LAB SETUP

- Install Virtualbox
- Acquire and import Metasploitable2 and DVWA
- Set up VM host network addresses:
    - Adapter IP: 10.10.1.1
    - DHCP Server:
        - IP: 10.10.1.2
        - Lower Address Bound: 10.10.1.10
        - Upper Address Bound: 10.10.1.254
    - Subnet Mask: both at 255.255.255.0

COURSES:

1. Planning & Scoping
2. Survey the Target
3. Select your Attacks
4. Select Your Attacks II
5. Selecting Pentest Tools
6. Using Scripting in Pentesting
7. Reporting & Communication

Course 1 - Planning & Scoping

Planning a pentest:

- Get written permission
- Clearly define Scope, know how much work you have to do - don't do more than that, STAY IN SCOPE, be aware or scope creep
- Read over PTES http://www.pentest-standard.org/index.php/Main_Page
- Importance of planning
    - Lots of options in each step
    - Each pentest is often conducted differently
    - Easy to waste time and effort - experience helps avoid this
    - Project management skills are important here - will keep pentest on track

Rules of engagement:

- Know your target audience
    - Who is sponsoring the pentest?
    - What is the purpose for the test?
- Rules of engagement (ROE) - governs the pentester's activities
    - Schedule - start, stop, temporal restrictions
    - Team composition, location, access
- Test scope
    - technical/physical/personnel
    - Target limits (inclusio, invasiveness, etc)
- Communication escalation path

    - Risks of pentesting
        - Crashing devices, services, whole servers, etc
        - Corrupting data
        - Degrading performance
        - Terms of service (TOS)/regulation/legislation violation
    - Escalation path
        - Who to contact if things go wrong?
        - Communication expectations (content, trigger, frequency)

Resources & budgets

- Resources & requirements
    - What does each party provide?
    - At what point does the engagement begin?
    - Confidentiality of findings
    - Known vs unknown - is the test a secret?
- Budget
    - How much will each section of the test cost?
    - Every task in the test should have a value
        - Want to add more test? It'll cost more - provide the cost
    - One of the most important factors
        - Directly impacts available resources and time

Impact & Constraints

- Set expectations
    - Impact
        - The result of testing
        - Report vulnerabilities
        - Remediation plan/guidance - outline how client should respond
    - Disclaimers
        - Point-in-time assessment - results only valid now
    - Comprehensiveness
- Technical constraints
    - Any technical limitations that reduce test scope
    - Production (live) components
    - out -of-service devices
    - Can't access - physical/geographic access limitations
    - legal/regulatory out of scope

Support Resources

- WSDL/WADL - web services/application description language
    - XML file with lots of info about web service/application and its interface requirements
    - Check if publically available for target(s)
- SOAP project file(s)
    - Simple object access protocol - used to exchange info for web services, being displaced by REST but still in play in many areas
    - Not exposed to public

    - Used by developers in development environment
    - Project file provides low level web service interface details (input/output/server info)
- SDK documentation
    - Software development kit docs help provide info on tools used to develop software
- Swagger document
    - Popular open-source framework for developing REST services
    - REST is a lightweight API
    - Document can provide internal info on REST services exposed to clients
- XSD
    - XML schema definition - defines XML document content
- Look at sample application requests
    - Well-formed requests, generally to web services
    - Useful when testing web services/applications of all types
- Architectural diagrams
    - Diagrams of networks and connected devices
    - Helpful when determining targets to attack

Legal Groundwork

- Legal concepts - contracts
    - Statement of work (SOW)
        - Clearly states what tasks are to be accomplished
    - Master service agreement (MSA)
        - Specifies details of the business arrangement
    - Non-disclosure agreement (NDA)
        - Agreement that defines confidentiality, restrictions and/or sharing information
- Environmental differences
    - Export restrictions - restrictions on shipments, transfer of technology, or services outside the US
    - National or local restrictions
        - Differ among countries
        - Local customs differ
    - Corporate policies
        - Differ between most organizations
- Written authorization
    - Obtain signature from proper signing authority
        - “Get out of jail free” card
        - Pentests can reveal sensitive or confidential information
        - Activities may be illegal without proper permission
        - Signed permission makes you a white hat hacker
    - Third party authorization when necessary
        - Ex: from a Cloud service provider

        - Get permission for any outside resources used (cloud, ISP, etc)

