Lucas McMahon
9/4/24

Using Wazuh and Sysmon for Cyber Defense

Table of Contents

Introduction

In this lab, we will use Wazuh, an open-source security monitoring and incident response tool, to investigate and identify a security breach on a compromised system. Wazuh provides comprehensive visibility into system activities, allowing us to detect anomalies and identify potential attack vectors. This lab focuses on using Wazuh's capabilities to analyze an intrusion, determine how the attacker gained initial access, and uncover subsequent malicious activities, such as the creation of scheduled tasks, the generation of new user accounts, and the dumping of credentials.

By the end of this lab, we will gain a better understanding of how to leverage Wazuh to detect, analyze, and respond to an attack in a real-world scenario, enhancing our skills in threat hunting and incident response.

Objectives

- Identify how the attacker gained initial access to the system.
- Determine which scheduled tasks were created by the attacker.
- Retrieve the credentials of a newly created user account.
- Discover the name of the executable file used by the attacker to dump credentials.
- Extract the credentials that were leaked during the attack.

Lab Environment

This lab is run using TryHackMe's Monday Monitor challenge, which provides the credentials for a Wazuh manager account and two Windows Servers as agents. I am accessing the manager account from my personal Windows browser.

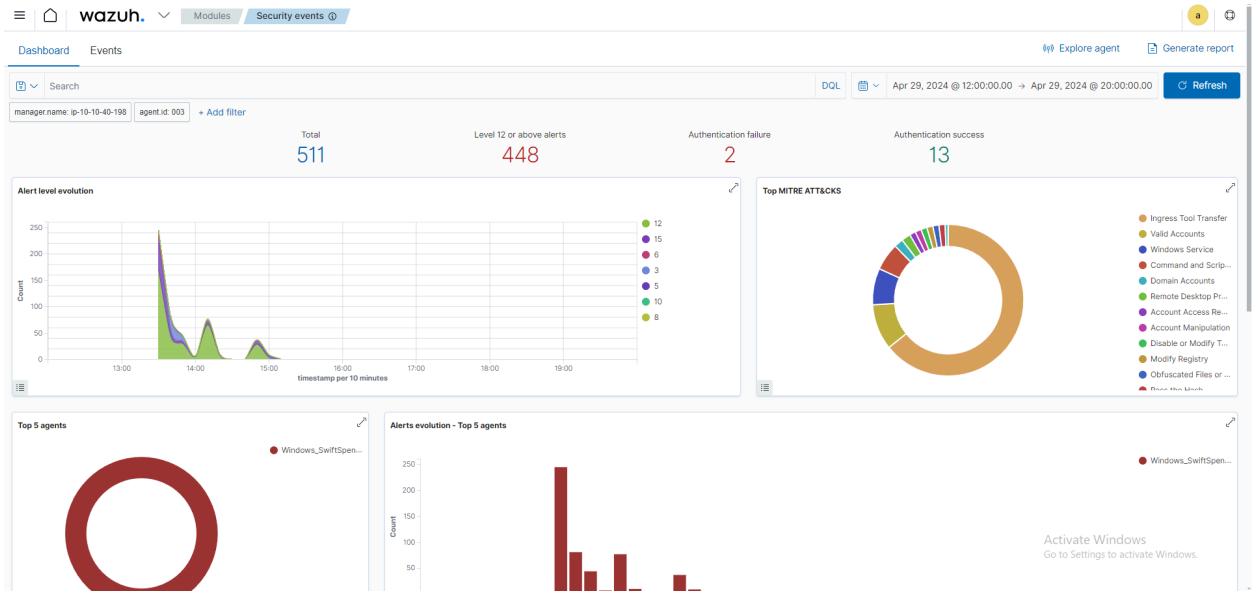
Lucas McMahon
9/4/24

Tools and Technologies Used

- Wazuh EDR
- Sysmon
- Windows Server 2019

Step-by-Step Process

Step One
The first step is logging onto the provided Wazuh account with access to the logs of two Windows Server agents. We navigate to the security events module, where a filter is provided for analysis.



After setting the date and filter, Wazuh provides us with information such as the number of alerts on the system, the types of alerts, and when they occurred.

We know that initial access was obtained through a downloaded file on the localhost, and our first goal is to find that file. Wazuh allows us to perform a basic search across all the logs obtained, so we start by filtering all events associated with localhost.

Lucas McMahon
9/4/24

Security Alerts

	Time ↑	Agent	Agent name	Technique(s)	Tactic(s)	Description
›	Apr 29, 2024 @ 13:45:31.277	003	Windows_SwiftSpend2			Microsoft Office Product Spawning Windows Shell
›	Apr 29, 2024 @ 13:50:12.344	003	Windows_SwiftSpend2			Microsoft Office Product Spawning Windows Shell
›	Apr 29, 2024 @ 13:55:36.383	003	Windows_SwiftSpend2			Detects suspicious file execution by wscript and cscri

There are only three events, and the third one mentions suspicious file execution. Wazuh allows you to click on the event to reveal more specific information. A file named **PhishingAttachment** executed another file from the local host, which gives us the answer we need.

```
\"powershell.exe\" & {$url = 'http://localhost/PhishingAttachment.xlsm' Invoke-WebRequest -Uri $url -OutFile $env:TEMP\\SwiftSpend_Financial_Expenses.xlsm}
```

Step Two

Our next goal is to find the command executed to create a scheduled task. A quick search for scheduler events returns 10 results.

