. Unlike the other broad and general purpose methodologies we've touched on so far, the OWASP guide focuses purely on web application security testing. From a web app perspective, this document is an excellent description of the various kinds of testing that need to be done, providing great depth and a wide variety of tools to use in the process.

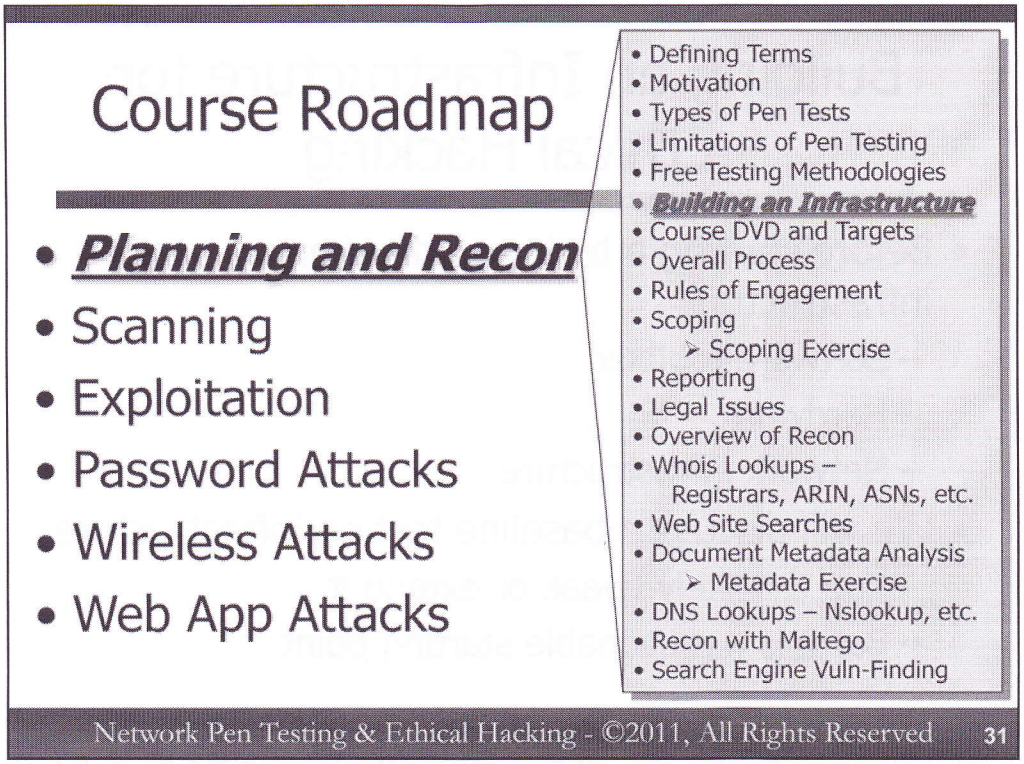
One of the best aspects of the OWASP guide is its detailed description of determining the business risk posed by findings. The OWASP guide rates risk based on the impact it could have to the business, and the likelihood it has of occurring. From those two aspects, the overall risk rating of a given finding is derived, giving the enterprise appropriate guidance on prioritizing their findings.




And, finally, we have the deepest of the free testing methodologies that we're covering: The Penetration Testing Framework by Toggmeister and Lee Lawson. This web site provides a step-by-step walk through of every aspect of a network penetration test, including very specific tools (with links to each and every tool) and the individual commands to use for each tool.

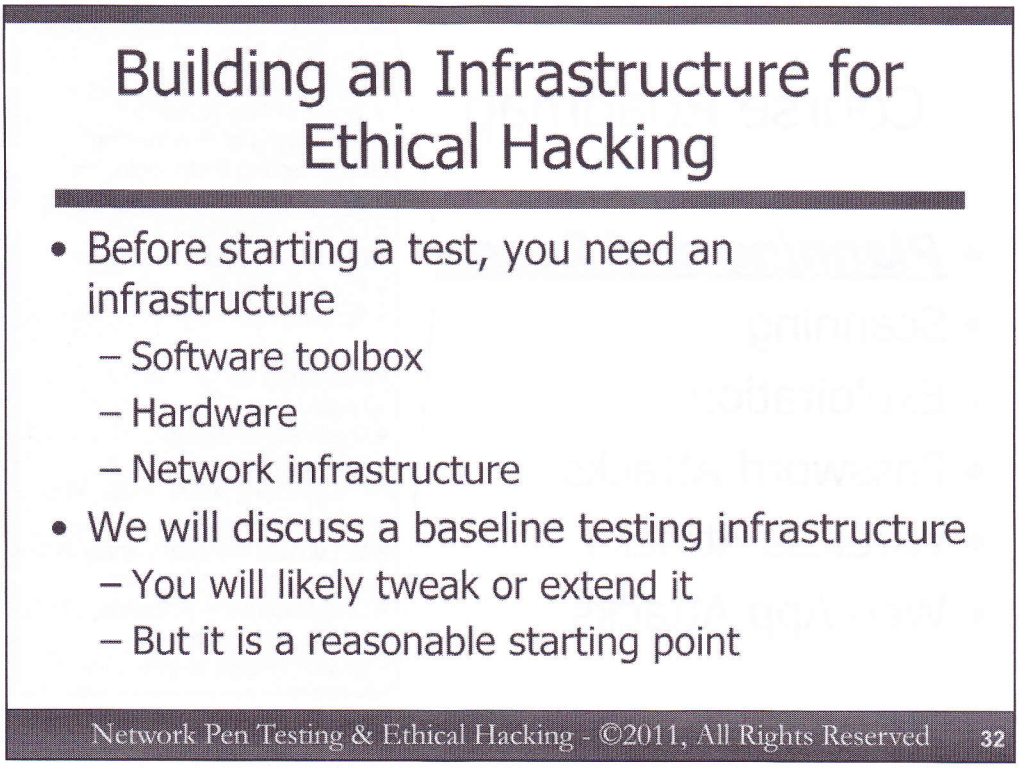
The document walks its reader through several concepts, step by step, covering reconnaissance, social engineering (via e-mail and the phone), scanning, enumeration of target systems, exploitation, configuration review, and more.

Several sections focus on specific technology, such as Voice over IP (VoIP), assessing the security of AS/400 machines, Bluetooth security analyses, and wireless LAN assessments. The section on analyzing Cisco routers and related devices is also quite helpful.

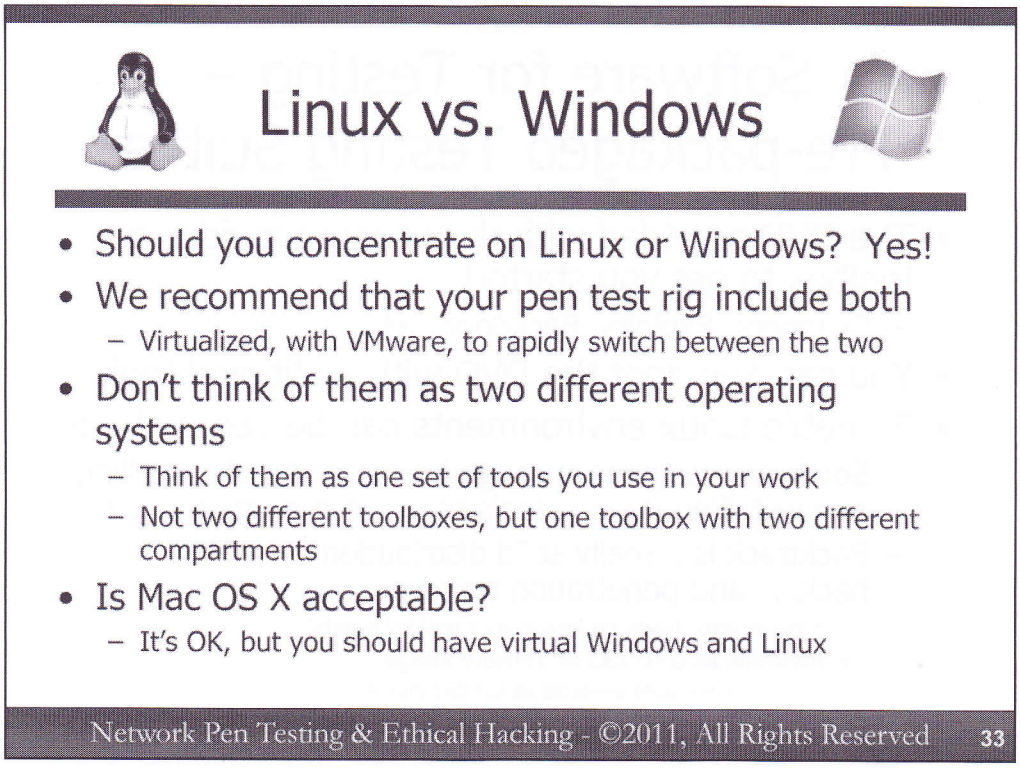



A well-stocked lab and an arsenal of testing tools are crucial to the success of an ethical hacker and penetration tester. Let's now discuss the hardware, software, and network connectivity used by testers in their work.

Keep in mind that these infrastructure items we'll discuss are not a one-size-fits-all proposition. Instead, we'll cover the areas of tools that you'll need, with some notable examples. Then, based on your budget, expertise, and test types, you can construct an appropriate arsenal to match your test regimen.



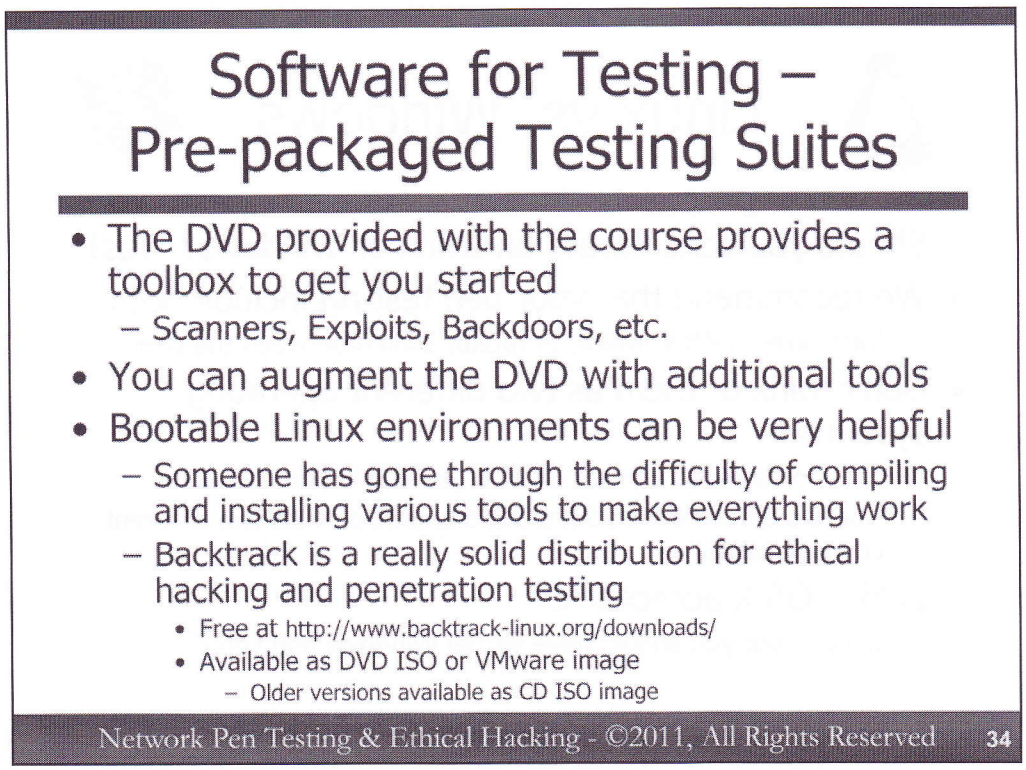

To conduct a thorough test, ethical hackers and penetration testers first have to establish an infrastructure from which to do their work. Detailed planning in advance is essential, pulling together the proper software, hardware, and network infrastructure. We will discuss some tips for doing this properly, but keep in mind that the technical infrastructure necessary for performing these tests is not "one-size-fits-all". Consider the items we will discuss to be a baseline infrastructure, which you can tweak or extend to meet your own specific testing needs. We will discuss a software toolbox, the hardware, and network infrastructure you should consider when doing tests.




A common question among penetration testers and ethical hackers is: "Should I focus my skills and toolbox on Linux or Windows?" When confronted with this question, we recommend that your pen test toolset include both operating systems, side-by-side, working together to maximize your efficiency and capabilities. The truth is, some tools work better on Linux while others work better on Windows. Some tools work just fine on both, while other tools have only been released for one of those platforms. Thus, if you choose to work in only one OS, or at least just focus on that OS, you'll be missing out on a lot of very useful tools and techniques. To improve productivity and streamline workflow, we recommend virtualizing one of these two OSs, perhaps using VMware, and running the two simultaneously on the same hardware so you can quickly switch between them.

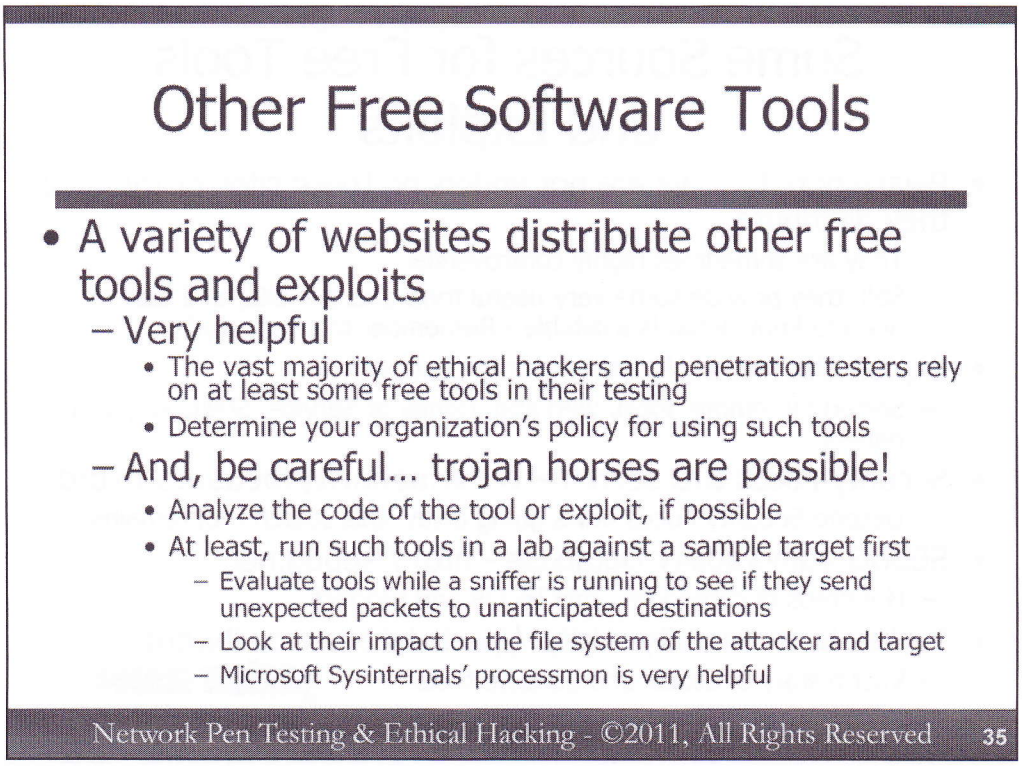
In fact, the whole question posed at the start of the last paragraph illustrates a mindset that should be transcended. Don't think of them as two different operating systems. Think of them as one set of tools that you use in your penetration testing and ethical hacking job. As a carpenter or plumber would use the best tool available and convenient for a given job, so should you. To continue with that analogy, don't think of Windows and Linux as two different toolboxes. Instead, they are two different compartments in your single toolbox.

Some of you are no doubt wondering whether Mac OS X is an acceptable platform for penetration testing and ethical hacking. It is, with remarkable stability and ease of use. However, there are some tools for Linux and Windows that will simply not run on Mac OS X, no matter how hard you try to get them installed. Thus, if you plan to use Mac OS X, make sure you get a virtualization solution for it (such as VMware Fusion or Parallels) so that you can also run both Windows and Linux on top of Mac OS X.




First off, you'll need software for your testing regimen. The DVD included with this course is full of tools used in ethical hacking and penetration testing. Furthermore, the VMware image on the DVD includes tools pre-installed, and in many cases, pre-configured so that you could apply them directly in your own testing.

Another useful source of tools are the bootable Linux distributions various people have made freely available, loaded with useful assessment and attack tools. A really solid set of tools is included in the Backtrack DVD, freely available at the URL on this page. The latest version of the Backtrack distribution is a DVD ISO image (suitable for burning to a DVD) or a VMware image. Older versions of Backtrack are available as CD ISO images, which will fit onto a CD instead of a DVD. Numerous similar CDs are also available, but Backtrack is one of the best because of its comprehensive set of tools, compatibility with a wide range of hardware, and carefully designed organization and lay-out.

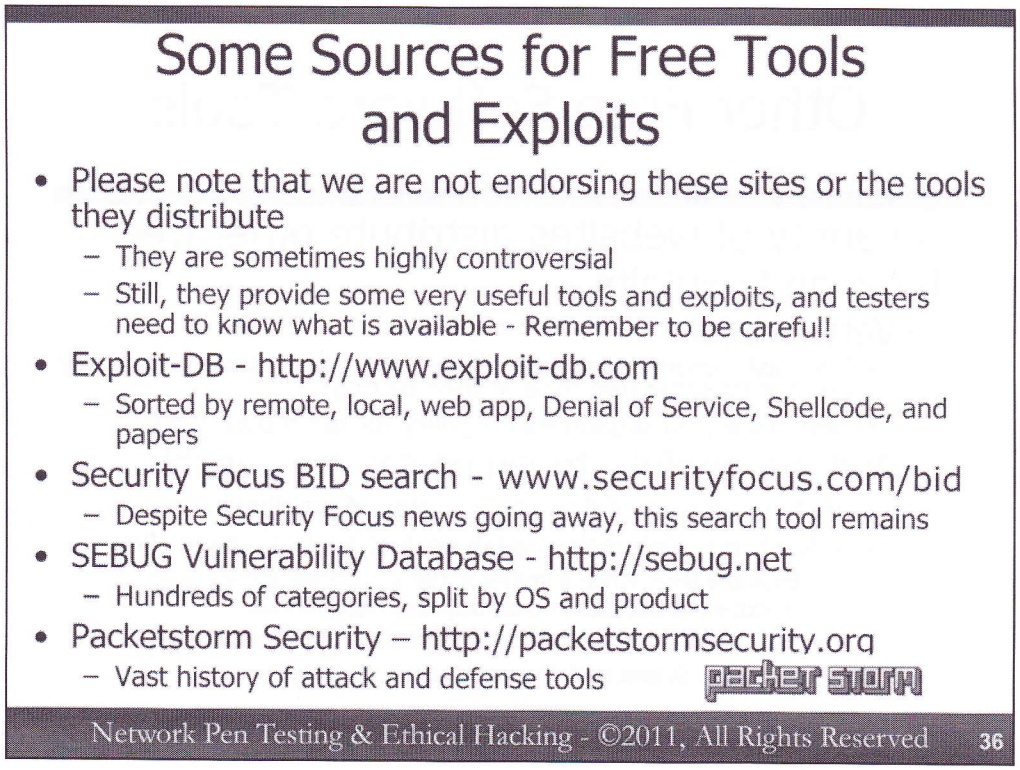



In addition to the course DVD and free bootable Linux environments, a variety of websites offer vast arsenals of free tools and exploits, which can be incredibly helpful. The vast majority of professional penetration testers and ethical hackers rely on at least some of these free tools when doing their jobs. Before considering whether you can run such tools in your environment, you need to determine your organizations' policy regarding the use of such third-party security assessment and exploitation tools. Some organizations strictly forbid the running of any tools beyond a standard baseline of already-approved tools. Others allow additional tools to be used, but only if they are carefully vetted.

Consider this scenario: a tester scans a target environment, discovering a listening service which has a version number that is known to be exploitable. With a little research, the tester discovers a freely downloadable exploit for that specific version of the service from an exploit distribution site. Suppose further that the rules of engagement for the test allow actual exploitation of the target machine, and, furthermore, the tester's own organization allows for the use of third-party, free exploits. What should the tester do?

We strongly urge you to be careful with free, downloaded tools and exploit code. Historically, some of the tools and exploit code freely distributed on the Internet have included backdoors that let a bad guy control any system on which the tool was run, or even control the target machine the tool was run against. Also, some of the tools may cause a crash in a target service or system.

Thus, we recommend that testers analyze all free tools carefully in a laboratory before using them in a test. If you have the skills, review the source code for the tool before using it, making sure it does exactly what it says it does, with no hidden backdoor functionality or other trojan horse capabilities in the tool. If you cannot review the code, then, at a minimum, run the free tool in a laboratory environment, carefully reviewing the traffic it sends across the network (looking for unexpected packets going to unexpected destinations) and any changes it makes in the file system of the attack and target machines. The free Microsoft Sysinternals processmon tool is very helpful in analyzing file system and registry interactions. Processmon has subsumed the earlier tools, filemon and regmon, extending their functionality in a single tool.




While there are numerous exploit and attack tool repositories on the Internet, some of the most comprehensive archives that are updated on a regular basis include "The Exploit Database", Inj3ct0r, and Packetstorm Security. Several other sites come and go on a regular basis, but these sites are long-standing and tend to have relatively higher quality tools.

The Exploit Database (exploit-db for short) is maintained by the same group that maintains the Backtrack Linux distribution. Their site hosts well over 10,000 exploits, and sorts them into useful categories such as Remote Exploits, Local Exploits, Web Applications, Denial of Service / Proof of Concept, Shellcode, and papers. For each exploit in these categories, they list the platform (Windows, Linux, PHP, etc.) and the author.

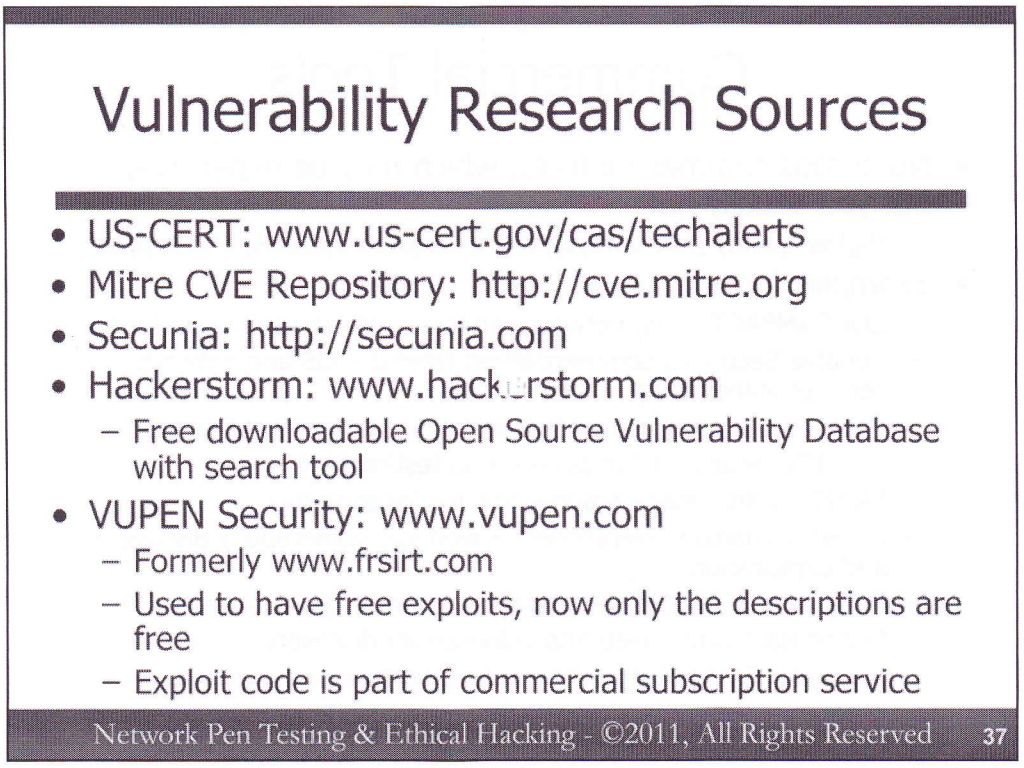
Inj3ct0r sorts its exploits by different operating systems (ranging from AIX to Windows, with almost everything in between). Additionally, they are sorted chronologically, based on the release date of the exploit code. Another page at Inj3ct0r breaks exploits down into local (which tend to let an attacker with limited privileges jump to higher privileges such as root, administrator, or SYSTEM in a local privilege escalation attack) versus remote exploits.

The Security Focus BID website also has information about various vulnerabilities along with exploits for some of them, available at www.securityfocus.com/bid. The older Security Focus news site was shut down, but the very useful BID search is still available.

Also, the SEBUG site has hundreds of categories of vulnerabilities, including exploit code for many different issues that they inventory.

Packetstorm Security has an archive of attack and defense tools that spans over a decade. It's really quite an impressive assortment of useful tools, exploits, and security research papers.

Please note that we are not endorsing these sites or the tools that they distribute. These sites have been quite controversial, and you need to be careful with any code you download from them. Still, ethical hackers and penetration testers need to know about these sites to do their jobs.




Beyond the tool and exploit sites, there are also numerous vulnerability research sites available. While these sites do not distribute exploit code freely, they do publish information about vulnerabilities. These detailed vulnerability descriptions are invaluable in letting a tester know that there is an issue with a system type or service version discovered in a test. Even though an exploit might not be available (in fact, an exploit may have never been publicly released or even created at all), the tester still needs to understand the vulnerabilities so that they can be included in the test report.

Some of the best sites with vulnerability research and detailed descriptions are the following sites:

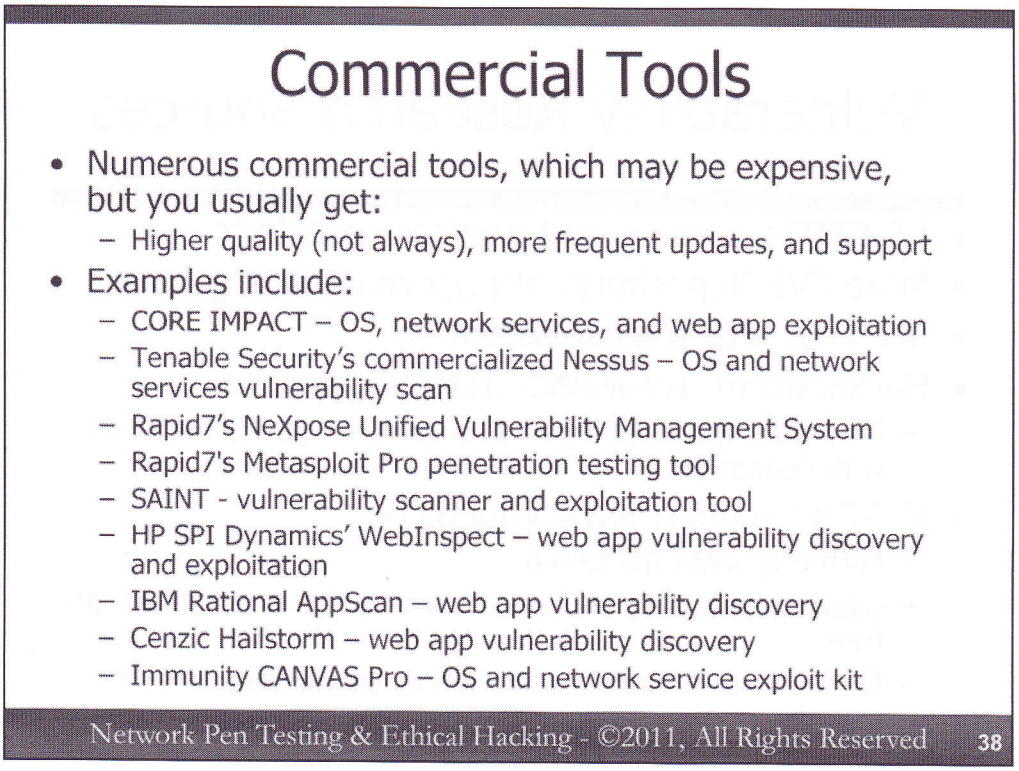
The United States Computer Emergency Readiness Team (US-CERT), maintained by the US Department of Homeland Security (DHS)

The Common Vulnerabilities and Exposures (CVE) repository operated by Mitre

The Secunia vulnerability list

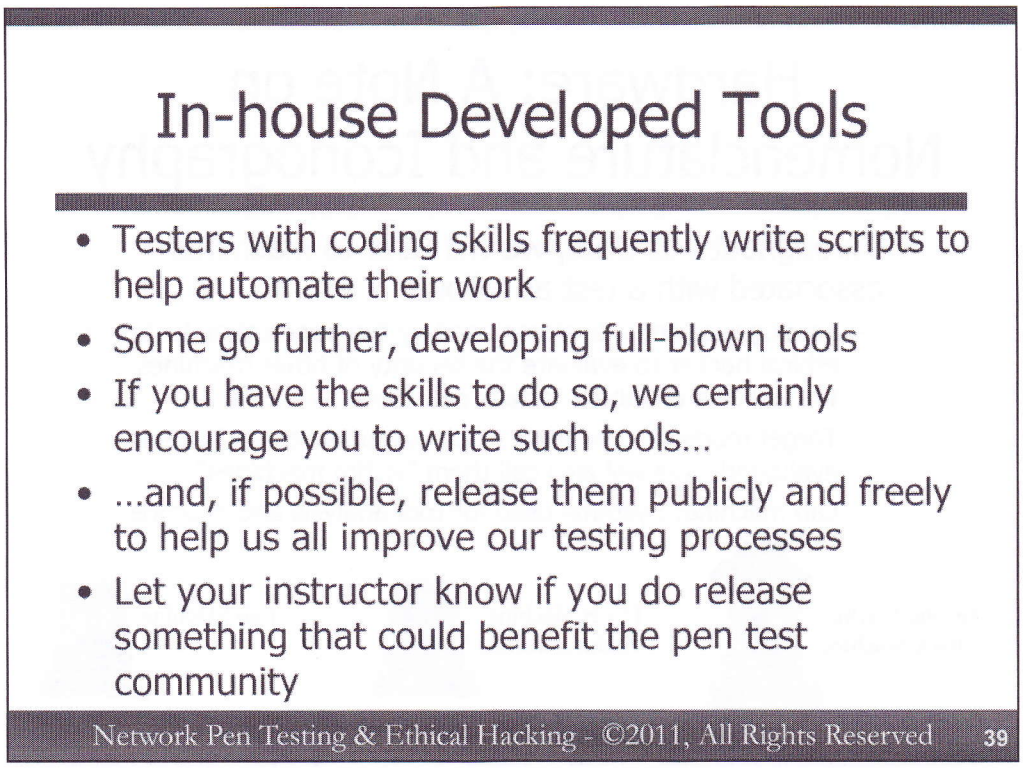
The Hackerstorm website, which includes a free, downloadable Open Source Vulnerability Database tool, that can be stored locally by a tester for searching even without Internet access

The VUPEN Security group, which used to distribute exploits on a free basis, but now publishes vulnerability details for free and offers exploits only to subscribing customers. This site used to be called the French Security Incident Response Team (FrSIRT), although it is in no way related to the French government.




In addition to the free tools we've been discussing, some penetration testers and ethical hackers rely on commercial tools for testing. There are a large number of commercial tools, with new ones released on a regular basis. The advantages of commercial tools include generally higher quality (but not always), typically more frequent updates (given the vendors' paid teams of software developers), and technical support if issues arise during testing.

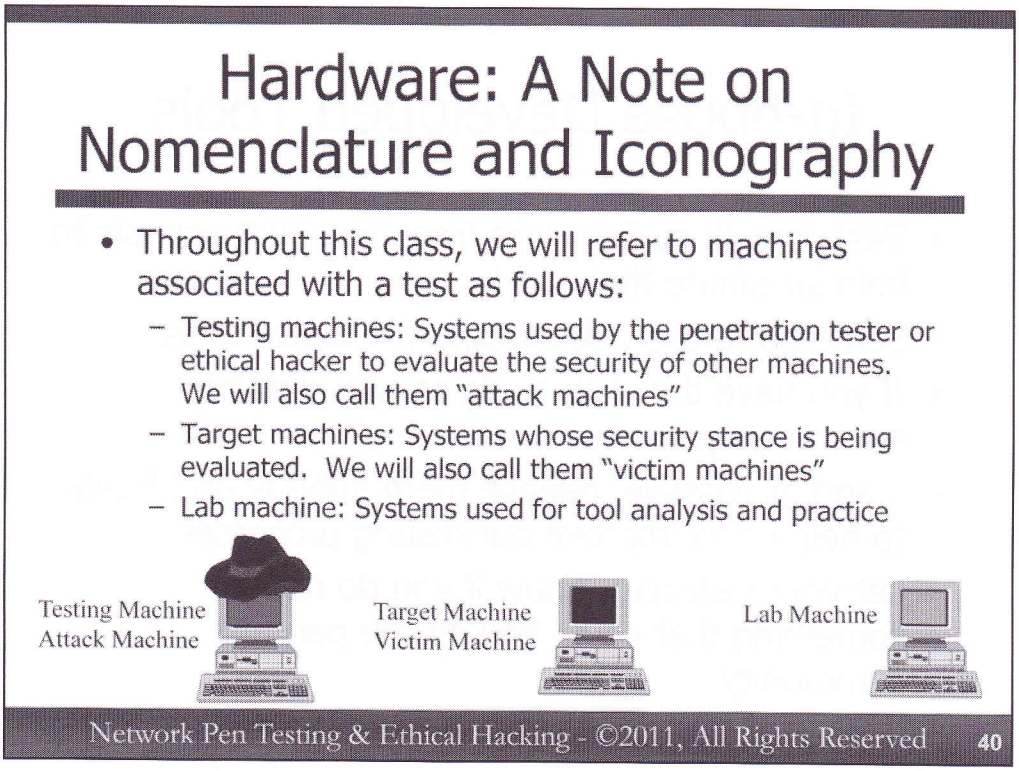
Although this course is taught from a vendor-neutral perspective, professional penetration testers and ethical hackers do need to know about some of the commercially available tools, even if they don't use them. That way, they can make sure that their test regimen made up of non-commercial (free or in-house) tools includes similar concepts and capabilities of the commercial tools. This slide lists a few of the more popular and comprehensive tools for testing.




Some penetration testers and ethical hackers with coding skills develop scripts that automate some portions of their test to help improve efficiencies. Some go even further, writing full-blown tools of their own that improve upon publicly available tools or conduct very specific tests tuned to their target environment.

If you have the development skills to do so, we encourage you to create custom scripts and tools to help in your testing. If you do create a high-quality tool that would benefit others, we encourage you to release it publicly.

If you do release a tool, please let your instructor for this class know about it. It could help us all improve our testing processes, and may be added to a future version of this course.

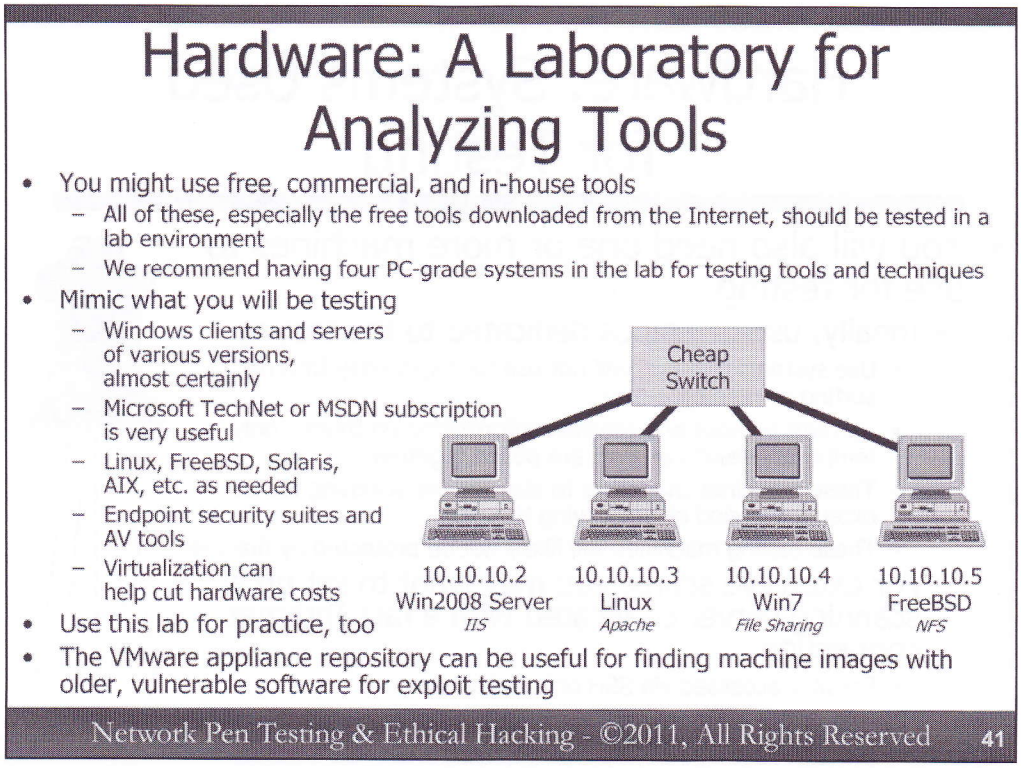



For the remainder of this class, we will need to carefully differentiate between the machines used by the penetration tester or ethical hacker and the machines whose security is being evaluated.

We will use the terminology *testing machines* and *attack machines* to refer to the systems that the tester is using to evaluate the security of other systems. These testing machines often run the attacker's scanning and exploitation tools. In figures, these machines will be represented with a red screen and black hat. We use these pictorial clues to help you rapidly identify where the attacker's machines are in a diagram. However, please do not think that the black hat on this computer implies that the attacker is somehow evil. The attackers we are referring to here are professional ethical hackers and penetration testers. The black hat just makes this system easier to quickly locate in the figure.

The machines whose security is being evaluated are referred to as *target machines*, and, occasionally, as *victim machines*. They are represented pictorially as a standard machine, with no hat or colored screen.

*Lab systems* will often have a light blue screen. These machines are used to evaluate tools and practice our techniques. We will refer to them as laboratory attack machine or laboratory target machines when the distinction is warranted.



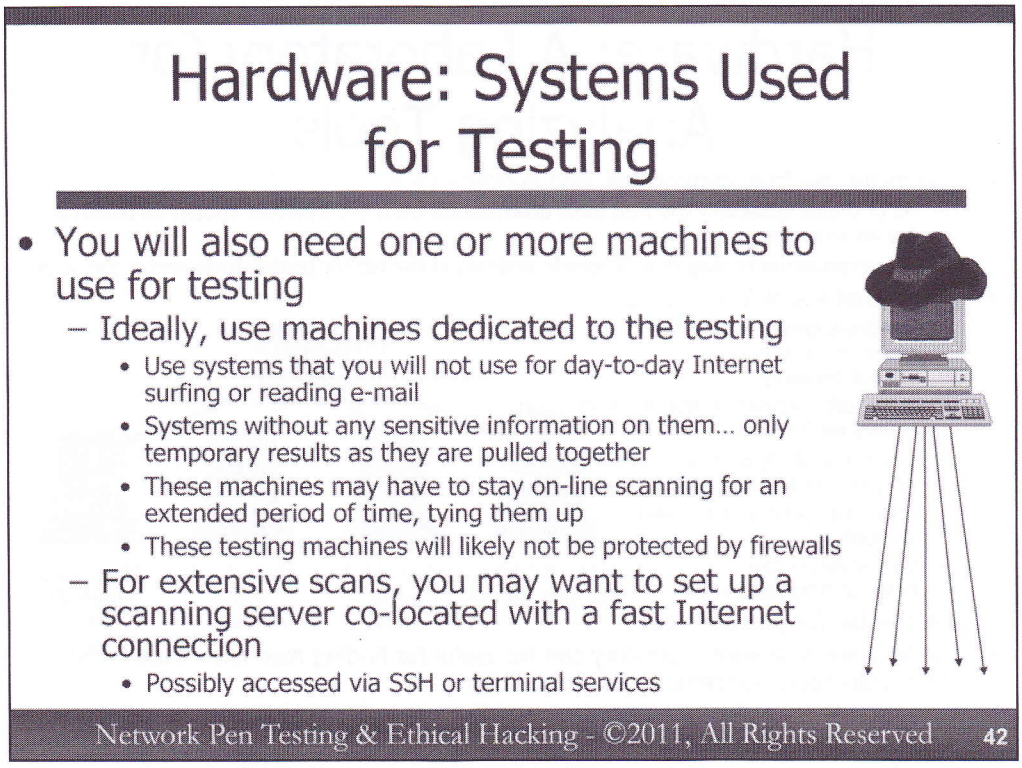

As we've seen, your tests might rely on free, commercial, and/or in-house tools, depending on the policies of your organization. Whichever tools you use, you should test them in a laboratory environment to make sure you understand how they work and their potential impact on a target machine. Such laboratory testing and analysis is especially vital for free tools downloaded from the Internet because of concerns about quality, the potential to crash a target, and hidden functionality that could compromise the test systems.

In creating a test lab, we recommend that you get at least four inexpensive PC-class systems. That way, you can have a variety of different kinds of operating systems and software applications to analyze. Since the speed of these machines isn't vitally important, inexpensive two- or three-year-old PCs can be used. If you want a more portable environment and can afford higher performance computers, you also may want to consider using virtual machines running in VMware or other virtualization products for your test-bed.

Your test-bed should include operating systems that mimic what you will encounter in your actual tests, likely at least a variety of Windows clients and servers. A Microsoft Technet or MSDN subscription really comes in handy, as it provides access to many Microsoft products for laboratory testing purposes (not production use). Additionally, include Linux, FreeBSD, and other operating systems that make sense for what you will test. You should also have access to the most popular end point security suites and anti-virus tools (especially those used in your target environments) to evaluate how your attacks will function against them and to tweak your approach in light of the target's defenses.

A well-equipped lab will also be useful for practice hacking to help improve your skills over time.

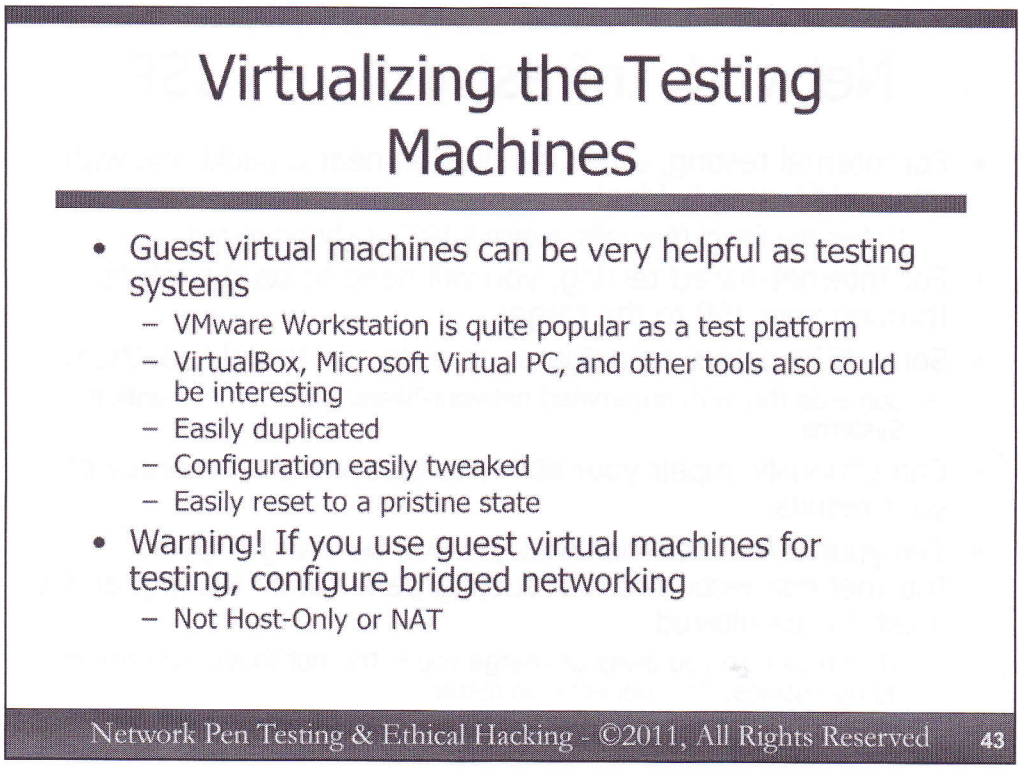
Another useful source of vulnerable applications pre-installed on their own operating systems is VMware's virtual appliance repository, at www.vmware.com/appliances. They have a vast archive of virtual machines freely available for download (typically focused on freely redistributable operating systems, such as Linux and FreeBSD). Many of the older images include applications that are known to have vulnerabilities, which you can test and evaluate in the lab.




Next, you'll need the actual systems that will be doing the testing. We recommend that you use one or more machines that are dedicated solely to testing. That is, ideally, your testing machines should not be used for routine web surfing and e-mail reading, or even vulnerability research during a test. Additionally, these machines used for testing should not be storing any sensitive information, other than the temporary tools output as the test is running. In other words, these machines should be focused exclusively on testing.

There are several reasons for this exclusivity. First off, these machines may have to stay online for a significant period of time, as you wait for a lengthy scan to complete. Secondly, these machines used for testing are often not firewalled, either by a personal firewall or a network firewall. We'll address this firewall issue in more detail shortly. And, thirdly, any additional software that you run on the testing machine could slow down the progress of your test or otherwise impact the results.

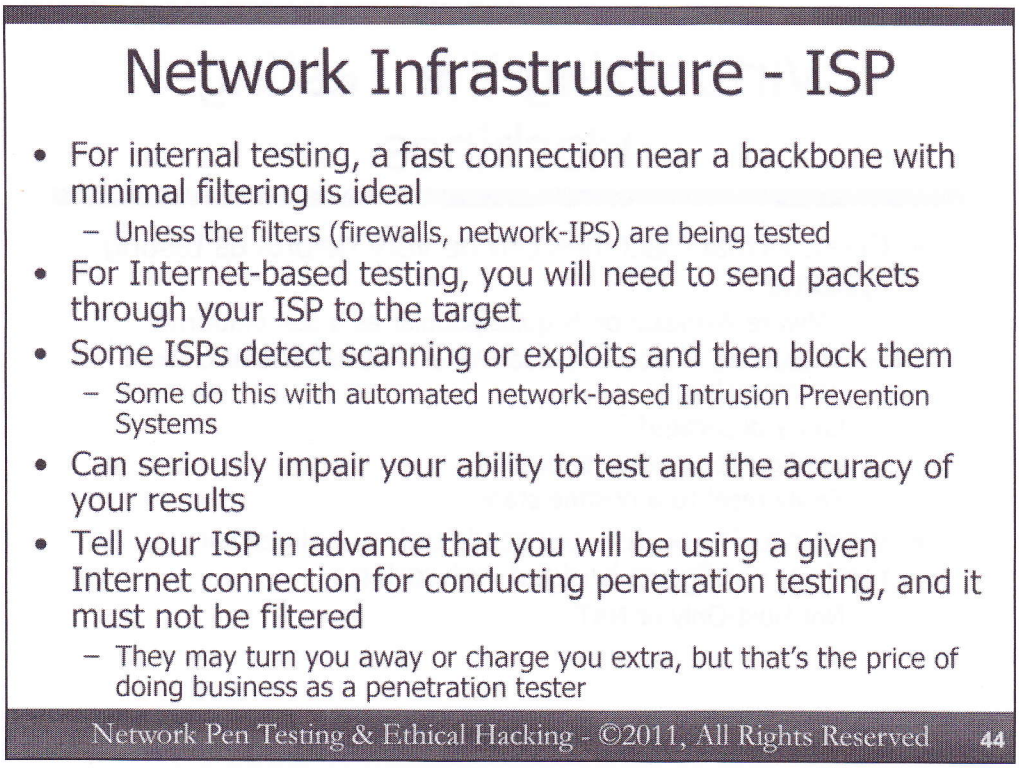
For extensive, long-term scans that might run for several days, you may want to consider buying a server-grade machine that is housed in a secure location, either at your own facilities or in a locked cage at a co-location facility with high bandwidth. You could then use Secure Shell or terminal services to control testing software on that machine which is dedicated to testing.




Many penetration testers and ethical hackers use guest virtual machines as their testing systems. Most rely on VMware Workstation because of its rich set of features and relatively low cost. Some use VirtualBox, a free virtualization platform. Others rely on Microsoft's virtualization products, although they are less common in penetration testing than VMware's products. This difference is due to the early market lead VMware established with a lot of useful features. Microsoft has been playing catch-up in virtualization, and may one day surpass VMware, but the jury is still out.

Virtual machines are useful for penetration testers for many reasons. First, they are easily duplicated. A tester can simply replicate the disk image of the VMware guest and have another identical testing guest machine ready to run. Also, the tester can easily modify the configuration of a guest machine, altering network settings, the amount of RAM, disk image sizes, and so forth. And, with VMware's "Revert to Snapshot" feature, a guest machine can be quickly reset to a pristine state, in case a tool causes problems with the machine.

There is one very important note about using guest virtual machines for testing – configure them to use Bridged Networking. Many VM environments offer other networking alternatives that will get in the way of your testing. In addition to Bridged Networking, VMware also supports Host-Only and NAT Networking. Bridged Networking makes the guest look like it is on the same subnet as the host machine, a desirable property. As a tester, we want our packets to get out from our testing system to the target with as little interruption and alteration as possible. Having the guest connected via a virtual bridge has very little impact on the packets. NAT Networking, on the other hand, will perform Network Address Translation on the packets, altering them and potentially dropping them if the NAT table fills up. Finally, Host-Only Networking just doesn't make sense in an across-the-network test, as the guest can only can only reach the host and no other systems. Thus, when using guest VMs as the testing systems, use Bridged Networking or you will miss things in your findings!




For internal testing, the testers should have a fast network connection near a network backbone with minimal filtering between it and the target systems, unless the filtering devices themselves, such as firewalls and network-based Intrusion Prevention Systems (IPSs), are being tested.

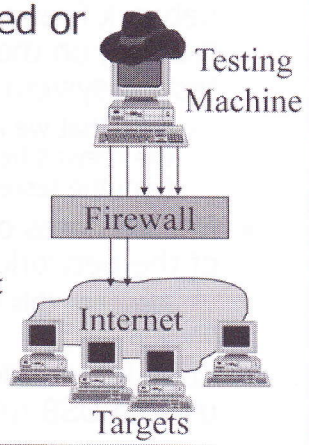
For testing across the Internet, the testers will of course need an Internet connection from an ISP. We recommend that they get a stable, relatively high-speed connection, such as a T1 or better. Cable modems, DSL, and FIOS lines are a also possibility, but they are often less reliable than a T1. Regardless of the connection type, the testers will be sending some unusual traffic across that line. Their scanning will generate a large number of unusual packets. Furthermore, they may run actual exploits against targets across that ISP. This could be a problem, unless it is cleared with the ISP first.

Some ISPs detect network scans and throttle them, slowing down a test. Others block certain TCP and UDP ports for consumer-grade connections (like consumer cable modems, DSL lines, and FIOS) to help protect consumers on their networks. For example, connections to or from TCP port 25 are blocked by some ISPs to lower the chance that e-mail-relaying bots can be installed on their subscribers PCs. Such port blocking will prevent the tester from evaluating the security of services on those ports. Some ISPs even identify exploitation attempts and then block the exploit packets using network-based IPS tools. Such an action would prevent the tester from being able to move forward on that part of a test. Any of these issues would result in inaccurate results. Furthermore, performing scanning or exploitation through your ISPs network could be a violation of their terms of service. Violating those terms could result in service termination and a possible lawsuit.

Because of the concerns associated with using the ISP connection for testing, it is vital that you tell the ISP in advance that you plan on using the connection for penetration testing and security assessments (you may want to avoid the phrase "ethical hacking" as they may misunderstand the term "hacking"). They may tell you to seek another ISP, but it is better to determine their policy in advance than to find it out in the middle of a test. They may charge you extra for the connection or force you to buy a different grade of service. Still, such expenditures are a reasonable price of doing business.

# Testing Network Infrastructure – Firewall Concerns

- If your testing machines are firewalled from the Internet, your attacks might be blocked or neutered
  - NAT or PAT
    - Scan could fill up the tables, dropping packets
    - Attempts at reverse shell connections from target to attack system will not be carried back in
  - HTTP proxy
    - Exploit encoding could be altered, breaking exploit
  - Application-level inspection
    - May drop packets that don't conform to app-level protocol
    - Or try to "clean-up" protocol



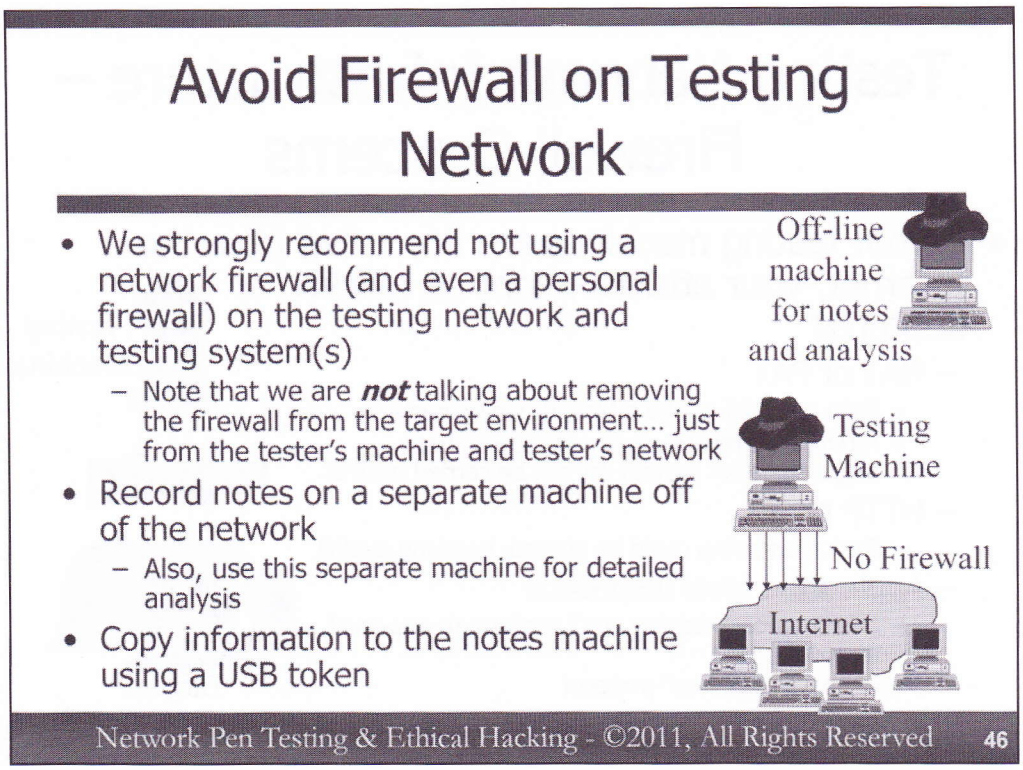

 

For the testing machines scanning across the Internet, you may want avoid personal and network firewall technologies. Firewalls may block inbound or outbound packets, yielding inaccurate results for a test. That's why most testers who need accurate, professional results don't use a firewall on their testing machines. Paul Asadoorian, host of the Pauldotcom podcast that covers various security and hacking issues, refers to the practice of hacking without a firewalls as "hacking naked." Without a firewall, the testing machine is not as protected as other systems in your network. Thus, don't store sensitive data on the testing machine, other than the temporary test results, which must be moved to another box in a timely fashion.

Network Address Translation (NAT) of firewalls cause problems for testers because they alter the source IP address of packets on their way out of the network, and map any response packets back to the original IP address. Port Address Translation (PAT) functionality not only alters source IP addresses of outbound packets, but it also alters source port numbers so that the response to the packet can be associated back with the originating host. Both NAT and PAT rely on tables in the firewall. A scan that creates millions of packets may overwhelm the NAT or PAT tables, causing them to drop packets, including subsequent probes and their responses. Thus, the scanning tool won't be able to see all of the results, missing open ports and vulnerabilities.

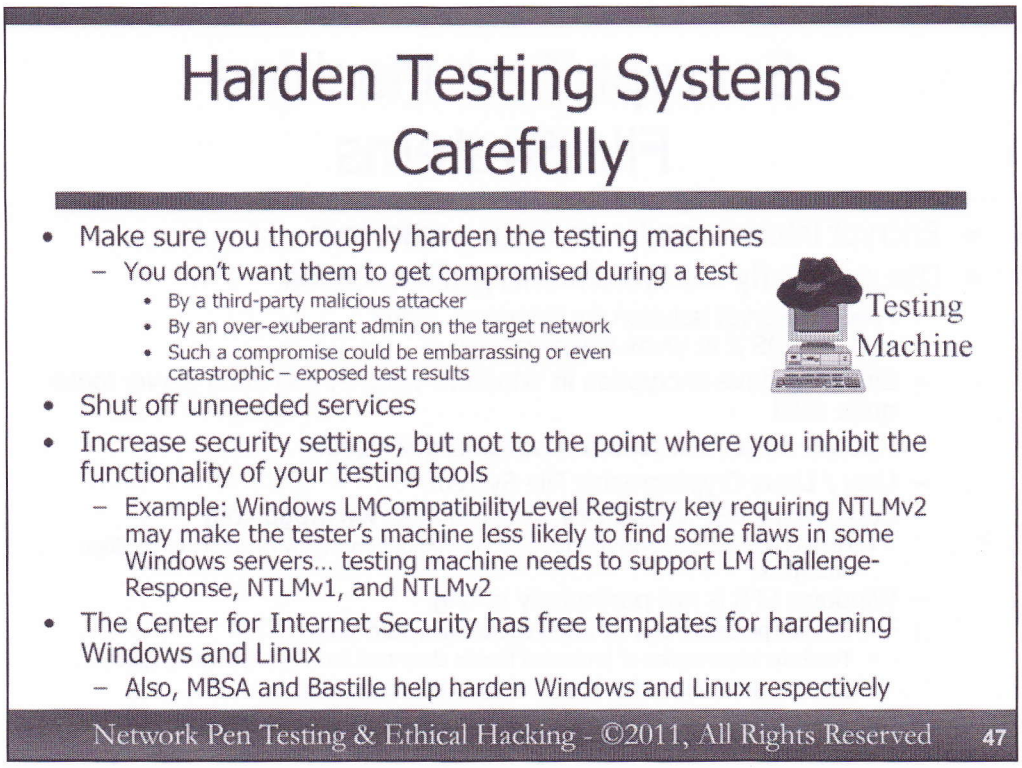
Furthermore, HTTP proxies sometimes alter the encoding of various web traffic that passes through them, which could break carefully calibrated exploitation code. An exploit passing through an HTTP proxy may simply not function at all or could cause a target system to crash.

The application-level inspection technology of some firewalls drops packets that do not properly confirm to the protocol, or even clean up some packet settings, forcing them to match the protocol spec as interpreted by the firewall vendor. Again, these changes could neuter an exploit or cause damage to the target.




For all of these reasons, we recommend that the testing machine not be located behind a network firewall, nor should it use a personal firewall on that specific system. Of course, the target network likely has a firewall, if it is being tested across the Internet. We're not talking about removing that firewall. Instead, we're referring to the pen tester's own testing systems connected to the Internet, and how they should not be located behind a firewall.

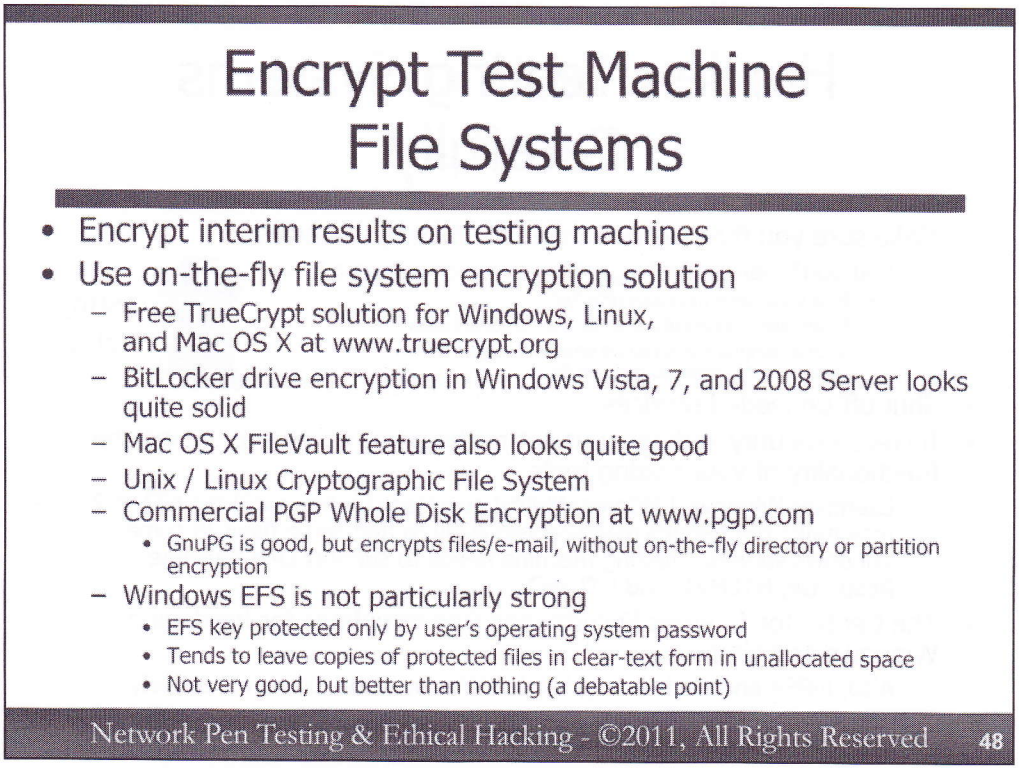
Additionally, use a separate, off-line system, not connected to the testing network, to take notes while you are performing a test. This separate machine will provide a more secure location for housing your notes, plus it gives you flexibility to analyze interim results while the testing machine is tied up with performance-draining scans and other activities. You can move files between the testing machine and the off-line machine using a USB token.




Although we recommend that you avoid using a network or personal firewall to protect your testing machines because of their potential impact on your test results, we do caution you. Make sure you that carefully harden the testing machines before starting a test. You must guard against compromise of the attacking machines during a test, by either a third-party malicious attacker or even an over-exuberant system administrator in the target organization. There have been cases of an administrator on a target network launching a counterstrike during a test, hacking back to the penetration tester's machines and compromising them. If someone compromises your testing machines, they could steal your interim test results and even alter the results they leave behind. Such exposure of test results could be, at best, very embarrassing, and at worst, catastrophic for your career as a penetration tester or ethical hacker.

For this reason, keep patches up to date on all of your testing machines, and shut off unneeded services. For a penetration testing system, you likely need no or only minimal listening services on the machine. The only services that should be listening are specific ones you need for your test, such as a web server set up to deliver a client-side exploit or a file server needed to serve up files to compromised target machines. You want to increase security settings of the testing machines beyond the defaults for the given operating system, but make sure that you don't inhibit the functionality of your testing tools. Harden the boxes, but verify on lab systems that your hardening process doesn't break needed functionality of your test tools. One area of hardening involves configuring a Windows machine to speak the stronger NTLMv2 protocol for authentication using the LMCompatibilityLevel Registry key. To attack less secure Windows servers, the tester's machine needs the ability to speak the older LANMAN Challenge-Response and NTLMv1 authentication mechanisms in addition to NTLMv2. We'll explore the differences between these authentication mechanisms in more detail in 560.4. For now, keep in mind that hardening a tester's machine to avoid these older and weaker protocols may hamper the system's ability to attack weakly configured targets.

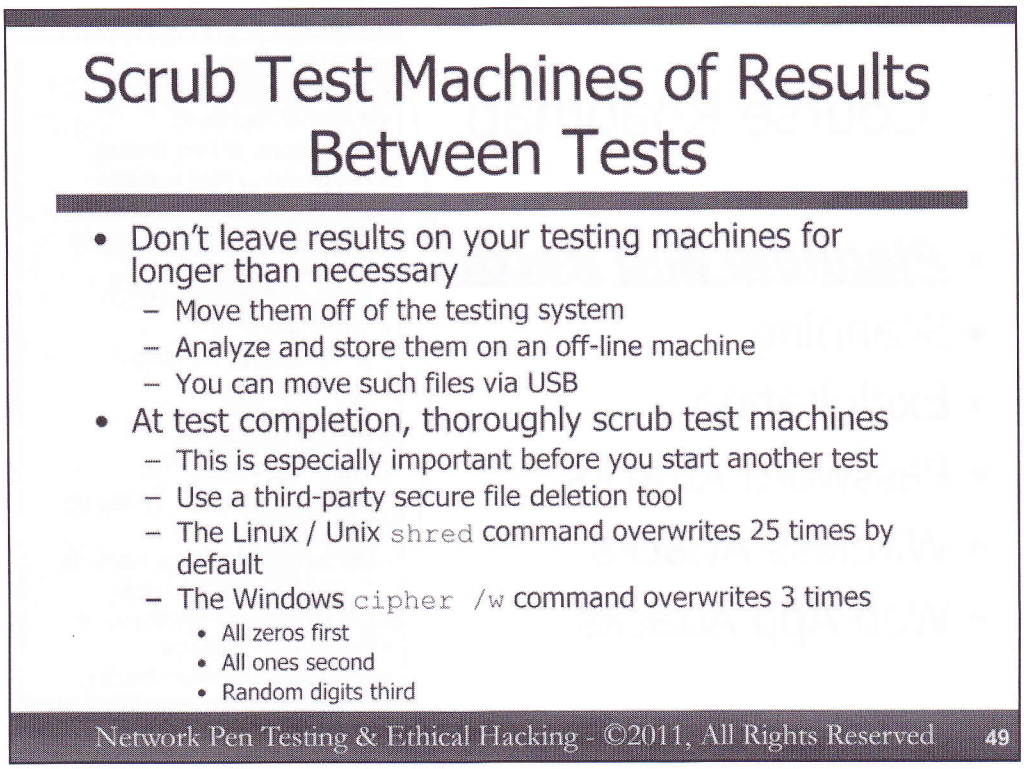
To help with this hardening process, the Center for Internet Security (www.cisecurity.org) has a large number of free templates for hardening various kinds of systems, including Windows and Linux. Download and use these templates. Furthermore, Microsoft's free Microsoft Baseline Security Analyzer (MBSA) can be used to check the security settings of Windows machines to verify that they have up-to-date patches and have been hardened. Also, the Bastille Linux project can help to harden Linux machines with solid security configurations.




In addition to hardening your testing machines, you also need to be very careful with the data on those machines. Professional penetration testers and ethical hackers should consider using a file system encryption solution on their testing machines to lower the probability of test results exposure. Ideally, you'll use an on-the-fly encryption solution that encrypts a whole directory or entire partition, letting you seamlessly drop files into a directory to have them automatically encrypted.

Numerous file system encryption solutions are available. TrueCrypt is a fantastic free disk encryption tool, available for Windows, Linux, and Mac OS X at www.truecrypt.org. Alternatively, the Windows BitLocker feature, included in Windows Vista, 7, and 2008 Server looks quite solid, although it only functions on Windows machines. The Mac OS X FileVault feature likewise provides a good boost of security in protected files, but is again focused on a single platform: Mac OS X. The Cryptographic File System (CFS) has been ported to most Linux and Unix variations. From a commercial perspective, the PGP Whole Disk Encryption tool provides very useful functionality, with vendor support. Gnu Privacy Guard (GnuPG) is a fine free, open-source tool for encrypting files and e-mail. However, as of this writing, GnuPG does not support on-the-fly directory or partition encryption.

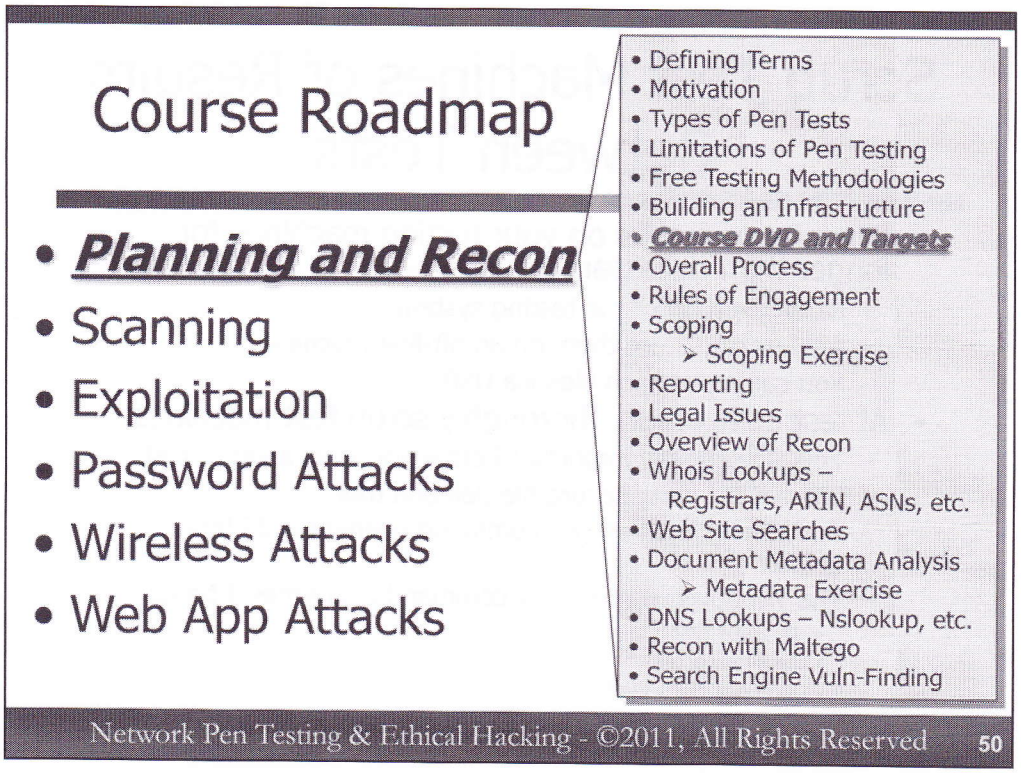
Windows has built-in file system encryption functionality called the Encrypted File System (EFS). Unfortunately, the security of EFS leaves a lot to be desired. It protects crypto keys only with the user's operating system password, which can be extracted and cracked, as opposed to the separate passphrase most other solutions employ. Furthermore, EFS often leaves clear-text copies of recently deleted or encrypted files in unallocated space of the file system, until they are wiped. Although EFS is not a particularly strong solutions, many people believe it is better than nothing. However, some worry that EFS, with is problems, may lull people into a false sense of security. Thus, we recommend that you use a stronger solution, such as TrueCrypt or PGP.



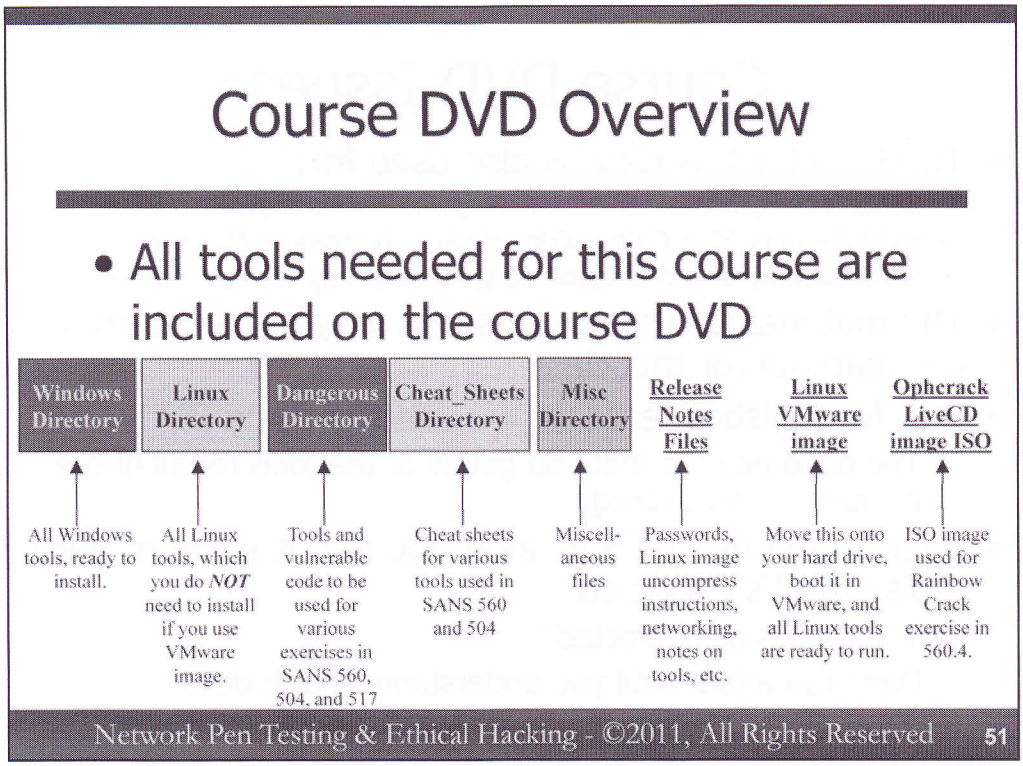

By itself, encrypting your sensitive test results information is not enough. You must also securely remove your test results from your testing machines. Don't leave data on the testing machines for any longer than is necessary during a test. Move results files, including output from scanning tools, files containing password guessing results, and any notes that you create, to your notes and analysis machine on a regular basis using a USB token.

Periodically during a test, and especially when a test is completed, you must securely wipe interim results files from the testing machines. This is especially important before you start another test of another target environment, because you don't want to mix up your results, or have the results of one target exposed to another target.

Merely deleting the files isn't enough, because that simply moves them into unallocated space, making them still recoverable. Instead, use a secure file deletion tool that overwrites files with alternating zeros and ones multiple times to make sure the file cannot be recovered. Most versions of Linux have the `shred` command, which overwrites files with alternating zeros and ones 25 times (and can be configured for more), making files almost impossible to recover. Windows has a built-in tool called `cipher`, which can be used with the /w:[file] option to overwrite all free space on the partition where [file] resides with all zeros, followed by all ones, followed by random numbers (three passes total). While three passes is less than we'd like to see, it is acceptable for most environments.




We'll now go over some of the technical infrastructure components of the class, including the course DVD and the target machines we'll be working with all week. Make sure you have a copy of the course DVD handy as we go through the next several slides.




On the course DVD, at the top of the directory structure, there are a handful of files and directories. Among the most important of these files are the Release Notes for the course DVD, which are named "Release Notes for SANS [version_number]". They are included in both .doc and .txt format, so you should be able to read them in any doc-format compatible word processor or even any text editor. The Release Notes include the userIDs and passwords for the VMware image for the course, as well as information about getting that VMware image uncompressed, booted, and networked. They also contain additional notes about some of the individual tools. ***As long as you have the course DVD, you will also have these Release Notes, and therefore will have access to the root password for the course VMware image. Remember that, please, so you won't be stranded without the password.***

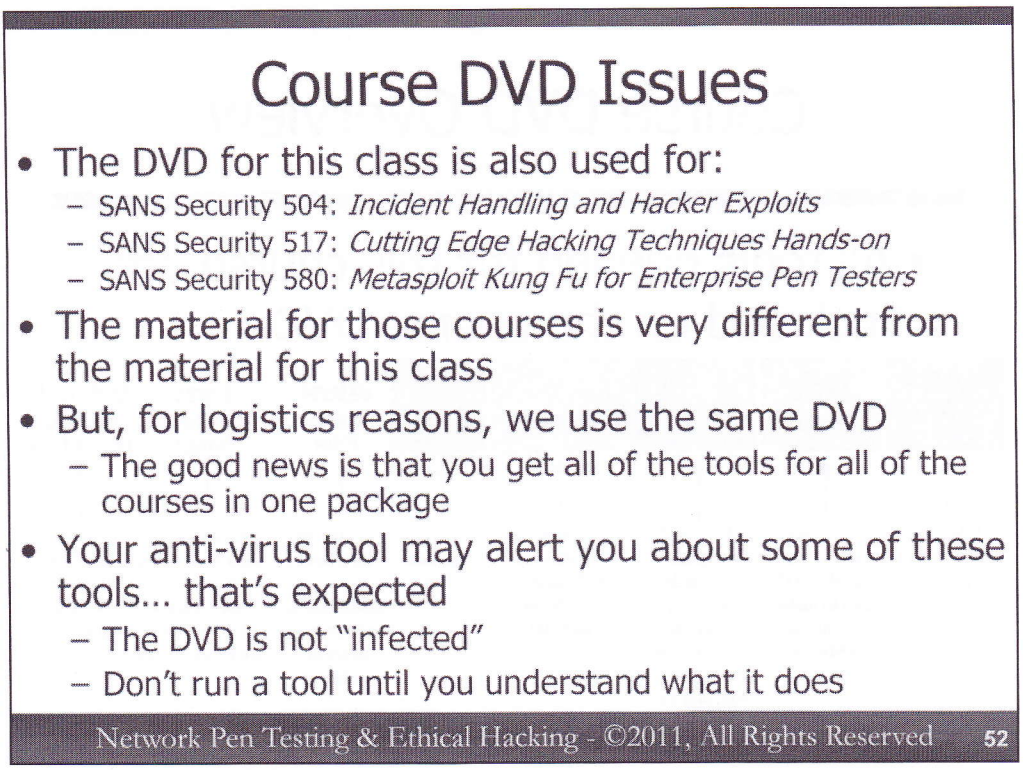
Next, we have the course Windows directory. All Windows tools that you'll need for the course are included here. You will have to install them on your Windows machine when the time comes for each exercise. Please do wait, though, until we start a given exercise so you can understand a tool before installing it. We also have a Linux directory that includes all of the Linux tools for the course. Note that most attendees will actually not use this directory, because you'll be running the pre-installed tools included in the VMware image. We've included this directory for completeness.

The next directory is called "Dangerous" because it includes programs that could introduce vulnerabilities on your system or open backdoors. We'll use such programs in exercises at various points in the course to illustrate pen testing techniques for exploiting such flaws. Do not run these items until you understand what they do.

Another directory, called "Cheat_Sheets", contains cheat sheets for various tools covered in this class. Feel free to look through it. These sheets can be very helpful in exercises throughout the course and the pen test workshop in 560.6. And, a Misc directory contains miscellaneous files, including a sample pen test report.

The next element of the DVD is a large file called SANS504Linuxv[Version].zip. This zipped file contains a VMware image of a Linux machine, ready to boot in VMware Workstation, VMware Player, or VMware Fusion (the Mac OS X product). You were required to bring VMware with you. VMware is not included on the course DVD, due to redistribution limitations imposed by VMware.

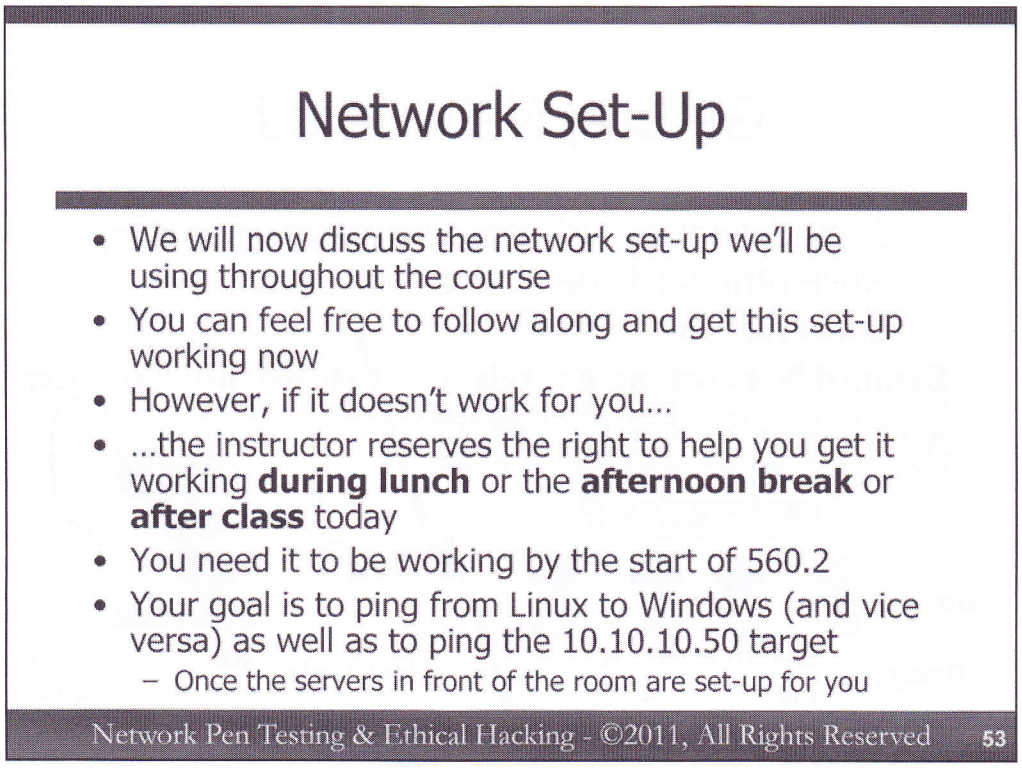
And, finally, we have an ISO image of a bootable CD called the Ophcrack LiveCD. We'll use this image in an exercise in 560.4 to conduct Rainbow table password cracking.




The DVD that you received for this course is also used for other SANS courses, including SANS Security 504 (the Incident Handling and Hacker Exploits course), SANS Security 517 (the Cutting Edge Hacker Techniques Hands-On course), and SANS Security 580 (Metasploit Kung Fu for Enterprise Pen Testers).

The material and topics covered for each of these courses is very different, but we keep them all on the same DVD for logistics reasons (making sure that the appropriate DVD is in the right place at the right time would be more complicated with different DVDs for each course). This arrangement has positive implications for you, because you get all of the tools for each of the courses, all on one handy DVD, with a VMware Linux image pre-configured to run them all.

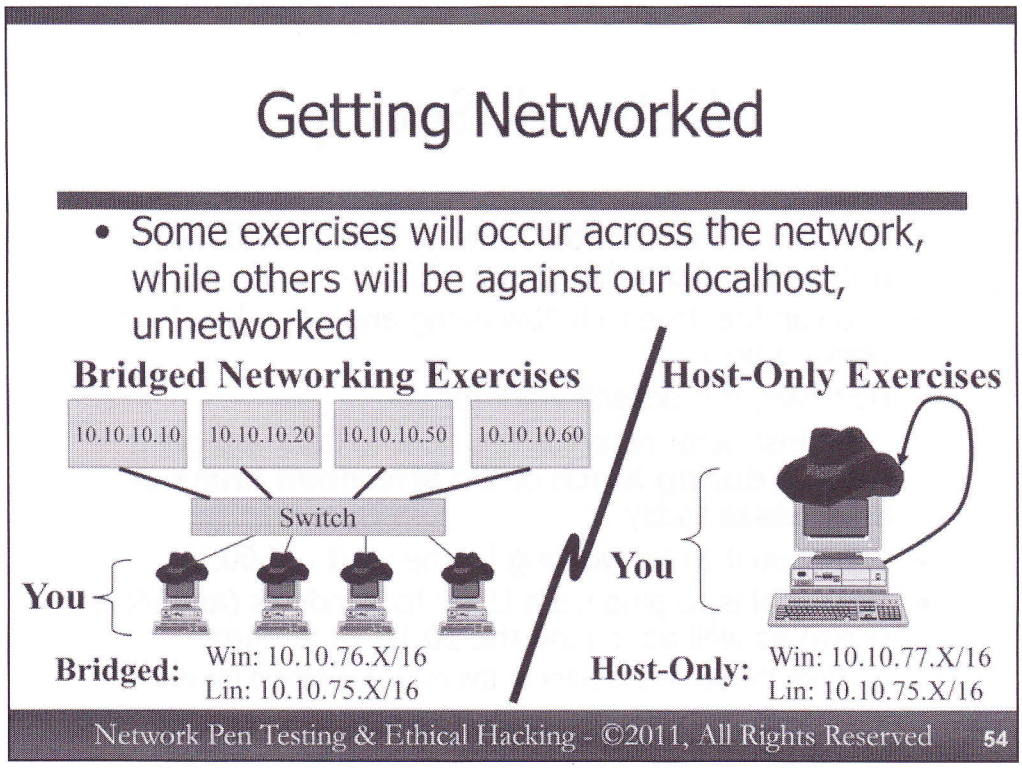
However, please note that your Anti-Virus and Anti-Spyware tools may alert you regarding some of the items on the course DVD. The classes that the DVD supports deal with computer attacks and exploits, and some anti-malware vendors classify particular tools as malicious code. The course DVD is not "infected." Instead, it merely contains software that can be used in a malicious fashion, and therefore it triggers the signature-based detection of some Anti-Virus and Anti-Spyware tools. Don't run a program that your AV/AS tool alerts you about unless you understand what the program is designed to do and until you understand how to use the program safely.




We'll now discuss the VMware network configuration we'll be using for the course. You can follow along and set up the networking during this session, if you'd like.

But, please note: If the your network configuration does not work, the instructor may help you trouble shoot it during lunch today, the afternoon break today, or at the end of class today. Not everyone will get the network configuration functional right now, and the instructor needs to keep the course moving. **Don't worry if the networking doesn't work for you right now**. The instructor will help you get it functioning later today. You will only need the networking for the course from 560.2 through 560.6. It is not needed for the afternoon exercise today.

