# CHAPTER 2 Information Gathering and Vulnerability Scanning



# Episode 2.01 - Scanning and Enumeration

Objective 2.2 Given a scenario, perform active reconnaissance

#### PENTEST+ EXAM DOMAINS

DOMAIN	PERCENTAGE OF EXAM
1.0 Planning and Scoping	14%
2.0 Information Gathering and Vulnerability Scanning	22%
3.0 Attacks and Exploits	30%
4.0 Reporting and Communication	18%
5.0 Tools and Code Analysis	16%
TOTAL	100%



# INFORMATION GATHERING

#### Scanning

- Process of looking at some number of "things" to determine characteristics
- Commonly used in pen testing to uncover target vulnerabilities
- Many types of scan targets
  - Networks
  - Network devices
  - Computers
  - Applications/services

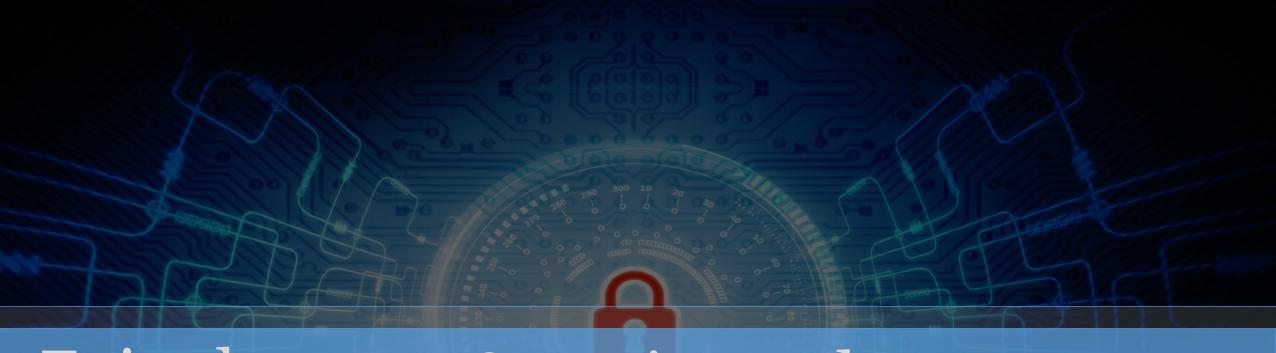
#### ENUMERATION

- Counting the detected instances of some target class
- Pen testing target classes
  - Hosts
  - Networks
  - Domains
  - Users
  - Groups
  - Network shares

- Web pages
- Applications
- Services
- Tokens
- Social networking sites

# QUICK REVIEW

- Scanning helps to determine what is "out there"
- Don't just scan for computers look for all devices and services
- Start collecting and classifying target information
- Use more than just utilities that scan networks



# Episode 2.02 - Scanning and Demo

Objective

2.3 Given a scenario, analyze the results of a reconnaissance exercise

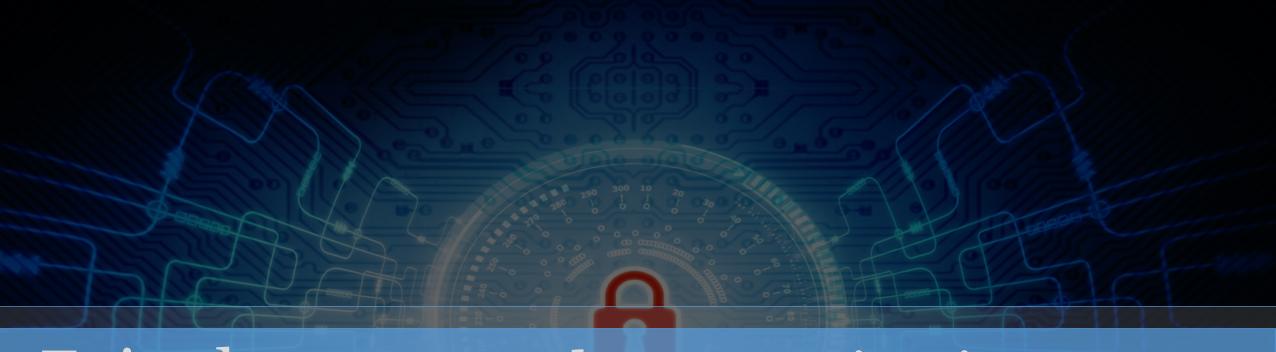
2.4 Given a scenario, perform vulnerability scanning

# SCANNING AND ENUMERATION DEMO – NMAP AND WHOIS

- Nmap demo
- Whois demo

# QUICK REVIEW

- nmap is the most common tool you'll see on the exam
- Know how to use nmap and what the main options do
- Be able to explain nmap output



# Episode 2.03 - Packet Investigation

Objective

2.2 Given a scenario, perform active reconnaissance

2.3 Given a scenario, analyze the results of a reconnaissance exercise

#### PACKET INVESTIGATION

- Packet crafting
  - Creating specific network packets to gather information or carry out attacks
  - Tools netcat, nc, ncat, hping
- Packet inspection
  - Capturing and analyzing network packets
  - Wireshark



