Azure Sentinel and Sysmon 4 B!ue T3amer$

eshlomo.us/azure-sentinel-and-sysm0n-4-blue-teamers/

Elli Shlomo March 10, 2021

Recently, there have been massive cyberattacks against cloud providers and on-premises environments, the most recent of which is the attack and exploitation of vulnerabilities against Exchange servers – The HAFNIUM. This post focus on Azure Sentinel and Sysmon 4 B!ue T3amer$.

Recent attacks require us to increase attention alongside tools to provide us with advanced visibility and investigative options. The recent attack on Exchange servers has shown that the richer information we have, the more advanced investigation we can achieve.

Event Viewer alone cannot provide us the relevant information. We must expand how we collect logs and if it is advanced event log management or PowerShell advanced logging and others.

In the recent blog post, we saw how PowerShell advanced logging could provide us useful information. With this blog post, we can see how Sysmon can offer more capabilities to the incident response with Azure Sentinel.

Sysmon in a nutshell

Sysinternal System Monitor (Sysmon) is a Windows system service and device driver that remains resident across system reboots to monitor and log system activity to the Windows event log once installed on a system. It provides detailed information about process creations, network connections, and changes to file creation time.

By collecting the events it generates using Windows Event Collection or SIEM agents and subsequently analyzing them, you can identify malicious or anomalous activity and understand how intruders and malware operate on your network.

More about Sysmon – Windows Sysinternals | Microsoft Docs

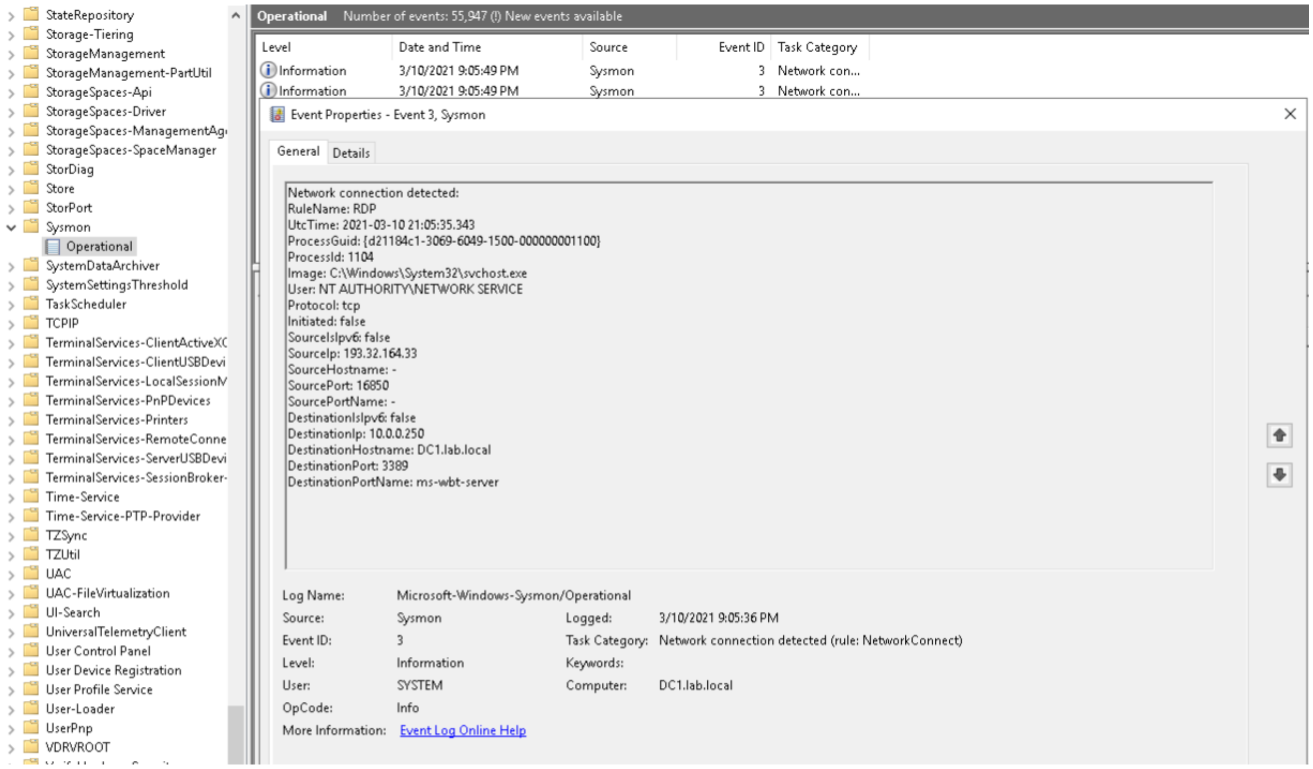
What are the Capabilities of Sysmon? In a nutshell, readable and useful process information. You can get valuable details that are not found in the raw Windows log, but most significantly, these fields, for example:

- Process id
- Parent process-id
- File image names
- Hash of file image
- Process command line

- Parent process command line

Sysmon installs as a device driver and service. Its key advantage is that it takes log entries from multiple log sources, correlates the information, and places the resulting entries into one folder in the Event Viewer.

For example, this particular command line should trigger some suspicions. Using cmd.exe to run another command, then while redirecting the output to a strangely named file, is the stuff of some C2. It's a way to create a shell using specific services.



Operational Number of events: 55,947 (!) New events available

Level	Date and Time	Source	Event ID	Task Category
Information	3/10/2021 9:05:49 PM	Sysmon	3	Network con...
Information	3/10/2021 9:05:49 PM	Sysmon	3	Network con...

Event Properties - Event 3, Sysmon

```
Network connection detected:
RuleName: RDP
UtcTime: 2021-03-10 21:05:35.343
ProcessGuid: {d21184c1-3069-6049-1500-000000001100}
ProcessId: 1104
Image: C:\Windows\System32\svchost.exe
User: NT AUTHORITY\NETWORK SERVICE
Protocol: tcp
Initiated: false
SourceIsIpv6: false
SourceIp: 193.32.164.33
SourceHostname: -
SourcePort: 16850
SourcePortName: -
DestinationIsIpv6: false
DestinationIp: 10.0.0.250
DestinationHostname: DC1.lab.local
DestinationPort: 3389
DestinationPortName: ms-wbt-server
```