Scope Considerations

- Types of assessments
    - Goal-based
        - Goals created up front
        - Tests set up to fulfill goal(s)
    - Objectives-based
        - Define a resource to attack
        - Tests use all angles to attack protected objectives
    - Compliance-based
        - Mandates by standard regulation, or legislation
        - Ex: PCI DSS
    - Red team
        - Typically internal
        - Ongoing
        - A single compromise is success
    - Blue team
        - Defense against the red team
- Special scoping considerations
    - Premerger
        - Part of due diligence prior to mergers
        - Used to harmonize security efforts
    - Supply chain
        - Partners often provide software and/or hardware to interface with an organization
        - Weaknesses in interfaces can provide unauthorized access
- Target selection
    - Internal (on-site vs off-site)
    - External
    - First-party vs third-party hosted
    - Physical
    - Users
    - SSIDs
    - Applications

Project strategy & risk

- Considerations
    - Whitelisted?
        - No one can access resources unless specifically granted
    - Blacklisted?
        - Everyone can access unless specifically blocked
    - Security exceptions
        - IPS / WAF whitelist
        - NAC (network access control)
        - Certificate pinning (public key pinning)

    - Explore company policies to learn about security considerations
- Know which role you are taking
    - Black-Box pentesting
        - Zero prior knowledge
        - Most familiar to real attacker
        - Generally a surprise to internal personnel
    - White-Box pentesting
        - Full access to internal information
    - Gray-box pentesting
        - Some internal information available
        - Consistent with an insider threat with limited access
- Risk acceptance
    - Pentests can be risky - service can be interrupted
    - devices/servers can become unresponsive
    - How much risk is the client willing to accept?
    - Acceptance means = willing to accept risks, based on likelihood & impact
    - Tolerance to impact
        - Is risk is realized, what is client's tolerance to the results?
        - How much disruption is tolerable?

Scope Vulnerabilities

- Scheduling & Scope Creep
    - Scheduling
        - When can/should test be run?
        - Who should be notified?
        - When must tests be completed?
    - Scope creep - common in nearly all projects
        - Client requests additional tasks after SOW if signed
        - Many tasks may seem "doable", but always stay in scope
        - Takes resources away from core SOW tasks
        - MUST get authorization for any SOW modifications
    - Threat actors
        - Aversary tier - what role should the pentester assume?
            - APT (Advanced Persistent Threat)
            - Script kiddies
            - Hacktivist
            - Insider threat
        - Capabilities
            - What resources does the attacker(s) have?
            - Organized & sponsored attackers have more equipment & sophistication
        - Intent
            - power/revenge
            - status/validation
            - Monetary gain

        - Ideology
    - Threat model
        - Gather information and identify assets

Course 2 - Survey the Target

Rank assets > rank vulnerabilities > rank exploits

Scanning & Enumeration

- Information gathering
    - Scanning
        - Process of looking at some number of “things” to determine characteristics
        - Commonly used in pentests to uncover target vulnerabilities
    - Many types of scan targets
        - Networks
        - Network devices
        - Computers
- Enumeration
    - Counting the detected instances of some target class
    - Pentesting target classes
        - Hosts
        - Networks
        - Domains
        - Users
        - Groups
        - Network shares
        - Web pages
        - Applications
        - Services
        - Tokens
        - Social networking sites

DEMO: scanning & enumeration demo

- Nmap to scan
    - nmap -sP [IP Address range] = (ping sweep)
- arp-scan [IP Address range] = arp scan
- whois.domaintools.com to look up domains, look for IP Address Ranges

Packet investigation

- Packet crafting
    - Creating specific network packets to gather information or carry out attacks
    - Tools - netcat, nc, ncat, hping
        - ncat = newer version of netcat, created by Nmap
- Packet inspection
    - Capturing and analyzing network packets - wireshark
- Inspecting targets

    - Fingerprinting
        - Determining OS types and version a target is running
    - Cryptography
        - Inspecting certificates
- Eavesdropping
    - RF communication monitoring
    - Sniffing - intercepting packets and inspecting their contents
        - Wired - wireshark and tcpdump
        - Wireless - aircrack-ng

Application open-source resources

- Decompilation
    - Compiler - translates source code into executable instructions
    - Decompiler - attempts to convert executable instructions back into source code
        - Output is generally awkward to read at best
    - Sometimes a target is not a direct executable (i.e. Java)
- Debugging
    - Running an executable in a controlled manner
    - Debuggers make it easy to stop and examine memory at will
    - Can reveal a program's secrets and weaknesses
    - Tools - windbg
- OSINT
    - Open source intelligence gathering
    - Sources of research
        - CERT https://www.us-cert.gov/
        - NIST https://csrc.nist.gov/
        - JPCERT https://www.jpcert.or.jp/english/
        - CAPEC https://capec.mitre.org/
        - Full disclosure mail list - https://seclists.org/fulldisclosure/
        - CVE https://cve.mitre.org/
        - CWE https://cwe.mitre.org/index.html