#### INSPECTING TARGETS

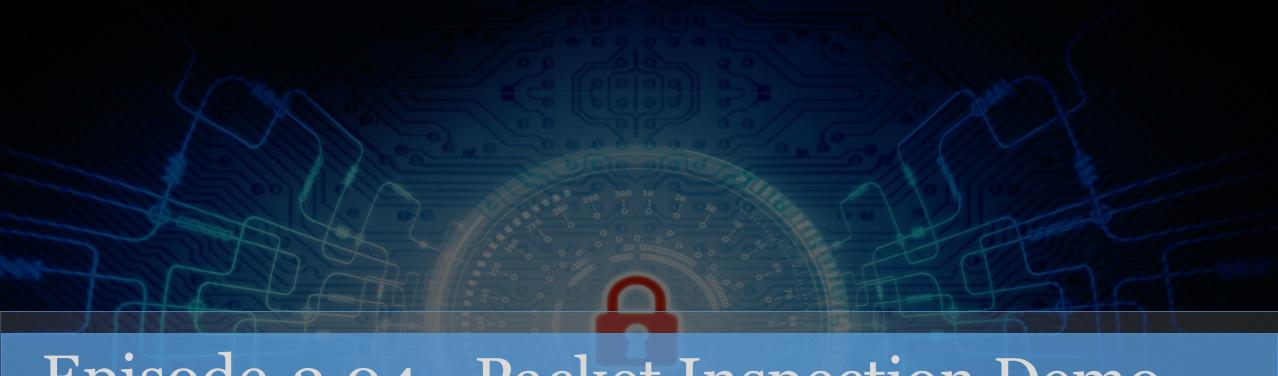
- Fingerprinting
  - Determining OS type and version a target is running
- Cryptography
  - Inspecting certificates

#### EAVESDROPPING

- RF communication monitoring
- Sniffing
  - Intercepting packets and inspecting their contents
  - Wired
    - Wireshark and tcpdump
  - Wireless
    - Aircrack-ng

# QUICK REVIEW

- netcat, nc, ncat, and hping can all help craft packets
- Constructed packets can help determine what a target is
- Wireshark is a common packet capture and inspection tool
- Fingerprinting tells you what operating systems your targets are running



# Episode 2.04 - Packet Inspection Demo

Objective 2.2 Given a scenario, perform active reconnaissance

#### PACKET INSPECTION DEMO

• Wireshark Demo

# QUICK REVIEW

- Wireshark allows you to inspect network traffic
- Useful to see what is being sent between nodes
- Practice examining network traffic in your lab



Objective (none)

# Lab: Introducing Labtainers

- Labtainers overview
  - https://nps.edu/web/c3o/labtainers
- Installing Virtualbox
  - https://www.virtualbox.org/
- Downloading Labtainers appliance
- Installing Labtainers
- Configuring/updating Labtainers
  - https://nps.edu/web/c3o/support1
- Launching Labtainers



# Episode 2.06 – Labtainers Lab (Wireshark)

Objective 2.2 Given a scenario, perform active reconnaissance



# Episode 2.07 - Application and Open-Source Resources

Objective 2.1 Given a scenario, perform passive reconnaissance



#### **DECOMPILATION**

- Compiler translates source code into executable instructions
- Decomompiler –
   attempts to convert
   executable instructions
   back into source code
  - Output is generally awkward to read at best
- Sometimes target is not a direct executable (i.e. Java)



#### **DEBUGGING**

- Running an executable in a controlled manner
- Debuggers make it easy to stop and examine memory at will
- Can reveal a program's secrets and weaknesses
- Tools windbg

#### OPEN SOURCE INTELLIGENCE GATHERING

- Open Source Intelligence Gathering (OSINT)
- Sources of research
  - CERT (Computer Emergency Response Team) <a href="https://www.us-cert.gov/">https://www.us-cert.gov/</a>
  - NIST (National Institute of Standards and Technology) <a href="https://csrc.nist.gov/">https://csrc.nist.gov/</a>
  - JPCERT (Japan's CERT) <a href="https://www.jpcert.or.jp/english/vh/project.html">https://www.jpcert.or.jp/english/vh/project.html</a>

# OPEN SOURCE INTELLIGENCE GATHERING, cont'd

- More sources of research
  - CAPEC (Common Attack Pattern Enumeration & Classification) <a href="https://capec.mitre.org/">https://capec.mitre.org/</a>
  - Full disclosure Popular mailing list from the folks who brought us nmap <a href="http://seclists.org/fulldisclosure/">http://seclists.org/fulldisclosure/</a>
  - CVE (Common Vulnerabilities and Exposures) <a href="https://cve.mitre.org/">https://cve.mitre.org/</a>
  - CWE (Common Weakness Enumeration) <a href="https://cwe.mitre.org/">https://cwe.mitre.org/</a>