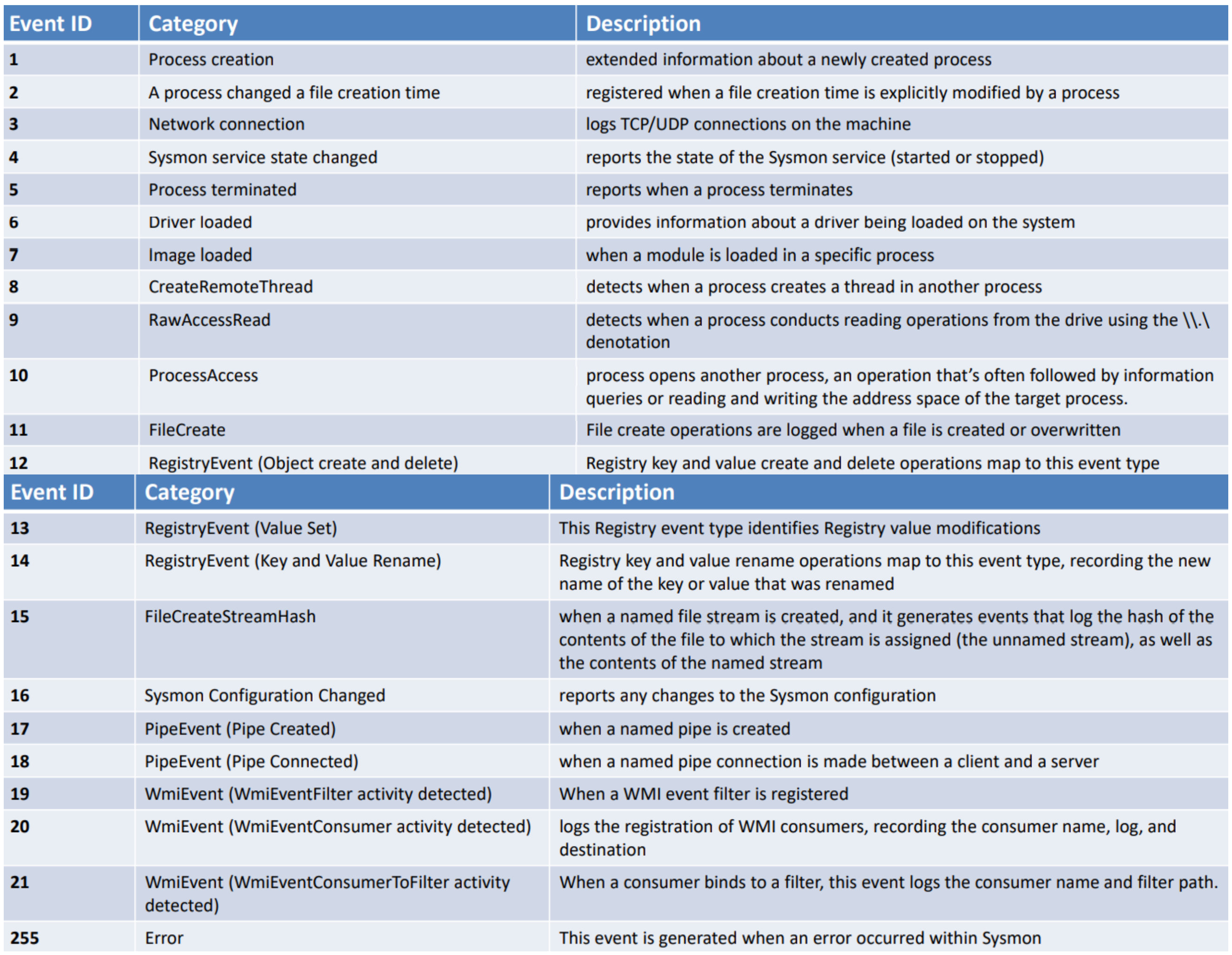
Log Name:	Microsoft-Windows-Sysmon/Operational		
Source:	Sysmon	Logged:	3/10/2021 9:05:36 PM
Event ID:	3	Task Category:	Network connection detected (rule: NetworkConnect)
Level:	Information	Keywords:	
User:	SYSTEM	Computer:	DC1.lab.local
OpCode:	Info		
More Information:	Event Log Online Help		

Sysmon Highlights

- Sysmon includes the ability to filter events before they are written to the Event Log.
- You can build (or download) configuration files.
- They make it easier to deploy a preset configuration and filter captured events.
- You can log network events from processes named “specific.exe” or locate in C:\Temp, drivers not signed by Microsoft, etc.
- It's up to you to determine how much data you want to include.
- Sysmon configuration file
  - install: sysmon -i -accepteula c:\SysmonConfig.xml
  - update: sysmon -c c:\SysmonConfig.XML
  - use Psexec or PowerShell during an IR
- Each event is specified using its tag

- To see all tags, dump the full configuration schema:
    - sysmon -s
    - on the match can be “include” or “exclude.”
    - Include and exclude refer to filter effect

Sysmon Event ID Numbers



Event ID	Category	Description
1	Process creation	extended information about a newly created process
2	A process changed a file creation time	registered when a file creation time is explicitly modified by a process
3	Network connection	logs TCP/UDP connections on the machine
4	Sysmon service state changed	reports the state of the Sysmon service (started or stopped)
5	Process terminated	reports when a process terminates
6	Driver loaded	provides information about a driver being loaded on the system
7	Image loaded	when a module is loaded in a specific process
8	CreateRemoteThread	detects when a process creates a thread in another process
9	RawAccessRead	detects when a process conducts reading operations from the drive using the \\.\ denotation
10	ProcessAccess	process opens another process, an operation that’s often followed by information queries or reading and writing the address space of the target process.
11	FileCreate	File create operations are logged when a file is created or overwritten
12	RegistryEvent (Object create and delete)	Registry key and value create and delete operations map to this event type
13	RegistryEvent (Value Set)	This Registry event type identifies Registry value modifications
14	RegistryEvent (Key and Value Rename)	Registry key and value rename operations map to this event type, recording the new name of the key or value that was renamed
15	FileCreateStreamHash	when a named file stream is created, and it generates events that log the hash of the contents of the file to which the stream is assigned (the unnamed stream), as well as the contents of the named stream
16	Sysmon Configuration Changed	reports any changes to the Sysmon configuration
17	PipeEvent (Pipe Created)	when a named pipe is created
18	PipeEvent (Pipe Connected)	when a named pipe connection is made between a client and a server
19	WmiEvent (WmiEventFilter activity detected)	When a WMI event filter is registered
20	WmiEvent (WmiEventConsumer activity detected)	logs the registration of WMI consumers, recording the consumer name, log, and destination
21	WmiEvent (WmiEventConsumerToFilter activity detected)	When a consumer binds to a filter, this event logs the consumer name and filter path.
255	Error	This event is generated when an error occurred within Sysmon

Install Sysmon

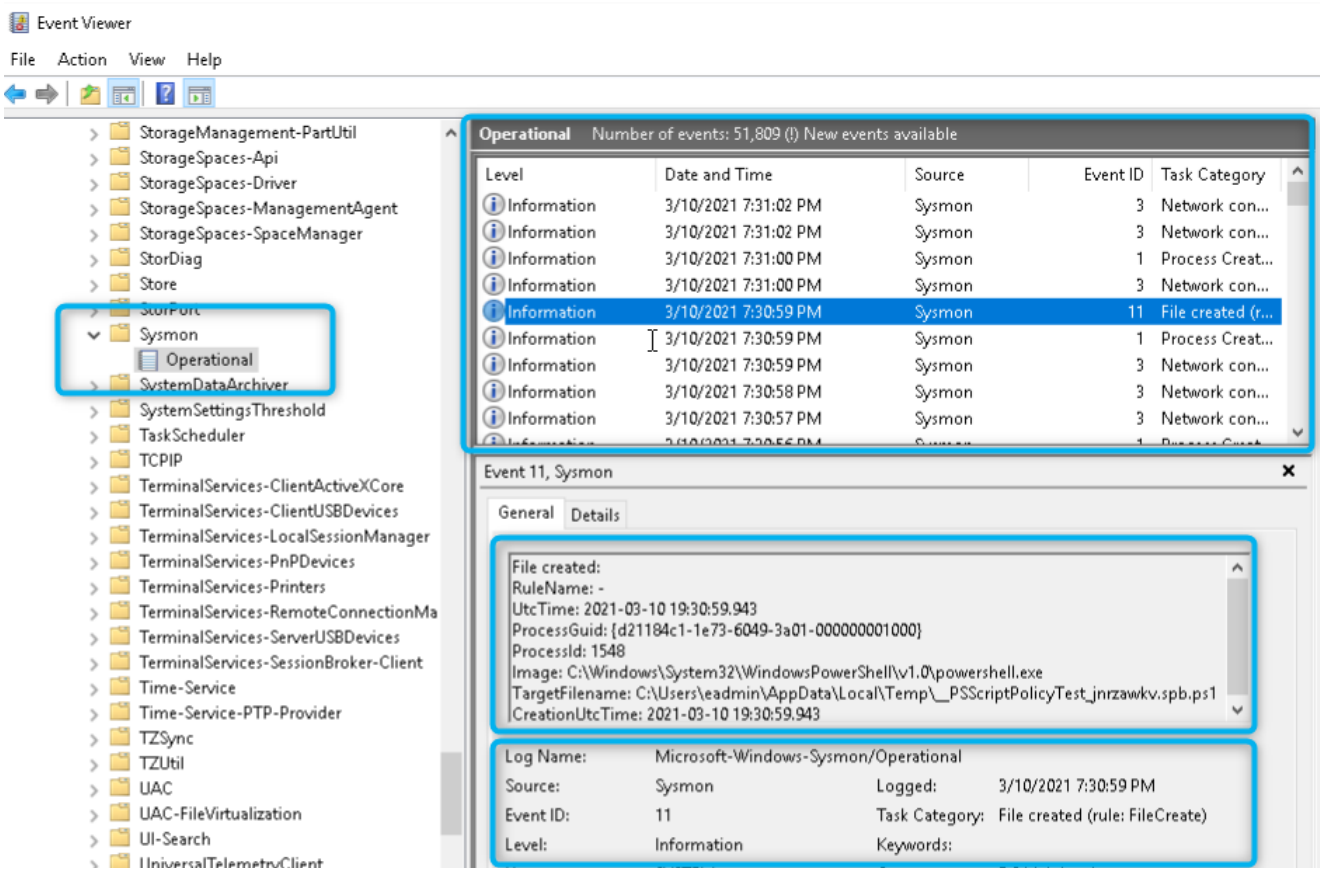
The first thing is to install Sysmon on relevant servers, such as Domain Controllers, Exchange servers, Application server, and whether it’s on a VM on Azure or any cloud provider. Once Sysmon is installed, we can configure Azure Sentinel to collect information from the relevant servers.