Vulnerability scanning

- Vulnerability scan
    - Structured approach to examining targets to identify know weaknesses
    - Many different types
    - Determine if any known weaknesses exist
- Credentialed vs non-credentialed
    - Credentialed (authenticated) - accessing resources using valid credentials
        - More detailed, accurate information
    - Non-credentialed (non-authenticated) - anonymous access to exposed resources
        - Fewer details, often used in early phases of attacks/tests
- Types of scans
    - Discovery san - used to find potential targets
        - identity/info gathering early on
        - Nmap ping sweep [nmap -sP target]

    - Full scan - scans ports, services, and vulnerabilities
        - Full scan with fingerprinting
            - Nmap -A <target> (very noisy)
        - perl nikto.pl -h <target>
        - OpenVAS
            - Open-source version of nessus
    - Port scan
        - Nmap -p <ports> <target>
    - Stealth scan - attempt to avoid tripping defensive control thresholds
        - nmap -sS <target>
    - Compliance - scan for specific known vulnerabilities that would make a system non-compliant

DEMO - vulnerability scanning

- Nmap stealth scan: nmap -sS <target>
- Nmap port fingerprinting: nmap -p <port> -A <target>
- Nikto vulnerability scan: nikto -h <target>
- Install OpenVAS in Kali
    - apt update
    - apt install openvas
    - Openvas-setup
        - Don't forget to copy down password at end of setup!
        - OpenVAS runs on port 9392
            - Access web gui = 127.0.0.1:9332
            - Username = admin
    - Create targets > create tasks > start scanning > observe results for info

Target considerations

- Container
    - Lightweight instance of a VM
    - Runs on top of host OS
    - Docker, puppet, vagrant
- Applications
    - Application scan
        - Dynamic analysis
            - Target environment is running and responds to queries
        - Static analysis
            - Scan input consists of post-execution data stores
- Scanning considerations
    - Time to run scans - approved schedule (planning)
    - Protocols used - largely dependent on target selection
    - Network topology - network layout (diagram_ of test targets
    - Bandwidth limitations - tolerance to impact (affects availability)
    - Query throttling - slow down test iterations to avoid exceeding bandwidth
        - Nmap -T
    - Fragile systems/non-traditional assets

- Analyze scan results
    - Asset categorization
        - Identify and rank assets by relative value
        - Vulnerable assets with little value could be a waste of time
    - Adjudication
        - Determine which results are valid
            - Filter out False positives
    - Prioritize vulnerabilities
        - Highest impact vulnerabilities - ease of exploit vs payoff
    - Common themes
        - Vulnerabilities
        - Observations
        - Lack of best practices

NMAP timing and performance options

- Cheatsheet: https://www.stationx.net/nmap-cheat-sheet/
- Utilize -T for either stealthy or rapid depending on situation
- Learn NMAP!

Prioritization of vulnerabilities

- Leverage information
    - Leveraging info to prepare for exploitation
- Map vulnerabilities to potential exploits
    - Look up vulnerabilities found for possible exploits
    - Nmap - vulners and vulscan scripts
    - Metasploit (search vulnerability)
- Prioritize activities to prepare for pentest
    - Will standard exploits work?
    - Will exploits need to be “tweaked”?
    - Additional steps to prepare test?
- Install NMAP scripts
    - cd /usr/share/nmap/scripts (from / directory)
    - Git clone https://github.com/vulnersCom/nmap-vulners.git
    - Git clone https://github.com/scipag/vulscan.git
    - Use scripts
        - nmap --script nmap-vulners -sV <Target>
            - Quicker, only uses 1 database
        - nmap --script vulscan -sV <Target>
            - Longer, uses ALL databases
        - nmap --script vulscan --script-args vulscandb=exploitdb.csv -sV <Target>
            - Telling nmap to use the specified db file instead of all

Common attack techniques

- Some Windows exploits can be run in Linux
- Cross-compiling code
    - Compile exploit for another OS

    - https://www.hackingtutorials.org/exploit-tutorials/mingw-w64-how-to-compile-windows-exploits-on-kali-linux/
    - Changing exploit code can change the fingerprint of the exploit
- Exploit modification
    - May need to modify for success of evasion
- Exploit chaining
    - Compromise one device/system to gain access to another
- Proof-of-concept development
    - Exploit development
- Social engineering
    - Help me, urgent, deceptive
    - Credential brute forcing
    - Enlightened attacks
        - Dictionary (wordlist) - for online attacks
        - For offline, get the password file(s) , hashes
        - Rainbow table(s)

Credential attacks

- Hydra
    - Hydra -L <usernamelist.txt -P passwordlist.txt> <target>
    - Example: hydra -L usernamelist.txt -P passwordlist.txt ftp://10.10.1.11
- Get username list & password list, feed hydra both
    - Google common usernames > create custom list
    - You can search for password lists, utilize seclists
        - https://github.com/danielmiessler/seclists
    - The quality of your attack will depend on quality of the lists
- Start with good online resources and modify for your own purposes