# QUICK REVIEW

- Decompilers and debuggers can help to see what a program is doing
- Lots of useful attack information is available online
- Use scan output to determine target vulnerabilities
- Efficient penetration testing depends on correlated information



### Episode 2.08 –Passive Reconnaissance

Objective 2.1 Given a scenario, perform passive reconnaissance

- Cloud vs. self-hosted
- Social media scraping
- Cryptographic flaws
  - Secure Sockets Layer (SSL) certificates
    - https://gbhackers.com/testssl-sh-tls-ssl-vulnerabilities/
  - Revocation
    - Learn how and why tokens are revoked
    - Avoid allowing revocation of tokens used in tests

- Data
  - Password dumps
    - https://haveibeenpwned.com/
  - File metadata
    - Location/date/time/additional info
  - Strategic search engine analysis and enumeration
    - Shodan
      - https://www.shodan.io/
    - Censys
      - https://censys.io/

- Data
  - Web site archiving and caching
    - <a href="https://www.exploit-db.com/google-hacking-database">https://www.exploit-db.com/google-hacking-database</a>
    - Waybackmachine
      - https://archive.org/

- Data
  - Public source code repositories
    - GitHub
      - <a href="https://github.com/">https://github.com/</a>
    - Sourceforge
      - https://sourceforge.net/
    - Bitbucket
      - https://bitbucket.org/

### OPEN-SOURCE INTELLIGENCE (OSINT)

- Tools
  - Recon-ng
    - https://www.kali.org/tools/recon-ng/
  - ThreatPinchLookup
    - https://github.com/cloudtracer/ThreatPinchLookup
  - vFeed
    - https://github.com/toolswatch/vFeed
  - Vulnix
    - <a href="https://github.com/flyingcircusio/vulnix">https://github.com/flyingcircusio/vulnix</a>
  - Great list of OSINT tools
    - https://securitytrails.com/blog/osint-tools

### OPEN-SOURCE INTELLIGENCE (OSINT)

- Sources
  - National Vulnerability Database (NVD)
    - https://nvd.nist.gov/
  - CVE versus NVD?
    - https://cve.mitre.org/about/cve and nvd relationship.
       html

# QUICK REVIEW

- Passive reconnaissance collects information without announcing attention
- Tactics include collecting external data
- Published information and social media are common sources



# Episode 2.09 Active Reconnaissance

Objective 2.2 - Given a scenario, perform active reconnaissance

- Defense detection
  - Load balancer detection
  - Web application firewall (WAF) detection
    - WafWoof (Python tool)
      - <a href="https://github.com/sandrogauci/wafwoof">https://github.com/sandrogauci/wafwoof</a>
    - http-waf-detect nmap script
      - <a href="https://nmap.org/nsedoc/scripts/http-waf-detect.html">https://nmap.org/nsedoc/scripts/http-waf-detect.html</a>

#### PERFORM ACTIVE RECONNAISSANCE

- Defense detection
  - Antivirus
    - Examine e-mail headers and footers
    - BeEF / mitmf (Kali)
      - <a href="https://securityonline.info/detect-antivirus-software-victim-machine-without-user-interaction/">https://securityonline.info/detect-antivirus-software-victim-machine-without-user-interaction/</a>
  - Firewall
    - Firewalk (Kali)
      - https://www.kali.org/tools/firewalk/

#### PERFORM ACTIVE RECONNAISSANCE

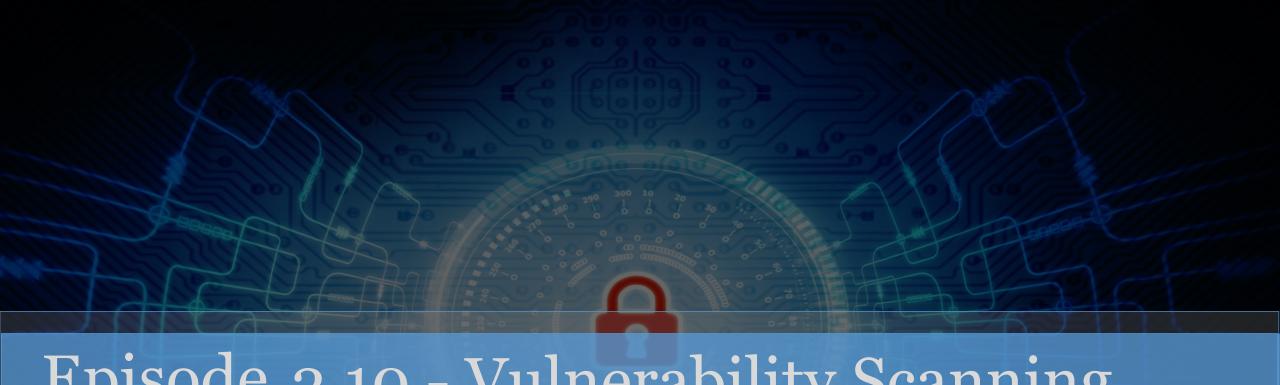
- Wardriving
  - Great overview/resource
    - https://www.geeksforgeeks.org/wardrivingintroduction/
- Cloud asset discovery
  - CloudBrute (Kali)
    - https://www.kali.org/tools/cloudbrute/
  - OWASP Amass
    - <a href="https://github.com/OWASP/Amass">https://github.com/OWASP/Amass</a>