The way to achieve Sysmon with Azure Sentinel is straightforward and can be done by the following actions.

- Download Sysmon and install it on the relevant servers
- Make sure the Sysmon services are up and running and writing logs to the event viewer.

- Make sure to update the configuration of an installed Sysmon with the command: Sysmon64.exe -c c:\Windows\sysmonconfig.XML

TIP: Download the Sysmon config file from here and monitor additional apps and exe


Event Viewer
File
Action
View
Help
StorageManagement-PartUtil
StorageSpaces-Api
StorageSpaces-Driver
StorageSpaces-ManagementAgent
StorageSpaces-SpaceManager
StorDiag
Store
StorPort
Sysmon
Operational
SystemDataArchiver
SystemSettingsThreshold
TaskScheduler
TCPIP
TerminalServices-ClientActiveXCore
TerminalServices-ClientUSBDevices
TerminalServices-LocalSessionManager
TerminalServices-PnPDevices
TerminalServices-Printers
TerminalServices-RemoteConnectionMa
TerminalServices-ServerUSBDevices
TerminalServices-SessionBroker-Client
Time-Service
Time-Service-PTP-Provider
TZSync
TZUtil
UAC
UAC-FileVirtualization
UI-Search
UniversalTelemetryClient
General
Details

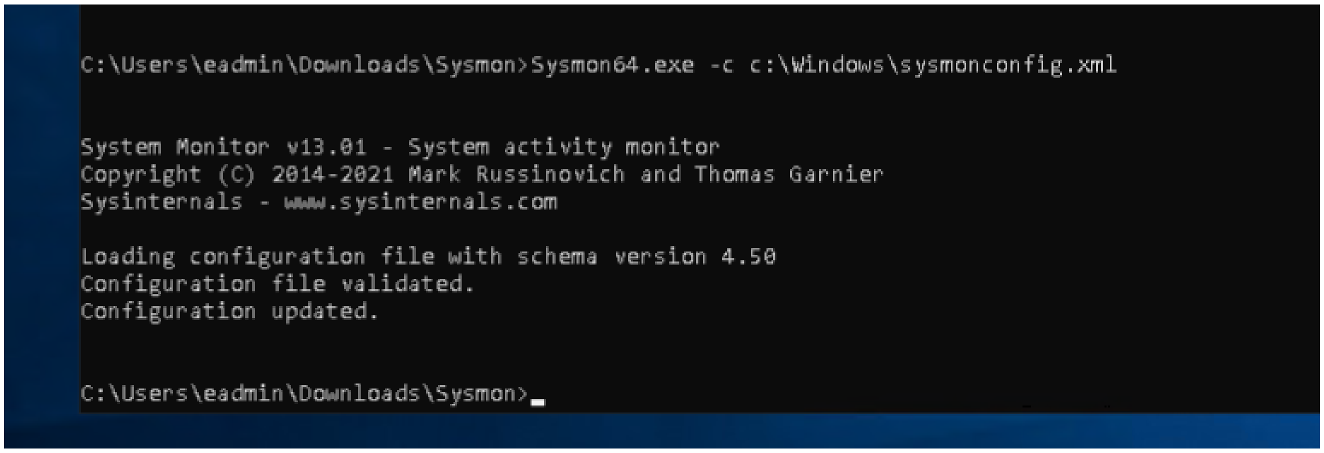
Operational   Number of events: 51,809 (!) New events available

Level	Date and Time	Source	Event ID	Task Category
Information	3/10/2021 7:31:02 PM	Sysmon	3	Network con...
Information	3/10/2021 7:31:02 PM	Sysmon	3	Network con...
Information	3/10/2021 7:31:00 PM	Sysmon	1	Process Creat...
Information	3/10/2021 7:31:00 PM	Sysmon	3	Network con...
Information	3/10/2021 7:30:59 PM	Sysmon	11	File created (r...
Information	3/10/2021 7:30:59 PM	Sysmon	1	Process Creat...
Information	3/10/2021 7:30:59 PM	Sysmon	3	Network con...
Information	3/10/2021 7:30:58 PM	Sysmon	3	Network con...
Information	3/10/2021 7:30:57 PM	Sysmon	3	Network con...

Event 11, Sysmon

```
File created:
RuleName: -
UtcTime: 2021-03-10 19:30:59.943
ProcessGuid: {d21184c1-1e73-6049-3a01-000000001000}
ProcessId: 1548
Image: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
TargetFilename: C:\Users\eadmin\AppData\Local\Temp\__PSScriptPolicyTest_jnrzawkv.spb.ps1
CreationUtcTime: 2021-03-10 19:30:59.943
```

Log Name:	Microsoft-Windows-Sysmon/Operational		
Source:	Sysmon	Logged:	3/10/2021 7:30:59 PM
Event ID:	11	Task Category:	File created (rule: FileCreate)
Level:	Information	Keywords:	



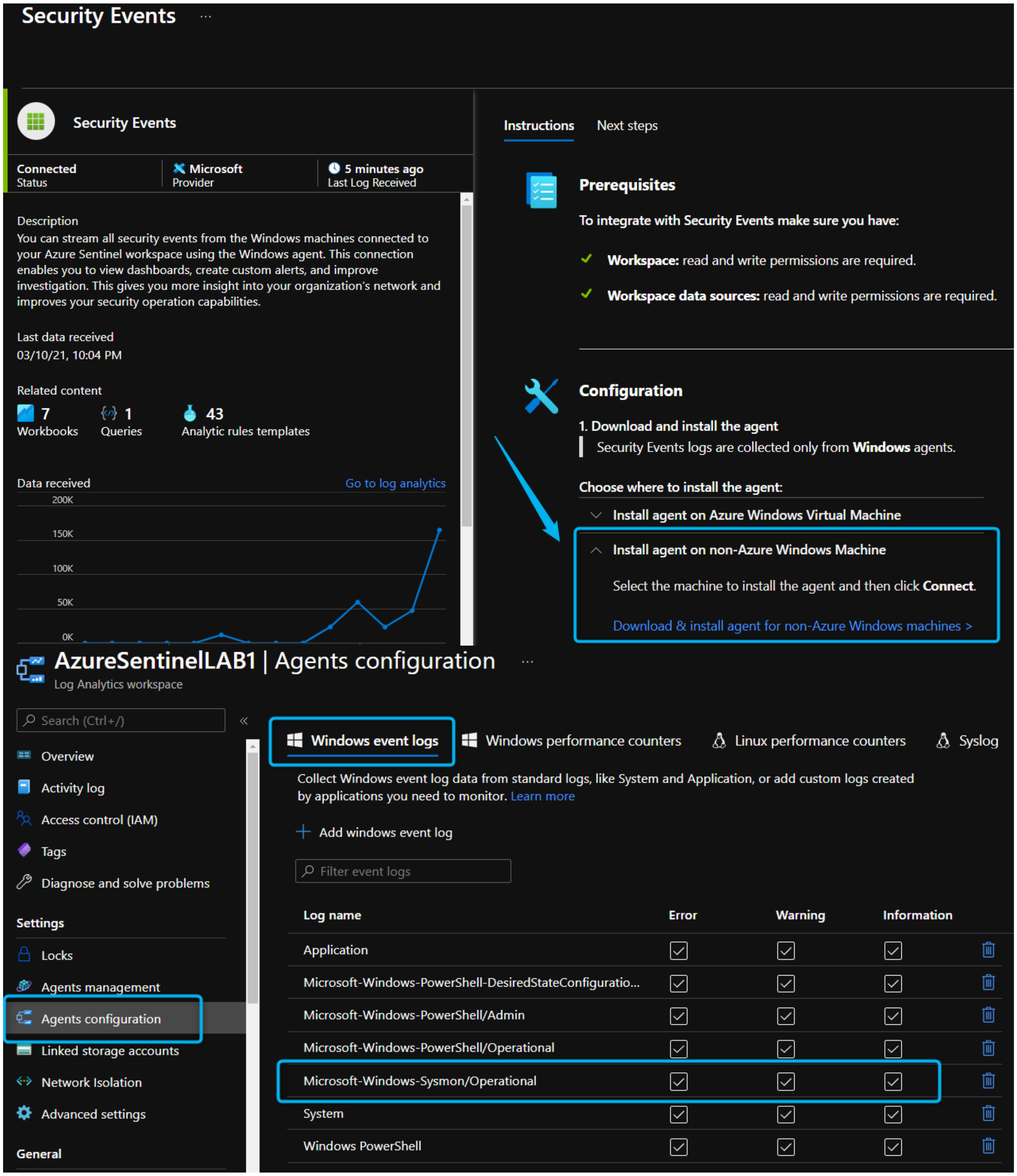
```
C:\Users\eadmin\Downloads\Sysmon>Sysmon64.exe -c c:\Windows\sysmonconfig.xml


System Monitor v13.01 - System activity monitor
Copyright (C) 2014-2021 Mark Russinovich and Thomas Garnier
Sysinternals - www.sysinternals.com

Loading configuration file with schema version 4.50
Configuration file validated.
Configuration updated.


C:\Users\eadmin\Downloads\Sysmon>_
```