Your goal in getting your systems networked is to ping from Windows to Linux (and vice versa) as well as to ping 10.10.10.50, a machine that will be located in front of the room.




Throughout this course, we'll have numerous hands-on exercises so you can gain experience by practicing the various techniques we'll describe. Most of the hands-on exercises for this class will occur across a network, but some will occur against your local host.

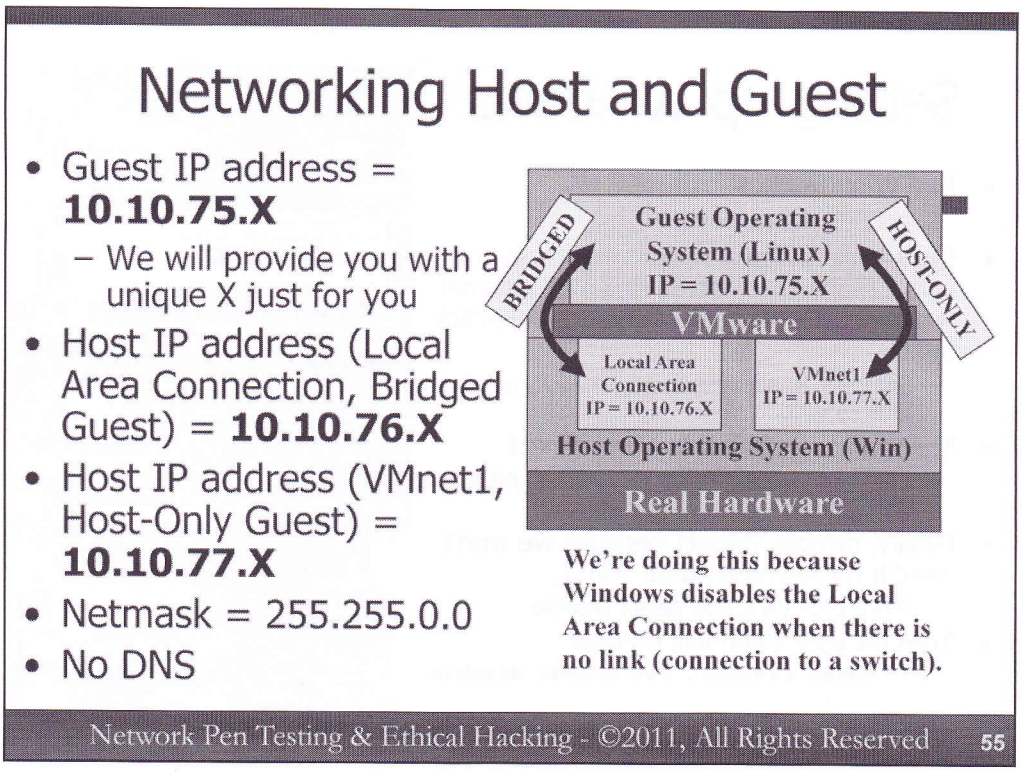
For Networked Exercises, you'll be attacking four target machines on the 10.10.10 subnet, including 10.10.10.10, 10.10.10.20, 10.10.10.50, and 10.10.10.60. YOU ARE ALLOWED TO ATTACK ONLY THESE MACHINES. DO NOT ATTACK YOUR FELLOW ATTENDEES' SYSTEMS. IF YOU DO ATTACK OTHER MACHINES OUTSIDE OF THE 10.10.10 NETWORK, YOU MAY BE DISMISSED FROM THE CLASS, AND THERE COULD BE LEGAL IMPLICATIONS.

If you are taking this class at a live conference, you will be connected to these targets using one or more switches provided in the room. If you are taking this class across the Internet (via vLive, OnDemand, or other offering), please flip forward to the slide that describes your VPN Configuration.

For *Bridged Networked Exercises*, your virtual machine configuration (VMware) should be set to Bridged networking, with your Windows IP address being 10.10.76.X (X will be given to you by your instructor or a room facilitator) with a netmask of 255.255.0.0. No default gateway or DNS is required. Your Linux IP address will be 10.10.75.X (the same X you received from the instructor or facilitator) with the same netmask.

For the *Host-Only Exercises*, you will be running attacks from your virtual machine guest against your host machine. In these exercises, you'll need to configure your virtual machine for host-only networking, and your Windows IP address will be 10.10.77.X (note that it is 76.X for Networked Exercises and 77.X for Host-Only Exercises). Your Linux IP address will always be the same: 10.10.75.X.

We will now cover how to configure your systems for these exercises.



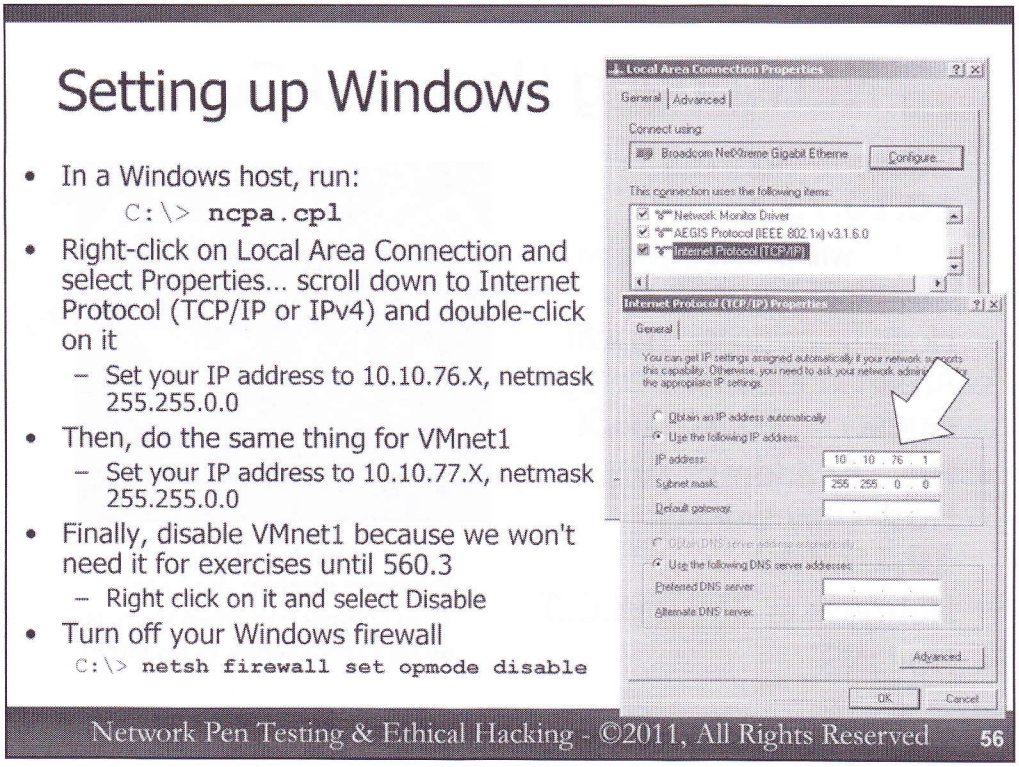

We will now discuss the networking configuration that you will apply to your machines for the exercises in this class. Please pay close attention, as you'll need to know the networking configuration as you work through the hands-on exercises throughout the remainder of the course. Most people who take this course are using a Windows Host machine, running VMware, with a Linux Guest machine inside of VMware. If you are using a different model, such as a Linux Host or a Mac OS X Host, we'll cover your configuration in a few slides.

Your Linux Guest will have an IP address of 10.10.75.X, where X, the last octet of your IP address, will be given to you by the course instructor. Remember this X. In fact, write it down so that you have it with you for the remainder of the course.

Your Windows Host machine will actually have two IP addresses, for two different interfaces. Its Local Area Connection address, which you will assign to your physical LAN adapter when connected to our network, will be 10.10.76.X. This will be the address you'll use when VMware is configured for Bridged networking for a Networked Exercise.

Your other Windows Host IP address will be for VMnet1, a network interface created when VMware was installed. Its address should be 10.10.77.X. You'll use this address when performing exercises against your own machine (Host-Only Exercises) when not connected to a switch, using VMware in host-only mode.

The netmask for all of your interfaces should be 255.255.0.0, a /16 network. We have no DNS server or default gateway for the exercises. There is a DNS server located at 10.10.10.60, but you do not need to configure your machine to use it. We are a flat (non-routed) network here, so there is no default gateway for the exercises in 560.1 through 560.5. We will have a separate network configuration with routers for 560.6, when we conduct our final penetration testing workshop.




You can set up your Windows networking by opening up your network interfaces. One of the easiest ways of doing this is to launch (at an Administrator cmd.exe prompt):

```
C:\> ncpa.cpl
```

You should see all of your networking interfaces. Right click on your *Local Area Connection* interface and select Properties. Then, scroll down to where it says "TCP/IP" or "TCP/IPv4" and double click. Then, set your IP address to 10.10.76.X and netmask to 255.255.0.0. Click OK and then Close.

Then, configure *VMnet1* by right clicking on its network interface and selecting Properties. Scroll down to TCP/IP or TCP/IPv4, double click, and set your IP address to 10.10.77.X and netmask to 255.255.0.0. Again, click OK and Close.

Our first set of exercises for the class will utilize bridged networking. When using Bridged networking, we want to disable VMnet1, otherwise it may absorb packets. So, in your network connections screen, right click on VMnet1, and select Disable. We'll re-enable it later when we have a Host-Only Exercise.

And, finally, we'll need our Windows firewall to be disabled so that we can get unfettered access to and from our machines. Disable the built-in Windows firewall by running:

```
C:\> netsh firewall set opmode disable
```

If you have a third-party firewall on your Windows box, you will need to disable it, or create exceptions to allow your guest and various applications to communicate for our exercises.

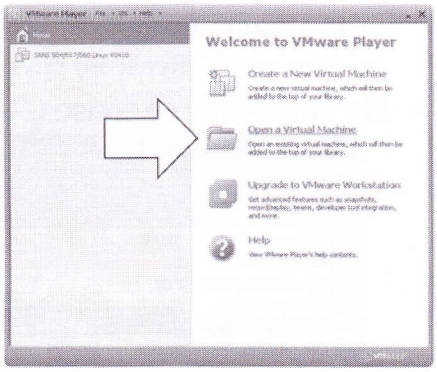
# Unzipping Linux

- Unzip the Linux image from the course DVD
  - Large ZIP file (> 1 Gig), version indicated on the DVD face
  - It is typically best to just unzip it to your desktop... unzipped requires ~ 5 GB
- Run VMware, open the VM, and boot it
- In Linux, login to the system
  - username=student
  - password=!linuxpw!
- Then, get a root prompt:

  $ **su -**
  - Please remember to always use that minus after su!
  - Type in root password of !templinpw!
  - Change the root password:

  # **passwd**
    - Enter a new password twice, and remember it!
  - Change the password for the student account:

  # **passwd student**






Next, we'll unzip the Linux guest VM image from the DVD into the hard drive. Unzip all of the files included in the large ZIP image on the course DVD. It is usually best to just unzip its entire contents into a directory on your desktop for easy access throughout the course. This file will require approximately 5 GB of space on your hard drive when unzipped.

After the file is unzipped, run VMware, select "Open a Virtual Machine", and boot your guest system. When prompted, login to the guest machine using the following credentials:

**Username=student**

**Password=!linuxpw!**

Now, su to root by running:

$ **su -**

REMEMBER TO ALWAYS USE THE MINUS AFTER THE SU FOR THIS CLASS. That way, you'll have root's privileges AND root's environment variables (including the PATH). Otherwise, your system won't have the proper PATH, and it won't be able to find some commands we'll want to run.

Type in the root password of !templinpw! to finish su'ing to root.

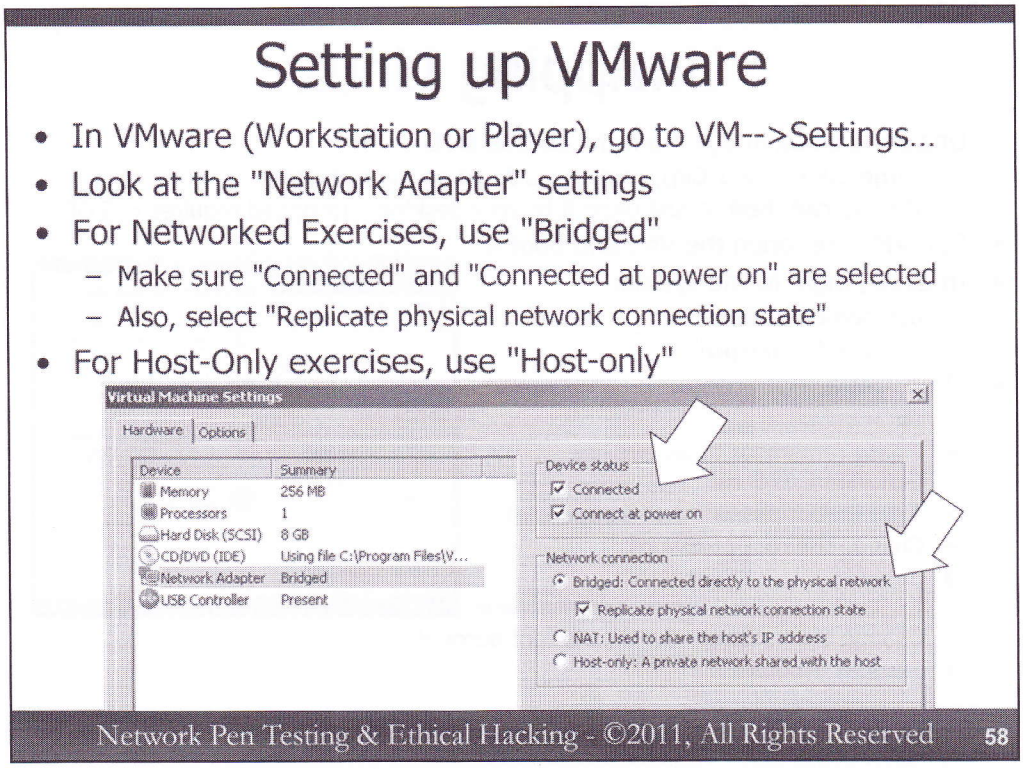
Now, change root's password to a value you'll remember but that isn't easily guessed or cracked. We'll be connected to a network with other students in this course, and you do not want them to know the password for your Linux VMware image.

# **passwd**

Enter your chosen password once, and then again to set it.

Finally, change the password for the student account:

# **passwd student**



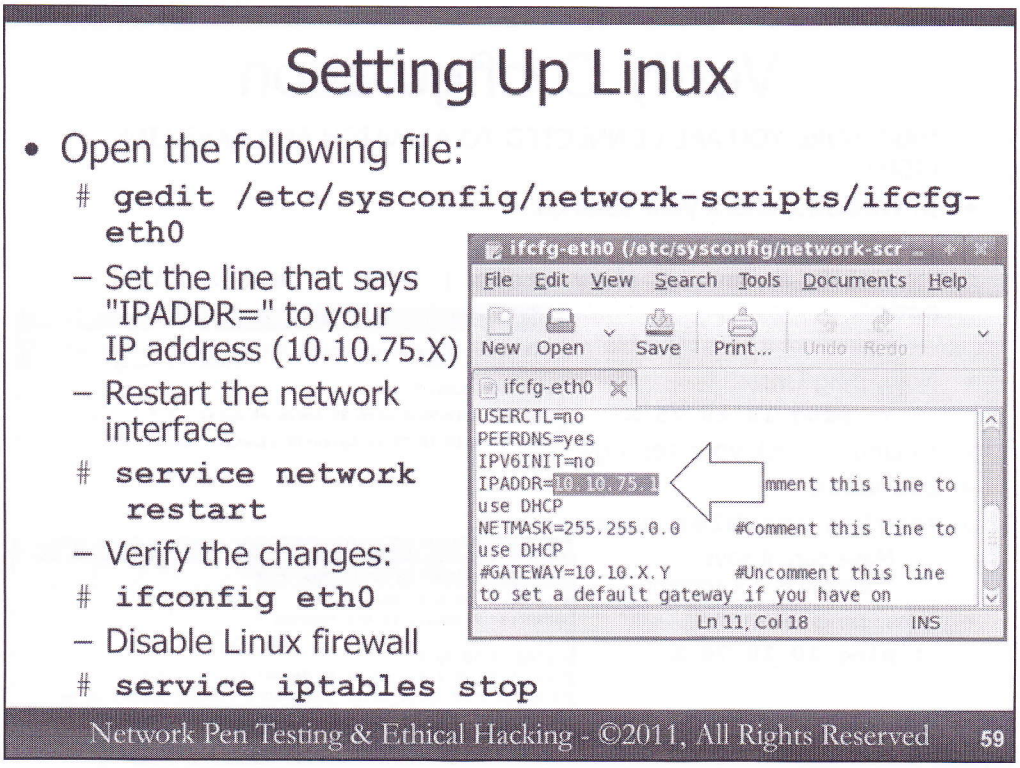

Next, we'll set up VMware. If you are using VMware Workstation 7.0 or later, or VMware Player 3.0 or later, you can set your network adapter's configuration by going to VM-->Settings… If you are using an earlier version of VMware, the configuration is similar, but may have slightly different wording, depending on the particular version you are using. If you have trouble locating the settings in those earlier versions of VMware, please ask your instructor.

In VM-->Settings, click on "Network Adapter".

For Networked Exercises, select "Bridged" networking. Also select "Replicate physical network connection state". Make sure that "Connected" and "Connected at power on" are both enabled.

When we switch to a Host-Only exercise, you'll need to go back into this setting (VM-->Settings…) and select the Host-only radio button.

For now, make sure you are set to Bridged networking, for our first set of exercises.




Now, we'll set up our Linux networking. In your Linux guest virtual machine, DO NOT SET THE IP ADDRESS USING THE GUI-BASED NETWORK CONFIGURATION TOOL. IT TENDS TO BE BUGGY.

Instead, set the IP address by editing a file, which you can access by running this command:

```
# gedit /etc/sysconfig/network-scripts/ifcfg-eth0
```

Let the tab-autocomplete do most of that typing of this command for you, so you can make sure you avoid any typos.

Inside the file, find the line that says "IPADDR=" and change it to the IP address for your Linux machine (10.10.75.X), again using the X value provided to you for the class.

Restart your network interface to apply the changes:

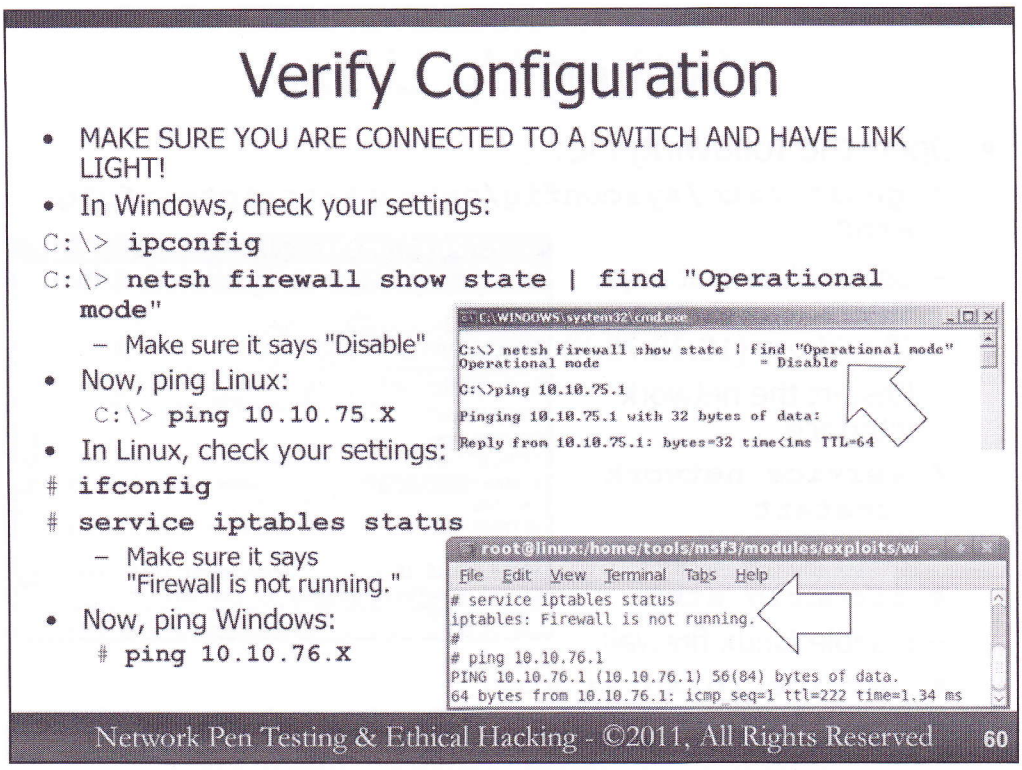
```
# service network restart
```

Let's verify that the changes were applied (look at the IP address in the output):

```
# ifconfig eth0
```

And, finally, let's disable our firewall for Linux so that we can make arbitrary outbound and inbound connections:

```
# service iptables stop
```




Finally, let's verify our configuration and make sure we have connectivity between our guest and host machines. We'll start from Windows and then work our way to Linux.

FIRST, BECAUSE WE ARE USING BRIDGED NETWORKING, YOU NEED TO MAKE SURE YOU ARE CONNECTED TO A SWITCH. YOU MUST HAVE LINK LIGHT FOR BRIDGED NETWORKING TO WORK WITH A WINDOWS HOST MACHINE.

In Windows, first check your IP address for your Local Area Connection:

```
C:\> ipconfig
```

Then, check your firewall settings, making sure the output line says "Disable":

```
C:\> netsh firewall show state | find "Operational mode"
```

Now, from Windows, try to ping Linux:

```
C:\> ping 10.10.75.X
```

Then, in Linux, verify your network configuration:

```
# ifconfig
```

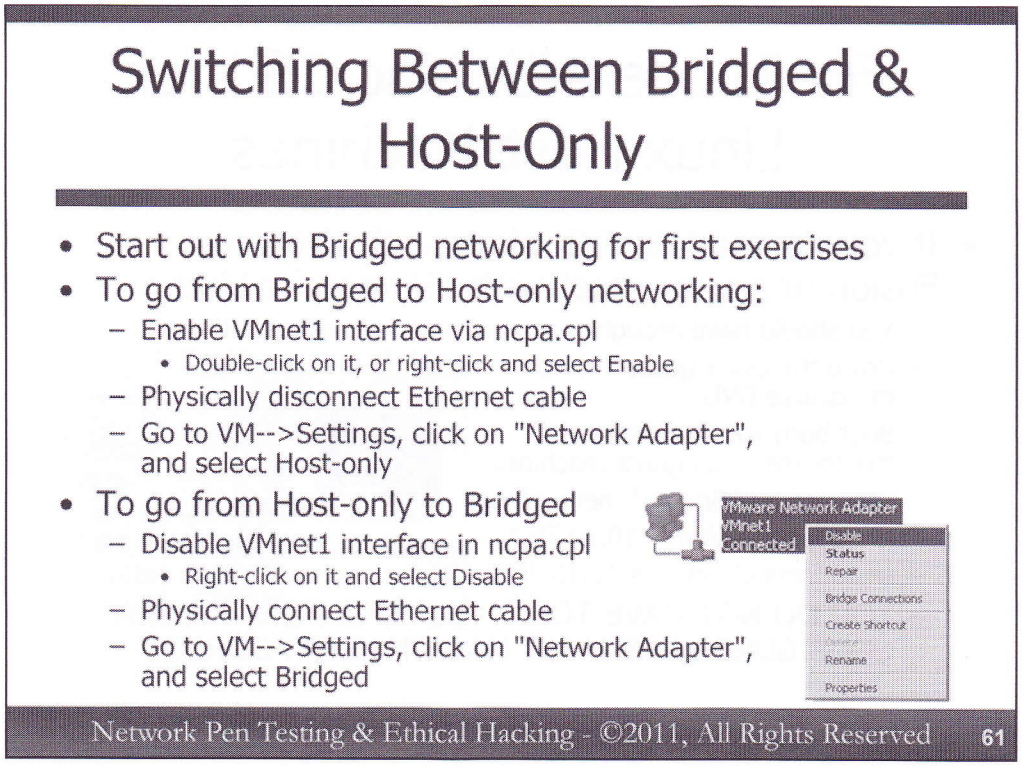
And, check that your firewall is disabled (making sure it says "Firewall is not running."):

```
# service iptables status
```

And, finally, try to ping Windows:

```
# ping 10.10.76.X
```

If you see ping responses from Windows to Linux and from Linux to Windows, you are configured and ready for the exercises. If not, please contact the instructor or room facilitator.




As we discussed earlier, most exercises in this class (including our first set of exercises), will require Bridged networking. But, a few of the exercises are Host-only, to protect your Windows machines from attacks when we are running vulnerable software on them.

Periodically in the class we'll need to switch from Bridged networking to Host-only, or vice-versa, to address the transition between the two different kinds of exercises.

To go from Bridged networking to Host-only networking, you will need to do the following steps:

1)Enable the VMnet1 interface by running ncpa.cpl and double-clicking on the appropriate interface icon (or right-click on it and select "Enable").

2)Physically disconnect your Ethernet cable.

3)Inside of VMware, go to VM-->Settings, click on "Network Adapter", and select the Host-only radio button. Make sure your Linux machine can ping Windows VMnet1 (10.10.77.X)

To go from Host-only to Bridged networking, you will need to do the following steps:

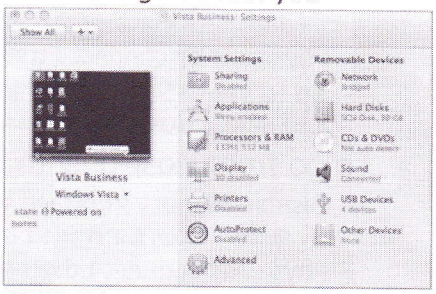
1)Disable the VMnet1 interface by running ncpa.cpl, right clicking on the interface, and selecting "Disable".

2)Physically connect your Ethernet cable to your computer. Check to make sure you get link light.

3)In VMware, go to VM-->Settings, click on "Network Adapter", and select "Bridged". Make sure your Linux machine can ping the Windows Local Area Connection (10.10.76.X).

# For Those with Mac OS X or Linux Host Machines

- If you have a Mac OS X host with VMware Fusion or a Linux host with VMware for Linux:
  - You should have brought your own Windows guest with you
  - Unzip the Linux guest VM from the course DVD
  - Boot both your Windows and the course Linux guest machines
  - Set both for "Bridged" networking
  - IP address of Win = 10.10.76.X
  - IP address of Linux = 10.10.75.X
  - YOU DO NOT HAVE TO CONFIGURE YOUR HOST MACHINE'S IP ADDRESS






Some people who take this class do not use a Windows host machine, but instead rely on Mac OS X with VMware Fusion or Linux with VMware for Linux as their environment. Such systems will still work with our guest machine, and the networking becomes a little bit easier.

If your have a Mac OS X or Linux host machine, you will still need to unzip the VMware Linux system we provided on the course DVD to your hard drive. This will be one of your guest machines. You were required (in the course laptop instructions on the registration page for the course) to bring a Windows guest machine with you.

Boot both your Windows guest VM, and the Linux guest VM unzipped from the course DVD.

YOU WILL NOT HAVE TO PROVIDE AN IP ADDRESS TO YOUR HOST MACHINE AT ALL. Set both of your guest machines to "Bridged" networking. Now, on Windows, using ncpa.cpl, configure the IP address of your Windows guest machine Local Area Connection to 10.10.76.X.

In Linux, configure the IP address of eth0 to 10.10.75.X by running:

```
# gedit /etc/sysconfig/network-scripts/ifcfg-eth0
```

Alter the line that says IPADDR=, adding your IP address of 10.10.75.X to it.

And, run:

```
# service network restart
# service iptables stop
```

Disable your Windows firewall:

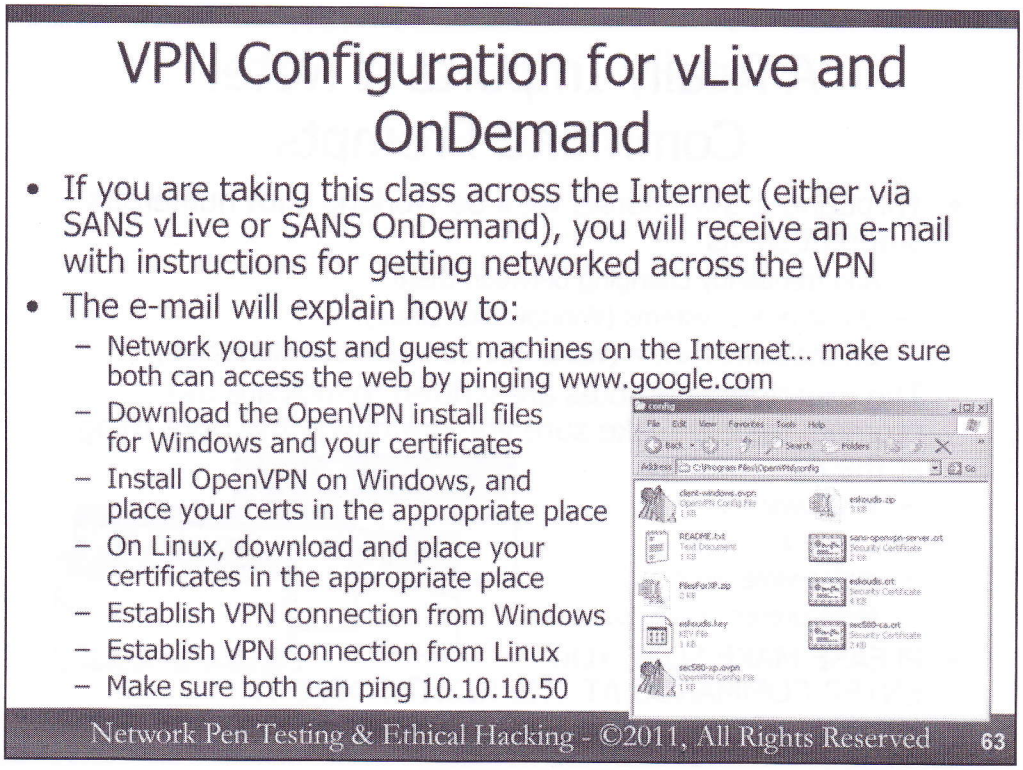
```
C:\> netsh firewall set opmode disable
```

And ping from Windows to Linux:

```
C:\> ping 10.10.75.x
```

Finally, you should be able to ping from Linux to Windows:

```
# ping 10.10.76.X
```

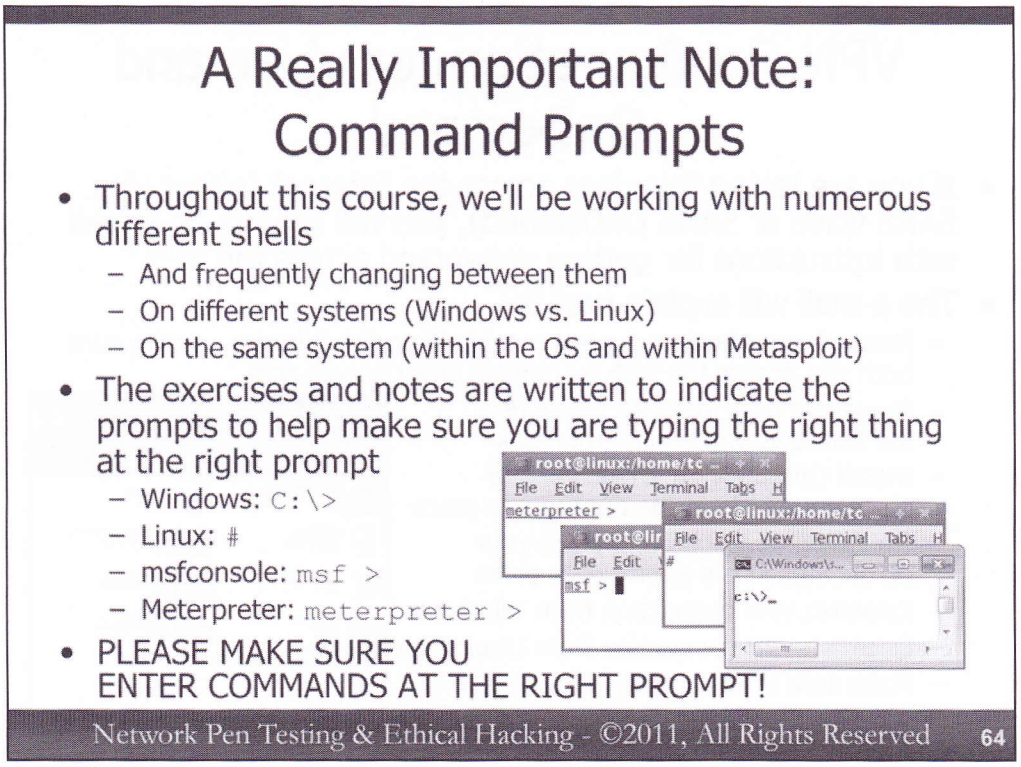



If you are taking this course across the Internet (either via SANS vLive or SANS OnDemand), you will need to set up OpenVPN on your Linux and Windows machines to conduct the bridged networking exercises in the class so you can reach target systems we have prepared.

You will receive an e-mail from SANS NOC personnel that describes in detail the process for configuring your system to use the VPN. The e-mail will explain various steps, including:

- Set up your Linux guest and Windows host machine on the Internet. Both machines must be able to reach Internet destinations. For Linux, use bridged networking, and configure your guest machine with an IP address for your environment or pull one using DHCP (edit /etc/sysconfig/network-scripts/ifcfg-eth0). If you are using hard-coded IP addresses, simply set it in the line that says "IPADDR=". If you are using DHCP, make sure you set BOOTPROTO=dhcp. Make sure both your Windows and Linux machines can ping some site on the Internet, such as www.google.com.
- Download the OpenVPN install files for Windows, along with your certificates, as described in the e-mail from the SANS NOC. Put your certificates in the appropriate place (C:\Program Files\OpenVPN\config). You do not need to install OpenVPN software on the Linux guest image we provided, as this software is already installed.
- On Linux, place your downloaded certificates in the appropriate place (/etc/openvpn).
- Establish the VPN connection from Windows (by right-clicking on the OpenVPN icon in your tooltray, and selecting "Connect"). Provide your SANS portal password to connect.
- Establish the VPN connection from Linux (by running "service openvpn start"). Again, provide your SANS portal password when prompted.
- Finally, make sure both Windows and Linux can ping 10.10.10.10 while the VPN is connected.

Note: To communicate between your Linux guest and Windows host while connected to the VPN, you could use the IP address assigned to you by the VPN (viewable via the OpenVPN tooltray client in Windows and as the tap0 interface displayed by the "ifconfig" command on Linux), or the IP address of your network adapter (Local Area Connection in Windows and eth0 in Linux).




Throughout this course, we'll be using numerous different shells, both in our operating system and within Metasploit. We'll frequently be changing between these different shells as we switch back and forth between Linux and Windows, and within different aspects of Metasploit itself. Sometimes, even on a single page in the book, you will be using two or even three different types of shell to do something and then observe the results.

All of the exercises and notes were carefully written to indicate the proper shell you are supposed to be using at any given time by including the shell prompt right before each command you are supposed to type. That is, each exercise command is preceded by the prompt indicating which shell to use. The shell types you will encounter throughout this class include:

A Windows cmd.exe with the prompt:

```
C:\>
```

A Linux bash shell (which we'll run as root) with the prompt:

```
#
```

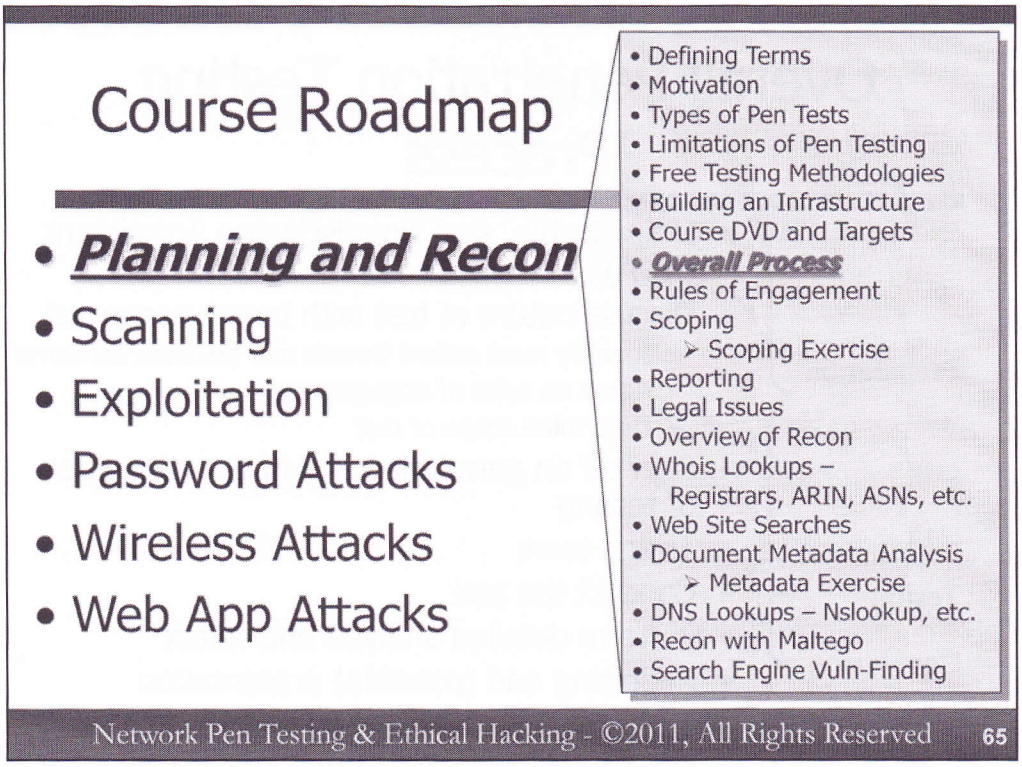
The Metasploit Framework Console with the prompt:

```
msf >
```

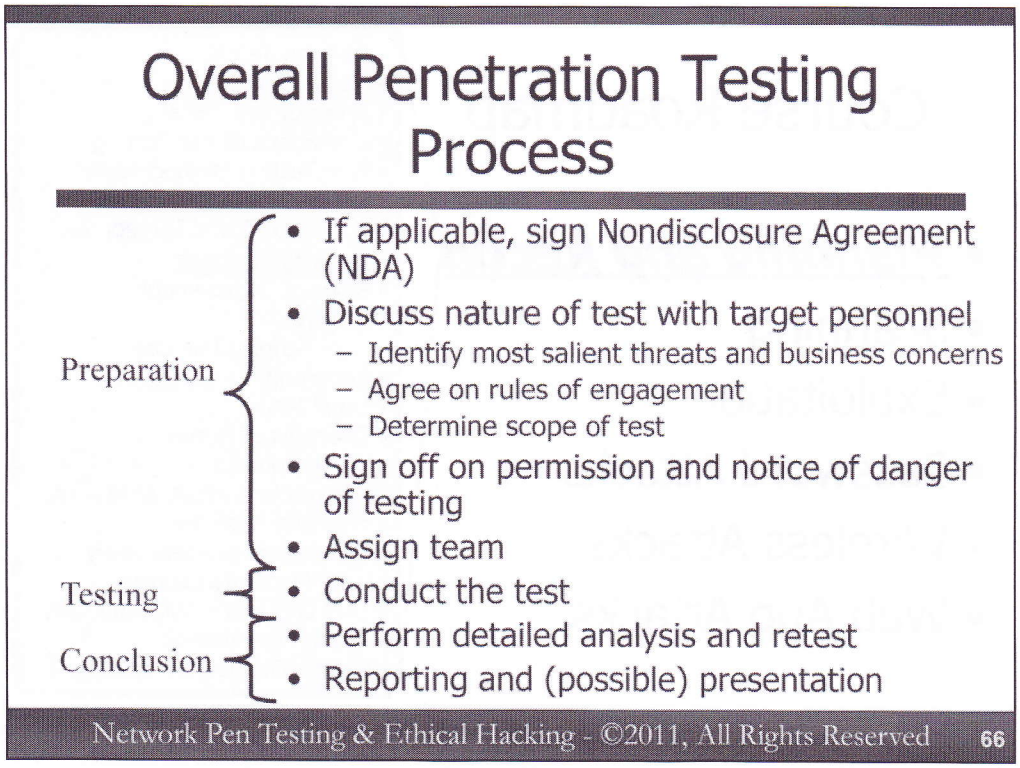
A Meterpreter shell with the prompt:

```
meterpreter >
```

PLEASE DOUBLE CHECK AT EACH EXERCISE STEP THAT YOU ARE ENTERING THE PROPER COMMAND INTO THE PROPER SHELL. Otherwise, a given exercise step will not work for you properly.




With our infrastructure in place, let's now go over the overall ethical hacking and penetration testing process. This process should be applied to all of the testing that you do. Be very careful in skipping any of the steps we'll describe. Some steps are designed to ensure that you've conducted a test with appropriate legal protections. Furthermore, other steps help ensure that you are providing demonstrable value to the organization you are testing.



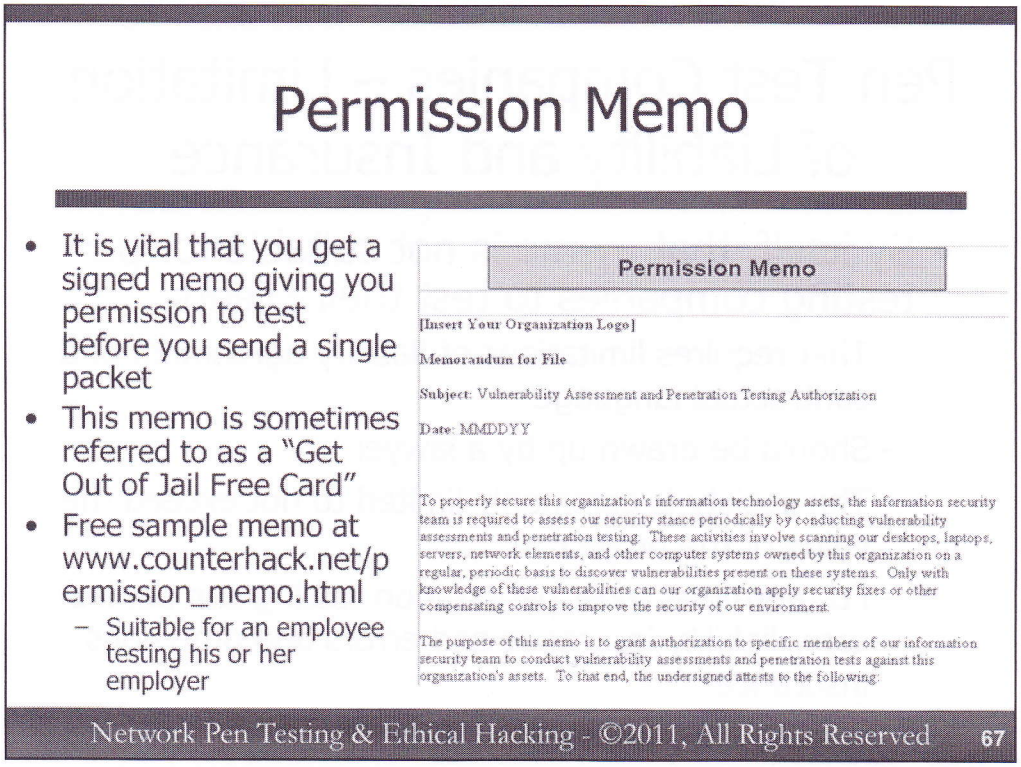

The overall penetration testing process involves preparation, testing, and conclusion phases.

During the preparation phase, the parties participating in the test may sign a non-disclosure agreement (NDA), especially if the test is conducted by a third-party organization. Then, the testers and the target personnel discuss the most significant concerns of the target organization. What are the biggest threats? Which systems are the most sensitive? What kind of information is the most valuable? We also agree on rules of engagement that describe how the testing will occur. Next, the scope of the test is determined, a process we'll discuss in depth later.

The next step is absolutely crucial. You need to get official, written permission to conduct the test, even if it is against targets in your own organization. This permission should notify the personnel associated with the target systems that there is some danger of their systems being crashed or impaired by the testing. We'll discuss this permission memo in more detail shortly. Then, based on the nature of the test, a team of appropriate testers is assigned, based on their technical areas of expertise and business knowledge of the target environment.

The test then occurs, potentially lasting from a day to many months.

To conclude, the team then analyzes the results, trying to discern their business implications. The technical details and business implications are described in detail in a final report. As findings are addressed, single issue retests could occur, or an entire comprehensive retest may happen. Some tests conclude with a wrap-up final presentation.




Let's zoom in on that one crucial element of the previous slide: the permission memo. It is absolutely vital that you get a signed memo from a leader of the target organization giving you permission to test their environment. This memo is sometimes called a "Get Out of jail free card", or GOOJFC for short.

There is a free sample memo on the Counter Hack website at www.counterhack.net/permission_memo.html. Among other things, this memo states:

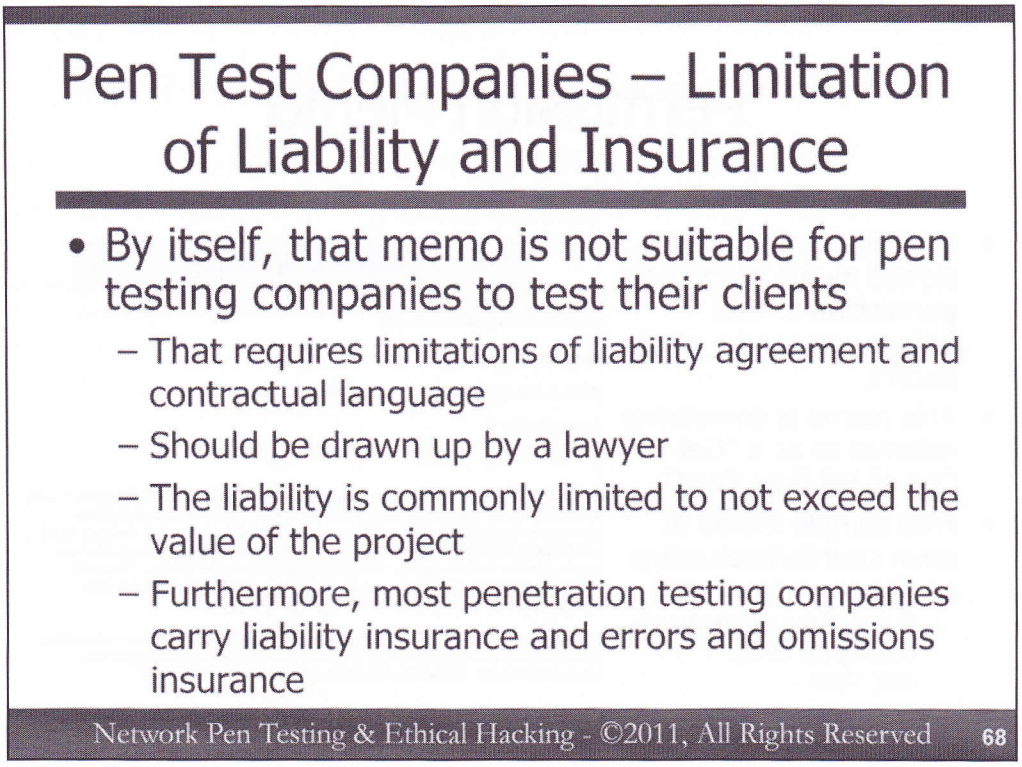
*"The purpose of this memo is to grant authorization to specific members of our information security team to conduct vulnerability assessments and penetration tests against this organization's assets. To that end, the undersigned attests to the following:*

*1)[Insert name of tester], [Insert name of tester], and [Insert name of tester] have permission to scan the organization's computer equipment to find vulnerabilities. This permission is granted for from [insert start date] until [insert end date].*

*2)[Insert name of approver] has the authority to grant this permission for testing the organization's Information Technology assets."*

Have your legal team review, tweak, and approve this language. Then, print it on corporate letterhead and have a chief information officer or similar level of management sign off on it.

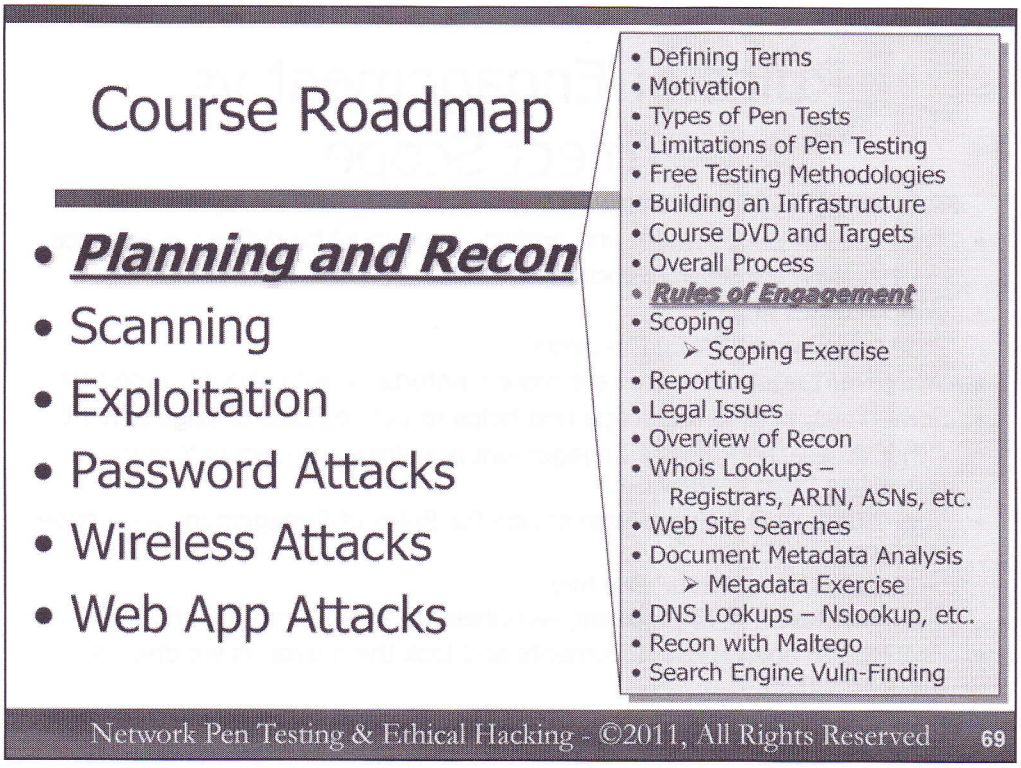
Note that this memo is suitable for employees to test the security of computer equipment owned by their employers.



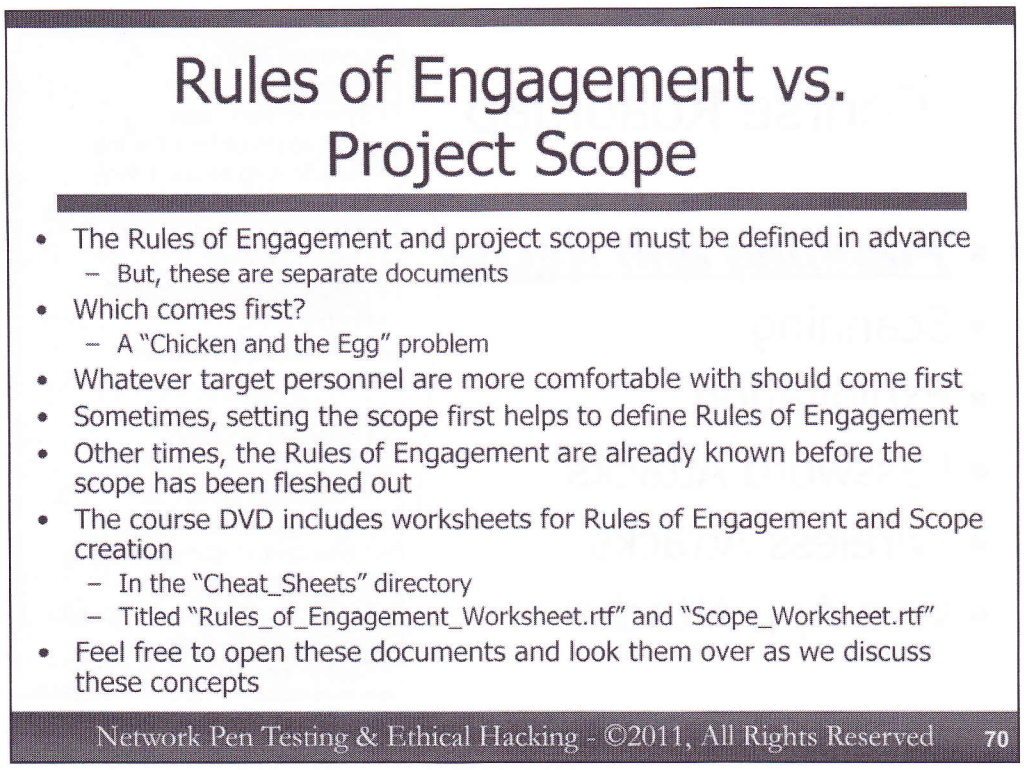

While the permission memo on the previous slide is acceptable for employees testing their employers, it is not, by itself, suitable as a vehicle for a penetration testing company to test their customers' environments. It can act as a starting point for that more comprehensive document. But such third-party penetration testing agreement for client networks must also include a limitation of liability agreement and a contract. These items should be drawn up by a lawyer associated with either the penetration testing company or the client.

Most penetration testing companies include a limitation of liability agreement that caps the liability for any problems associated with the project at the price of the project itself. You wouldn't want one strangely configured target environment or errant test to destroy the testing organization. Thus, if the test costs $ 50,000, the limitation of liability agreement caps the liability at that value.

To help address this issue further, most penetration testing companies also carry liability and errors and omissions insurance in addition to getting the limitation of liability agreement. Some common insurance levels are $ 1 Million, $ 2 Million, and $ 5 Million. In fact, some clients require their testers (or anyone else doing business with the organization) to get such insurance.




Our next topic is Rules of Engagement, a set of practices that must be defined before a penetration test or ethical hacking project can begin. Both the people responsible for the target environment and the testing team must agree on these rules. Without proper Rules of Engagement agreed upon in advance, a penetration test or ethical hacking project could go seriously awry, resulting in devastating consequences for the target organization and the testers, including system downtime, financial damage, reputational damage, personnel firing, and possibly civil lawsuits or even criminal prosecutions.




The Get Out of Jail Free Card, limitation of liability agreement, and insurance help protect the testers legally. But, these documents must be shored up with a carefully decided rules of engagement memo that is documented in advance.

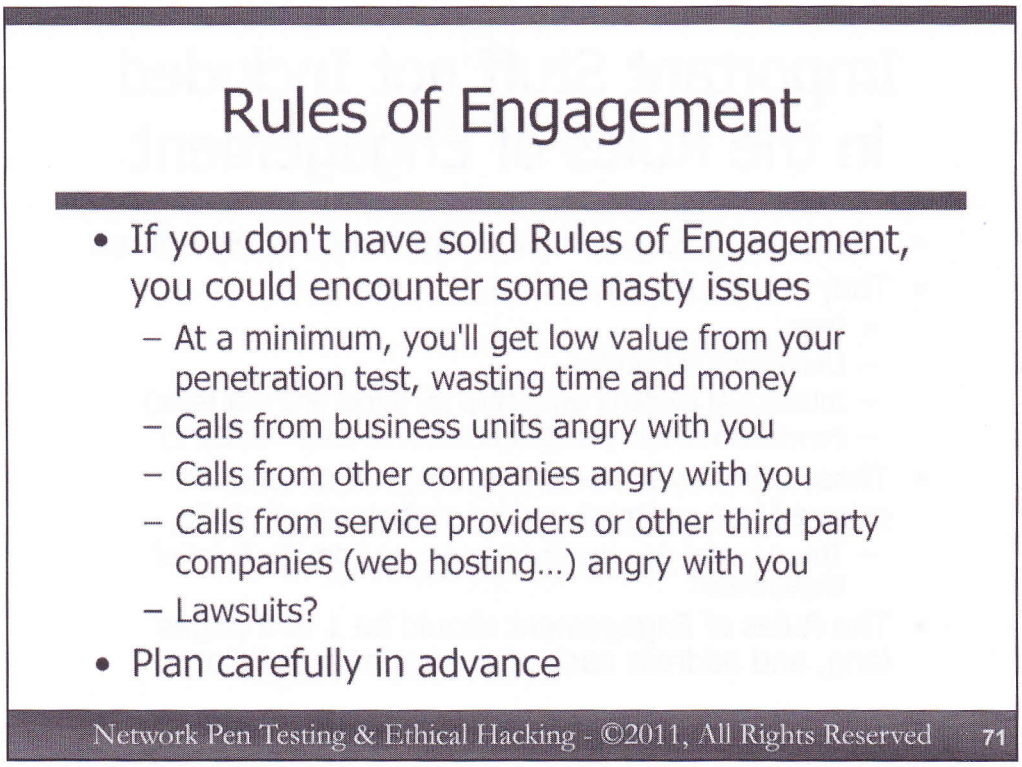
Some testers define the rules of engagement with a client before they devise a detailed scope of the test. That way, the target organization can have in mind the way the test will be conducted to help make decisions about what is in and out of scope.

Others reverse the flow, defining the scope before agreeing to rules of engagement, so that they know what they will test and can then craft the rules of engagement around the given test targets.

Either approach is acceptable – defining the rules of engagement first followed by scoping, or scoping the project first and then defining rules of engagement. The important point is that both issues be covered in advance. In fact, these two phases may be iterative. That is, an initial rules of engagement is discussed, followed by a scoping conversation. Then, when the scope is agreed upon, the rules of engagement may be further tweaked, until the scope and rules line up appropriately.

The course DVD includes worksheets for helping to define the Rules of Engagement and Scope of a penetration test. These documents, in the "Cheat_Sheets" directory of the DVD, are in RTF format. Feel free to open them up on your computer to follow along with the next section of the course as we explore the concepts more thoroughly.

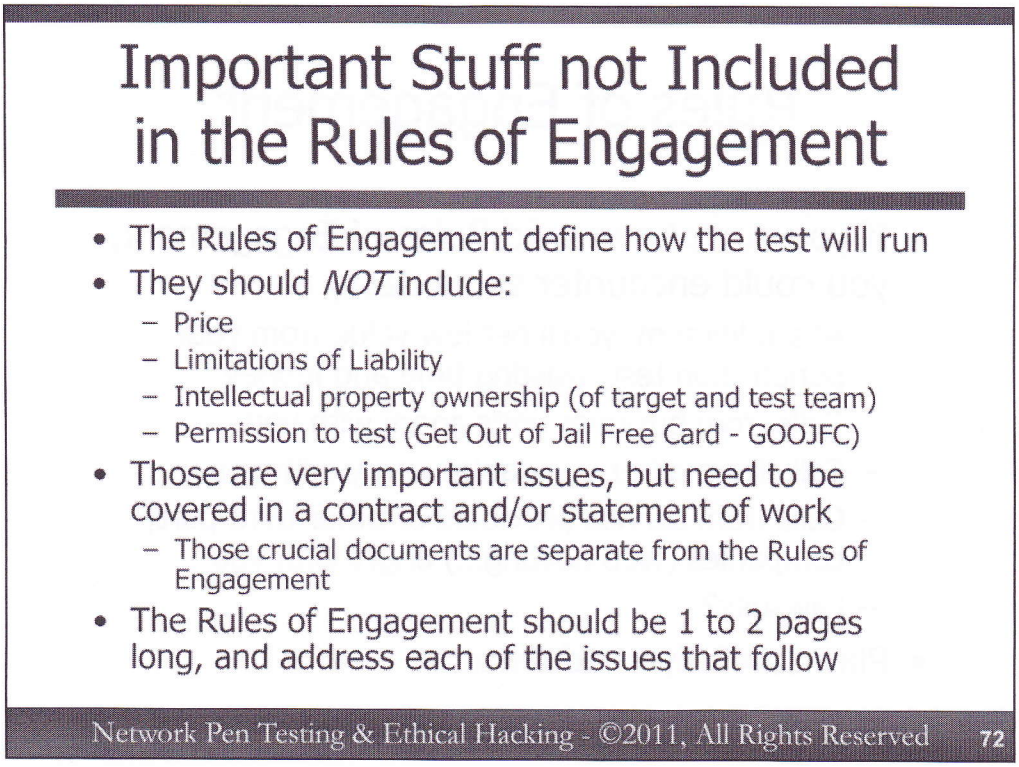
Please consider these worksheets as a starting point, which you can modify or add to in creating your own worksheets customized for your organization's testing.




If you don't nail down written Rules of Engagement in advance, you could run into significant trouble.

The minimum bad news you'll face without good Rules of Engagement is a wasted effort. That is, your penetration test results will be of low value, with little insight into your security stance. You'll have wasted your time and money.

But the consequences could be far more dire than that! Without good Rules of Engagement, you could receive nasty calls from business units, other companies, or even your own service providers complaining that they did not authorize any test of your environment. In effect, someone attacked them, possibly giving them legal standing for a suit against you! Plan carefully in advance to minimize that risk.




The Rules of Engagement are intended to focus on how the test will be conducted, and not to cover all of the details between the target organization and the tester. The Rules should be kept short and focused; we recommend 1 to 2 pages, with pithy sentences dealing with each of the issues we'll discuss in this section.

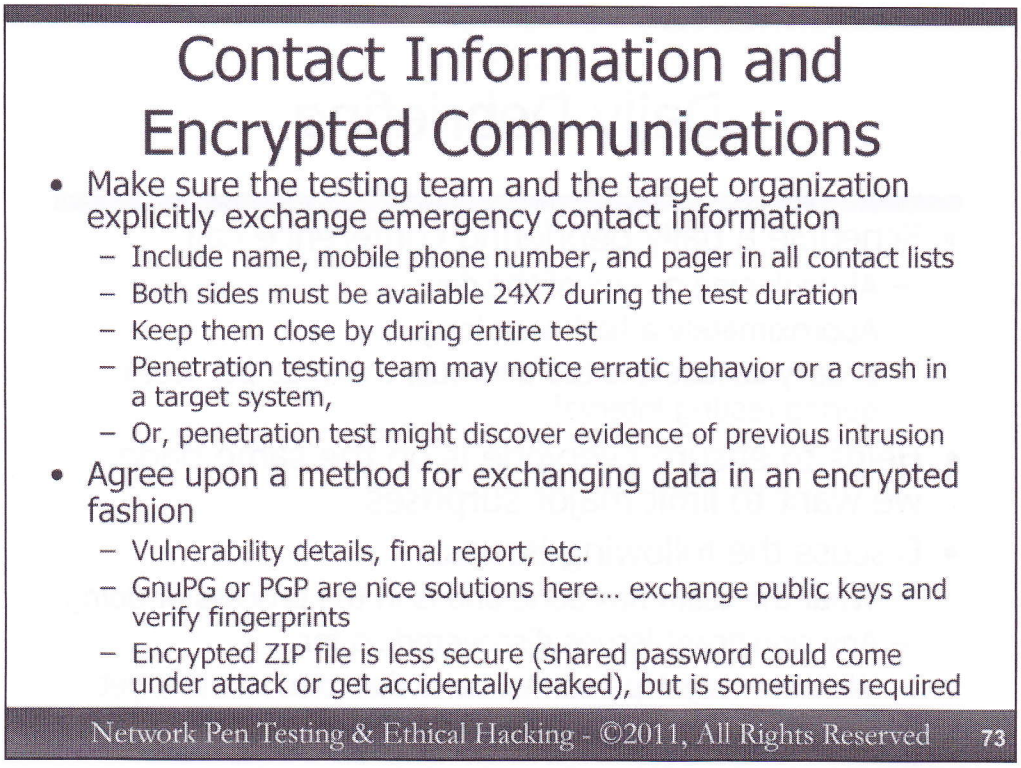
In particular, the Rules of Engagement should not include the following items:

The price of the test service – Such info shouldn't be in the Rules of Engagement.

Limitations of liability – Who is liable and for how much money if a system or service is accidentally disabled, resulting in financial loss? This sticky issue is better handled in the contractual agreement instead of the Rules of Engagement.

Intellectual property ownership – Do the resulting report and the methodology behind it belong to the target organization, the tester, or both? Again, this concept should be included in the contract, not the Rules of Engagement.

Permission to conduct the test – The target organization should explicitly authorize all testing in writing, even if the tests are conducted by in-house personnel. This is the "Get Out Of Jail Free Card", we discussed earlier.

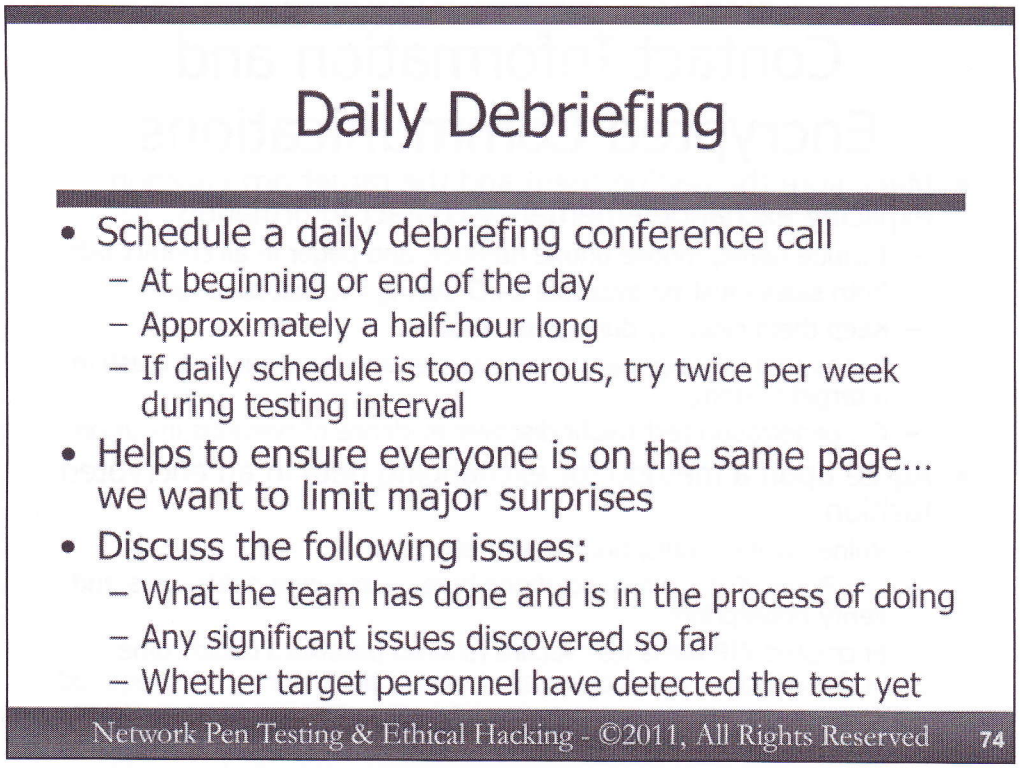



Another crucial element of the Rules of Engagement involves exchanging contact information between the testing team and the target organization. During a test, the testing team may crash a target, discover an urgent, high-risk vulnerability, or find evidence of a previous intrusion. In such cases, they may need to contact personnel immediately in the target company to report the issue.

Sometimes, the target company personnel need to reach the testing team to verify that an observed attack is coming from the testers. What if an evil attacker starts hitting the target at the same time that the penetration testers begin their work? The target organization needs to be able to contact the testing team 24X7 during the duration of the test. Therefore, both sides need to be available around the clock during the test.

Once you've identified points of contact, be sure to agree upon a secured method of communication regarding vulnerability details and the final report. The penetration testing team will be handling some sensitive data, which you may need to e-mail to the points of contact among target system personnel. You do not want to send this information in clear text. Instead, choose a suitable encryption solution. The free, open source Gnu Privacy Guard or commercial PGP are good solutions here. If you choose either, make sure to exchange public keys and verify their fingerprints.

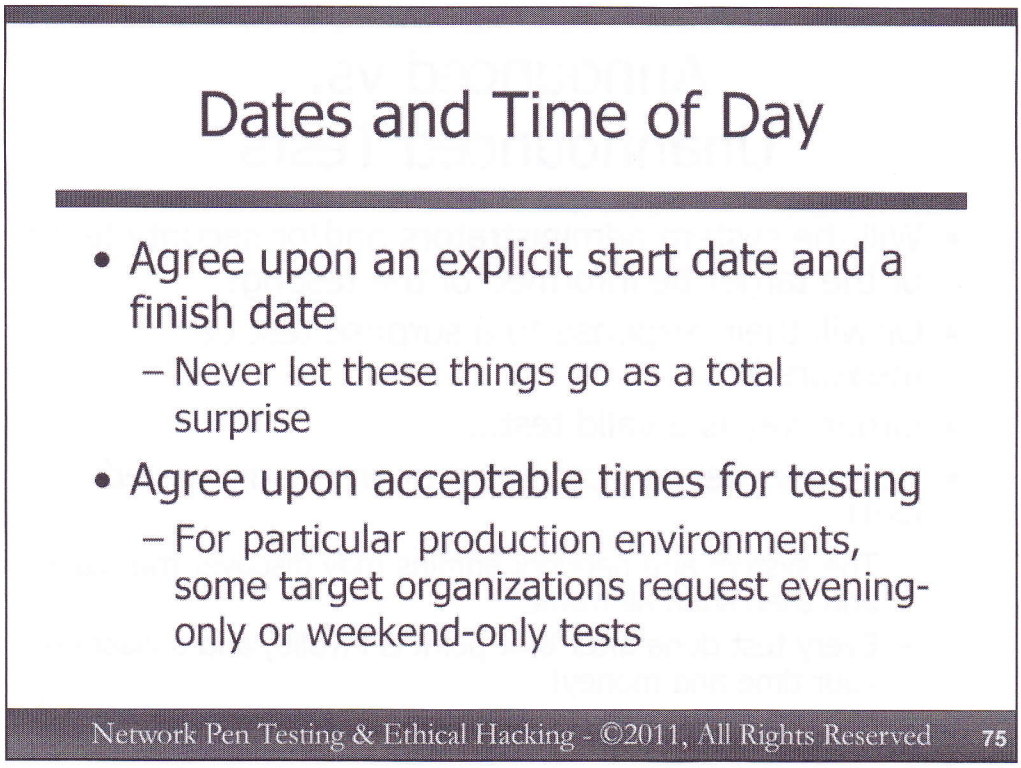
Encrypted ZIP files are less secure, because the shared password used to protect the symmetric key in the ZIP file could come under attack. A bad guy could steal the shared password, or it could leak if it is shared among too many people. For example, sometimes, target system personnel may be tempted to send the password for the ZIP file in e-mail, potentially exposing it. Alternatively, an attacker who gets the ZIP file itself could stage a brute-force attack against the password for the ZIP file. However, even though encrypted ZIP files are less secure, they are sometimes the only option we have, given that other encryption solutions are forbidden or are not accessible to target system personnel.




Another element that we find very useful to define in our Rules of Engagement is to require a daily debriefing conference call. At the beginning or end of each testing day, schedule a brief session between the testing team and one or two representatives of the target organization. These calls help to make sure that everyone understands the progress of the test and to identify any issues early on.

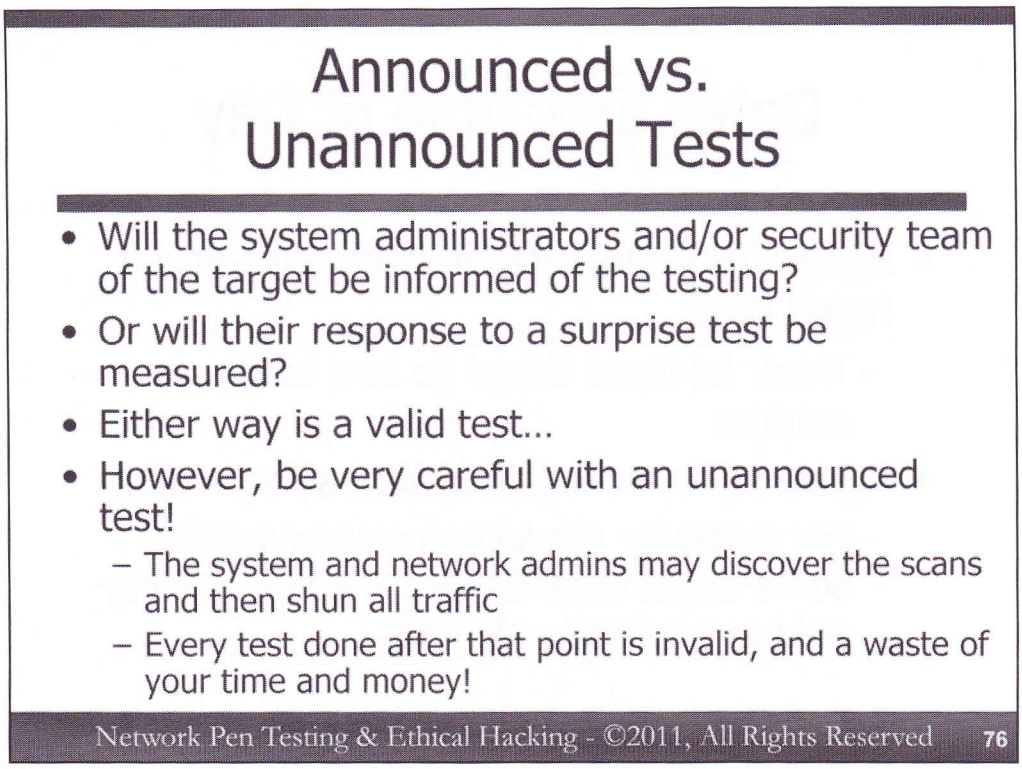
During the debriefing, the team should discuss what the team has done so far, and what they plan to do next. Additionally, any major findings can be reviewed. Finally, the target organization can confirm whether their detection mechanisms (IDS, IPS, log review, etc.) have been triggered by the test.

If a daily call is too onerous given the busy schedules of target environment personnel, consider conducting a debriefing call twice per week during the duration of the test.




Of course, the Rules of Engagement should explicitly state the start and end dates of the test, as well as valid test times.

Some penetration tests and ethical hacking projects run around the clock, while others with more sensitive infrastructures and business needs require testing during off-hours or weekends only. When such off-hours tests are conducted by a third-party company, there is typically a slight additional charge for such off-hours testing, but it is usually quite reasonable. Also, such limited testing time windows will require a longer total duration to complete the test.



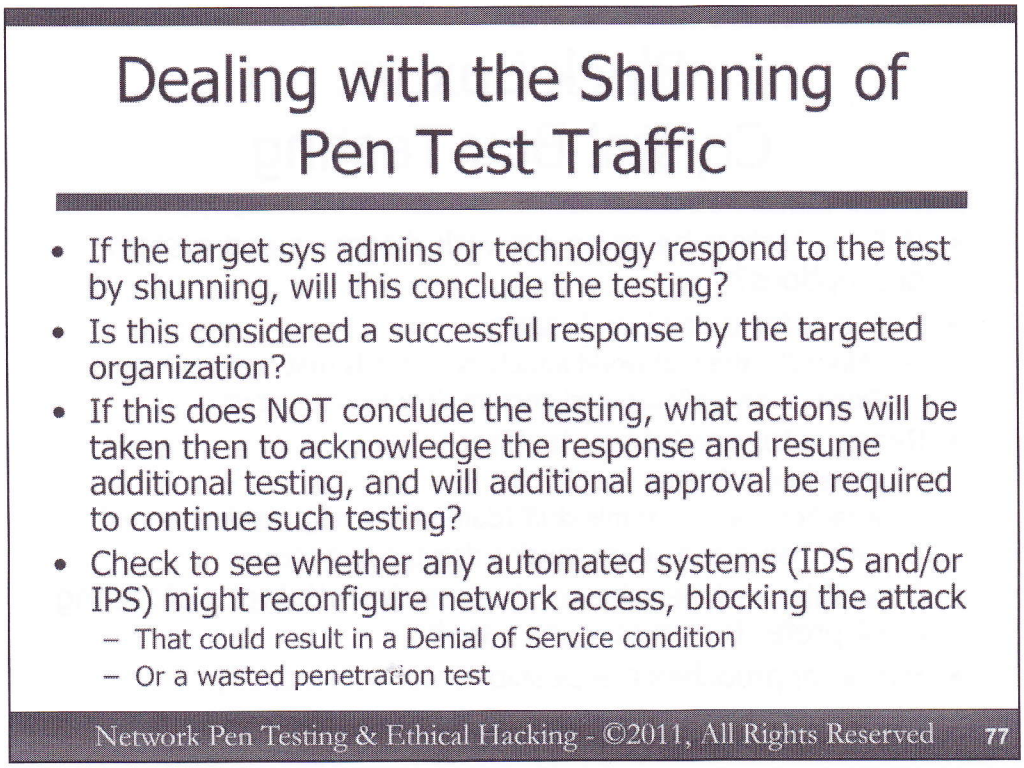

Here's another point that can cause controversy with some target organizations – should the penetration test be announced to the people responsible for running the target infrastructure? Or, will the test be a surprise to them?

Performing an unannounced test does have some advantages. First off, if any of your admins are purposely running backdoors or side-businesses on your servers, the testing team might find them during an unannounced test. If you announce the test, the admins will likely temporarily shut off their shifty activities during the test duration. Some penetration testers have discovered deliberate, sys-admin-planted backdoors during an unannounced test. Other testers have identified pay-for-porn services on target web servers that were being run by the web administrator. Such findings are important results of a penetration test.

Secondly, an unannounced test gives you a chance to evaluate the detection and response mechanisms and processes of the target organization. Does information flow properly through the organization concerning a computer attack?

However, unannounced tests also have a downside. The system or network administrators might detect the attack and start shunning the traffic, blocking it from reaching the target systems. Then, any testing activities after the shunning is applied are invalid, a waste of time and money. If you do perform an unannounced test, make sure the system and network administrators are watched to verify that no shunning occurs.

Also, to prevent controversy about such tests, make sure that target personnel know that they could occur at any time, and are just a normal part of the way the organization does business.




Another aspect to take into account with shunning involves what to do afterwards. If manual or automated shunning occurs, will that conclude the test? And, if so, was the test successful or not?

Also, if and when shunning is engaged, make sure the target organization contacts the testing team to let them know, and to determine which of the three options on the previous page you will utilize.

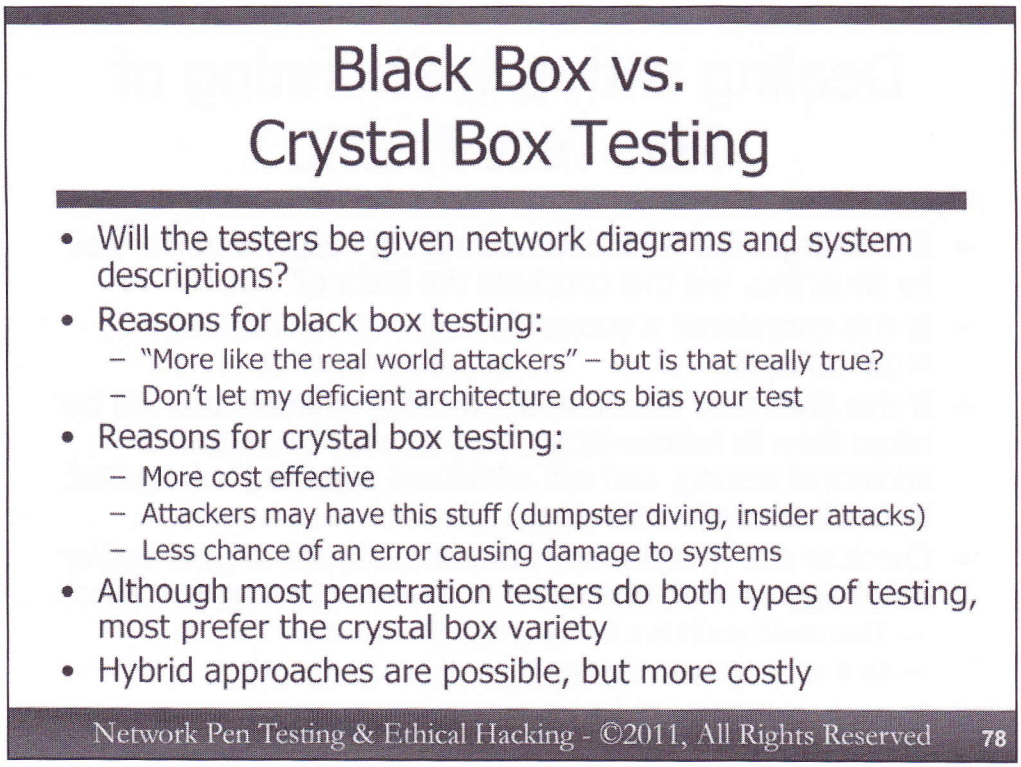
Some organizations have deployed automated shunning functionality. If an IDS spots an attack, it could reconfigure a firewall to automagically block it. Such functionality could lead to a denial of service attack, perhaps even inadvertently by the penetration testing team.

The Rules of Engagement must explicitly take into account any automated shunning functionality the target organization may have. Auto-shunning that is too widely applied could turn a run-of-the-mill penetration test into a Denial of Service attack. Or, narrowly focused blocking would just render much of the test useless. To avoid these conditions, you should do one of the following, and document it in the Rules of Engagement:

Have target organization personnel disengage shunning functionality

Include an exception for the source addresses of the testing organization, or

Announce the test, and get an alert when shunning starts so that it can be manually deactivated, allowing traffic from testers through

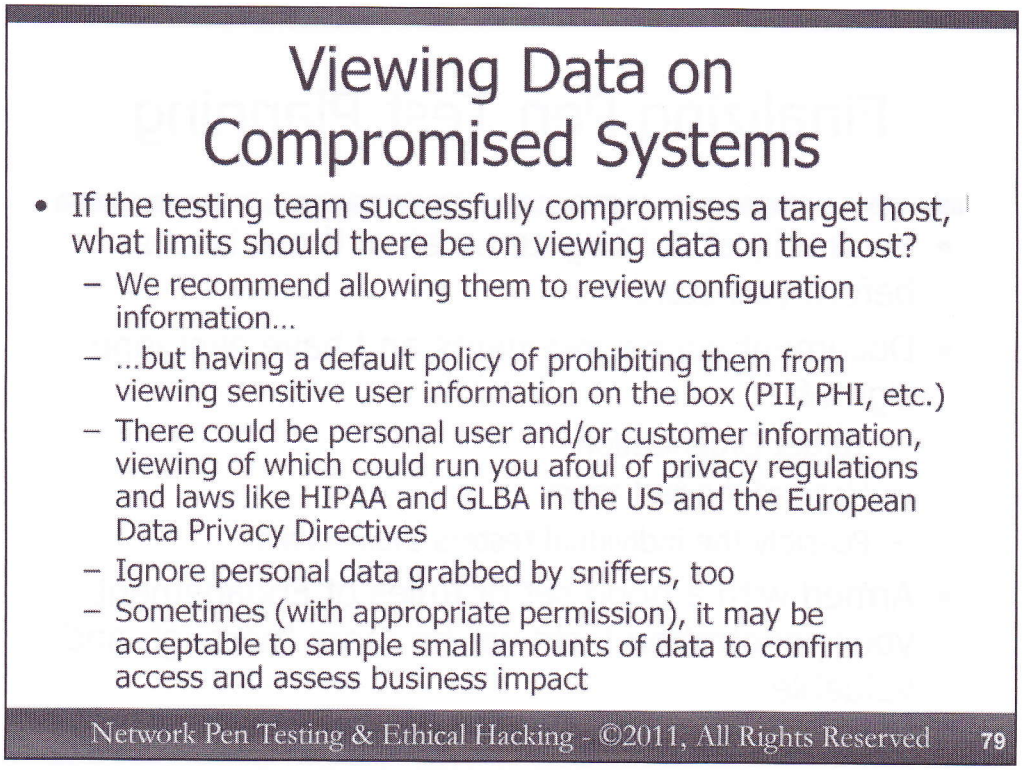



Here's another point of some controversy. Should the testing team be given a copy of the network diagram, listing hosts, topology, operating systems, and so on? Such crystal-box testing allows testers to see inside the target before launching any scans. Alternatively, the test might be a black-box engagement, where the testing team is only given a domain name and is expected to figure out the network topology and targets through reconnaissance and scanning.

The reasons some organizations say that they opt for black-box testing is that it is, "More like what a real-world hacker would see in our environment." Unfortunately, that's not necessarily true. A real-world attacker might have a picture of your network architecture, stolen from a dumpster, faxed from a duped employee, or swiped by a temporary employee. Working from a network diagram lets the penetration testing team analyze a worst-case scenario from the target's perspective.

A better argument for black-box testing is that some organizations are worried that a network diagram may bias the testers so that they miss items. Many network diagrams have significant limitations or errors, missing a lot of detail and sometimes overlooking hosts or whole networks.