#### PERFORM ACTIVE RECONNAISSANCE

- Third-party hosted services
  - Similar approach to cloud asset discovery
  - Often more restrictions (external)
- Detection avoidance
  - Passive reconnaissance presents the lowest risk of detection
  - Active reconnaissance options may be noisy and more likely to trigger alarms
  - Determine the level of stealth desired

## QUICK REVIEW

- Active reconnaissance collects information directly from a target's environment
- Easier to detect than passive techniques
- Often returns more detailed information
- Useful in identifying targets and developing specific attacks



## Episode 2.10 - Vulnerability Scanning

Objective 2.4 Given a scenario, perform vulnerability scanning



#### VULNERABILITY SCAN

- Structured approach to examining targets to identify known weaknesses
- Many different types
- Determine if any known weaknesses exist

#### CREDENTIALED VS. NON-CREDENTIALED

- Credentialed (authenticated) accessing resources using valid credentials
  - More detailed, accurate information
- Non-credentialed (non-authenticated) anonymous access to exposed resources
  - Fewer details, often used in early phases of attacks/tests

#### TYPES OF SCANS

- Discovery scan used to find potential targets
  - Identity/info gathering early on
  - nmap ping sweep
    - nmap –sP target

#### TYPES OF SCANS

- Full scan scans ports, services, and vulnerabilities
  - Full scan with fingerprinting
    - nmap –A <target>
      - Not stealthy
    - perl nikto.pl -h <target>
    - OpenVAS
      - Open-source version of Nessus
- Port scan
  - nmap -p <ports> <target>v

#### TYPES OF SCANS

- Stealth scan attempt to avoid tripping defensive control thresholds
  - nmap -sS <target>
- Compliance scan for specific known vulnerabilities that would make a system noncompliant

## **QUICK REVIEW**

- Structured approach to discovering target vulnerabilities
- Correlates known vulnerabilities with target characteristics
- Scans can be general (find any weaknesses) or targeted (see if specific weaknesses exist)
- Scans can range from quiet to very noisy



## Episode 2.11 - Vulnerability Scanning Demo

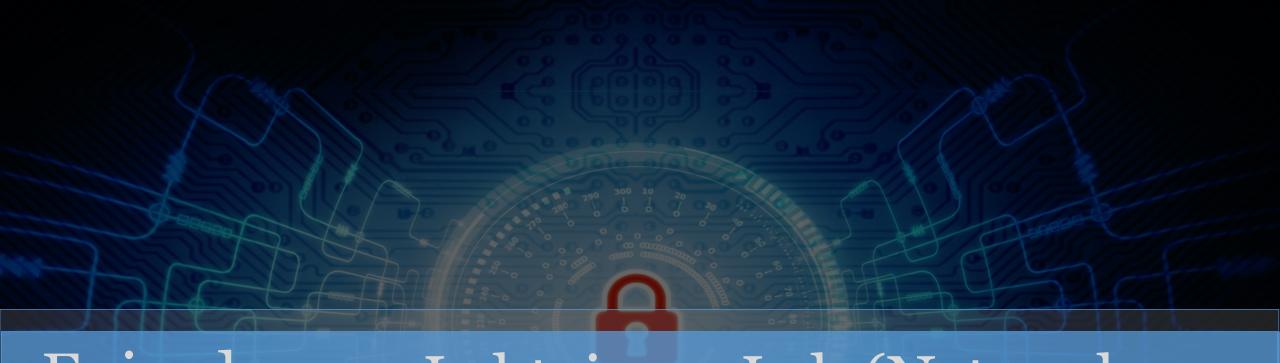
Objective 2.4 Given a scenario, perform vulnerability scanning

## Scanning Demo

- Nmap
- Nikto
- OpenVAS

## QUICK REVIEW

- Practice with various nmap scan options
- Use Nikto to perform your own scans in the lab environment
- Try using OpenVAS to perform different scans in your lab



## Episode 2.12 Labtainers Lab (Network Basics)

Objective 2.2 Given a scenario, perform active reconnaissance

### Lab – Networking: network-basics

• Intro lab



# Episode 2.13 Labtainers Lab (Nmap Discovery)

Objective 2.4 Given a scenario, perform vulnerability scanning

## Lab – Networking: nmap-discovery

Show ssh



## Episode 2.14 - Target Considerations

Objective 2.4 Given a scenario, perform vulnerability scanning



#### CONTAINER

- Lightweight instance of a VM
  - Runs on top of host OS
- Docker, Puppet, Vagrant



#### APPLICATION SCAN

- Dynamic analysis
  - Target environment is running and responds to queries
- Static analysis
  - Collect artifacts for post-execution analysis



#### SCANNING CONSIDERATIONS

- Time to run scans
  - Approved schedule (planning)
- Protocols used
  - Largely dependent on target selection
- Network topology
  - Network layout (diagram) of test targets
- Bandwidth limitations
  - Tolerance to impact (affects availability)



#### SCANNING CONSIDERATIONS

- Query throttling
  - Slow down test iterations to avoid exceeding bandwidth
  - nmap -T
- Fragile systems/nontraditional assets
  - How to avoid impacting fragile mission critical systems?