Azure Sentinel and Sysmon Configuration

Connecting servers to Azure Sentinel occurs via dedicated agents (non-Azure Windows Machine). We need to install the Agent together with the workspace ID and its primary key on the server-side.

If you're working with the Security Event, the agent can be downloaded via the Security Event connector.

TIP: It's recommended to work with common Security Events alongside the Sysmon



Description
You can stream all security events from the Windows machines connected to your Azure Sentinel workspace using the Windows agent. This connection enables you to view dashboards, create custom alerts, and improve investigation. This gives you more insight into your organization's network and improves your security operation capabilities.

Last data received
03/10/21, 10:04 PM

Prerequisites

To integrate with Security Events make sure you have:

- ✓ **Workspace:** read and write permissions are required.
- ✓ **Workspace data sources:** read and write permissions are required.

Configuration

1. Download and install the agent

Security Events logs are collected only from **Windows** agents.

Choose where to install the agent:

- Install agent on Azure Windows Virtual Machine
- Install agent on non-Azure Windows Machine

Select the machine to install the agent and then click **Connect**.

Download & install agent for non-Azure Windows machines >

AzureSentinelLAB1 | Agents configuration
Log Analytics workspace

Collect Windows event log data from standard logs, like System and Application, or add custom logs created by applications you need to monitor. Learn more

Log name	Error	Warning	Information
Application	✓	✓	✓
Microsoft-Windows-PowerShell-DesiredStateConfiguratio...	✓	✓	✓
Microsoft-Windows-PowerShell/Admin	✓	✓	✓
Microsoft-Windows-PowerShell/Operational	✓	✓	✓
Microsoft-Windows-Sysmon/Operational	✓	✓	✓
System	✓	✓	✓
Windows PowerShell	✓	✓	✓

Azure Sentinel Sysmon Queries

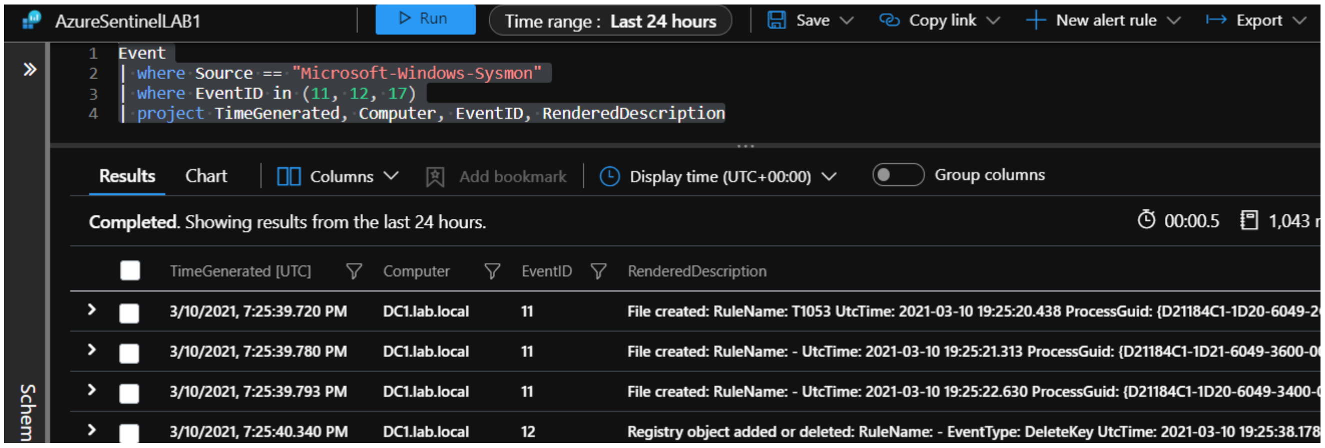
Once Sysmon is installed and configured on the Windows servers and configured on the Azure Sentinel, we can run queries for Sysmon. The query can be run with the commands below:

Check basic Sysmon event.

> *Event*
> *| where Source == "Microsoft-Windows-Sysmon"*

Check for important events.

> *Event*
> *| where Source == "Microsoft-Windows-Sysmon"*
> *| where EventID in (11, 12, 17)*
> *| project TimeGenerated, Computer, EventID, RenderedDescription*



```
Event
| where Source == "Microsoft-Windows-Sysmon"
| where EventID in (11, 12, 17)
| project TimeGenerated, Computer, EventID, RenderedDescription
```

Completed. Showing results from the last 24 hours. 00:00.5 1,043 r

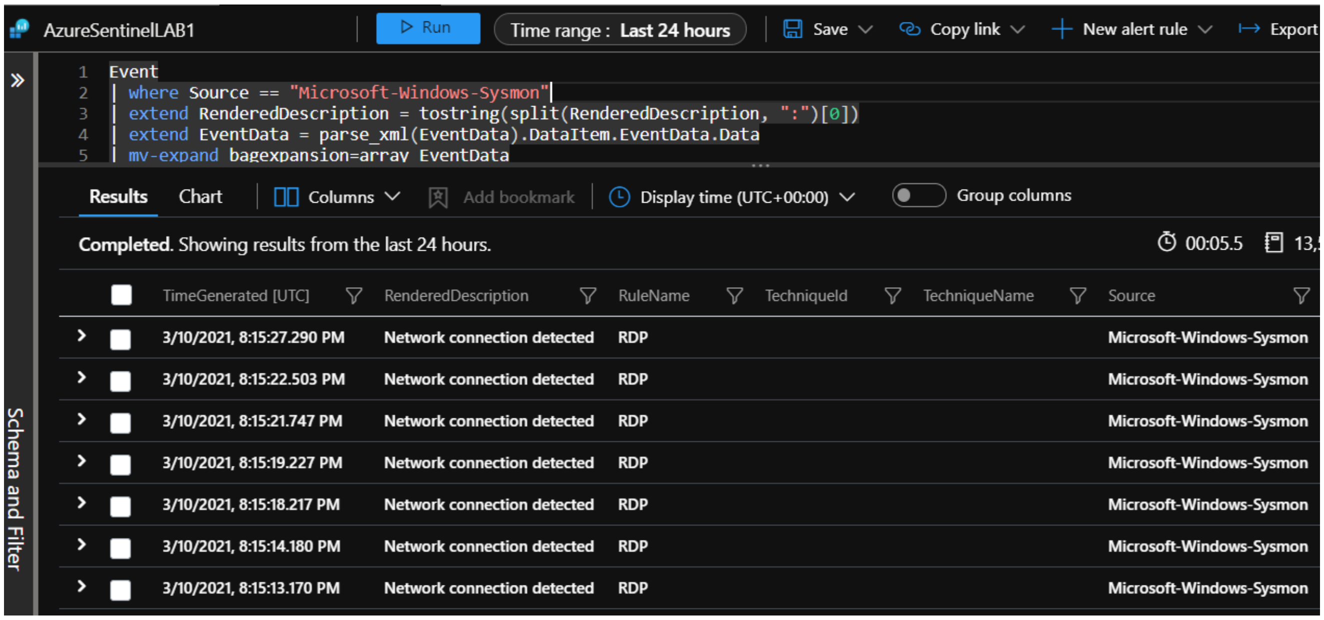
TimeGenerated [UTC]	Computer	EventID	RenderedDescription
3/10/2021, 7:25:39.720 PM	DC1.lab.local	11	File created: RuleName: T1053 UtcTime: 2021-03-10 19:25:20.438 ProcessGuid: {D21184C1-1D20-6049-2
3/10/2021, 7:25:39.780 PM	DC1.lab.local	11	File created: RuleName: - UtcTime: 2021-03-10 19:25:21.313 ProcessGuid: {D21184C1-1D21-6049-3600-0
3/10/2021, 7:25:39.793 PM	DC1.lab.local	11	File created: RuleName: - UtcTime: 2021-03-10 19:25:22.630 ProcessGuid: {D21184C1-1D20-6049-3400-
3/10/2021, 7:25:40.340 PM	DC1.lab.local	12	Registry object added or deleted: RuleName: - EventType: DeleteKey UtcTime: 2021-03-10 19:25:38.178