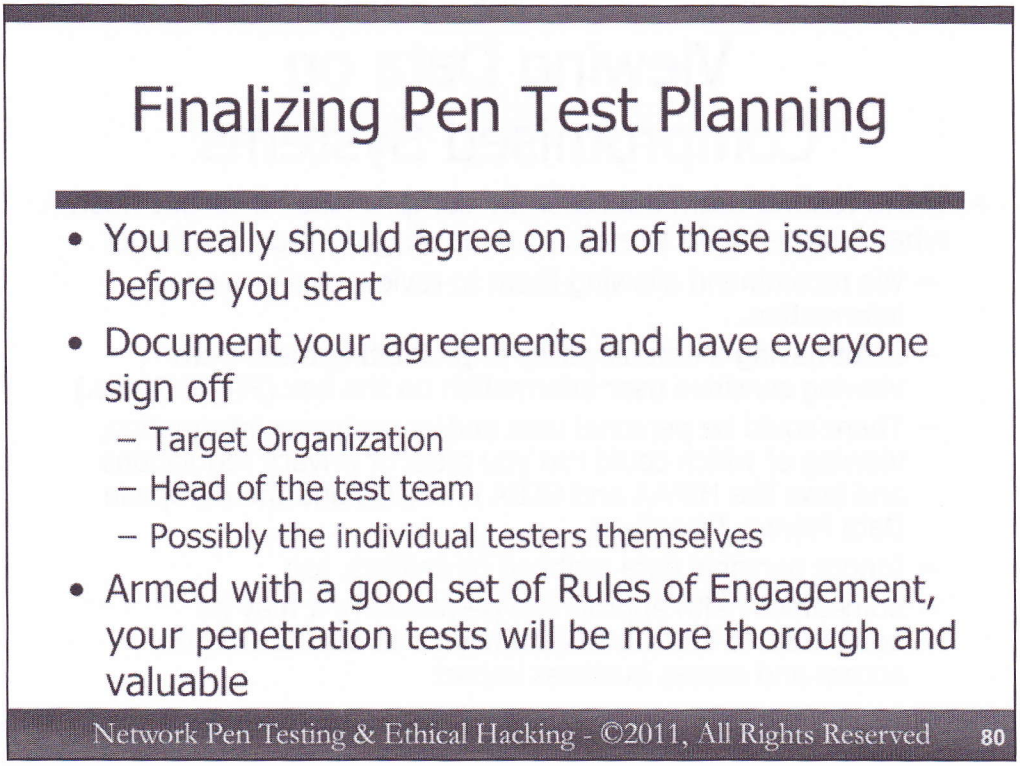
In the end, crystal-box testing tends to be more cost effective, because the attackers can more quickly perform their reconnaissance and scanning. There is also a lowered chance of an error causing damage to an out-of-scope system. While most penetration testers perform both black- and crystal-box testing, the latter approach is usually recommended. Of course, you could take a hybrid approach, where the testing team starts out with nothing more than a domain name and performs detailed recon and scanning. Then, after a week or so, the target organization provides a detailed diagram for further testing and analysis. When performing such hybrid testing, explicitly label in your report the elements the testers discovered themselves, and which items were given to them by the target organization. Hybrid tests usually take a little longer, and therefore cost somewhat more.



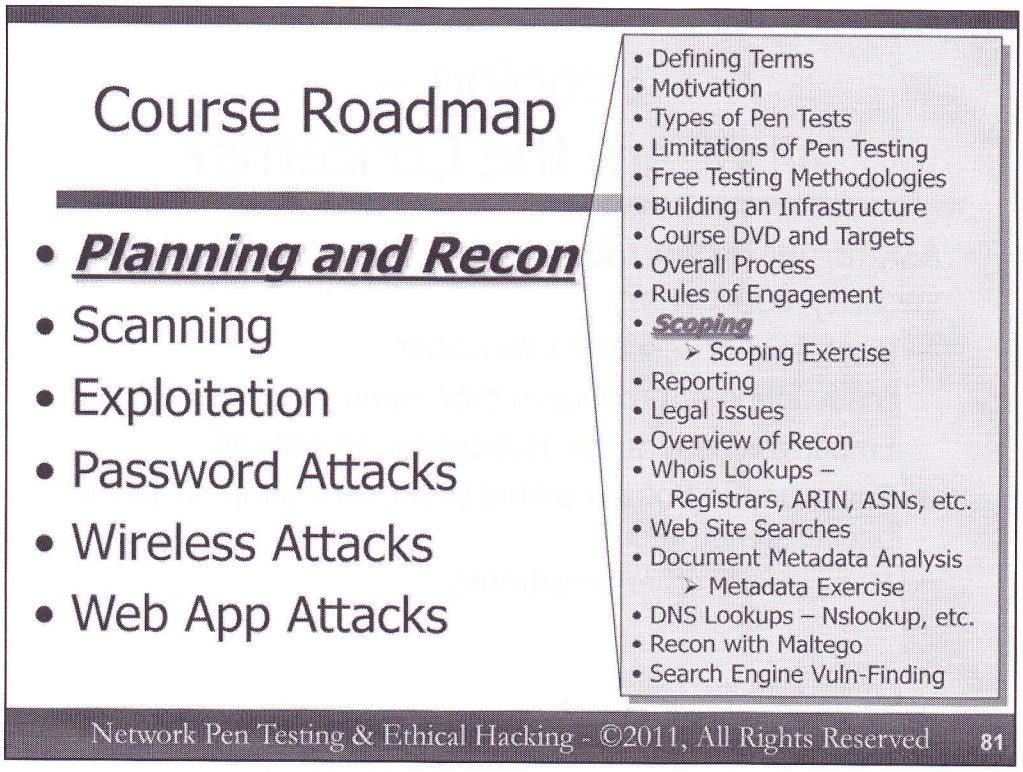

Here is a crucial issue to emphasize in your rules of engagement – If the team successfully penetrates a target, should they avoid viewing data on the target? Think about it… you might have sensitive employee or customer data on the target. A penetration tester has just gotten root on the box, and might be able to view all of that data. Only your Rules of Engagement prevent that from happening, so make sure they are clear on this topic. For some kinds of information, including healthcare and personal financial data, only duly authorized representatives with a need to know are allowed to see such information, under various regulatory initiatives like the Health Insurance Portability and Accountability Act (HIPAA) and the Gramm-Leach-Bliley Act (GLBA) in the US, and the European Data Privacy Directives.

We recommend that your default rules of engagement allow the testers to view configuration data from a machine, but to avoid looking at any customer and user data on the machine. Also, if sniffers are used on the compromised box, make sure the team explicitly documents the fact that any personal data captured from the network will be ignored by the testing team.

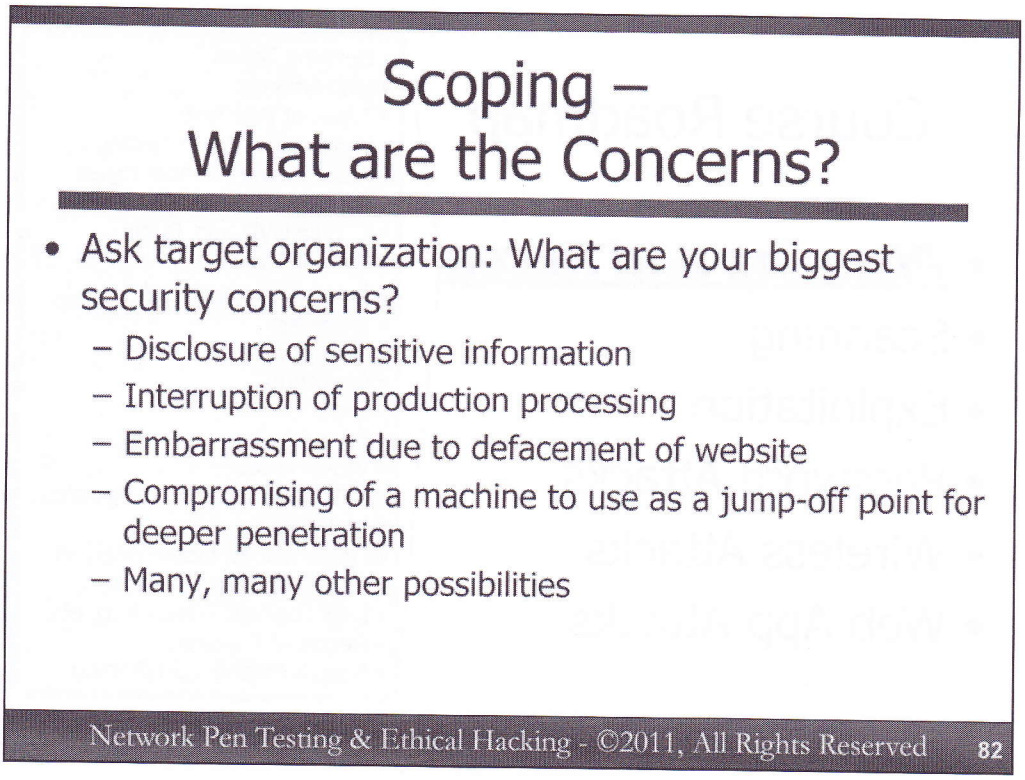
While the default policy is usually to avoid accessing sensitive user data, in some tests, it may be appropriate to sample small amounts of sensitive user data to confirm access and assess business impact. Such access should be done sparingly, and only with written permission from target system personnel. Also, try to focus more on counting the number of sensitive records you can access (based on file length, number of lines, number of files, number of rows in a database table, or other metric based on the type of data and the form of access you've gained) than on retrieving the sensitive content itself.




To conclude this segment of the course, make sure you have a solid, signed set of Rules of Engagement before you embark on any penetration test or ethical hacking project. You need to make decisions about these crucial issues in advance. If you do, you'll have a far more high-valued ethical hacking and penetration testing experience.




We'll now look at the scoping process, determining what should be tested and what should not be tested. In addition to determining individual target systems and networks, this scoping process will also look at some types of testing that may or may not be in scope.



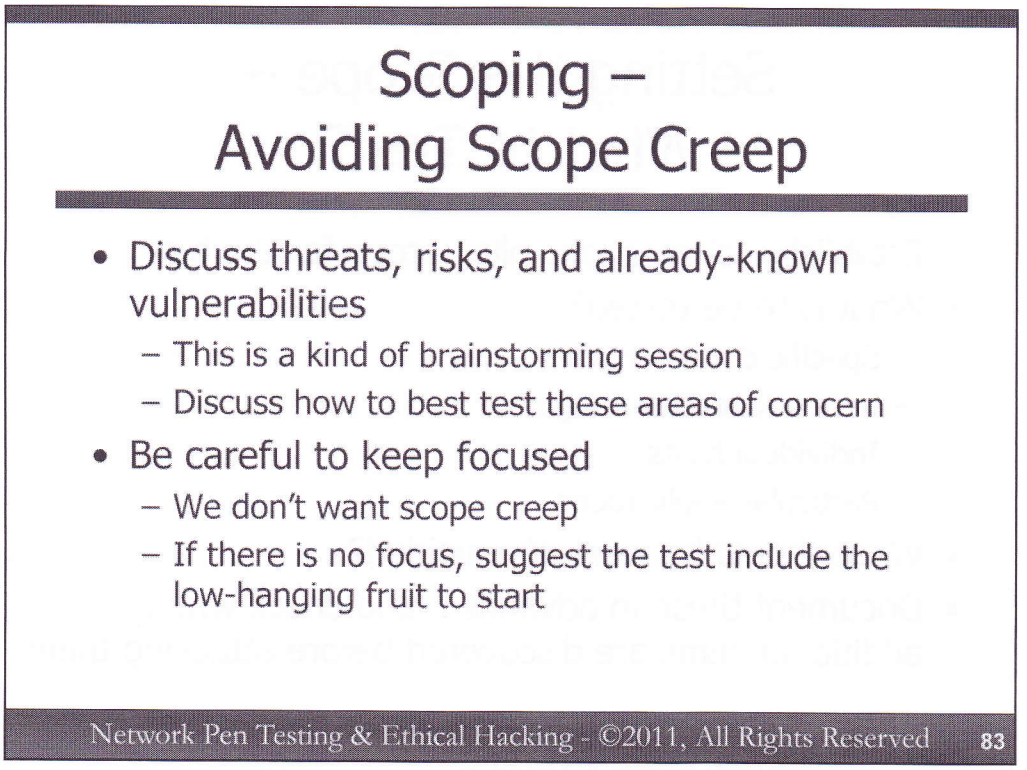

To start out a scoping conversation, ask members of the target organization about their biggest security concerns. What worries them? Some example concerns could include:

*Disclosure of sensitive information:* Some organizations, such as financial services institutions, healthcare companies, educational organizations, and others store information that, if stolen by an attacker, would require public disclosure to comply with various laws. Furthermore, compromise of this information could result in the loss of customers, costing the victim organization dearly.

*Interruption of production processing:* If certain critical systems crash, the target organization could be severely impacted, going out of business for the time it takes to restore the machines. Manufacturers, financial institutions, law firms, and many others are often concerned about this threat.

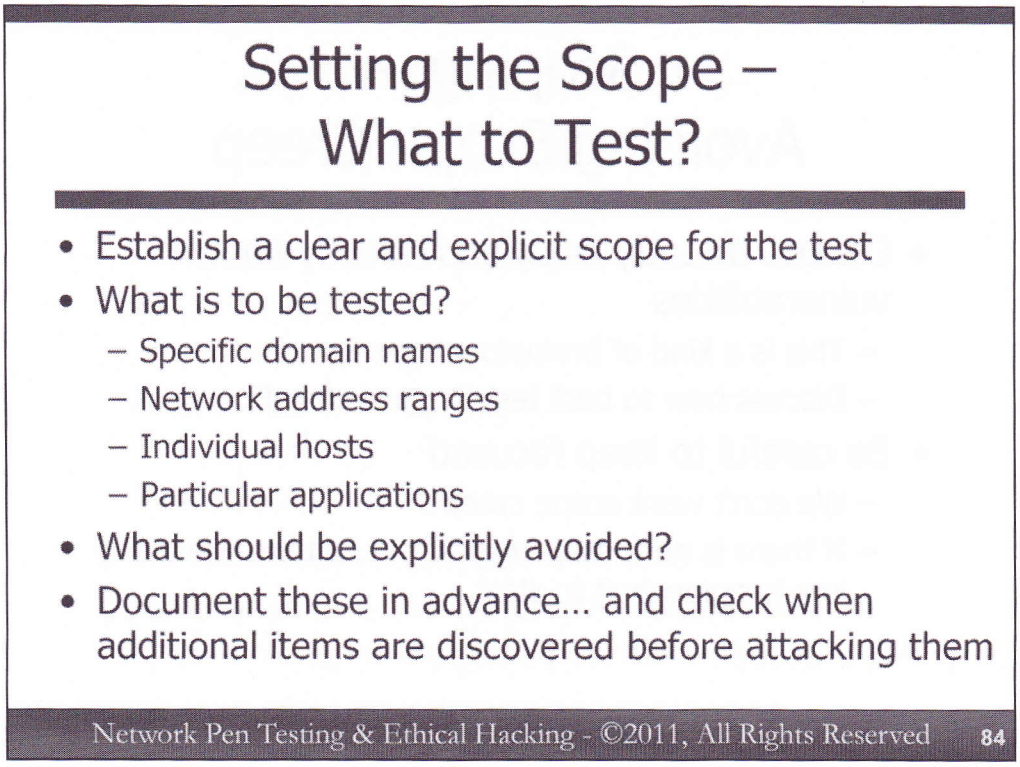
*Embarrassment due to defacement of a website:* Some organizations would be badly impacted if their website is altered in an embarrassing way. Advertisers, government agencies, and others would be severely hurt if the public and their customers no longer trust them.

*Compromise and pivot:* If an attacker takes over one machine on the target organization's perimeter and uses it for deeper access into the organization's network, the attacker could cause significant damage. Almost every type of organization faces this issue.




Note that the list of concerns on the previous slide is not comprehensive because many different organizations have widely different concerns, and some of them may be very specialized based on their business. But, this list should get the conversation started, allowing the person scoping the project to work with the client to brainstorm the most significant issues. Determining the primary concerns up front can help narrow the focus of a test. Discuss threats, risks, and already-known vulnerabilities with the target organization's representatives.

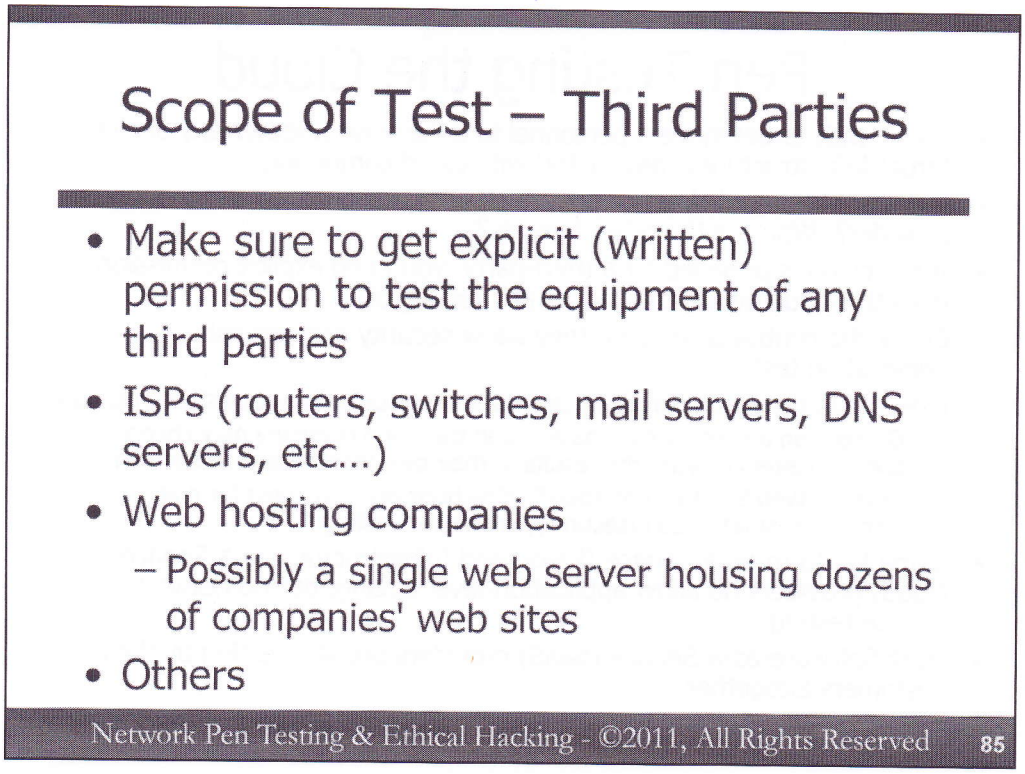
It is vital to focus this conversation to determine exactly what needs to be tested. The last thing a tester wants is a blurry scope that could lead to scope creep. With scope creep, a misunderstanding of what should be tested leads the target organization to add more systems, target networks, target types, and types of testing to the test as it proceeds, a dangerous and costly proposition for a tester. If determining the focus is difficult because there are so many areas of concern, the scoper could suggest that the test focus on low-hanging fruit, those systems and networks that are of most concern and easiest to reach or compromise.




One of the most important elements to include in the project scope is a succinct statement of what is to be tested. Spell out explicitly those domain names, network address ranges, individual hosts, and particular applications that are included in the test.

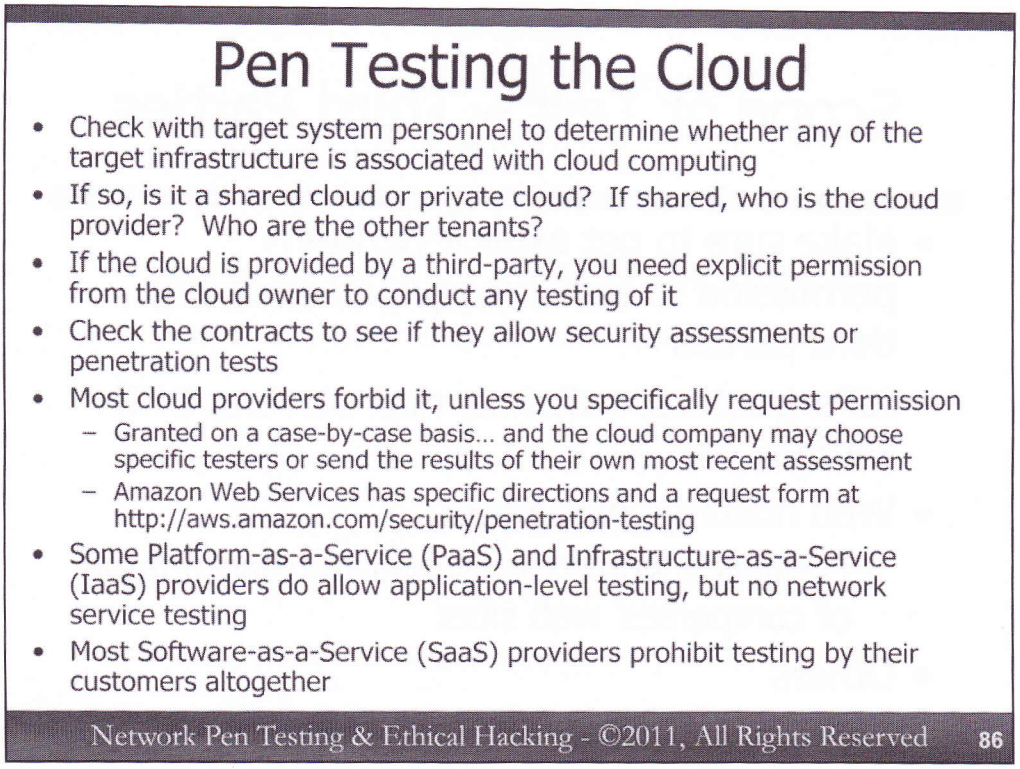
Also, if there are particularly sensitive elements of the organization that should not be tested, explicitly spell out that list of off-limits machines.

While the team is performing the test, they may discover additional systems within network address ranges or domain names that weren't considered in the original formulation of scope. Make sure the Rules of Engagement direct the testing team to check with the target organization before testing any other machines outside of the original scope that are discovered through reconnaissance or scanning after the test starts.




Many major organizations today use at least one third-party to manage at least part of their computing and network infrastructure. Some companies outsource this altogether, with their infrastructure actually being owned by the third party itself. Examples include Internet Service Providers, whose routers, switches, mail servers, and DNS servers may fall into the scope of a test. A particularly significant concern is associated with third-party web hosting companies. Sometimes, a single web server may house the web presence for a dozen or more companies. If one of those companies contracts a penetration test without letting the web server owner know about it, there could be significant trouble!

If any third-party owned or managed systems are included in the scope, make sure to get written permission from these parties before the test begins. The target organization is responsible for getting this permission, and the testers are responsible for making sure the target organization does this.

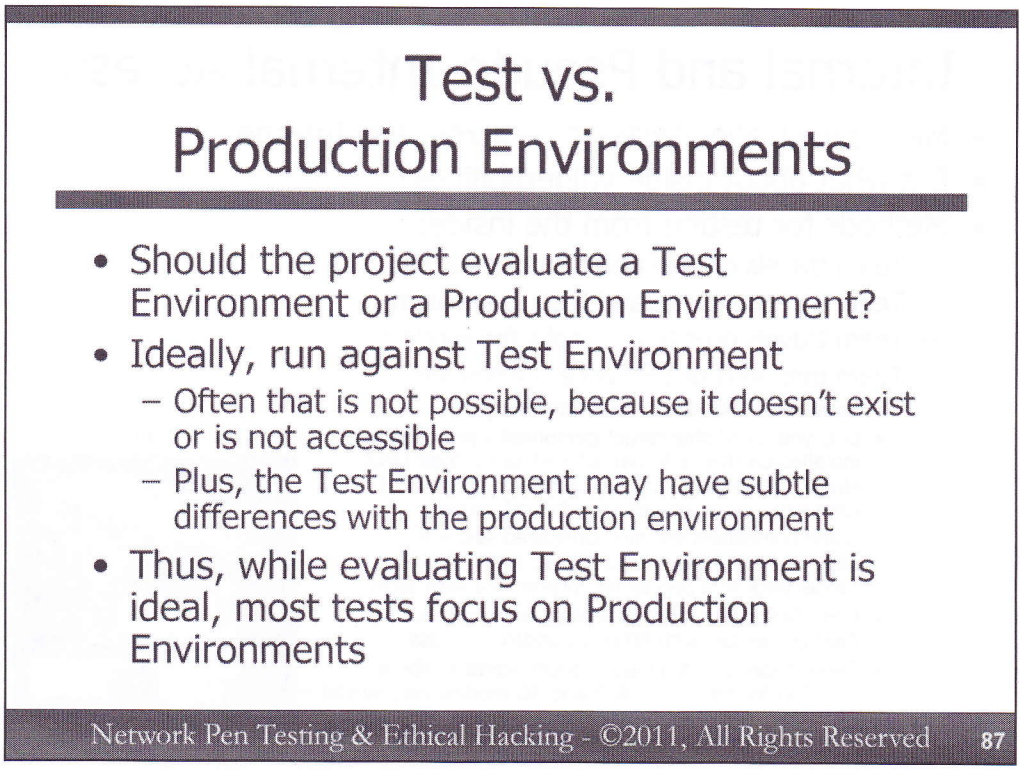



Cloud computing deployments are rapidly increasing, and penetration testers are sometimes called upon to test cloud environments. Be very careful in getting permission to test such environments.

Ask target system personnel whether the cloud is shared among multiple enterprises acting as tenants on a third-party-provided infrastructure, or whether this is a private cloud run only by the given target organization. If this is a shared cloud operated by a third-party, you must get permission to test from the cloud provider.

For third-party provided clouds, check the contract to see if it allows or explicitly forbids security assessments or penetration tests. Most cloud providers explicitly forbid such tests, unless you specifically ask for and receive written permission, which is granted on a case-by-case basis. Some cloud providers will simply provide the results for their most recent assessment to their customers upon request, or choose a specific set of testers to honor a client's request for a needed test. Amazon Web Services has been quite forward-looking here, with directions for penetration testers and a specific AWS Vulnerability / Penetration Testing Request Form available on-line.

We've seen many cloud providers that offer underlying virtualized systems, in a "Platform-as-a-Service" (Paas) or "Infrastructure-as-a-Service" (IaaS) arrangement, allow customers to perform application-level testing (such as web app testing), but prohibit them from testing for underlying network and listening service flaws. Many "Software-as-a-Service" (SaaS) providers prohibit customers from conducting their own tests whether at the network/service layer or the application layer. Make sure you get appropriate written permission before launching any tests.




One of the important issues to discuss before launching an ethical hacking or penetration testing project is whether to evaluate a Test / Quality Assurance Environment or a Production Environment. Some organizations have full-blown test environments where changes and business functionality are evaluated before being applied in production. Some organizations have a third environment as well, often called Development, where new code is created and scrubbed before being moved into the Test Environment.

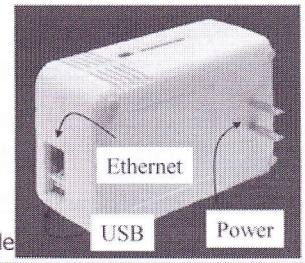
Ideally, we'd like to conduct our penetration tests and ethical hacks against a Test Environment. After all, it was designed for testing. Any system or service crash conditions that are discovered in a test environment shouldn't impact production processing, which is certainly good news.

Unfortunately, we often cannot hit this ideal. Most organizations do not have a fully functional Test Environment, forcing us to conduct penetration testing and ethical hacking against the Production Environment. Even for those organizations that do have a Test Environment, these systems often have both major and subtle differences with the Production Environment that might invalidate our findings.

Thus, while testing against the Test Environment is ideal, the vast majority of real-world penetration tests and ethical hacking exercises occur against production environments. We should strive for Test Environments, but face this reality of most testing.

## Internal and Pseudo-Internal Access

- Many penetration tests occur across the Internet
- But what about inside vulnerabilities?
- Methods for testing from the inside:
  - Team travels on-site and is granted access
  - Team travels on-site and tries to sneak in
  - Team travels on-site and looks for wireless
  - Team gets VPN or SSH access internally
    - Requires lots of coordination and support
    - But, you could ship target personnel a pre-configured laptop, ready to go, installed on their internal network or on their DMZ, which uses a cron job to establish a reverse SSH tunnel to you to implement inbound access
    - Some companies sell pre-configured systems designed to maintain a persistent reverse SSH tunnel back to a pen tester, pre-loaded with tools
    - Form factor of a large wall plug, running Linux on ARM processor, with Ethernet and/or wireless
    - Pwnie Express' PWN Plug product starts at about US $250 for Ethernet. Wifi and 3G models also available

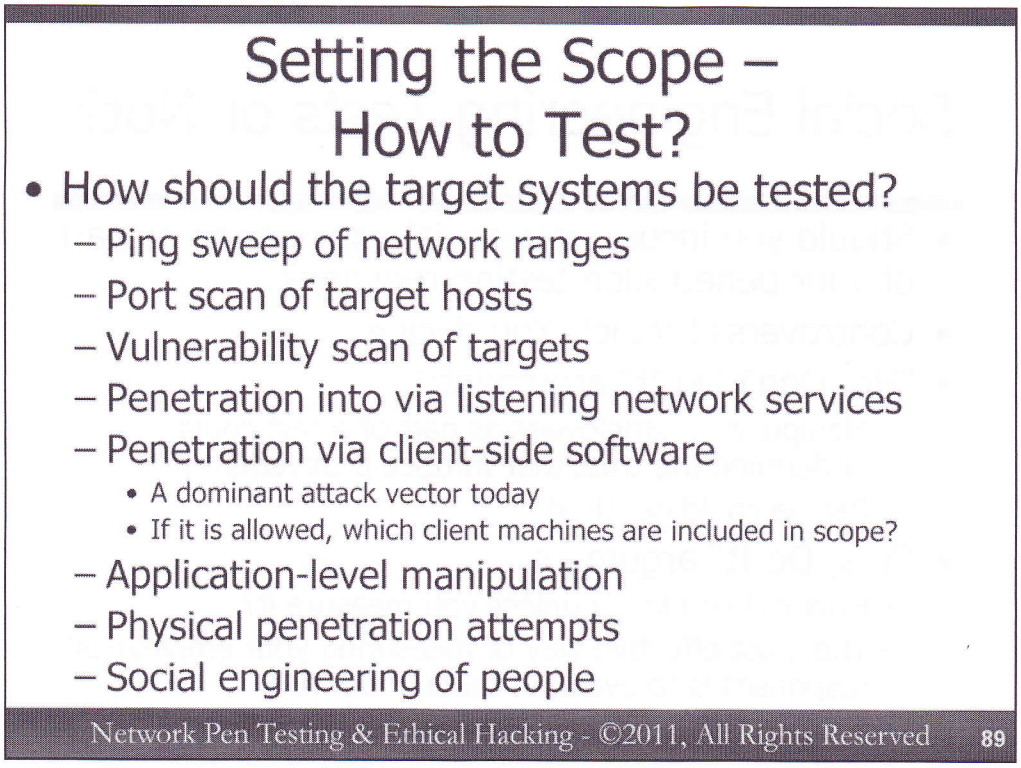





The majority of penetration testing occurs across the Internet against publicly available servers and network equipment. However, what would an inside attacker see? To get this view, the penetration testing team would need inside access to scan and explore the internal environment. There are many ways to grant such access, such as traveling on-site, attempting physical break-in, looking for unsecured wireless LAN access, or using a VPN for remote internal access across the Internet. A particularly promising approach involves sending target personnel a pre-configured laptop set up with a cron job that periodically tries to make a reverse SSH connection across the Internet to one of your systems, setting up an SSH tunnel. Then, you can use that outbound SSH access to gain inbound access to your preconfigured machine and the rest of the network. This approach, of course, requires the organization to allow outbound SSH tunnels, which is typical in many organizations today.

Some companies sell penetration testing systems pre-configured to ship to target system personnel so that they can easily hook the machine up in their network, granting remote access to penetration testers. Many of these systems have the form factor of a large wall plug, with an Ethernet port. They tend to run Linux on an ARM processor, loaded with tools and the ability to create a reverse SSH tunnel back to the penetration tester who configures the system. Some have wireless interfaces as well, looking for wifi or even 3G. A company called Pwnie Express offers their PWN Plug product, which starts at about US $250 for the Ethernet model, which also includes a USB interface to provide additional storage or to move tools to the system via a thumb drive. Wifi and 3G models are also available from Pwnie Express, but cost a bit more.

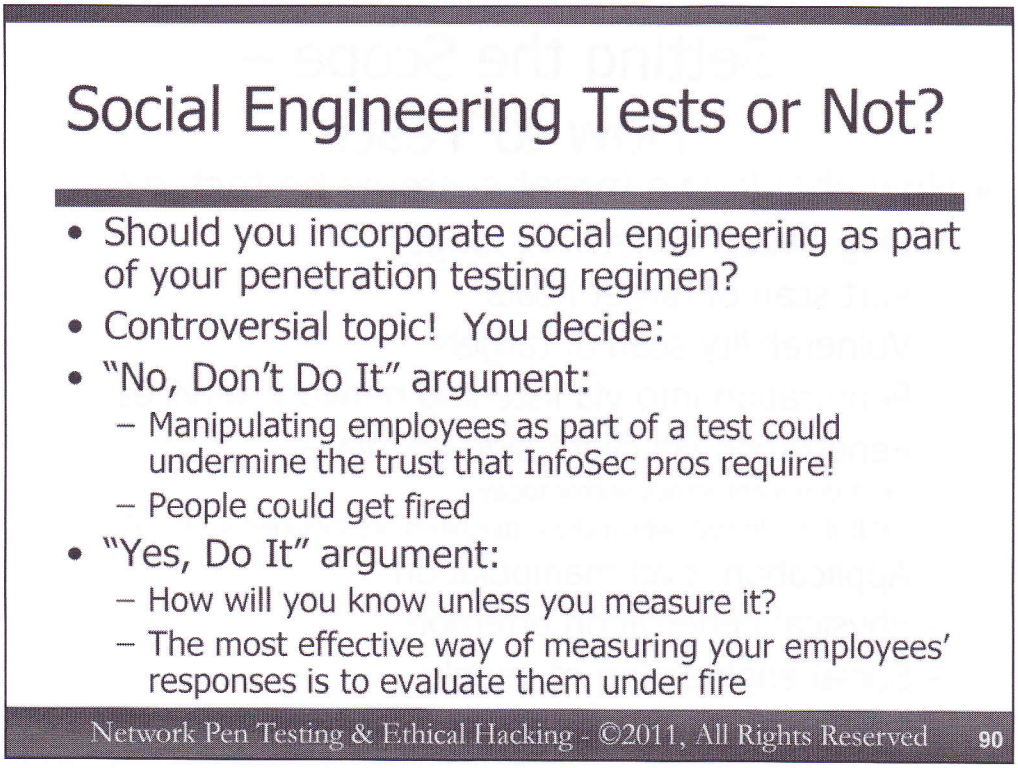
Make sure the scope clearly spells out how you will do internal testing, if at all.




Beyond what should be tested, the scope should specify the level of testing that should occur. Will the test merely be a network scan for targets and vulnerabilities, which would include a ping sweep, port scan, and vulnerability scan? Or, should the testers go further and actually penetrate the target systems, getting access (such as a remote shell) on the targets if possible? If penetration is allowed, should it focus on listening network services, or will client-side software exploitation be allowed as well? Client-side software exploitation is a dominant attack vector today. If client-side exploitation is allowed, which client machines will be included in the scope?

Will the test include any application-level or client-side web component testing? Will it include physical penetration attempts, with the team trying to walk into an environment? Should social engineering, which involves duping human beings typically via the telephone, be attempted?

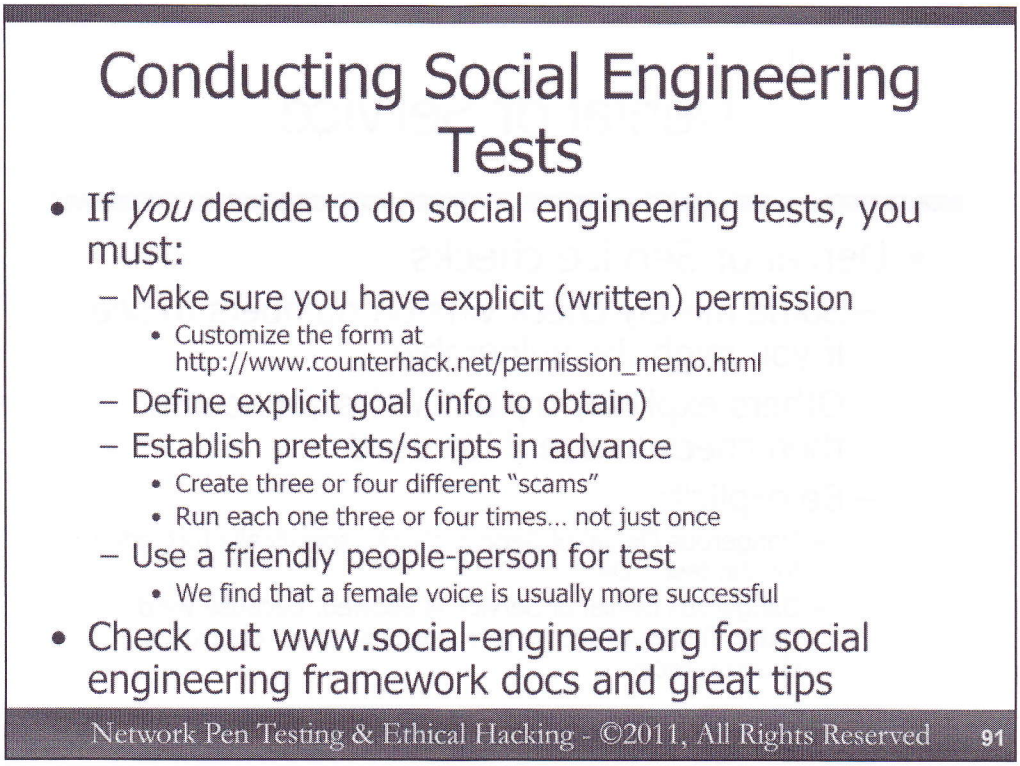
Your Rules of Engagement should specify each element on this list that is included in the scope.




An area of considerable controversy in the information security industry involves doing social engineering penetration tests. This unique form of a penetration test usually involves calling employees on the phone and trying to get sensitive data from them using social engineering techniques. If your employees succumb to such a test, how could they possibly react properly to a real social engineering attack?

Some security professionals say that you should never conduct such a test. They argue that manipulating employees, even under test circumstances, would taint the security group in the eyes of the rest of the organization. Additionally, if the results of such a test are not carefully handled, employees discovered not to be following processes could be fired. That's not a stellar way to improve the reputation of your security organization!

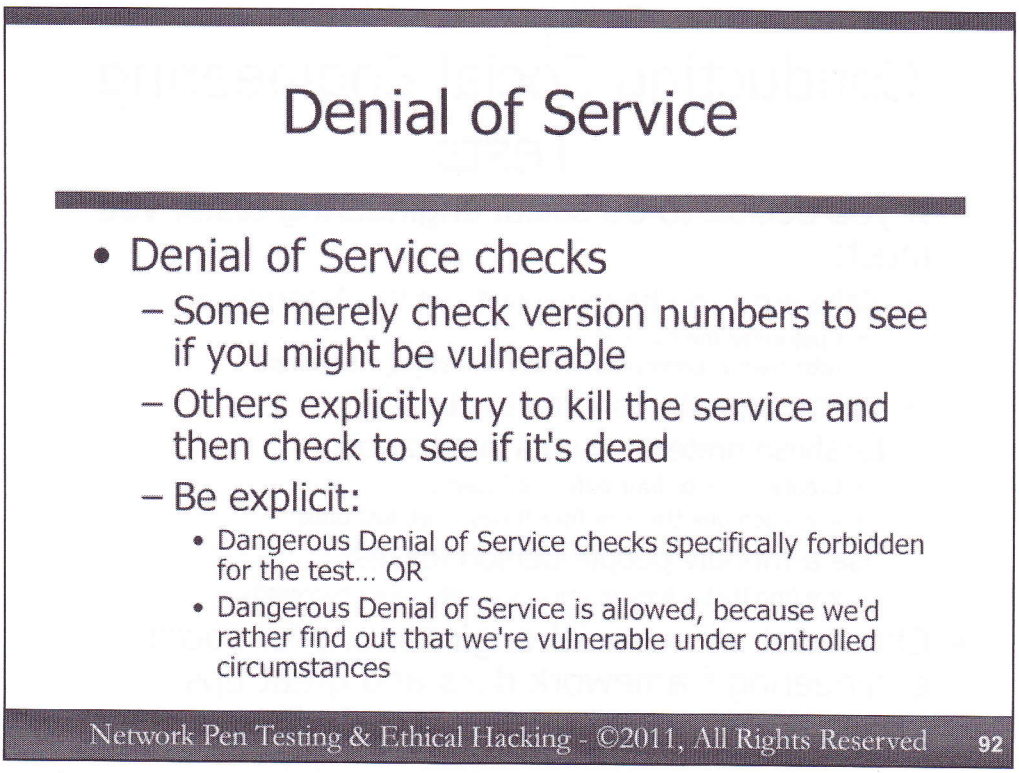
On the other hand, there is a strong argument for doing this type of testing. How can you know where and when you need to improve your defenses without actually measuring your weaknesses? With a carefully planned and executed social engineering penetration test, you can verify your security stance and improve it in a much more efficient manner.




If you do decide to perform social engineering tests as part of your test regimen, make sure everyone authorizing such a test understands the tradeoffs discussed on the previous slide.

If you are going to conduct such a test, after getting the GOOJFC signed by an appropriate person in management, make sure you define a clear goal. What policy or process are you going to test? What information are you trying to retrieve? Establish three or four different scenarios to test, and run each test several times to get a good feel for how your organization will respond. Create written scripts of the anticipated dialog with the target person, but make sure you are ready to improvise as needed. Finally, use a friendly person to make the social engineering test calls. We have found that a female voice on the phone is far more successful in getting people's cooperation. For some reason, people are less suspicious of a friendly female voice and therefore give up information more freely. Both women and men seem to trust a female voice more than a male voice.

The website www.social-engineer.org has a framework / methodology for conducting social engineering tests, along with many good tips and a podcast recorded periodically.

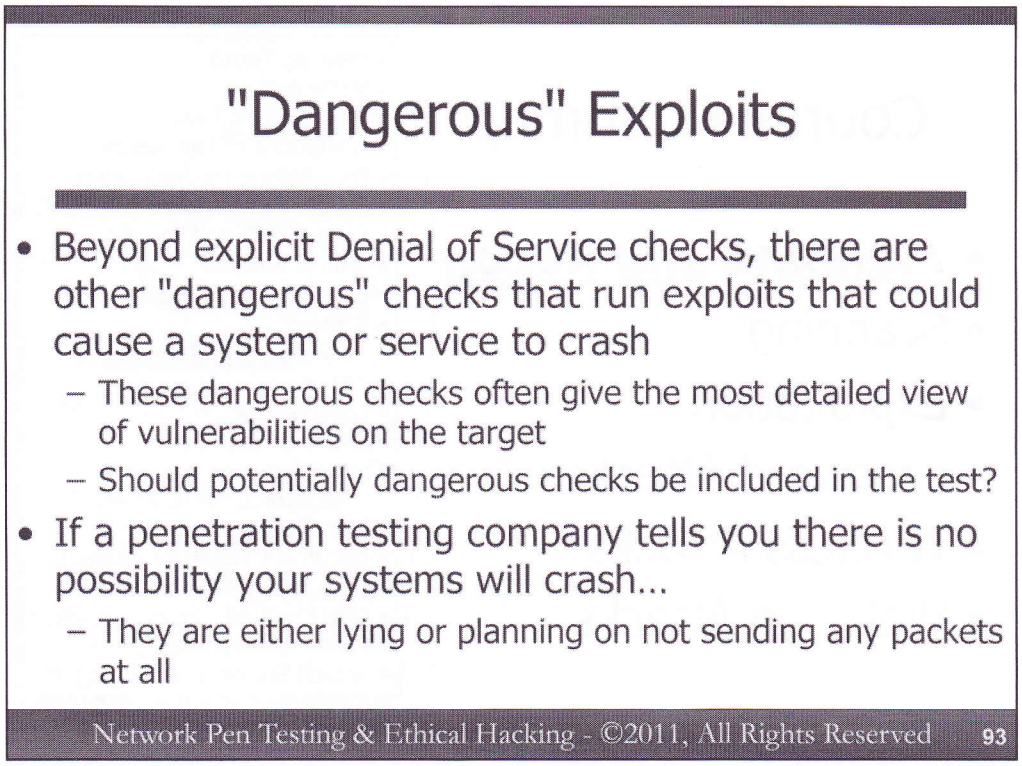



Another big decision to make during scoping a project involves Denial of Service. Some Denial of Service checks merely measure the version number of the target service and look it up in a list to determine if it is known to be subject to a Denial of Service attack. Such tests aren't very dangerous at all, and should be included in the scope.

The other kind of Denial of Service checks are more dangerous. They first verify that a service is running on the target. Then, they launch the Denial of Service attack. Then, they measure whether the service has died. If it has… the target machine is indeed vulnerable. But the target service is also dead, a potentially devastating impact on a production environment.

Based on this concern, most penetration test scoping agreements explicitly rule out all dangerous Denial of Service checks. However, if a service can be crashed by a remote attacker, it may be better to learn about it during the controlled circumstances of a penetration test rather than waiting for a vile script kiddie to hit it in the middle of the night during a holiday weekend or some crucial crunch-time processing. Most organizations forbid dangerous exploits during a test, while others do allow them.

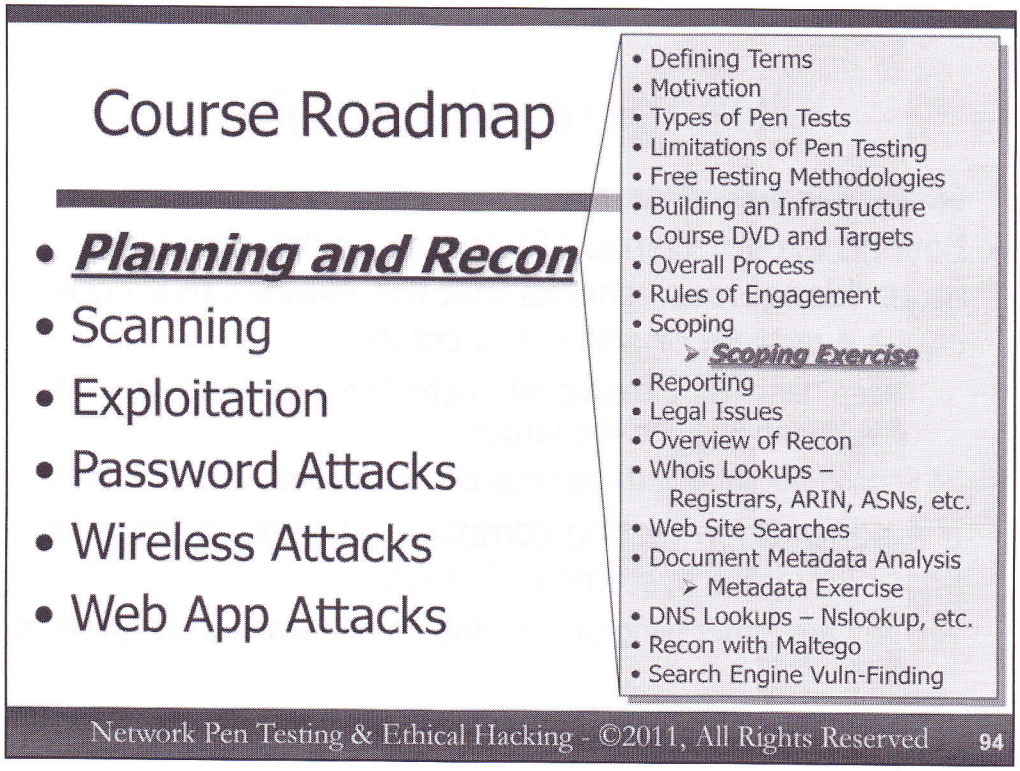
Therefore, you must be explicit about whether or not you want dangerous Denial of Service checks in your test's scope.



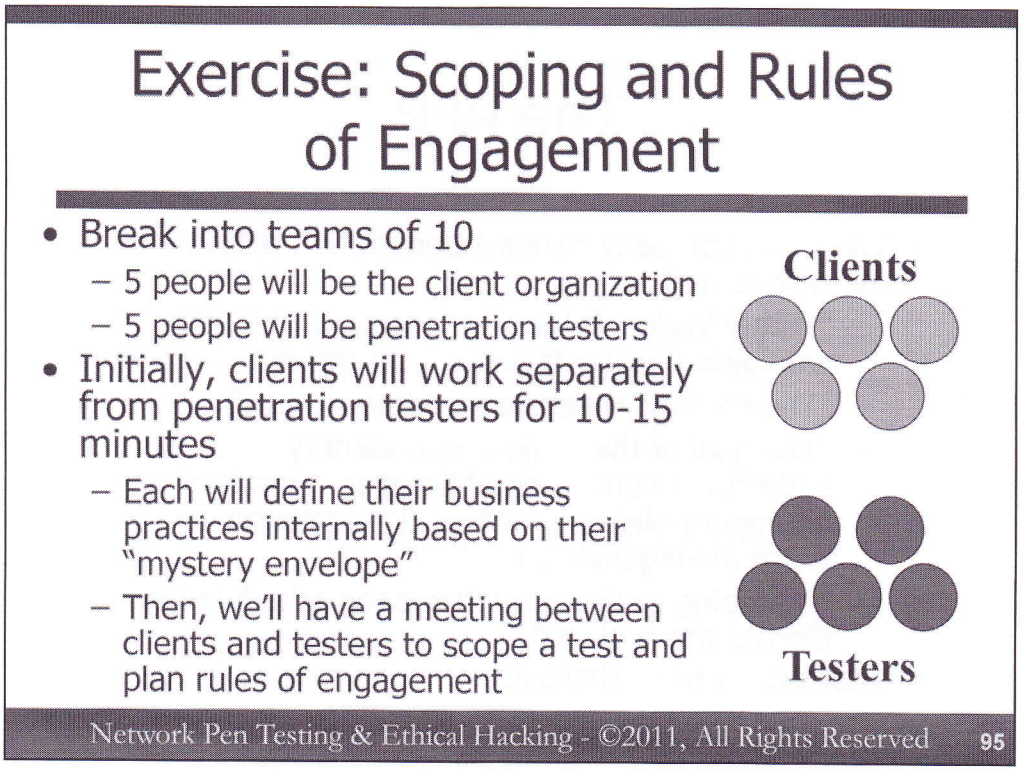

Beyond the checks that are explicitly designed for Denial of Service, other classes of checks could potentially crash a target service or system. These potentially dangerous exploits often times try to spawn a shell for the attacker by clobbering a buffer overflow exploit on the target. However, such techniques manipulate memory and therefore sometimes render a service inaccessible. Unfortunately, running these potentially dangerous checks is often the best way to get very detailed information about whether a target is vulnerable. Running a penetration test without dangerous checks gives less information about the target.

So, while scoping a project, target environment personnel and the testing team must decide: should potentially dangerous checks be included in a test, or avoided? Include a statement in your project scope documenting your decision.

Another important thing to keep in mind is that, during a penetration test, there is always some residual risk that the test could cause a system or service to crash, even if you explicitly disallow all dangerous checks. Even with no dangerous checks, the tester's tools will generate some strangely formatted packets that could cause a particularly feeble or unusual system to crash. Or, the additional traffic load could push a highly loaded network's performance over a cliff. Therefore, if a penetration testing company ever tells you that there is no possibility of their tests crashing your systems, they are either lying to you or planning on not sending any packets at all. You should respond to such a company, "Ummm… I know there is always a possibility of a crash in a target system. Tell me what you will do to minimize that chance, and your processes for detecting such a circumstance." In subsequent sections of this course, we'll look at scripts for monitoring services to make sure that they continue to run while a test is underway.




Because establishing firm rules of engagement and scoping a project properly are so important, we will now conduct an exercise on the topic. In this exercise, class attendees will formulate questions and responses associated with a penetration-testing Request for Proposal (RFP). The goal of this exercise is to help testers get a feel for the kinds of questions and answers that should be asked to scope a project and to determine the rules of engagement, sharing information about risks and benefits between clients and testers.

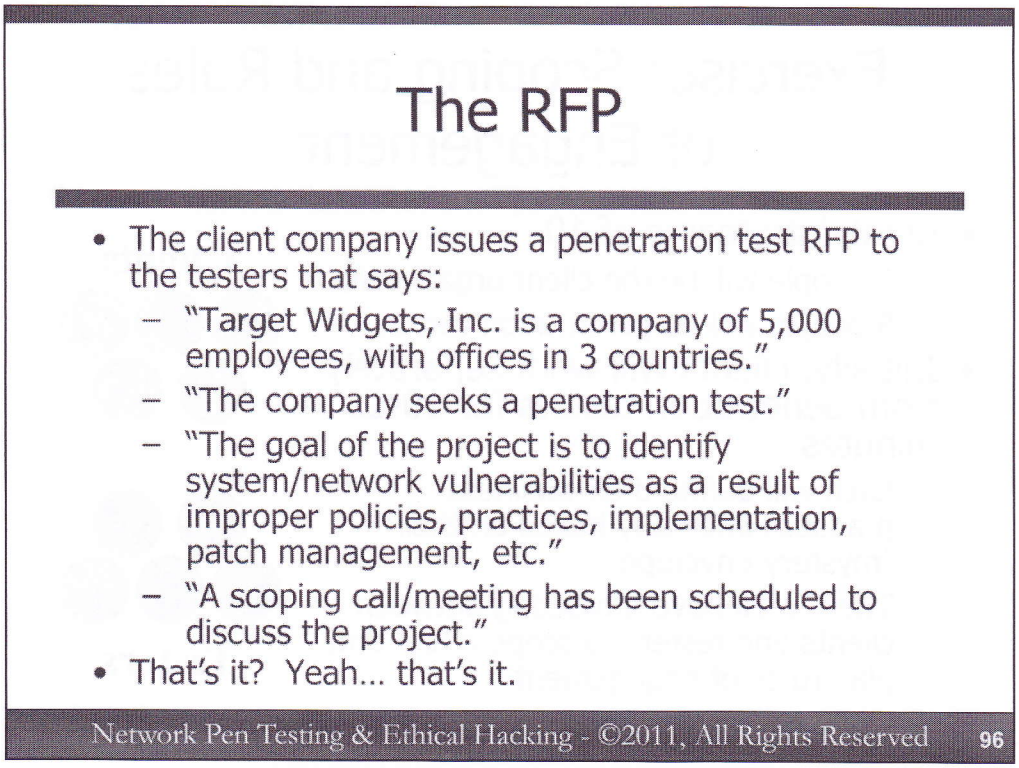



For this exercise, your instructor will break you into teams of approximately ten people (if you are taking the class via SANS@Home, you will be assigned to a discussion room for the exercise). Each team of ten will itself be broken into two subteams, each with five people: the clients and the testers.

Five people will represent the clients, an organization that has issued an ambiguous RFP for a penetration testing project. The other five people on the team will represent the penetration testers, who need to get more information about the project for scoping. Furthermore, both sides need to agree on rules of engagement. The testers should also describe the risks and recommended approach for the project to the client.

You could view this exercise as an external penetration testing company getting information to prepare a bid for a client, or you could view it as a set of internal testers preparing for a test of their own organization by discussing the test with the target business unit. Either approach is acceptable.

With the teams established and working in different parts of the room, the clients and testers will work separately for about ten to fifteen minutes, reviewing the details of a mystery envelope that each subteam will receive. The client envelope will describe the client's business with information that the testers should ask about. The tester envelope will describe the background of the testers to help them plan their approach.



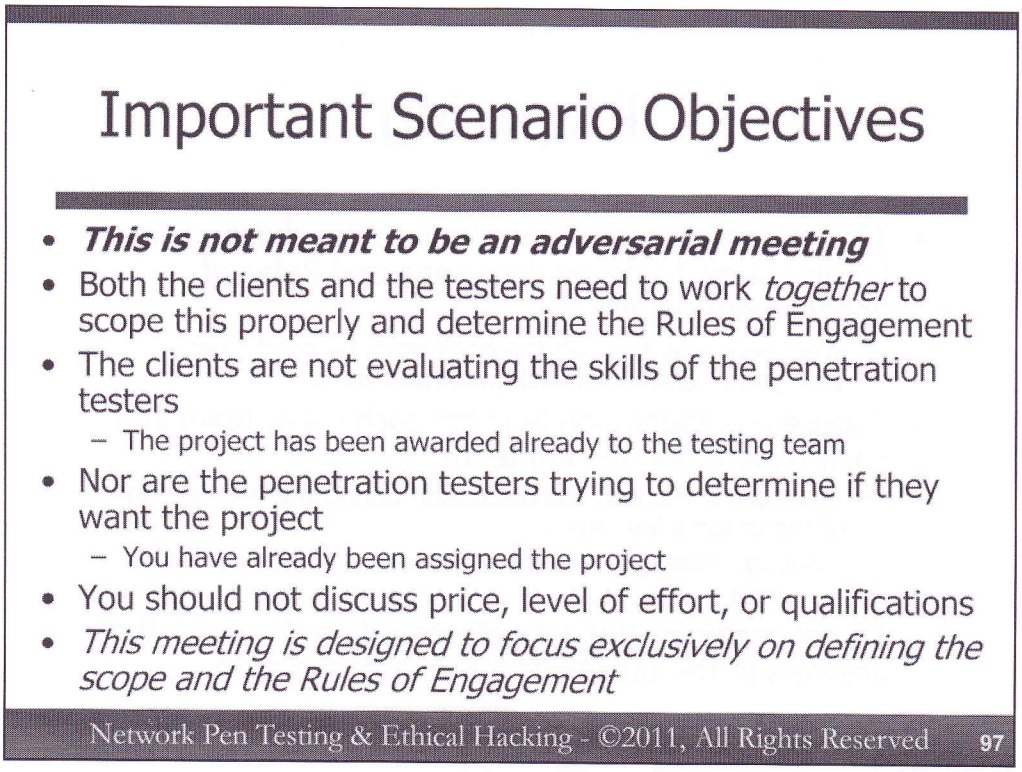

The RFP issued by the client company provides rather limited details about the test. Quite often, in the penetration testing business, testers are presented with very limited information about a potential project up front, making the scoping task vitally important so that both the client and the testers are on the same page for the test.

The RFP includes the following facts:

- The test will be performed for Target Widgets, a manufacturing company with 5,000 employees and offices in 3 countries.
- The company wants a penetration test (either from an outside penetration testing company, or from a technical group within the company; either is a valid approach for our purposes here).
- The goal of the project is to find security flaws that may have resulted from improper policies, practices, implementation, patch management, etc.
- That's it… The RFP includes no further information.

For the exercise, the testers will receive a "tester's envelope", while the clients will receive a "client's envelope". The instructions in the envelope will provide more detail about the given organization, as well as certain items to make sure are covered during the scoping and rules of engagement meeting.

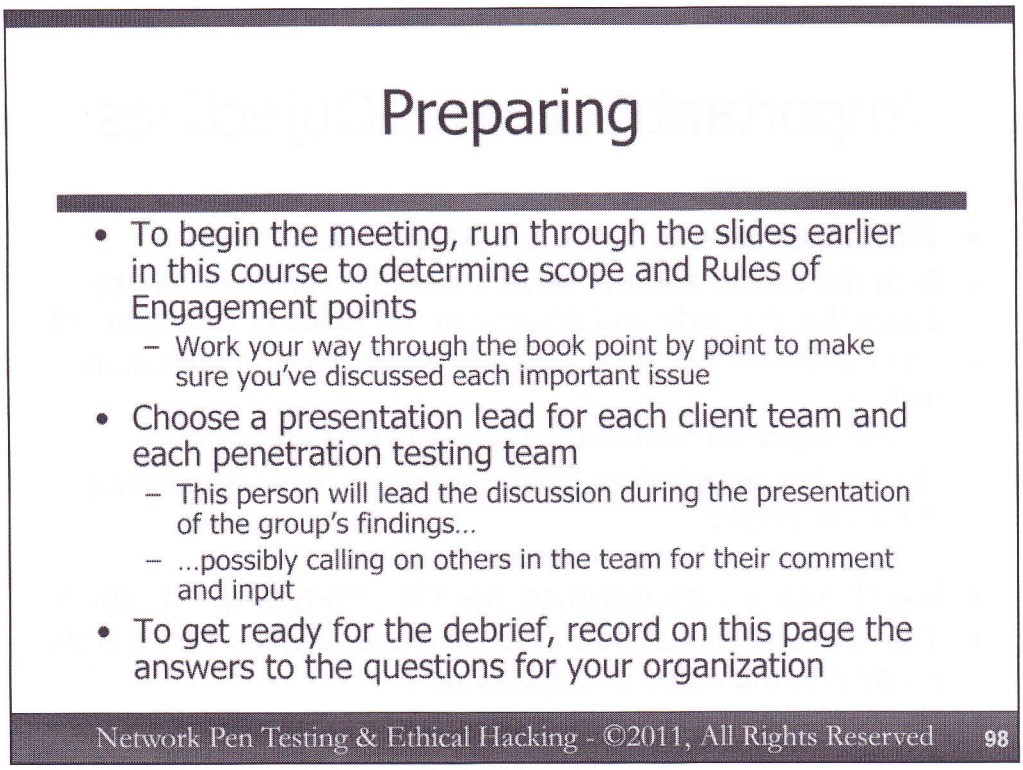
Make sure you take notes in your books and/or on the pages in the envelope, because you must prepare for this meeting. If you have any further questions, please feel free to ask the instructor.




For this exercise, please keep in mind that we are focusing exclusively on scoping and setting the Rules of Engagement. ***The meeting is not meant to be adversarial.*** Engage in a positive discussion to determine the proper scope and Rules of Engagement, improvising where necessary.

The clients are NOT evaluating the skills or background of the penetration testers. Furthermore, the penetration testers are not trying to evaluate whether they want to engage in the project. The project has been awarded to the testing team, and both sides are delighted with the decision. The point here is to devise an appropriate scope and Rules of Engagement.

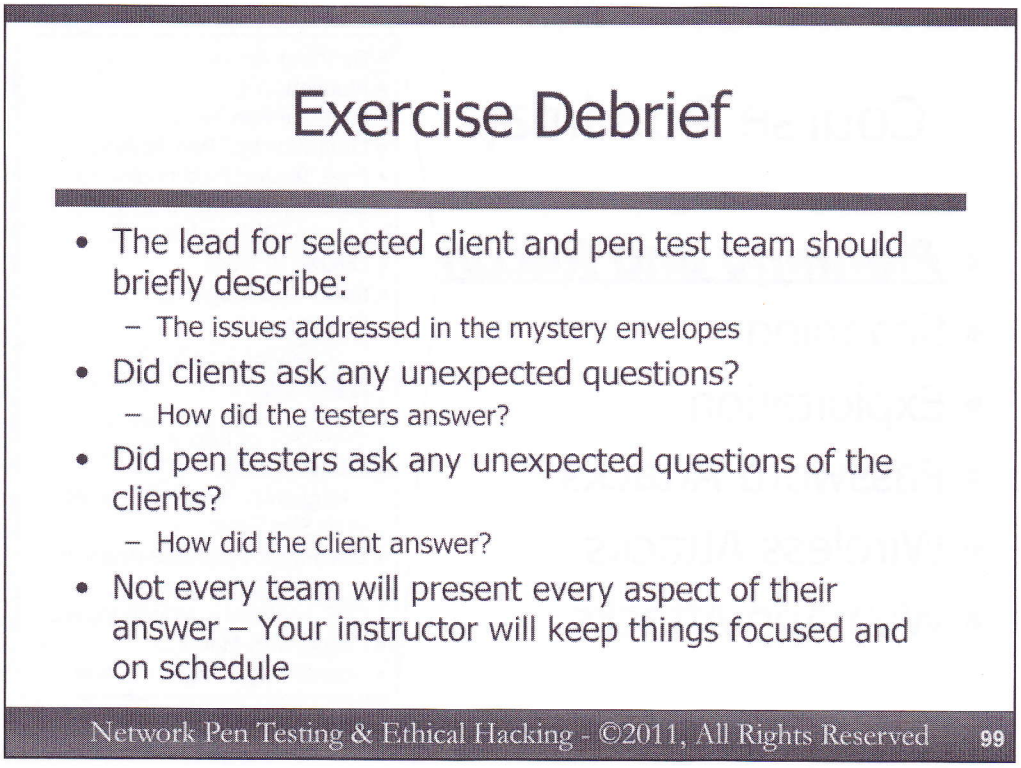
Do not discuss price, level of effort, or qualifications during this meeting, because we need to focus on scope and Rules of Engagement.




To begin the meeting between clients and testers, run through the slides earlier in this session (560.1) to devise a set of questions for scoping and setting Rules of Engagement. Work your way point by point through the book to make sure you've covered each issue.

Each testing team and each client team should choose a lead presenter who will lead the discussion for your teams when we re-convene the class for a debriefing. The presentation lead may call on other people from the team to help answer specific questions or share a particularly useful insight.

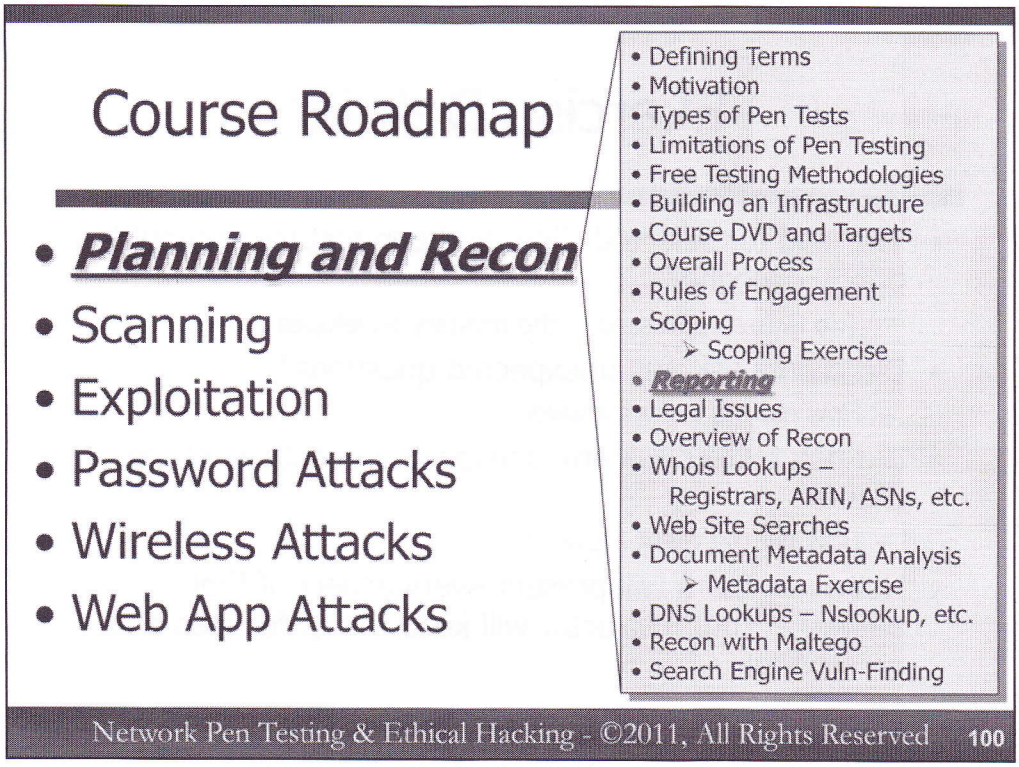
To get ready for the debrief, record your answers to the mystery envelope questions on the preceding page and on this page (or on those pages themselves).




Now, let's conduct a debrief session for the exercise. The course instructor will lead a discussion, choosing people from each group to present on the results of their scoping discussion during the exercise.

In particular, you will be asked whether the issues in the mystery envelopes were properly addressed during the scoping discussion. Also, did the clients ask any unexpected questions? How did the testers answer? Did the testers ask anything out of the ordinary, and how did the clients respond?

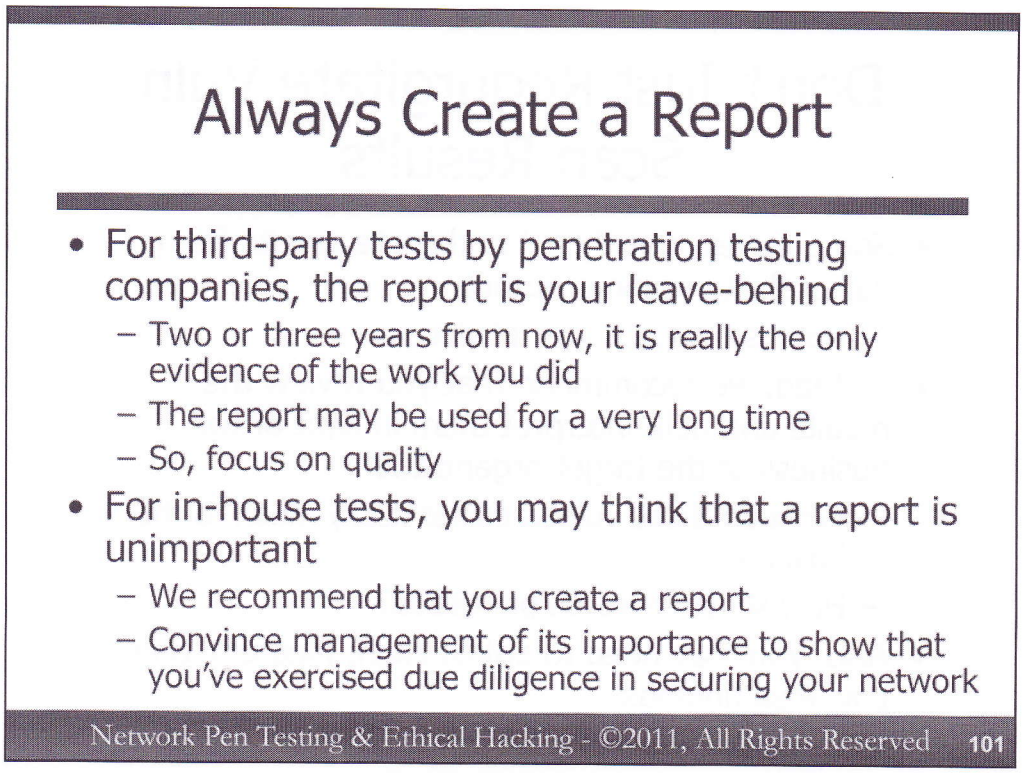
Unfortunately, there may not be time for every team to present every aspect of their results. Your instructor will help keep the conversation focused so that the most salient points will be addressed while keeping the class on schedule.




Next, we'll discuss reporting our results for the penetration test or ethical hacking project. Note that the base report structure that we'll cover is very versatile, lending itself to network scans, vulnerability scans, penetration tests, physical security tests, shrink-wrapped product reviews, and more.

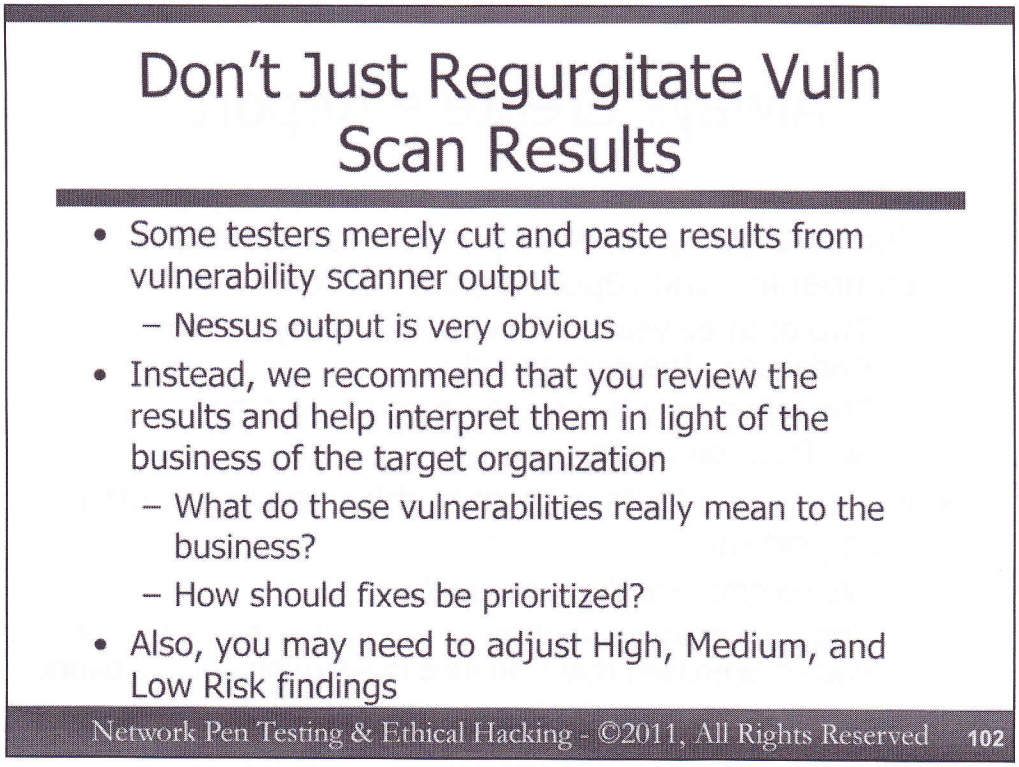
Pay careful attention to the reporting component of your project, as it is where you'll usually provide the most value for the organization you are testing.

Note also that the reporting format we'll discuss is not only paper-based. The same overall structure can also apply to final presentations describing the project and its findings.




Testers should always create a final report describing their work and findings. If the testers work for a third-party penetration testing or ethical hacking company, the report is really the only evidence they leave behind of the project they performed. A high-quality report may be used for several years after an engagement, as fixes are deployed, new architectures are considered, and new products and configurations are rolled out. Because of this potential long-term value of the report, make sure you focus on creating a high-quality deliverable. Reports that are technically incorrect, improperly formatted, or poorly written will linger and could cause significant problems in the future.

Sometimes, testers who perform vulnerability scans of their employers in-house think that they don't need to write a detailed report. After all, you're just scanning your own enterprise network to see how things look. Why waste the extra time documenting your results and findings? Even in such circumstances, we still recommend that you create a report, because it helps codify the work you've done and provides an organizational memory of the value you've provided. If management won't provide enough time or staff resources for creating a report, work hard to convince them of its importance. A report is concrete proof that your organization is exercising its due diligence in conducting vulnerability scans on a regular (or at least sporadic) basis. If there are major security problems in the future (such as a major breach), and your organization is investigated by the government or share holders, the vulnerability scanning reports will be helpful in showing your attempts to measure your security stance in the past.

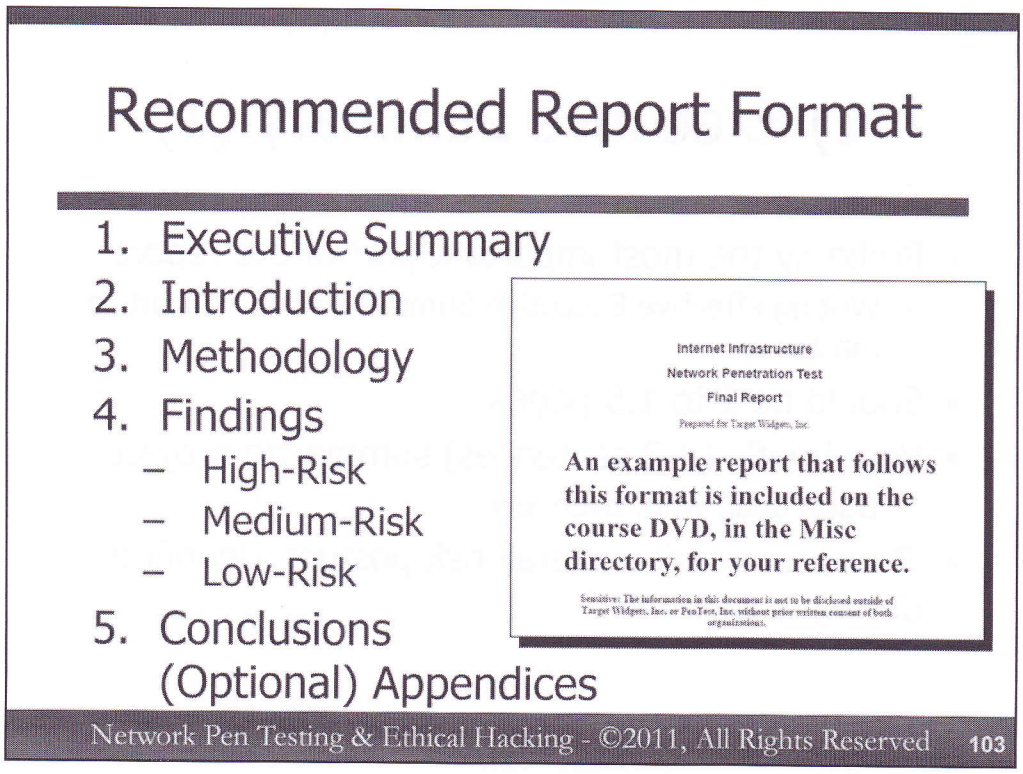



A common mistake of testers in writing reports is merely to cut and paste the output of their vulnerability scanning tools into a document, and turn it in as the official report. Sometimes, they don't even format their results, simply throwing Nessus output over the wall hoping that someone will act on it.

Stock vulnerability scanner output is very obvious. its format is very canned, and often not easily interpreted by non-technical personnel. Many of the issues that are identified are false positives, and the risks and recommendations do not tie in at all with the target organization's business objectives.

We strongly recommend that you review the results of your vulnerability scanning tools and other techniques, and then help interpret them in light of the business of the target organization. What do these vulnerabilities really mean? How could they impact the business? Given a realistic threat that exploits the vulnerability, what risk does it pose? And, how should fixes be prioritized, whether they be patching, configuration changes, architecture alterations, etc.?

Also, note that you may need to adjust the High/Medium/Low risk indications of your vulnerability scanning tools. What Nessus or another scanner calls a "High" risk might be a Low risk for the target organization. Of course, the opposite is also possible. All of these issues have to be addressed in the business context of the target organization.




This slide contains a recommended report format. Of course, you may iterate on this structure, adding or removing components. Your organization may already have a report format for penetration tests and related work. If so, get it and use it. If not, feel free to adopt this structure, reviewing it and tweaking it to meet your needs.

We recommend that the penetration testing, vulnerability assessment, or ethical hacking report includes these elements:

*Executive Summary:* This brief up-front matter is meant for executives who may not read the full report, providing them the most important conclusions from the work.

*Introduction:* This component describes the project at a high level, answering the who, where, when, and why aspects of the project.

*Methodology:* This part of the report describes the "what" of the project – what did the team do? It covers the process of the penetration test or ethical hacking engagement.

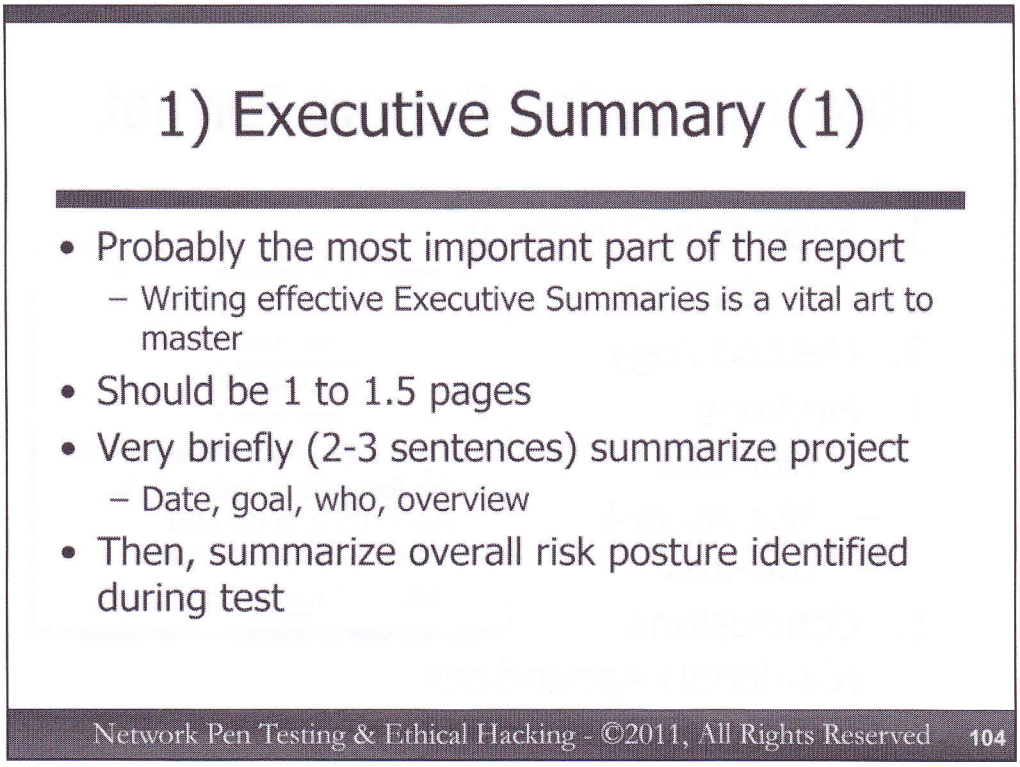
*Findings:* This section presents the actual findings, listed one by one, in the target environment with detailed technical descriptions. The findings are sorted so that the most significant risk issues are discussed first.

*Conclusions:* This last section summarizes the project results, and is very reminiscent of the Executive Summary.

The report also may include optional Appendices of reinforcing data and results.

An example report that follows this format is included on the course DVD, in the Misc directory.

Let's look at each section in more detail.

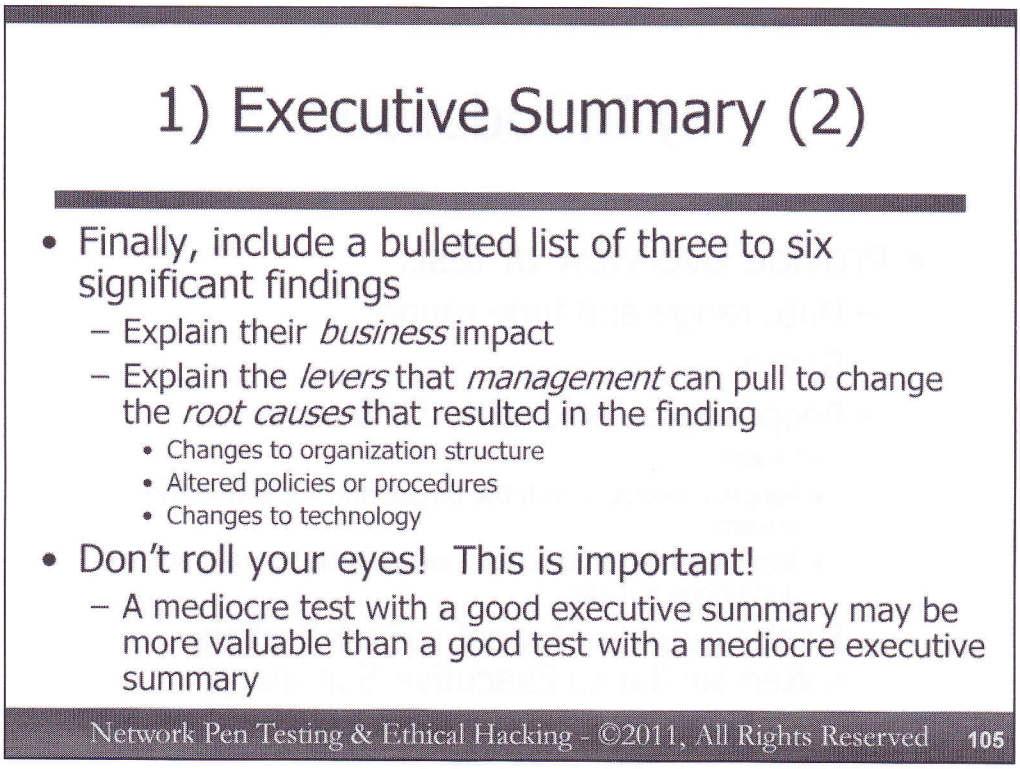



The Executive Summary is probably the most important section of the entire report. Composing an effective Executive Summary is quite an art form, and an important one for testers to master. The purpose of this section of the report is to describe to decision makers what the results of the report mean, and, more importantly, recommendations for the actions they need to take based on its results.

The Executive Summary should be quite short. Ideally, it will be under 1 page in length, and should be no more than 1.5 pages. It should not get highly technical, nor should it go over every finding. It should be composed so that it can be read by itself, offering firm conclusions and advice for someone who doesn't necessarily read the rest of the document.

The Executive Summary starts with a brief description of the project, in three or fewer sentences, describing the project's date and goals, as well as who participated in the project. It also provides a very brief overview of what was done.