Weaknesses in specialized systems

- ICS (industrial control systems)
    - Environmental conditions
    - Exposure to real world (live) events
- SCADA (supervisory control and data acquisition)
    - SCADA is the control system that interfaces with industrial processes
    - PLC (programmable logic controller) - the electronic board(s) that power the manufacturer's processes
- Mobile - lack of updates, compromised settings, dangerous apps, etc.
- IoT (internet of things) - default (weak) security (wide open)
- Embedded devices
- Point-of-sale system
    - Attractive due to connection to payment devices (cash, readers, etc.)
- Biometrics - accuracy is still evolving
    - What if primary reader fails to detect?
    - What is the backup manual process?
- Application containers
    - Container and VMs are not foolproof sandboxes

    - Compromising (breaking out) may allow access to external resources
- RTOS (real-time operating system)
    - Designed to provide fast, lightweight services, not security -- usually designed for sensory environments

Course 3 - Select Your Attacks

Social engineering

- Tricking or coercing people into violating security policy
- Depends on willingness to be helpful
- Human weaknesses can be leveraged
- May rely on technical aspects
- Phishing - people are contacted by a seemingly legitimate imposter in an attempt to extract sensitive information
    - Spear phishing - specific target(s)
    - SMS phishing
    - Voice phishing
    - Whatling - going after the ‘whales’

DEMO - Spear Phishing

- SET - social engineering toolkit
- Setup sendmail to have a relay available:
    - Apt install sendmail
    - Sendmailconfig
- SET > social engineering > spear phishing > mass email attack > choose attack
    - #6 example: RTF (rich text format file)
        - Select #2 > loads msf on machine through meterpreter and reverse shell back to us
        - Set LHOST > (internal virtual IP address)
        - Payload created into RTF file
        - Rename file
        - Choose single or mass mailer > #1
        - Choose predefined template of one-time use > #2
            - Create subject line, HTML, and body message
            - Ctrl + C when done
        - Choose who to send email to
            - Select gmail or own server/relay > #2
            - Create desired email to use & name
            - Select relay IP address & final options

In-person social engineering

- Elicitation
    - Gathering info about a system from authorized users
- Interrogation
    - Informal interviews with crafted questions to extract info

- Impersonation
    - Pretending to be someone with authority
- Motivation techniques
    - Authority
    - Scarcity
    - Social proof
    - Urgency
    - Likeness
    - Fear

Network based exploits

- NETBIOS name service (NBNS)
    - Part of NetBIOS-over-TCP
- LLMNR (link-local multicast name resolution)
    - Protocol based on DNS packet format
    - Allows IPv4 and IPv6 name resolution on the same local link
- SMB (server message block) exploits
    - Protocol used in Windows to provide file and printer access, and remote service access (SAMBA in linux)
    - TCP ports 139 and 445
    - Some ransomware (eternalblue, wannacry, use SMB to propagate
- SNMP (simple network management protocol) exploits
    - Query and manage IP devices
    - Multiple versions - SNMPv1 is not secure
- SMTP (simple mail transfer protocol) exploits
    - Standard protocol for transmitting mail
    - Open relay, local relay, phishing, spam, etc.
- FTP (file transfer protocol) exploits
    - Overall insecure protocol for transferring files
    - No encryption for transfers and credentials
    - Easy for attackers to use for data exfiltration if FTP is available

DEMO - ftp exploit

- Use nmap to find FTP vulnerabilities on metasploitable
    - nmap --script vulscan --script-args vulscandb=exploitdb.csv -sV -p 21 <target>
- Choose which vulnerability to exploit, this example we will do the backdoor
- Open metasploit -- msfconsole
    - Select exploit ||| use exploits/unix/ftp/vsftpd_234_backdoor
    - > info (to verify details) > set RHOST > run

Man-in-the-middle exploits

- Additional network exploits
    - MiTM
        - Family of attacks where the attack intercepts messages between a sender and receiver
        - Attack may modify, regenerate, or forward intercepted messages

    - DNS Cache Poisoning
    - ARP Spoofing
        - Similar to DNS poisoning, but with local MAC addresses
    - Pass the hash
        - Attacker intercepts an NTLM hash (user credential) and reuses it to appear as an authenticated user to Windows
    - Replay attack
    - Relay attack
    - SSL stripping
    - Downgrade
    - DoS / stress test
    - NAC (network access control) bypass
    - VLAN hopping