You might notice that not all information is available right away in the form of columns. Instead, the real important data is stored inside of the two columns "ParameterXml" and "EventData":

KQL queries can parse those columns via the query below:

> *Event*
> *| where Source == “Microsoft-Windows-Sysmon”*
> *| extend RenderedDescription = tostring(split(RenderedDescription, “:”)[0])*
> *| extend EventData = parse_xml(EventData).DataItem.EventData.Data*
> *| mv-expand bagexpansion=array EventData*
> *| evaluate bag_unpack(EventData)*
> *| extend Key=tostring([‘@Name’]), Value=[‘#text’]*
> *| evaluate pivot(Key, any(Value), TimeGenerated, Source, EventLog, Computer, EventLevel, EventLevelName, EventID, UserName, RenderedDescription, MG, ManagementGroupName, Type, _ResourceId)*
> *| parse RuleName with * ‘technique_id=’ TechniqueId ‘,’ * ‘technique_name=’ TechniqueName*
> *| order by TimeGenerated desc*

The Sysmon prase query



```
Event
| where Source == "Microsoft-Windows-Sysmon"
| extend RenderedDescription = tostring(split(RenderedDescription, ":")[0])
| extend EventData = parse_xml(EventData).DataItem.EventData.Data
| mv-expand bagexpansion=array EventData
```

Completed. Showing results from the last 24 hours. 00:05.5 13,

TimeGenerated [UTC]	RenderedDescription	RuleName	TechniqueId	TechniqueName	Source
3/10/2021, 8:15:27.290 PM	Network connection detected	RDP			Microsoft-Windows-Sysmon
3/10/2021, 8:15:22.503 PM	Network connection detected	RDP			Microsoft-Windows-Sysmon
3/10/2021, 8:15:21.747 PM	Network connection detected	RDP			Microsoft-Windows-Sysmon
3/10/2021, 8:15:19.227 PM	Network connection detected	RDP			Microsoft-Windows-Sysmon
3/10/2021, 8:15:18.217 PM	Network connection detected	RDP			Microsoft-Windows-Sysmon
3/10/2021, 8:15:14.180 PM	Network connection detected	RDP			Microsoft-Windows-Sysmon
3/10/2021, 8:15:13.170 PM	Network connection detected	RDP			Microsoft-Windows-Sysmon

Sysmon IR

Sysmon Events

- The service logs events immediately.
- The driver installs as a boot-start driver to capture activity from early in the boot process.
- Sysmon does not replace your existing event logs.

Important events for Incident Response

- Event ID 11: FileCreate – Useful for monitoring autostart locations and available places malware drops during initial infection.

- Event ID 12: RegistryEvent – Useful for monitoring changes to Registry autostart locations or specific malware registry modifications.
- Event ID 17: PipeEvent – Malware usually uses named pipes for inter-process communication.

The Events – 4688

- Sysmon events can detect new EXEs and DLLs.
- Can detect ransomware such as Petya or Wannacry, which used SMB to spread.
- Log event is produced every time an EXE loads as a new process.
- Known EXE and compare each 4688 against that list and identify new actions, like Petya's EXEs, that run on your network.
- The only problem with using 4688 is it's based on the EXE name and including the path.
- What happens if the attacker uses a name similar to that of a known file
- Sysmon event ID 1 is logged simultaneously as 4688, but it also provides the EXE hash.
- If an attacker replaces a known EXE, the hash will change.
- Comparison against known hashes will fail, and detecting a new EXE executing for the first time in your environment.
- Logs process creation with a full command line for both current and parent processes
- Records the hash of process image files using SHA1, MD5 or SHA256
- Includes a process GUID in process create events to allow for correlation of events even when Windows reuses process IDs
- Optionally logs network connections, including each connection's source process, IP addresses, port numbers, hostnames, and port names.

Use Cases

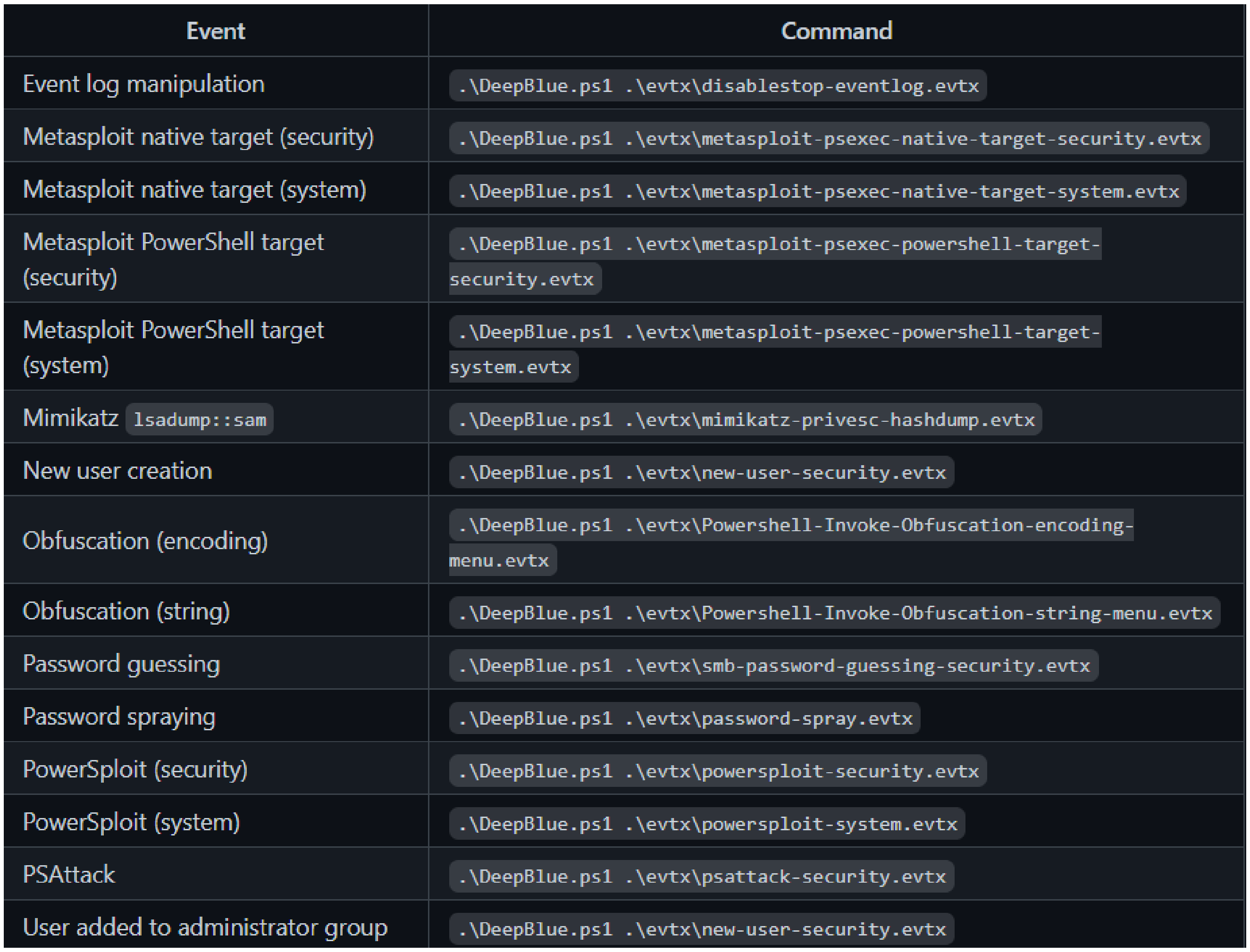
The main use cases with Sysmon hunting:

- Productivity App (e.g., Word, Excel, PowerPoint, Outlook) launches cmd.exe or powershell.exe
- Abnormal parent of svchost.exe
- Whoami.exe running
- net.exe use
- Webshell
- Data exfiltration
- Mimikatz
- Process injection

Sysmon Simulation

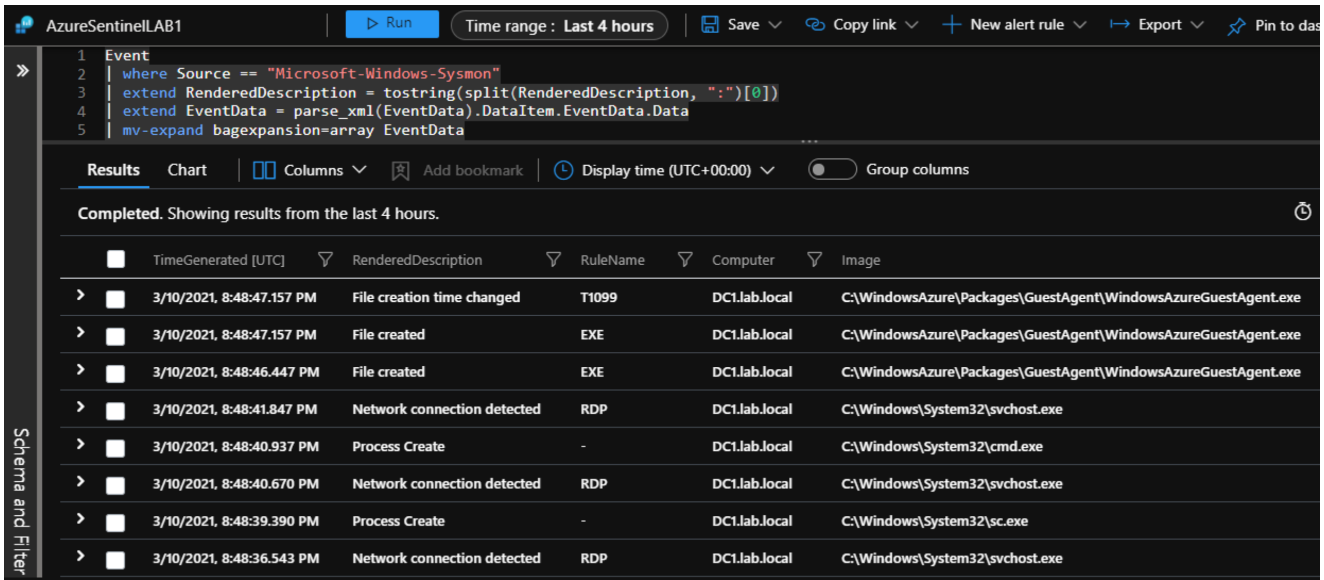
You can simulate to check how the Sysmon event logs are working with many tools, and with this example, I'm using the DeepBlueCLI.

The DeepBlueCLI can be downloaded from GitHub > sans-blue-team/DeepBlueCLI (github.com), and once you've downloaded it, you can run with the scenarios below.



Event	Command
Event log manipulation	`.\DeepBlue.ps1 .\evtx\disablestop-eventlog.evtx`
Metasploit native target (security)	`.\DeepBlue.ps1 .\evtx\metasploit-psexec-native-target-security.evtx`
Metasploit native target (system)	`.\DeepBlue.ps1 .\evtx\metasploit-psexec-native-target-system.evtx`
Metasploit PowerShell target (security)	`.\DeepBlue.ps1 .\evtx\metasploit-psexec-powershell-target-security.evtx`
Metasploit PowerShell target (system)	`.\DeepBlue.ps1 .\evtx\metasploit-psexec-powershell-target-system.evtx`
Mimikatz `lsadump::sam`	`.\DeepBlue.ps1 .\evtx\mimikatz-privesc-hashdump.evtx`
New user creation	`.\DeepBlue.ps1 .\evtx\new-user-security.evtx`
Obfuscation (encoding)	`.\DeepBlue.ps1 .\evtx\Powershell-Invoke-Obfuscation-encoding-menu.evtx`
Obfuscation (string)	`.\DeepBlue.ps1 .\evtx\Powershell-Invoke-Obfuscation-string-menu.evtx`
Password guessing	`.\DeepBlue.ps1 .\evtx\smb-password-guessing-security.evtx`
Password spraying	`.\DeepBlue.ps1 .\evtx\password-spray.evtx`
PowerSploit (security)	`.\DeepBlue.ps1 .\evtx\powersploit-security.evtx`
PowerSploit (system)	`.\DeepBlue.ps1 .\evtx\powersploit-system.evtx`
PSAttack	`.\DeepBlue.ps1 .\evtx\psattack-security.evtx`
User added to administrator group	`.\DeepBlue.ps1 .\evtx\new-user-security.evtx`

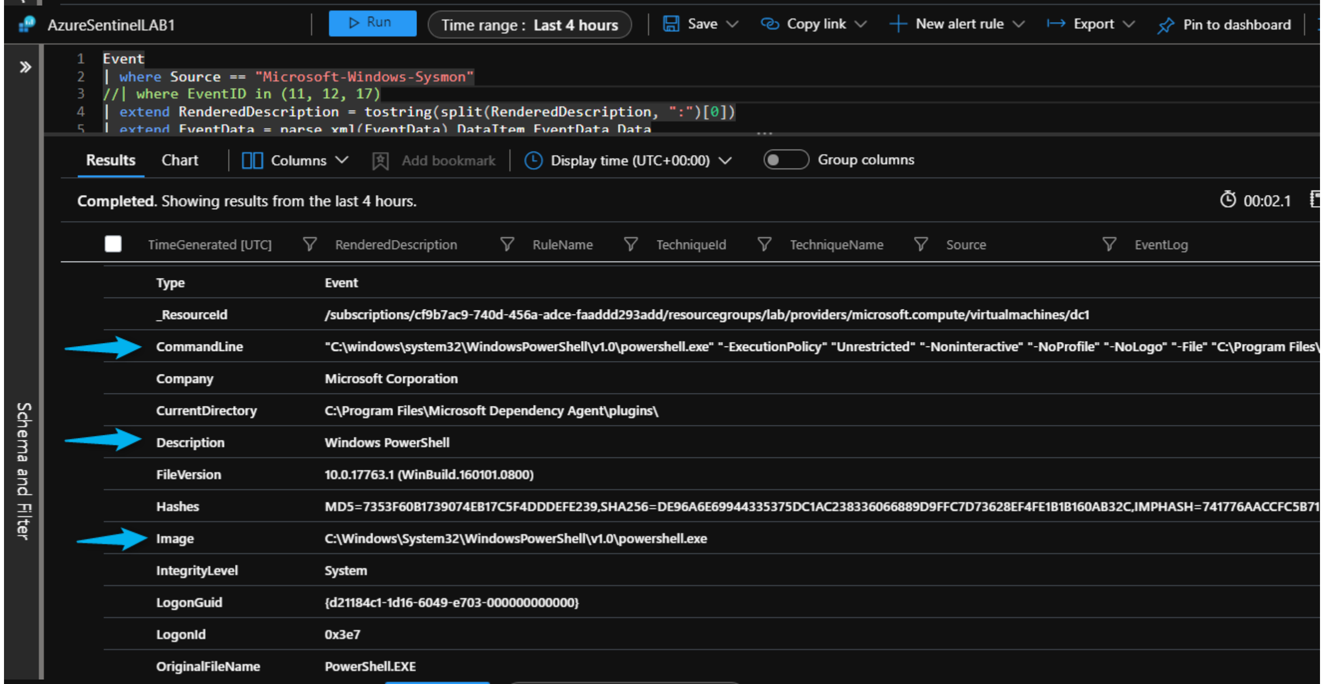
When I ran the DeepBlueCLI tool with many scenarios, I received much useful information from Azure Sentinel. Once I started with the hunting phases, I saw many indicators with File created, Process Create, and Network connection detected, including C2 connections.



```
Event
| where Source == "Microsoft-Windows-Sysmon"
| extend RenderedDescription = tostring(split(RenderedDescription, ":")[0])
| extend EventData = parse_xml(EventData).DataItem.EventData.Data
| mv-expand bagexpansion=array EventData
```

Completed. Showing results from the last 4 hours.