#### ANALYZE SCAN RESULTS

- Asset categorization
  - Identify and rank assets by relative value
  - Vulnerable assets with little value could be a waste of time
- Adjudication
  - Determine which results are valid
    - False positives
    - Filter out false positives



#### ANALYZE SCAN RESULTS, cont'd

- Prioritization of vulnerabilities
  - Highest impact vulnerabilities ease of exploit vs. payoff
- Common themes
  - Vulnerabilities
  - Observations
  - Lack of best practices

## QUICK REVIEW

- Know how to determine if targets are physical machines or are virtualized (i.e. footprinting)
- Be aware of client restrictions when running scans (i.e. bandwidth use, schedule, etc.)
- Don't waste time on results that have little value focus on the most meaningful results
- Prioritize the highest impact vulnerabilities



## Episode 2.15 Analyzing Scan Output

Objective 2.3 Given a scenario, analyze the results of a reconnaissance exercise

- DNS lookups
  - Nslookup
  - Whois
    - https://www.whois.com/whois/

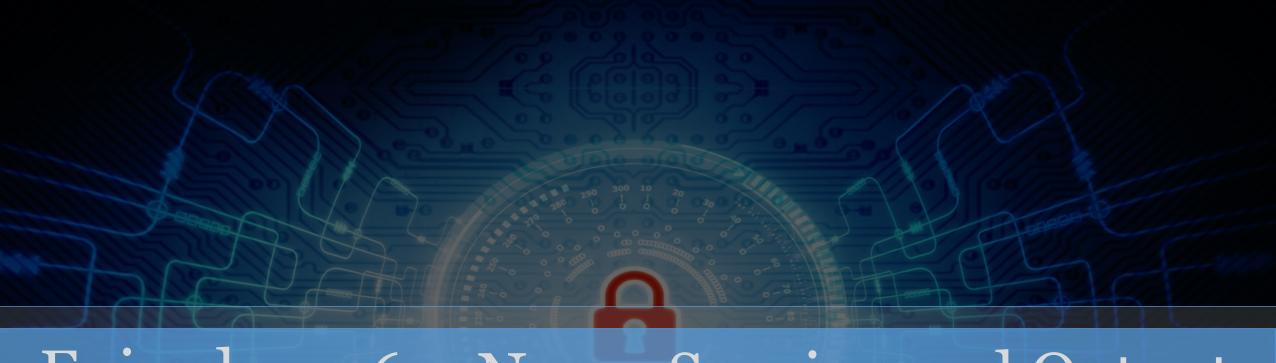
- Crawling websites
  - Netcat
  - W3af
    - http://w3af.org/
  - Burp Suite (Kali)
    - https://www.kali.org/tools/burpsuite/
  - Crawlergo
    - <a href="https://hacker-gadgets.com/blog/2021/10/15/crawlergo-a-powerful-browser-crawler-for-web-vulnerability-scanners/">https://hacker-gadgets.com/blog/2021/10/15/crawlergo-a-powerful-browser-crawler-for-web-vulnerability-scanners/</a>
  - OWASP zaproxy (Kali)
    - https://www.kali.org/tools/zaproxy/

- Network traffic
  - Wireshark
    - https://www.wireshark.org/
- Address Resolution Protocol (ARP) traffic
  - Wireshark
    - https://www.wireshark.org/

- Nmap scans
- Web logs
  - IIS
    - https://www.sumologic.com/blog/iis-logs/
  - Apache
    - <a href="https://linuxconfig.org/how-to-analyze-and-interpret-apache-webserver-log">https://linuxconfig.org/how-to-analyze-and-interpret-apache-webserver-log</a>

## QUICK REVIEW

- Different tools provide different levels of details
- Analyzing tool output identifies important information
- Provides input for building an attack plan



## Episode 2.16 – Nmap Scoping and Output Options

Objective 2.4 Given a scenario, perform vulnerability scanning



#### **NMAP**

- nmap (Network Mapper)
  - One of the most common and most useful tools for reconnaissance
  - nmap -A does much of what we're about to see



## SYN SCAN vs. FULL CONNECT SCAN

- SYN (stealth) scan
  - nmap –sS target
  - Sends SYN packet and examines response (SYN/ACK means the port is open)
  - If SYN/ACK received, nmap sends RST to terminate the connection request
- Full connect scan
  - nmap –sT target
  - Completes the handshake steps to establish a connection (more reliable)



#### PORT SELECTION (-p)

- Scans a range of ports
   nmap-p <range of
   ports> target
  - -p 21
  - -p 1-10000
  - -p U:53,137,161T:21-37,80,8080
  - OR –exclude-port <range of ports>



