Service Account Attack: LDAP Reconnaissance with PowerShell

blog.netwrix.com/2022/08/31/discovering-service-accounts-without-using-privileges

Jeff Warren

In the introductory post of this series, we reviewed what an Active Directory (AD) service account is, explained why these privileged accounts are a serious security risk, and promised to detail 4 types of attacks on service accounts in future posts. This post explores the first of those attacks: LDAP reconnaissance, which attackers can use to discover service accounts in an IT environment while avoiding detection.

Handpicked related content:
[On-demand Webinar] 4 Service Account Attacks and How to Protect Against Them

Specifically, in this post, we will explore how attacker with no special permissions, such as Domain Admin or even local Administrator rights, can use simple PowerShell queries to find service accounts, without having to install or learn specialized tools like Bloodhound.

Finding privileged accounts

Active Directory offers many security and management benefits, but also can make things a little too easy for a curious attacker. Due to the architecture of AD, once an attacker has infiltrated any domain-joined computer, they are able to query the directory and its objects. And because by default Active Directory does not provide a mechanism to audit and alert on the suspicious activity, they can often avoid detection.

Handpicked related content:
[Free Guide] Active Directory Security Best Practices

Here are some of the ways an attacker can discover service accounts by querying LDAP.

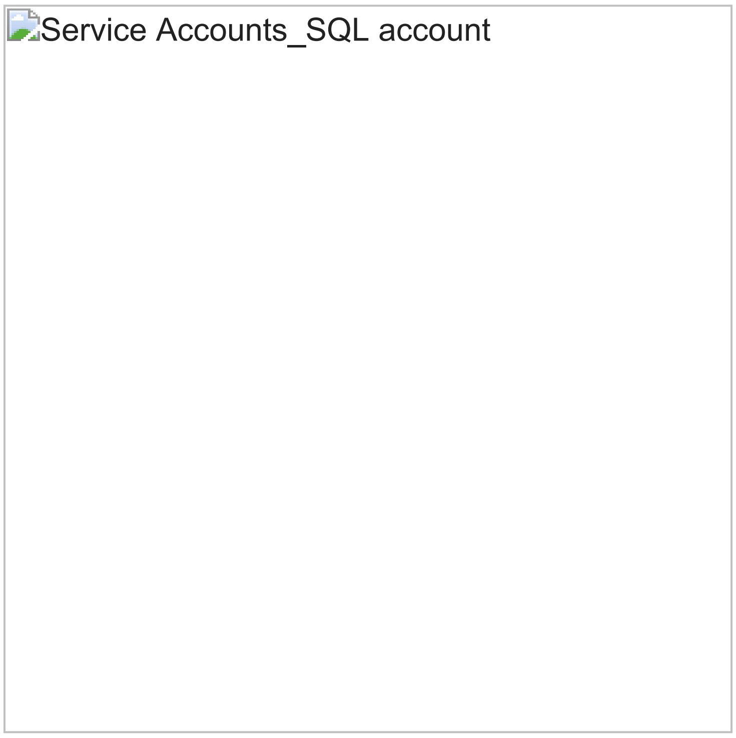
Service Principal Name (SPN) discovery

Service accounts leverage SPNs to support Kerberos authentication. While this provides improved security, it also leaves a trail of exactly where these accounts are used and what they are used for. This information can be easily exploited by an attacker. SPNs are commonly used to run services to support applications like Microsoft SQL Server and SharePoint.