TimeGenerated [UTC]	RenderedDescription	RuleName	Computer	Image
3/10/2021, 8:48:47.157 PM	File creation time changed	T1099	DC1.lab.local	C:\WindowsAzure\Packages\GuestAgent\WindowsAzureGuestAgent.exe
3/10/2021, 8:48:47.157 PM	File created	EXE	DC1.lab.local	C:\WindowsAzure\Packages\GuestAgent\WindowsAzureGuestAgent.exe
3/10/2021, 8:48:46.447 PM	File created	EXE	DC1.lab.local	C:\WindowsAzure\Packages\GuestAgent\WindowsAzureGuestAgent.exe
3/10/2021, 8:48:41.847 PM	Network connection detected	RDP	DC1.lab.local	C:\Windows\System32\svchost.exe
3/10/2021, 8:48:40.937 PM	Process Create	-	DC1.lab.local	C:\Windows\System32\cmd.exe
3/10/2021, 8:48:40.670 PM	Network connection detected	RDP	DC1.lab.local	C:\Windows\System32\svchost.exe
3/10/2021, 8:48:39.390 PM	Process Create	-	DC1.lab.local	C:\Windows\System32\sc.exe
3/10/2021, 8:48:36.543 PM	Network connection detected	RDP	DC1.lab.local	C:\Windows\System32\svchost.exe

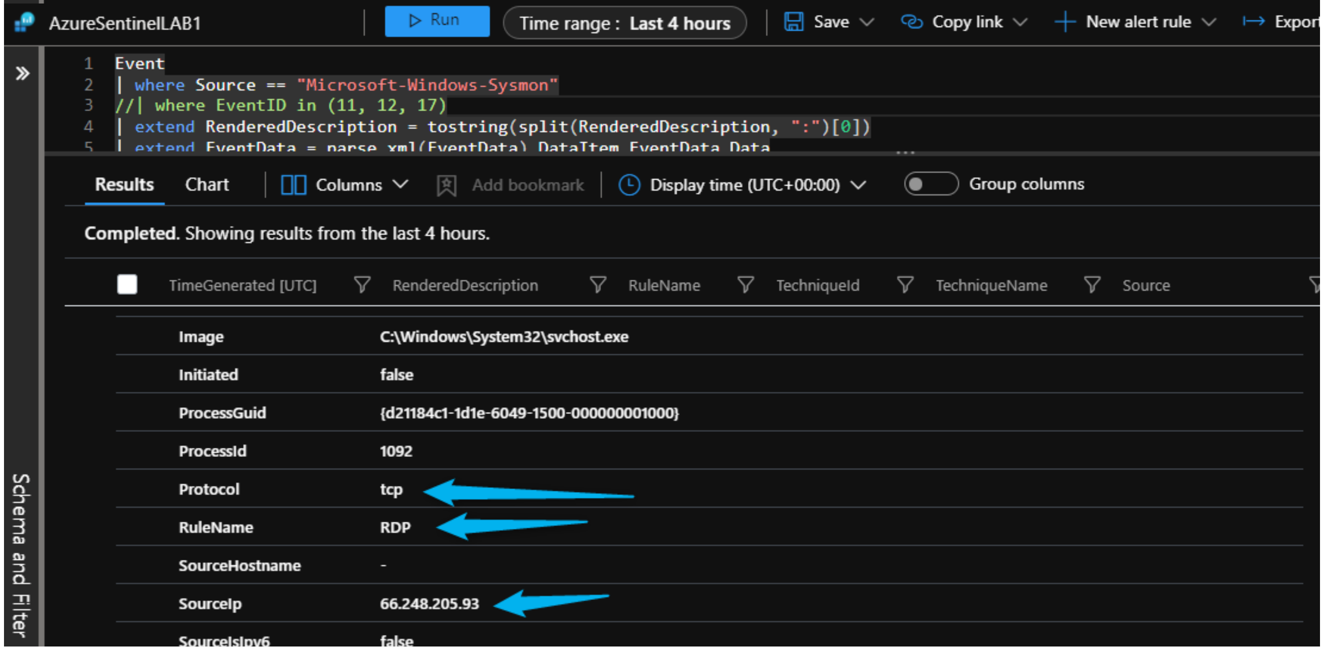


```
Event
| where Source == "Microsoft-Windows-Sysmon"
//| where EventID in (11, 12, 17)
| extend RenderedDescription = tostring(split(RenderedDescription, ":")[0])
| extend EventData = parse_xml(EventData).DataItem.EventData.Data
```

Completed. Showing results from the last 4 hours.

TimeGenerated [UTC] | RenderedDescription | RuleName | TechniqueId | TechniqueName | Source | EventLog

Type	Event
_ResourceId	/subscriptions/cf9b7ac9-740d-456a-adce-faaddd293add/resourcegroups/lab/providers/microsoft.compute/virtualmachines/dc1
CommandLine	"C:\windows\system32\WindowsPowerShell\v1.0\powershell.exe" "-ExecutionPolicy" "Unrestricted" "-Noninteractive" "-NoProfile" "-NoLogo" "-File" "C:\Program Files\
Company	Microsoft Corporation
CurrentDirectory	C:\Program Files\Microsoft Dependency Agent\plugins\
Description	Windows PowerShell
FileVersion	10.0.17763.1 (WinBuild.160101.0800)
Hashes	MD5=7353F60B1739074EB17C5F4DDDEFE239,SHA256=DE96A6E69944335375DC1AC238336066889D9FFC7D73628EF4FE1B1B160AB32C,IMPHASH=741776AACCFC5B71
Image	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
IntegrityLevel	System
LogonGuid	{d21184c1-1d16-6049-e703-000000000000}
LogonId	0x3e7
OriginalFileName	PowerShell.EXE

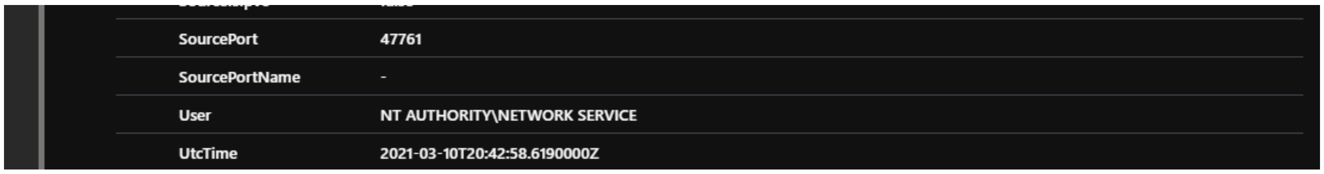


```
Event
| where Source == "Microsoft-Windows-Sysmon"
//| where EventID in (11, 12, 17)
| extend RenderedDescription = tostring(split(RenderedDescription, ":")[0])
| extend EventData = parse_xml(EventData).DataItem.EventData.Data
```

Completed. Showing results from the last 4 hours.

