

Forum:[Hack Forums](#)

Thread: [Advance Active Directory Enumeration and Recon](#)

[BLANK] - January 29, 2018, 2:06 pm

Red Team Active Directory Enumeration Techniques

While there are several well known and older techniques for performing reconnaissance in an active directory, this article will focus on newer techniques that modern Red Teams utilize. These techniques are aimed at being stealthy by leveraging native Windows/Active Directory functionalities and components.

Instead of focusing on zero-day exploits that may crash a box or cause loud noise on a network, we will focus on identifying misconfigurations or exploiting trust relationships to eventually gain domain administrator.

We will begin with User Hunting. The two tools we will use for this enumeration include:

[PowerView](#)

[AD PowerShell Module](#) - which should be installed on a compromised host with an elevated shell.

We can begin enumeration by performing DNS lookups using LDAP. With this method we could ask for a list of specific computers or all the DCs and the associated IP addresses through just an LDAP call.

To identify machines inside the domain:

```
Code >> get-adcomputer -filter * -Properties ipv4address | where {$_.IPv4address} | select name, ipv4address
```

Do a reverse lookup via LDAP:

```
Code >> get-adcomputer -filter {ipv4address -eq 'IP'} -Properties  
Lastlogondate,passwordlastset,ipv4address
```

Next we will find enterprise services using SPN scanning. SPN scanning uses LDAP queries which use and look for Service Principal Names. This lets us know which services are running that support Kerberos authentication, no need for port scanning!

Some SPN types include MSSQLSvc, TERMSERV, WSMAN, and exchangeMDB. This allows us to find all the SQL servers easily and the format has the SPN type, server name, and SQL often has a port number or an instance at the end.

[SPN Directory List](#)

A great automated tool for enumeration using SPN scanning can be found here:

[Find-PSServiceAccounts](#)

For manual testing we can use the following PowerShell command:

```
Code >> Get-ADComputer -filter {ServicePrincipalName -Like "*SPN*"} -Properties  
OperatingSystem,OperatingSystemVersion,OperatingSystemServicePack, PasswordLastSet,
```

LastLogonDate, ServicePrincipalName, TrustedForDelegation, TrustedtoAuthForDelegation
Next we will find all the group policies in the organization that authenticated users have read access, which is all of them by default. What we are looking for in the group policies include:

- + PowerShell logging policy
- + Full auditing policy
- + Configurations like "prevent local account at logon", "add server admin to local administrator group",
- + EMET config,
- + AppLocker config
- + And more...

To discover all the group policies in the domain we would use the following PowerView command.

```
Code >> Get-NetGPO | select displayname,name,whenchanged
```

Now onto the User Hunting! We want to be able to identify users, where they have privileges etc and we want to be able to do it as an unprivileged user.

PowerView leverages a couple of native API calls to perform this task. One of the API calls commonly used is Add-Type but we will avoid it because it is not fileless. To be stealthy we must avoid writing files onto hosts as much as possible. Instead we will utilize straight reflection for API interaction through PowerShell, see <https://github.com/mattifestation/PSReflect> for more information.

An example of using PowerView to find Domain Admins:

```
Code >> Get-NetGroupMember 'Domain Admins' -Recurse
```

A more advance method of performing User Hunting with PowerView is to use:

```
Code >> Invoke-UserHunter -Stealth -ShowAll
```

This will enumerate all the distributed file systems and DCs and pulls all user objects, script path(s), home directories etc. It will pull certain type of fields that tend to map where file server are and a user AD schema. The stealth factor is it will target computers that are heavily trafficked so our queries will not be unusual. It will then perform a Get-NetSession against those systems. The sessions of those systems can give us about 70% mapping of the network, due to their heavy traffic.

The primary difference between using the -Stealth feature and not using it is the amount of systems we are sending queries to. This will greatly reduce our overall network fingerprint.

Last, we will look at how to query a single domain controller's Global Catalog to get all the users of an AD forest. We don't even need Administrator privileges.

I recommend the following powershell script to perform the query:

[Retrieve-All-Users-from-AD](#)

After running the above script we should see AD user information including host name, IP location, domain, forest, roles and more.