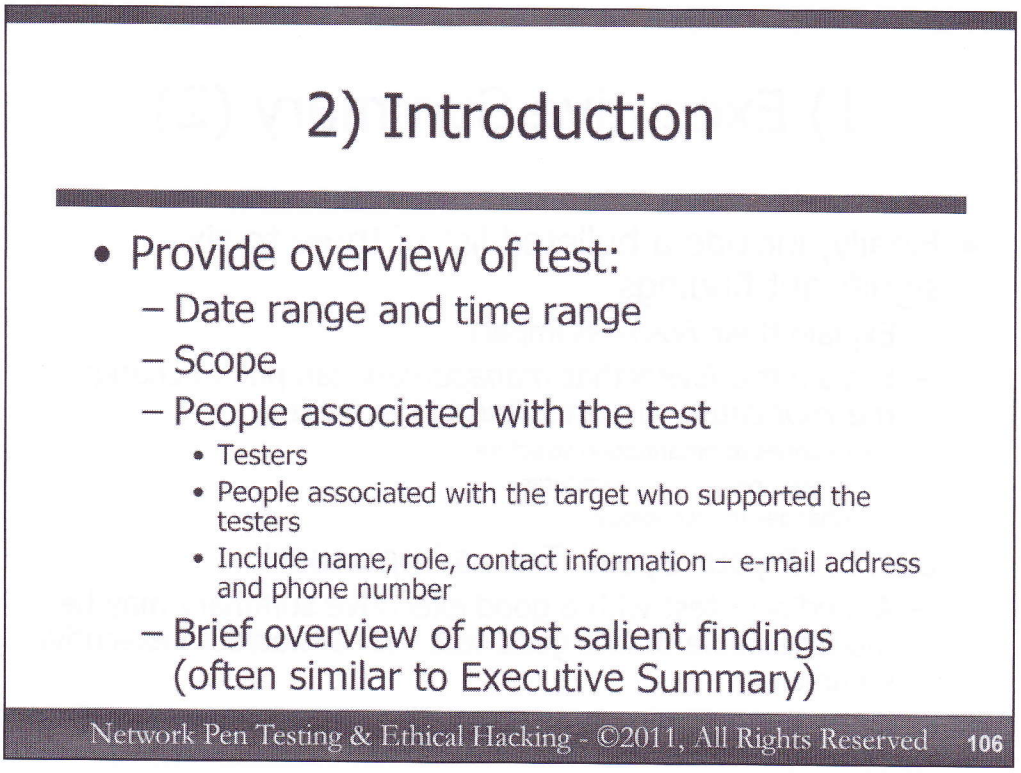
Next, the summary covers the overall risk posture of the target environment. Were major problems discovered? Is the environment worse than expected by testers and/or the people reading the Executive Summary? Or, did the test results show that the security of the target environment appeared to be sound?




And now comes the vital part of the Executive Summary. You need to explain the business impact of your findings. You can use a bulleted list of the biggest three to six findings, followed by several sentences that explain the business impact in terms of risk for each finding. Focus on the institutional levers management can pull to change things to eliminate not just a given discovered vulnerability, but the underlying reason that the vulnerability exists. Such levers could include changing organization structure, altering policies and procedures, adding personnel, deploying new technology, or many other possibilities.

Don't roll your eyes about this component of the report and this mindset for composing it, thinking that it is merely management fluff. Effectively communicating your results to management is critical. A mediocre test with a good Executive Summary is very likely much more valuable in improving an organization's security stance than a really good test with a mediocre Executive Summary. The former provides impetus for change and improvement, while the latter just reinforces security vulnerabilities in the minds of technical people who, in all likelihood, know what those flaws are already.

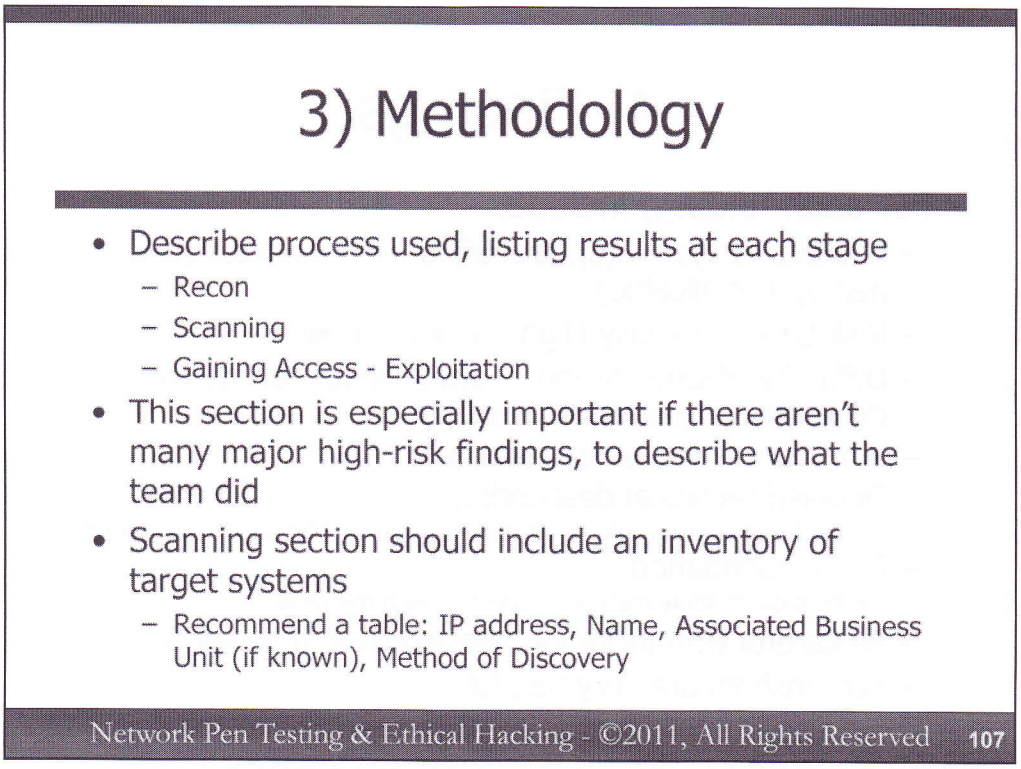
Remember that the Executive Summary should be able to stand alone as a document without the rest of the final report. Often, these one or two pages are split from the rest of the report and sent up a management chain.




Next, we move on to the Introduction component of the report. This one-to-three page section provides an overview of the project so that the reader understands when the project occurred, what was included in the scope (and possibly items purposely left out of the scope, if applicable), and who participated in the test.

Include a list of people, organized in a table, that identifies the testers and the individuals associated with the target environment that supported the testers (usually those who participated in the daily/weekly debriefings). The table should include each individual's name, role, and contact information such as a phone number and e-mail address. Then, provide a brief overview of the most salient findings, using language similar to that included in the Executive Summary, but possibly with more technical detail.

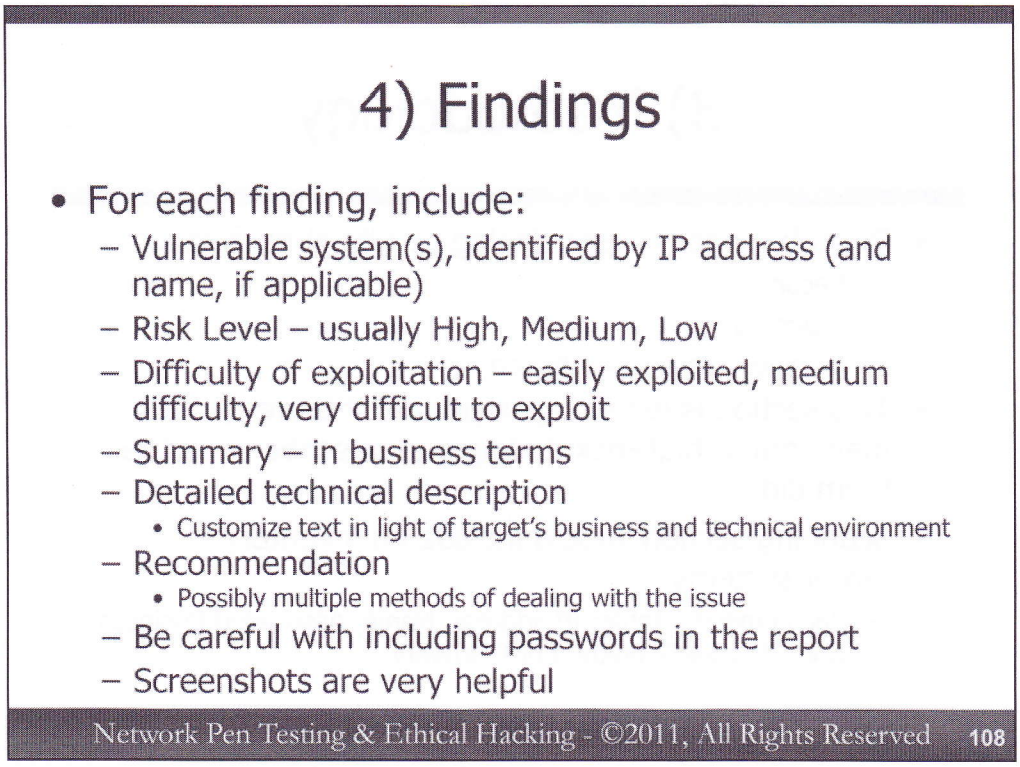
These Introduction components of reports are immensely helpful six months or later after the project in analyzing what was tested. Many organizations frequently refer back to these Introductions to determine what else in their environment needs to be tested, and what needs to be retested.




The next part of the final report includes a description of the methodology used in the testing. Provide a description of the step-by-step process used, and briefly list the findings discovered at each phase. Depending on how you conduct the test, this section may include details of the recon, scanning, and gaining access phases.

This methodology section may run from one to ten pages, depending on how the test was conducted. If there are very few high-risk findings associated with the project, this section actually grows in importance to demonstrate the value provided by the testing. Sometimes, a testing project finds very little wrong with the target environment. This is, of course, good news. But, if the Findings section (which we'll discuss next) is skimpy, the target organization may question the quality, veracity, and completeness of the test. Thus, in situations where there aren't many findings, we need to provide even more detail in this Methodology section to show at a fine-grained level what we did during the test. That demonstrates not only that we conducted a thorough test (again illustrating and recording our due diligence), but it also makes the test more reproducible in the future. A year later, the target organization may want to redo the test, and they can rely on this report to replicate it exactly, performing a gap analysis on changes in the interim, or purposely tweak the process next time.

The scanning component of the Methodology section should include an inventory of all of the machines that were included in the test. We recommend a table with one row per target machine, listing the IP address, machine name(s), the associated business unit (if it can be determined), and its method of discovery (DNS, ping sweep, Google searches, etc.). During the recon discussion later in this course, we will discuss maintaining this inventory while the test occurs. We can directly use this inventory, in a condensed form, in our final report. For some tests, this inventory can get very long, and should be moved into optional appendices at the end of the report.




And now, we get to the Findings section, where the technical results are described. We recommend starting with high risk findings, prioritized so that the highest of the high-risk issues comes first. Then, move down to medium and low risk, again prioritizing each within its subsection.

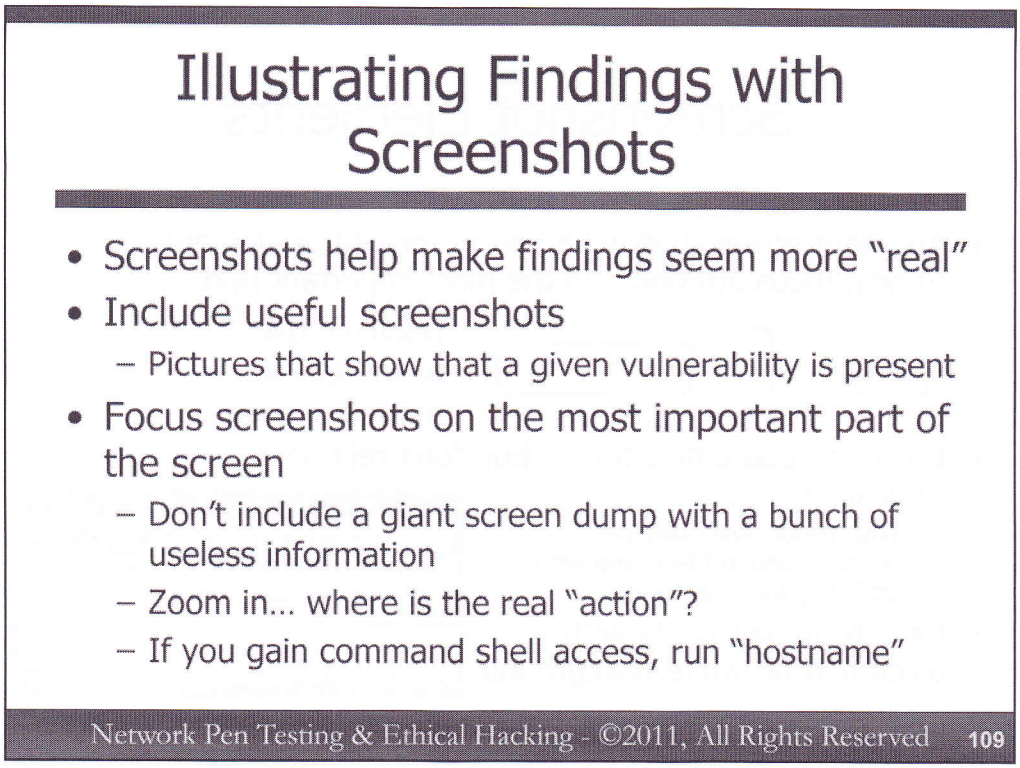
For each finding, list the system(s) that exhibit the given flaw, identifying the machine by IP address and name (if you know it). Include the risk level and estimate of how difficult it would be to exploit the given issue. The risk level is typically sorted into High, Medium, and Low, but your given organization may want to characterize risks in another fashion. Then, provide a two sentence summary of the finding, again focused on the business risk to the extent that you can. For some of the lower-risk findings, the business issue may be fairly small.

We then get into the technical details. Here, we provide a description of what the flaw is, and how an attacker could exploit it, causing an impact to the target organization's business. The discussion is described in technical terms, but again ties back into the business risks.

And, finally, we include a recommendation section, which should provide one or more means of remediating the discovered flaw. If there are multiple ways of addressing the issue, include them all, but point out the one most likely to match the business needs and technical environment of the target organization.

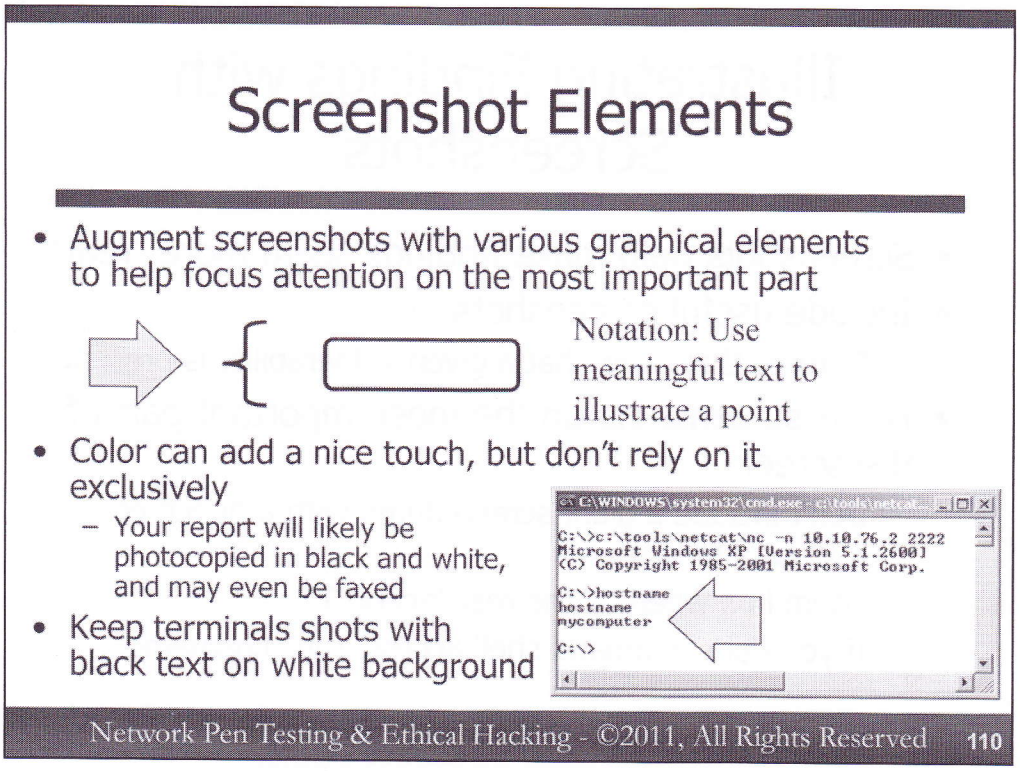
Be careful with including the successfully guessed or crack passwords in your report, because these reports are often passed around to numerous people in a target organization. Including in a report passwords that might be re-used elsewhere could be a security risk, so verify with the target organization whether such information should be included in a report. Many penetration testers have a policy of describing the characteristics of cracked passwords (such as length or character set), but not including the actual passwords in the report.

In the technical findings section, screenshots that illustrate the issue can be very helpful in conveying a lot of information and making the results feel more "real". Network diagrams can also go a long way in explaining some technical ideas.




Screenshots aren't just eye candy. They can really help a report have its intended effect (motivating the organization to improve its security stance) because they show the success of a penetration tester or ethical hackers' work.

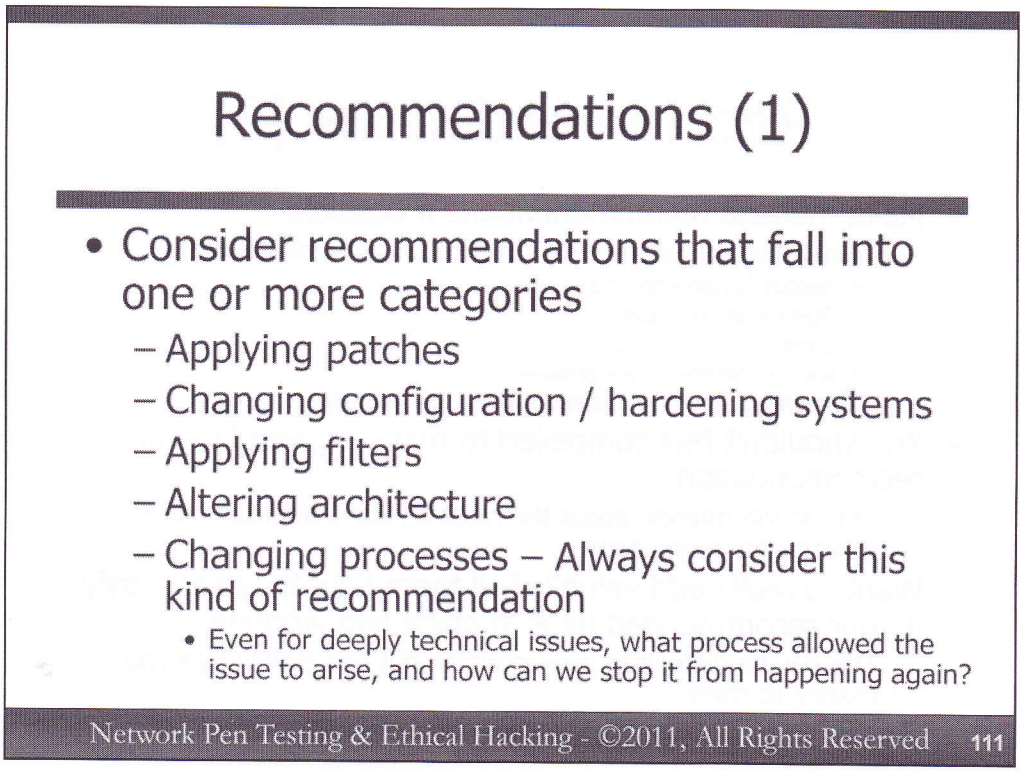
When including screenshots in your reports, make sure to focus the screenshot on the real issue you are trying to illustrate in the prose of your report. Some penetration testers include giant screenshots with a lot of detail distributed around the picture, but only a small portion of the figure is actually meaningful. The crux of your screenshot could be lost amid a sea of other unimportant information in the same figure. Focus your screenshots on the real action you are trying to show, which might be a vulnerability discovered by a scanning tool, a command shell returned by a successful exploit (often running "hostname" to show the machine's name), or other useful items.




Screenshots should almost always be augmented with graphical clues as to where the most important information is located in the figure. Use arrows, brackets, or squares overlaid on the screenshot to focus the reader's attention on the most important information. Small textual notations can also help hammer home the point of a given element on the screen. Try not to make your screenshots too busy, however. Remember, we are trying to demonstrate that a given vulnerability is real, not bewilder the reader with so much obscure detail in our pictures that they will not understand.

Color can help to bring out salient points. For example, you could using red graphical elements to really zoom attention on a big problem. However, keep in mind that your beautiful color report will likely be passed on to other people in the target organization via photocopying and faxes, and will likely not retain the color elements you create. Thus, while color can help, don't rely on it exclusively to make your point. Make sure that there is contrast between your screenshots and your graphical elements.

Terminals shown in screenshots often work best with black characters on white background. They are easier to see that way (especially when faxed), and require less toner from printers.




Because they are so valuable, let's look at the recommendations element of the Findings section of the report in more detail. Most of your penetration testing and ethical hacking project recommendations will fall into one of the following categories:

*Applying patches:* Many times, a test discovers vulnerable software that can be fixed by upgrading to a more recent version or applying a patch. Such changes tend to be easy, provided that they are conducted in coordination with the change-management process of the organization.

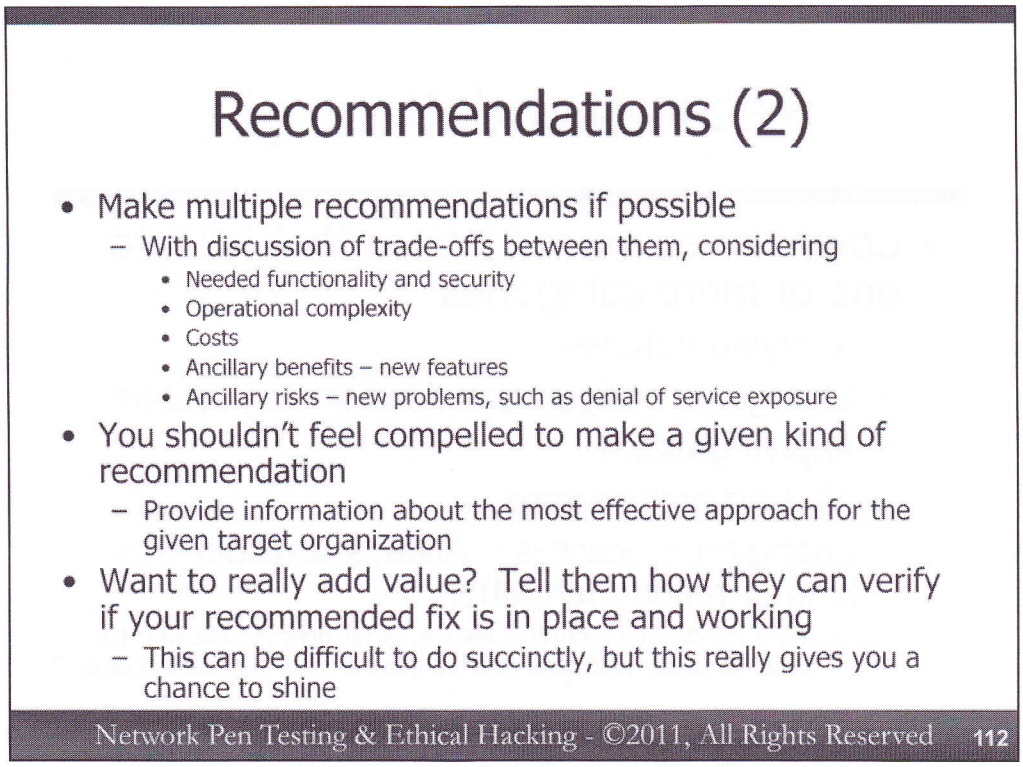
*Changing configuration / hardening systems:* Tests often have findings associated with shutting off an unneeded service, removing software that doesn't have a defined business need, or otherwise changing the configuration of a target machine to harden it from a security perspective.

*Applying a filter:* Some issues identified during a test can be mitigated by applying a filter in front of the target service, through a network firewall, network-based Intrusion Prevention System, or even a host-based filtering technology, such as a personal firewall or host-based IPS.

*Altering architecture:* The most effective way of dealing with some flaws is to alter the overall architecture of the target environment, moving systems around, deploying new kinds of technology, applying filters in different order, or tweaking the flow of data in the environment. Such solutions tend to be among the most expensive when compared to other issues in this list.

*Changing process:* For each one of your recommendations, always consider making a recommendation to improve processes. Even for deeply technical issues, some process associated with the given technical issue allowed the vulnerability to arise. Make recommendations for improving the process to avoid such an issue happening again in the future.

Most recommendations in a penetration testing or ethical hacking report fall into the first three categories here: patches, hardening, and filtering.



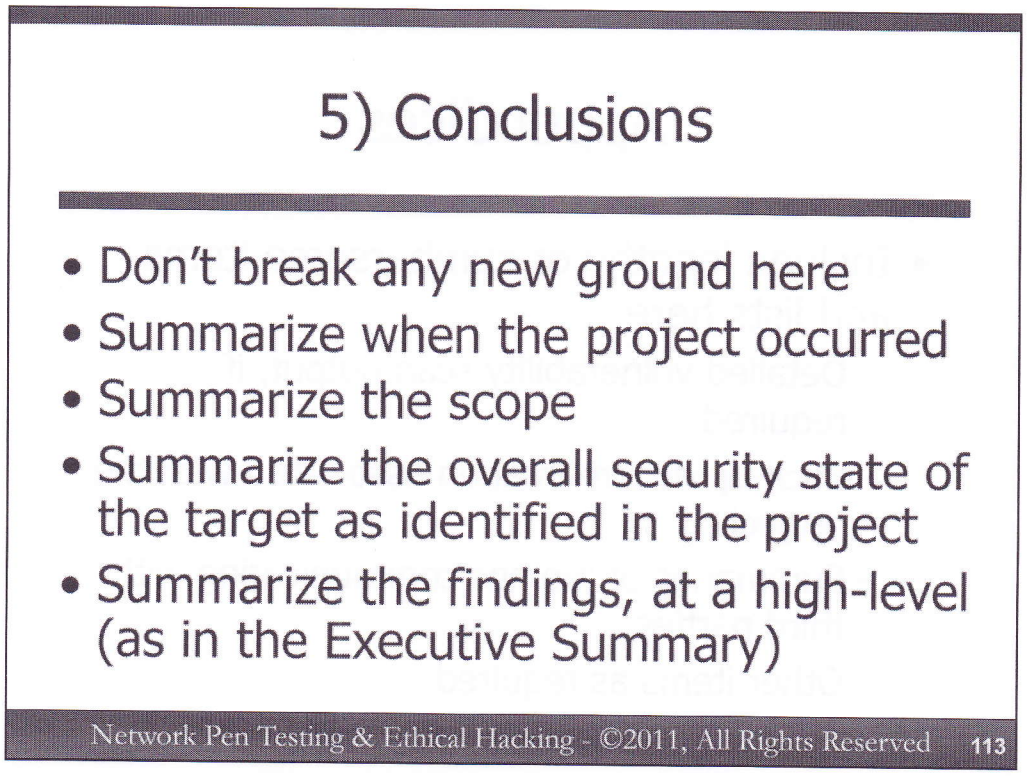

Where possible, provide multiple recommendations for addressing a given discovered vulnerability. Discuss the trade-offs between the recommended solutions in light of the target organization's environment, which could be associated with different impacts on operational complexity, varying costs, and ancillary benefits of some solutions over others (including the target organization's familiarity with a given type of operating system or software package, the enabling of future services or features from one solution over another, etc.) Some solutions could have ancillary risks, as they open new vectors of attack, possibly including denial of service exposure.

Make sure that you base your recommendations on what you feel is truly the best way for the given organization to deal with a discovered issue, independent of pressures to raise or lower costs. If cost is a concern, you should list both higher- and lower-cost alternatives, and discuss which ones make the most sense in light of your understanding of the target environment.

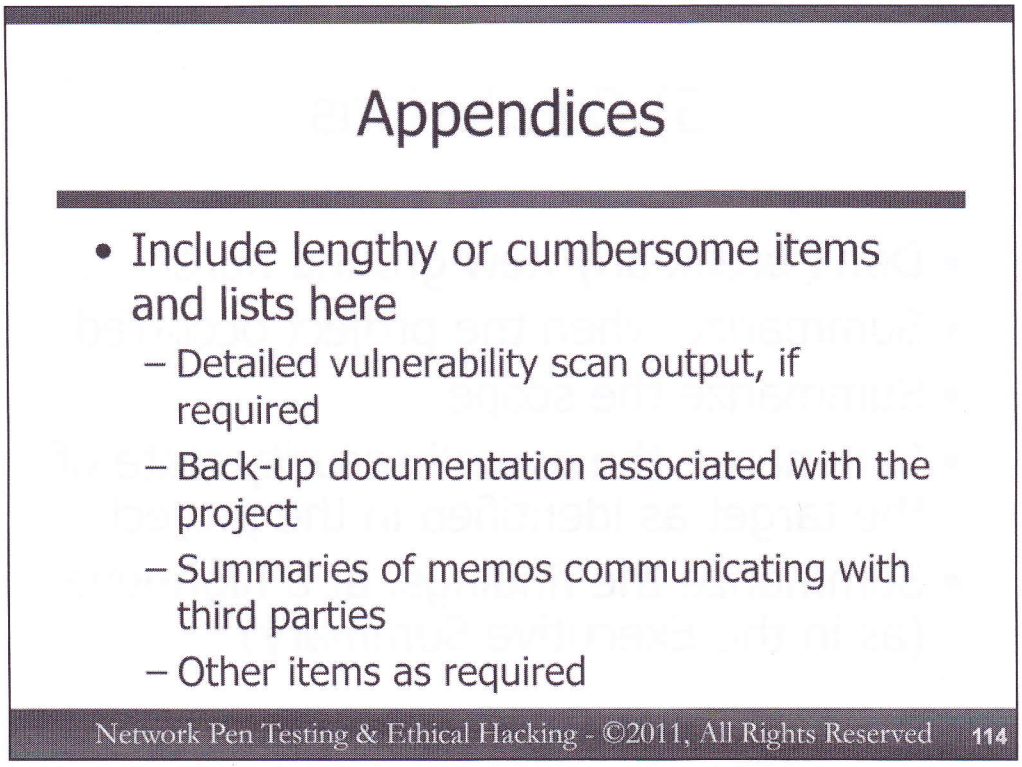
Don't be pressured to recommend only high-cost solutions, which some third-party penetration testing companies might do to raise the price of follow-on mitigation projects.

From the opposite perspective, recommend budget-conscious approaches to make sure you have a possibility of improving the security stance of the organization. But, don't be pressured to use only low-cost solutions even if budgets are slim.

If you really want to go above and beyond in providing value in your penetration tests, provide in your recommendation some steps an organization can take to verify that your recommended fix is in place and working effectively. For example, if you recommend applying a patch or a filter, provide target system personnel with some command line activities they can run to verify that the patch is in place or that the filter is properly filtering. Such fix verification advice can be difficult to formulate succinctly, but it provides some extra verification for target system personnel that their defenses have improved because of your work. Might this prevent you from getting some extra follow-on testing work? Yes, that is possible, but it makes your initial test results and report much more valuable to the organization.




The final component of the report is its Conclusions section. This section is usually about a page in length, and summarizes the various aspects of the project covered elsewhere. This part of the report shouldn't break any new ground. Instead, it briefly summarizes topics included already in the Executive Summary and Introduction, such as when the project occurred, its scope, and the overall security state of the target organization determined by the testing. The conclusions can also include the bulletized list of major findings presented in the Executive Summary.

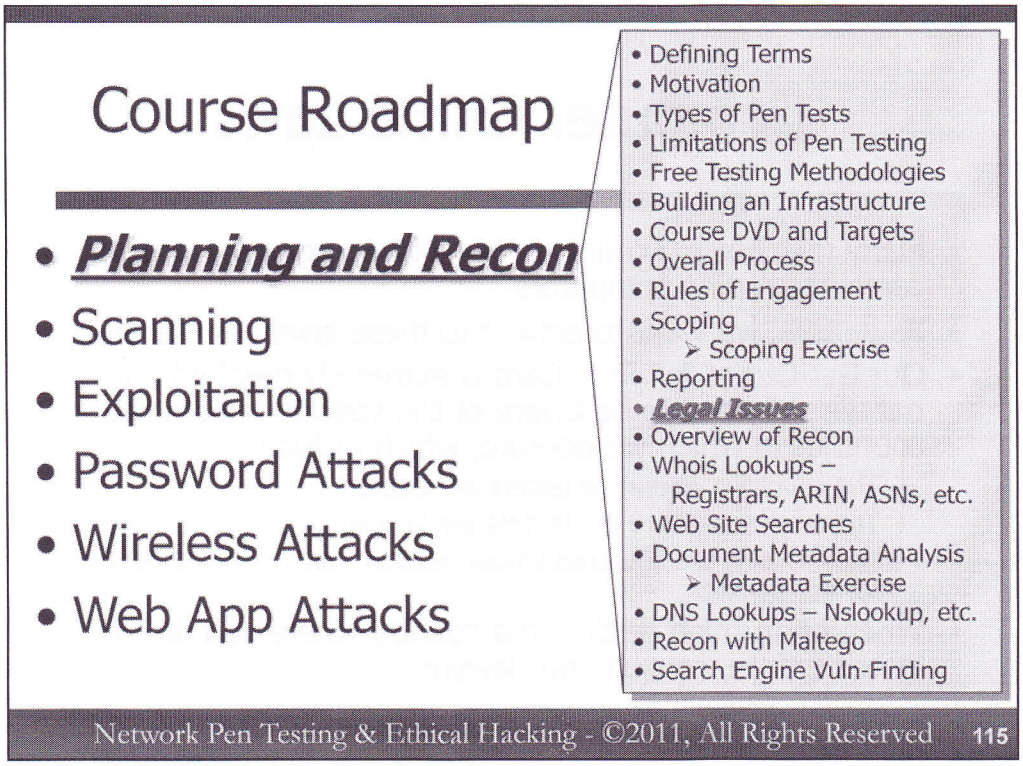



The report appendices should include lengthy outputs that would be cumbersome to put in-line with the rest of the report. You could include detailed vulnerability scan output in an appendix. However, make sure that the report recipients really want such information. It is often counter productive, and may make it look like you are merely adding weight to the report without any useful business or technical reason.

You could include back-up documentation associated with the project in an appendix, like the rules of engagement or scope description.

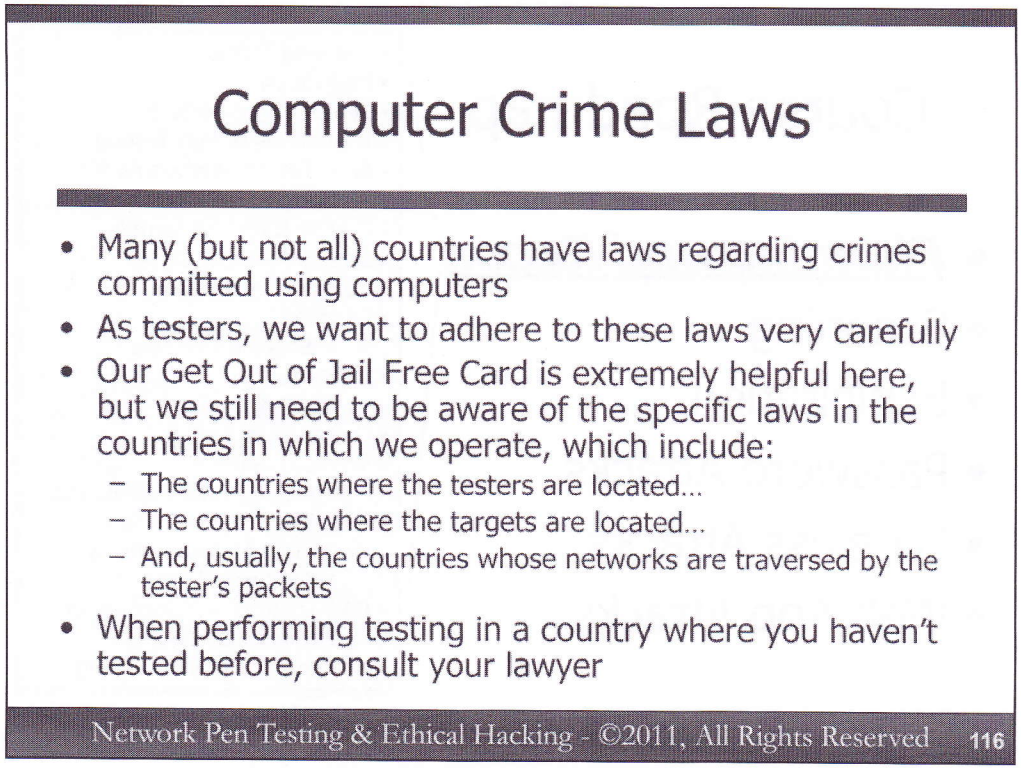
You also may want to include any memos that were sent to third parties getting their permission to scan their systems, such as a DNS server or web hosted environment. Also, if any requests were sent to these organizations to change their configuration, you may want to include a copy of the request and response for the change in an appendix.

Include other items as they are required or helpful.




Our final topic in this preparation component of the course involves an analysis of the legal issues associated with computer crime in various countries. While conducting a test, we certainly don't want to run afoul of the law. Thus, we need to know about the relevant laws and some of their implications.

We'll provide a brief survey of the legal issues in various countries associated with computer crime, but we cannot give you legal advice for the specifics of your organization, your target enterprises, and the details of a given test. This section will brief you about relevant laws, but for detailed legal advice and decisions, please contact a lawyer.

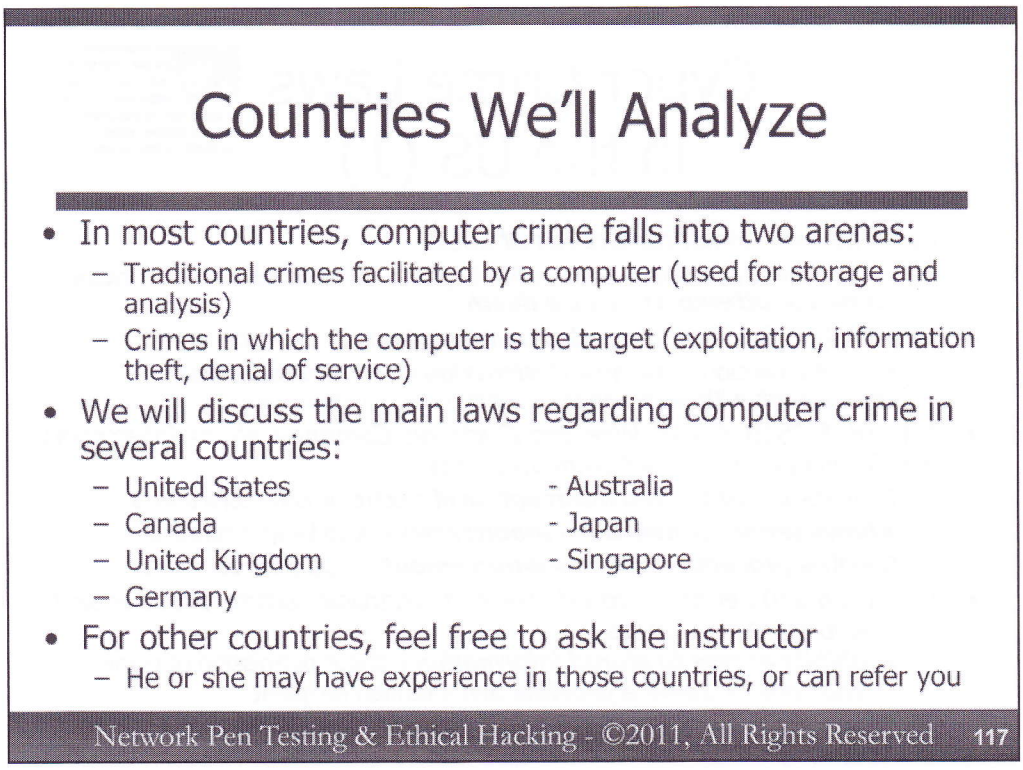



Many countries have instituted laws for dealing with crimes committed using a computer, so-called "cyber crime" laws. Not all countries have such laws, and indeed, attackers sometimes move to countries or operate through countries without such laws or where cyber crime laws are not enforced.

As penetration testers and ethical hackers, we want to make sure we carefully adhere to the laws of the countries in which we operate. Your Permission Memo (the "Get Out of Jail Free Card") is a very helpful thing in assuring that you have permission of the target organization that owns the systems you will test. Still, beyond that memo, we still need to have a feel for various countries' laws so that we can make sure we follow them.

It is important to note that the tester not only has to follow the laws where he or she is located, but also the laws in the country where the target machines are located. Furthermore, some countries (such as the United States) have indicated that any packets associated with a computer crime that traverse the country's borders fall within their law enforcement jurisdiction, regardless of where the packets originate or terminate. In other words, an attack from Germany against targets in Japan that traverse US networks would be subject not only to German and Japanese law, but also to US law.

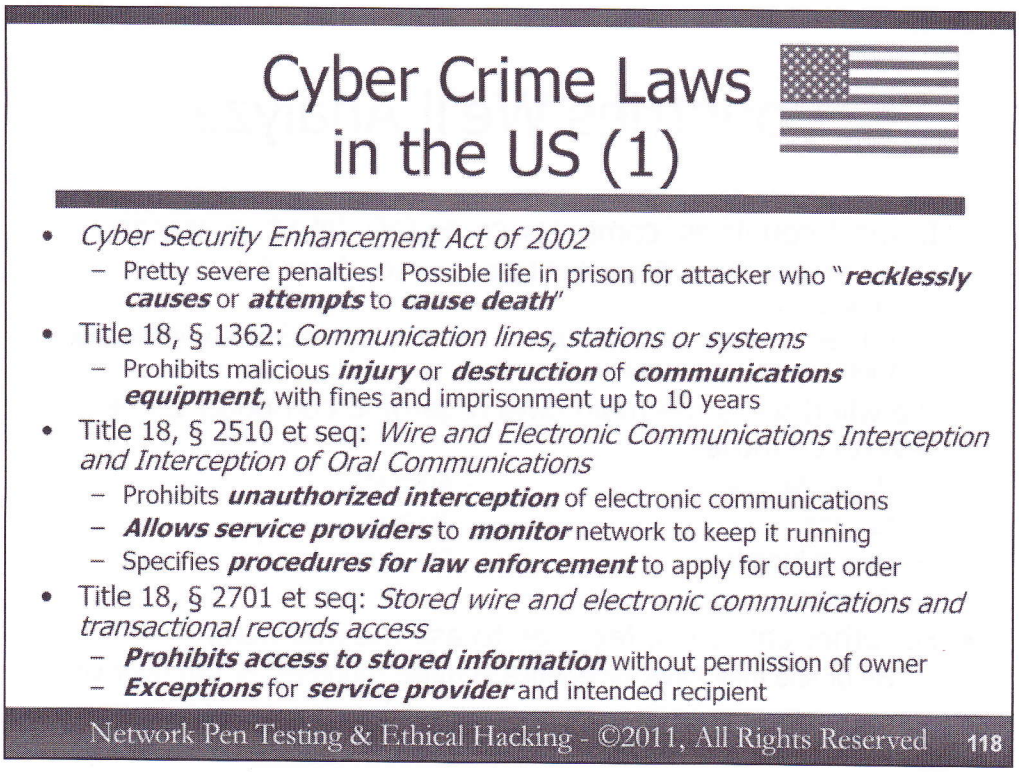
Because some of the legal issues associated with testing in different countries can grow very complex, we strongly recommend that you consult a lawyer when testing in a country where you haven't tested before. The lawyer can help explain the rules of the road to you, helping to make sure you don't run afoul of the law. For the most part, with a carefully documented Permission Memo, Rules of Engagement agreement, and Project Scope, you can operate in most countries legally without incident. Still, a lawyer's review can be very helpful in establishing peace of mind and making sure you've taken any late-breaking legal issues into account when formulating your test plans.




Generally speaking, in most countries, computer crimes fall into two arenas: crimes in which the computer was used to facilitate traditional criminal activity (acting as a storage or analysis device for the criminal who might be dealing drugs, participating in organized crime, trading child pornography, or engaging in terrorist activities), and crimes in which computers are the target of the attack (where criminals break into, steal information from, or crash computers). Some criminal activity fits into both arenas at the same time.

For this class, we will briefly describe cyber crime laws in several countries where the class is most often delivered, namely the United States, Canada, the United Kingdom, Germany, Australia, Japan, and Singapore. However, given that this course is not a legal class and its attendees are not lawyers, we will provide only a high-level view of the legal framework in each country, summarizing the most dominant cyber laws.

Many people who take this class are from other countries, whose laws we cannot cover due to time constraints. To address every country for each participant in this course, we'd take up a whole week just going over cyber crime laws. Thus, if your country is not covered in the list we'll address, and you have a significant interest in the cyber crimes laws of your country, please feel free to contact your instructor off-line during a break, to ask about the cyber crime situation in your country. Your instructor may have experience in that country, or may be able to refer you to someone who does.



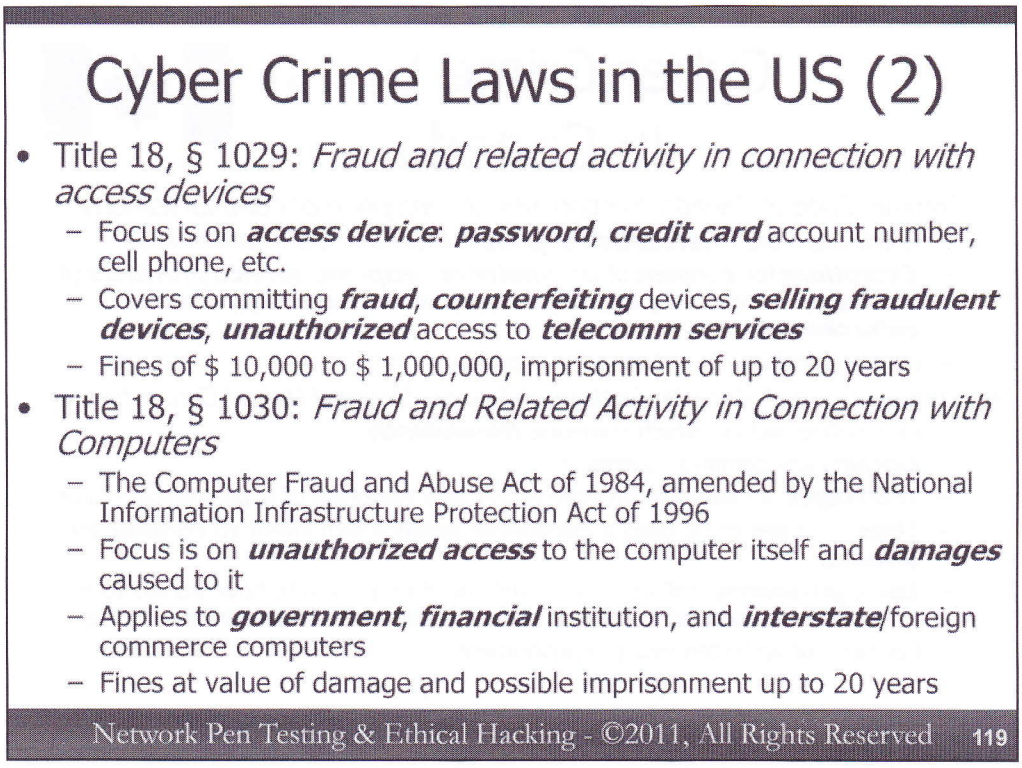

This slide presents some significant computer crime laws in the United States.

First, we have the Cyber Security Enhancement Act of 2002. This law was designed to modernize cyber crime legislation in the United States, particularly issues associated with terrorism. Indeed, the law includes a clause that states: ''...if the offender knowingly or recklessly causes or attempts to cause death from conduct in violation of subsection (a)(5)(A)(i), a fine under this title or imprisonment for any term of years or for life, or both.'' A penalty of up to a life sentence is very serious.

Next, Title 18, Section 1362 addresses malicious injury or destruction of communications equipment, such as radio, telegraph, and telephone or cable equipment operated or controlled by the United States or associated with US military or civil defense. Penalties include fines and imprisonment for up to 10 years.

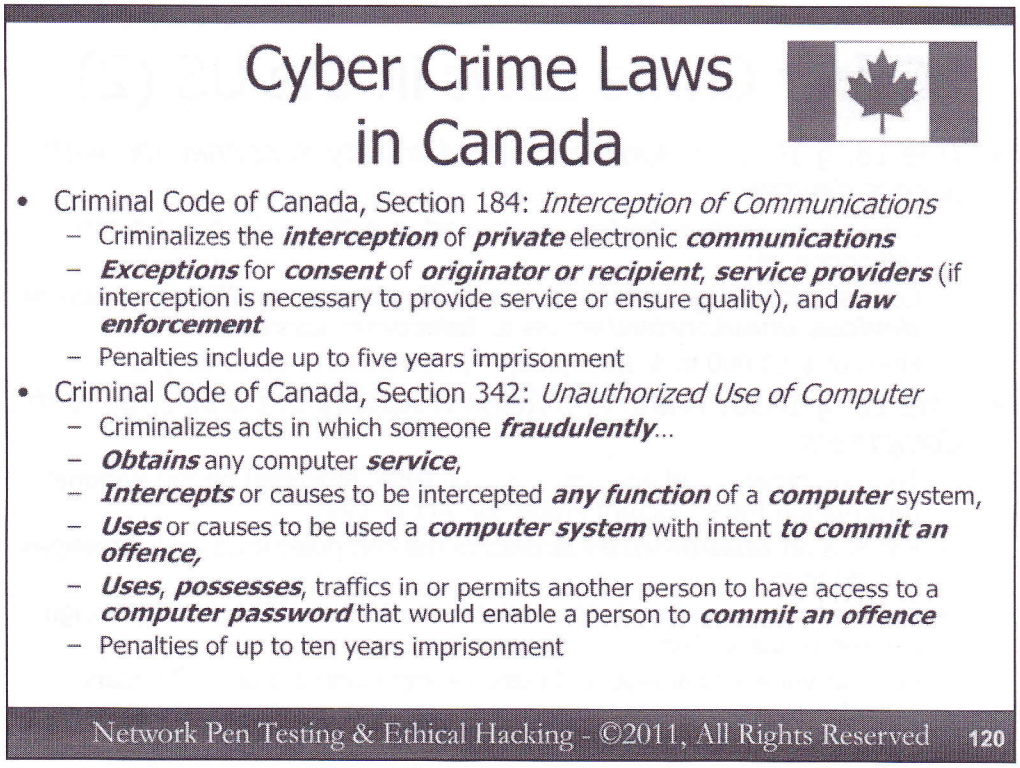
Thirdly, Title 18, Section 2510 and its subsequent sections govern interception of electronic communication, prohibiting interception of such information without explicit permission. The law includes exemptions for people who operate the network itself, so long as their actions are associated with protecting the network and keeping it running. Section 2518 specifies how law enforcement can apply for a court order to monitor communications.

While Section 2510 focuses on electronic communication in transit, another law, Title 18, Section 2701, protects stored electronic information. This brief law prohibits access to stored communications with penalties of fines and imprisonment ranging between one year and ten years. Exceptions are granted for the service provider, as well as the legitimate, intended recipient of the information.




While the laws covered on the previous slide are important, most criminal prosecutions associated with cyber crime in the United States focus on two other laws: Title 18, Section 1029 and Section 1030. The first of these laws (Section 1029) is associated with committing fraud using computer access devices, such as stolen, cracked, or guessed passwords, stolen credit card account numbers, cloned cell phones, and related items. In other words, the focus of this law is on the access mechanism for computers and its fraudulent use. The law deals with committing fraud, making counterfeit devices, selling or trafficking in counterfeit devices, and gaining unauthorized access to telecomm services.

Title 18, Section 1030 is the major law in the US under which most cyber crime cases are prosecuted. Originally known as the Computer Fraud and Abuse Act, this law protects federal interest computer systems from being attacked. In other words, this law focuses on the computers themselves, and the damage that an attacker may cause to them. According to this law, it is illegal to cause damages to or even to gain unauthorized access of a protected computer, as long as the value of that unauthorized computer use is over $ 5,000. Federal interest computers included machines owned by or operated on behalf of government departments and financial institutions. It also applies to computers associated with interstate and foreign commerce, which includes almost all systems connected to the modern Internet, as long as there is an exchange of financial compensation across state boundaries associated with the use of the given computer.




In Canada, while numerous legal provisions mention computer devices and electronic communications, two laws dominate from a cyber crime perspective: Criminal Code of Canada Sections 184 and 342. CC 184 is called *Interception of Communications*, and deals with unauthorized monitoring of private communications. It prohibits the unauthorized interception of such communications regardless of the form, specifically citing, "electro-magnetic, acoustic, mechanical, or any other device," that might be used for the communication. The law carves out a series of exceptions. If the originator or the recipient authorizes the monitoring, it does not violate the law. Likewise, service providers are allowed to intercept communications, if such interception is required for them to provide service or ensure quality. And, the law spells out the need for law enforcement to get approval from a judge for monitoring potential criminal communications.

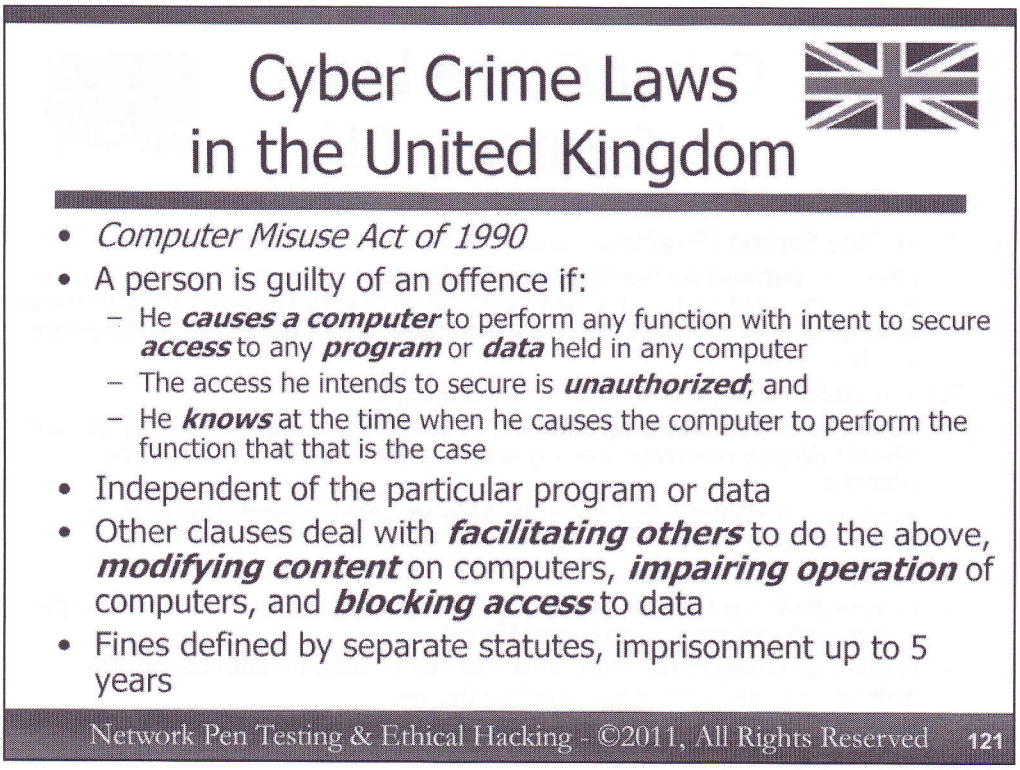
Penalties for violating this law include up to five years imprisonment.

The second major Canadian law associated with cyber crime is CC 342: *Unauthorized Use of Computer.* This law criminalizes the use of computers for fraudulent activities, including:

- Obtaining computer service without authorization
- Intercepting any function of a computer system
- Using a computer system with intent to commit an offence defined in other laws
- Using, possessing, or trafficking in passwords used to commit an offense

Penalties for violating this section range up to ten years of imprisonment.

The entire criminal code for Canadian law, which includes these two sections, can be downloaded from http://laws.justice.gc.ca/en/c-46/280634.html.



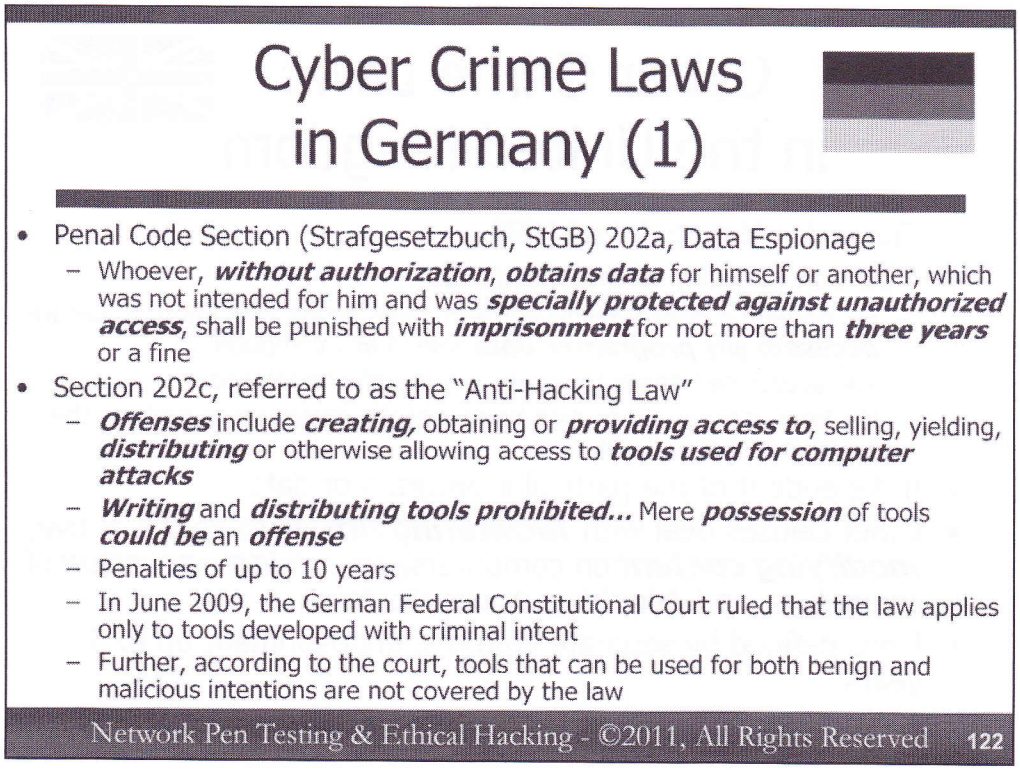

In the United Kingdom, the dominant cyber crime law is the "Computer Misuse Act of 1990." This law specifically says that, "A person is guilty of an offence if:

- He causes a computer to perform any function with intent to secure access to any program or data held in any computer
- The access he intends to secure is unauthorized; and
- He knows at the time when he causes the computer to perform the function that that is the case."

These specific conditions apply regardless of the particular program or data the perpetrator accesses. Other elements of this law deal with a person who facilitates others to commit the above acts, an issue that could be associated with people who write exploitation tools. Beyond securing illicit access, other clauses apply almost identical language to acts involving modifying data on computers, impairing the operation of computers, and blocking access to data.

The fines for such violations are not set forth in the law itself, which contains references to statutory maximum fees defined in separate statutes. Imprisonment for violation of this law could range up to five years.

Details of this law are available at http://www.opsi.gov.uk/acts/acts1990/Ukpga_19900018_en_1.htm.

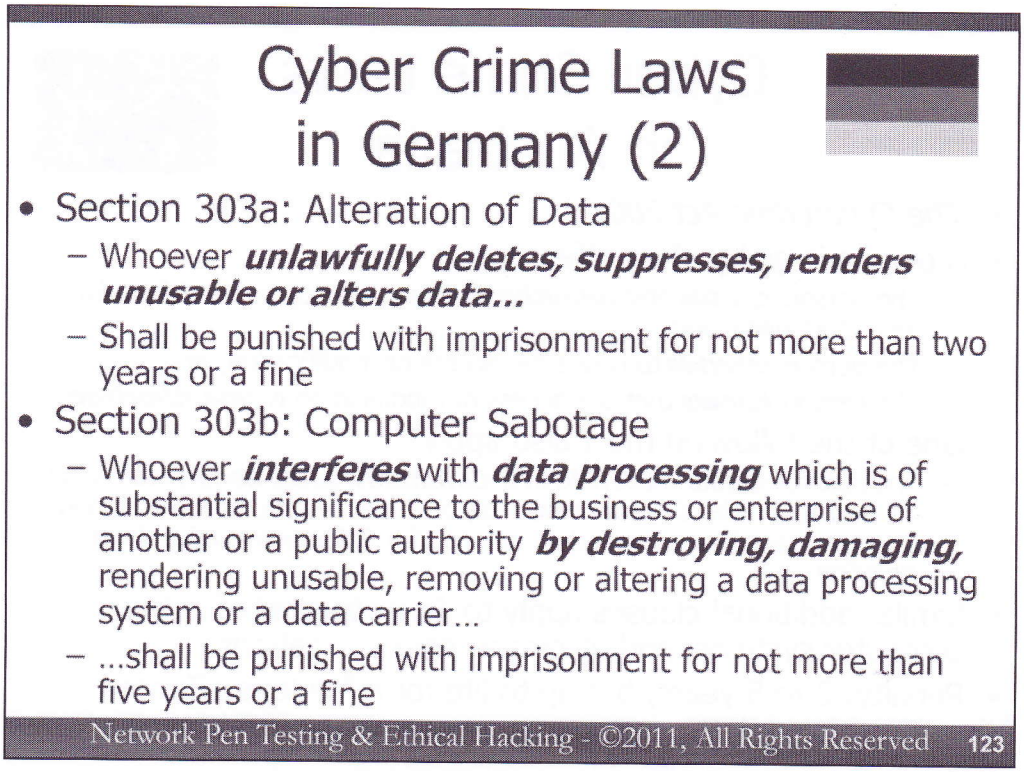



German cyber crime laws are contained in the penal code (referred to as Strafgesetzbuch or StGB for short), specifically in Sections 202 and 303. Section 202a is associated with Data Espionage, and states: "Whoever, without authorization, obtains data for himself or another, which was not intended for him and was specially protected against unauthorized access, shall be punished with imprisonment for not more than three years or a fine." Later language specifies that the data associated with this law must be stored or transmitted electronically or magnetically or in an otherwise not immediately perceivable manner. In other words, this section is intended to apply to electronic data, not paper records.

Section 202c, passed in 2007, is very controversial, sometimes referred to as the German "Anti-Hacking" Law by its detractors. It defines as a criminal offense the creation of and distribution of tools used for compromising computers. Many people have interpreted this law to mean that security researchers cannot create or distribute scanning tools, sniffers, and exploitation software in Germany. In fact, mere possession of such tools in Germany could be an offense, although such interpretations still need to be tested under this law as of this writing. With penalties up to 10 years, some security researchers and underground computer hackers have moved their operations outside of Germany, relocating to other countries including the Netherlands, the US, and the UK.

In June 2009, the German Federal Constitutional Court issued a ruling which clarified this law somewhat. The court said that the law applies only to tools developed with a criminal intent. According to the course, dual-use tools, which can be applied for both benign and malicious purposes, are not covered by this law.

An English translation of the entire StGB (including Sections 202 and 203, but not 202c) is available at http://www.iuscomp.org/gla/statutes/StGB.htm.




The other two major sections of the law associated with computer crime in Germany are Section 303a, which deals with the alteration of data, and Section 303b, which addresses computer sabotage.

Section 303a specifically prohibits the unlawful deletion or suppression of data, as well as acts that render it unusable or altered, thus covering both integrity attacks and denial of service attacks. Violations of this law are punishable with up to two years imprisonment or a fine.

Section 303b deals with interfering with data processing equipment that has substantial significance to other people from a business perspective. Destruction or damage to such systems is prohibited, with violations punishable with up to five years imprisonment or a fine.

# Cyber Crime Laws in Australia

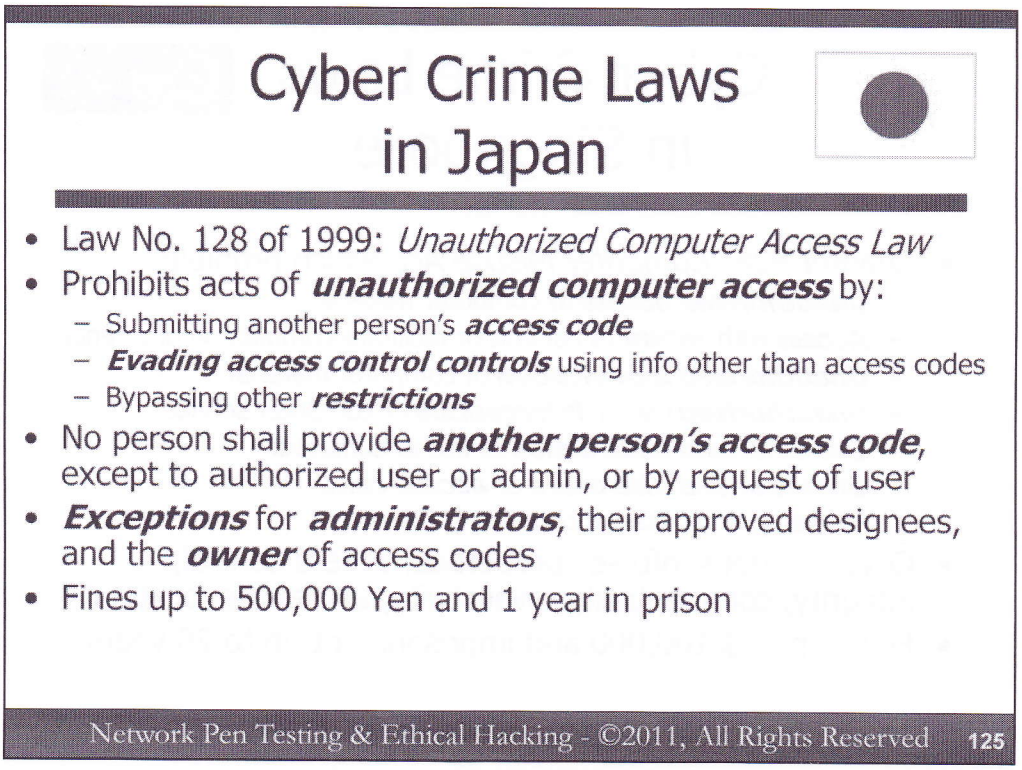
- *The Cybercrime Act 2001*
- A person is guilty of an offence if:
  - The person ***causes*** any ***unauthorized access*** to, or ***modification*** of, restricted ***data***; and
  - The person ***intends*** to cause the access or modification; and
  - The person ***knows*** that the access or modification is ***unauthorized***
- One of the following must also apply
  - The restricted data must held in a ***Commonwealth computer***, is held on ***behalf of the Commonwealth***, or the access to, or modification of, the restricted data is caused by means of a ***telecommunications service***
- Similar additional clauses apply to ***impairment*** of data, computer systems, and electronic communications
- Penalty: 2 to 5 years, but up to life for a "serious offence"

 124

A dominant law regarding cyber crime in Australia is the *Cybercrime Act 2001*. This law prohibits unauthorized access to or modification of data, computer systems, and electronic communications, and reserves its harshest penalties for impairment of any of these items. To be guilty of an offence, someone would have to cause unauthorized access to, modification of, or impairment to restricted data, with intent and knowledge that their actions are unauthorized. Furthermore, to classify as an offence, the data associated with the case must be either stored on Commonwealth computers, held on behalf of the Commonwealth, or associated with a telecommunications service. In other words, the law protects government computers and data, as well as systems accessed via public communications networks.

The penalties associated with such violations vary significantly depending on the specifics of the case, but tend to range between 2 and 5 years for most offences. The Act does describe imprisonment for life for those items characterized as a "serious offence".

The text of this law is available at http://www.cybercrimelaw.net/Australia.html.




Japan's cyber crime laws center around Law Number 128 of 1999, often referred to as the "Unauthorized Computer Access Law." This law prohibits acts of unauthorized access to computer systems.

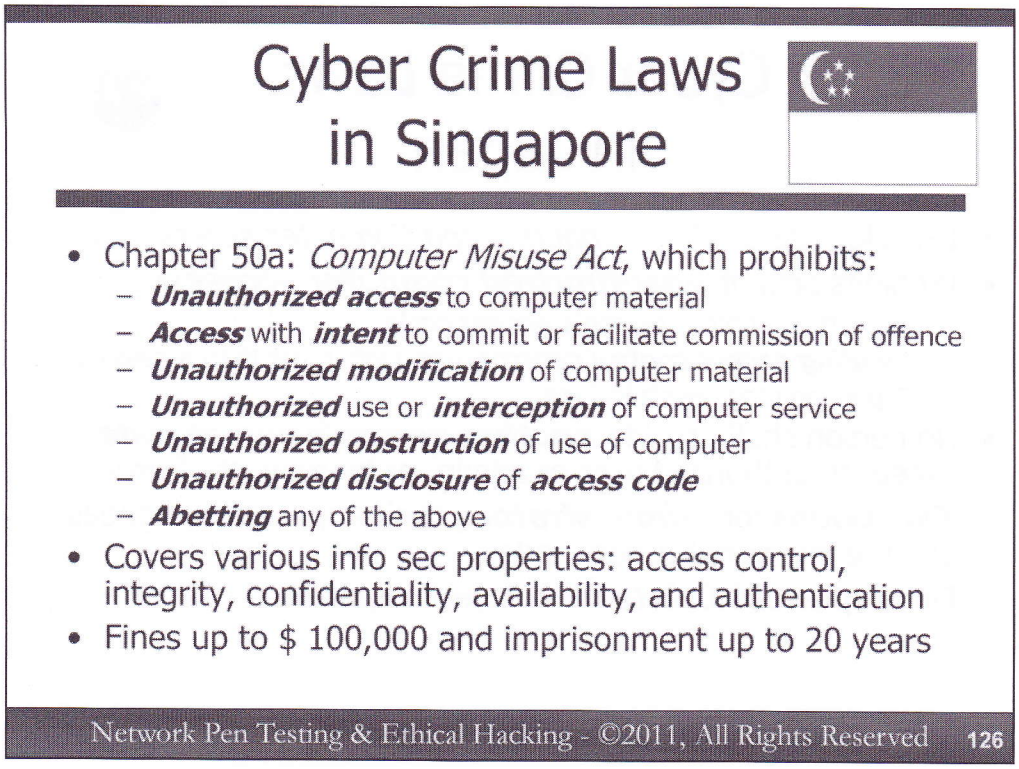
Specifically, it includes Article 3, which deals with unauthorized access to computers identified in the following categories:

- Entering another person's access code into a computer via telecommunications lines
- Entering information other than an access code that evades access control mechanisms, again via telecommunications lines
- Entering information that evades other restrictions on computers via telecommunications lines

According to Article 4, no person shall provide another person's identification code relating to an access control function, unless approved by the administrator, an approved person authorized by an administrator, or the user who owns the identification code.

Penalties, defined in Article 8, include fines ranging up to 300,000 to 500,000 Yen, and imprisonment of up to one year.

Details are available at www.cybercrimelaw.net/Japan.html

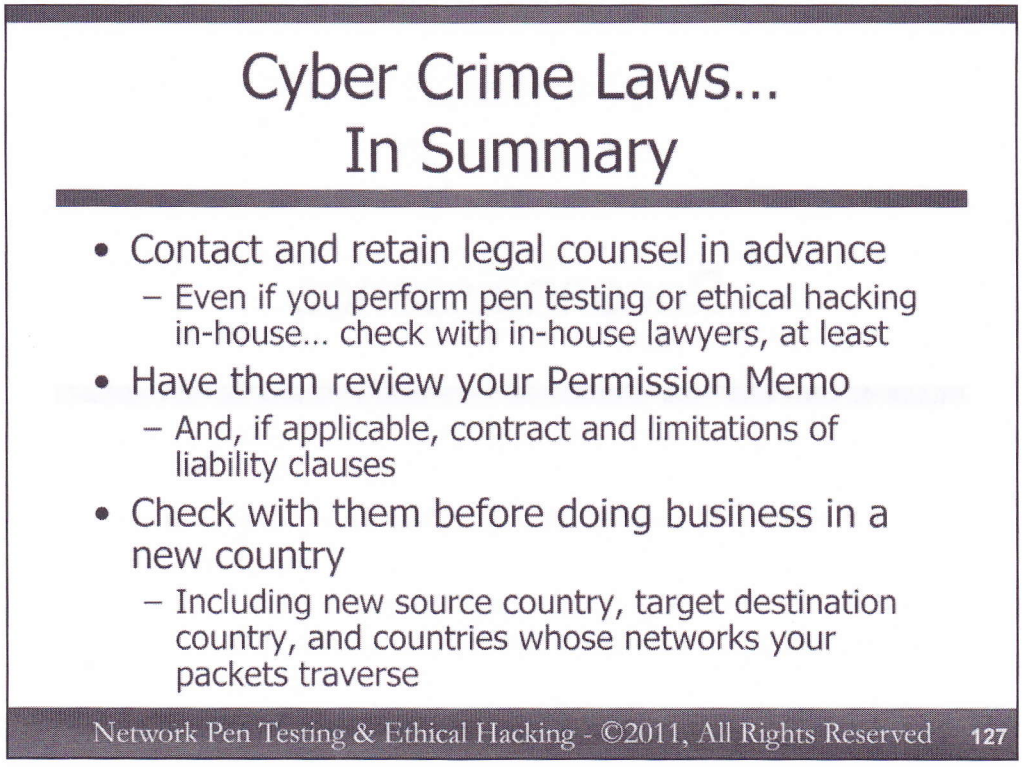



The primary cyber crime law in Singapore is known as Chapter 50a, Computer Misuse Act. This Singapore law specifically cites the UK Computer Misuse Act and the Canadian Criminal Law Amendment Act 1985 as references, showing its influences.

This law carves out a series of offenses that are closely aligned with various information security models describing the aspects of secure systems, including access control, integrity, confidentiality, availability, and authentication. In particular, the law prohibits seven specific kinds of acts:

- Unauthorized access to computer material (which would violate access control mechanisms)
- Access with intent to commit or facilitate commission of offence (which again violates access controls)
- Unauthorized modification of computer material (which violates the integrity of computer data)
- Unauthorized use or interception of computer service (which violates confidentiality)
- Unauthorized obstruction of use of computer (which violates the availability of computer systems)
- Unauthorized disclosure of access code (which would impact the authentication mechanisms in use)
- Abetting any of the above (which would support any one or more of the above)

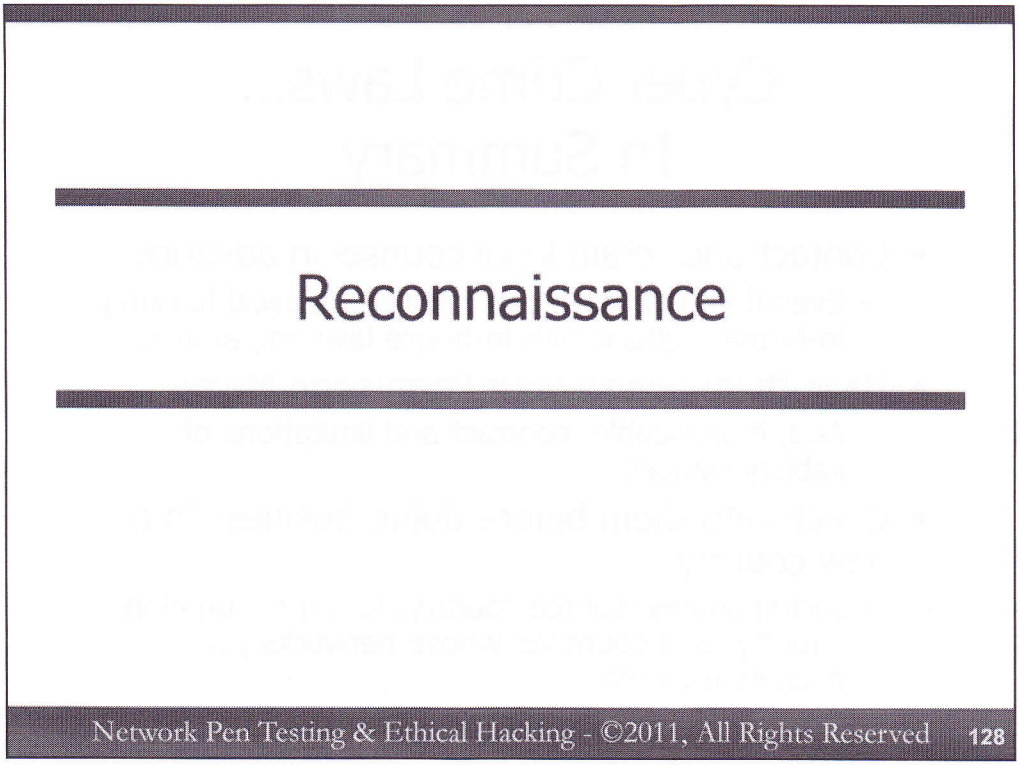
The entire set of Singapore laws, including the Computer Misuse Act, are available at http://statutes.agc.gov.sg, with this law located under section "C", for Computer Misuse Act.




In summary, the legal issues surrounding computer crime and their relationship to penetration testing and ethical hacking can get complex. To help address the situation, we strongly encourage professional penetration testers and ethical hackers to contact legal counsel and retain them in advance, before conducting your first test. Even if you are performing penetration testing, ethical hacking, or vulnerability assessment in-house for your own employer, you should still contact at least in-house legal counsel to verify that it is acceptable for you to perform your work.

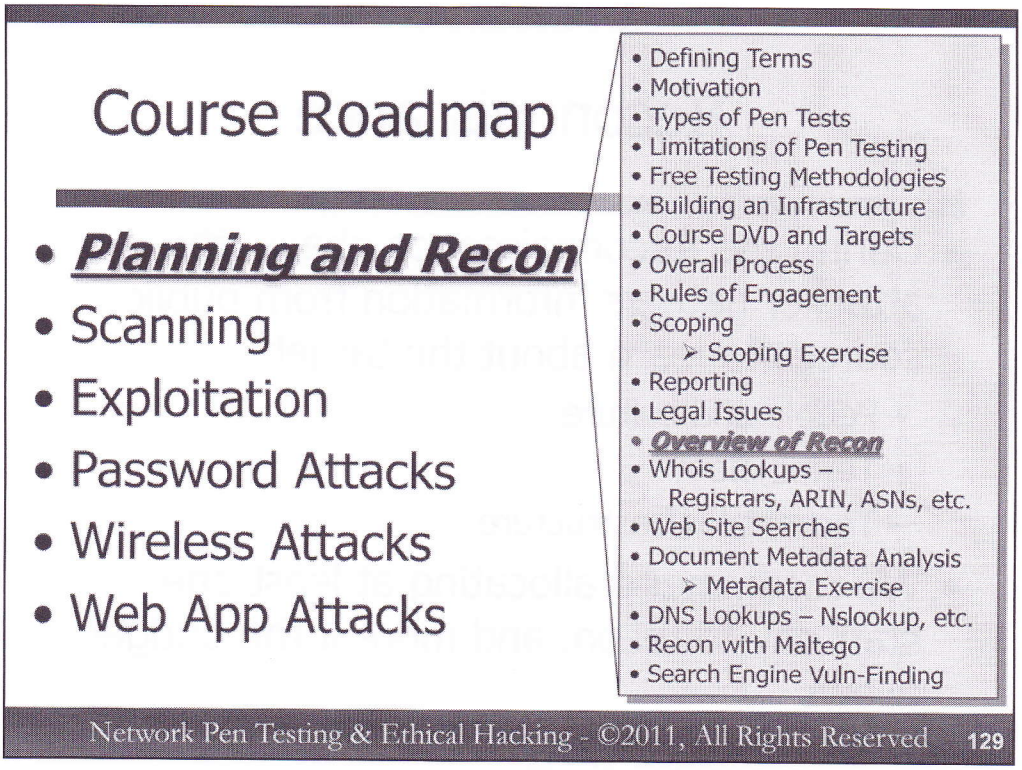
Have them review your Permission Memo. And, if you are performing third-party penetration testing against a client organization, have your lawyer review your contract and its limitations of liability clauses.

Finally, if you are planning on performing penetration testing or ethical hacking in a country in which you haven't previously done business, ask your lawyers for a quick legal review of the given country to check if there are any recent laws or judicial decisions that could impact your work. This applies to countries that may be the location of the source of your test, the destination target of your work, or any countries whose networks your packets will traverse.

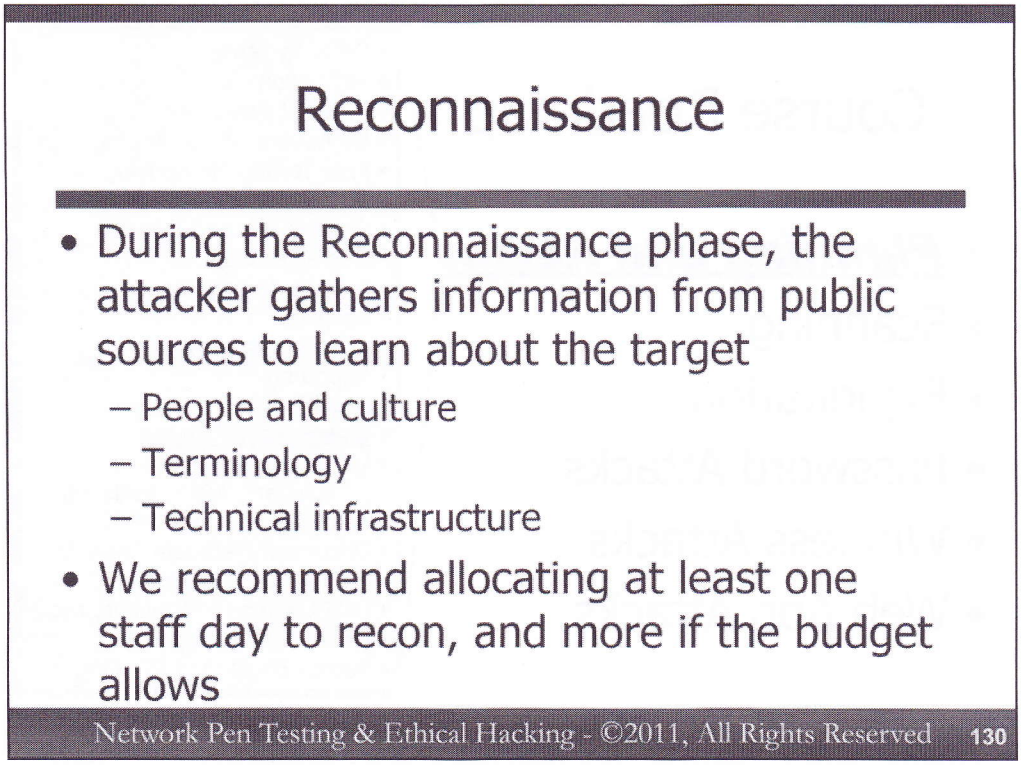



The next portion of the course will deal with the first phase of the test: Reconnaissance. During this phase, the tester will learn as much as possible about the target organization. One important aspect of this recon phase is building an inventory of potential target machines that are likely associated with the target organization. This inventory must be vetted carefully to ensure that it is indeed in scope before any scanning activities (the next phase of the process after reconnaissance) can begin.

When planning a test, we recommend that you budget and schedule at least 8 to 10 hours of detailed reconnaissance work, or more if resources allow. Don't skip recon. It can provide crucial insights for the remainder of the entire test.



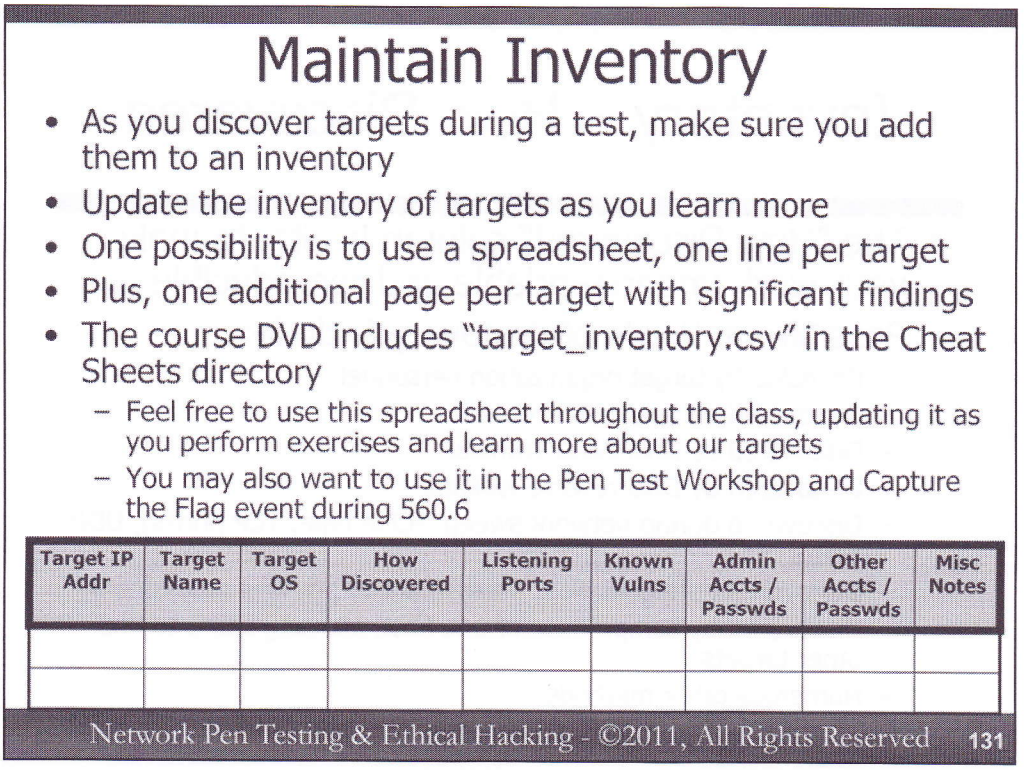

In our next section, we'll start with a brief overview of the Reconnaissance phase, looking at an overview of its goals and methods. We'll then look at some very helpful recon tools, including Internet-accessible search tools and software you can run on your own systems for recon analysis.