### SERVICE IDENTIFICATION (-sV)

- Service identification (-sV)
  - nmap -sV <target>
  - Attempts to determine service and version info
    - --version-intensity<level>, where levelcan be o (light) to 9(execute all probes)



#### TIMING (-T)

- Changes how long
   nmap waits for a
   response (default is –
   T 3)
  - Values range from o (Paranoid, slow) to 5 (Insane, fast)



#### OUTPUT PARAMETERS

- -oA Combined format
  - Normal .txt, XML .xml, and grepable .txt
- -oN
  - Normal output file (.nmap)
- -oG
  - Grepable output file (.gnmap)
- -oX
  - XML output format (.xml)



#### GATHERING INFORMATION WITH NMAP

- OS fingerprinting (-O)
  - Detects target OS
  - nmap -O <target>
- Disabling ping (-Pn)
  - Skips host discovery (assumes all are online)
  - nmap -Pn <target>

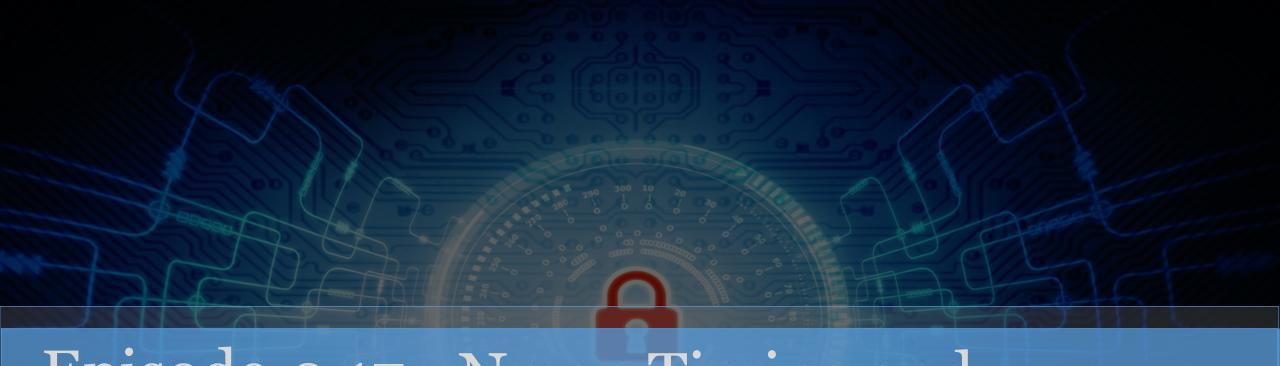


#### GATHERING INFORMATION WITH NMAP

- Target input file (-iL)
  - Uses a text file that contains a list of targets
    - nmap -iL <inputFileName>
  - Can also exclude targets from a range
    - nmap –excludefile <excludeFileName>

## **QUICK REVIEW**

- Stealth scans (nmap -sS <target>) create fewer network packets than full connect scans (nmap -sT <target>)
- Nmap service identification (nmap -sV <target>) attempts to identify the service and version monitoring each port
- Nmap returns results faster if targets aren't pinged and are just assumed they're alive (nmap -Pn <targets>)
- To avoid detection, use the nmap timing option with a lower number (nmap -T0 <target> or nmap -T1 <target>)



# Episode 2.17 - Nmap Timing and Performance Options

Objective 2.4 Given a scenario, perform vulnerability scanning

## Scanning demo

• Nmap demo

## QUICK REVIEW

- Understand the nmap timing option values (-T 0-5)
- Be able to explain what actions nmap -A performs
- Show how to restrict nmap scans to specific ports



# Episode 2.18 - Prioritization of Vulnerabilities

Objective 2.4 Given a scenario, perform vulnerability scanning



## LEVERAGE INFORMATION

- Leveraging information to prepare for exploitation
- Map vulnerabilities to potential exploits
  - Look up vulnerabilities found for possible exploits
  - Nmap vulners and vulscan scripts
  - Metasploit (search vulnerability)