The techniques above will assist a penetration tester who has compromised a Windows computer on a domain. Once we have compromised the host we can search for permissions of the current users of the host we are on, map to other hosts with our privileges, search for computers that a higher privileged user is on and how to eventually access that computer and gain domain admin credentials. I provided a summary of the different methods but more research will be needed and practice required, to really understand the process.

sulphur - January 29, 2018, 3:09 pm

Thanks for this tutorial. Good job

CyberStrike - January 30, 2018, 1:25 am

Well done EasyGhost I notice you like PowerTools/PowerView quite a bit . We should be glad that an average user account on the Domain can grab up a ton of info on the AD (users/OUs/Groups/) and then relationships so we can cut down in the user hunting phase. What are your feelings on using BloodHound btw?

[BLANK] - January 30, 2018, 5:42 am

(Jan 30, 2018 02:25 AM)sirru5h Wrote: [\[CITE\]](#) Well done EasyGhost I notice you like PowerTools/PowerView quite a bit . We should be glad that an average user account on the Domain can grab up a ton of info on the AD (users/OUs/Groups/) and then relationships so we can cut down in the user hunting phase. What are your feelings on using BloodHound btw?

PowerShell is great for what it does. It is essentially Bash for Windows and can be highly weaponized. Bloodhound is amazing, the techniques I wrote about are only part of what Bloodhound does. I have used Bloodhound in a large network, hundreds of hosts and servers etc. And after compromising just one host on the domain I was able to get a Path to Enterprise Admin. What could have taken days or more, took only a couple hours.

HackRain - January 30, 2018, 6:06 am

Great share.

Good to see fellow testers using PS.

I've been using pshell for the last couple of years, both writing blueteam tools for new client onboardings as well as red team privEsc tools for my upcoming OSCP exam.

Have you had much experience with powersploit / Empire?

[BLANK] - January 30, 2018, 7:14 am

(Jan 30, 2018 07:06 AM)HackRain Wrote: [\[CITE\]](#) Great share.

Good to see fellow testers using PS.

I've been using pshell for the last couple of years, both writing blueteam tools for new client onboardings as well as red team privEsc tools for my upcoming OSCP exam.

Have you had much experience with powersploit / Empire?

Nice to see a purple teamer on HF! Yea I have used Empire quite a bit, it is what I use if I can't use Cobalt Strike which is more often then not these days. You can even use PowerShell on Linux and Mac if you wanted, ive messed around with it but Bash is still preferable.

As for OSCP writing custom priv escalations may be a bit over board. The priv escalations you will need will all be pre-made standard stuff, most of it pretty old. The point of the OSCP is to show your general knowledge and your ability to research. I passed on my first try, got 7/8 of the flags. Would have got the final flag but the exploit I used to get into the box just wouldn't work again, no matter how many times I reset it. My advice is to go through an initial enumeration, get past the exploit development box (don't forget to look for bad characters) and then decide which Windows box you want to use Metasploit on. And yes, use Metasploit on one box, no need to hamstring yourself. I was able to quickly pop one of the medium level boxes with Metasploit and probably saved me a few hours. Don't waist it on the low point box, those are usually very easy.

Anyways good luck!

Baud - February 1, 2018, 3:00 am

Your threads are making me want to learn powershell more seriously, it definitely is a powerful resource. Also very interesting methods, I just saved this for further reading.

[BLANK] - February 1, 2018, 9:05 am

(Feb 1, 2018 04:00 AM)Baud Wrote: [\[CITE\]](#) Your threads are making me want to learn powershell more seriously, it definitely is a powerful resource. Also very interesting methods, I just saved this for further reading.

PowerShell is 100% worth learning. I am really only just discovering all the great tools and functions that you can do with PowerShell. Yes, Windows 10 has some new security features that limit PowerShell functionality more then past versions, but you can still do a lot with PowerShell, and there are ways of bypassing some of those security features.

CyberStrike - February 1, 2018, 8:12 pm

You know I kind of converted my everyday machine with a few tools for some testing anyhow I ran into TrustVisualizer it takes the results of the AD enumeration csv output and then using trustvisualizer and yed-3.17.2, it creates a diagram.

<https://github.com/HarmJ0y/TrustVisualizer>

Edit: Link

[BLANK] - February 1, 2018, 9:04 pm

Nice, so similar to Bloodhound.

CyberStrike - February 1, 2018, 11:06 pm

Same idea using different things though since in Bloodhound everything is uploaded to neo4j after shooting the powershell script through here it's formatted using python then you use yED to display the results, Now that I think of it, should probably touch on KRBTGT account tons of admins leave it and the tickets are all signed with it's hash, grabbing it allows guys to start writing their own tickets.