Wireless Exploits

- Wireless & RF vulnerabilities
    - Broadcast is wide open
    - Aircrack-ng
- Evil twin - rogue WAP used to eavesdrop
    - Karma attack (karma attacks radio machines automatically)
    - Downgrade attack - attempt to negotiate a more insecure protocol
- Deauthentication attacks
- Fragmentation attacks
    - DoS attack, floods a network with datagram fragments
- Credential harvesting
    - Process of capturing or discovering valid login
    - Social engineering, etc.
- WPS implementation weaknesses
    - Several consumer grade WAPs could allow an attacker to learn the WPS PIN
- Bluejacking
    - Unsolicited messages to a bluetooth-enabled device
- Bluesnarfing
    - Stealing info from bluetooth-enabled device
- RFID (radio frequency identification)
    - RFID cloning - unauthorized copy of device’s RF signal
    - Jamming - DoS attack, disabled communication among devices
- Repeating
    - Receiving and re-transmitting a signal to increase range

Application Exploits

- Injection attack
    - Inserting additional data into application beyond what is expected
    - SQLi (structured query language injection)
        - Adding specially crafted SQL input to extract/modify data or execute commands
    - HTML injection

        - Adding HTML code/submitting data to change how a page works or the data is handled
    - Command injection
        - Adding command line options that change the way commands operate
    - Code injections
        - A generalization of SQLi - adding code in any language to change a program's behavior
- **DEMO - SQLi**
    - Set security to low
    - SQLmap
        - sqlmap -u <"website"> --cookie=<"cookie">
            - Get cookie from developers tools > network > headers > Cookie
        - Automates tried sqli and presents what is working
        - Can utilize metasploit as well
- Authentication:
    - Credential brute force
        - Offline cracking (hydra)
    - Session hijacking
        - -intercepting and using a session token (generally_ to take over a valid distributed (web) session
    - Redirect
        - Sending the user to a different site from what they expected (phishing)
    - Default credentials
        - Out of the box artifacts (you have to clean these up!)
    - Weak credentials
        - This is why password cracking works
    - Kerberos exploits
        - Forged tickets to allow unauthorized access to resources
- Authorization
    - Parameter pollution - providing custom input parameters to alter service/API operation
    - Insecure direct object reference
        - Programming mistake that can allow an attacker to bypass access controls and access resources or data
- Cross-site scripting (XSS)
    - Injection attack in which an attacker sends malicious code (client-side script) to a web application that a subsequent client runs
        - stored/persistent
            - Attack data (script) stored discreetly on the server
        - Reflected
            - Non-persistent attack in which attack code is sent to another client
        - DOM (document object model)
            - XSS attack that uses XML, not HTML, to transport attack code
- Cross-site request forgery (CSRF/XSRF)

    - Similar to XSS; occurs within an authenticated session
    - CSRFXSRF attacks the user, XSS attacks the server
    - Attacker can cause authorized user to take some action by clicking a link
- Clickjacking
    - Tricking user into clicking a different link or object that was intended
    - Attackers can use transparent or opaque layers to embed attack links
- Security misconfiguration
    - Directory traversal
        - Allows users to navigate outside a web server's root directory
    - Cookie manipulation
        - Access to cookies can allow an attacker to change the way in which a web application operates in general, or just for a specific user/session
    - File inclusion
        - Related to directory traversal
        - Attacker is allowed to build path to .exe file or a file to access
        - File can be local or remote

Course 4 - Select Your Attacks II

DEMO - XSS

- DVWA > XSS Reflected > Ivan <script>alert("XSS")</script>
    - Try injecting HTML: Ivan <body onload=alert("XSS")>
- XSS can allow an attacker to run almost any script code
- If successful, XSS attacks can compromise many client computers and devices
- Compromise can include remote control, data exfil, and setup for further attacks

Code vulnerabilities

- Unsecure code practices
    - Comments in source code
        - Good for developers and technical personnel
        - Bad for keeping secrets
    - Lack of error handling
        - Bad things happen - developers don't think of everything
    - Overly verbose error handling
        - Error messages can give too much info
        - Bad error message
            - "Password invalid for this user"
        - Better error message:
            - "User ID or password is invalid"
    - Hard-coded credentials
        - Happens often 0 compiled and interpreted (strings command)
    - Race conditions
        - Resource should be validated before it's used
            - E.g. checking a file is in place

        - TOC (time of check) / TOU (time of use)
            - Gap between checking a condition and using that resource
            - Attackers can influence other events and affect operation
    - Unauthorized use of functions/unprotected APIs
        - Unintended API usage
    - Hidden elements
        - HIDDEN attribute in XML and HTML (doesn't hide data in the source code)
        - Sensitive information in the DOM
    - Code signing
        - Certificates can authenticate author's identity, ensure integrity
    - Lack of code signing
        - Lack of signing allows attackers to modify code between deployment and execution