## LEVERAGE INFORMATION

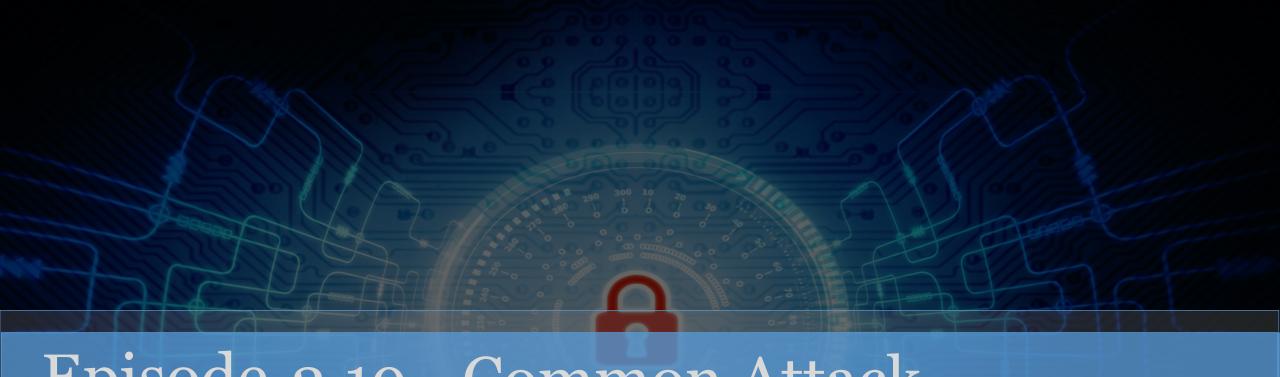
- Prioritize activities in preparation for penetration test
  - Will standard exploits work?
  - Will exploits need to be 'tweaked'?
  - Additional steps to prepare test?

### Demo

• Demo

## QUICK REVIEW

- A key step in pen test planning is to map vulnerabilities to potential exploits
- Use nmap scripts (vulners and vulscan) to find exploits for detected vulnerabilities
- Use Metasploit to search for exploits



### Episode 2.19 - Common Attack Techniques

Objective 2.4 Given a scenario, perform vulnerability scanning



### COMMON ATTACK TECHNIQUES

- Some Windows
   exploits can be run in
   Linux
- Cross-compiling code
  - Compile exploit for another OS
  - https://www.hackingtutor
     ials.org/exploit tutorials/mingw-w64 how-to-compile-windows exploits-on-kali-linux/



### COMMON ATTACK TECHNIQUES

- Exploit modification
  - May need to modify for success of evasion
- Exploit chaining
  - Compromise one device/system to gain access to another
- Proof-of-concept development
  - Exploit development

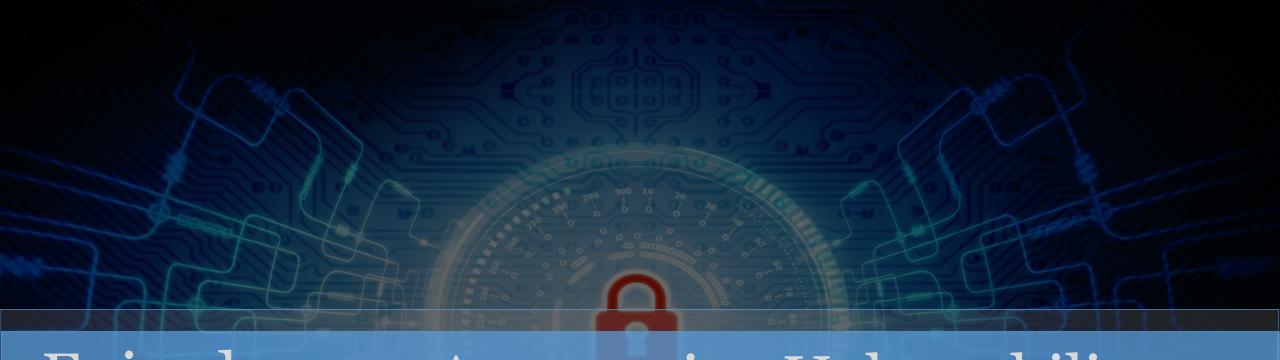


## COMMON ATTACK TECHNIQUES

- Social engineering
  - Help me
  - Urgent
  - Deceptive
- Credential brute forcing
- Enlightened attacks
  - Dictionary
  - Rainbow table

## QUICK REVIEW

- Some exploits may need "tweaking" to work in your tests
- Be able to recognize exploit chaining
- Many exploits involve some social engineering
- Credential attacks are time consuming and are rarely carried out as pure brute force attacks



# Episode 2.20 Automating Vulnerability Scans

Objective 2.4 Given a scenario, perform vulnerability scanning

### AUTOMATION

- Repeatability
- Minimal interaction
- Exception-only notification
- Must handle multiple environments
  - Multi-OS
  - Cloud, on-premises, and third-party

## VULNERABILITY TOOLS AND AUTOMATION

- Command-line tools are easy to automate
  - Scripts
  - Schedulers
  - Analysis to minimize false positives

## QUICK REVIEW

- Automation makes actions repeatable
- Increases speed and quality
- Easier to carry out multiple actions with minimal interaction