TimeGenerated [UTC] | RenderedDescription | RuleName | TechniqueId | TechniqueName | Source

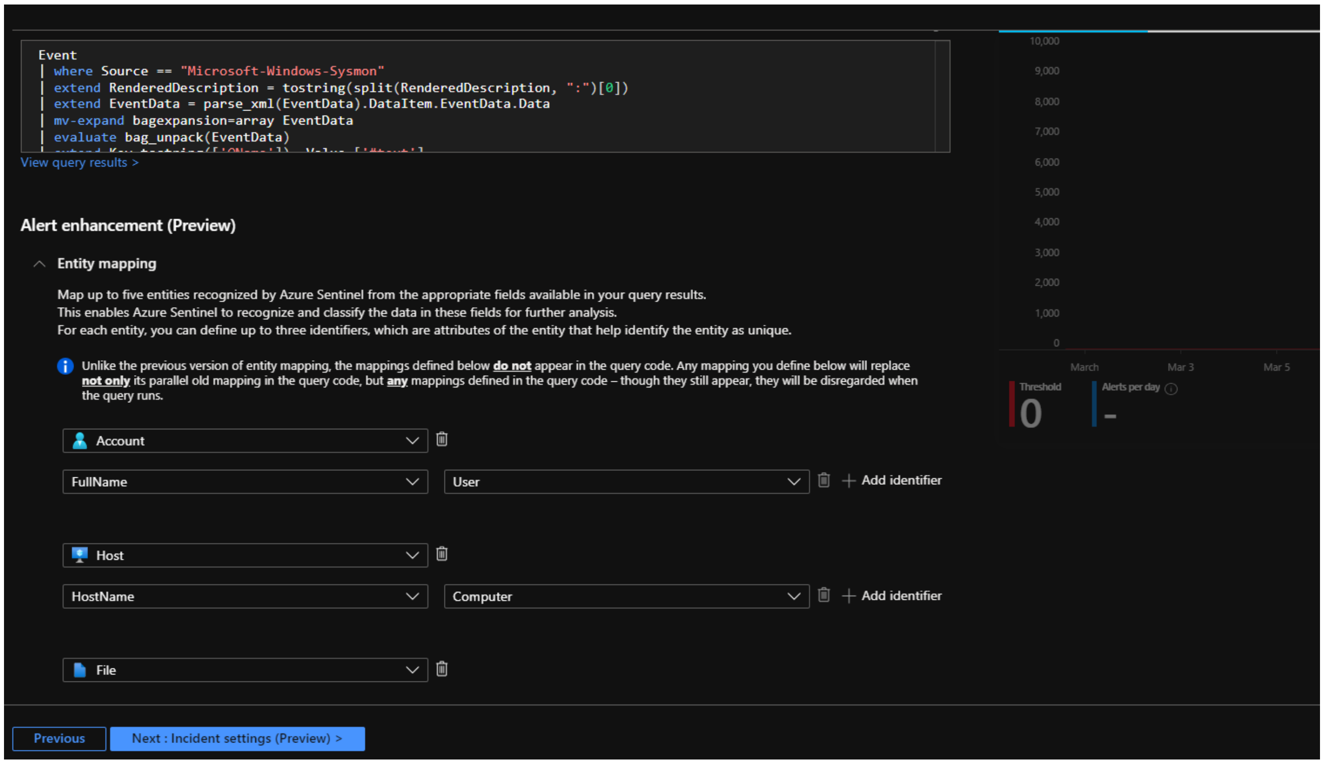
Image	C:\Windows\System32\svchost.exe
Initiated	false
ProcessGuid	{d21184c1-1d1e-6049-1500-000000001000}
ProcessId	1092
Protocol	tcp
RuleName	RDP
SourceHostname	-
SourceIp	66.248.205.93
SourceIsIpv6	false



```
SourcePort        47761
SourcePortName    -
User              NT AUTHORITY\NETWORK SERVICE
UtcTime           2021-03-10T20:42:58.6190000Z
```

Azure Sentinel Incident

Like any attack, we must create an incident and raise an alert when the attack appears on servers and provide a way to investigate with all indicators from Sysmon. With this example, the query includes the Sysmon parse query and the new entity mapping for account, host, and files.



```
Event
| where Source == "Microsoft-Windows-Sysmon"
| extend RenderedDescription = tostring(split(RenderedDescription, ":")[0])
| extend EventData = parse_xml(EventData).DataItem.EventData.Data
| mv-expand bagexpansion=array EventData
| evaluate bag_unpack(EventData)
```

View query results >

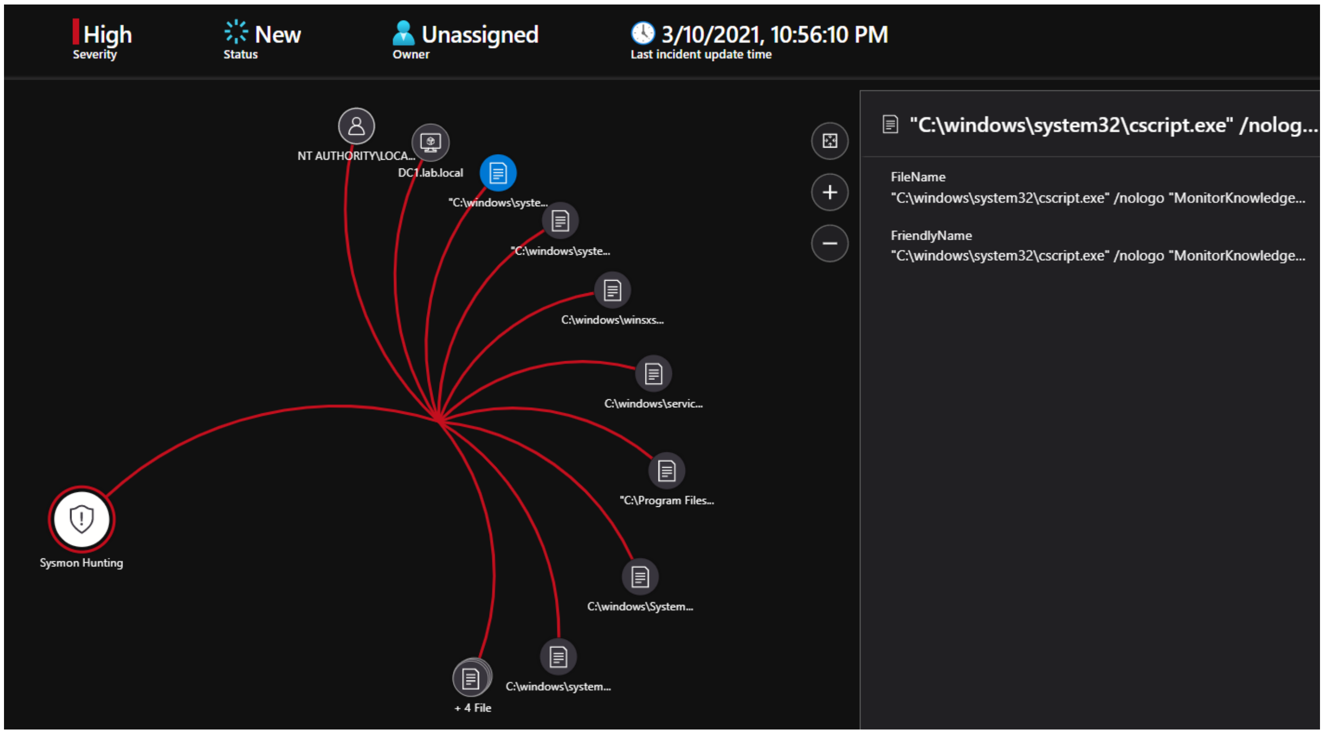
Alert enhancement (Preview)

Entity mapping

Map up to five entities recognized by Azure Sentinel from the appropriate fields available in your query results.
This enables Azure Sentinel to recognize and classify the data in these fields for further analysis.
For each entity, you can define up to three identifiers, which are attributes of the entity that help identify the entity as unique.

Unlike the previous version of entity mapping, the mappings defined below **do not** appear in the query code. Any mapping you define below will replace **not only** its parallel old mapping in the query code, but **any** mappings defined in the query code – though they still appear, they will be disregarded when the query runs.

Account	
FullName	User
Host	
HostName	Computer
File	



"C:\windows\system32\cscript.exe" /nolog...

FileName
"C:\windows\system32\cscript.exe" /nologo "MonitorKnowledge...

FriendlyName
"C:\windows\system32\cscript.exe" /nologo "MonitorKnowledge...

More about Azure Sentinel