Local host vulnerabilities

- Once fingerprinting is done and OS info is gathered, research vulnerabilities on CVE database for target(s) OS(s).
- Every OS has its own specific vulnerabilities
- Online vulnerability repositories make it easy to determine which vulnerabilities apply to a specific target

Privilege escalation - LINUX

- Linux specific priv esc
    - SUID/SGID programs
        - Permissions to execute a program as executable's owner/group
        - Ls shows 's' in executable bit or permissions
            - -r-sr-sr-x (SUID and SGID set)
    - Unsecure SUDO
        - Authorized users execute commands as if logged in a root
    - Ret2libc
        - stack/buffer overflow attack
        - Replaces current stack return address with attacker-chosen address of another subroutine
        - Libc includes useful calls, such as 'system'
    - Sticky bits
        - Directory permission
        - Multiple users can create, read, and write files, but only the owner can delete
- SUID/SGID and sudo make systems easier to use, but can make them easier to compromise
- Ret2libc is a potential attack vector for hijacking processes
- Sticky bit directories can allow attackers to write files and executables

Privilege escalation - WINDOWS

- Cpassword - group policy preference attribute that contains passwords
    - SYSVOL folder of the DC (encrypted XML)

- Clear text credentials in LDAP (lightweight Directory Access Protocol)
- Kerberoasting - domain users can query Kerberos tickets for other users
- Credentials in LSASS (local security authority subsystem service)
    - Enforces security policy
- Unattended installation
    - PXE (preboot execution environment) credentials
- SAM database (security account manager)
    - Database that contains user passwords
- DLL hijacking (dynamic link library)
    - Forcing a loader to load a malicious DLL
- Cpassword and LDAP creds may contain valuable creds
- PXE creds can be used to access system as an authorized user
- DLL hijacking is an attack vector that could allow an attacker to load malware

Miscellaneous privilege escalation

- Exploitable services
    - Unquoted service paths
        - Allow abbreviated attack paths (without spaces)
    - Writable services
        - Allow attacker to replace services with malicious programs
- Unsecure file/folder permissions - root installs allow read/write by any user
- Keylogger - records keystrokes
- Scheduled tasks
    - Attacker may add new task to run persistently with elevated privileges
- Kernel exploits
    - Unpatched systems are vulnerable
- Unquoted service paths and writable services can allow for service exploits
- Look for files and folders that allow excessive read/write permissions
- Footprinting can provide information on kernel vulnerabilities

Miscellaneous local host vulnerabilities

- Default account settings - disable accounts that are not being used
- Sandbox escape
    - Shell upgrade - gaining access to a shell with higher privilege
    - VM - escaping a VM may allow access to underlying environment
    - Container - similar to VM escape (i.e. Docker)
- Physical device security
    - Cold boot attack
        - Ability to physically reboot a system (can allow access to encryption keys)
    - JTAG debug (joint test action group)
        - Can allow attacker to interact with chips
    - Serial console
        - If not disabled, provides direct access to servers
- Look for easy attack paths - admins may have overlooked something

Physical security

- piggybacking/tailgating - unauthorized person following an authorized person through a physical control
- Fence jumping - physically bypassing a control
- Dumpster diving - looking through trash for useful information
- Lock picking - opening a lock without a proper key
- Lock bypass
    - Defeating a lock mechanism without picking (i.e. bolt cutter, remove hinges)
- Egress sensor
    - Senses a person approaching a door to leave a facility
- Badge cloning
    - Copying an RFID badge

Post-exploitation techniques

- What do you do once you're in
    - Make it easier next time
- Lateral movement
    - RPC/DCOM (remote procedure call / distributed component object model)
        - PsExec - utility that supports executing processes on other systems (e.e. telnet)
        - WMI (windows management instrumentation) - managing devices and applications from remote computers
        - Scheduled tasks
- Ps remoting / WinRM
    - Powershell remoting / windows remote management
- SMB (server message block)
    - Protocol for exposing shares to remote computers (Linux too)
- RDP
    - Ability to access a desktop from a remote computer
- Apple remote desktop
    - Apples RDP
- VNC (virtual network computing)
- X-server forwarding
    - X-windows access to Linux desktop
- Telnet
    - Unsecure remote access (everything in cleartext)
- SSH (secure shell)
    - More secure remote access to shell
- RSH / Rlogin (remote shell / remote login)
    - Legacy secure remote access
- Enable remote access if possible
- Use remote access to move laterally within a network