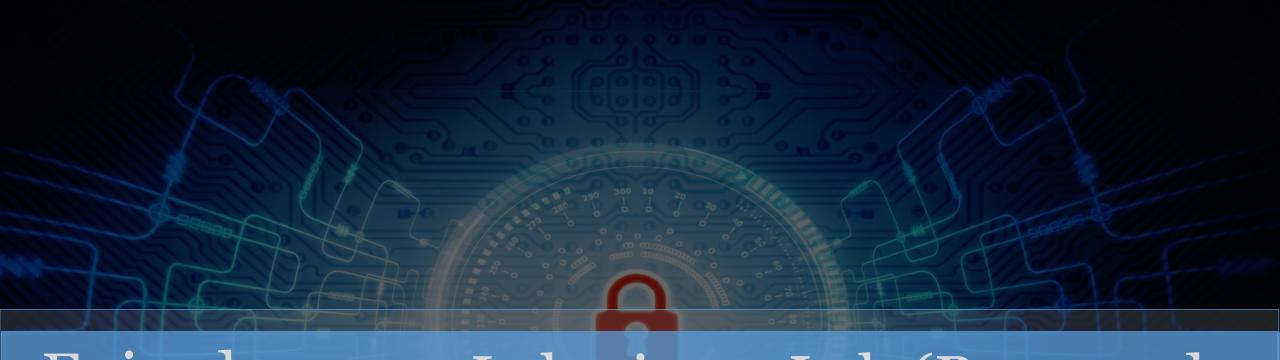
Objective (none)

### Demo – Password cracking

- Demo Hydra
- Bad usernames and passwords
  - Daniel Miessler's SecLists <a href="https://github.com/danielmiessler/SecLists/tree/m">https://github.com/danielmiessler/SecLists/tree/m</a> <a href="https://asswords">aster/Passwords</a>

## QUICK REVIEW

- Most credential attacks depend on good dictionaries
- Each pen tester must maintain username and password lists for credential attacks
- Start with good online resources, and modify for your own purposes

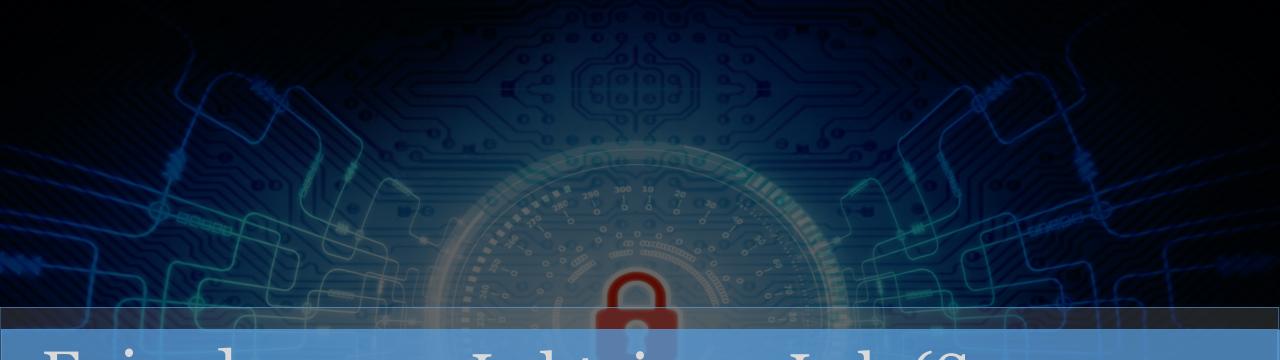


# Episode 2.22 - Labtainers Lab (Password Cracking)

Objective 2.1 Given a scenario, perform passive reconnaissance

### Lab Systems Security & Operations: passcrack

• Intro lab

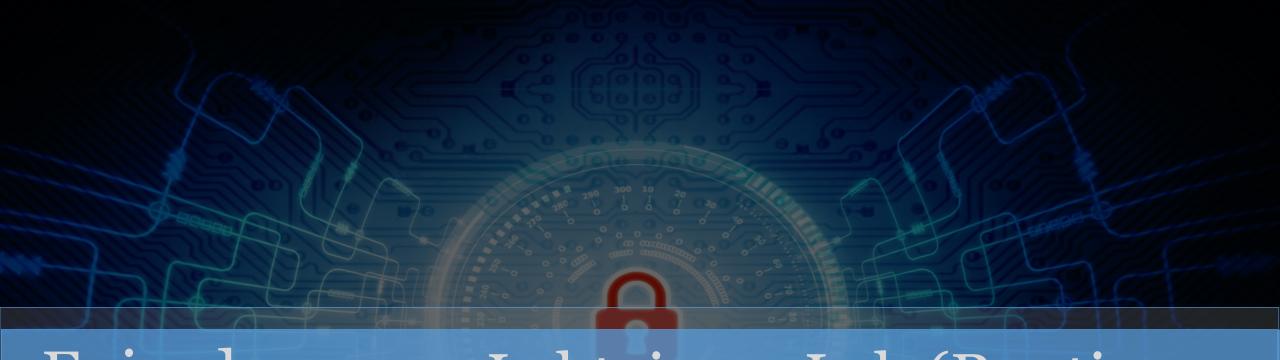


# Episode 2.23 - Labtainers Lab (Secure Socket Layers)

Objective 2.1 Given a scenario, perform passive reconnaissance

### Lab – Crypto Labs: ssl

- Intro purpose of lab
  - SSL demo
  - Wireshark



# Episode 2.24 - Labtainers Lab (Routing Basics)

Objective 2.1 Given a scenario, perform passive reconnaissance

### Lab – Networking: routing-basics

• Intro lab