After the test has been thoroughly scoped and any required agreements are signed, the test begins with the reconnaissance phase. In this phase, the tester gathers information about the target organization from various public sources. The tester needs to become very familiar with the target's people and culture, learning the specific business terminology used by people in the target organization. We try to find out what is important to the target, and what are they telling the public about how they do business. We also seek tidbits about the technical infrastructure of the target organization, looking for clues about its architecture, products, and configuration in public sources.

This recon phase is extremely important in conducting thorough penetration tests. Don't dismiss it because it doesn't get deep into technology. The information gathered during the reconnaissance phase will be helpful throughout all of the other testing phases, and will be instrumental in the development of the final report.

Given its importance, we recommend that the scope of the test include at least one staff day (eight to ten hours) for reconnaissance and more if the budget allows.



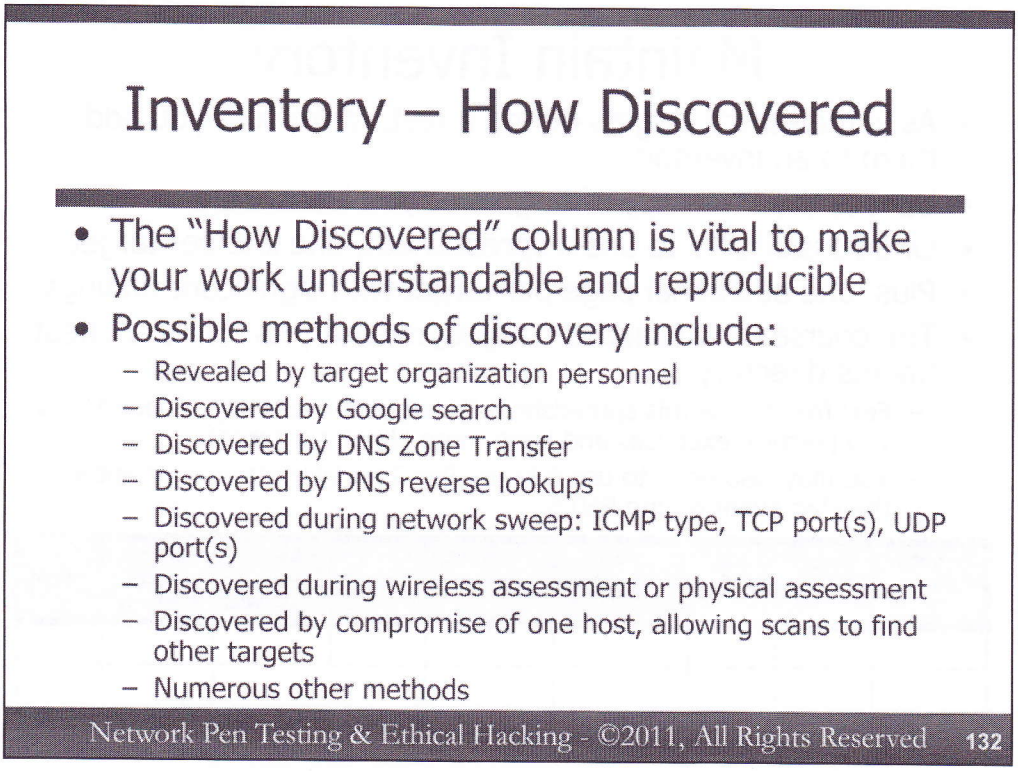

Throughout the test process (starting with Recon and Footprinting, but going throughout the exploitation phase and through the end of the test), it is vital that you record your results in an organized fashion. Disorganized penetration testers and ethical hackers are often far less successful. The last thing you want to do is to miss out on a vital vulnerability in a target organization because it was lost in disorganized clutter. We're not trying to stifle your creativity or limit your style. But, we are trying to keep focus on performing the highest quality tests that we can.

One of the most helpful tools for recording results is an inventory of all discovered targets and their associated details. A convenient way to store this inventory is by using a spreadsheet, such as the one shown on this slide. Each discovered target system gets one line in the inventory, with the details populated as they are discovered throughout the remainder of the test.

The inventory includes numerous fields, such as the Target's IP address, name, operating system type, etc. Some of the most important fields to populate are How Discovered (which we'll cover in more detail shortly), known vulnerabilities, and the accounts and passwords that are determined. Note that you might not populate every field for every discovered target. For example, it is possible that a given discovered target will have an IP address but you will not be able to discover its name. Perhaps the target does not have a DNS record, and you are unable to get access to the box. Thus, you will likely not know its name. Instead of leaving the field blank, we recommend entering "Not Found" or "Not Applicable", to at least show that the given field was not overlooked.

For hosts with more significant findings beyond what is listed in this table, you may want to have a document with one page per significant host. For example, if you are able to compromise a host, getting remote command shell access to it, you might use that shell to take an inventory of that machine's information assets, configuration weaknesses, and the other hosts that it can reach. This information will likely not fit into your spreadsheet, but must be recorded in all of its exciting detail. Store that information on a series of separate pages, with approximately one page per host.

The course DVD includes a file called target_inventory.csv in the Cheat Sheets directory to use as a sample throughout this class.




The "How Discovered" field on the inventory spreadsheet is one of the most important items, as it contains information about how you first identified that the given host was present in the target environment. Recording this information makes your test more understandable (to answer, "Where did all of this information come from?") and repeatable, so that others can verify the integrity of your results.

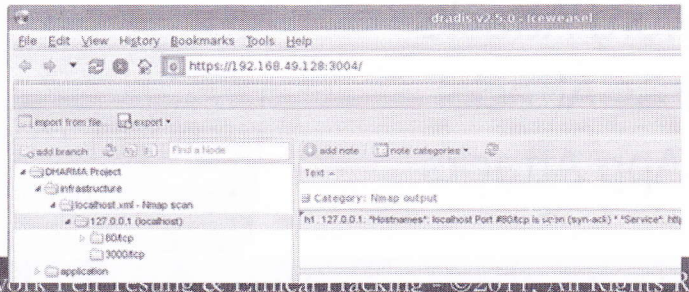
This field may include numerous potential values for how a given host was discovered. In fact, if you want to be thorough, it might include multiple methods as you verified the presence of a given host with alternative means. Some of the methods that will likely be recorded in this field include:

- *Revealed by target organization personnel:* At the outset of the test, the target organization's representatives may provide a list of targets as a starting point. We need to indicated those which were given to us.
- *Discovered by Google search:* Google is a treasure trove of information.
- *Discovered by DNS Zone Transfer:* DNS provides a great deal of information, if zone transfers are allowed.
- *Discovered by DNS reverse lookups:* We can find hosts by doing reverse DNS lookups.
- *Discovered during network sweep -- ICMP type, TCP port(s), UDP port(s):* There are numerous methods for sweeping through a network range to find hosts.
- *Discovered during wireless assessment or physical assessment:* If the test includes wireless, we may find some hosts via this method.
- *Discovered by compromise of one host, allowing scans to find other targets:* This is one of the most exciting methods for discovering hosts – pivoting through one and finding additional targets.
- *Numerous other methods:* This list cannot be exhaustive. There are a huge variety of other methods you may use in finding target machines.

Don't worry if you are unfamiliar with some of the items in this list right now. We'll spend the next several course sections going through how to perform each of these techniques.

## Other Pen Test Inventory, Recording, and Collaboration Mechanisms

- Some penetration testers store their results in a wiki, such as MediaWiki, to support collaboration among testers
- Alternatively, the Dradis tool is designed for collaborative recording and analysis among a group of pen testers, with auto report generation
  - Server runs on Windows, Linux, or Mac OS X
  - Command-line client, several thick clients, and web-based interfaces
  - Organized as a tree hierarchy, storing findings and detailed notes
  - Supports importing Nmap, Nessus, Qualys, Nikto, and Burp scan results





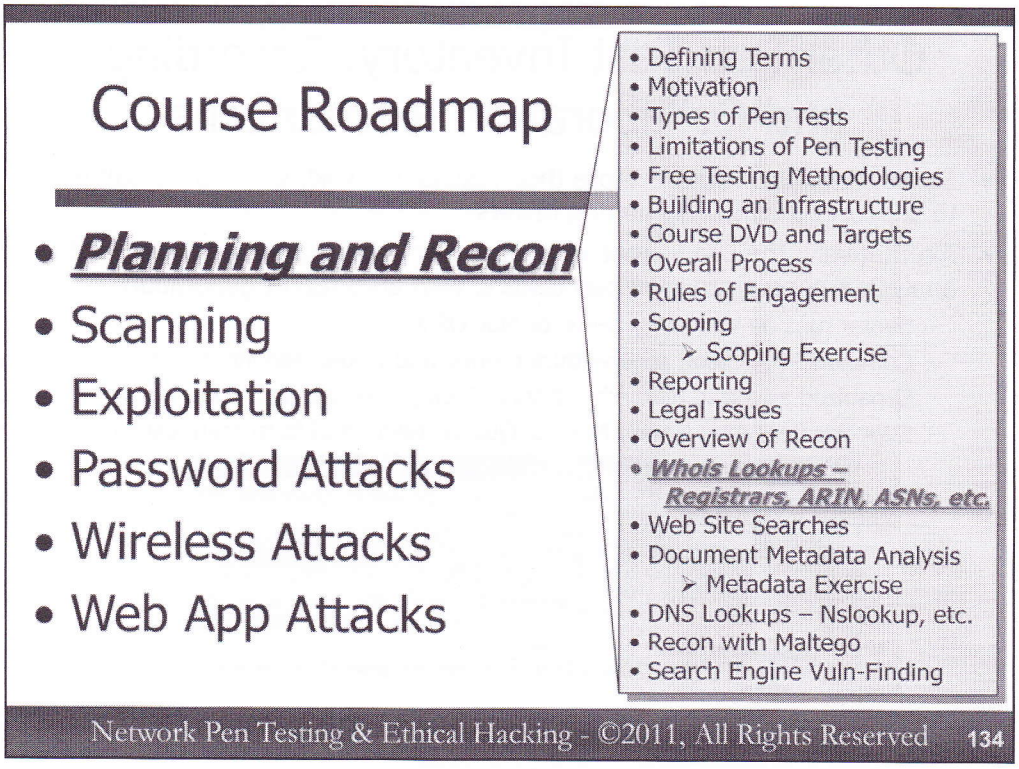

Instead of a spreadsheet and individual documents to store results, some penetration testers prefer an on-line collaborative environment for documenting and analyzing their findings. Such on-line systems are better suited for multi-person penetration tests than individual spreadsheet or word processing documents.

For on-line result storage and collaboration, some penetration testers set up a wiki (a website that allows easy updates of textual information and sharing of files among multiple people). MediaWiki is a common choice, freely available at www.mediawiki.org.

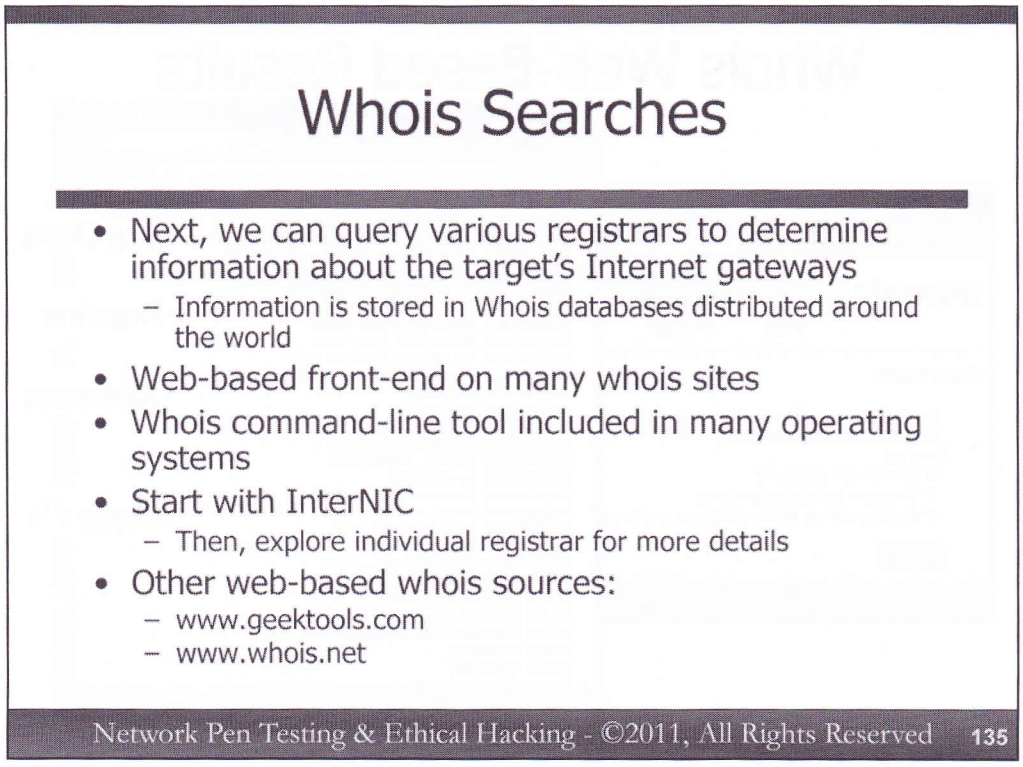
Alternatively, the Dradis tool is a Ruby-on-Rails project designed for recording information among multiple penetration testers working on one or more projects together. The Dradis server runs on Windows, Linux, or Mac OS X, and features multiple client options: a command-line client, several different thick client applications, or a web-based interface.

All results are organized as a hierarchical tree, typically organized starting by overall project, then split according to functional areas of the target infrastructure (e.g., DMZ/intranet/extranet, servers/network devices/clients, or other applicable divisions of the test's scope), then separated by individual devices, down to individual ports on those devices, and then through findings and notes associated with each port.

With Dradis, a tester can import results from the Nmap port scanning tool, the Nessus or Qualys network vulnerability scanning tools, the Nikto web server scanning tool, or the Burp web application attack tool. Additionally, the tester can manually enter findings and notes, or add analytical notes to results already imported.




Our first recon step will be to determine more information about the target domain names that we've gathered from our initial scoping information. We can look up domain names in Whois databases maintained by various domain name registrars and related organizations around the world. Our goal with Whois lookups is to find out about the people associated with these domains, to learn about related domains, to identify the Domain Name Servers used to resolve target names, and to determine the target's IP address assignments. All of this information and more is in Whois.



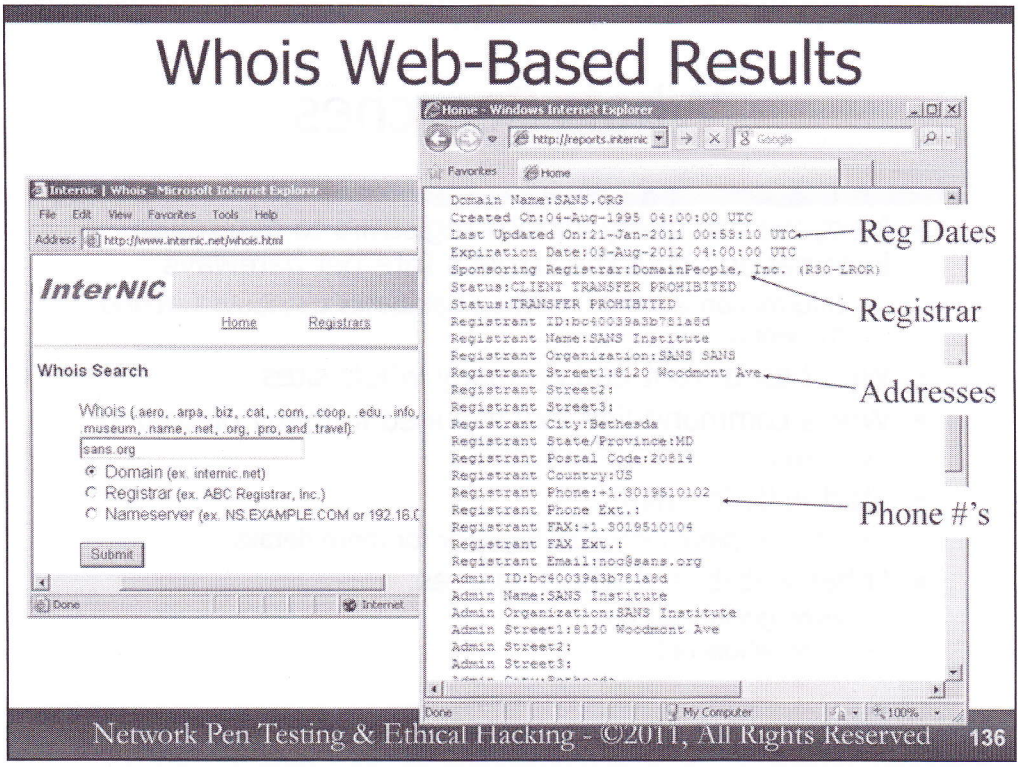

To determine more detailed information about a given target domain, we can look it up using various Whois databases distributed around the world. When a domain is registered, the registrar gathers a significant amount of information about the Internet gateway and people associated with the domain. Most registrars put this information in publicly accessible whois databases. Many of these databases, which are organized in a hierarchical fashion, have a web-based front-end so that they can be accessed via a browser. Alternatively, the whois command built into some operating systems can be used to formulate a whois query.

To start off a whois search using web-based tools, we usually do not know the particular registrar in advance used to register the target domain. Thus, we go to the Internet Network Information Center (InterNIC) Whois front end and perform a query on the domain. InterNIC will either forward the request to the appropriate registrar and return the information to us, or tell us who the registrar is so that we can query that registrar's whois database directly.

There are several websites devoted to getting whois information, each providing a portal that will query various whois servers on the Internet. Some of the most popular include:

www.geektools.com

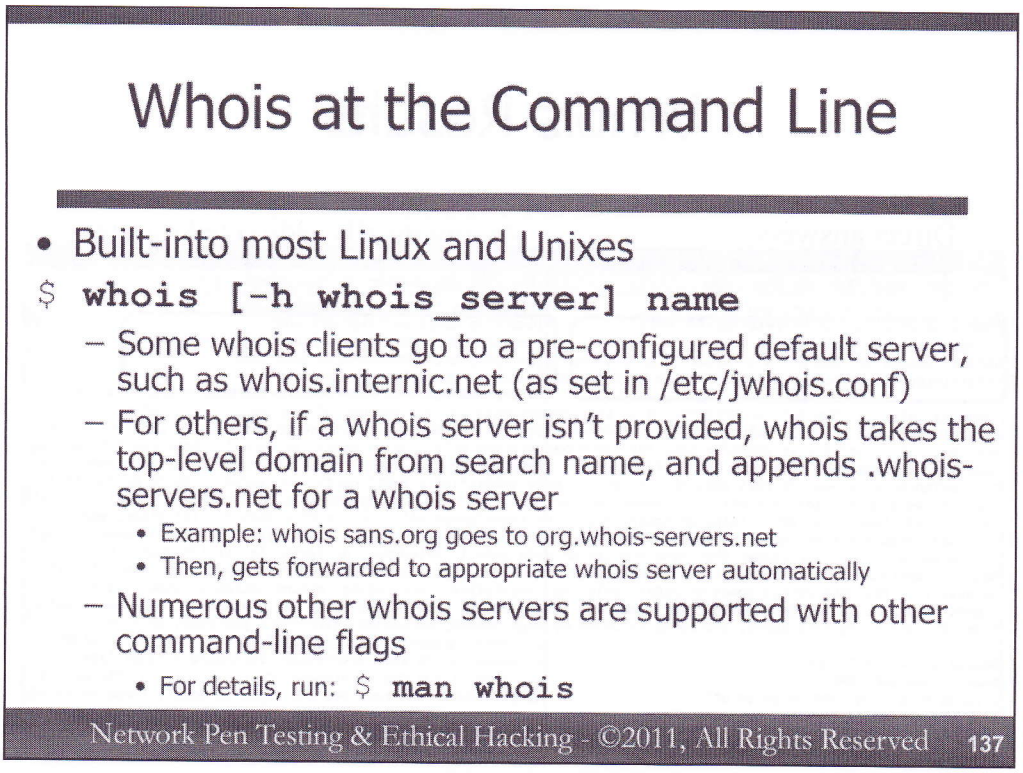
www.whois.net




Here, we have surfed to the InterNIC whois portal, and performed a search on sans.org.

The results show us some very useful information, including:

- The dates when the domain was registered and updated, as well as when it will expire
- The registrar that was used for the domain
- The name, phone number, postal address, and e-mail address of the Registrant, the Admin, and the Tech contact for the domain
- The DNS servers of the domain, listed in primary, secondary, and tertiary (if applicable) order




Most Linux and Unix variants have a whois command. In many Linuxes, the whois command actually invokes the jwhois program.

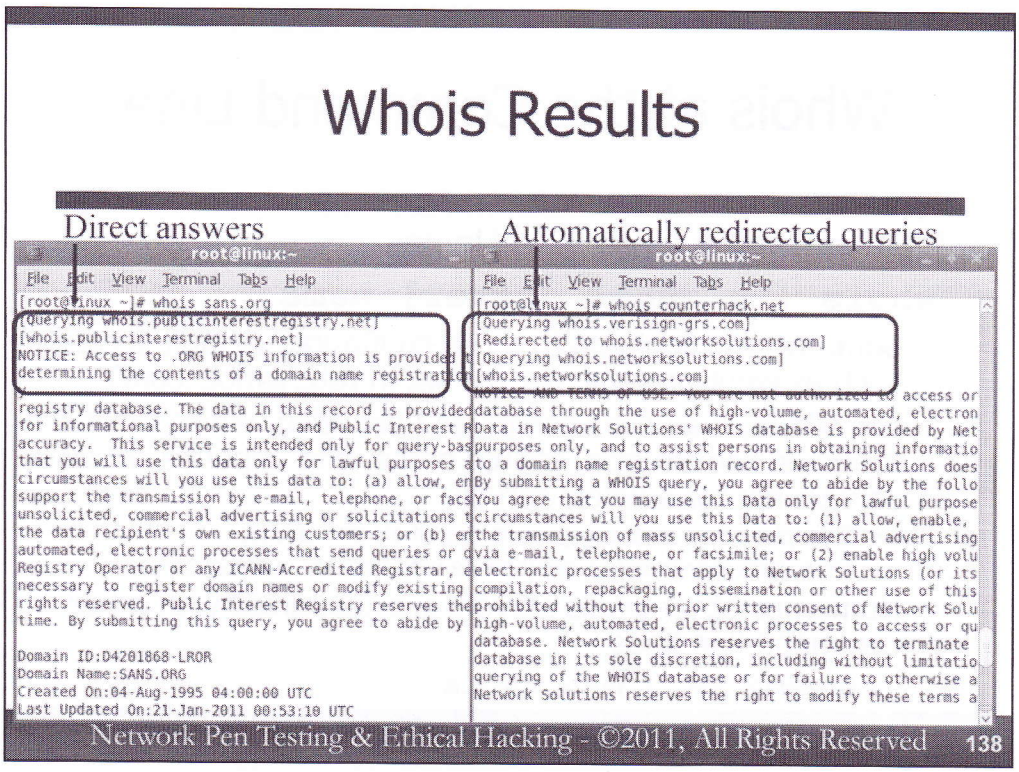
There are many command-line options for whois, but it is most commonly used this way:

```
$ whois [-h whois_server] name
```

The –h option tells the whois client to use a specific whois server to fetch information. If no –h and server are provided, the whois client will default to a whois server. The default server depends on the specific whois client being used. In many Linuxes, the /etc/jwhois.conf file is consulted, which will tell the client which server to use based on the top-level domain (.com, .org, .uk, .jp, etc.) that is being searched for. In many BSD-derived Unixes, the whois client takes the top-level domain suffix and appends .whois-servers.net to it as a default server for that domain.

The whois database may then return the information back directly, or redirect the request to another whois server.

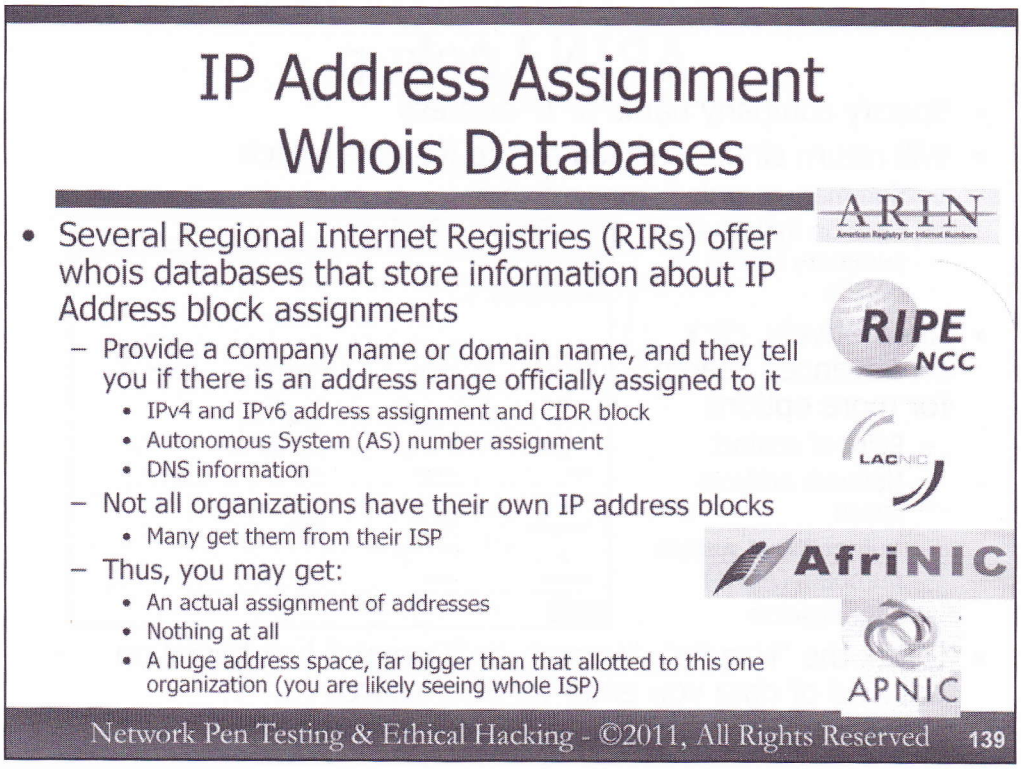
There are several other command-line flags associated with directing whois clients to different servers which often vary from system to system. These can be viewed in the whois man page.




In this slide, we've done whois searches from Linux for the domains sans.org and counterhack.net.

The whois client in this Linux version automatically sent the request for sans.org to publicinterestregistry.net. It then pulled the results from that whois database.

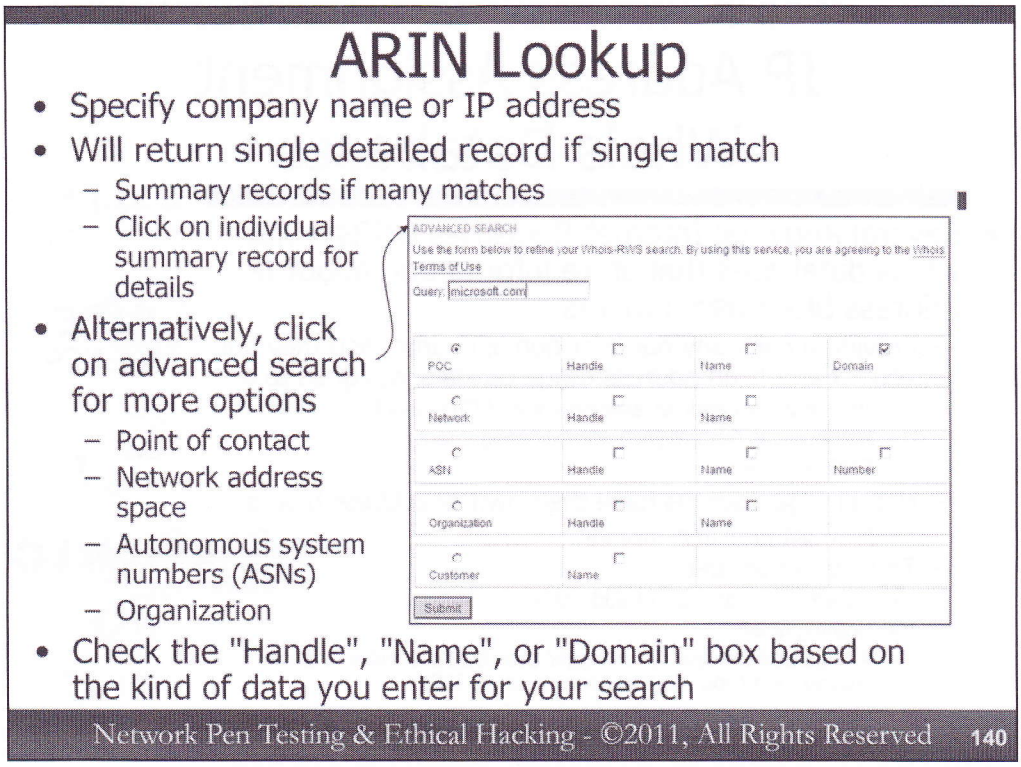
The whois client sent the request for counterhack.net to whois.verisign-grs.net, which then redirected it automatically to whois.networksolutions.com, where it found its answer. The query and redirect information is displayed on the screen.




Another important element of reconnaissance involves determining the IP address blocks that are assigned to the target organization. There are several Regional Internet Registries (RIRs) that store this information in whois databases. By surfing to the appropriate website, a user can provide a company name or domain name and retrieve official address assignments, including IPv4 and IPv6 addresses. Most records also include the CIDR (Classless Inter-Domain Routing) block, telling us the size of the target network. The American Registry for Internet Numbers (ARIN) covers North America, including the US, Canada, and certain Caribbean islands. The Réseaux IP Européens Network Coordination Centre (RIPE NCC) is the RIR for Europe, the Middle East, and parts of Central Asia. The Asia Pacific Network Information Centre (APNIC) covers the Asia-Pacific region. The Latin American and Caribbean Internet Address Registry (LACNIC) encompasses Latin America and most of the Caribbean. AfriNIC covers the continent of Africa.

Also, these databases provide autonomous system (AS) numbers, sometimes known as ASNs. An AS is a collection of IP networks and their associated routers under the control of a single technical administrator, such as an ISP or enterprise, that has a common routing policy with respect to the Internet. The AS will have its own internal routing policy, but presents a separate routing policy to the Internet, which moves packets between various autonomous systems using a routing protocol, like the Border Gateway Protocol (BGP). Each AS is assigned a unique ASN, which is stored in the Regional Internet Registries. These databases also store DNS information.

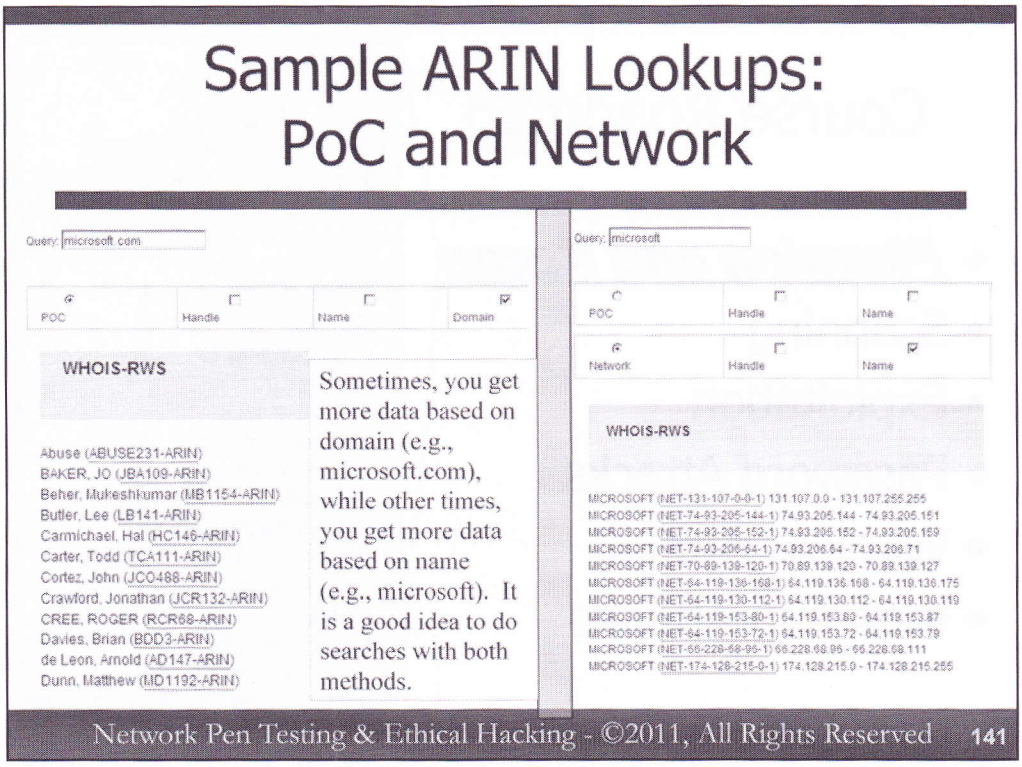
Not all organizations have an IP address block assigned to them. Some get IP addresses from their ISP. When searching a Regional Internet Registry for information, we may therefore not get exactly what we are looking for. When searching for an IP address block for a company, for example, we may get the actual results for that company, certainly a good thing. Alternatively, we may get nothing at all, implying that the given enterprise gets all of its IP address space from its ISP. Thirdly, we may get a giant block of addresses back which do not apply to only the organization we searched for, but instead apply to their entire ISP. Thus, we have to be careful when targeting the results we receive from a Regional Internet Registry, verifying that they are actually within the scope of our test.




Let's focus on ARIN for some search examples. By surfing to www.arin.net, we are presented with a field to enter data for a whois search. By providing a company name or IP address, we can get information about the associated organization.

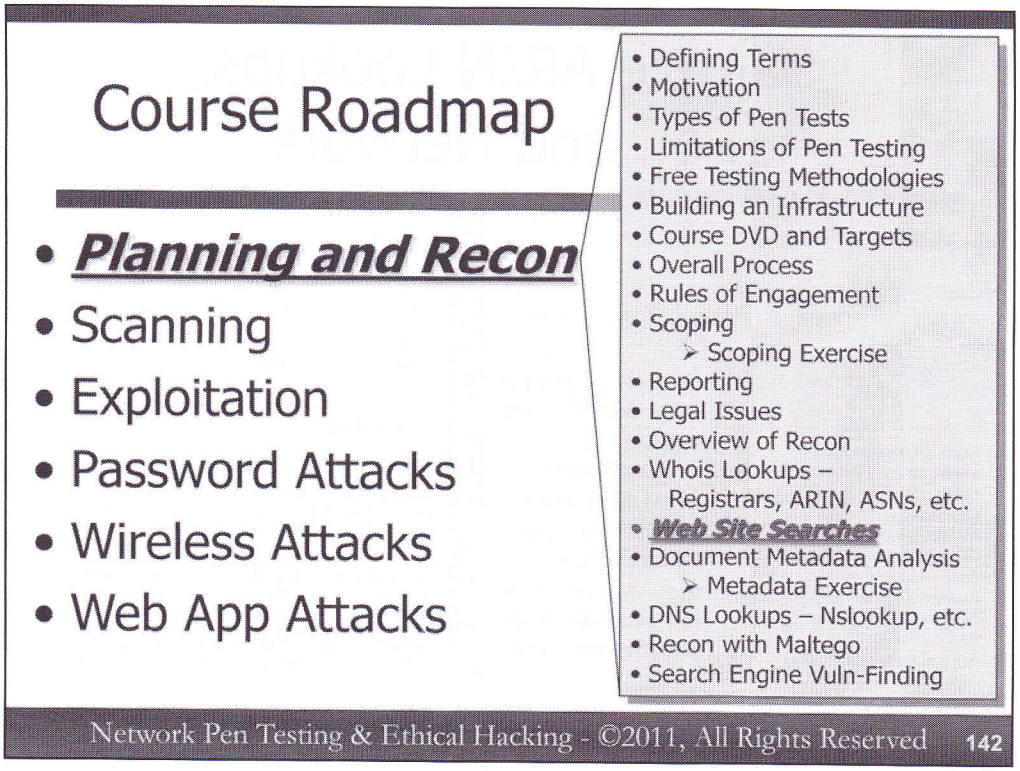
ARIN will respond with summary records if there are many matches, with each summary taking one line and providing a link for more detailed information. If there is only one record that matches our query, we will see the detailed data returned without a summary.

We can focus our searches to look for specific record types within ARIN by clicking on "Advanced Search". This option brings us to a page which lets us search for: points of contact for a provided company or domain, network address space allocated to the organization, Autonomous System Numbers (ASNs) of their groups of routers used in the Border Gateway Protocol (BGP), or organization details. Enter the data you want to search on (such as microsoft.com), select the radio button for the type of data you want to query (PIC, Network, ASN, etc.), and then check the box based on the type of data you've entered for your search, such as a Handle (a reference to various objects in ARIN), a Name (such as microsoft), or a Domain (such as microsoft.com). Click Submit to conduct the search.

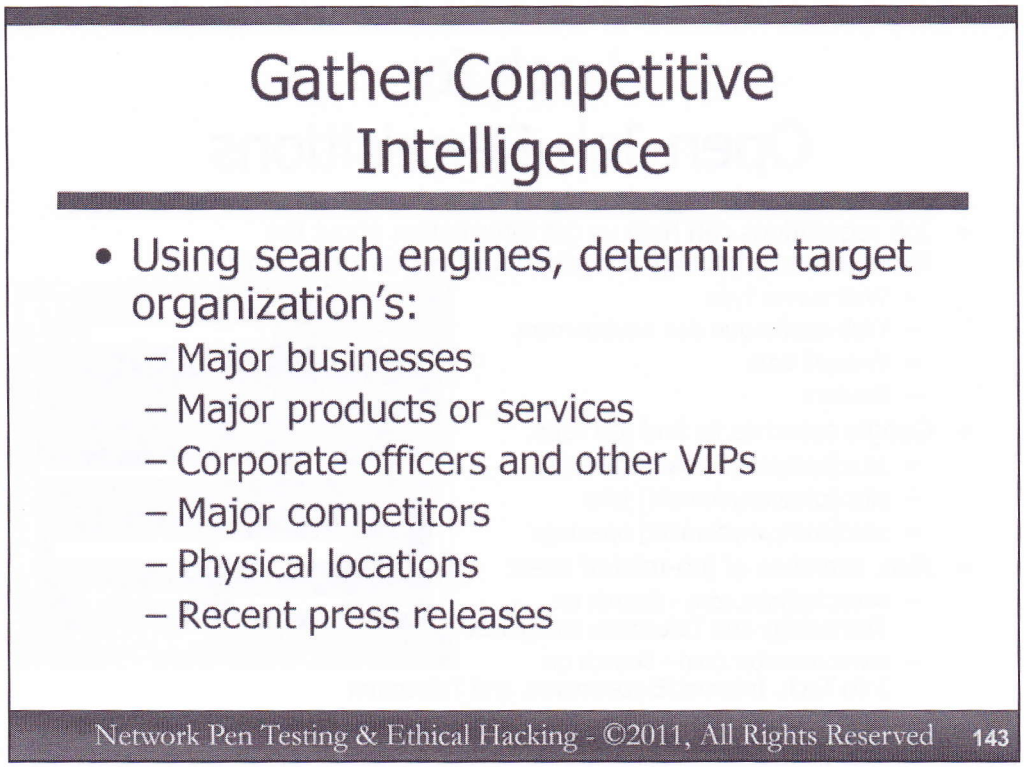



The screenshots on this slide show a Point of Contact (POC) and Network search associated with Microsoft. Note that ARIN searching is case insensitive. For the POC search on the left, we focused on entering in a Domain (microsoft.com), because it gave us more results than searching based on the name "microsoft". For the Network search on the right, we received more results by searching for the name "microsoft" than we did searching for the domain "microsoft.com". It is a good idea to do searches for both name and domain to help ensure you get the data you need for your reconnaissance phase.

We can see that Microsoft has a huge number of points of contact and network ranges.




Our next step in the reconnaissance phase will be to learn about the given target organization through searches in publicly available information sources. The world-wide web is a treasure trove of information, highly usable by us as testers. By searching appropriate websites directly affiliated with the target, as well as third-party search engines, job sites, blog postings, and so forth, we can build a complete dossier of information about the target organization.

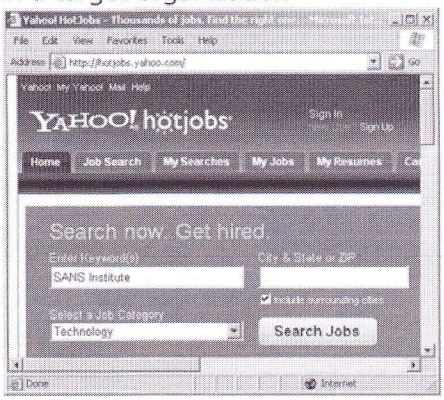



As a start of the recon phase, the tester can use a search engine such as Google to learn more about the target organization. In particular, we recommend conducting searches on the target organization's name to gather the following information, which should be recorded in the tester's results:

- *Major businesses:* What is the industry or industries associated with the target? Financial services? Government agency? Manufacturing?
- *Major products or services:* What does the target organization produce? What are the brand names of its products or services?
- *Corporate officers and other VIPs:* Who is most important in the target organization? Who are its leaders? Who is associated with its technical infrastructure?
- *Major competitors:* Who competes with the target organization? What is the target organization's relative performance vis-à-vis its competitors? Is it the market leader?
- *Physical locations:* Where are the major facilities of the target organization?
- *Recent press releases:* What has the target enterprise told the public lately about itself? What do they consider important from an image and marketing perspective?

# Look for Open Job Requisitions

- Job requisitions can help us get information about the information technology products used in a target organization
  - Web server type
  - Web application dev environment
  - Firewall type
  - Routers
- Google searches to find job reqs:
  - site:[companydomain] careers
  - site:[companydomain] jobs
  - site:[companydomain] openings
- Also, searches of job-related sites:
  - www.hotjobs.com - Search on Technology and Telecomm categories
  - www.monster.com - Search on Info Tech, Internet/E-commerce, and Telecomm





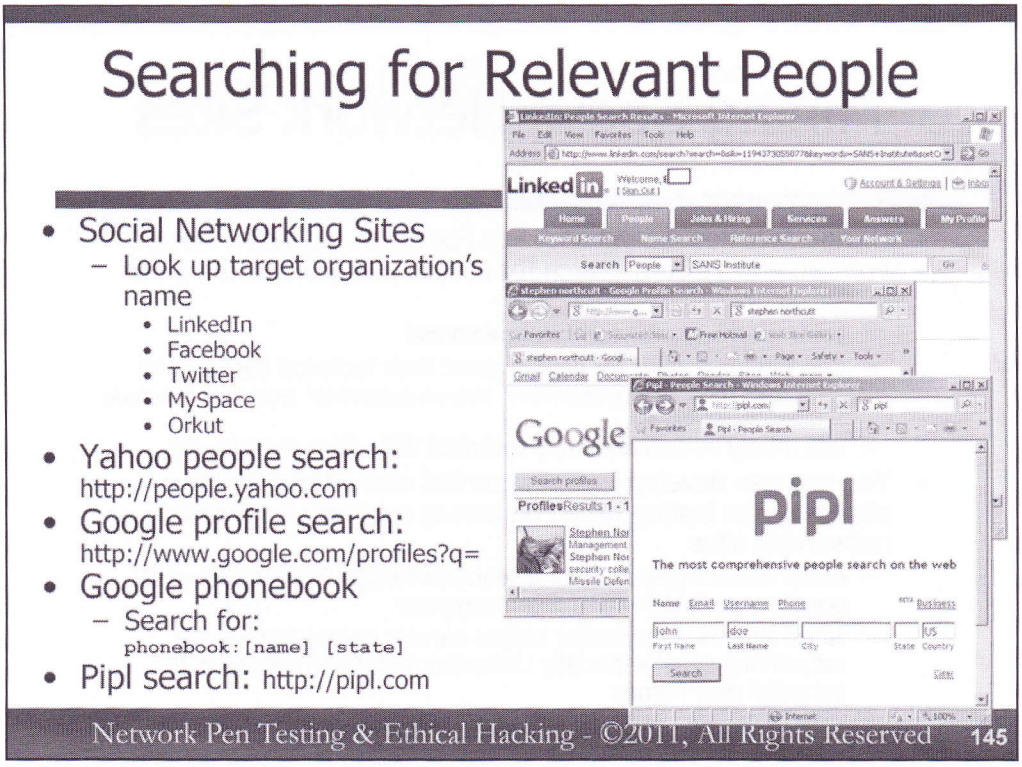

Most organizations have job requisition information available on the Internet, as they look to hire new staff. These job requests often contain detailed information about the technical environment of the enterprise. For example, if the target organization is looking for IIS administrators, we now know something about the web servers they use. If they seek skilled Checkpoint firewall admins, we have information about at least some of their firewalls. If they are looking for developers with Cold Fusion experience, we now know a little more about some of their web applications. What's more, if the job req is still active, we know that the target organization does not have enough experienced staff members to handle that part of their infrastructure. After all, if they did have the expertise already in house, why would they be seeking to hire people with those skills?

To search for job requisitions, you could use Google with the "site:" directive focused on the target's domain, followed by common terms used on pages for hiring. We recommend searches like the following:

```
site:[companydomain] careers
site:[companydomain] jobs
site:[companydomain] openings
```

You can narrow down your results further by inserting terms like Information Technology, Internet, e-commerce, firewall, and so forth.

Additionally, a thorough tester should look for job reqs on various job-hunting sites, like Yahoo's Hotjobs.com and Monster.com. Both of these sites let you search based on categories of jobs. For Hotjobs, both the Technology and the Telecomm categories are useful searches for our purposes. On Monster.com, the Information Technology, Internet/E-commerce, and Telecomm categories are helpful. You can even narrow down your searches based on geographic areas.



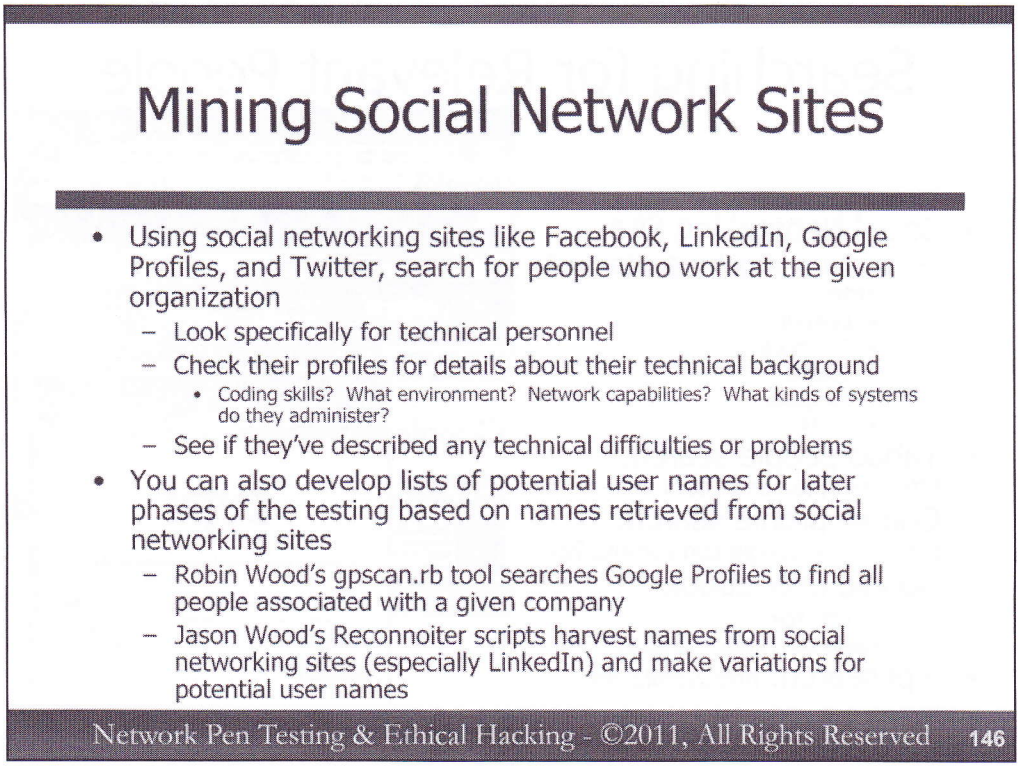

Other helpful areas to search are social networking sites. People put a significant amount of information about themselves on these sites, often including where they work. That employer information is exactly what we are looking for. By searching within the social networking site for people who work for the target enterprise, we can then focus in on their background and skill set. We may find, for example, that John Doe used to work in the organization ABC Widgets. Looking at John's profile, we may find out that he has developed applications that involve Microsoft's SharePoint Server product. Those are very useful nuggets of information.

LinkedIn and Orkut, with their greater appeal to professionals, often contain the best information for our purposes, but a quick search on MySpace and Facebook may also reveal useful data. Twitter is also useful in finding people associated with a given target and learning about their interests and work habits.

Beyond the social networking sites, Yahoo offers its people search capabilities, which scour numerous information sources to pull together address and phonebook information about given individuals. By searching for people associated with the target organization (including security personnel, other IT people, corporate officers, etc.), more useful information can be gleaned. These phone numbers could be useful in social engineering and modem scans.

Google profiles are also a useful source of reconnaissance information, based on searches of companies or individuals. Google also supports phonebook searches by using the "phonebook:" directive, followed by a name and state. Google's phonebook information tends to be US-centric, although they are starting to add information for other countries.

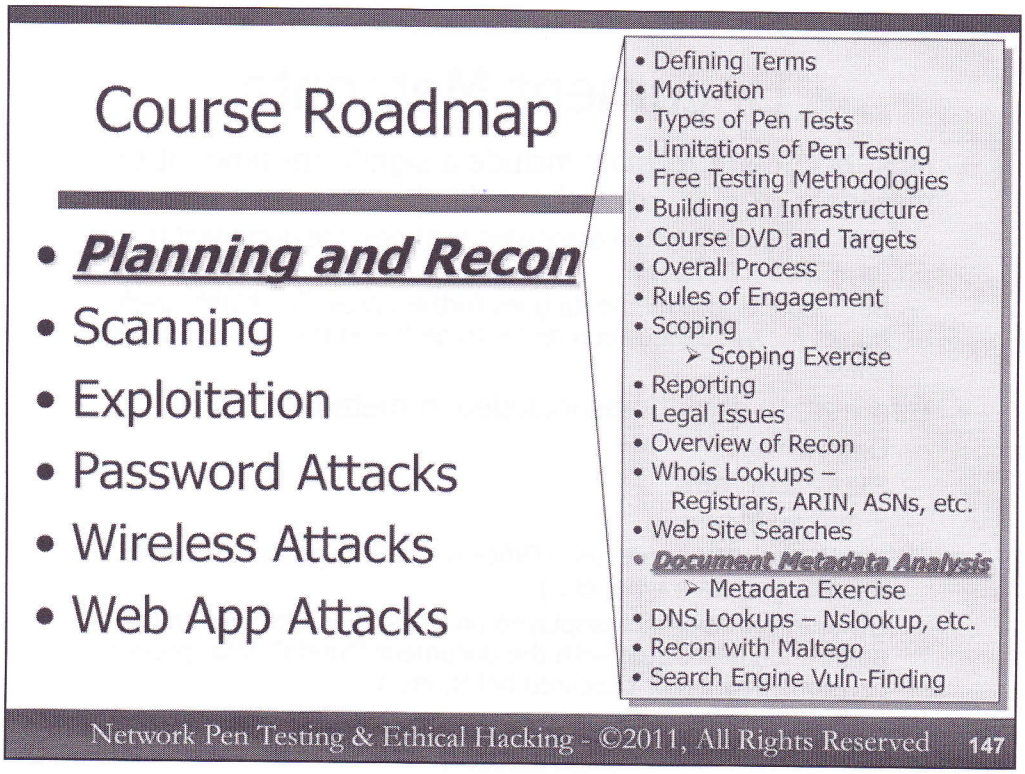
Finally, there is Pipl search, a web site devoted to pulling publicly available information about people. The site includes ages, addresses, estimates of income, and much more. They also include links to commercial services to pull more information for a fee, but even their free searches can be quite enlightening.



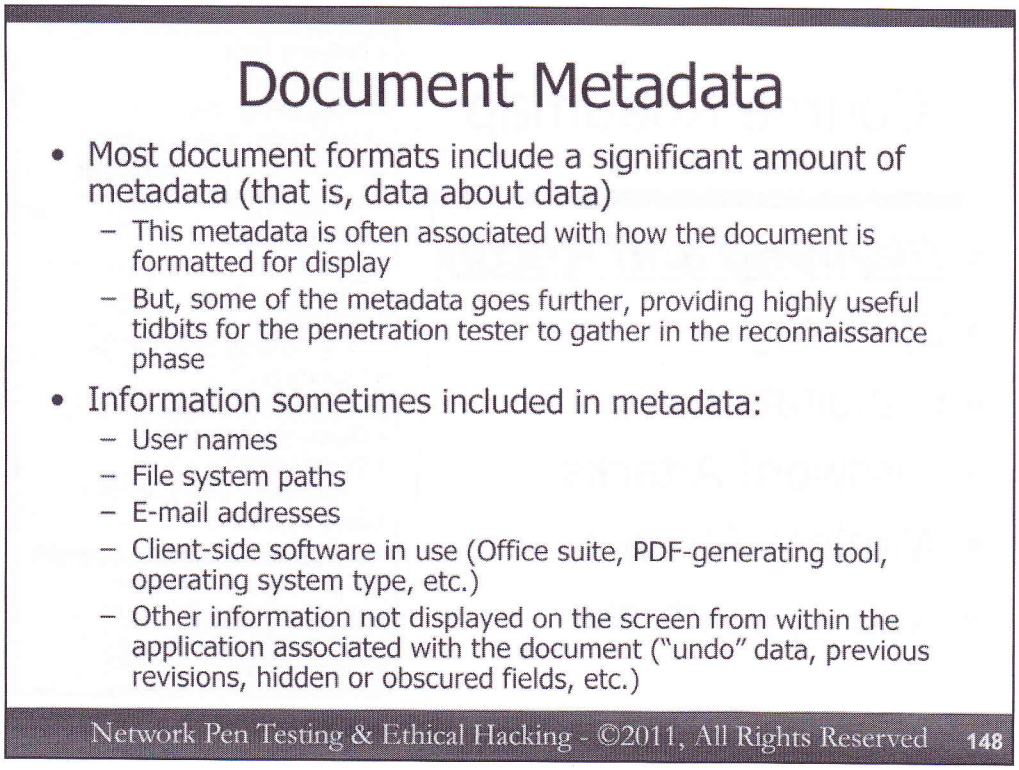

Many people reveal a surprising amount of information about themselves and their employers via social networking sites. A penetration tester can conduct very useful recon by searching for specific company names in Facebook, LinkedIn, Google Profiles, Twitter, and other social media sites. Current and previous employees of the target company may include their relationship to the company in their profile.

It can be especially useful to identify technical personnel, who may reveal details of their experience with various platforms that the penetration tester can target. Try to determine software developer's coding skills and development environment. Look to see which platforms various system administrators are familiar with. And, look carefully to see if there is any indication that someone is having or has had trouble configuring technical devices, perhaps posting requests for help on their walls or status pages.

Various tools can help automate the process of harvesting users associated with a given target enterprise from social networking sites. In particular, Robin Wood's gpscan.rb tool takes a company name on its user input and then searches Google Profiles to pull all users associated in some way with that company. Jason Wood (no relation to Robin) wrote a series of scripts he collectively calls Reconnoiter that harvests peoples' names associated with a given company in social networking sites like LinkedIn. It then creates various potential user names by making variations of these harvested names. Such a list may be useful in password guessing, which we will discuss in detail in 560.4.




Another very useful source of information during reconnaissance is the metadata stored inside of documents that penetration testers can gather from the target's website and target organization personnel. The next section of the class will analyze the topic of document metadata and how penetration testers can use various tools to gather and analyze it during the reconnaissance phase of a test.




As organizations create documents, the software that they use to create these documents embeds an enormous amount of information in the document files. Of course, much of this information is the contents of the file itself. But, a good deal of metadata (that is, data about data or data describing other data) is also included in the file. Much of this metadata is associated with formatting and display of the other data in the file. Besides this formatting metadata, a lot of file creation and editing tools include additional metadata entries that can be very useful for penetration testers during our reconnaissance phase, such as:

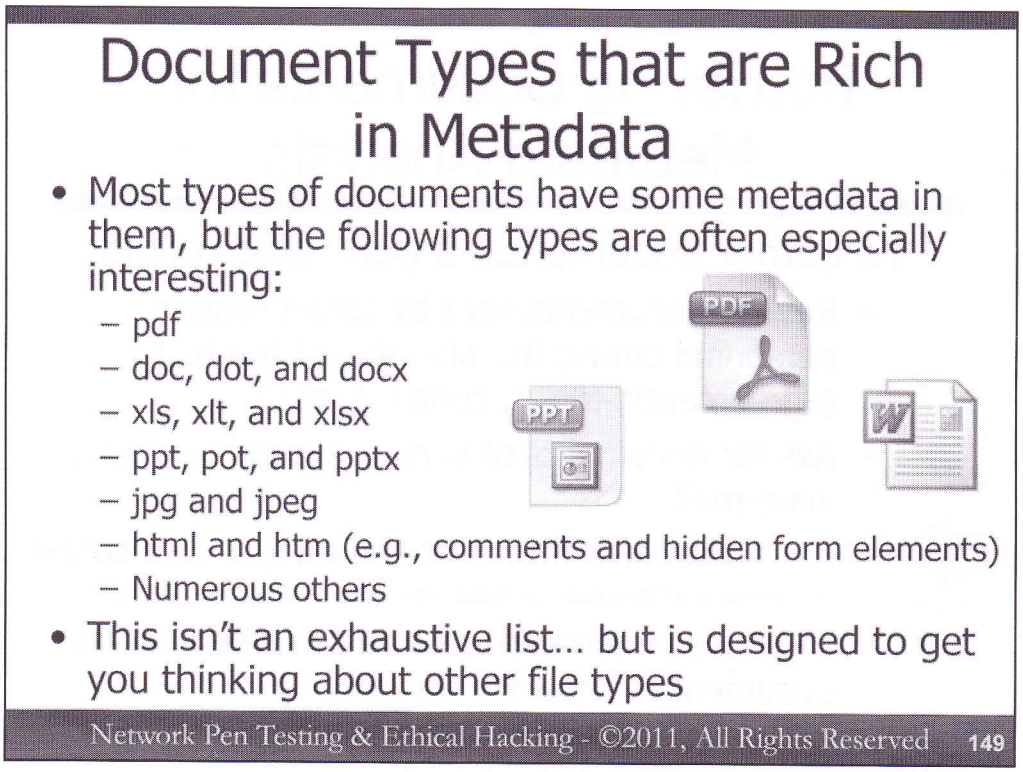
*User names:* Penetration testers often need user names for exploitation and password-guessing attacks, which we'll cover in 560.3 and 560.4 respectively.

*File system paths:* Knowing the full path of the original file when it was created can reveal useful tidbits about the target organization, including hints about important commonly mounted file servers, critical directories, and common practices of the given user.

*E-mail addresses:* This data can be useful if the penetration test scope includes spear phishing tests (sending e-mail to target personnel to see if they will click on links or open attachments). However, such tests should only be performed if they are explicitly allowed for the given target personnel explicitly included in the scope of the test.

*Client-side software in use:* Given that client-side exploitation is such a common attack vector, it can be helpful to penetration testers to know which client-side programs are in use, including the office suite, PDF-generating tool, and even operating system type. Metadata often also reveals version numbers of this software, but those versions were in effect when the document was created or last edited, and are not necessarily the current version.

*Other information:* Other useful information is often associated with content of the document that isn't displayed on the screen within the application itself, such as undo information, previous revisions, and hidden or obscured information (such as a collapsed column obscured in a spreadsheet, or critical text in the document hidden under a picture).




Almost every document type has some form of metadata, but some are richer in metadata than others. The following types of documents, generated and used by most enterprises, are of particular interest to penetration testers:

*pdf files:* These files are associated with Acrobat Reader and a variety of other pdf creation and editing tools.

*doc, dot, and docx files:* These files are associated with Microsoft Word, but are also used by several other related tools. While doc and docx are content files, a dot file is a template often used as a source to create other documents.

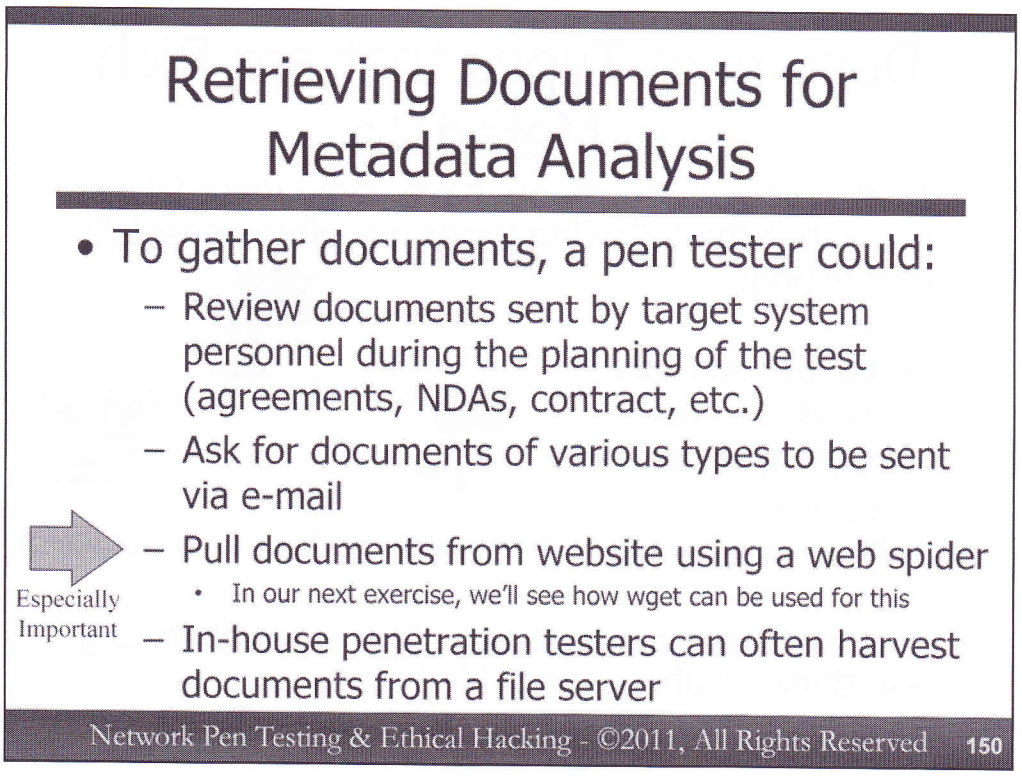
*xls, xlt, and xlsx:* These are common spreadsheet files, often associated with Microsoft Excel.

*ppt, pot, and pptx:* These files are associated with Microsoft PowerPoint and other slide-generating programs.

*jpg and jpeg:* These image files often contain a significant amount of metadata, including data about the camera used to take a picture, the file system of the machine where the image was edited, and details about the image-editing software.

*html and htm:* These file types contain web pages, and may at first seem uninteresting. However, their comments and hidden form elements could contain metadata that is very useful to a penetration testers. Additionally, scripts embedded in the HTML may reveal sensitive information or undocumented features of a web application.

Besides these types of documents, there are hundreds of others that may be interesting. The list above is not intended to be exhaustive, but is instead designed to get the reader thinking about interesting and useful document types to analyze during a penetration test.




To perform metadata analysis, a penetration tester must first retrieve files to analyze. Numerous methods could be applied to gather these documents.

First off, the penetration tester may have already received some documents generated or edited by target system personnel during the planning of the testing project. For example, the tester may have received rules of engagement agreements, scope information, diagrams, non-disclosure agreements, contracts, policies and procedures, and other information.

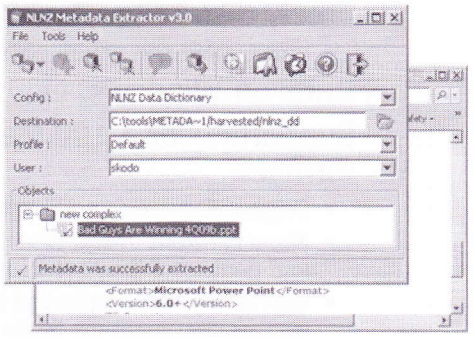
Additionally, a penetration tester can simply ask target system personnel for documents. During the recon phase of the test, you could ask them to e-mail you a document created by their word processor, spreadsheet, and slide editing tools. You could also request a PDF document from them.

One of the most common and especially important methods for harvesting documents for metadata analysis is to use a web spider tool against the target organization's website, pulling all potentially interesting documents onto the penetration tester's machine for analysis. In our next exercise, we'll discuss how the wget tool can be used for this.

And finally, if the penetration test is being conducted by in-house testers (employees of the same target organization), they typically can get an ample supply of documents for analysis from file servers in the organization.

# Tools for Analyzing Document Metadata

- There are dozens of different tools for analyzing document metadata
  - ExifTool
  - FOCA
  - Metadata Extraction Tool by the National Library of New Zealand (NLNZ)
  - Strings
  - Numerous others
- We'll look at two of the most useful and popular: ExifTool and Strings

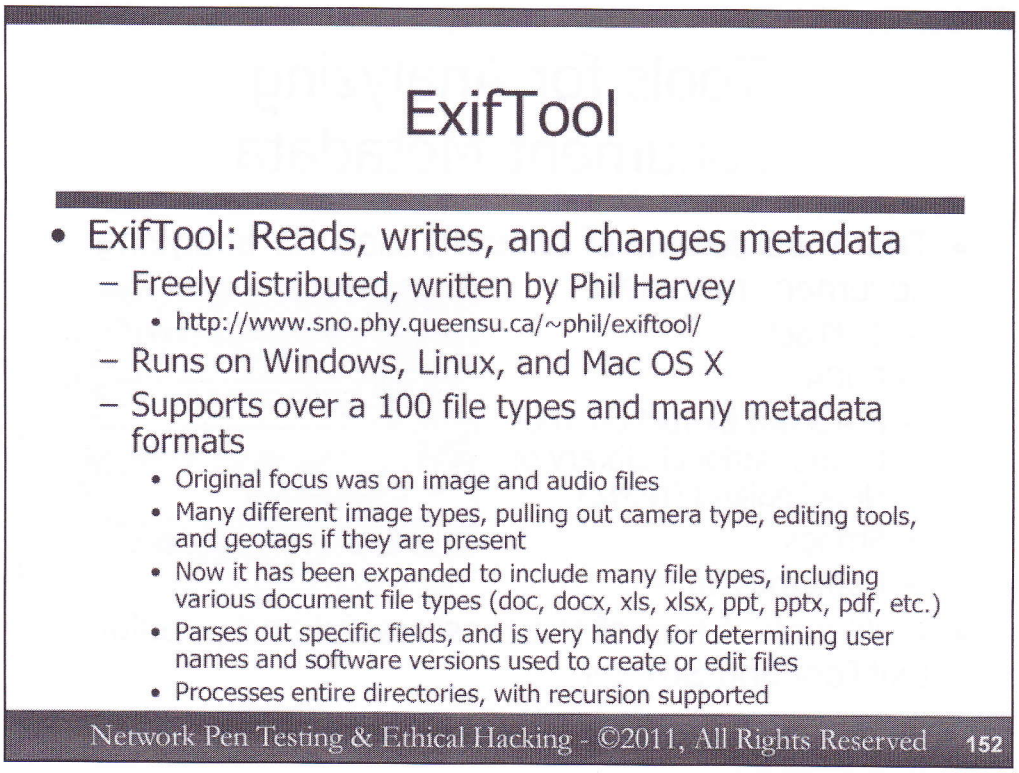





There are a variety of tools to extract metadata from documents. Some of the most powerful include ExifTool, FOCA, and the Metadata Extraction Tool by the National Library of New Zealand. All of these tools are free, and are focused on extracting specific, enumerated types of metadata for a specific set of files. That is, these tools pull structured metadata from documents that are organized in a specific format with specific locations and/or specific tags for the metadata.

On the other hand, the strings command pulls all strings from a given file, regardless of the structure. Strings isn't just focused on metadata, but can pull various types of information with an unknown structure from any kind of file. The strings command will often find metadata from files that other tools (which do not recognize the structure of the file) cannot find, but will bury it in a barrage of output with other uninteresting strings. Searching through the output of the strings command can be difficult, but it often provides some very useful tidbits.

We will zoom into two tools commonly used for metadata analysis, one focused on finding structured data in various known file formats (ExifTool) and the other focused on unstructured data (the strings command). We'll cover each tool and then perform an exercise using them.



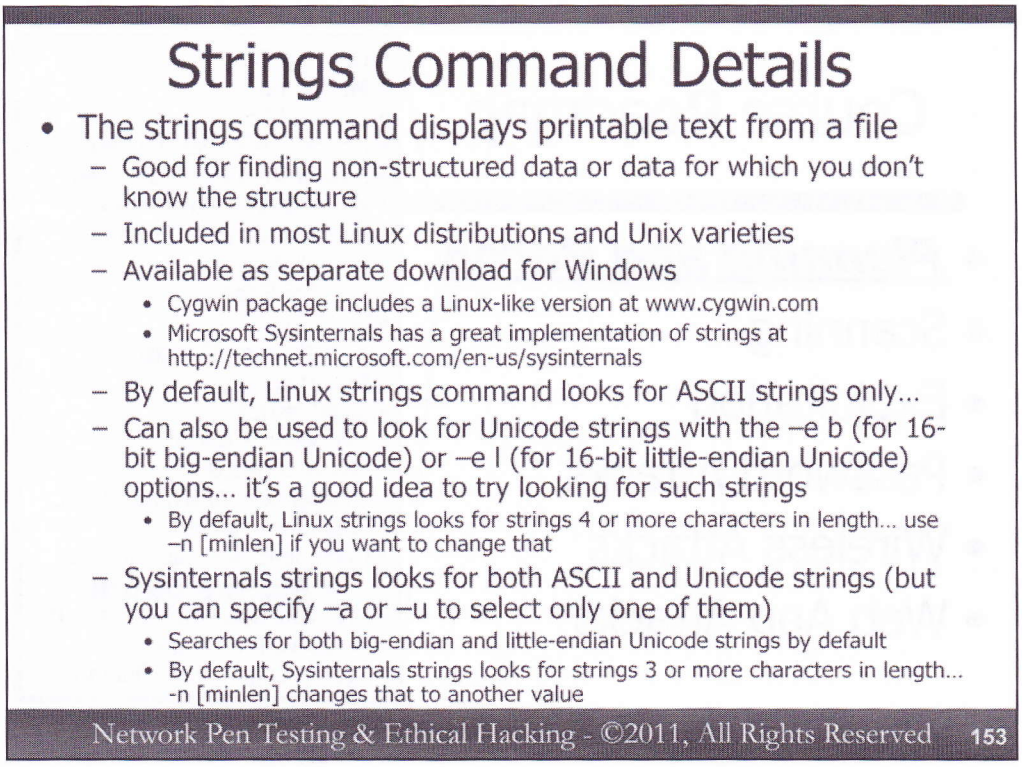

The ExifTool program focuses on reading, writing, or editing the metadata in over one hundred different file types, including images, audio files, videos, office documents (doc, dot, xls, ppt, and more), pdfs, and a multitude of other formats.

Written and freely distributed by Phil Harvey, ExifTool runs on Windows, Mac OS X, and Linux.

When it was first released, the original focus of ExifTool was on image and audio files. For images, it focused on pulling out the camera type and details about the format of the image itself. It also pulls information about any tools that were used to edit the image or audio. If the image includes geotags indicating the latitude and longitude of where the photo was created, ExifTool retrieves that information.

ExifTool has been significantly extended beyond its original roots in image and audio metadata, now pulling data from the vast majority of file formats a penetration tester is likely to encounter. Of particular interest to pen testers is ExifTool's ability to discern user names, e-mail addresses, and document editing tools from the files it analyzes.

By default, ExifTool handles one or more files provided to it on its command-line invocation. Alternatively, the tool processes entire directories on the local machine where it runs, handling every file in the directory, and can even be set to recurse through a directory structure, analyzing all files it finds.

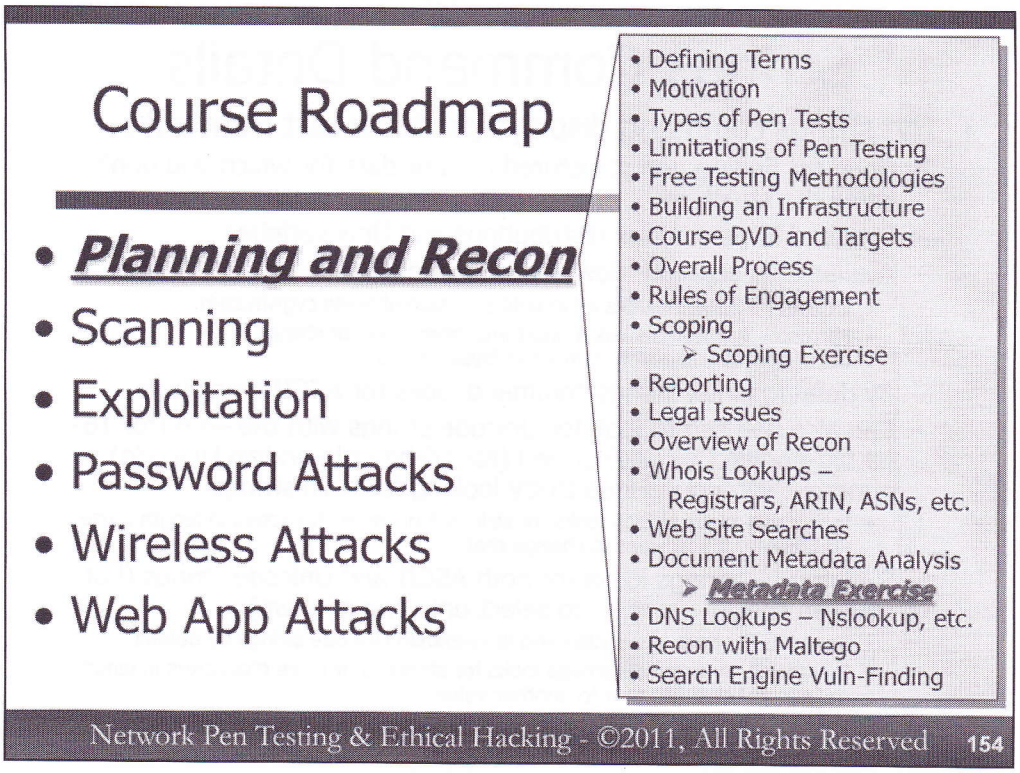



Unlike many metadata analysis tools which focus on structured data, the strings command is useful for finding unstructured data or data for which the structure is unknown. The strings command simply displays printable text from files. It is included in most Linux distributions and Unix varieties. The strings command is available as a separate download for Windows, in a variety of different packages. For example, it is available as a component of Cygwin, the free POSIX environment for Windows available at www.cygwin.com. Or, strings is available as a free stand-alone download from Microsoft Sysinternals.

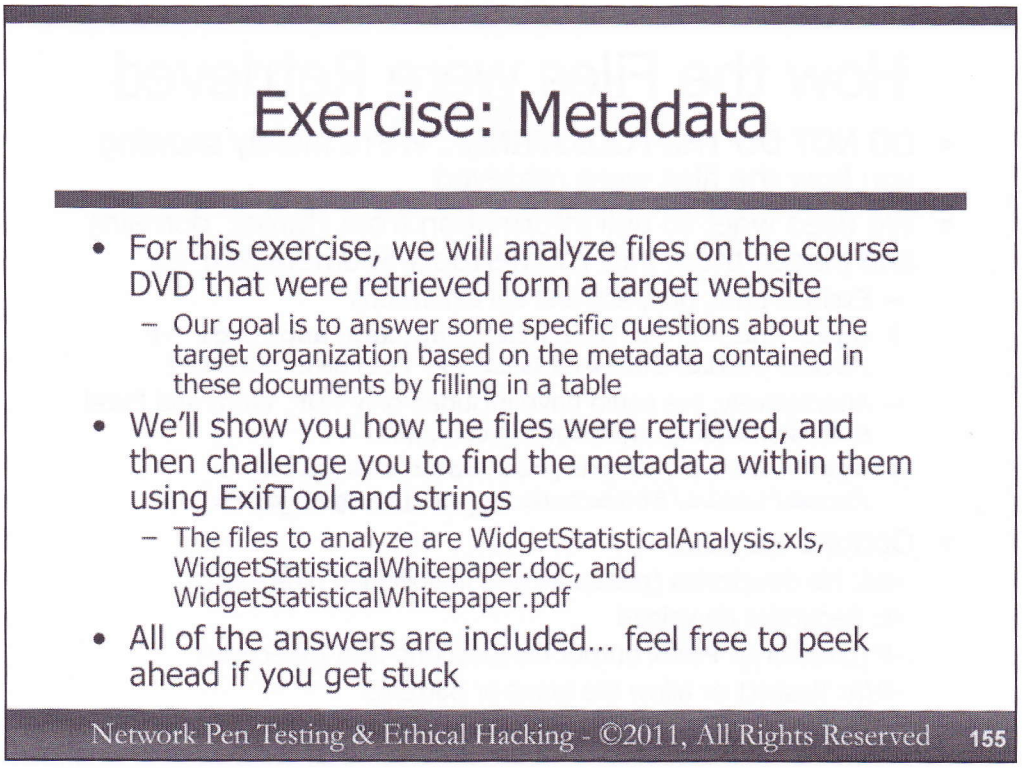
By default, the Linux version of strings looks for printable ASCII strings only. It searches through the file for four or more consecutive ASCII characters, and then prints them to Standard Output. To change the default minimum string length, strings can be invoked with the –n [N] option to specify whatever string length the user wants. The default of 4 characters is pretty reasonable for most uses.

Many document types, especially those associated with Microsoft Office programs (doc, docx, xls, xlsx, etc.) store some strings not as ASCII (an 8-bit character representation), but instead as Unicode (a 16-bit character representation). If you run Linux strings with its defaults, it will only show you ASCII strings, and you may miss out on some highly useful information. It's a good idea to run strings multiple times: once with its ASCII default, once with the –e b option (for "encoding" type of big endian 16-bit characters), and once with –e l (a lower-case "L" for 16-bit little endian encoding). Big endian and little endian refer to the way the bytes are ordered for the given string in the file. Most Microsoft document editing tools use little Endian encoding, but sometimes (even in the same file) will store some strings in big Endian format.

The Sysinternals version of strings looks for ASCII, big endian Unicode, and little endian Unicode strings by default (pulling each of those different formats in a single invocation), focusing on strings 3 or more characters in length. To focus only on ASCII or Unicode, the tool can be invoked with –a or –u respectively. And, to change the minimum character length, we can invoke it with –n [N].




Let's perform an exercise now, in which we'll use ExifTool and strings to harvest metadata from three sample files included on the course DVD. When conducting the exercise, think about how a penetration tester can use the information retrieved from the metadata during the remainder of the penetration test.




In this exercise, we'll analyze three files included on the course DVD, pulling metadata from them by running ExifTool and the strings command on the course VMware Linux image.

The goal will be to answer some questions about the target organization, filling in a table with information that would be useful in a penetration test.

We'll show you how the files were originally retrieved, although you will not have to retrieve the files for the exercise, given that they are already included on the course DVD.

The files to analyze are called:

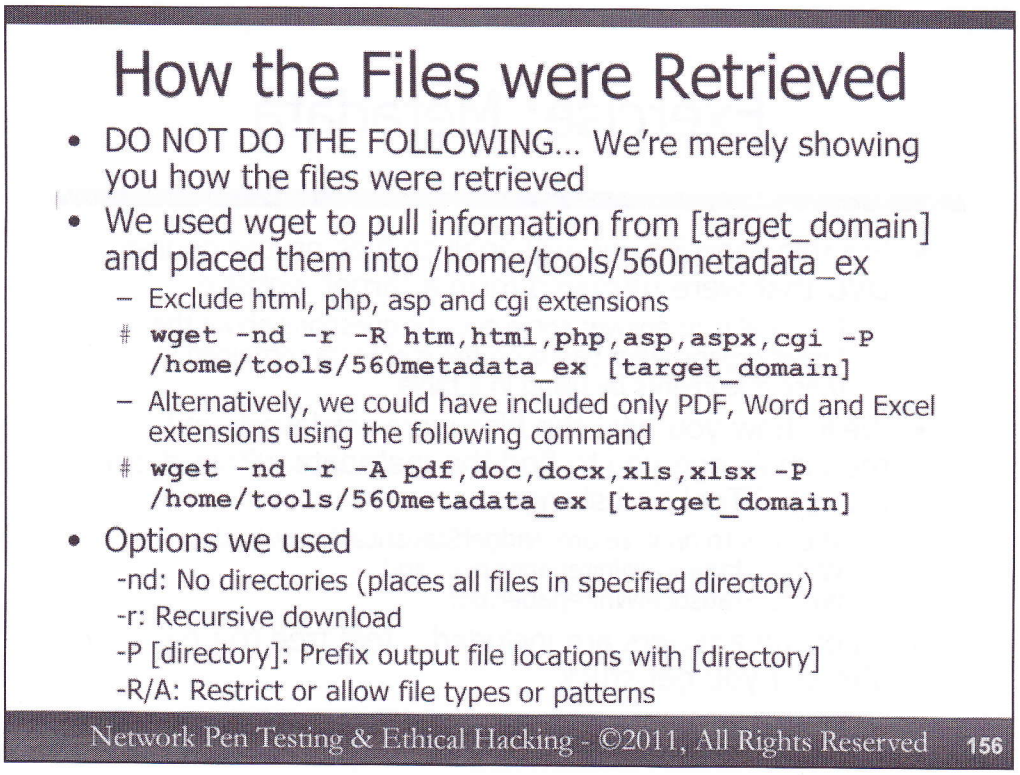
WidgetStatisticalAnalysis.xls

WidgetStatisticalWhitepaper.doc

WidgetStatisticalWhitepaper.pdf

The exercise is organized around a challenge-response format. We'll challenge you to populate a table with information, and then provide answers and a description of the techniques used to retrieve those answers.

Feel free to peek ahead at the answers and the techniques if you get stuck.




YOU DO NOT NEED TO PERFORM THIS STEP. WE ARE MERELY SHOWING YOU HOW THE FILES WERE RETRIEVED TO SET UP THIS EXERCISE.

The files for this exercise were retrieved from the target website using the wget command. This command pulls web pages from target web servers and supports a myriad of options. When used with the –r option, wget acts as a web spider, downloading pages from a target site, analyzing them for links, and pulling the pages they link to from the same target site.

When pulling files for document metadata analysis, we could use wget to simply get all files from the target site. However, sometimes we want to focus on only specific types of files. We can tell wget either to omit files of a certain type, getting everything else, with the –R option followed by a list of file types that we don't want to get. Or, we could specify a set of file types that we do want to retrieve, with the –A option to allow only certain types.

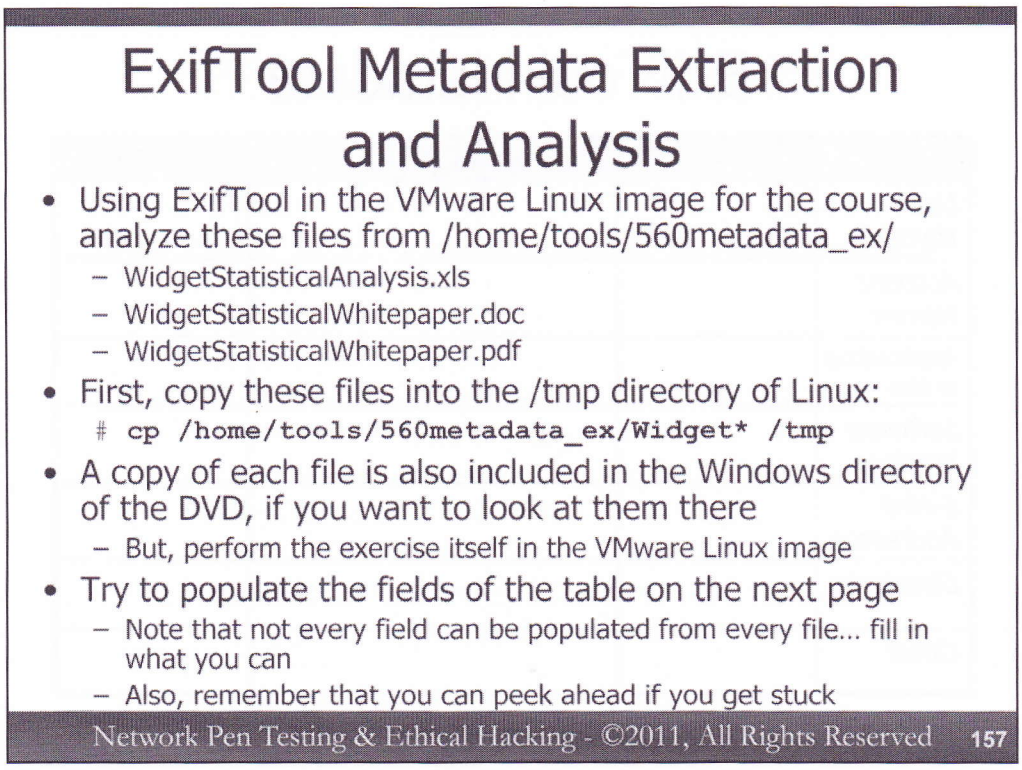
When pulling these files, we wanted to put them all in a single directory, without mimicking the directory structure of the target website, so we used the –nd option to make wget omit directories from the pulled files. Also, we wanted the files to be placed into the /home/tools/560metadata_ex directory, so we used the –P option to add a directory prefix.