Persistence & stealth

- Persistence
    - Scheduled jobs
        - Cron or Task Manager

    - Scheduled task
        - Same as above
    - Daemons
        - Background processes or services
    - Backdoors
        - Bypass standard security controls
    - Trojan
        - Malware that looks like it does something useful
    - New user creation
        - Makes later logins easier
- Stealth
    - Clean up files, including tools installed
    - Hiding files that you need to leave
    - Sanitizing log files (remove entries or entire logs)
    - Remove any traces of activity while accessing the environment
- Set up persistent processes to maintain a presence
- Install low profile tools and malware to make your job easier
- Leave artifacts that keep the attack going and make it easier to get back in
- Once the attack is over, clean up to avoid post-mortem detection

Course 5 - Selecting Pentesting Tools

Nmap scoping & output options

- One of the most common and most useful tools for reconnaissance
- Cheatsheet: https://www.stationx.net/nmap-cheat-sheet/
- SYN (stealth) scan
    - Sends SYN packet and examines response (SYN/ACK means the port is open)
    - If SYN/ACK received, nmap sends RST to terminate the connection request
- Full connect scan
    - nmap -sT target
    - Completes the handshake steps to establish a connection (more reliable)
- Service identification (-sV)
    - Nmap -sV <target>
    - Attempts to determine services running info
- OS fingerprinting (-O)
    - Detects target OS
    - Nmap -O <target>
- Disabling ping (-Pn)
    - Skips host discovery (assumes all are online)
    - nmap -Pn <target>
- Target input file (-iL)
    - Uses a text file that contains a list of targets
        - Nmap -iL <input file name>

Pentesting toolbox

- Use case: reconnaissance
  - Nmap
  - Whois
  - Nslookup
  - Theharvester
  - Shodan
  - recon-NG
  - Censys
  - aircrack-NG
  - Kismet
  - Wifite
  - SET
  - Wireshark
  - Hping
  - MSF
- Use case: enumeration
  - Nmap
  - Nslookup
  - Wireshark
  - Hping
- Use case: vulnerability scanning
  - Nmap
  - Nikto
  - openVAS
  - SQLmap
  - Nessus
  - W3AF
  - OWASP ZAP
  - MSF
- Use case: credential attacks
  - Offline cracking:
    - Hashcas
    - JtR
    - Cain and abel
    - Mimikatz
    - Aircrack-ng
  - Brute forcing services (online approach)
    - SQLmap
    - Medusa
    - Hydra
    - Cain and abel
    - Mimikatz
    - Patator
    - W3AF
    - Aircrack-ng
- Use case: persistence (once you have exploited a target, use these to make sure you can get back in
  - SET
  - BeEF
  - SSH
  - NCAT
  - NETCAT
  - Drozer
  - Powersploit
  - Empire
  - MSF
- Use case: configuration compliance (to evaluate a config to determine if it's compliant with a standard of regulation):
  - Nmap
  - Nikto
  - OpenVAS
  - SQLmap
  - Nessus
- Use case: evasion
  - SET

    - Proxychains
    - MSF
- Use case: decompilation (to decompile executables)
    - Immunity debugger
    - APKX
    - APK studio
- Use case: debugging
    - OLLYDBG
    - Immunity debugger
    - GDB
    - Win DBG
    - IDA
- Use case: software assurance:
    - Findsecbugs
    - Sonarqube
    - YASCA
- Use case: fuzzing
    - Peach
    - AFL

Scanners and credential tools

- Scanners are “meta” tools that provide several levels of output
- Scanners are powerful, but very noisy and using them risks being detected
- Credential cracking tools run either in online or offline modes
- Effective dictionary attacks depend on good user/password lists

Code cracking tools

- Debuggers are advanced tools and can reveal how a program works
- Debuggers can also allow testers to modify data as the program is running
- Software assurance tools can help to identify vulnerabilities in applications

Open source research tools (OSINT)

- Whois - domain details (contacts, name servers, etc)
- Nslookup - DNS info
- Foca - fingerprinting organizations with collected archives
- Theharvester - gathers info from many sources (email, hosts, open ports, etc)
- Shodan - finds internet connected devices
- Maltego - data mining for investigations
- Recon-NG - web recon
- Censys - similar to shodan

Web pentesting tools

- Web proxies
    - OWASP ZAP - zed attack proxy - web app security scanner
    - Burp Suite - graphical tool for testing web app security

Remote access tools

- SSH - secure shell
- NCAT - similar to nc, but from Nmap devs

- NETCAT - same as nc
- Proxychains - forces TCP connections through a proxy
- Bind shell, reverse shell
- Remote access is often followed by priv esc attacks and/or preceded by credential attacks

Other pentesting tools

- Powersploit - post exploitation framework (PS PowerShell)
- Responder - microsoft network poisoner
- Impacket - python classes for working with network protocols
- Empire - powershell/python post-exploitation agent