In another blog post, we demonstrate how to perform advanced AD reconnaissance; however, there are simpler ways to get the information we need for our purposes here. Using the following simple LDAP query, an attacker can get a list of AD accounts that have registered SPNs, as well as the computers, applications and data they provide access to:

```
#Build LDAP filters to look for users with SPN values registered for current
domain
$ldapFilter = "(&(objectclass=user)(objectcategory=user)(servicePrincipalName=*))"
$domain = New-Object System.DirectoryServices.DirectoryEntry
$search = New-Object System.DirectoryServices.DirectorySearcher
$search.SearchRoot = $domain
$search.PageSize = 1000
$search.Filter = $ldapFilter
$search.SearchScope = "Subtree"
#Execute Search
$results = $search.FindAll()
#Display SPN values from the returned objects
foreach ($result in $results)
{
    $userEntry = $result.GetDirectoryEntry()
    Write-Host "User Name = " $userEntry.name
    foreach ($SPN in $userEntry.servicePrincipalName)
    {
        Write-Host "SPN = " $SPN
    }
    Write-Host ""
}
```

The SPN values in the results will reveal where each account is registered and what service it is registered for on that system. For example, here is the SPN value for a SQL service account:



Service account discovery using generic object attributes

While SPNs are very reliable and informative, they will not produce a comprehensive list of service accounts because many accounts do not integrate with Kerberos through SPNs and will have no SPN values set. However, attackers with no domain rights can often still discover service accounts through other means. In particular, most organizations use naming conventions, such as having all service account names start with “SVC” or something similar, and service accounts are typically placed into their own organizational units (OUs) or groups.

Finding service accounts based on account naming conventions

Here is a PowerShell script that will find all accounts that contain “svc” in the name:

```
#Build LDAP filter to find service accounts based on naming conventions
$ldapFilter = "(&(objectclass=Person)(cn=*svc*))"
$domain = New-Object System.DirectoryServices.DirectoryEntry
$search = New-Object System.DirectoryServices.DirectorySearcher
$search.SearchRoot = $domain
$search.PageSize = 1000
$search.Filter = $ldapFilter
$search.SearchScope = "Subtree"

#Add list of properties to search for
$objProperties = "name"
Foreach ($i in $objProperties){$search.PropertiesToLoad.Add($i)}

#Execute Search
$results = $search.FindAll()
#Display values from the returned objects
foreach ($result in $results)
{
    $userEntry = $result.GetDirectoryEntry()
    Write-Host "User Name = " $userEntry.name
    Write-Host ""
}
```

Finding service accounts based on OU

And here is an LDAP filter that can be substituted into the preceding script to find any OUs that contain “Service” or “svc” in the name:

```
$ldapFilter = "(&(objectClass=organizationalUnit)(|name=*Service*)(name=*svc*)))"
```

The attacker can then search those OUs for all user objects to find your service accounts. This script focuses on one OU with “Service” in its name:

```
$search = New-Object System,DirecoryServices.DirectorySearcher
$search.SearchRoot = "LDAP://OU=Service Accounts,OU=JEFFLAB,DC=local"
$search.PageSize = 1000
$search.Filter = $ldapFilter
$search.SearchScope = "Subtree"
```

Service account discovery via user account control

Another sneaky way to search Active Directory for service accounts is to investigate account control settings, since service accounts often have different settings than regular user accounts. The best example of this is the “password never expires” setting —

service accounts may have passwords set to never expire because the act of resetting them can be tedious and result in application or service outages.

Service Accounts_Account options

With PowerShell, it is possible to find all accounts with this value enabled, using a slightly more complicated LDAP filter:

```
$ldapFilter = "(&(objectclass=user)(objectcategory=user)
(useraccountcontrol:1.2.840.113556.1.4.803:=65536))"
```

Privileged discovery

While this post focused solely on ways to find service accounts without leveraging any privileges, if an attacker does find an account that has privileges on one or more systems within the network, there are many other effective ways to discover service accounts. They include:

- Enumerating services on endpoints and extracting the startname account
- Searching for connection strings in web.config, scripts and other locations where service accounts may be hard-coded

- Exploring the local policy for users granted the “Log on as a Service” right
- Extracting event logs for non-interactive logon types
- Finding application pool and other web application service accounts

Exploiting the discovered accounts

Once an adversary has a list of service accounts, their next step is to exploit them. Discover some of their options in the following posts:

Learn more about LDAP reconnaissance and other attack techniques in the Netwrix attack catalog.

Jeff Warren