The resulting wget command that we used to pull these files for this exercise was:

```
# wget -nd -r -R htm,html,php,asp,aspx,cgi -P /home/tools/560metadata_ex
[target_domain]
```

DO NOT RUN THAT COMMAND HERE… It is just an example of how we could retrieve the files.

Alternatively, we could have specified we only wanted pdf, doc, docx, xls, and xlsx files using the syntax on the slide above. In document metadata analysis for real-world pen tests, it is typically better to create a blacklist of file types you do NOT want (the first wget command) rather than creating a white list of types you do want (the second wget command on the slide). With the first command, we may get a file type that we did not expect, but which includes particularly juicy information that the second command would omit.




To start the exercise, let's first create a copy of the files in the /tmp directory, so we can perform our analysis:

```
# cp /home/tools/560metadata_ex/Widget* /tmp
```

That should copy over the three files you want to analyze:

WidgetStatisticalAnalysis.xls

WidgetStatisticalWhitepaper.doc

WidgetStatisticalWhitepaper.pdf

The goal of this exercise will be to run ExifTool first on each of these files, trying to populate the table on the next slide. Later, in the second half of the exercise, we'll run the strings command.

A copy of each of these files is also included on the course DVD in the Windows directory. You can open them on Windows and look at them if you'd like, but the exercise itself should be performed in Linux, which has ExifTool and strings installed.

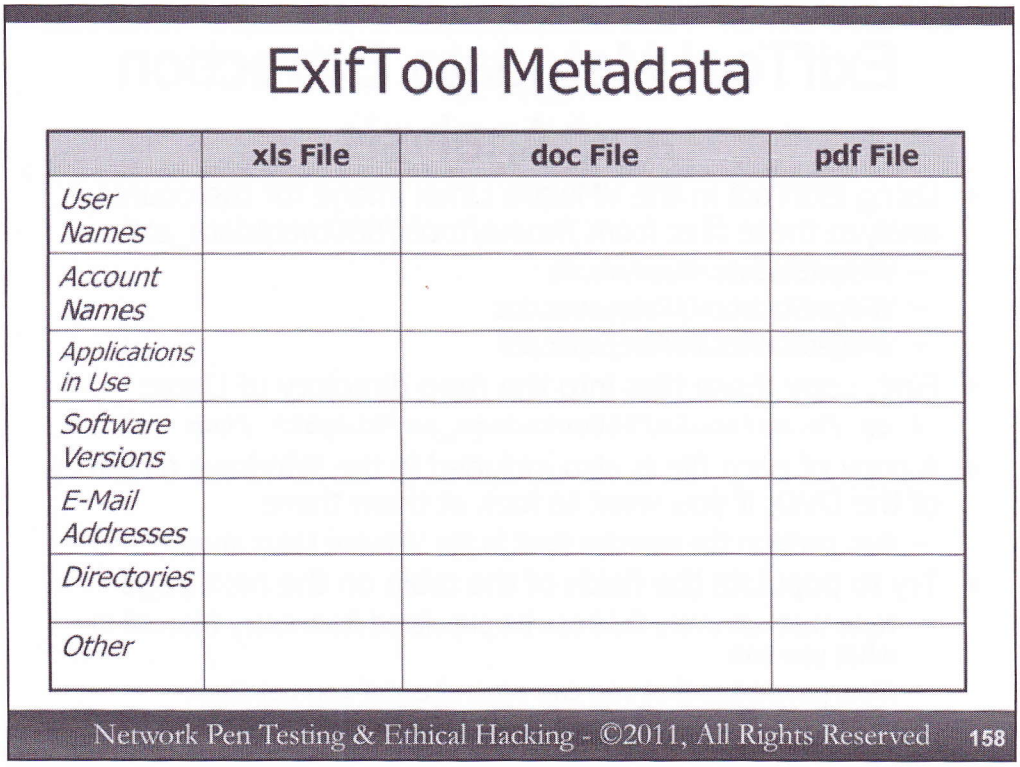
ExifTool can be invoked on the VMware Linux image to analyze a file by running:

```
# exiftool [filename]
```

Try this for each of the files, and enter the data you discover in the table on the next slide.

REMEMBER THAT YOU WON'T BE ABLE TO POPULATE EVERY FIELD IN THE TABLE FOR EVERY FILE.

ALSO REMEMBER THAT YOU CAN FEEL FREE TO PEEK AHEAD AT THE ANSWERS AND THE APPROACH USED TO DETERMINE THEM.

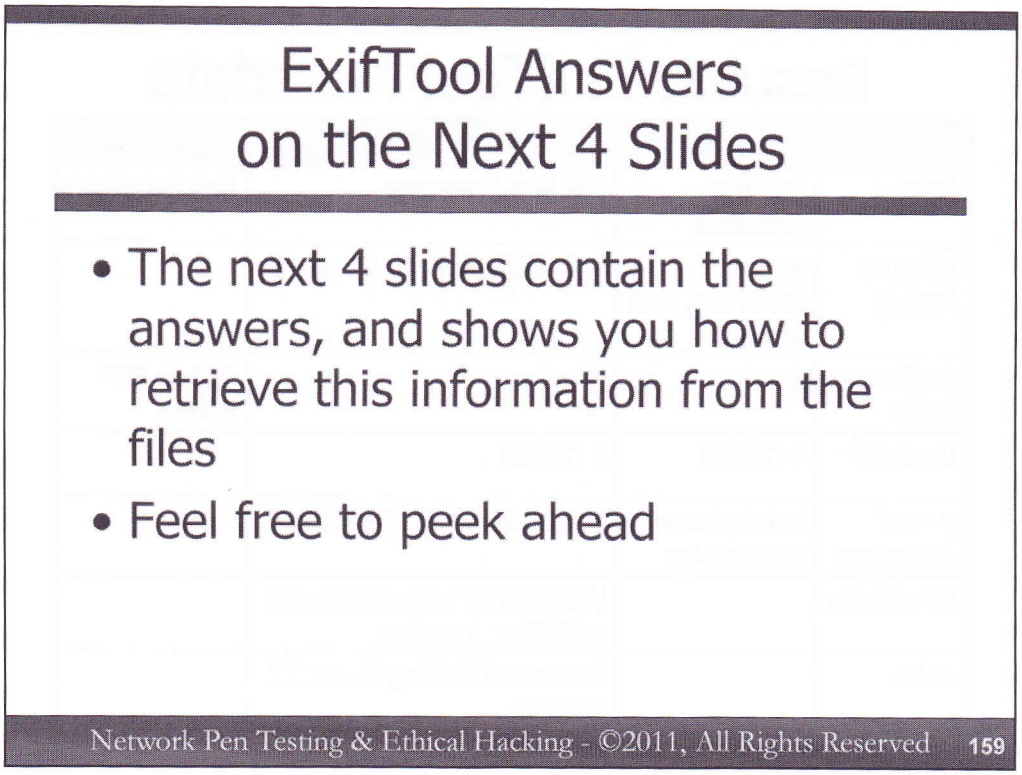



Try to populate the fields above using exiftool on each of the three files:

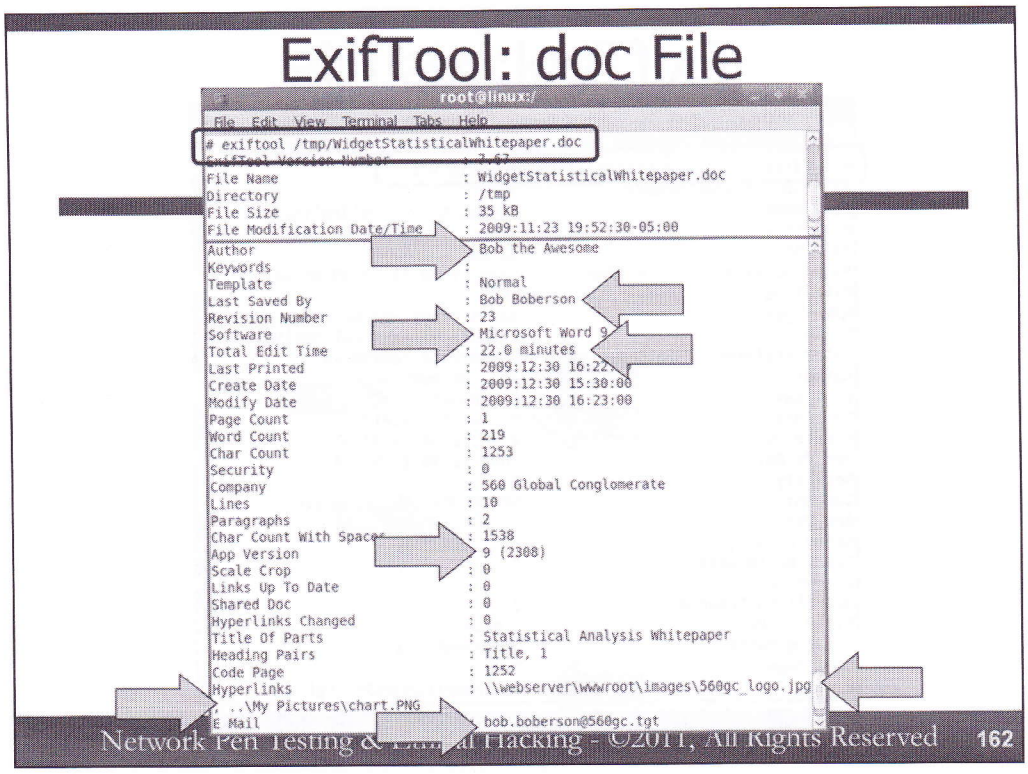
```
# exiftool /tmp/WidgetStatisticalAnalysis.xls
# exiftool /tmp/WidgetStatisticalWhitepaper.doc
# exiftool /tmp/WidgetStatisticalWhitepaper.pdf
```

REMEMBER THAT YOU WON'T BE ABLE TO POPULATE EVERY FIELD IN THE TABLE FOR EVERY FILE.

ALSO REMEMBER THAT YOU CAN FEEL FREE TO PEEK AHEAD AT THE ANSWERS AND THE APPROACH USED TO DETERMINE THEM.



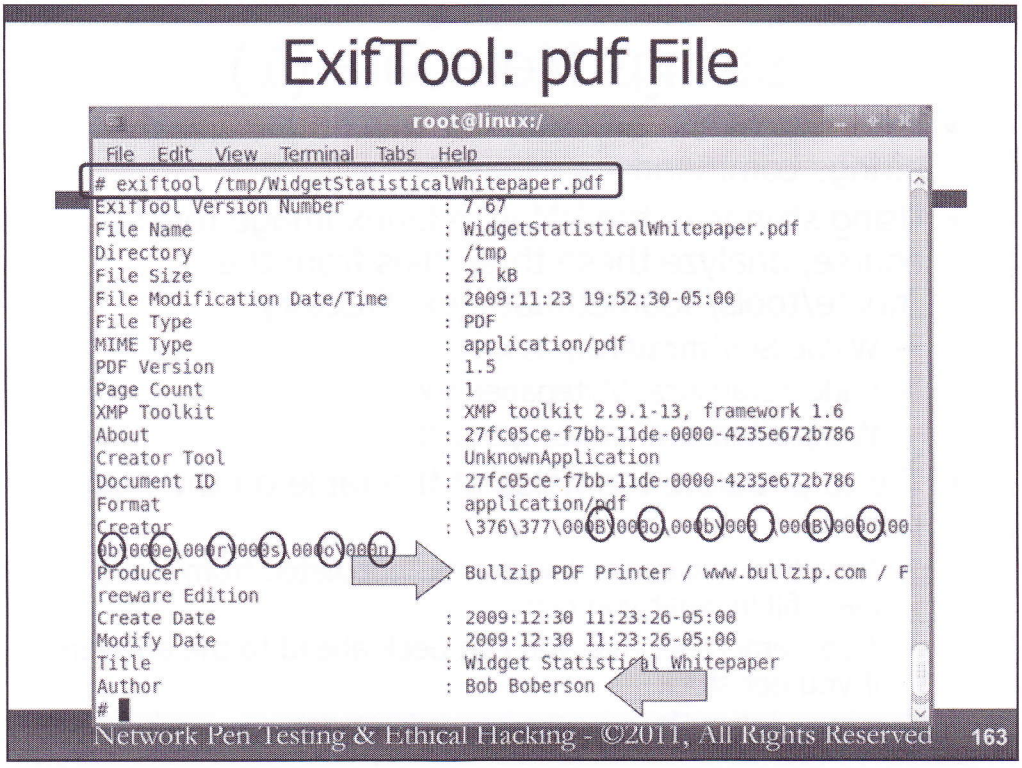

The answers to the Exiftool portion of the exercise appear on the next slide. Feel free to peek ahead if you haven't finished the previous part of the exercise and need a hint or inspiration. Or, even if you have finished with the ExifTool component of the exercise , you can look ahead to check your work and then move on to the strings command section of the exercise.




We can run ExifTool against the doc file as follows:

```
# exiftool WidgetStatisticalWhitepaper.doc
```

Again, we retrieve very similar information to what we had before, but now we even get a little more data: a hyperlink to an image included in the original doc file. That image was pulled from a share on a server called "webserver". We can see the full path to that image on the server.

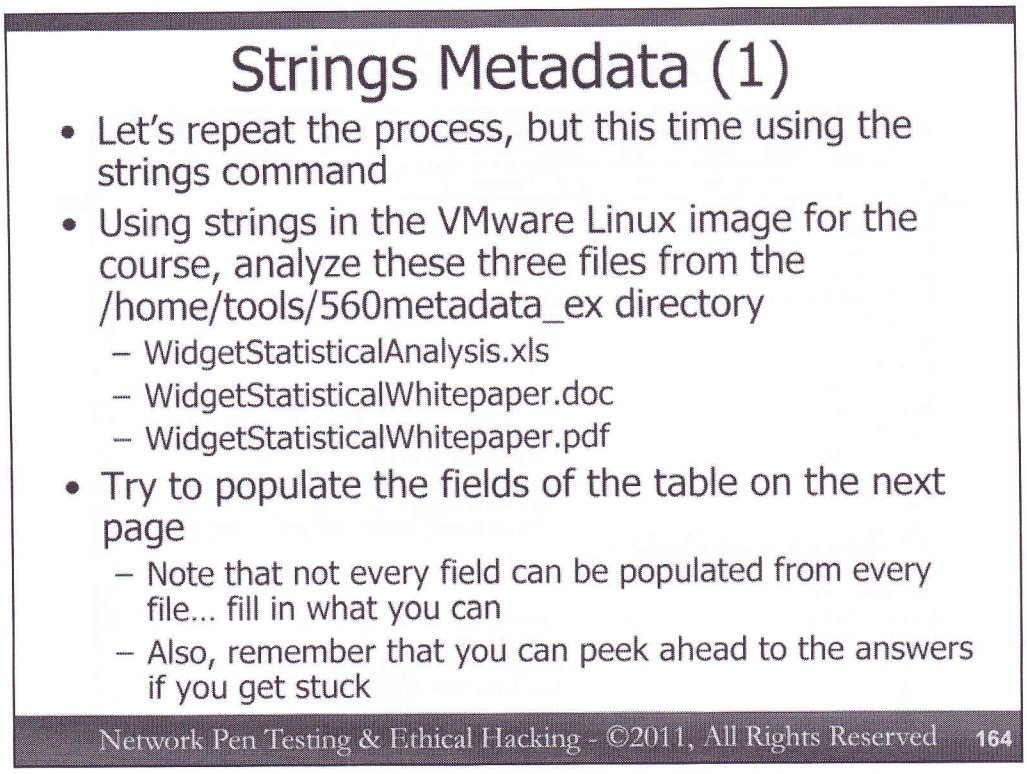



And, finally, here is the information we can get from the pdf file:

```
# exiftool WidgetStatisticalWhitepaper.pdf
```

Note that the pdf file has less metadata than we saw elsewhere. Also note that the Creator name is somewhat obscured, with each letter distributed throughout some other characters, but still revealing Bob Boberson as the document's creator. We see this same information also in the Author field.

The pdf metadata also shows us that the Bullzip PDF Printer was used to generate the file. In a penetration test, we could research whether that particular program has a known security flaw. Unfortunately, it has no known flaws as of this writing.



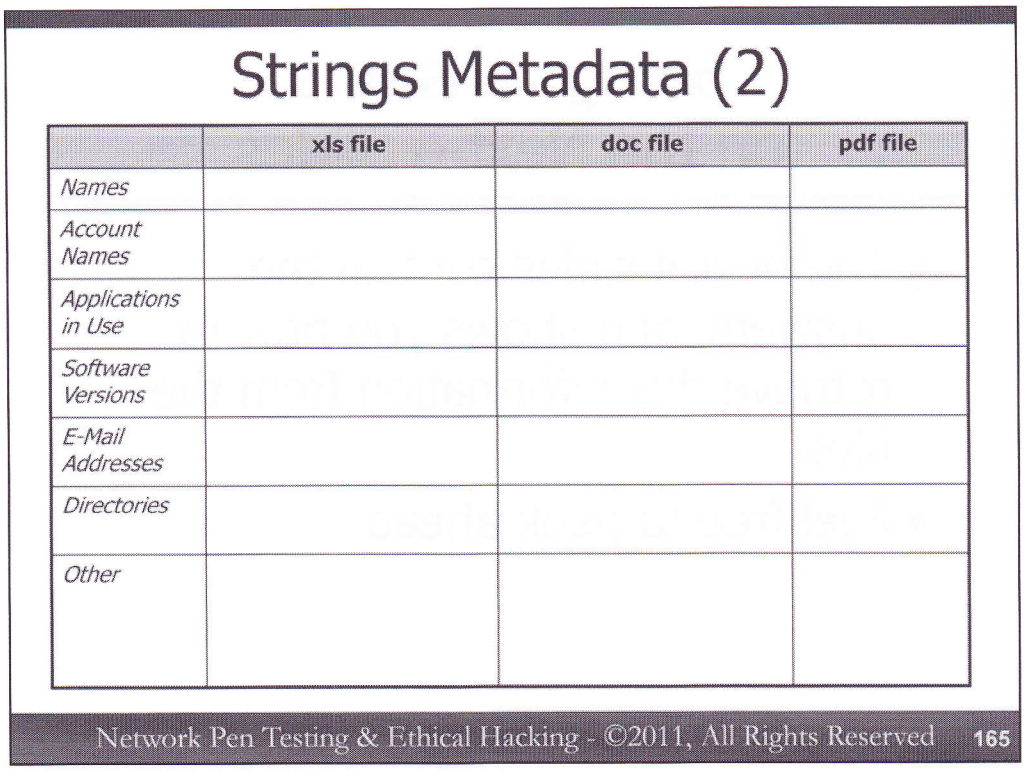

Next, let's use the strings command to analyze the metadata in the same files. Note that we will get unstructured output here, making it harder to find the information assets we are looking for. But, as a bonus, we may find additional information that ExifTool wasn't able to discern.

We'll use the same three files for this analysis, populating the fields on the table that follows.

You can run strings against each file using the simple command:

```
# strings [filename]
```

Again, remember that not every field can be populated from every file, and also remember that you can peek ahead at the answers.

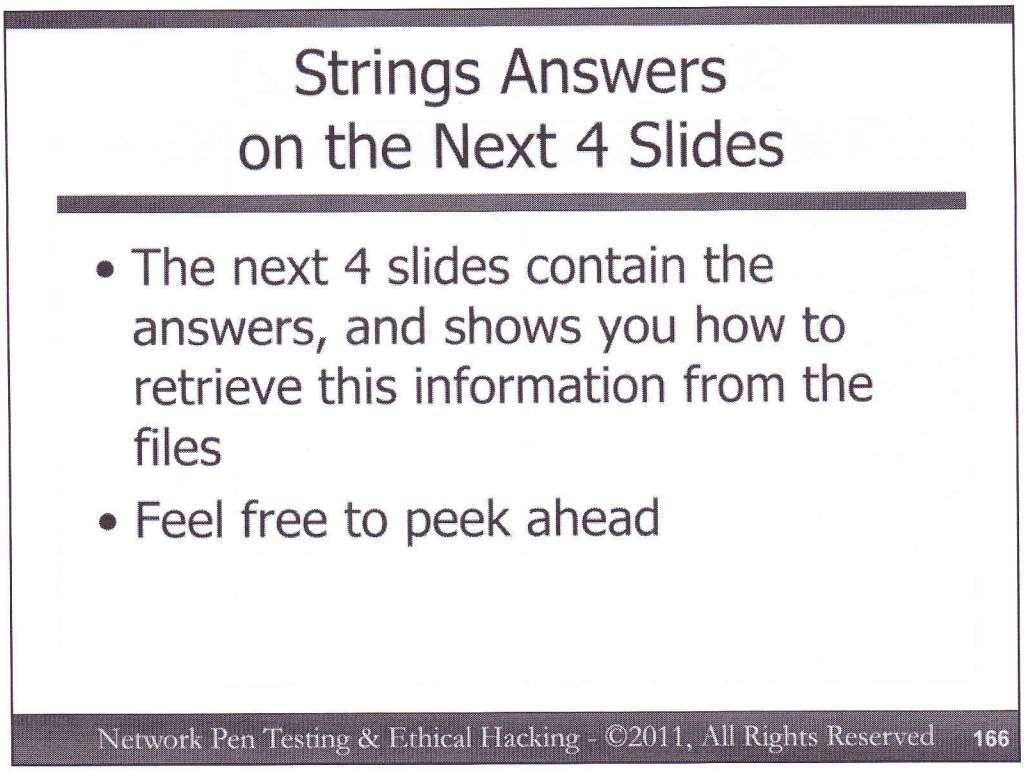



Run the strings command against each file as follows, recording your output after each command runs:

```
# strings /tmp/WidgetStatisticalAnalysis.xls
# strings /tmp/WidgetStatisticalWhitepaper.doc
# strings /tmp/WidgetStatisticalWhitepaper.pdf
```

Note that this invocation shows only ASCII strings. You may want to try –e l (that's a lower-case L) for each file as well, to look for little-endian Unicode strings. Also, you may also want to include –e b for each file to get big-endian Unicode strings.

Pay particular attention to the "Other" row, as you will likely find a lot more information there when running strings than you did with ExifTool.




The answers for the strings portion of the exercise are on the next page. Feel free to look ahead and review them as a hint, even before you finish the exercise.

# Results: Strings Metadata

| | xls file | doc file | pdf file |
|---|---|---|---|
| *Names* | Bob the Awesome | Bob the Awesome | Bob Boberson |
| *Account Names* | Bob, Boberson, and variants | Bob, Boberson, and variants | |
| *Applications in Use* | Excel, Office | Word | Bullzip PDF Printer |
| *Software Versions* | 9.0, 9.0 | 9.0 | |
| *E-Mail Addresses* | bob.boberson@560gc.tgt | bob.boberson@560gc.tgt | |
| *Directories* | C:\Users\Bob Boberson\My Pictures\560gc_logo.jpg<br>C:\Program Files\Microsoft Office\Office\ | \\webserver\wwwroot\images\560gc_logo.jpg<br>C:\Users\Bob Boberson\ | |
| *Other* | Customer PII:<br>- Names<br>- Social Security #'s | "Note to Self: Sandra asked to open port 8000 on the Windows Web Server Firewall for something called IceCast. Do this before lunch." | |

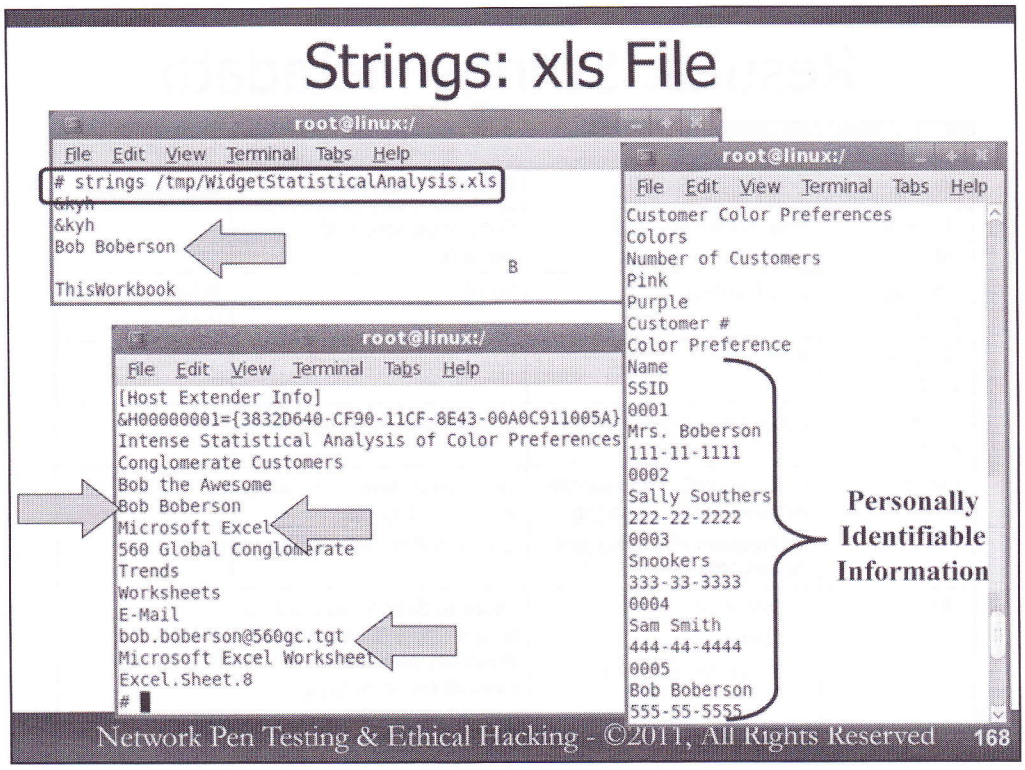
167

Note that we were able to retrieve very similar information to what we saw with the ExifTool program, but we found some additional useful items here.

Most notably, we found some directories, including a directory structure on Bob Boberson's computer system as well as the location of the Office installation on a machine. Furthermore, we were able to retrieve some very interesting information about customer Personally Identifiable Information (PII) from an obscured field in the spreadsheet file. This field isn't shown on the display, because it is obscured on the display by an image. But, it holds customer names and Social Security Numbers.

Also, in the doc file, we were able to find a comment that Bob left for himself regarding the opening of a port on the firewall for a service associated with a program called IceCast. We could research that and determine that there are known buffer overflow vulnerabilities in the IceCast service.

The following slides contain screenshots showing how each of this information was found using the strings command.

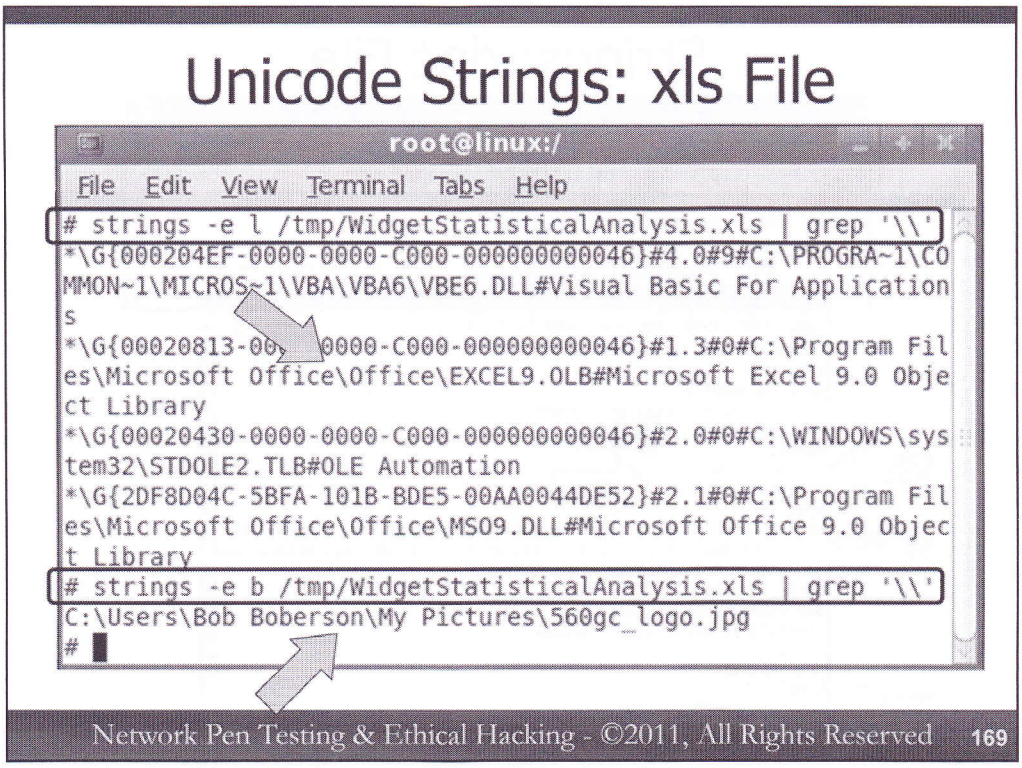



We start by running strings against the xls file:

```
# strings WidgetStatisticalAnalysis.xls
```

Here, we can easily see the "Bob Boberson" string appear on the screen. If we look further, we can also see that Microsoft Excel was in use, and it was likely registered with 560 Global Conglomerate corporation. We can see an e-mail address, and an indication that Microsoft Excel was in use.

Of most interest to us, though, is the data that appears later in the file, which is a list of Personally Identifiable Information (PII) for customers of this organization. They have inadvertently leaked this information in the documents on their website.




We should also run strings against the xls file to find Unicode information, starting with little-endian encoded information. Let's focus our results on directory structures by sending the output through the grep command. We'll be using grep to find any strings that contain a backslash (\), the common indication of a directory structure on Windows machines.

```
# strings -e l WidgetStatisticalAnalysis.xls | grep '\\'
```

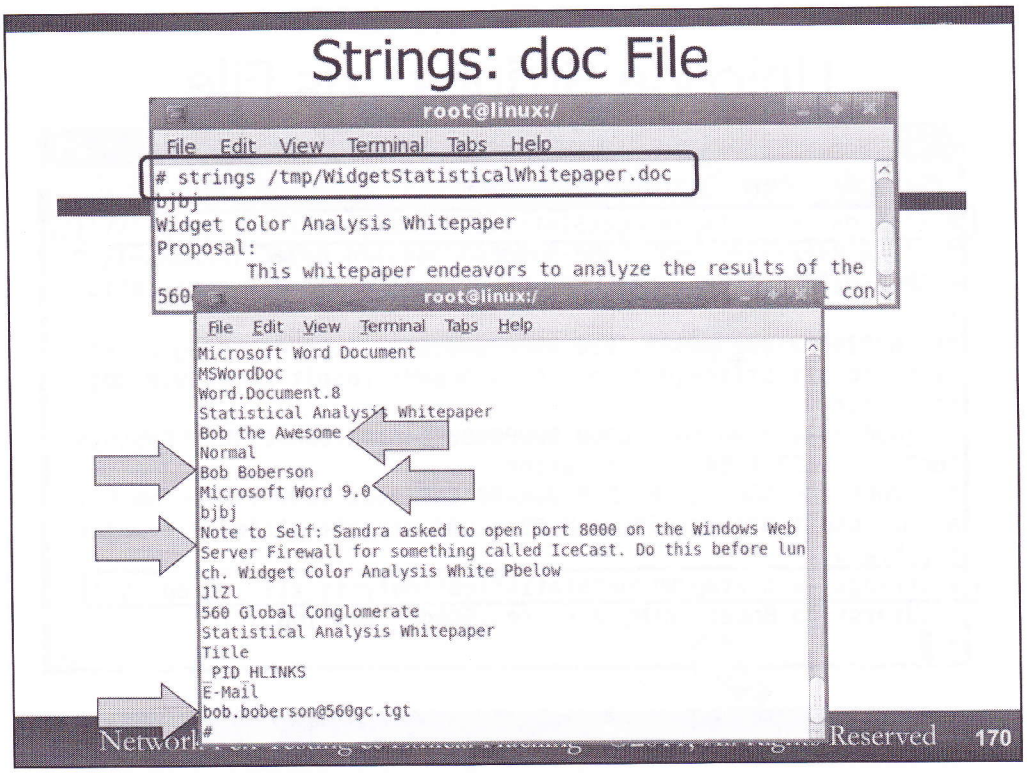
NOTE THAT THAT COMMAND CONTAINS A –e FOLLOWED BY A LOWER-CASE L. It may look like a one, but it is indeed a lower-case L. Also, the single quotes around the \\ tell the shell to send the \\ to grep without interpreting it in any way. The first \ is an escape sequence to tell grep that the second backslash should be interpreted as a backslash. In other words, '\\' is one way we can get a single backslash into grep.

Here, we can see that Microsoft Office was in use, specifically version 9.0.

We can look for big-endian Unicode strings using the following command (again, focusing on directories in the C: partition):

```
# strings -e b WidgetStatisticalAnalysis.xls | grep '\\'
```

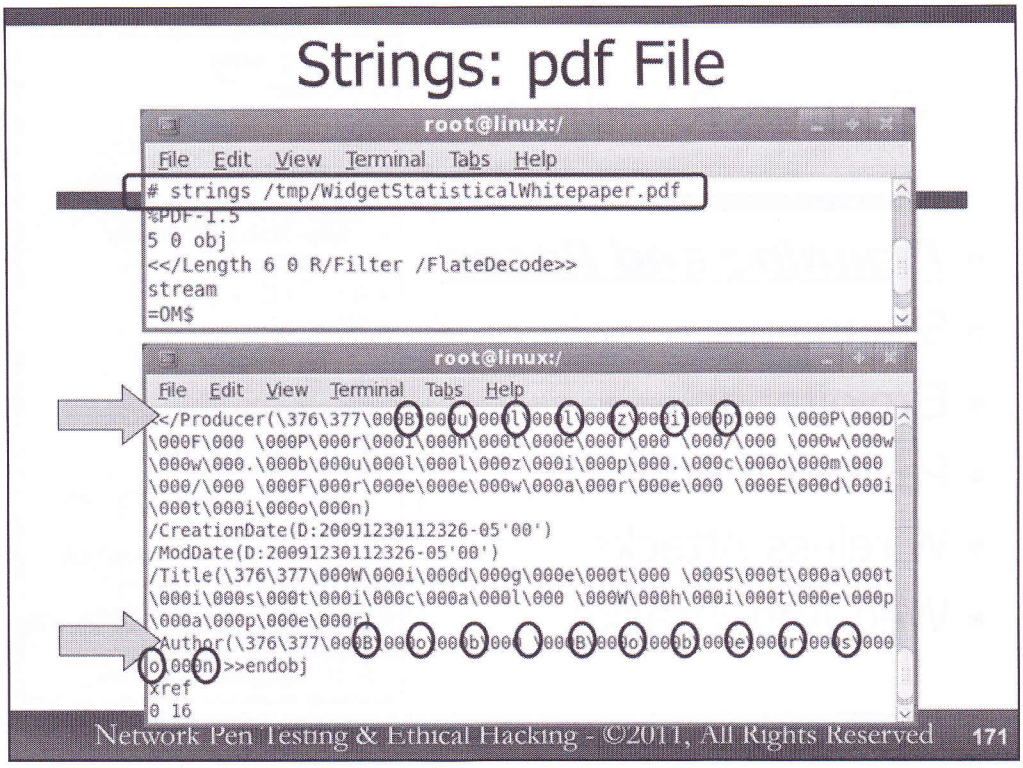
Here, we can see the directory structure on Bob's computer.




Next, let's turn our attention to the doc file, running:

```
# strings WidgetStatisticalWhitepaper.doc
```

Here, we can see the contents of the file itself near the top. But, look further, and you can see the document's author, an indication of Word 9.0, and that comment that Bob inserted into the document about the IceCast service and port 8000. We also see Bob's e-mail address.



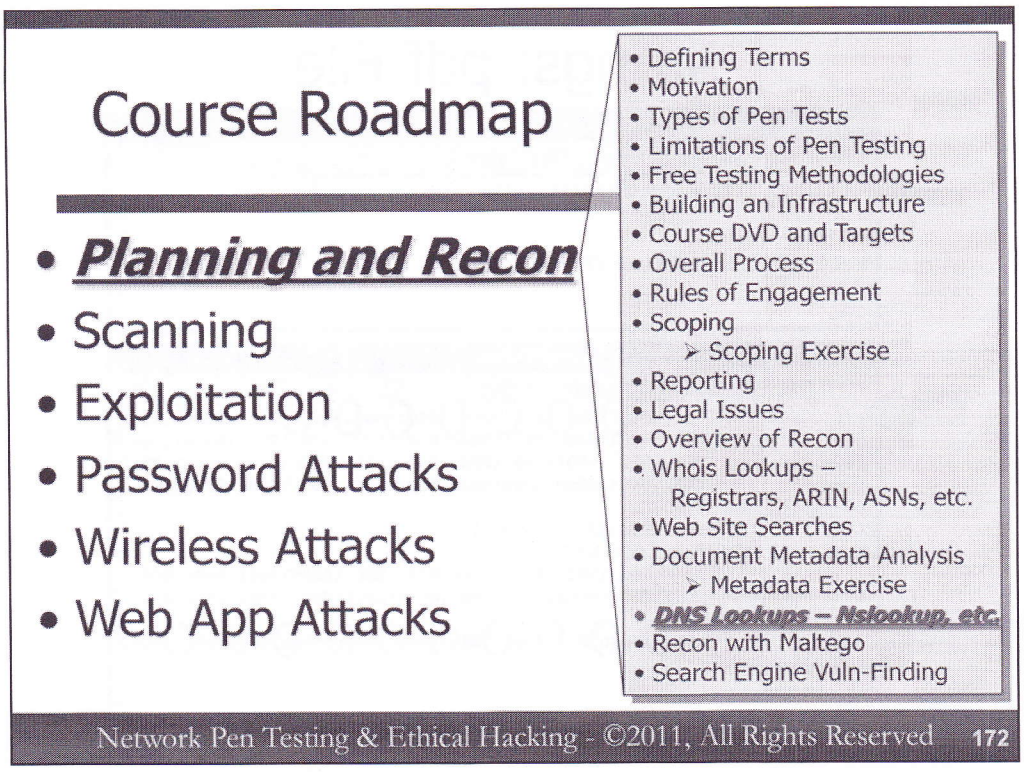

And, finally, we run strings against the pdf file:

```
# strings WidgetStatisticalWhitepaper.pdf
```

Here, we must look very carefully to see the program used to create the pdf file, listed near the "Producer" string, separated by various other characters. Piecing together the results, we can see B-u-l-l-z-i-p.

Likewise, the author information is separated by different characters, but we can piece together B-o-b- -B-o-b-e-r-s-o-n.

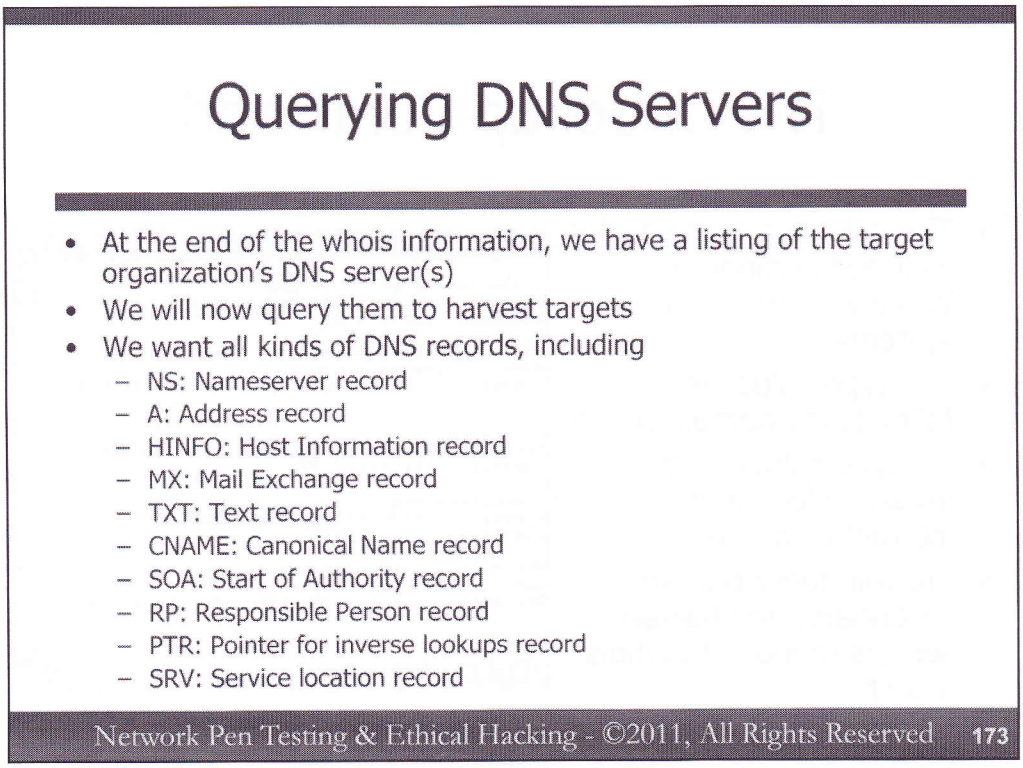
In this exercise, we've seen how we can use ExifTool and the strings command to pull data. We've seen the advantages of structured data and ExifTool in pinpointing useful information. We've also seen the advantages of looking for unstructured data with the strings command to find something that ExifTool isn't designed to show: obscured fields and comments. We've also seen how to transcend the default limitation of ASCII strings on Linux with the –e option to look for Unicode strings, both big and little endian.




Now that we've retrieved a list of DNS servers associated with the target from our Whois lookups, we want to query those servers to gain an inventory of potential target machines associated with the given target domain(s).

The techniques we'll cover next will likely identify numerous systems that are directly and indirectly associated with the target. Be careful before moving beyond the recon phase with respect to each of these servers. Don't just start scanning them! It is highly worthwhile to conduct a status call with target organization personnel to make sure that all machines identified throughout the recon phase, and especially those discovered during the DNS interrogation phase, are within the testing scope.

You may find that some systems identified in this phase are outside of the scope, but are unknown to personnel in the target organization before the test begins. That's why it is crucial to verify their inclusion in or exclusion from the test's scope.



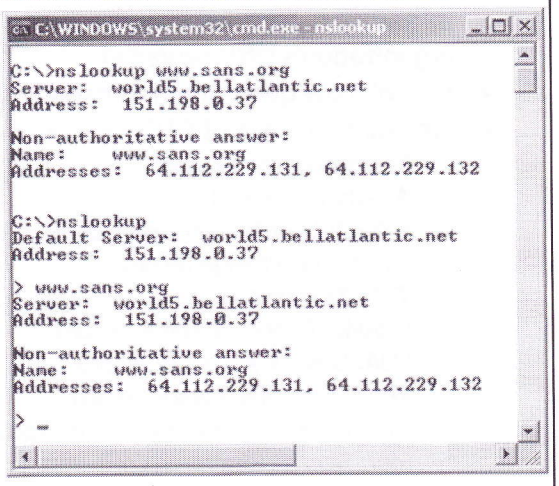

The last elements of the whois record include the Domain Name System (DNS) Servers associated with the target organization, listed in the order of primary, secondary, and tertiary (if it exists) DNS servers. We will next try to harvest records from those name servers.

Name servers are focused on resolving domain names into IP addresses, but that isn't their sole function. They also indicate which machines are mail servers for a given domain, among other useful information. DNS servers house a variety of different records, including:

- NS: Nameserver record, which indicates the name servers associated with a given domain.
- A: Address record, which maps a domain name into an IP address.
- HINFO: Host Information record, which associates an arbitrary set of information with a domain name, formerly used to indicate system types.
- MX: Mail Exchange record, which identifies the mail servers for the given domain.
- TXT: Text record, which includes an arbitrary text string for the domain.
- CNAME: Canonical Name record, which indicates aliases and alternative names for a given host.
- SOA: Start of Authority record, which indicates that a server is authoritative for that DNS zone (set of records).
- RP: Responsible Person records, which are informational, not functional (i.e., they have no impact on DNS functionality), and indicate the human responsible for a given domain (seldom used).
- PTR: Pointer for inverse lookups records, also called a reverse record, indicating an IP address to domain name mapping.
- SRV: Service location records, which provides information about available services, including port and host name (seldom used).

# The nslookup Command

- The nslookup command is included in modern Windows, Linux, and Unix systems
- Can type nslookup followed by domain name
- Or, type nslookup to invoke nslookup in interactive mode
- We will query primary, secondary, and tertiary servers found with whois search





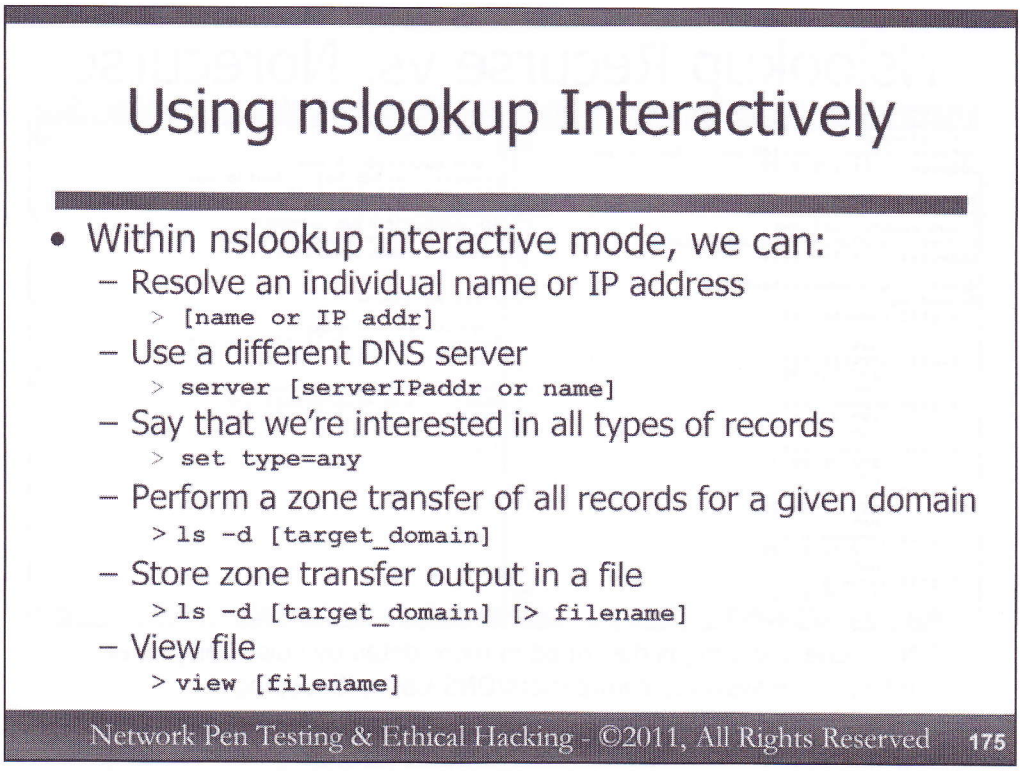

To query DNS servers, we can use the nslookup command, built into modern Windows systems. It is also included in most Linux and Unix variants.

Nslookup can be used in two ways. First, we could simply type nslookup followed by the domain name that we want to query. The nslookup command will use the local operating system settings to determine a name server, to which it will submit the request, displaying the results.

Alternatively, we could use nslookup in interactive mode, by running "nslookup" by itself and hitting enter. Then, we are given an nslookup prompt ">", into which we can type names for resolution or directives to control nslookup's configuration. We can redirect nslookup from within this prompt to use other DNS servers.

In the screenshot on this slide, we show both types of invocation for nslookup, used in both ways to resolve www.sans.org.

Next, we'll try to harvest records from DNS, querying the primary, secondary, and tertiary DNS servers that we learned from our whois lookup.




In nslookup's interactive mode, we can simply resolve a name by typing it at the > prompt.

To tell nslookup to use a different DNS server, we could use this syntax:

```
> server [serverIPaddr or name]
```

By default, nslookup tries to pull Address records. We are often interested in other record types, such as MX. We can use the "set type=" directive to look for other kinds of records. If we want all kinds of records for a given domain, we can type:

```
> set type=any
```

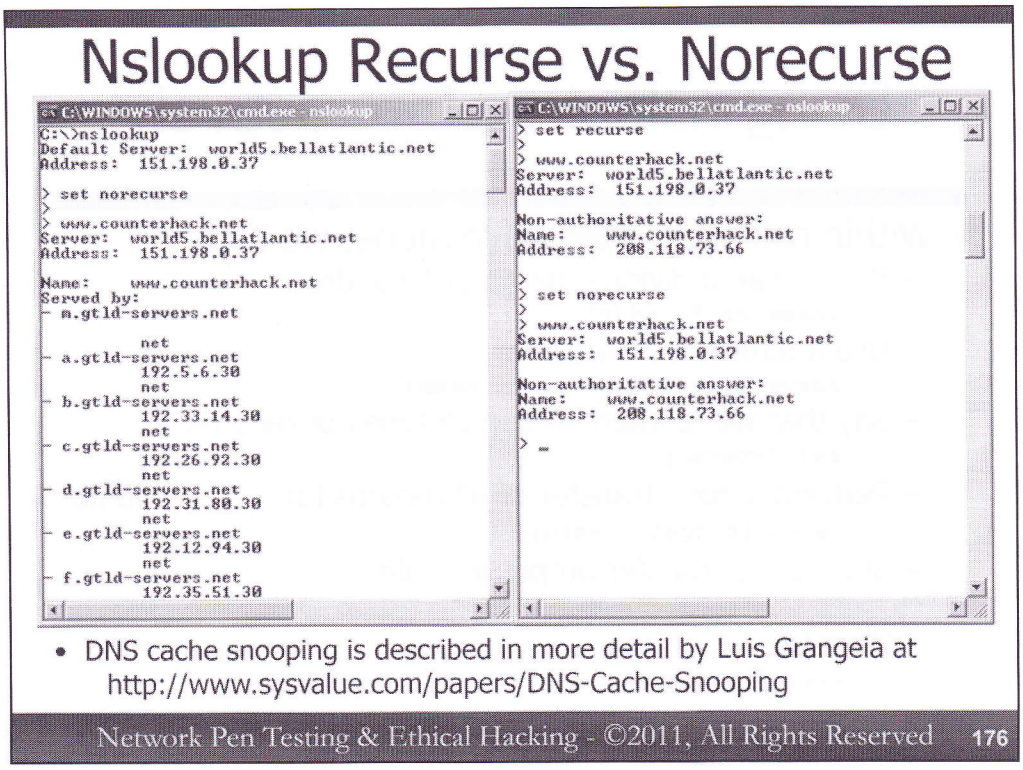
A zone transfer asks the DNS server to transmit all records it has for a given domain. If the DNS server supports zone transfers from the source IP address where we are running nslookup, a complete list of records will be displayed on the screen.

```
>ls -d [target_domain]
```

To place the zone transfer results in a file, we could simply redirect its output into a file (> filename). We can display this file from within nslookup with the view command, as follows:

```
>view [filename]
```

Zone transfers were designed so that secondary DNS servers could update their records from primary name servers. DNS zone transfers are carried over TCP port 53, whereas most DNS queries and responses rely on UDP port 53. Many DNS servers block zone transfers from arbitrary locations on the Internet, either by being configured to allow them only for certain addresses (the primary name server) or by a firewall blocking TCP to port 53.

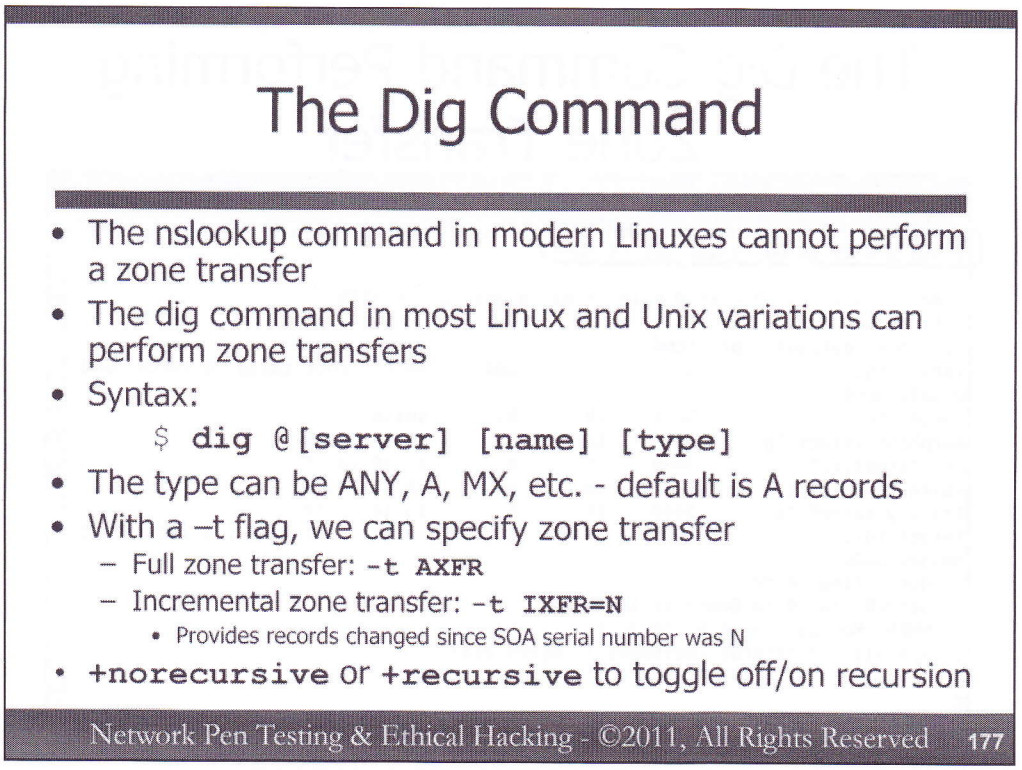



If a DNS server does not have the information we request, it can forward that request to other DNS servers to retrieve the information in a process known as a recursive lookup.

By default, nslookup will ask for recursion from the name servers it queries (the RD – recursion desired bit – in the DNS query is set to 1). The nslookup command can be configured to create queries that do not request recursion using the "set norecurse" syntax, which sets the RD bit to zero.

With the norecurse option set, we can determine the records that a DNS server has loaded in its cache. In the screenshot here, we have set the norecurse option and tried to resolve www.counterhack.net. We are given back a set of root name servers, implying that the name server didn't have the information we requested. Then, we use the "set recurse" option. We again tried to resolve www.counterhack.net, getting our results because the target name server used recursion to find our answer. We then ran "set norecurse". When we tried to resolve the name again, we got our answer back because it was now loaded into the DNS server's cache.

This technique of investigating what a given target DNS server has cached is known as DNS cache snooping. The technique is described quite well in more detail by Luis Grangeia in his research paper at http://www.sysvalue.com/papers/DNS-Cache-Snooping. In this paper, Grangeia uses the dig command to query DNS servers with and without the RD bit set.




In many recent Linux and Unix systems, the nslookup command has been altered so that it can no longer perform zone transfers. On these systems, we can use the dig command for various kinds of DNS research, including zone transfers.

The dig command has the following syntax:

```
$ dig @[server] [name] [type]
```

The types we can specify include the abbreviations listed earlier, including A, MX, SOA, etc. To receive all kinds of records, we use the ANY type. If no type is specified, dig defaults to A (address) records.

To get dig to perform a zone transfer, we invoke it with the –t AXFR notation, as:

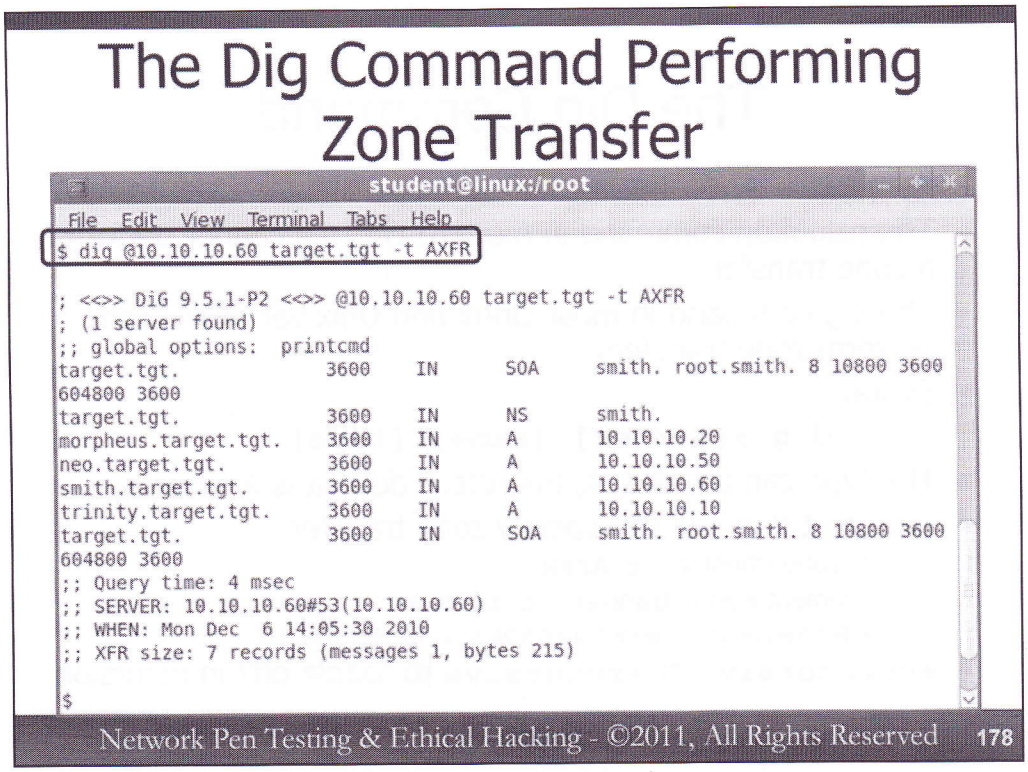
```
$ dig @[server] [domain] -t AXFR
```

This syntax will pull all information about a given domain. Alternatively, dig can perform an incremental zone transfer, pulling only recently updated records, using this syntax:

```
$ dig @[server] [domain] -t IXFR=[N]
```

N is an integer that refers to the serial number of a Start of Authority record. The incremental zone transfer request will pull all records that have changed since the SOA serial number was the N we specified in our dig request.

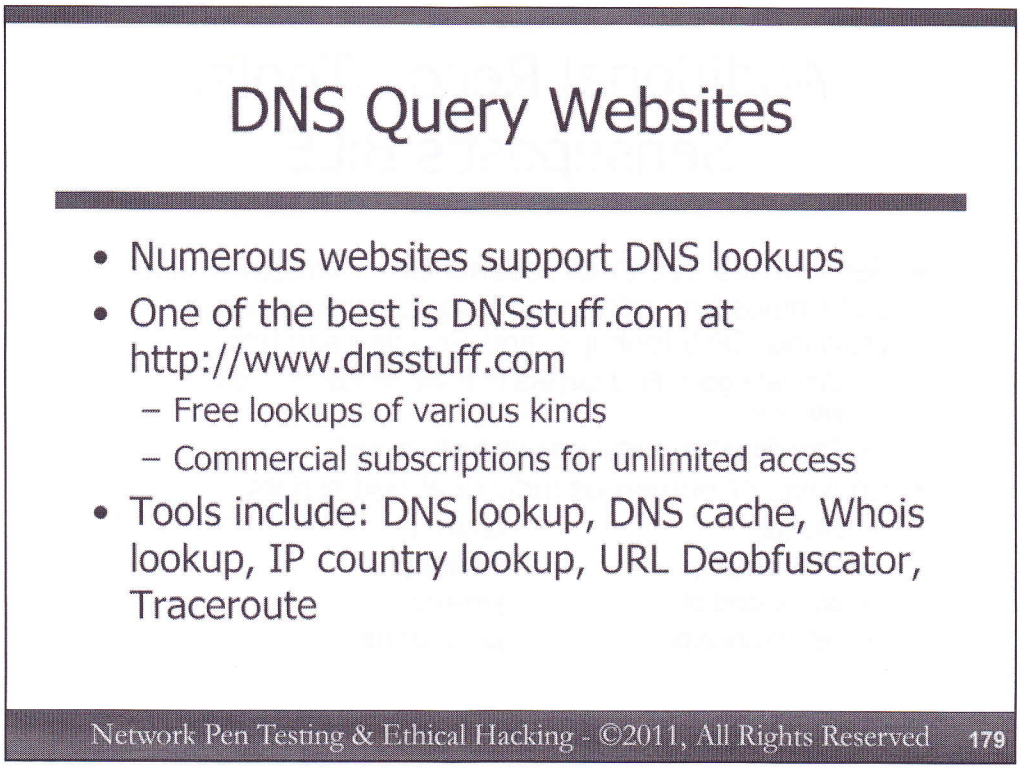
Dig also supports turning on or off the Recursion Desired (RD), with the **+norecursive** or **+recursive** syntax. By default, dig performs recursive searches.




In this screenshot, we've directed the dig command to contact the name server on 10.10.10.45, asking it for information about the target domain called target.tgt. We've requested a zone transfer (-t AXFR). Putting this all together, we get the following command.

```
$ dig @10.10.10.45 target.tgt -t AXFR
```

The output shows a good deal of information about potential target systems in the target.tgt domain.

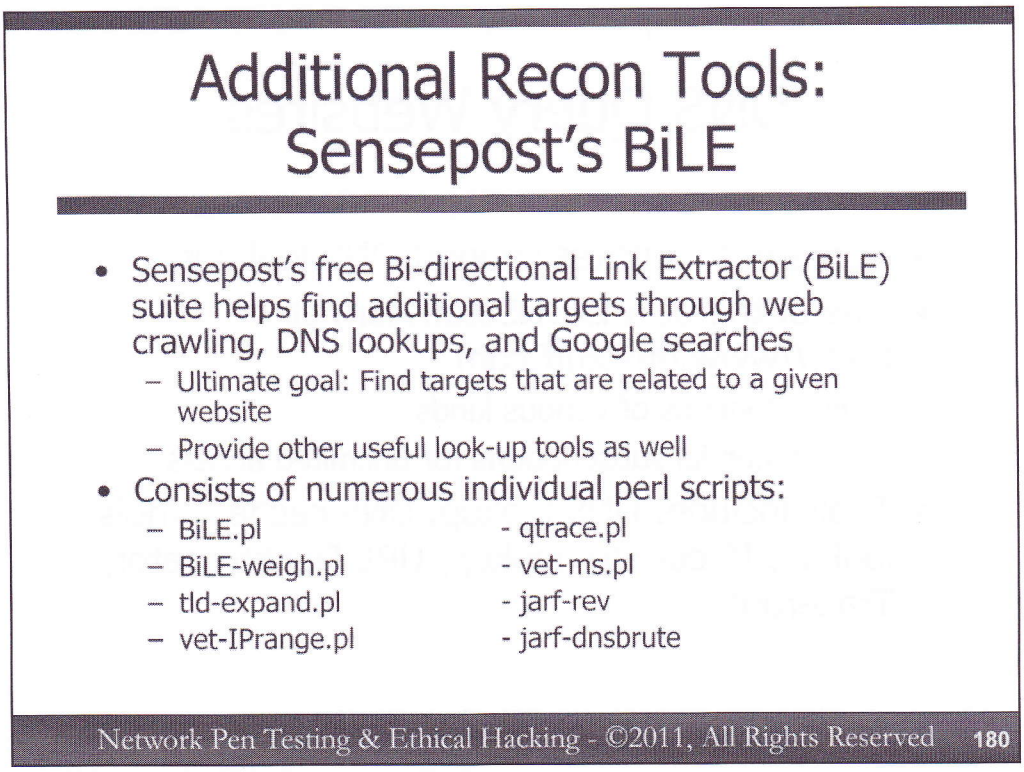



Instead of relying on built-in tools like nslookup and dig, or third-party download DNS query tools, we could use various websites devoted to resolving domain names.

One of the best websites for performing DNS lookups is DNSstuff.com. It offers a variety of free services, all implemented by a series of simple web forms into which you can type information to get responses. For an additional monthly or annual fee, you can get better performance, unlimited searches, and some additional services.

The free services on DNSstuff.com include:

- DNS Lookup: Maps domain names into IP addresses.
- DNS Cache: Checks various large ISP name servers to see if they have a given name cached.
- Whois Lookup: Implements a standard whois search.
- IP Country Lookup: Maps an IP address to the country it is (most likely) located in.
- URL Deobfuscator: Tries to convert a deliberately obscured URL that might be encoded in Unicode, Hex, or other formats, into a more easily readable form. Phishing attacks often rely on obscured URLs.
- Traceroute: Shows all of the router hops from the DNSstuff traceroute servers to the given destination you provide.

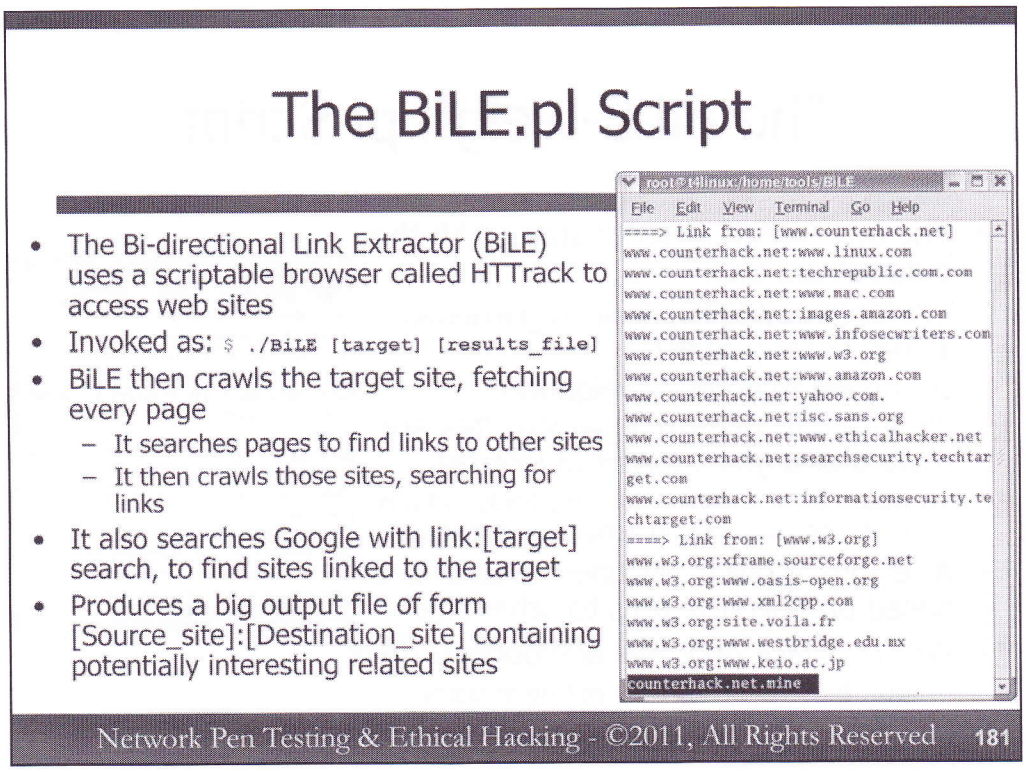



The security consultancy and research company Sensepost has released a variety of high-quality tools useful for penetration testers and ethical hackers. Among these tools is a suite called BiLE, which stands for Bi-directional Link Extractor. That is not a typo in the capitalization of its name; it is indeed BiLE.

The goal of this suite is to perform various automated recon steps using web crawling, DNS lookups, and Google searches to find as many target machines that might be associated with a given domain as possible. The suite also includes other tools that help look up information about the target environment.

Each of the tools included in the BiLE suite is a Perl script. Most of the tools end in a .pl suffix, except for the last two, jarf-rev and jarf-dnsbrute. Those last two are Perl scripts as well, but they lack the .pl suffix in the released version of the BiLE suite.

Note that the various scripts included in the BiLE suite are meant to be run in an iterative fashion. That is, for many of the tools, the output file of one of the scripts is meant to be fed as input to the next script, which performs additional lookups and analysis, creating output that in turn could be fed into another script. The output is stored in a series of files, however, and is not usually redirected or piped between BiLE components on standard input or standard output.




The first member of the BiLE suite is called, appropriately enough, BiLE. This tool takes a target website's full domain name (e.g., www.counterhack.net) as its first argument, and the name of a file in which to put its results.

BiLE then uses the scriptable text-based browser called HTTrack (which must be installed to use BiLE) to crawl the target website. A web crawler is software that systematically grabs pages from a target website, in an operation that some people refer to as "spidering" the website. When it finds a link within the target website to another page in that site, it gathers that other page as well, recursively gathering every page in the target site. BiLE analyzes each of these fetched pages to find links to other web sites. It then crawls/spiders those other sites, fetching all of their links.

Next, BiLE performs a Google search on the target website, inserting the "link:" directive in front of the web site name. This search makes Google respond with a set of URLs that link to the given target domain name.

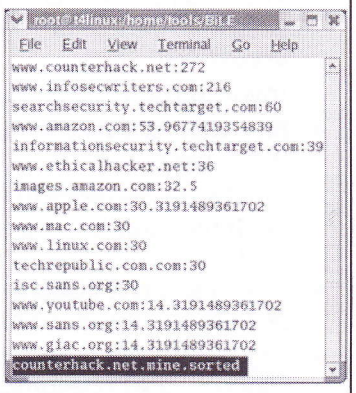
BiLE takes the crawling and Google results and creates a big file that shows all of the site-to-site links discovered in its crawling, sorted by web server domain name. In the BiLE output file example on the slide, we can see all of the sites that www.counterhack.net links to, as well as all of the sites that they link to, indicating that there is a possible relationship between these sites.

# The BiLE-weigh.pl Script

- The BiLE-weigh.pl script starts with the output from BiLE

```
$ ./BiLE-weigh.pl [site_of_interest] [BiLE_output.mine]
```

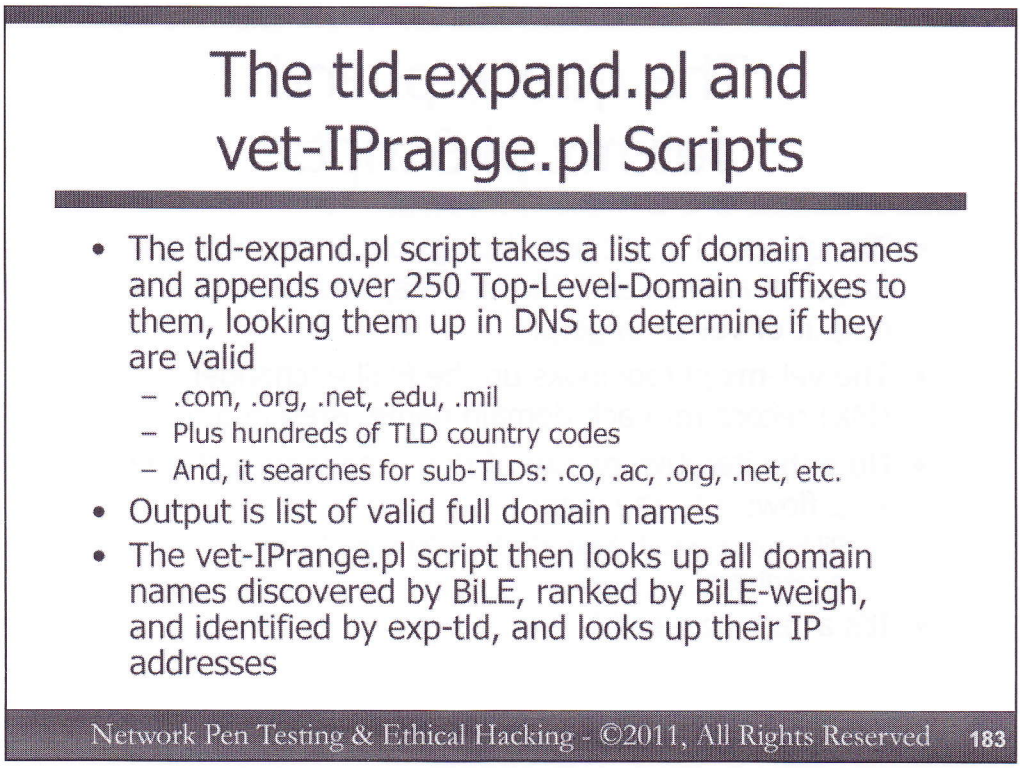
- It applies a complex weighting algorithm to determine which sites are related to a given target of interest
  - Based on the number of links back and forth between the target machines and domains
- A relative score is assigned to each site, based on how it relates to other sites
- We now have potential additional targets
  - But, beware! They may not be in scope!






The next script in the BiLE suite is called BiLE-weigh.pl. It takes the output file from BiLE, as well as a site of interest, and applies a complex formula to determine which sites relate most to the given site of interest. Based on the number of links back and forth between the site of interest and each site in the BiLE file, a numeric ranking is applied to each web site. The numbers themselves are irrelevant; it's the relative size of these numbers that count. The higher the number, the more likely the given website is to be associated with the site of interest.

Using this technique, a tester can find other potential targets to test. But, beware! These other machines may be outside of the agreed-upon scope of the test. Or, they could belong to an entirely different organization that is merely "related" to the target organization. Attacking them would likely be illegal without appropriate permission. Such findings should be reviewed with the target organization before they are scanned.

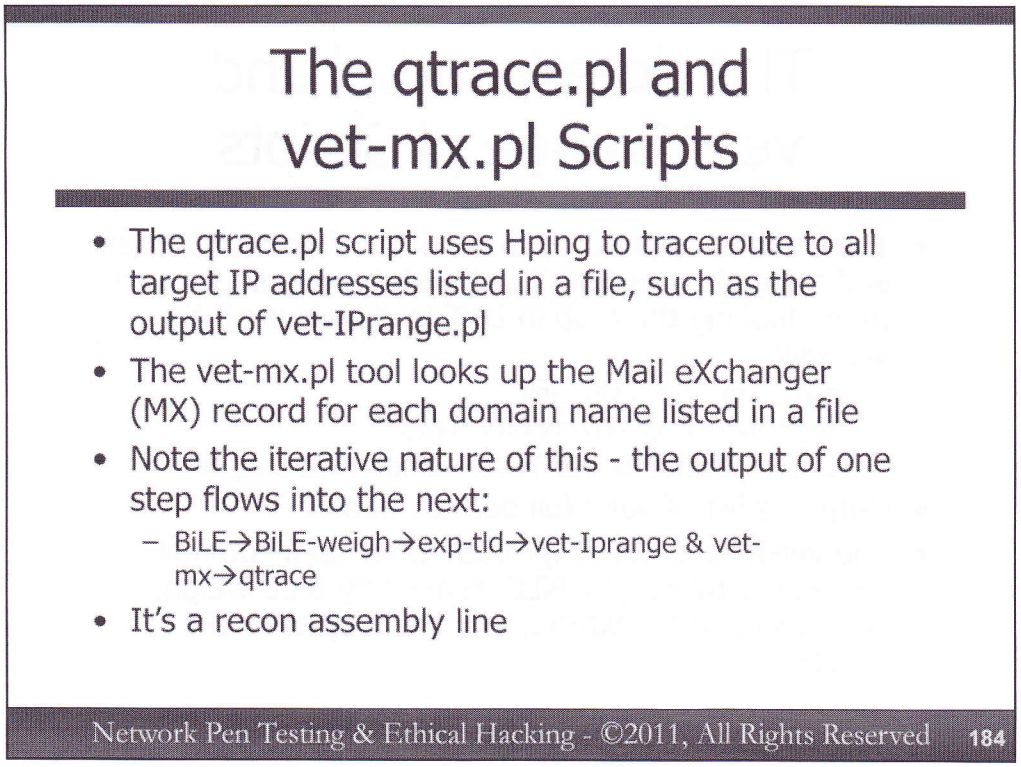



The next scripts in the BiLE suite are called tld-expand.pl and vet-IPrange.pl. The first one looks for a set of given names with other Top-Level-Domain (TLD) suffixes. The script takes the domain name, and then appends over 250 TLDs to it (including the familiar .com, .net., .org, .edu, and .mil, as well as hundreds of country codes). It then performs a DNS lookup on the results to see if there are any registered domains. For example, when provided www.counterhack.net, this script will search for: www.counterhack.com, www.counterhack.org, www.counterhack.us, www.counterhack.uk, www.counterhack.jp, and many, many others.

It will also lookup sub-TLDs, including the .co, .ac, .org, .net, and others. For example, it will attempt to lookup www.counterhack.co.uk, etc.

The result is thousands of DNS lookups, attempting to find other domains associated with the target. The output is a list of valid full domain names that were discovered to have a DNS record associated with them.

The vet-IPrange.pl script then looks up in DNS all of the domain names discovered by the tools run so far, including BiLE, BiLE-weigh, and tld-expand. Its output includes a list of IP addresses potentially associated with the target organization.



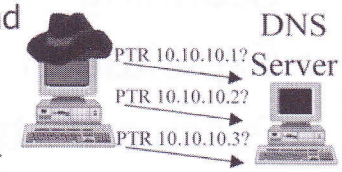

The qtrace tool takes a file containing a list of IP addresses as its input. It then determines the set of router hops between the system running qtrace and each target IP address, generating an output file that shows the hop-by-hop path to each IP address. We'll cover the technical details of traceroute techniques later in the class. To perform its traceroutes, qtrace relies on the Hping tool, which we will also cover in depth.

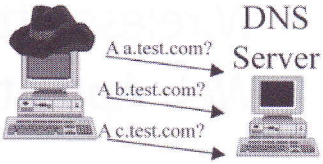
The vet-mx script looks up the Mail eXchanger (MX) records in DNS associated with each domain name provided in an input file. That way, we can determine the mail servers associated with the target and its related sites.

Note the general flow of all of the tools in the BiLE suite (as well as most of the other tools released by Sensepost). Each one iterates on the output of the previous tool, making a reconnaissance assembly line, with each tool refining and/or expanding on the results of the previous tool.

# The Jarf-rev and Jarf-dnsbrute Scripts

- Jarf-rev and Jarf-dnsbrute are helpful in getting information from DNS even when zone transfers are blocked
- Jarf-rev takes a target network range and performs reverse DNS lookups on all names in the range
  - Really helpful in finding obscure hosts that someone created a reverse lookup record for
- Jarf-dnsbrute takes a given domain and a file containing words, then performs a DNS lookup on each domain name
  - Can generate a huge amount of DNS queries
  - Still quite helpful



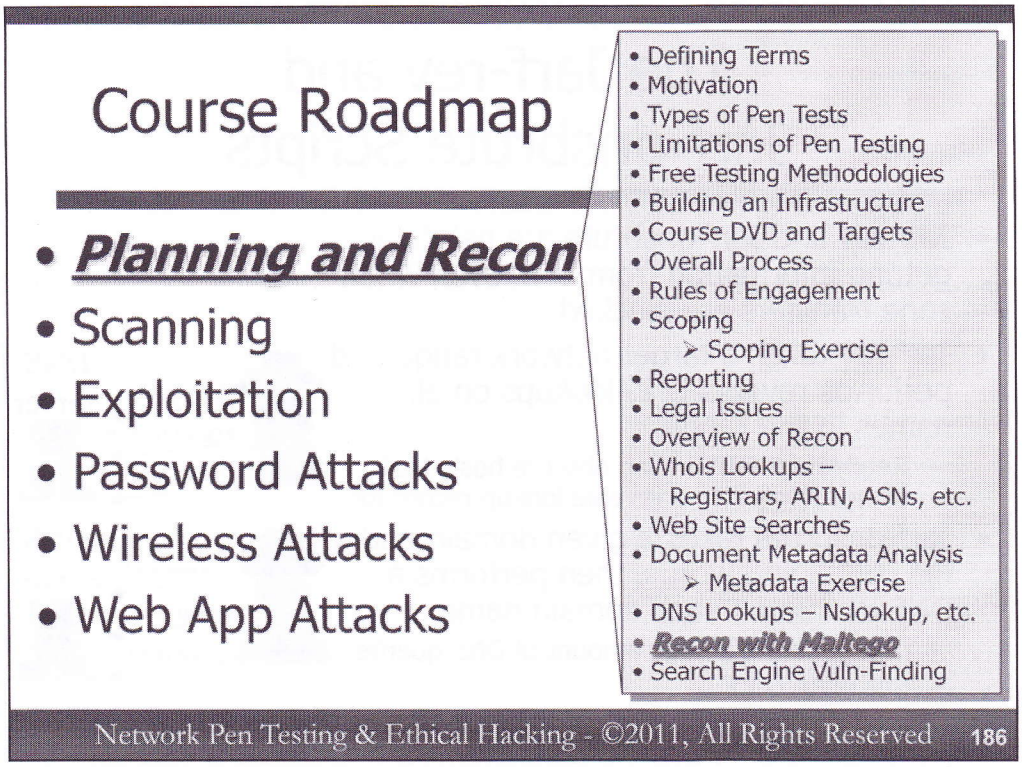



Two final tools in the BiLE suite are especially helpful: jarf-rev and jarf-dnsbrute. The first of these tools takes as its input a range of IP addresses, such as 10.10.10.1-10.10.10.255. It then performs a DNS reverse lookup for each individual IP address in the range to find if there are any records associated with that IP address. If there are, it is likely a valid IP address and should be included in our list of potential targets. This technique can be helpful in finding obscure hosts that are not indicated anywhere else, as long as they have a reverse DNS record associated with them.

The jarf-dnsbrute program takes a given domain name (such as counterhack.net) and a dictionary file containing a list of words with one word per line. It then prepends each word to the domain and performs a DNS lookup. For example, if the dictionary file contained the alphabet, with one letter per line, this script would look for a.counterhack.net, b.counterhack.net, and so on. Depending on the size of the dictionary file, this script could generate a very large number of DNS searches. Still, with a carefully generated dictionary file, it could find a needle in a haystack of other systems to include in the test.

Both of these tools are very helpful if the target environment's DNS servers block zone transfers. Of course, with a zone transfer, all of this information can be pulled out directly from the DNS server by transferring its zone file. But, with zone transfers blocked, jarf-rev and jarf-dnsbrute can still give us some useful information by automating a significant amount of DNS query work.

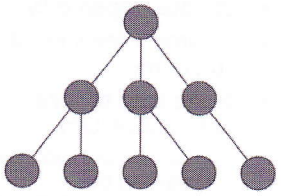



As we saw with BiLE, information gathered during a penetration test is often a series of related tidbits, starting with some initial piece of information, which is then used to look up more information in an iterative process. Each step reveals more and more about a potential target, but the penetration tester must make sure that all newly gathered data points still apply to the scope of the given project.

Working along the same lines, another very helpful reconnaissance tool is Maltego, which provides a graphical view of the relationship of information gathered about some target. Maltego displays the hierarchical relationships of information points allowing a penetration tester to perform detailed reconnaissance of target domains, IP addresses, or even people, in a systematic fashion.