Course 6 - Using Scripting

Using Scripting in Pentesting

- Automate tasks, repeatable
- What is a script?
  - Interpreted sequence of commands
  - Written in a specific language with its own syntax
  - Easy to code
- Resources:
  - Bash
    - https://github.com/awesome-lists/awesome-bash
    - https://www.commonexploits.com/penetration-testing-scripts/
    - https://github.com/averagesecurityguy/scripts
    - https://github.com/bitvijays/Pentest-Scripts
  - Powershell
    - https://www.businessnewsdaily.com/10760-best-free-powershell-training-resources.html
    - https://blog.netwrix.com/2018/02/21/windows-powershell-scripting-tutorial-for-beginners/
  - Ruby
    - https://learnrubythehardway.org/book/
    - http://ruby-for-beginners.rubymonstas.org/index.html
  - Python
    - https://lectures.quantecon.org/py/
- Need to know for scripting
  - Variables
    - Temporary data storage
  - Substitutions
    - Input parameters and environment variables
  - Common operations
    - Strings and comparisons
  - Logic

        - Looping and flow control
    - Basic I/O
        - Read input and write output (file, terminal, and network)
    - Error handling
        - When things don't work
    - Arrays
        - Simple data structure
    - Encoding/decoding
        - Handling special characters

Bash Scripting

- Bash is the default shell in Linux
- Bash makes it easy to combine multiple commands that can react to input
- Learn basic loops and conditional logic
- A few lines of a bash script can automatically execute many commands, such as scans

Powershell Scripting

- SetExecutionPolicy Unrestricted (allows you to run powershell scripts on machine, disabled by default)

Ruby Scripting

- Powerful object-oriented language that can do far more than just scripting
- Ruby's popularity is related to the Ruby on Rails server-side web app framework written in Ruby
- Ruby treats everything as an object and relies heavily on methods and attributes

Python Scripting

- Python is a popular language because it's easy to write very powerful programs in just a few lines of code
- Unlike many other languages, python depends on indentation to define blocks

Scripting Languages Comparison

	Bash	PowerShell	Ruby	Python
Comments	#	# or <# #>	# or =begin =end	#
Variables – assign	varName=value	$varName=value	varName=value	varName=value
Variables – display	echo $varName	Write-Host $varName	puts varName	print(varName)
Substitution – environment variables	$envVarName	Get-item Env:varName	ENV['varName']	Os.environ['varName']

	Bash	PowerShell	Ruby	Python
String length	${#string}	(string).Length	string.length	len(string)
String – substring	${string:position}	(string).Substring(start,end)	string[1..3]	string[start:end+1]
String – replace substring	${string/substring/replacement}	(string).Replace(substr,replStr)	string[1..3] = replStr	string.replace(old, new, count)
AND/OR	-a / -o	-and, -or, -not !	and &&, or ||, not !	and, or, not
Comparisons	-eq (==), -ne (!=), -lt (<), -le (<=), -gt (>), -ge (>=)	-eq, -ne, -gt, -ge, -lt, -le	==, !=, >, >=, <, <=	==, != (<>), >, >=, <, <=

	Bash	PowerShell	Ruby	Python
Looping	For	For, While, Do-While, Do-Until	while, until, for	for, while

Flow control	if condition
	then
	  commands
	elif
	  commands
	else
	  commands
	fi
		if (condition) {
		  statements
		} elseif (condition) {
		  statements
		} else {
		  statements
		}
			If condition then
			  statements
			elsif
			  statements
			else
			  statements
			end
				if condition:
				  statements
				elif condition:
				  statements
				else:
				  statemenst

Course 7 - Reporting & Communication

Writing reports

- Communicate findings AND recommendations
- Primary deliverable
- Only chance to make your points
- Digest of all activities and conclusions
    - Some conclusions are drawn during tests
    - Some result from post-test analysis
- Sample resource reports:
    - www.pentest-standard.org
    - https://github.com/juliocesarfort/public-pentesting-reports
    - https://www.offensive-security.com/reports/penetration-testing-sample-report-2013.pdf
- TIPS:
    - Tell your story
    - Know your audience(s)
        - Technical, management?
    - Leave the reader with a call to action
        - Includes steps to fix issues

    - Try to answer these questions
        - What did you ?
        - Why did you make the choices you made?
        - What did you find, and how did your findings affect your conclusions?

Post report activities

- Remove all test activity artifacts
- Get formal client acceptance
- Conduct “lessons learned” sessions with client and capture the findings
- Follow up on client add-on requests

Mitigation strategies

- Recommend mitigation activities for each identified vulnerability
- Suggest different classes of mitigations (technical, administrative, etc)

Communication

- Good communication is critical to pentest project success
- Managing communication expectations, including frequency, reduces conflict
- Define triggers that initiate communication