# Maltego for Pen Testers

- Maltego, by Paterva, is a general-purpose recon tool
  - Runs on Win, Lin, and Mac OS X: http://www.paterva.com/maltego
  - Commercial version
  - Community edition, with limits:
    - 15-second nag screen, can't save results, limits zoom
    - Only 75 transforms per day, no parallel running of transforms
- Built on concepts of "transforms"
  - Take one piece of data and convert it to another through a lookup of some sort
  - Over 50 different kinds of transforms, such as:
    - Domain name to IP address (dns)
    - IP address to org name (netblock)
    - Org name to person's name (whois)
    - Person's name to PGP key (public key servers)
    - PGP key to person's name (who signed the key?)
    - Persons' names to phone numbers (phone lookup)

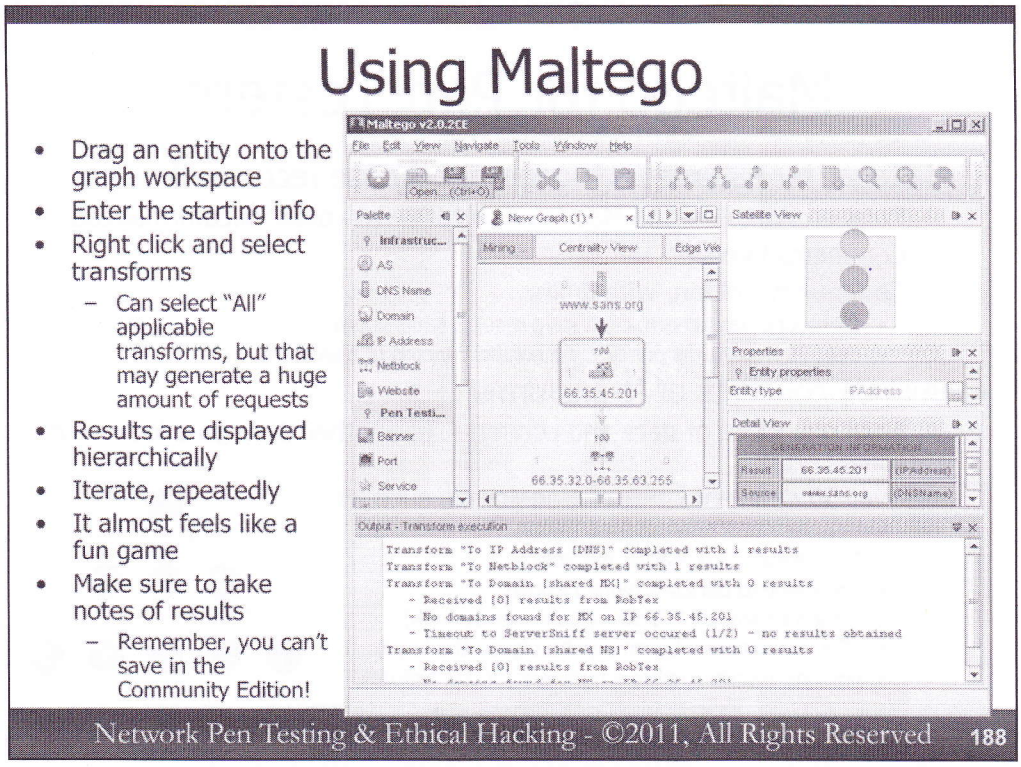


Paterva's Maltego is a general-purpose reconnaissance tool that runs on Windows, Linux, and Mac OS X. It is available in two versions. The commercial version requires a subscription, which costs approximately $ 430 per year. The community edition is free, but includes some limitations. First, the community edition version pops up a 15-second nag screen asking you to subscribe to the commercial version every time the tool is launched. Furthermore, the free version prohibits the saving of results between sessions, and limits the level of depth a user can zoom into results. Also, this version is limited to running 75 transforms per day. That is, 75 transformations from one piece of information to another are allowed every 24 hours. Also, all searches run sequentially in the community edition, without parallel threads to improve performance.

Maltego is built on the concept of transforms, taking one piece of information and performing a lookup to determine another piece of information. For example, we could start with the domain name of a given web server (www.someserver.com). We could then apply a transform that converts that name into an IP address. Maltego's transform will simply perform a DNS lookup to find the IP address. We can then apply another transform to map the IP address to an organization's name, through a netblock lookup at a Regional Internet Registrar. We can then do a whois lookup on the org name, to determine the name of people associated with that organization. We can proceed with additional transforms that will look up that person's name in various public PGP key servers to determine their public PGP key. We can then map that key to the names of people who have signed the key, getting the name of more people. We can then perform phone directory lookups of those names, and so on.

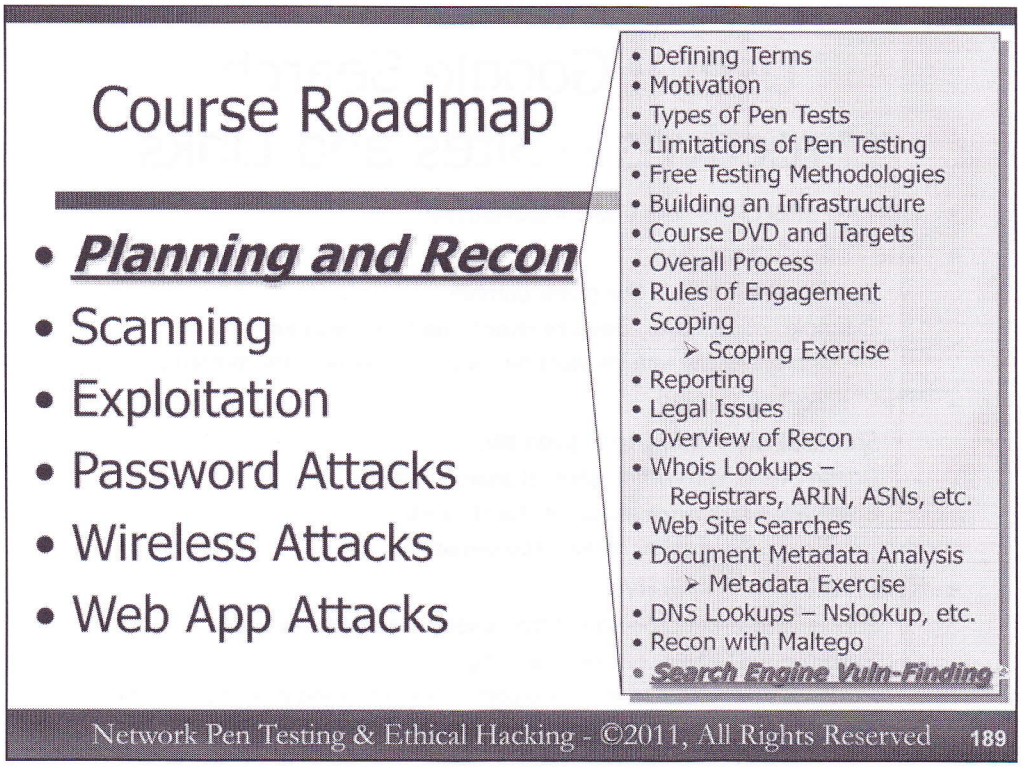
Maltego plots all of the results in a hierarchical diagram showing how this information relates together.




To use Maltego, we first drag an entity from the palette on the left onto our graphical workspace. We could drag any starting information point, such as a DNS name, a domain, or IP address. We then enter the appropriate information associated with that given entity, such as the DNS name of www.sans.org.

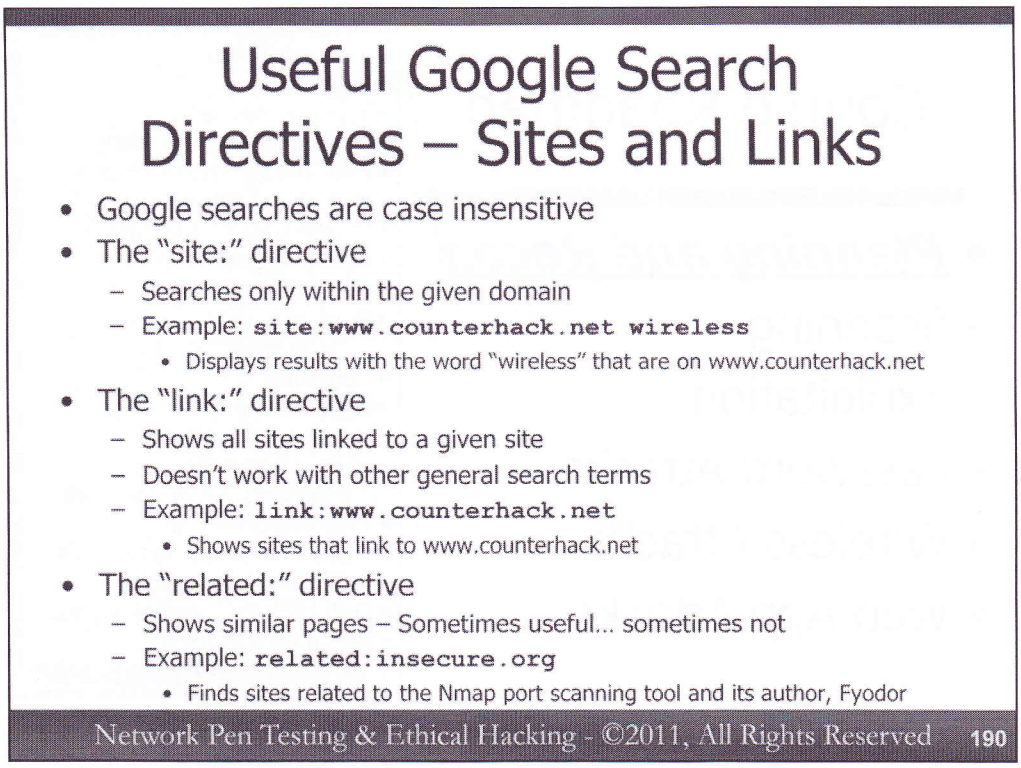
We then can right click on the entity in our workspace to get a list of transforms available. We can choose any one of the associated transforms, or apply all of them. Choosing "All" transforms may take quite some time and generate a lot of traffic. The resulting information pulled back from various public sources is displayed hierarchically related to our initial data point. We can then iterate, right clicking on any new data point and applying new transforms.

The result almost feels like a game, as we move from data point to data point, pulling more and more information about the target. But, we always have to remember to check our results to verify that we are still operating within the scope of the given project. Otherwise, it can be very easy to careen out of bounds with our reconnaissance, looking for information about systems or people that are not included in our given project.




Our next step will be to use publicly accessible search engines to look for signs of vulnerabilities on systems. Google, Yahoo, and Microsoft's Live Search all contain a great deal of information that could indicate the presence of vulnerabilities in systems associated with the target environment. By sending the appropriate queries to the search engines themselves, we may be able to identify vulnerable systems without actually sending any packets to those systems directly.

We'll focus on Google, because of its comprehensiveness and widespread use. Similar techniques can be applied to other search engines, but their specific syntax often differs and is less powerful than Google's.

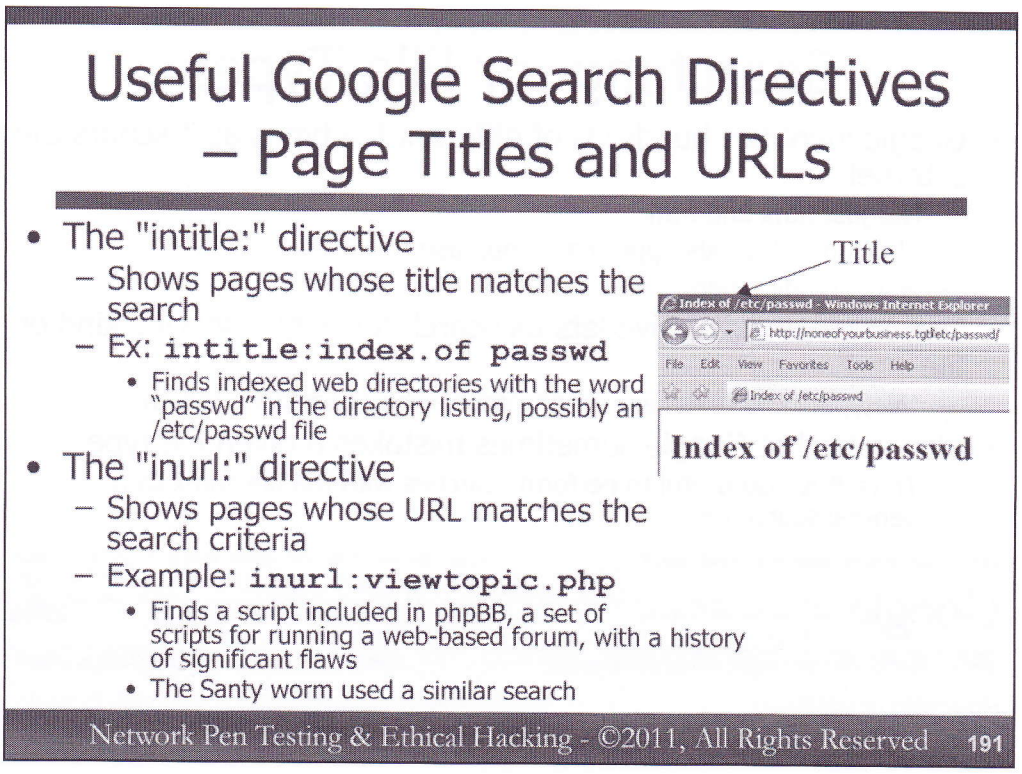



To understand how we can use search engines (and particularly Google) to find vulnerable systems, we are going to spend some time looking at more advanced search capabilities within Google. By taking the various search principles we'll discuss over the next several slides and combining them together in creative ways, we can use Google to find flawed systems in our target environments.

Let's explore directives associated with examining specific websites and domains. The "site:" directive allows an attacker to search for pages on just a single site or domain, narrowing down and focusing the search. For example, if you only want to search pages in the counterhack.net domain for the occurrence of the string "wireless", you could do a search for **`site:counterhack.net wireless`**. Have you ever seen a website that doesn't have its own built-in search capability (like www.counterhack.net), or one with a really lame search capability? With Google's "site:" directive, you now can use Google to search for results just associated with that site, relying on the power of Google and its very flexible search directives and operators. The "site:" directive can get very specific, like **`site:example.exampleuniversity.edu`**, or very broad, like **`site:.edu`**, which would cover all sites with a .edu suffix..

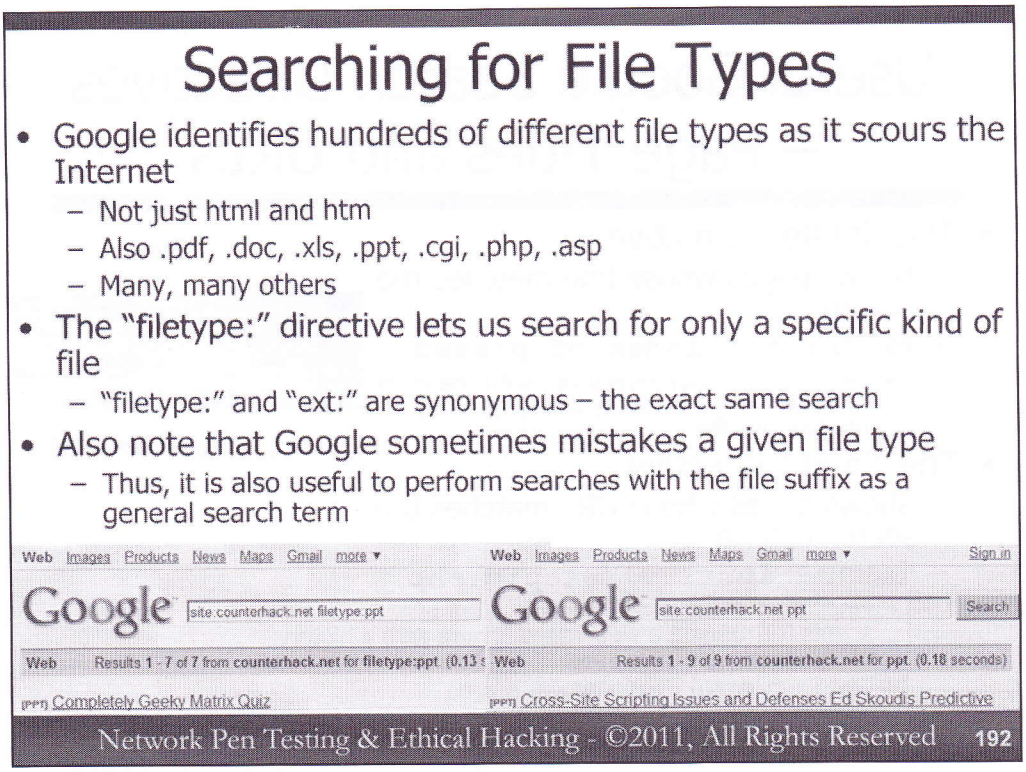
The "link:" directive shows sites that link to a given web site. During recon, this directive can be used to find business partners, suppliers, and customers. To look for everything linking to www.counterhack.net, you'd do a search on **`link:www.counterhack.net`**. Note that "link:" searches look for links that match exactly the given domain name used in the search. That is, a search for **`link:counterhack.net`** would look for links of the form "http://counterhack.net/[whatever]", but would *not* identify links to the website www.counterhack.net. For those, you'd have to search for **`link:www.counterhack.net`**.

The "related:" directive shows pages that have similar content and links to the searched page. Based on Google's patented Page Rank algorithm for analyzing pages, this search directive could return information about very subtle business relationships that might be missed in other search types. For example, if a given target bank has a business partnership with another bank, there may be no direct links between the sites. However, both may contain similar text terms, and they both may link to the same series of third-party sites. The "related:" directive then gives the tester a chance of finding out about these relationships.



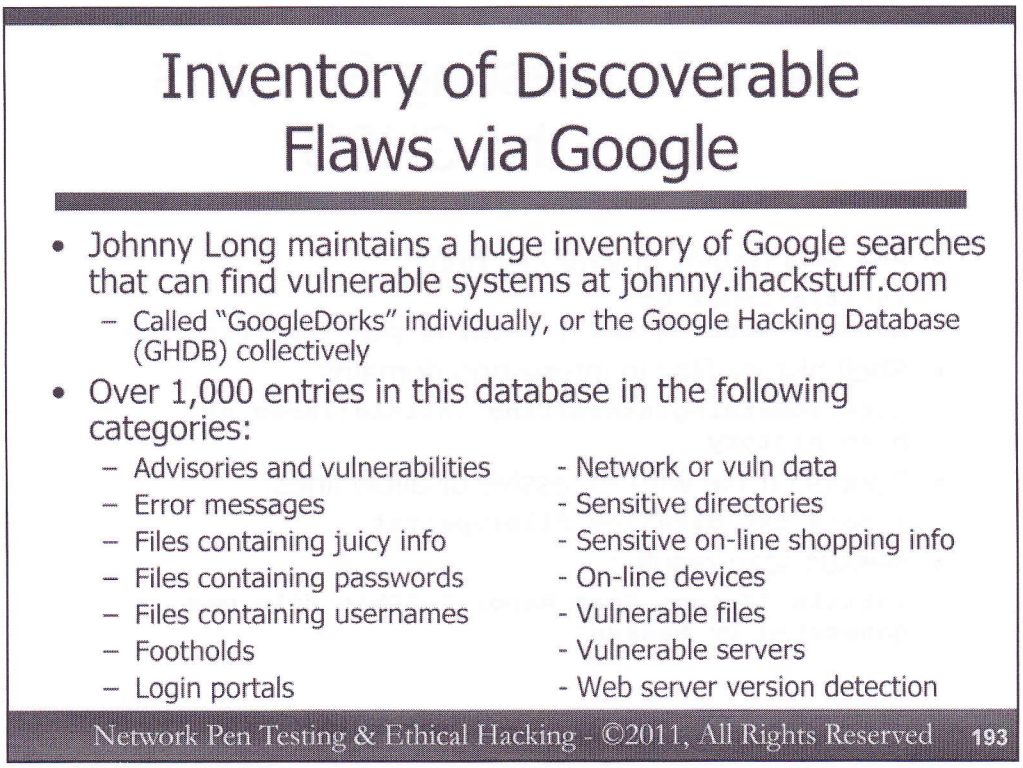

Next, let's look at searches associated with page titles and URLs. Often, we want to find pages with specific text in their titles. The title is the name of the page, often displayed at the very top of the browser window. We can perform searches within this title text by using the "intitle:" directive. One of the most useful title types to look for involves directory indexing on a website. Some web servers are configured to allow users to send requests not only for individual pages (e.g., index.html), but for directories (e.g., /exampledirectory/). On websites configured with directory index functionality, the web server will return to the browser the contents of that directory, with an auto-generated page with a title that includes the text "Index of". Thus, we can search for **`intitle:index.of passwd`** to look for directories that have a file in them called passwd. We may find an /etc/passwd file. Legitimate /etc/passwd files will have lists of user accounts on the target machine. If shadow passwords are not being used, it also might have encrypted passwords, suitable for cracking. Note that a number of the passwd files in interesting domains discoverable by Google searches are either honeypot /etc/passwd files, or sites that may try to exploit a browser surfing there! So, please be careful when doing this kind of search, unless you restrict it using the "site:" directive to domains that are more trustworthy.

The "inurl:" directive lets us search for specific terms to be included in the URL of a given site. This can be helpful in finding well-known vulnerable scripts on web servers, including CGI, ASP, JSP, PHP, and others. For example, searching for **`inurl:viewtopic.php`** will find sites with URLs that contain "viewtopic.php" in them. That script is commonly associated with the phpBB suite of tools for implementing web-based discussion forums. Historically, there have been numerous flaws in phpBB implementations, so locating those sites is helpful if a new vulnerability is discovered. Back in 2004, the Santy worm used a variant of this Google search to find vulnerable phpBB installations so that it could infect the website and deface its web page. Santy's initial spread located vulnerable systems via Google, rather than by scanning. Only after the initial set of targets was compromised did Santy turn to traditional scanning to find new vulnerable targets.




Often, we want to search Google for some specific kinds of files. Most of the web consists of html and htm pages, but there are numerous other kinds of files that interest us. As Google scours the Internet finding new websites and adding their pages to its search directory, it recognizes several hundred different file types, allowing us to search for those types of files. For example, we can look for .pdf files. Or, to make things more interesting, we can search for .xls files, which are commonly associated with Excel spreadsheets. We might also look for .ppt files to find PowerPoint presentations. Some organizations inadvertently put very sensitive .xls or .ppt files on their websites, for which we could search.

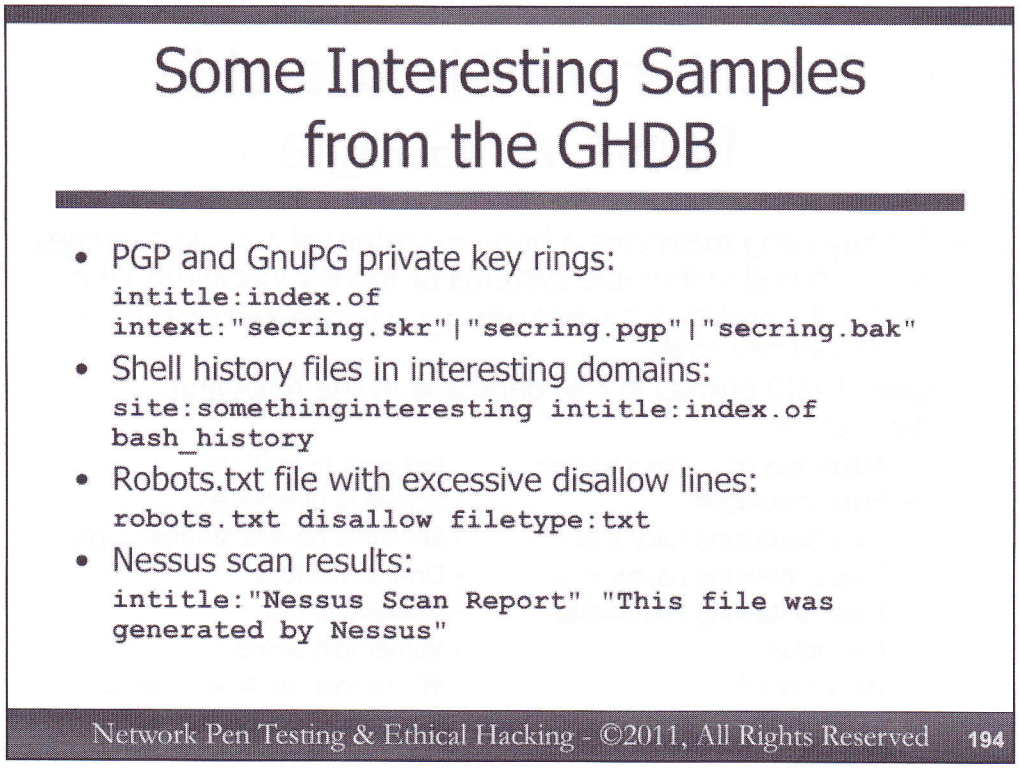
To perform such searches, we have two options. The first is to rely on Google's "filetype:" directive followed by the suffix of the file we want to find. Note that the "ext:" directive does the exact same thing as the "filetype:" directive; it is exactly the same search. For example, we can search for PowerPoint files in the counterhack.net domain by looking for **`site:counterhack.net filetype:ppt`** or **`site:counterhack.net ext:ppt`**. Our second option is to look for the suffix of the file as a general search term. Sometimes Google gets confused about a file type, messing up the appropriate association and omitting that given file from the filetype: results. Using the file suffix as a general search term without the filetype: or ext: directive will usually give us more results, because not only will we get files that have that suffix in their name, but we will also get web pages that merely include the text associated with that suffix. For example, if we search for **`site:counterhack.net ppt`**, we will not only get PowerPoint files, we'll also get a series of web pages that include the text ppt.




Johnny Long maintains a list of useful Google searches to find vulnerable systems at his website, johnny.ihackstuff.com. He calls each individual search a GoogleDork, and the entire inventory of all of these searches is known as the Google Hacking Database (GHDB). There are well over 1,000 different searches in the GHDB that can find several varieties of security flaws and related issues, all by simply searching Google.

Johnny has separated the entries in the GHDB into numerous categories. All of the individual categories are listed on the slide, but some of the most important and interesting to us are:

*Advisories and vulnerabilities*: These searches find vulnerable systems, usually by identifying a known flawed CGI script using the inurl directive, or a page with known flaws identified with the intitle directive.
*Files containing juicy info:* These searches find files that are often associated with caches and logging. While they don't look for passwords directly, this cache and log information could be useful in learning more about the target organization.
*Files containing passwords:* Numerous tools generate files that contain either clear text passwords or encrypted/hashed passwords. These searches identify when such files are available via a web server.
*Footholds:* These searches locate sites where an attacker may be able to get a foothold that can later be used to compromise the server. A lot of these searches find admin login pages for various common web-based software environments. The "Login portals" category is very similar.
*Network or vulnerability data:* These searches find pages that hold logs and/or configuration information about network devices, such as firewalls, VPNs, routers, Intrusion Detection Systems, etc.
*Sensitive on-line shopping info:* These searches find web sites that may reveal sensitive on-line shopping info, including customer orders, weak shopping cart implementations, and overly detailed product information.
*On-line devices:* This category of searches helps locate web-based video cameras, printers, and various kinds of appliances.
*Vulnerable servers:* These searches locate web servers that may have a vulnerability, a category very similar to "Advisories and vulnerabilities".



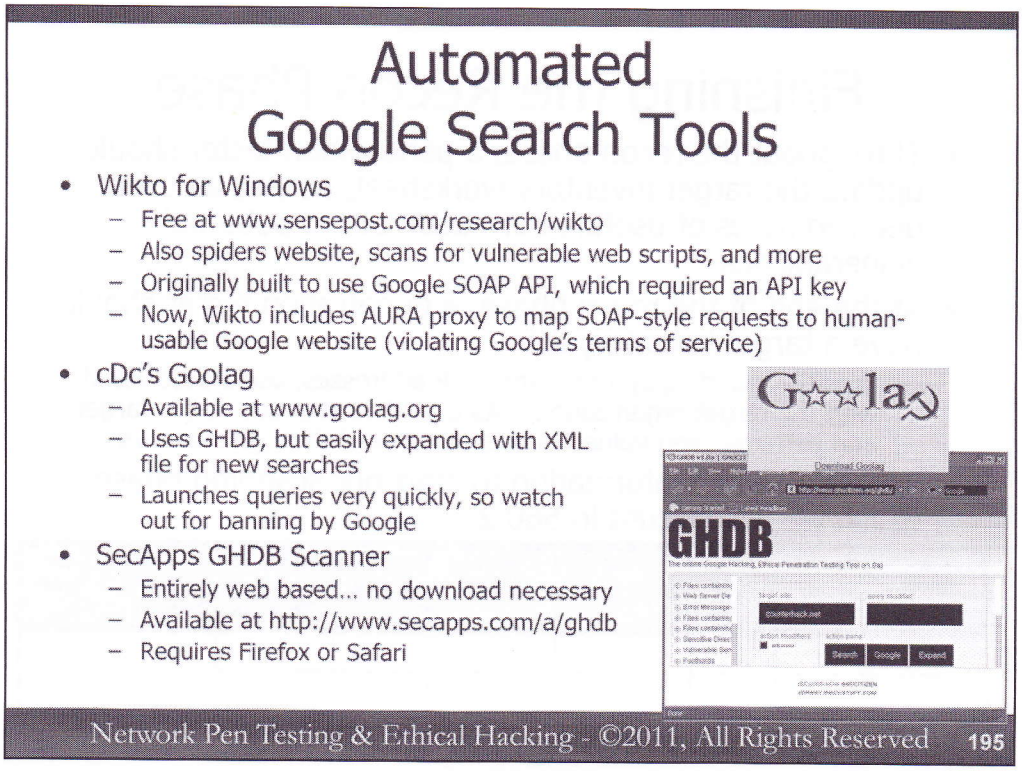

While the GHDB includes many hundreds of eye-opening Google searches to find vulnerable sites or sensitive data, let's explore a handful of them to get a feel of the power of the GHDB and Google.

The first search on this slide will look for web sites with directory indexing, in which we have files named secring.skr or secring.pgp or secring.bak. These are the common file names where private keys are stored for the Pretty Good Privacy (PGP) and Gnu Privacy Guard (GnuPG) tools. Although the private keys are encrypted with the user's passphrase, having them available on a web site and searchable via Google is a really bad idea. An attacker could grab the files and launch a password guessing attack against them.

The next search on the slide finds bash shell history files on websites, showing the commands some user typed into the shell. By putting in an interesting domain with the `site:` directive, we may find history files with sensitive information in them.

For the next search, we look for robots.txt files. A website administrator can use a robots.txt file to tell well-behaved search engine crawlers to ignore certain directories or pages on the website, with the Disallow syntax in this file. That way, those pages or directories won't be searchable or cached by the search engines. But, robots.txt is a double-edged sword, indicating where sensitive items may be stored. With the Google search on this slide, we can find websites that have a large number of disallow lines, indicating that a large number of individual directories or pages on the site have been disallowed. They might have something interesting to hide.

We can even ask Google to find sites that have output files from the Nessus vulnerability scanner. Somehow, this file was placed in a directory under the document root of a web server, so that Google could find it and make it searchable. A list of vulnerabilities of a target site could be very useful for an attacker. We can save time running a vulnerability scanner by analyzing the results of earlier scans already done by the target organization.

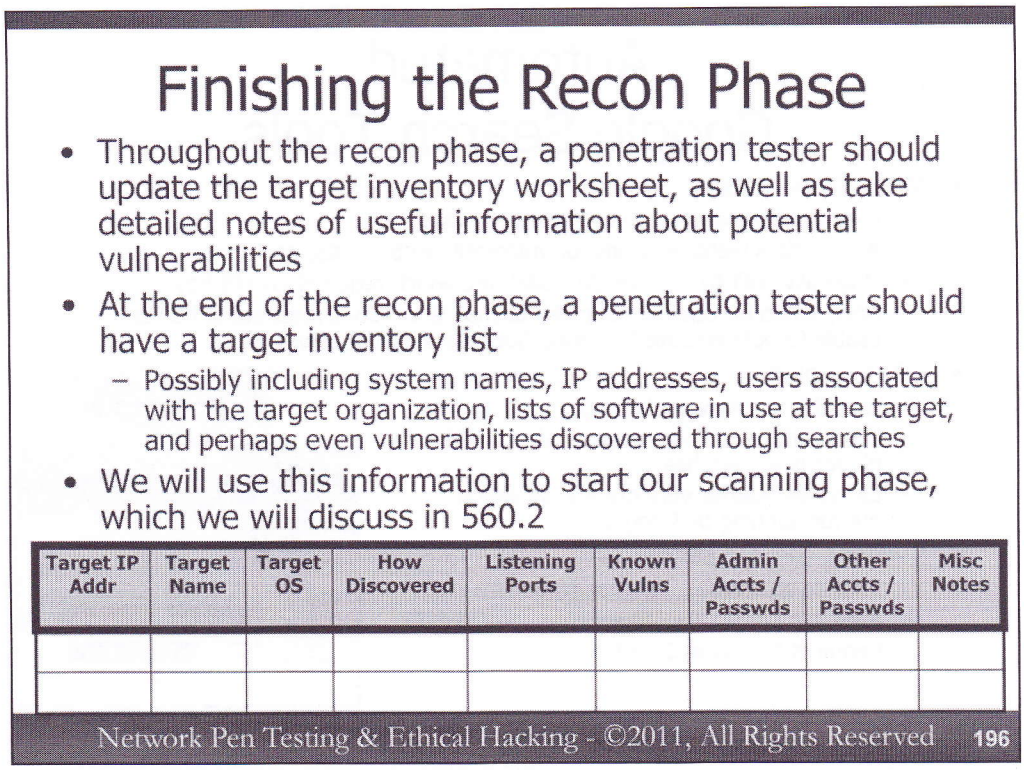



With the GHDB containing well over 1,000 searches, you aren't going to want to type in site:yourdomain.com followed by each different search string to find vulnerabilities in your domain. That would require weeks of effort, and, if you have multiple domains, would take a long time indeed. Instead, you'll likely automate the process. There are a variety of tools to automate Google searches to find vulnerabilities. Three of the most popular and powerful are Sensepost's Wikto, Cult of the Dead Cow's Goolag, and GnuCitizen's GHDB Scanner.

Wikto runs on Windows and performs Google searches using the GHDB against one or more user-provided domains. Wikto also provides several other features, including a scan of the target web servers looking for well-known vulnerable scripts. It has a web spider that will automatically crawl the target website to find directories and pages, and many other features. The original version of Wikto required a Google SOAP API key for sending requests to Google. Because Google stopped issuing SOAP API keys back in December 2006, Sensepost also released AURA, which stands for API Usable / Re-usable Again. This Google search proxy tool, bundled with Wikto, converts SOAP-style requests from Wikto to Google via the human-oriented Google web page, and then scrapes the results that come back. The Wikto user therefore no longer requires a Google SOAP API key. By scraping the results, we again have a likely violation of Google's terms of service, so be careful.

Other tools also send queries directly to Google's human-oriented interface at www.google.com, scraping the search results that come back, without the use of a SOAP proxy. The Goolag tool from the Cult of the Dead Cow (cDC) relies on the GHDB, and can easily be extended by writing new searches in a convenient XML format. Goolag runs very fast, launching a large number of queries. Thus, use of this tool can get an IP address banned from Google very quickly, so be careful in running it.

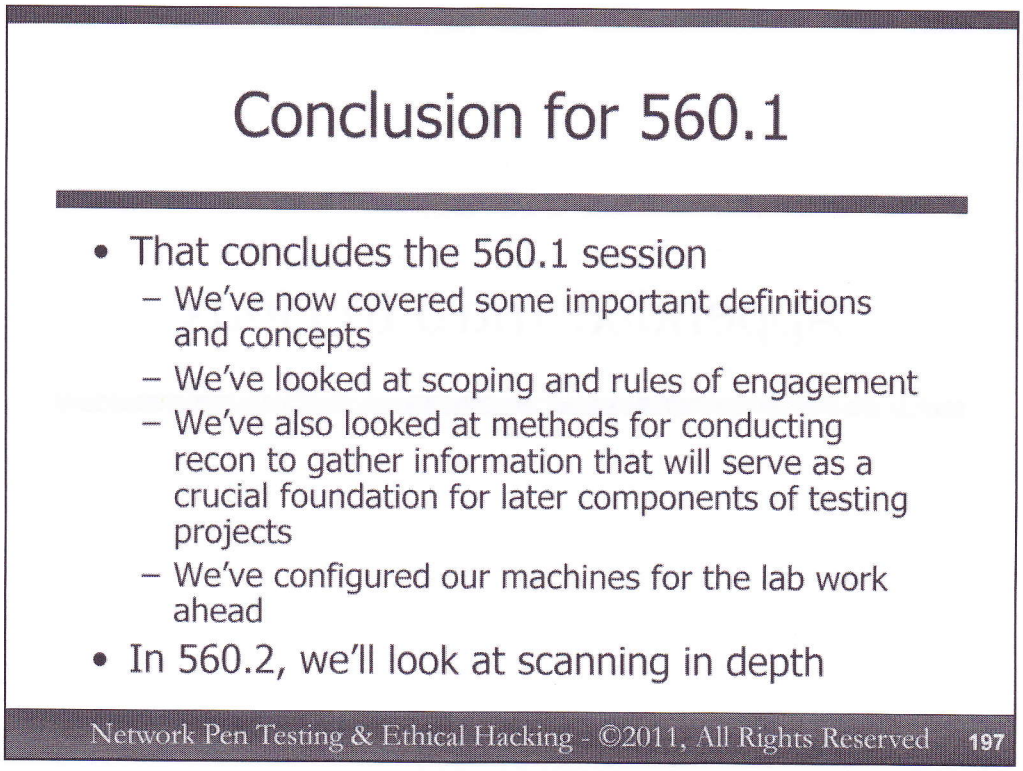
The SecApps team has released a web-based application that provides GHDB searches as well. No download is necessary, as all queries are run through the web-based interface. The tools relies heavily on browser scripts and currently only runs in Firefox or Safari.




Throughout the reconnaissance phase, a penetration tester should continue to populate the target inventory worksheet for each newly discovered system. Additionally, the tester should make detailed notes of other information assets and potential vulnerabilities identified during this phase.

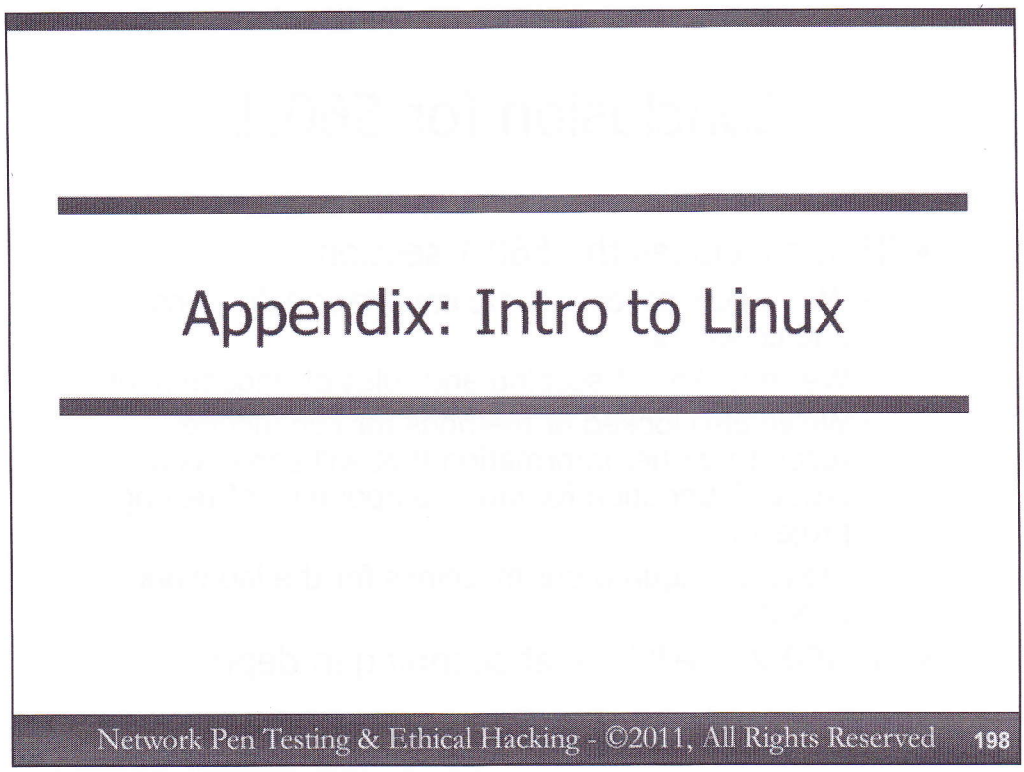
At the end of the recon phase, a tester should review the target inventory list, adding any final information that was discovered, and highlighting specific areas in the "Misc Notes" section that require further analysis and exploration.

This target inventory list will be a crucial input into our next phase, Scanning, which is described in 560.2.

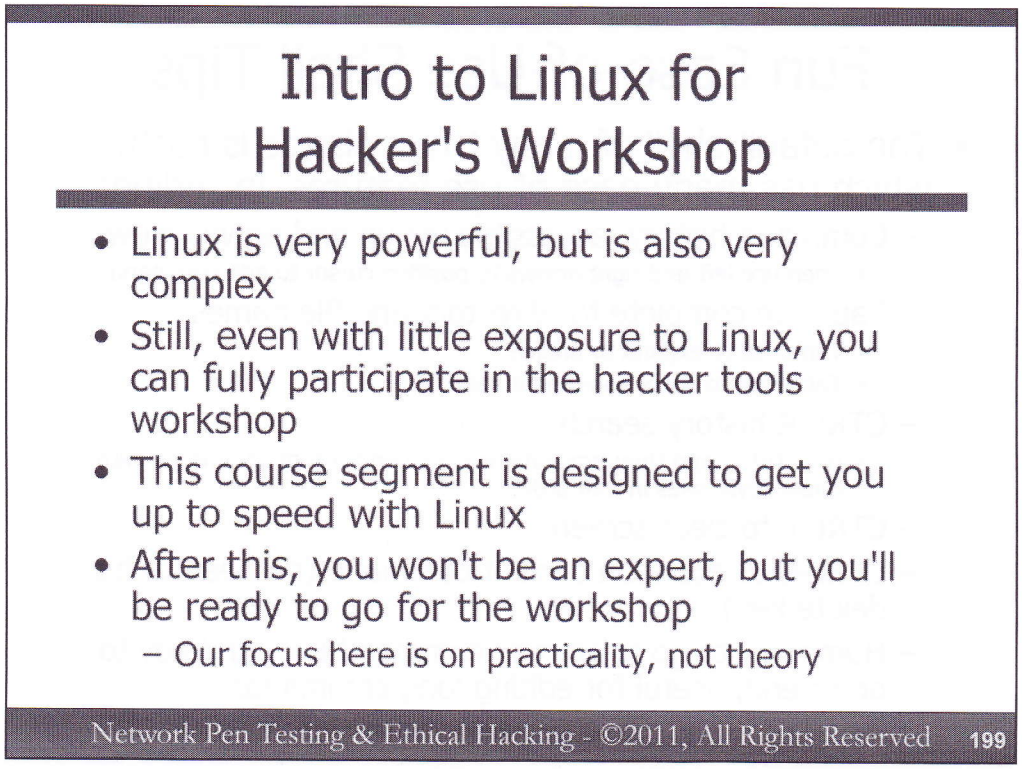



We now conclude our 560.1 section, in which we've addressed some important concepts in penetration testing and ethical hacking. Among the most important topics for the day have been proper scoping and formulation of rules of engagement. We also discussed the recon phase used in many penetration tests and ethical hacking projects, gathering information that will act as a firm foundation that testers will leverage for the remainder of a testing project. We've also configured our machines to prepare for the lab work we'll perform throughout the remainder of the class.

In our next section, 560.2, we'll take an in-depth look at scanning, the process used by penetration testers and ethical hackers to determine openings in the target environment.

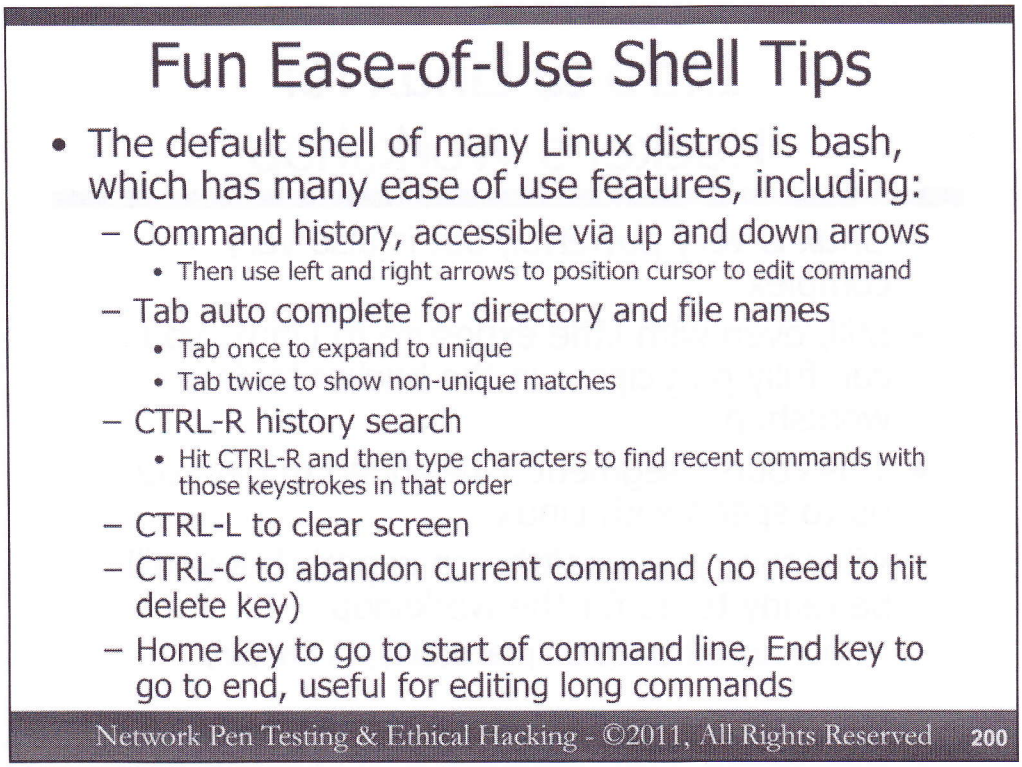



The next section contains an appendix that covers and Introduction to Linux. If you are new to Linux, you should review the section that follows, as we will be relying on its concepts throughout the remainder of the class. If you already have a good working knowledge of Linux, you should not need in-depth review of the following materials, but a cursory glance at them may be a helpful refresher.




To fully participate in this class, you need a basic working knowledge of Linux. We're not expecting you to be an expert, by any means. Everything you need to know about Linux for the workshop will be covered in this introductory workshop.

We will not be covering Linux installation. You should have done that before coming to the session, as described in the class requirements.




Throughout this session, we will be using bash as a command shell, one of the most common command shells in Linux distributions today. This shell includes many ease of use features that make interacting with Linux far simpler. You should memorize each of these items, as they will save you much time and effort, making Linux a lot friendlier for you.

Bash, like many other shells, remembers your shell history, letting you access it by hitting the up and down arrows to access and edit recent commands, which you can re-run by simply hitting enter.

Once you've chosen a previous command, you can hit the left and right arrow keys to position your cursor to edit the command.

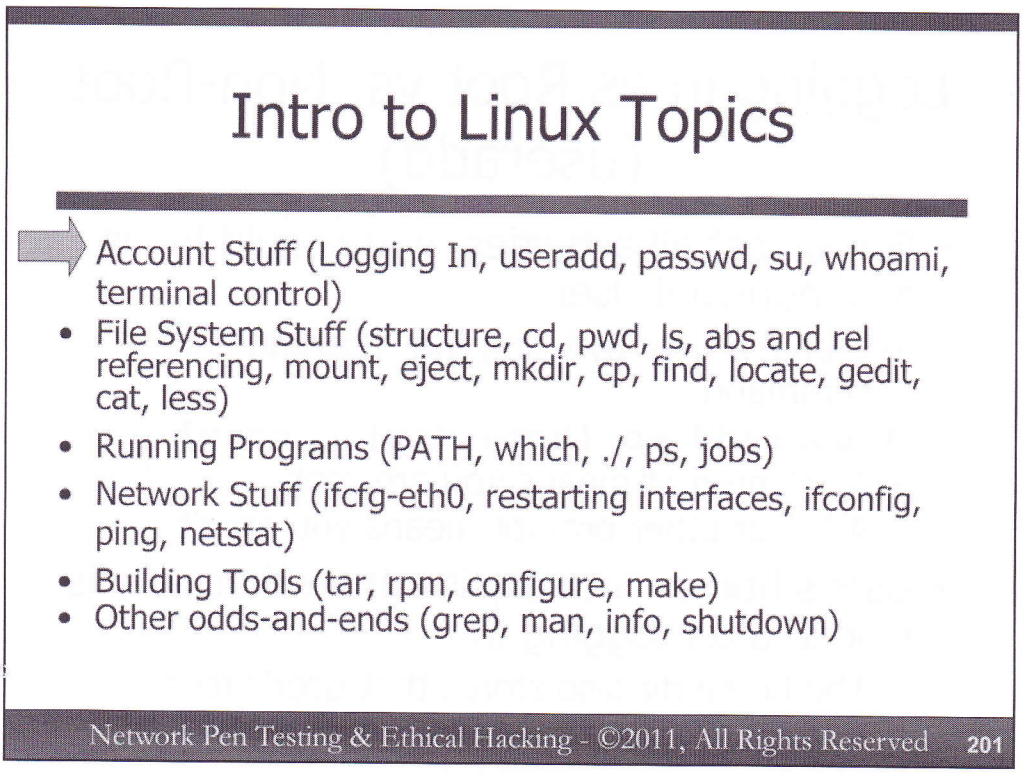
Also, bash supports tab auto-complete for the names of directories and files. When accessing something in the file system, just hit Tab for the shell to expand it to a unique name that matches what you've typed so far. If there are multiple items that match what you've typed (i.e., there is nothing unique yet), you can hit Tab again to show the names of all files or directories in your current working directory that match what you've typed so far. That is, Tab expands to a unique value, and Tab-Tab shows all items that match what you've typed so far if nothing is unique.

You can also search your history in bash by hitting CTRL-R at the start of a command line. Then, start typing characters, and bash will jump back to the most recent command that has the characters you typed in that order. You can then hit Enter to re-run that command, or the left or right arrow keys to edit the command.

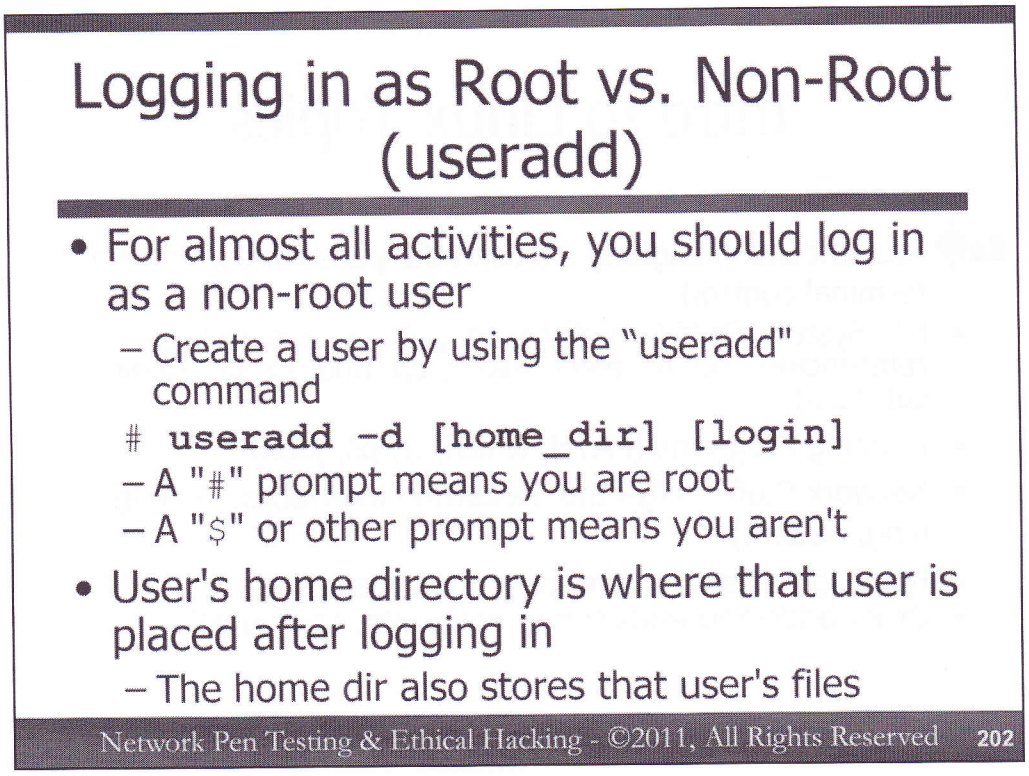
The CTRL-L option clears the screen, or you can simply type "clear".

The CTRL-C command lets you abandon the current command and get back to the command prompt. There is no need to delete the current command by holding down the backspace or delete keys. Just hit CTRL-C to get rid of the current command.

And, finally, the Home key included on some keyboards lets you jump to the beginning of a command-line, while the End key lets you jump to the end. These options can help you jump around in long commands to make altering them easier.




Here's an outline describing the topics we'll cover. We'll start with Account stuff.



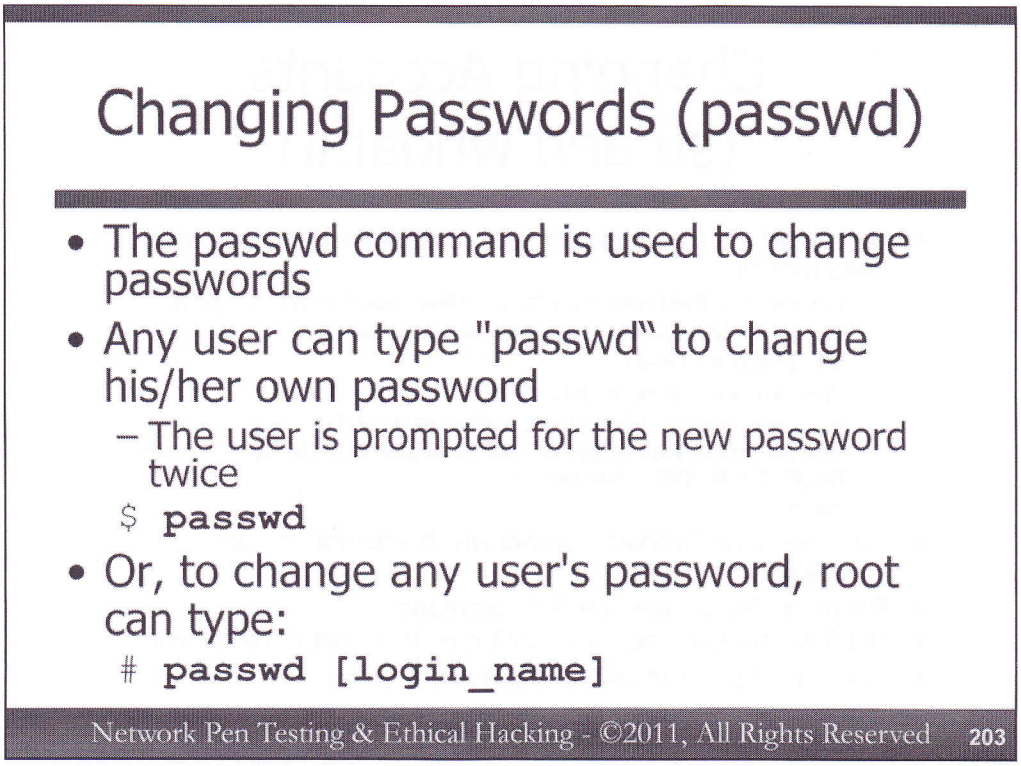

Go ahead and create a non-root account on your system.

Keep an eye on your prompt. If it's a "$", you just aren't root. If it's a "#", you are root.

As root, type the following:

```
# useradd -d /home/fred fred
```

The login account "fred" will be created, with a home directory of /home/fred. The system will automatically assign a non-root userID to the account. The userID is just a number associated with this account for the purposes of assigning permissions. The home directory is where config files and other personal files for this account are stored.




Currently, the fred account we created cannot be used because we haven't yet set a password. (The password isn't blank; the account is just disabled until we enter a password). We need to set a password for the new fred account by typing:

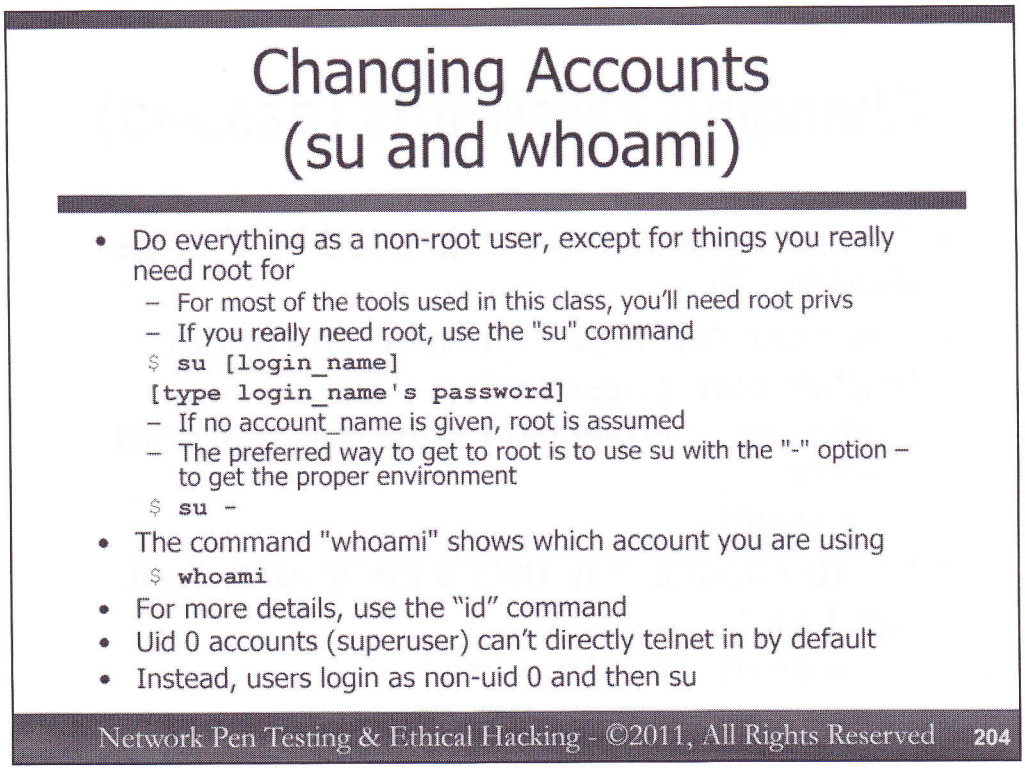
```
# passwd fred
```

[type account password here]

[retype account password to verify]

If fred wanted to change his/her own password, fred would type (from the fred account):

```
$ passwd
```




If you are logged in already, you can switch to a different user by typing the "`su`" command. If you are root, you can directly su to any other account without providing a password (after all, you are root already, so you should be able to change to anyone's account with your god-like privileges!)

If you are not root, you must type in the destination account's password to switch user. The "whoami" command shows who you are currently logged in as.

Type the following:

```
# whoami
# su fred
```

(Notice that the prompt changed!)

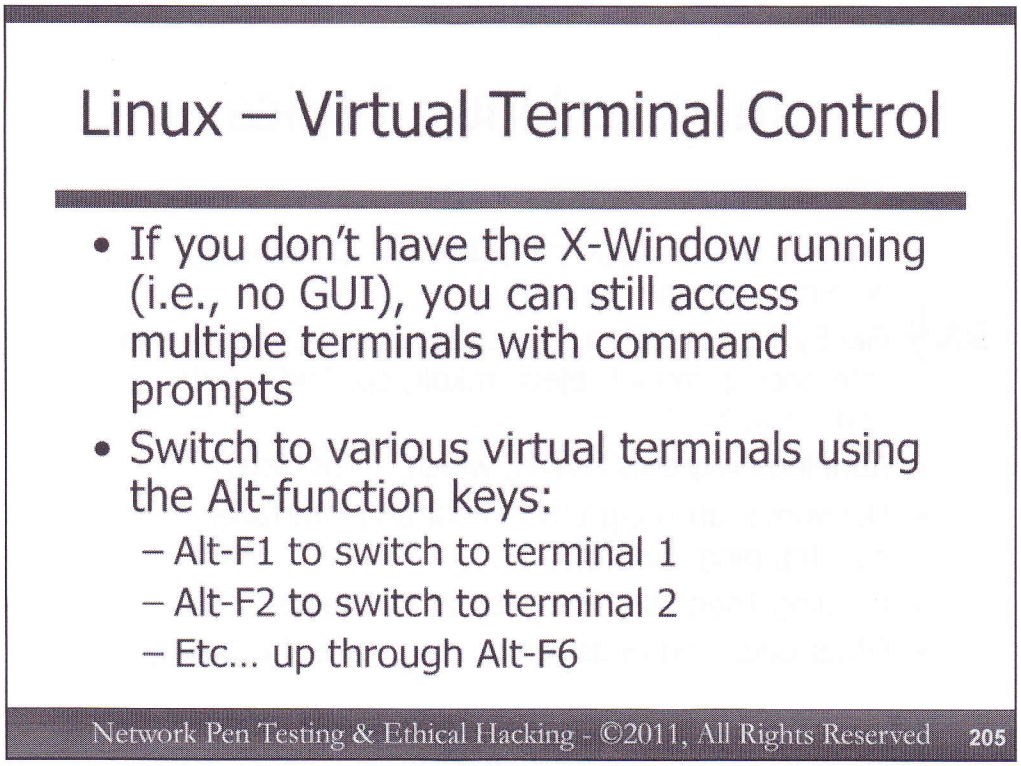
```
$ whoami
$ exit
```

(The exit means that we are leaving the user fred, and returning to root.)

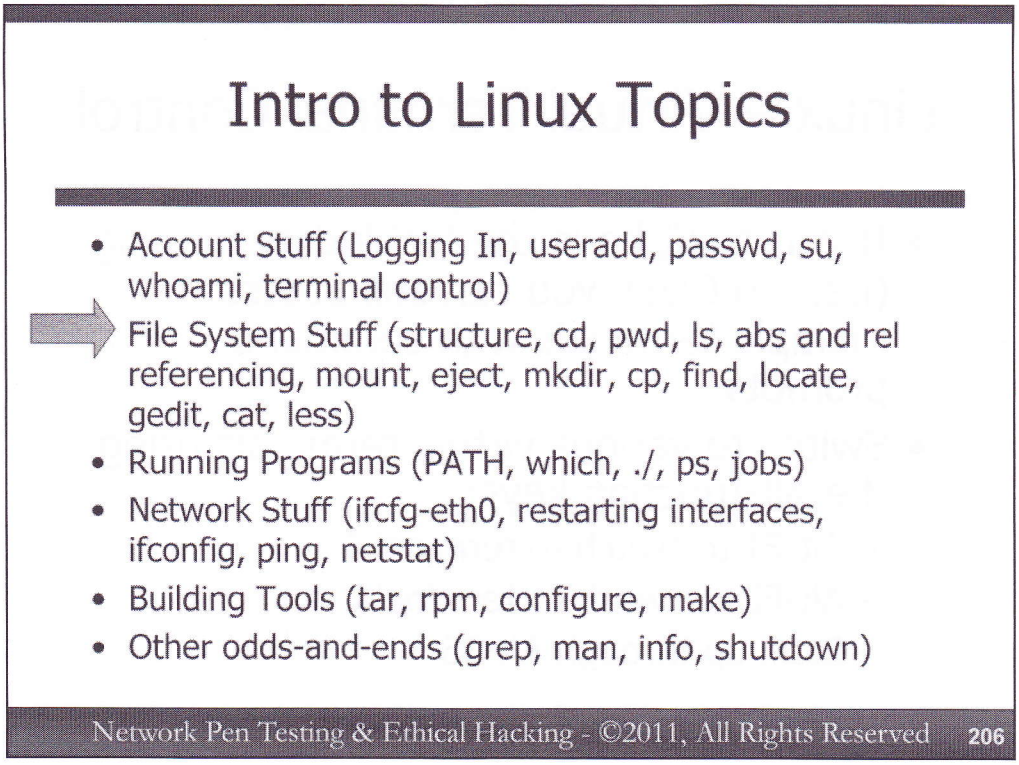
```
# whoami
```

For more details about your current user id and privileges, use the id command:

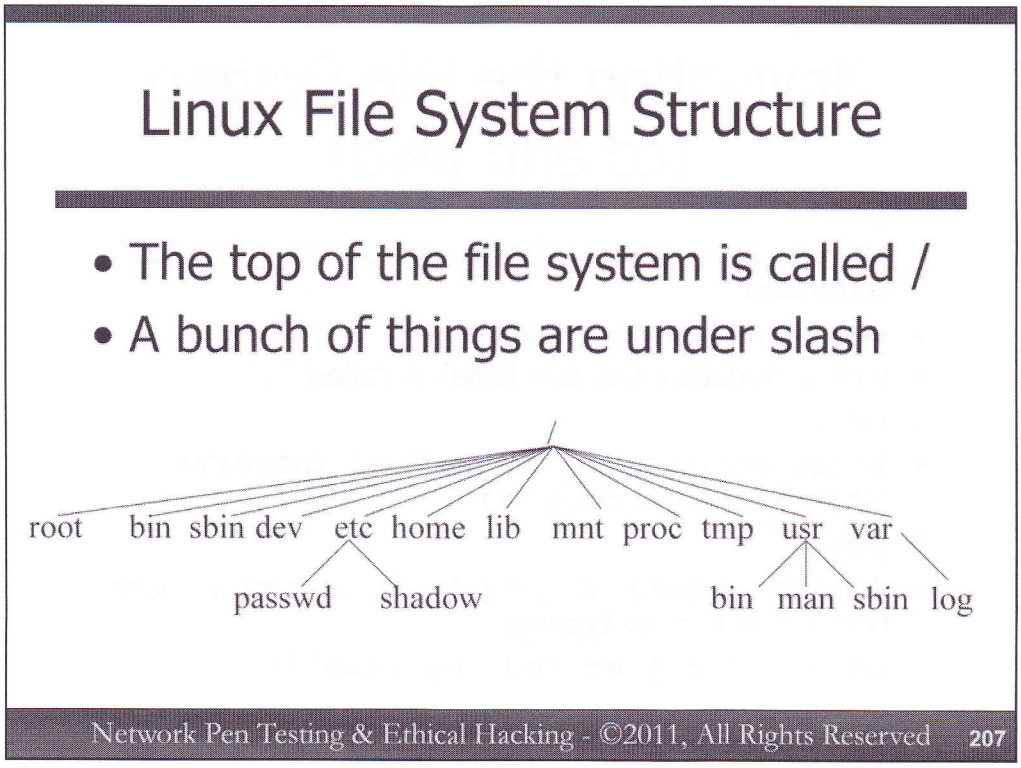
```
# id
```




If you just have a command-line interface, you can still switch between several different terminals. The Alt-Function keys can be used to change to six different virtual terminals. This can be very helpful if you need to simultaneously run different programs and get different streams of output, but don't have the X Window system GUI running.




Here's our pesky outline again. Let's cover File System Stuff next. This is the longest section, simply because so much of Linux is oriented around its file system.




Executable program are stored in /bin and /sbin.

/root is the root login account's home directory. This is hugely important, because if you login directly as root, this will be your initial location in the directory structure. If you login as an individual user other than root, you'll be put in that user's directory, typically somewhere inside of /home.

/dev stores devices (drives, terminals, etc.)

/etc holds configuration items, like the account information (stored in /etc/passwd) and hashed passwords (stored in /etc/shadow).

/home contains user's home directories.

/lib contains common libraries.

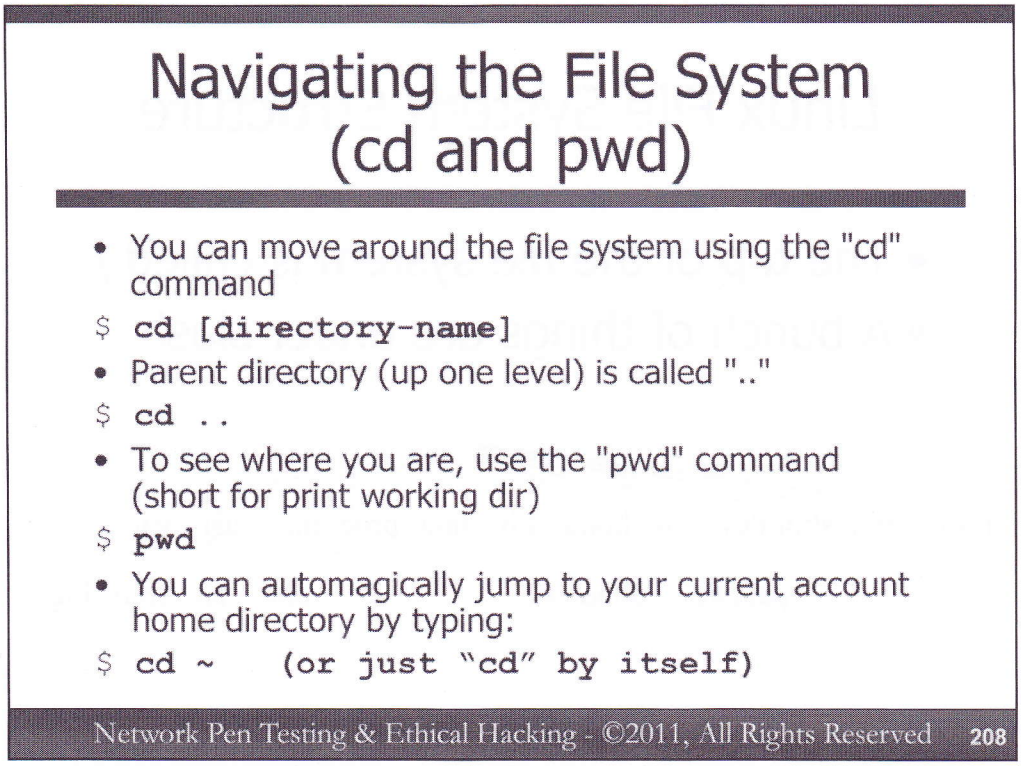
/mnt is where various remote and temporary file systems (CD-ROMs, floppies, etc.) are attached.

/proc is a virtual file system used to store kernel info.

/tmp is for temporary data, and is usually cleared at reboot.

/usr holds user programs and other data.

/var hold many different items, including logs (/var/log/).



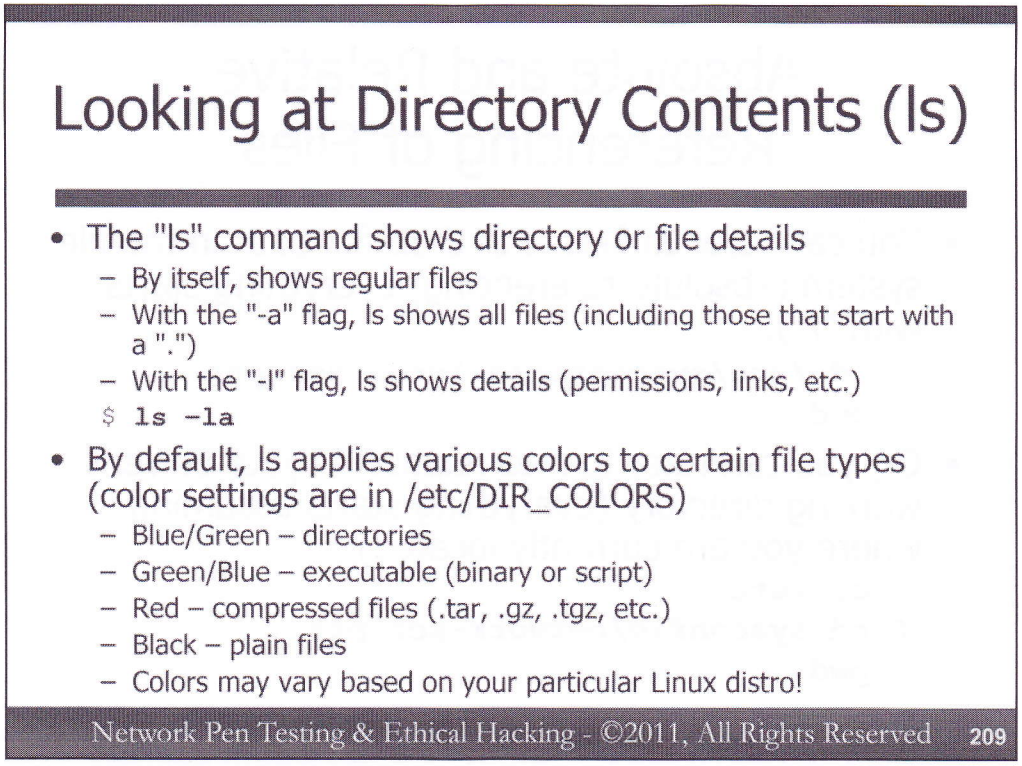

If you are following along, let's change to the top-level directory:

```
$ cd /
$ pwd
```

What do you see?

```
$ cd ~
$ pwd
```

What do you see?




If you are following along, type:

```
$ cd /etc
$ ls
```

What do you see?

```
$ ls -la
```

You now are looking at details associated with your /etc directory. System configuration information is stored here.

Note the different colors of the various files and directories. These colors indicate the type of file or directory, and are set in the /etc/DIR_COLORS. There are dozens of different colors assigned to files of various types, but the most important colors are:

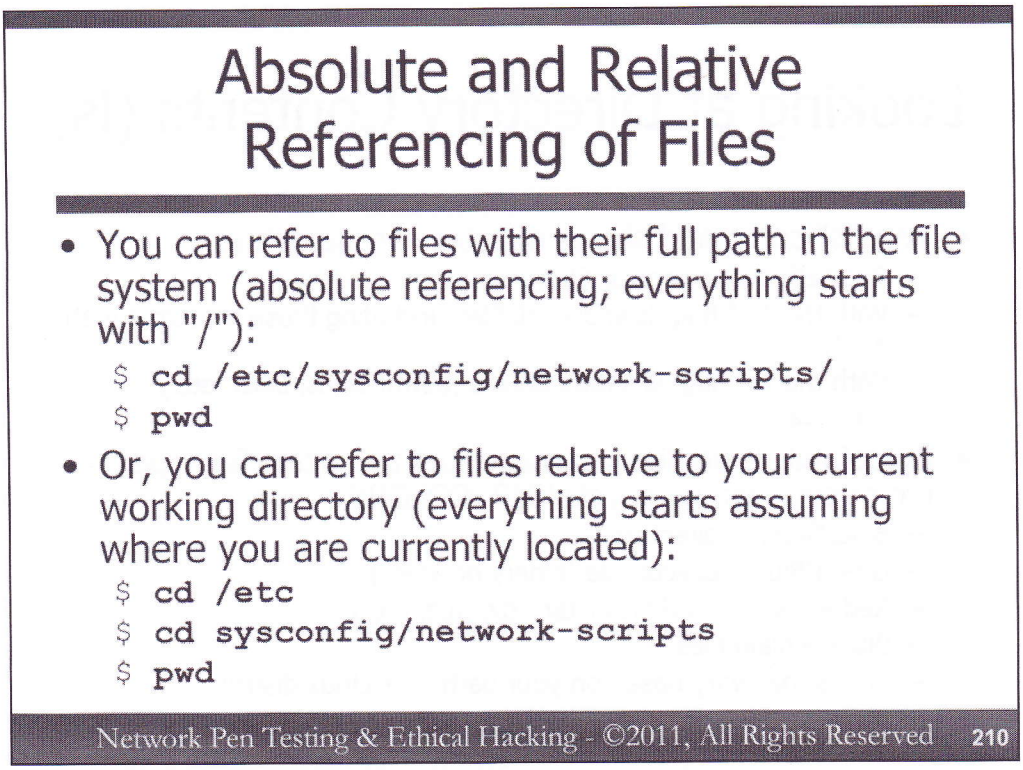
Blue, which indicates a directory. (On some terminals, these are green).

Green, which indicates a file with the executable permission set. It is usually an executable binary or script. (On some terminals, these are blue.)

Red, which indicates a compressed file (such as a .tar, .gz, .tgz, or other compression type).

Black/white, which indicates a regular file (no special type).

Please note that the colors may vary based on your particular Linux distribution!




For any file, you can refer to it using the relative reference (based on your current working directory), or the absolute reference.

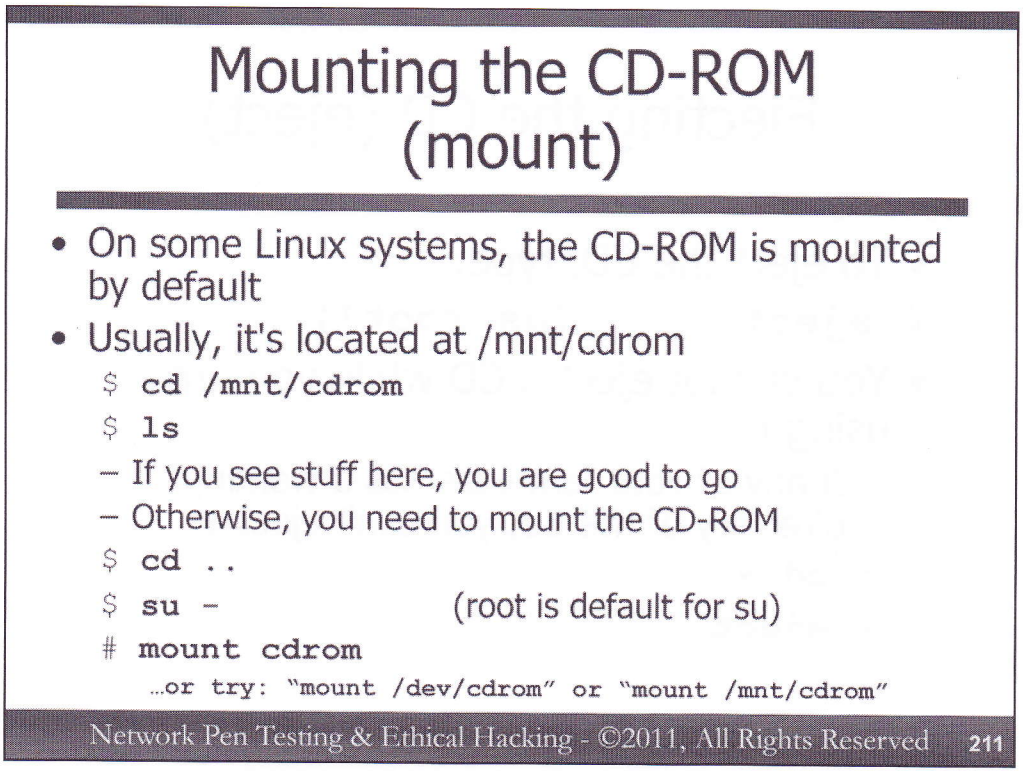
Try the following, using absolute referencing for the file:

```
$ cd /etc/sysconfig/network-scripts
$ pwd
```

Or, you can do it in two steps, using relative references:

```
$ cd /etc
$ pwd
$ cd sysconfig/network-scripts
$ pwd
```

What do you see?




On most Linux installations (particularly RedHat Linux, which was strongly recommended for the course), the CD-ROM is mounted by default. Just pop in the CD, and it will automagically be accessible.

Try this. Pop in a CD (such as the workshop tools CD). Does your file viewer pop up? If so, you are good to go. If not, try changing directories to see if the CD is mounted:

```
$ cd /mnt/cdrom
$ ls
```

If there's stuff located in this directory, you are ready to roll. Otherwise, you have to force the system to mount the CD-ROM. To do this, you have to change out of this directory, become root, and mount the CD-ROM, by typing:

```
$ cd ..
$ su -
```

[enter root password]

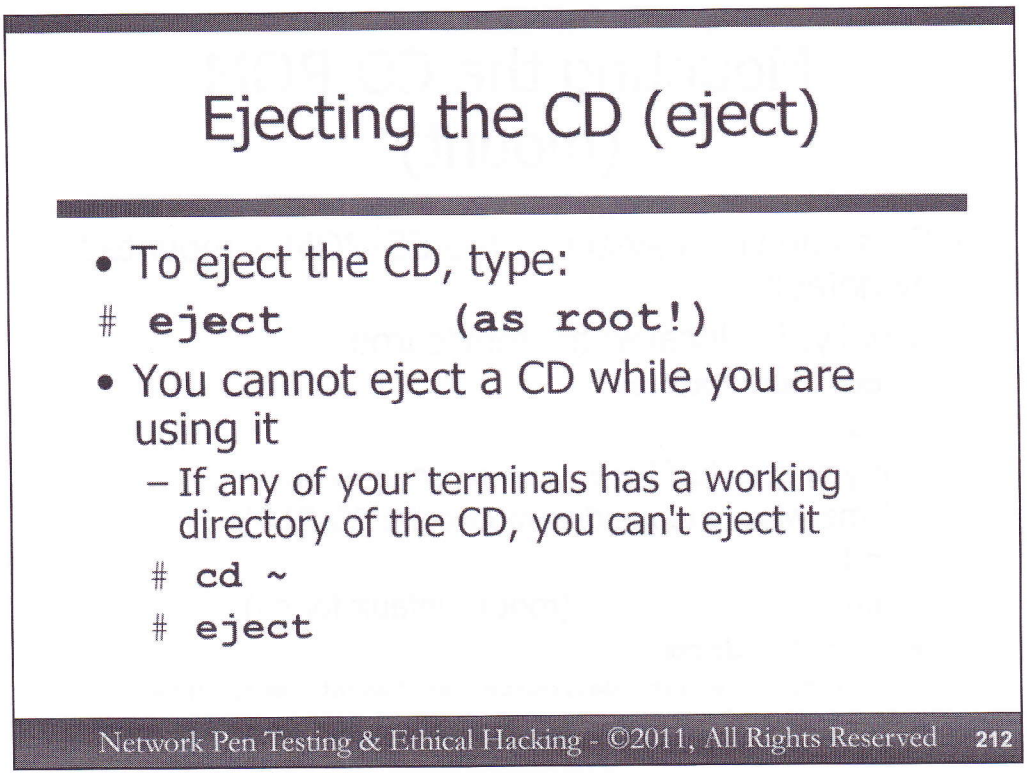
```
# mount cdrom
```

For some Linux variations, instead of "mount cdrom", you need to type:

```
# mount /dev/cdrom
```

For other Linux variations, instead of "mount cdrom", you need to type:

```
# mount /mnt/cdrom
```




To pop out the CD, you need to make sure all terminals are done using the CD. Change directories out of the /mnt/cdrom directory first, by typing:

```
# cd ~
```

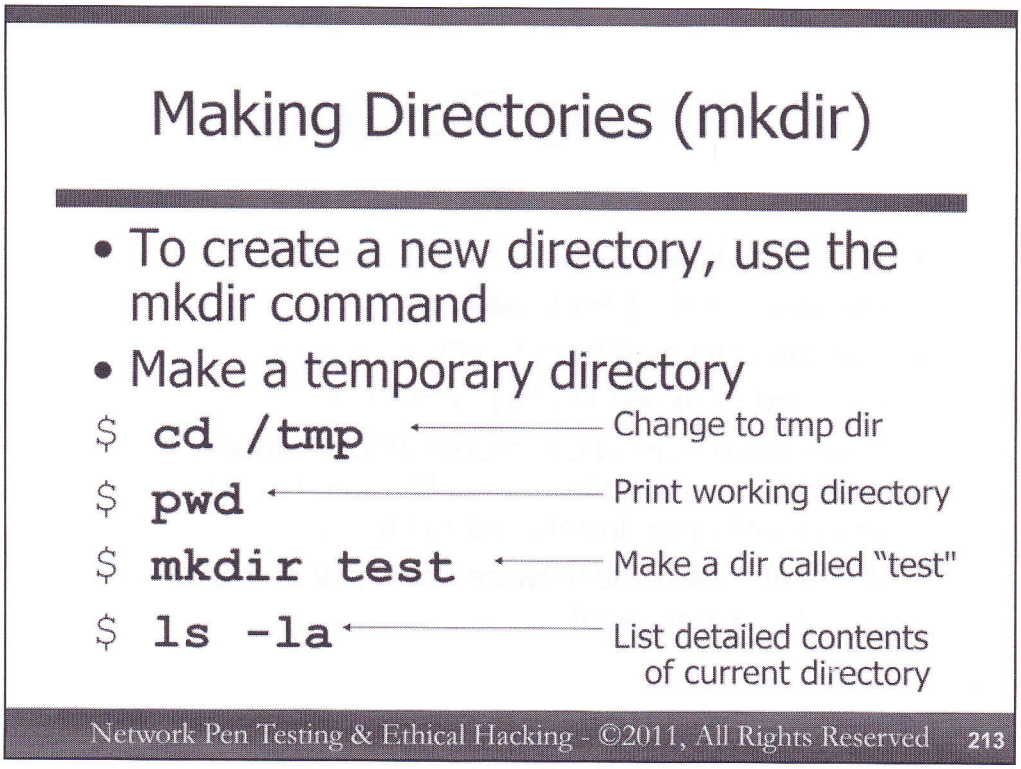
(This takes us back to the user's home directory).

To eject the CD, you must be root, so su there first.

```
# eject
```

The CD should pop out.

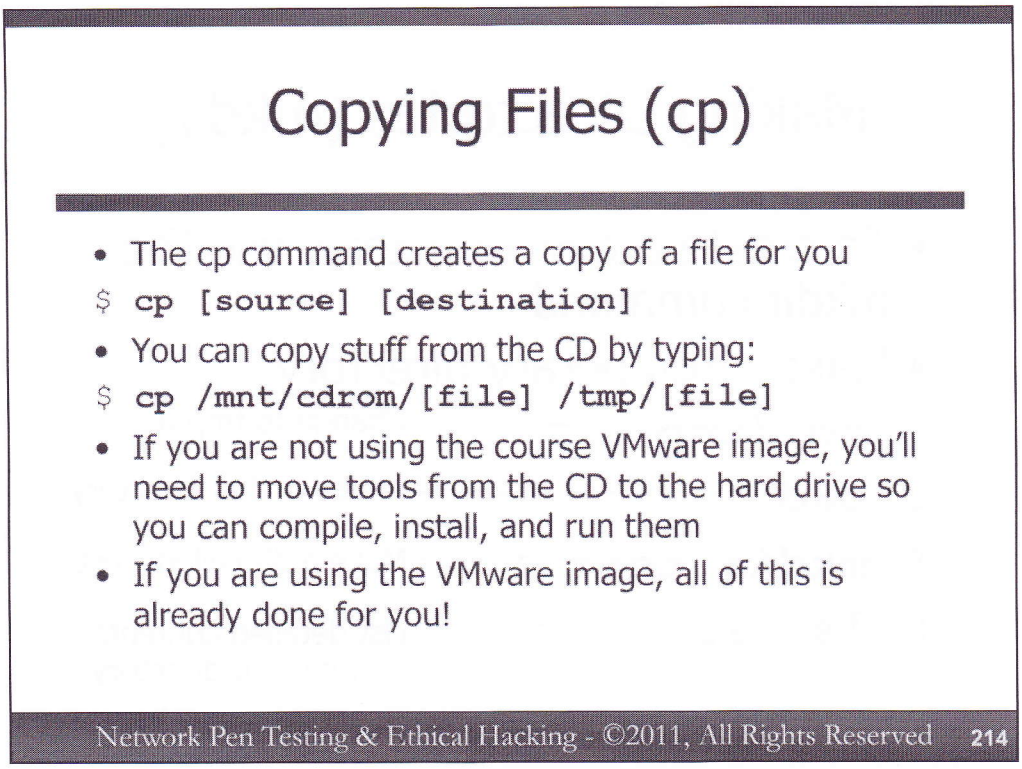
If you get a message saying that the CD is busy, one of your terminals (or some other application), currently has a working directory on the CD somewhere. You need to find it and change directories out of the CD before you can eject.




To create a directory, use the mkdir command. Let's create a test directory in our /tmp directory.

```
$ cd /tmp
$ pwd
$ mkdir test
$ ls -la
```

What do you see?

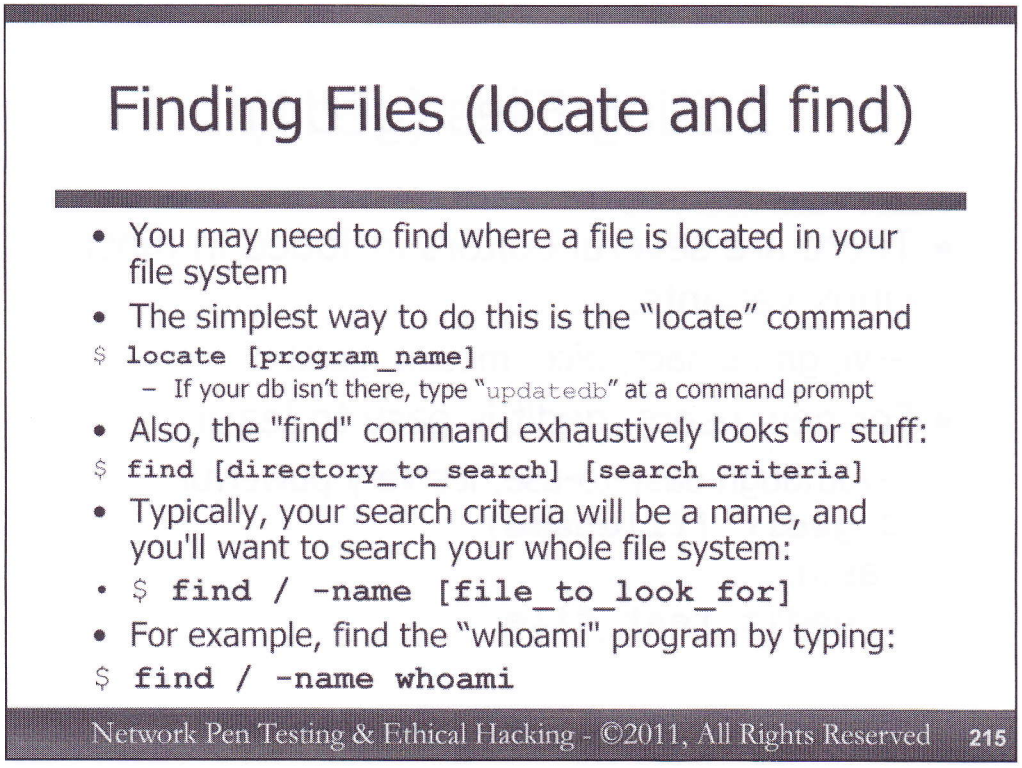



To copy a file from your CDROM, type:

```
$ cp /mnt/cdrom/[file] /tmp/[file]
```

The [file] should be the name of some valid file on the source drive.

To copy a bunch of files, you can use * to indicate all files in a given directory.




The "locate" command is a very efficient way to determine where files are located on the system. It consults a local database installed and updated by the system administrator for files that are frequently sought. It runs quickly and doesn't consume a lot of resources. However, it cannot locate items that are not loaded into its database.

To try locate, type:

```
$ locate whoami
```

If your system complains that there isn't a locate database or that it's out of date, you can manually update the database by typing the command:

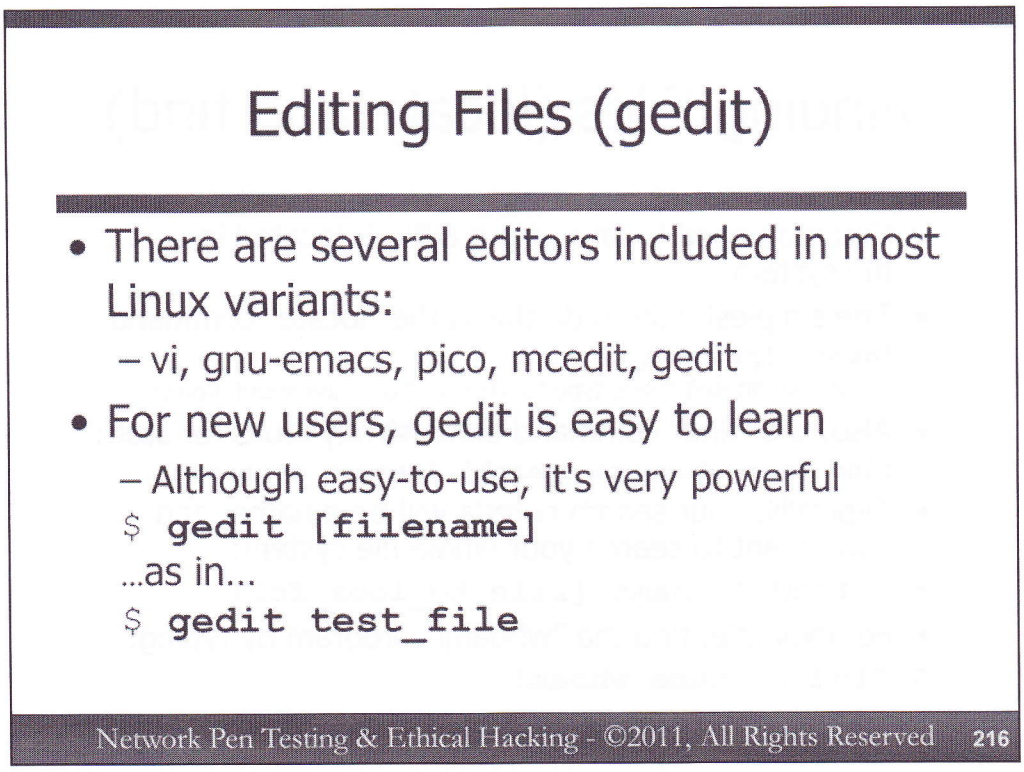
```
# updatedb
```

To do a comprehensive search of the directory, you can use the "find" command. This command consumes a lot of resources. Several finds running simultaneously will slow a Linux system to a crawl. Still, "find" is the best way to find something if locate doesn't work.

Let's try to find a file on the file system. Type the following:

```
$ find / -name whoami
```

What do you see?



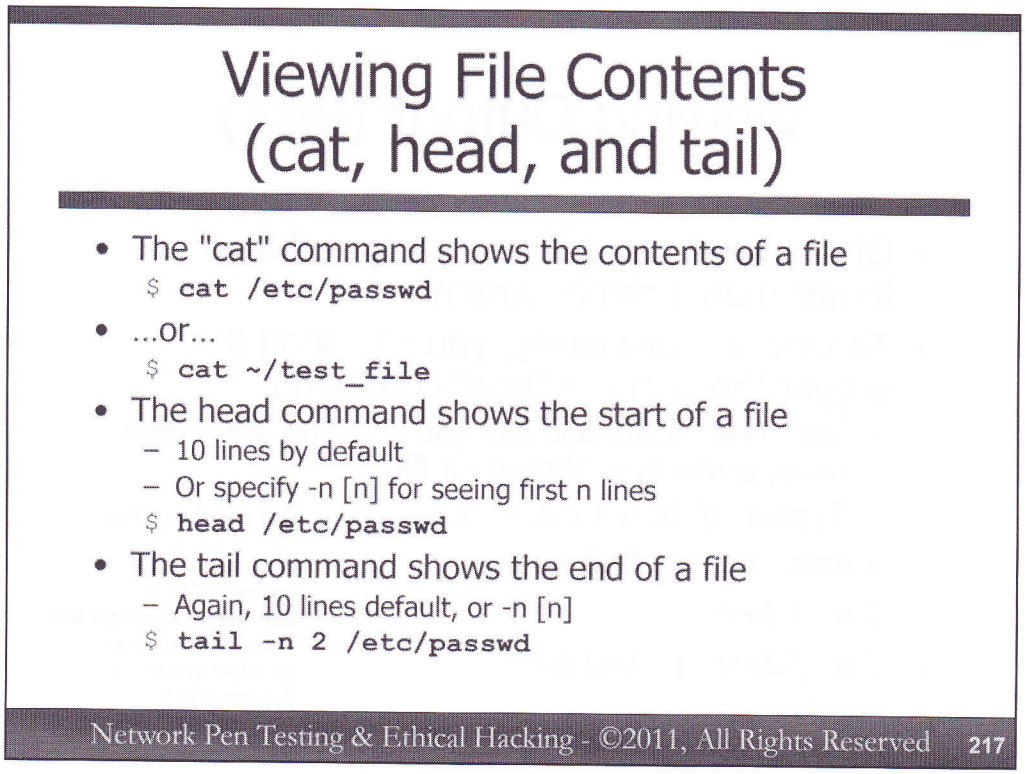

You may need to edit a file at some point. You can use any editor you are comfortable with. If you are new to Linux, you should consider using gedit, one of the easiest editing tools commonly installed in Linux. If you have a GUI, you can use gedit.

Let's create and edit a file:

```
$ cd ~                    (change to the home directory)
$ gedit test_file         (let's edit and create a file named "test_file")
```

Now, edit your file. Type in a bunch of junk. Use the function keys to save it.

I told you gedit was easy!




So, you just edited a file. How can you see its contents? You can use the cat command:

```
$ cat ~/test_file
```

Also, you can look at other files:

```
$ cat /etc/passwd
```

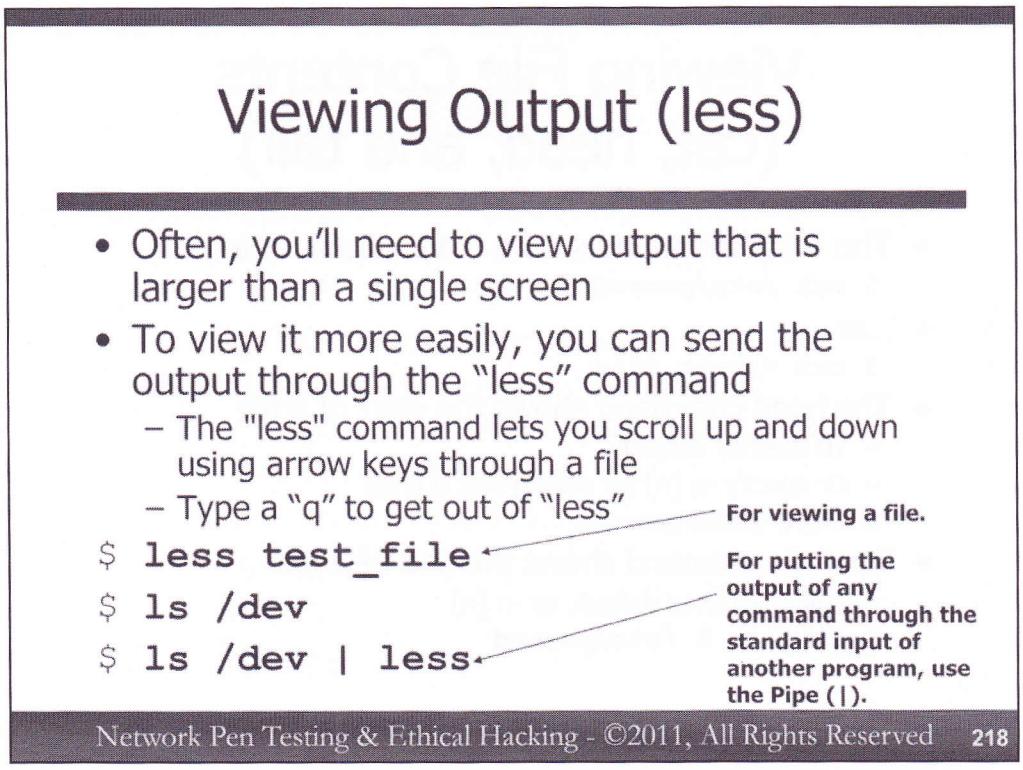
This shows the contents of the password file! (Note that on most Linux installations, the passwords are stored in another file, /etc/shadow). Typically, in most modern Unix installations, /etc/passwd just contains account information.

Alternatively, we can view portions of files using the head or tail commands. The head command shows the first 10 lines of a file by default. By specifying "head -n [n] [filename]", we can view just the first n lines. Similarly, the tail command shows the last 10 lines of a file by default, or we can use the "-n [n]" syntax to view a different number of trailing lines. Consider the following commands:

```
$ head /etc/passwd
```

```
$ head -n 1 /etc/passwd
```

```
$ tail -n 2 /etc/passwd
```




Besides "cat", there are other commands you can use to look at files. The "less" command is one of the best to use. Try typing:

```
$ less test_file
```

You should see the contents of the file.

In addition to looking at files, the less command can also be used to help look at lengthy output from a command. Try typing:

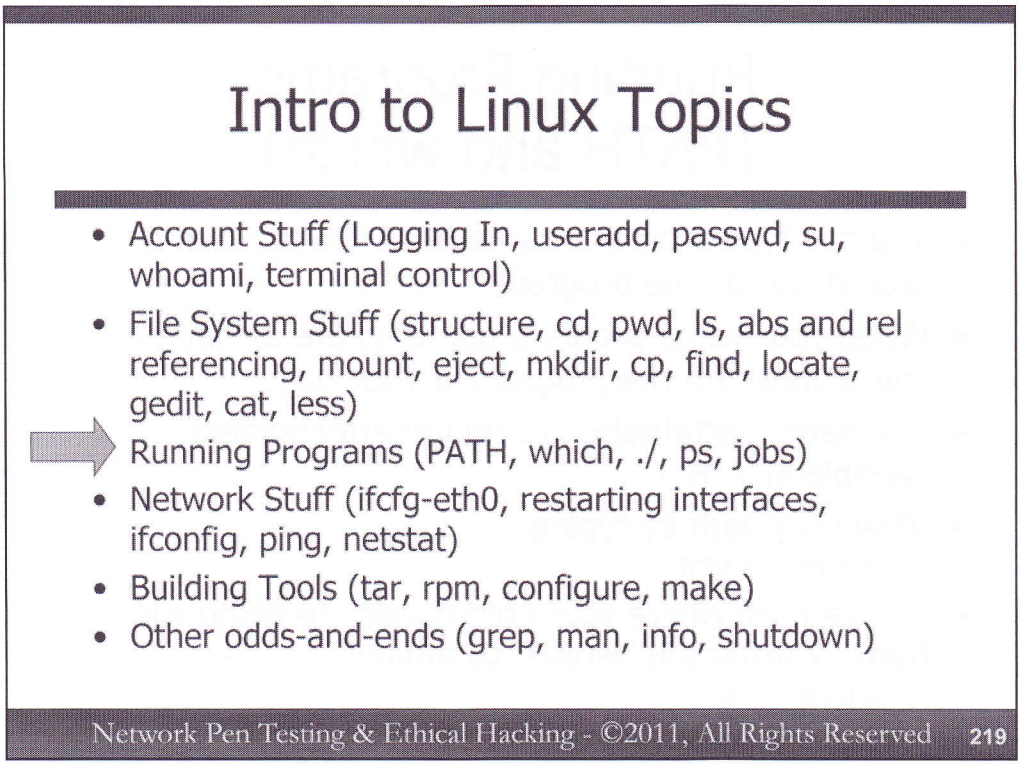
```
$ ls /dev
```

This shows you all of the devices (virtual and otherwise) on your system. It's a long, unwieldy list. The less tool lets you interact with this output in a better way.

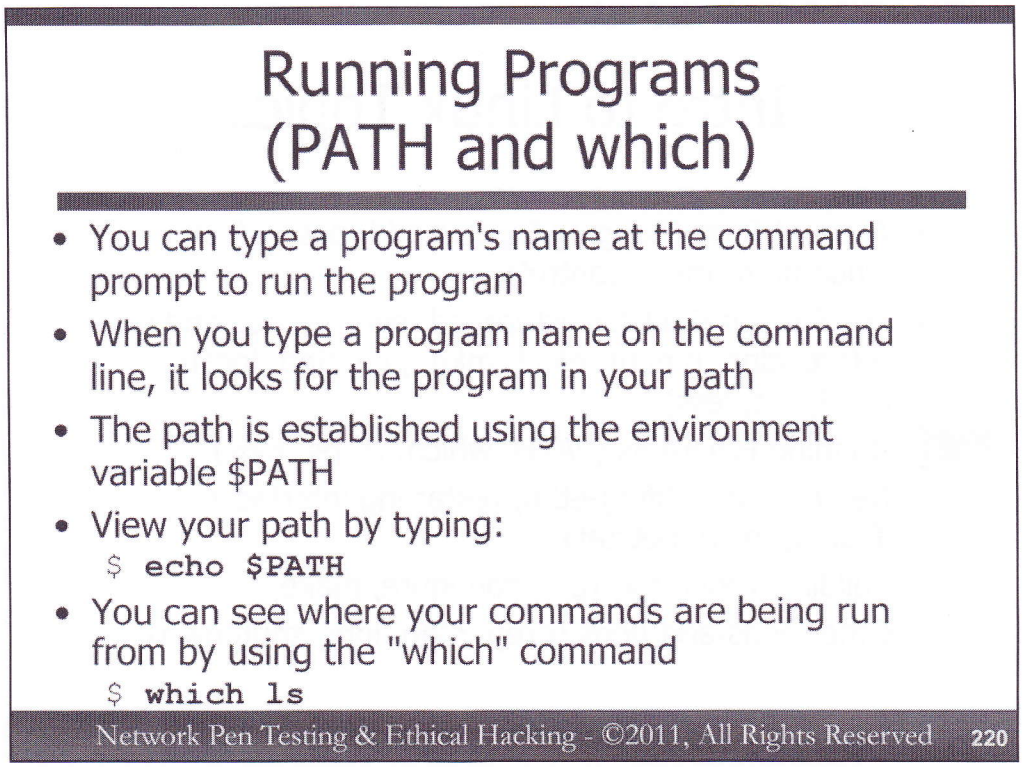
Type:

```
$ ls /dev | less
```

By piping the output of ls through less, you can now use the cursor keys to scroll up and down through the output. The space key jumps forward one page. Use the "q" key to quit. The pipe takes the output of one program and feeds it into the standard input of another program.




You guessed it… another outline slide. We will now discuss running programs. This section is pretty important (not that the other ones aren't important…) People frequently mess up on this stuff and get confused, because Linux works very differently from Windows in running programs.




When you type a command at the prompt, the system looks in your PATH to find the right program to run.

Look at your path by typing:

```
$ echo $PATH
```

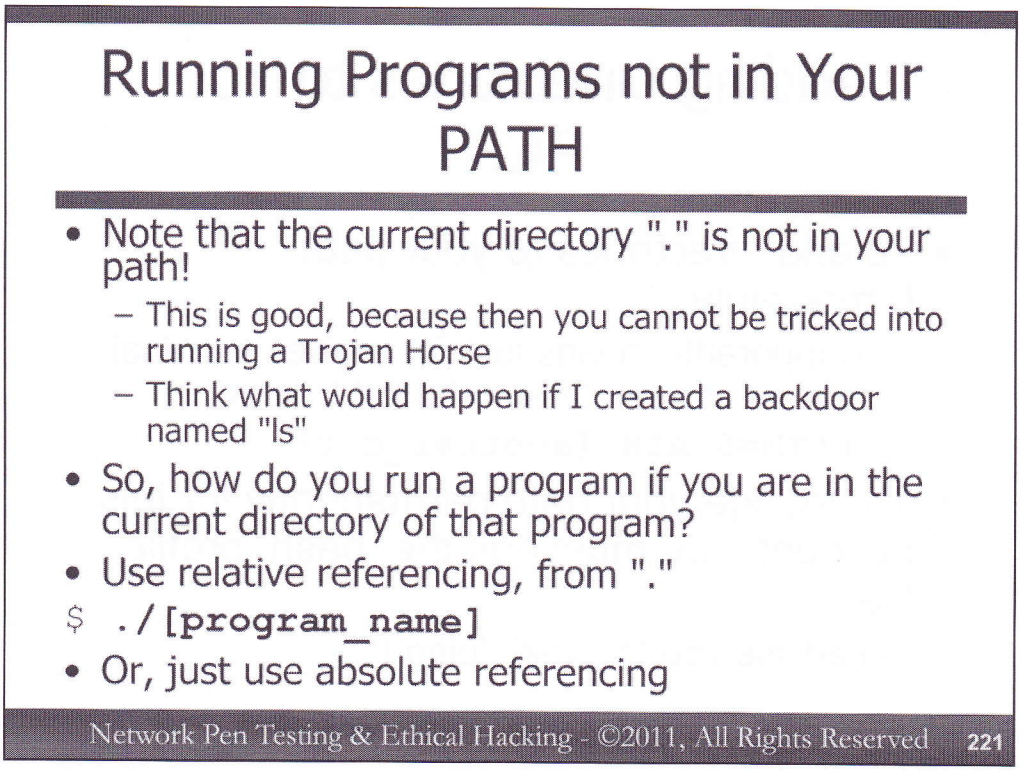
The echo command means "type the following." The $ before path means, "What follows isn't a string of characters; it's a variable." The variable we want to type is our PATH.

The result is a list of directories where the system searches for programs based on what we type at the command line. These directories are separated by a ":". If you type a program name at a command prompt, and the program isn't in your PATH, the system will tell you that it cannot find the program. You have to either refer to it absolutely or relatively, or add its directory to your PATH.

If you want to see where in your PATH a command has been found, you can use the "which" command. Try typing:

```
$ which ls
```

That's where your "ls" program really is!




**This is an important point that confuses people, because Unix functions differently from Windows on this issue.**

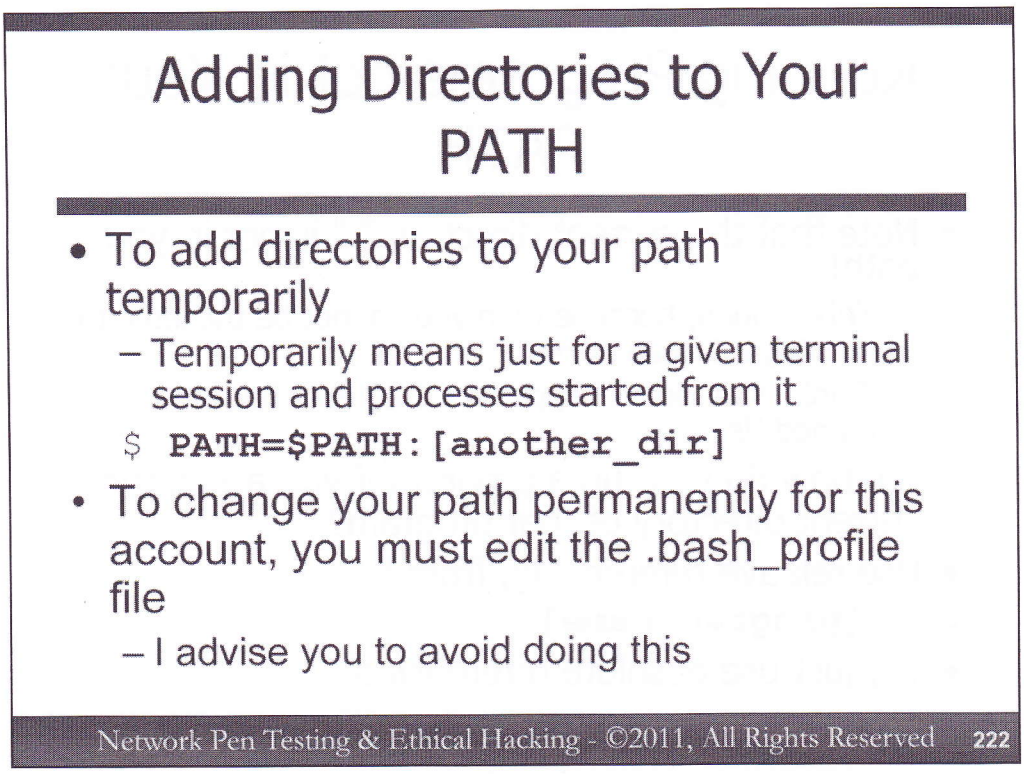
For security reasons, your current working directory (the one shown by "pwd"), also referred to as ".", is not in your PATH. That's a very good thing! If "." was in your path, an evil attacker could name an evil Trojan Horse program "ls", and put it in your home directory. When you ran ls to look at your home directory's contents, you'd run the evil Trojan Horse! For this reason, "." isn't in the path by default, and shouldn't be put in your path.

This also means that if you change directories to a place where a program file is located, you cannot just type the program's name to run it. Instead, to run the program, you have to type "./[program_name]" to run it.

If the system ever complains that it cannot find a file, but you can see the file in the current working directory using "ls", you likely just need to start the program by typing:

```
$ ./[program_name]
```

On Windows machines, the current working directory is in your path. Therefore, if you change to a directory with an executable and type the executable's name on Windows, the program runs. Yes, it's convenient… However, it's a security hole!




You can add your /home/tools directory to your PATH temporarily. Type the following:

```
$ echo $PATH
```

Look at your path. Then, type:

```
$ PATH=$PATH:/home/tools
```

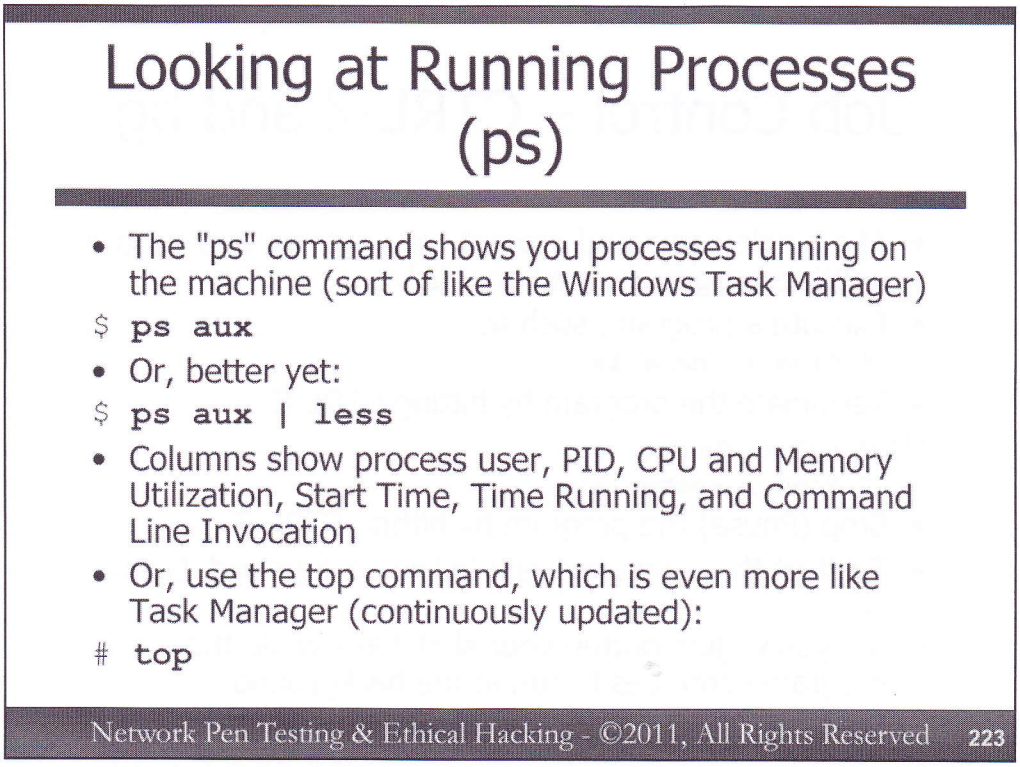
Now, type:

```
$ echo $PATH
```

See… /home/tools is added on the end.

This change applies only to this terminal and any processes started from this terminal. When you logout, this change goes away, and your path has its original settings.

To permanently change your path, you must edit the ~/.bash_profile file. I advise you to avoid editing this file if you are new to Linux. The default path setting is pretty good for most purposes.




Sometimes, you need to see what processes are running. The ps command shows you a bunch of info about all running processes. Try typing:

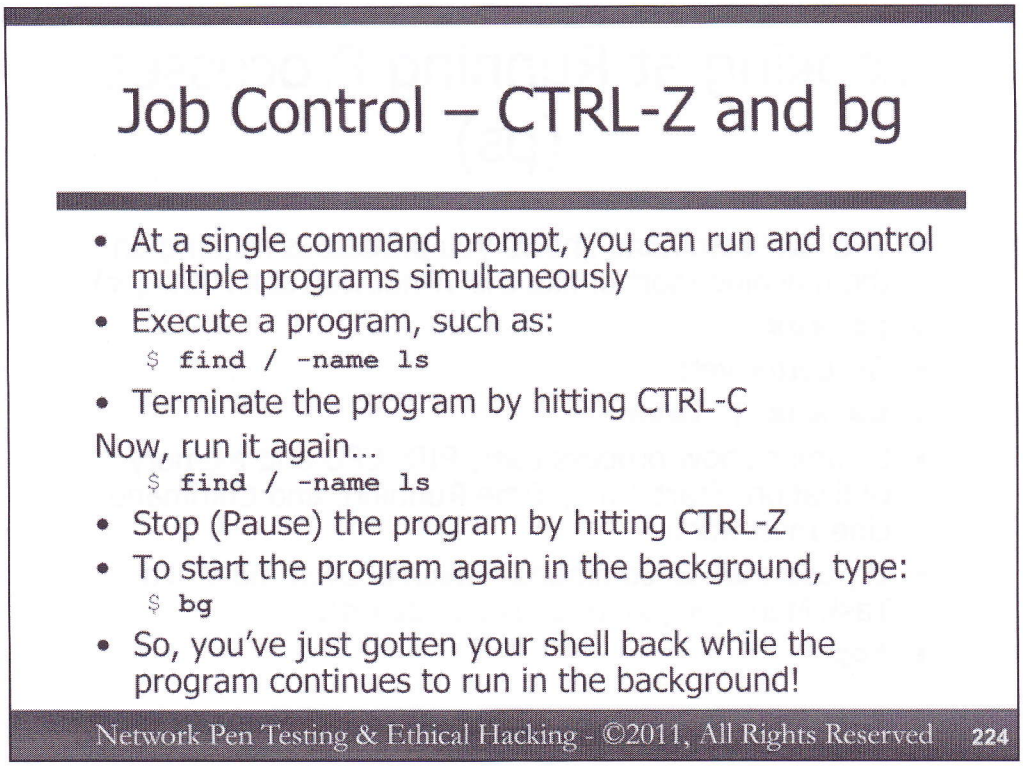
```
$ ps aux
```

You get an exhaustive (and exhausting) list of all running processes.

Let's use our little "less" trick to make this output more readable:

```
$ ps aux | less
```

Now, you can scroll up or down and get a better feeling for what's running on your system.




You can temporarily pause programs with CTRL-Z, and get your command prompt back. This is quite useful, as you can get your command prompt back to run more programs if you'd like.

Also, you can restart the paused program running in the background with the "bg" command. The "fg" command starts it running in the foreground, as you might expect.

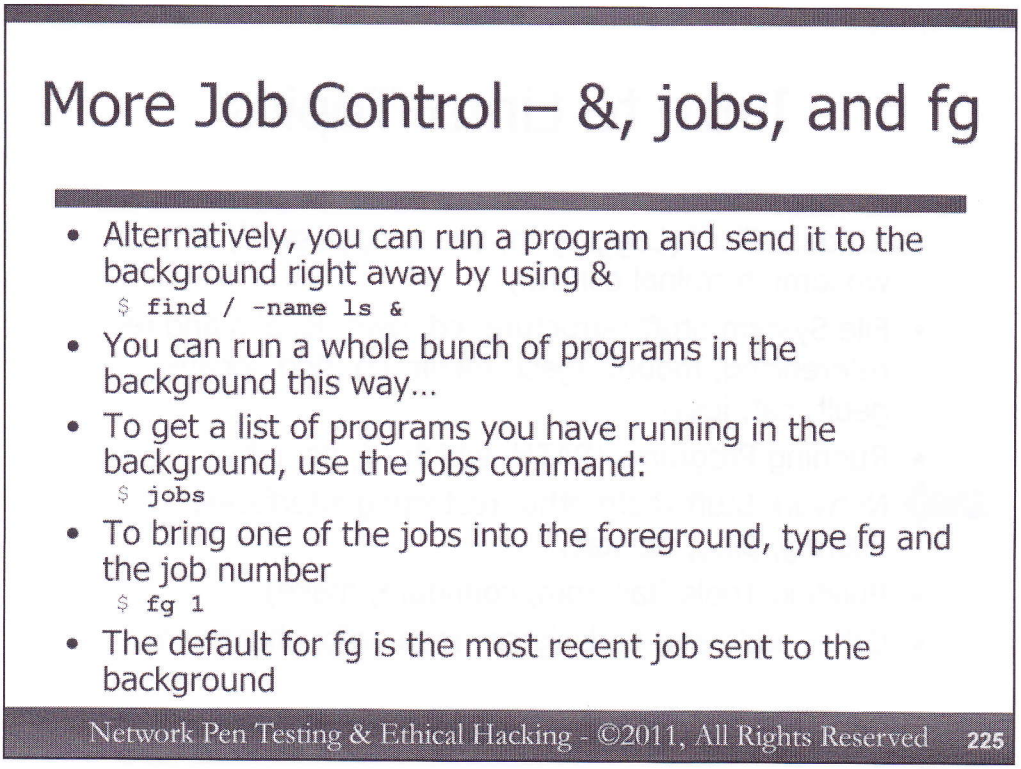
Let's try it. Type:

```
$ find / -name ls
```

Before it finishes running, hit `CTRL-Z`.

Now, restart the program in the background by typing:

```
$ bg
```




The jobs command gives you a list of all programs you have kicked off that are running in the background. The fg command can also be used to restart a specific paused program in the foreground, but giving the job number after the "fg" command.

If the find command from the previous slide has finished, type the same command again, but this time run it in the background using the & after the command invocation.

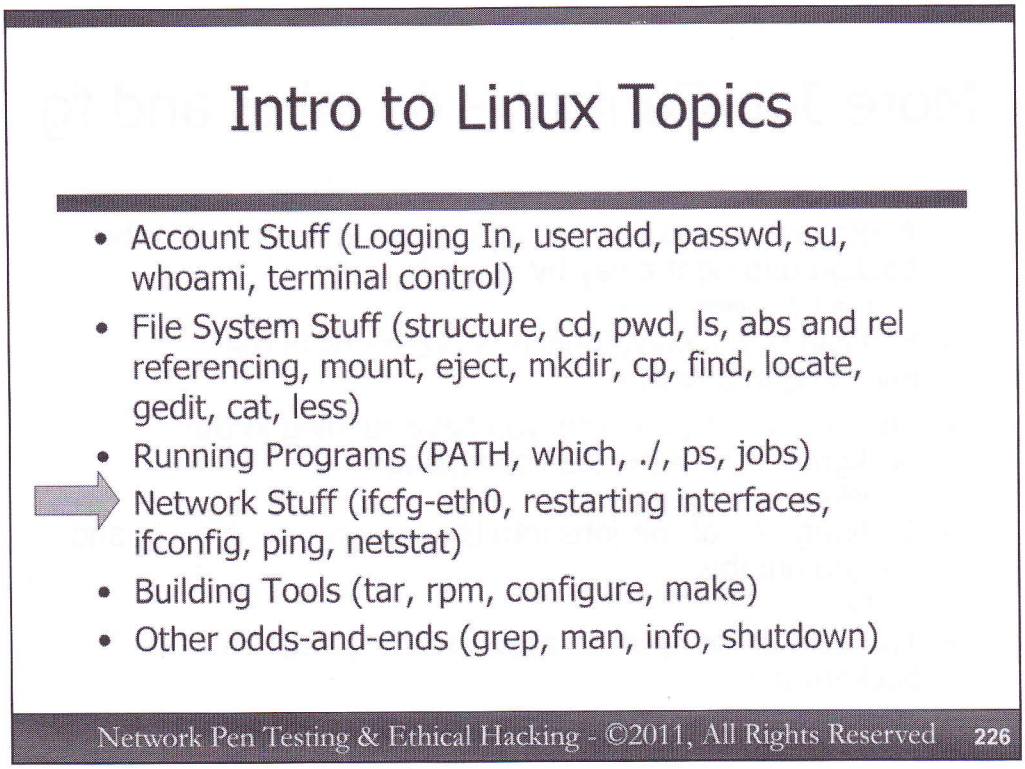
```
$ find / -name ls &
```

As it runs in the background, type the jobs command:

```
$ jobs
```

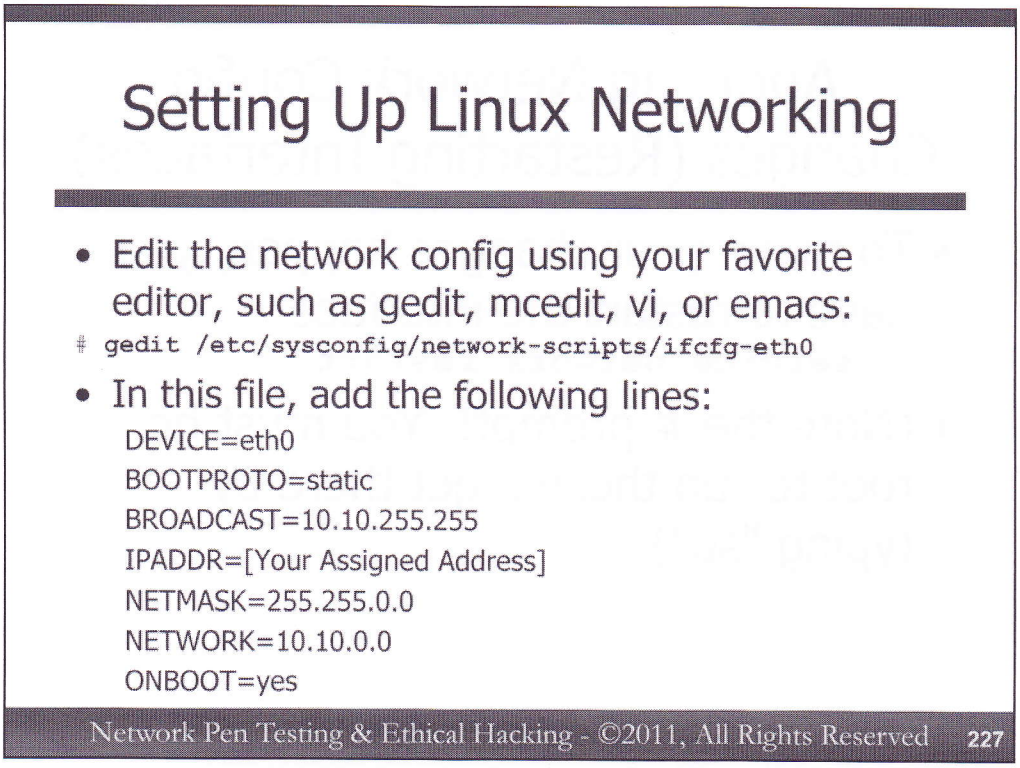
Look at the job running. You can move it to the foreground by typing:

```
$ fg 1
```




Another outline slide. Gee, these outlines are fun.

Now, we will cover getting and staying networked in Linux.

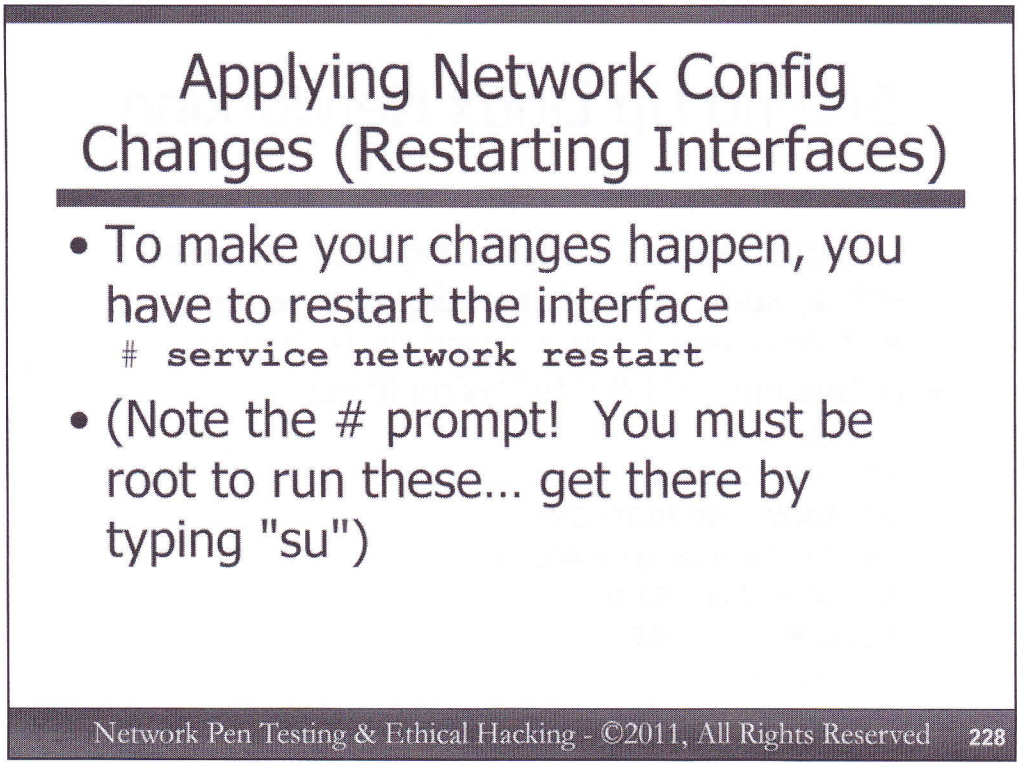



There are two ways to set your network interface options in Linux. You could use a GUI included with your Linux. However, different versions of Linux have different GUIs, and they change the GUI every time they release a new version. Because of this, I just update the file.

You can use gedit to edit /etc/sysconfig/network-scripts/ifcfg-eth0. By putting in the appropriate information, you can configure your interface.

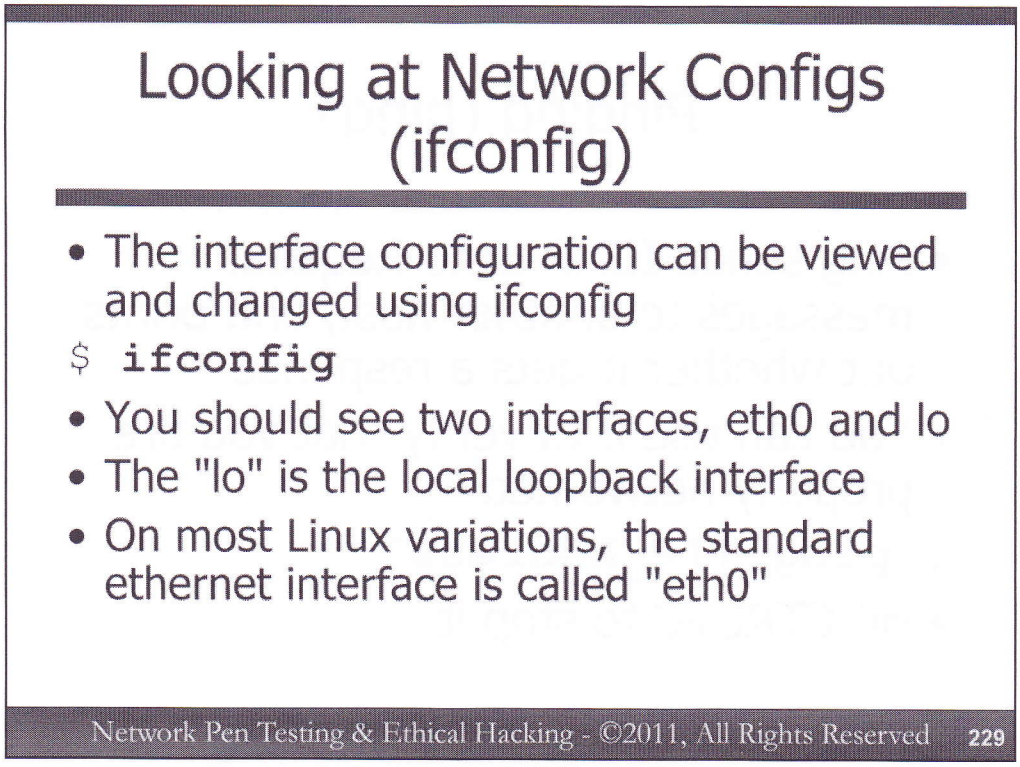
Enter the following information into this file:

```
DEVICE=eth0
BOOTPROTO=static
BROADCAST=10.10.255.255
IPADDR=[Your Assigned Address]
NETMASK=255.255.0.0
NETWORK=10.10.0.0
ONBOOT=yes
```




If you change the ifcfg-eth0 file, your changes will not be applied to the interface immediately. Instead, you need to restart your interface. Type (as root):

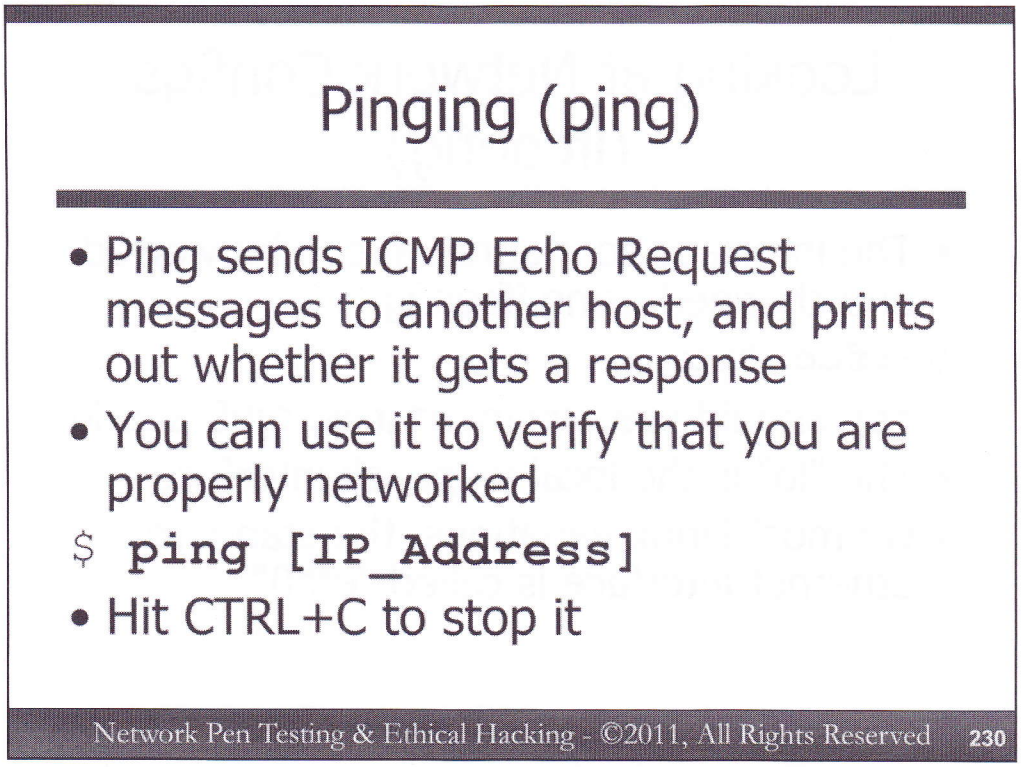
```
# service network restart
```



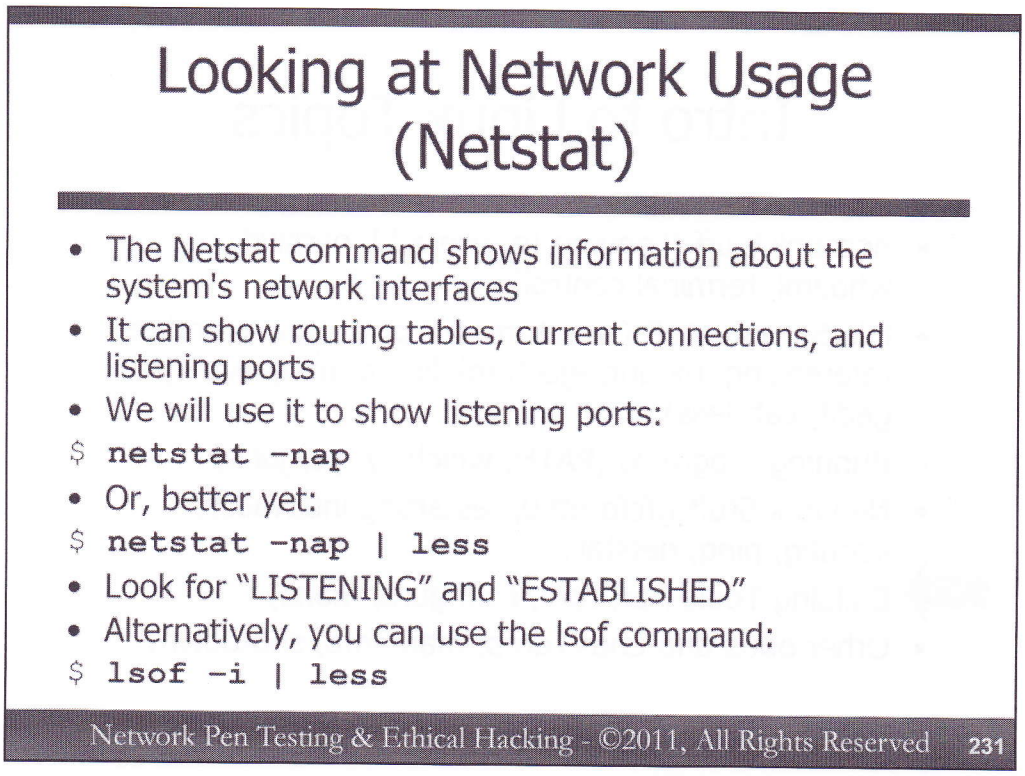

Let's see if our interface changes were applied to the system. To look at your interface configuration, type:

```
$ ifconfig
```

You will see your IP address, netmask, MAC address, and various other nifty items. If you have one ethernet card, you will see two interfaces… the local loopback interface with the address 127.0.0.1 and your ethernet interface, called "eth0".




To verify that you are properly networked, you can ping another machine. The ping command is similar, but not identical, to the Windows ping program. One of the biggest differences is that a Linux ping will keep on sending pings until you hit <CTRL+C> to stop it. By default, the Windows ping sends out 4 ICMP Echo Request packets and then stops. Linux just keeps going until you stop it.




Let's look at what's using your various TCP and UDP ports. Type:

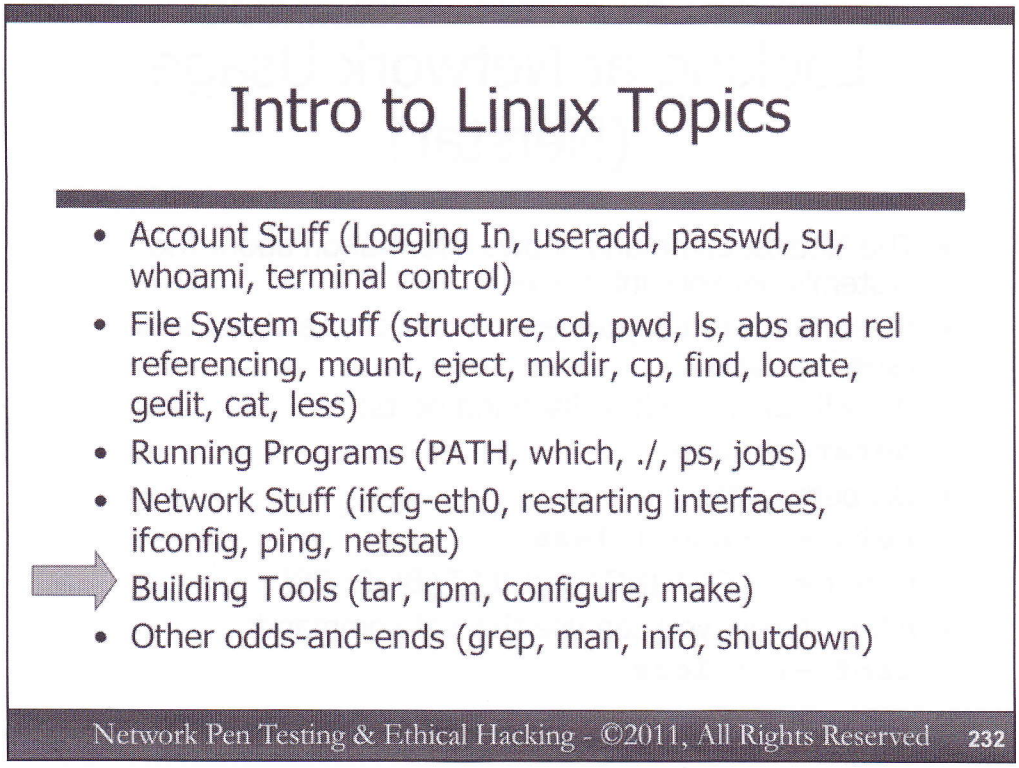
```
$ netstat -nap
```

There's a lot of stuff there. It can be a bit difficult to read as it scrolls by, so let's try:

```
$ netstat -nap | less
```

Now, you can scroll up and down through the output. We'll discuss how to do better searches through this later.

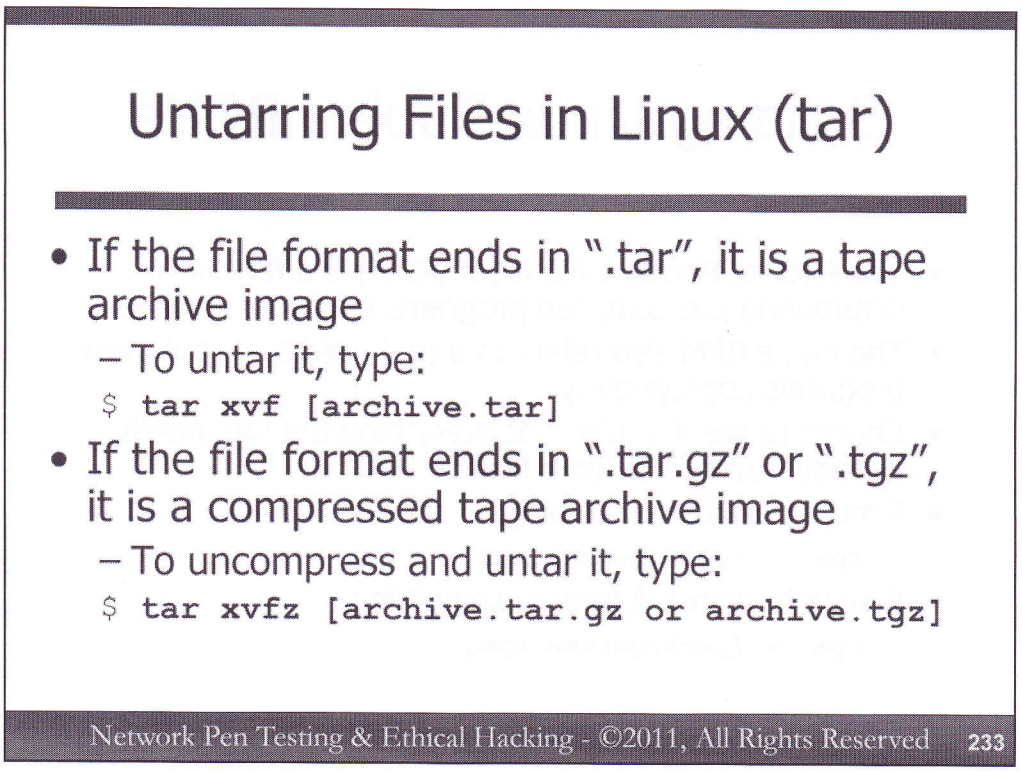
Note that various TCP and UDP ports are shown as "LISTENING". These are waiting for a connection. Others may indicate that they are "ESTABLISHED". These have existing connections.




You know, to use Linux in the workshop, you'll have to run tools. To run them, in many cases you'll need to build and install them. As we see from the outline slide, we'll cover building and installing tools next.

Some programs are installed by using tar files. Others are rpms. Still others use "configure" and "make". How do you know which tools use which format? Most of the tools include a README file. Look at the README file (using cat, less, or gedit) for instructions on installing the tool. The course notes for the main class also include directions for compiling and installing.

By the way, we are having you compile and install the tools so you can get experience with doing these tasks. In the wild, you may need to compile and install new versions of these and other tools, so we want to get you ready.




Some tools are stored at tape archives, abbreviated "tar". This doesn't mean they were on physical tapes… the lingo just lingers from the olden days. While tapes may or may not be used, tar files are used all the time. Think of them as being like ZIP archives in Windows. You take a bunch of files and glom them together in a tar file.

To open a tar file, you use the xvf parameters. x means "Extract." v means "Be verbose – give me lots of output to let me know what's going on." f means "Get this from a file."

If the tar file has been compressed using a tool called "gzip", its name will end with a suffix of ".tar.gz" or simply ".tgz". To open these up, you need to use the tar command with the xvf and z flags. The z flag means "Unzip this before you open the archive."

When the archive opens, all files and directories associated with it will be automatically created in the current working directory and below.

We won't demo this during the Intro to Linux mini-workshop. You'll get a chance to use tar files during the main class.

# Building Linux Tools - RPMs

- The RedHat Package Manager (RPM) is a format for distributing pre-compiled programs and their libraries
- The name RPM also refers to a tool used to install these programs appropriately
- Checks to see if you've got everything the tool needs already installed on your box
- Syntax to install something:

```
$ rpm -i [packagename.rpm]
```

- Syntax to uninstall (erase) something:

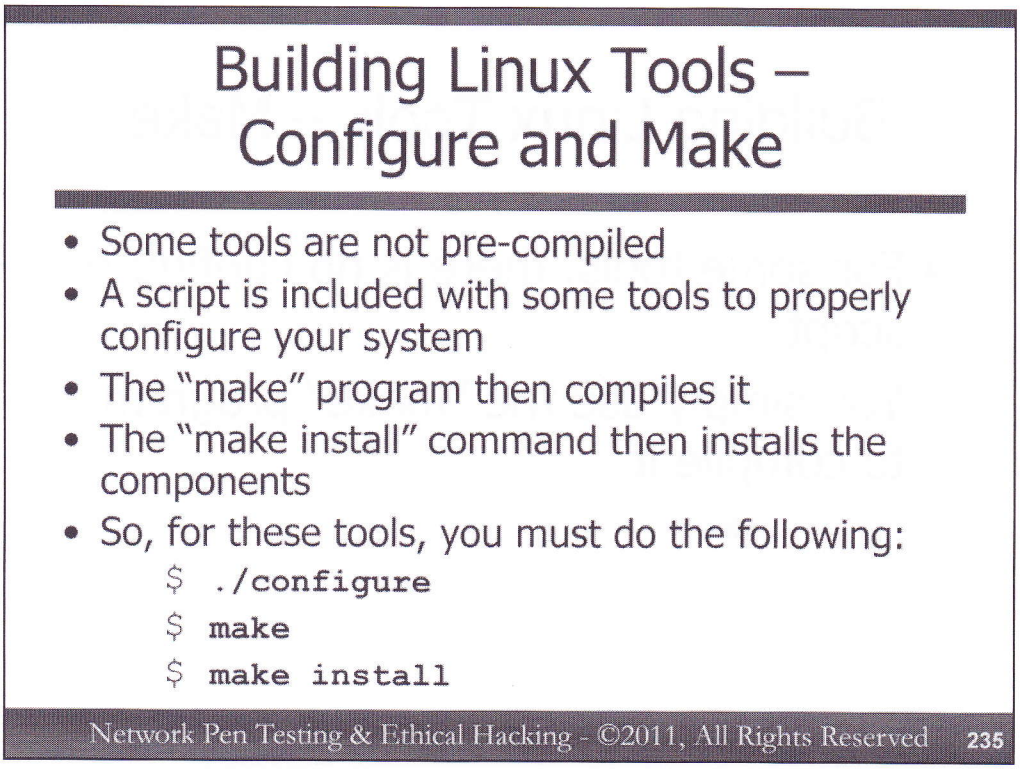
```
$ rpm -e [packagename.rpm]
```

RPM is a convenient (but sometimes frustrating!) way to package a bunch of files and programs together. It's convenient because it checks for all dependencies before installing stuff to make sure your environment can support the new tools. It's frustrating because sometimes it takes a long time to get your environment ready to support the new package. I guess that's not RPM's fault… but it still can be frustrating!

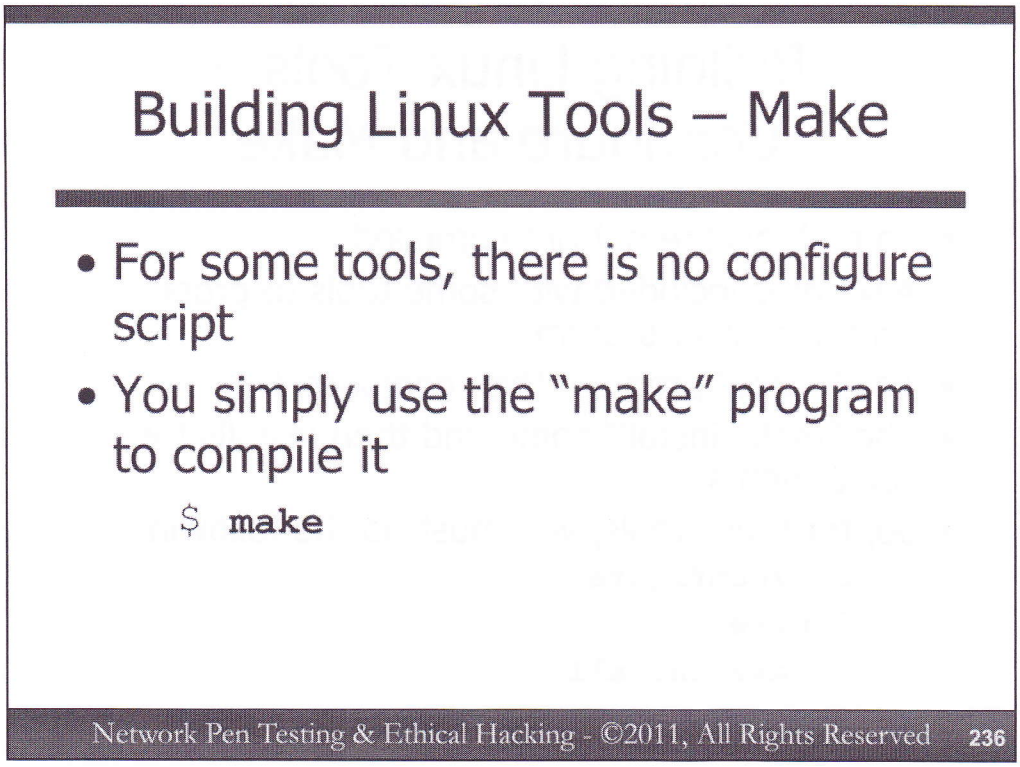
To install a package, you use the "-i" option. To uninstall, use "-e".

We won't demo this during the Intro to Linux mini-workshop. You'll get a chance to use RPMs at another time.

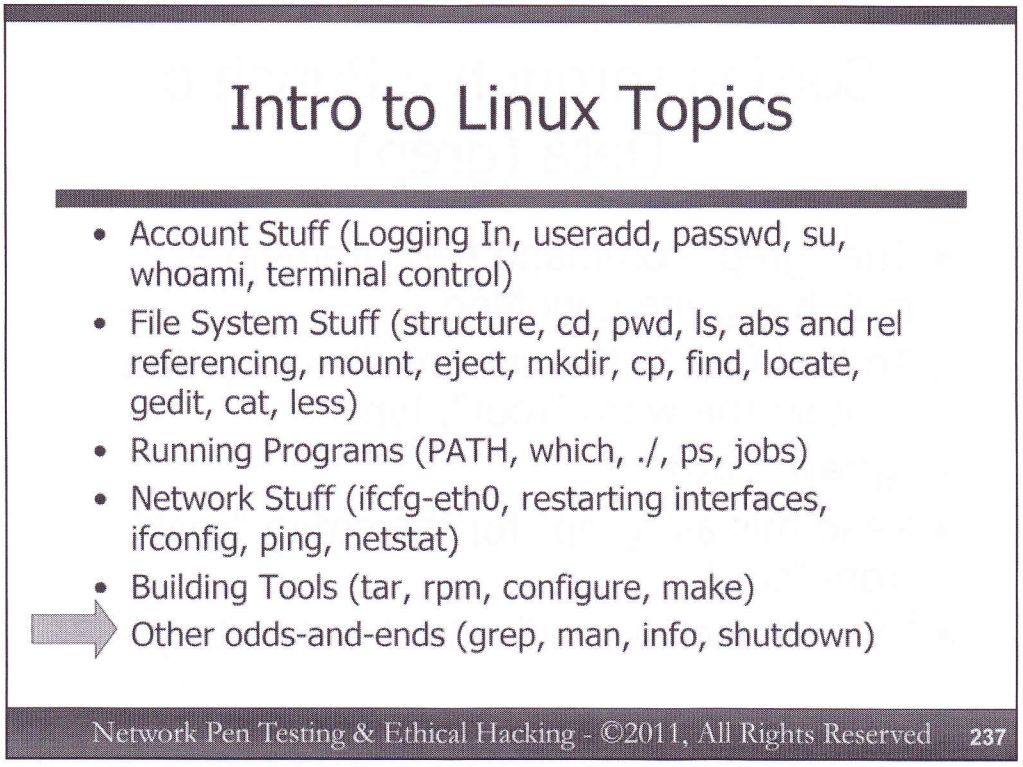



While some programs ship as tar files and others as RPMs, many just ship with a script called "configure". You need to run this script first, which checks your environment and creates a set of options necessary to get the tool compiled on your device. After running configure, you run the "make" command, which compiles and builds the tool. Then, by typing "make install", the program is loaded into the appropriate place.

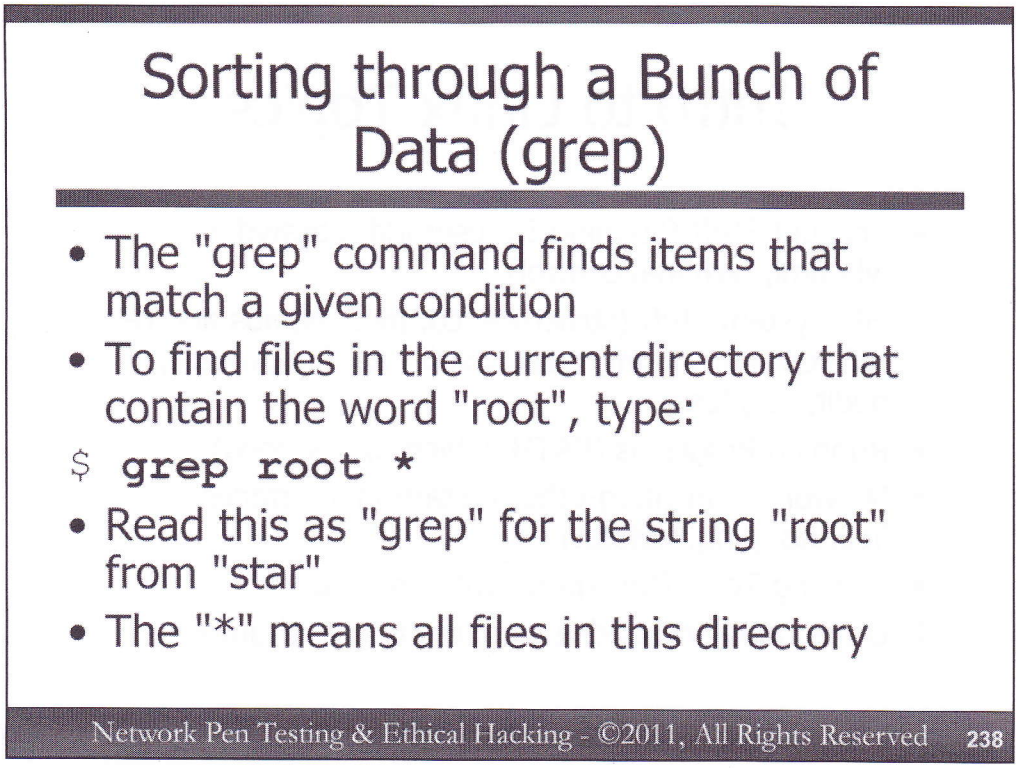
We won't demo this during the Intro to Linux mini-workshop. You'll get a chance to use configure and make during the main class.




Some tools don't have a "configure" script. For these, you just run the "make" command.




Here are some other odds and ends that will help us use Linux throughout this course.




Grep is a very powerful tool for finding data. It can look through files or the output from commands to identify particular strings. We will just scratch the surface of its use.

To look for a given string in a set of files in a directory, you can type:

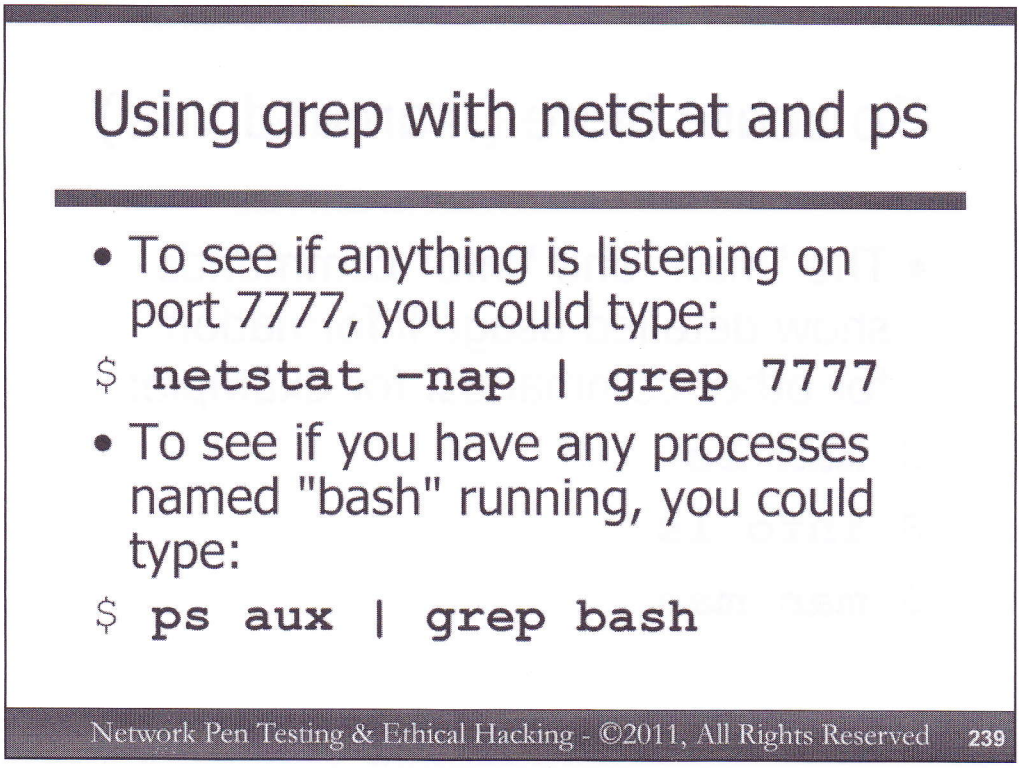
```
$ grep root *
```

This prints all occurrences of the word "root" and the file in which it appears in the current working directory. Try the following:

```
$ cd /etc
```

```
$ grep root *
```

See the word "root" in any files here? Which ones?




Grep can really help us isolate information about the usage of particular ports and processes.

At the command prompt, type:

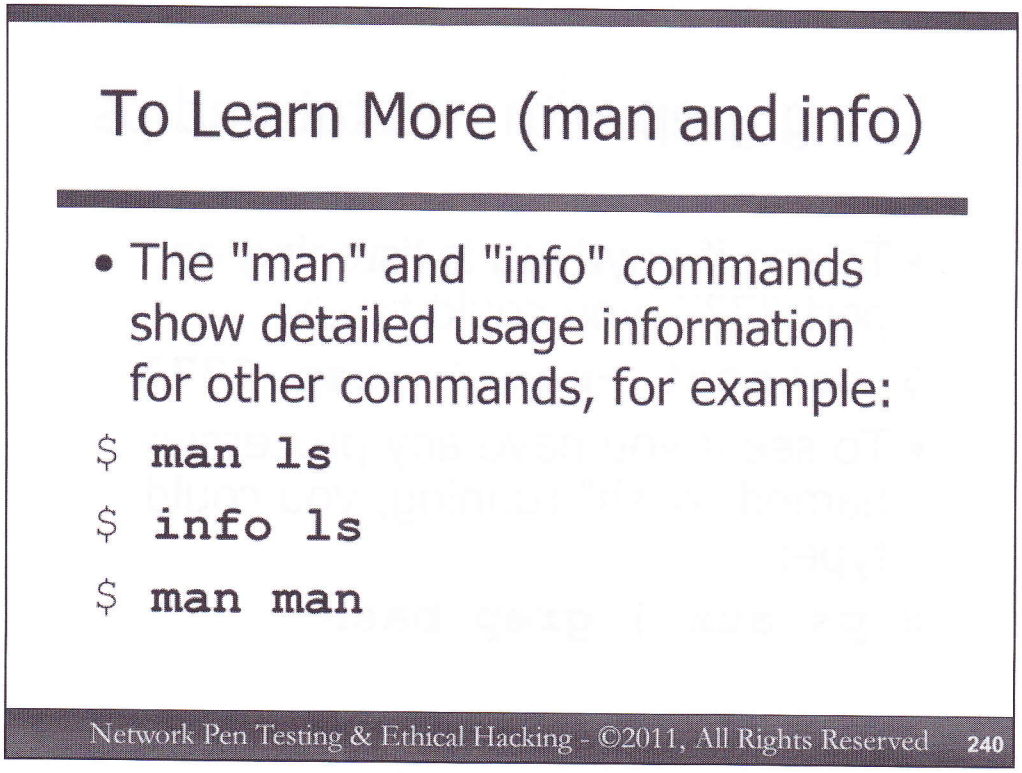
```
$ netstat -nap | grep 7777
```

This says, "Run the netstat command to show me TCP and UDP port usage, send the output to grep, and have grep show me any lines with the string 7777 in it." The results will indicate if anything is listening on or using port 7777.

Likewise, you can use grep to help you find particular programs. At a command prompt, type:

```
$ ps aux | grep bash
```

This will show you all processes running the bash program (the command shell you are running) that are currently executing on your system.




To learn more about Linux, you can use man or info.

Try the following:

```
$ man ls
```

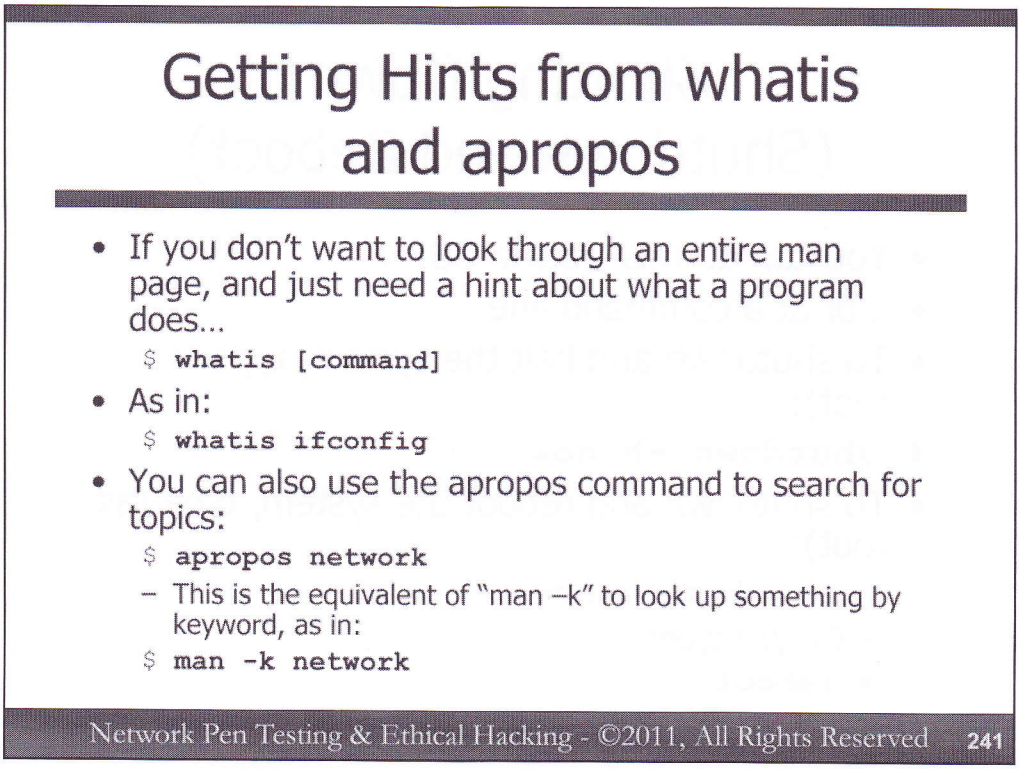
Interesting… and chilling. The "ls" command is very complex!

Also, try:

```
$ info ls
```

And, check this out to learn more about "man":

```
$ man man
```




The "whatis" command is useful for getting hints from the system about what various commands do. It won't change your life, but it might just jog your memory about some esoteric command.

I usually just use the man page myself, but some people prefer whatis.

Try typing:

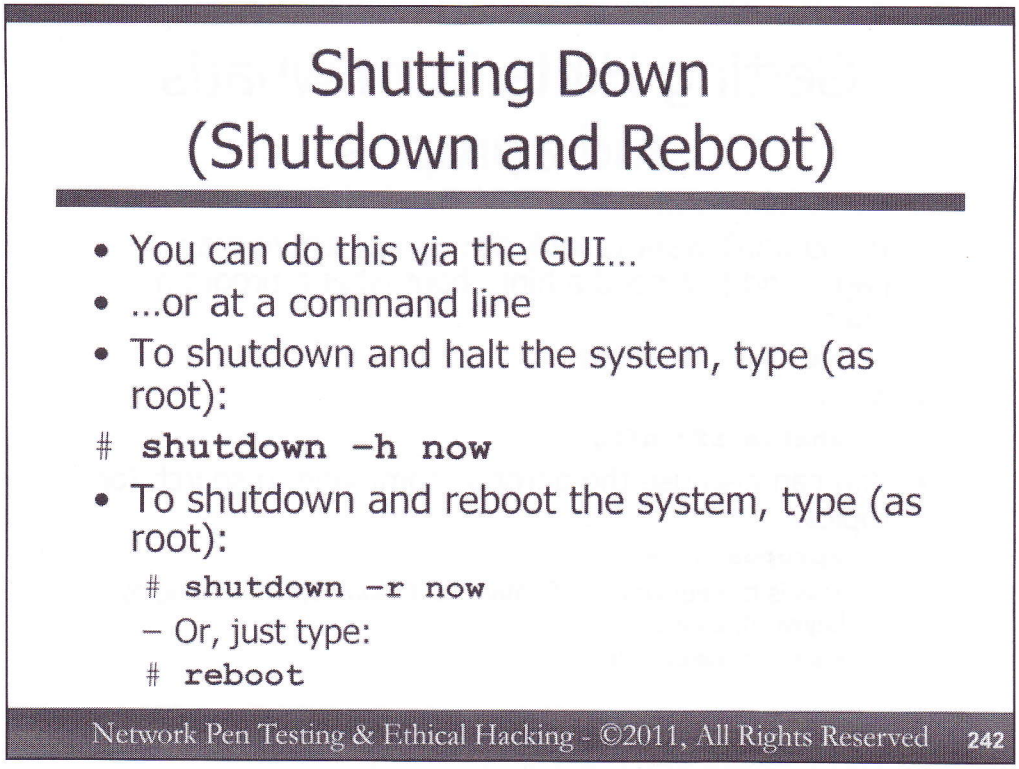
```
$ whatis ifconfig
```

You can also use the apropos command to search for topics and the commands related to those topics:

```
$ apropos network
```

This is the equivalent of "man –k" to look up something by keyword, as in:

```
$ man -k network
```




When you are done with Linux, you really should shut it down gracefully.

You can do this from the GUI, but I usually just do it from the command prompt.

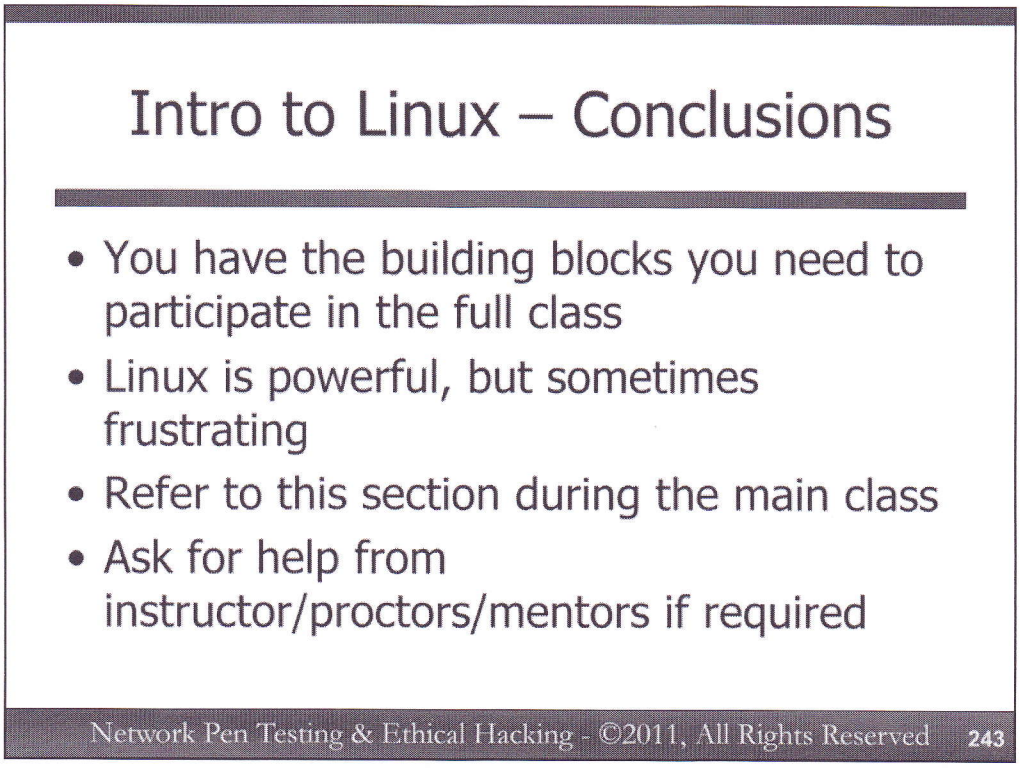
As root (you may need to su!), to gracefully shut down your system, type:

```
# shutdown -h now
```

The "-h" flag means "halt" the system. Of course, "now" means do it right away. You can actually schedule the system to shutdown at another time using this command, too.

You can also use the "shutdown" or "reboot" command to reboot the machine. To reboot, I usually just type:

```
# reboot
```




You are now ready for the full class Linux exercises! Use your new-found Linux skills for good, not evil!

## Results: ExifTool Metadata

| | xls File | doc File | pdf File |
|---|---|---|---|
| *Names* | Bob the Awesome | Bob the Awesome | Bob Boberson |
| *Account Names* | Bob, Boberson, and variants | Bob, Boberson, and variants | |
| *Applications in Use* | Excel | Word | Bullzip PDF Printer |
| *Versions* | 9 (2308) | 9 (2308) | |
| *E-Mail Addresses* | bob.boberson @560gc.tgt | bob.boberson@560gc.tgt | |
| *Directories* | | \\webserver\wwwroot\images\560gc_logo.jpg | |
| *Other* | | Document Editing Time: 22 Minutes | |

The information listed above was retrieved from the various files. Note that, while no single file gave us every item we sought, when the three files are considered together, we were able to populate each row of the table. That is, while some table cells were left empty, every row has some data in it, which we can then use in later stages of our penetration test.

THE NEXT FEW SLIDES SHOW HOW EACH OF THESE ITEMS WAS RETRIEVED FROM THE FILES USING EXIFTOOL. If you missed any of the data when working on the exercise, please review the following slides, which show you where the data was found via ExifTool.