Security Alerts

	Time ↓	Agent	Agent name
›	Apr 29, 2024 @ 14:58:09.823	003	Windows_SwiftSpend2
›	Apr 29, 2024 @ 14:26:08.348	003	Windows_SwiftSpend2
›	Apr 29, 2024 @ 14:12:43.386	003	Windows_SwiftSpend2
›	Apr 29, 2024 @ 14:00:31.016	003	Windows_SwiftSpend2
›	Apr 29, 2024 @ 13:58:09.792	003	Windows_SwiftSpend2
›	Apr 29, 2024 @ 13:39:44.679	003	Windows_SwiftSpend2
›	Apr 29, 2024 @ 13:39:43.364	003	Windows_SwiftSpend2
›	Apr 29, 2024 @ 13:39:42.373	003	Windows_SwiftSpend2
›	Apr 29, 2024 @ 13:39:42.345	003	Windows_SwiftSpend2
›	Apr 29, 2024 @ 13:39:38.921	003	Windows_SwiftSpend2

Rows per page: 10

Several events are part of normal activities, so to filter, I did a Google search for the command used to create a scheduled task in PowerShell, which is `schtasks.exe /create`. This search reveals the single result we are looking for. Expanding the event will show the command used to create an encoded task at 12:34.

```
cmd.exe /c reg add HKCU\SOFTWARE\ATOMIC-T1053.005 /v test /t REG_SZ /d cGluZyB3d3cueW91YXJldnVsbmVyYWJsZS50aG0= /f & schtasks.exe /Create /F /TN "ATOMIC-T1053.005" /TR "cmd /c start /min \"\" powershell.exe -Command IEX([System.Text.Encoding]::ASCII.GetString([System.Convert]::FromBase64String((Get-ItemProperty -Path HKCU:\\SOFTWARE\\ATOMIC-T1053.005).test)))" /sc daily /st 12:34
```

Step Three

Our next step is to find what was encoded into our scheduled task to see what the attacker is doing daily. I am going to use CyberChef and paste the encoded text into the Base64 decoder to find the answer.

Output

```
ping www.youarevulnerable.thm
```

It appears that our attacker is performing command-and-control (C2) activity after pivoting into one of the devices. Let's see what else we can find out.

Step Four

We are told the attacker has created a new account, and our job is to find the password set for it. This part of the challenge took some time for me. I was struggling to filter for a password change from the command line. After researching online, I came to understand that `net.exe` is an executable used to alter group policy settings, such as passwords.

After filtering the events for `net.exe`, I found evidence of the attacker running several commands to maintain persistence on the device.

```
\"powershell.exe\" & {net user get-localuser get-localgroupmember -group Users cmdkey.exe /list ls C:/Users get-childitem C:\\Users\\ dir C:\\Users\\ get-localgroup net localgroup}
```

```
\"C:\\Windows\\system32\\net.exe\" localgroup Administrators guest /add
```

```
\"C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe\" -ExecutionPolicy bypass
```

```
\"C:\\Windows\\system32\\net.exe\" user guest I_AM_M0NIT0R1NG
```

The last command listed shows the user logging in with the new password.

Lucas McMahon
9/4/24

Step Five

Next, our goal is to find out which executable file was used to dump credentials from this device. In one of my earlier modules, I learned about the use of Mimikatz, a powerful tool that security professionals and attackers use to extract data from a device. I went ahead and filtered for events using this tool.

data.win.eventdata.image	C:\\Tools\\AtomicRedTeam\\atomics\\T1003.001\\bin\\x64\\memotech.exe
data.win.eventdata.integrityLevel	High
data.win.eventdata.logonGuid	{c5d2b969-8a47-662f-8b54-0a0000000000}
data.win.eventdata.logonId	0xa548b
data.win.eventdata.originalFileName	mimikatz.exe

It did not take too much investigation to find the use of `mimikatz.exe` through a file called `memotech.exe`, which is the answer we are looking for.

Results and Analysis

1. **Initial Access Identification:**
    - After logging into the Wazuh manager account, the security events module was used to analyze the logs from two Windows Server agents. By filtering all events associated with the localhost, three events were identified, one of which mentioned suspicious file execution.
    - The event revealed that a file named "PhishingAttachment" executed another file on the local host, confirming that initial access was obtained through a downloaded file.
2. **Scheduled Task Analysis:**
    - A search for scheduler events revealed 10 results, including normal activities. By filtering with the command `schtasks.exe /create` (the PowerShell command for creating scheduled tasks), the specific task created by the attacker was identified.

- The command found was:
  bash

```
\"cmd.exe\" /c \"reg add HKCU\\SOFTWARE\\ATOMIC-T1053.005 /v test /t REG_SZ /d cGluZyB3d3cueW91YXJldnVsbmVyYWJsZS50aG0= /f & schtasks.exe /Create /F /TN \"ATOMIC-T1053.005\" /TR \"cmd /c start /min \\\"\\\" powershell.exe -Command IEX([System.Text.Encoding]::ASCII.GetString([System.Convert]::FromBase64String((Get-ItemProperty -Path HKCU:\\\\SOFTWARE\\\\ATOMIC-T1053.005).test)))\" /sc daily /st 12:34\"
```

    - The decoded Base64 string showed the attacker was performing command and control (C2) activity on a daily schedule.
3. **Discovery of New Account Credentials:**
    - Searching for evidence of password changes led to the use of `net.exe`, a tool for managing user accounts and group policies.
    - Events filtered for `net.exe` showed several commands executed by the attacker to maintain persistence, including creating a new user "guest" with the password `I_AM_M0NIT0R1NG`.
4. **Executable Used for Credential Dumping:**
    - Filtering for tools like Mimikatz revealed that the attacker used an executable named `memotech.exe` to dump credentials. This executable is effectively a disguised version of Mimikatz, commonly used for credential extraction.

Challenges Faced

- **Filtering Commands Effectively:**
    - It was challenging to find the correct events related to specific actions due to the large volume of logs. Understanding the specific commands and the correct filters was essential, requiring additional research to identify the appropriate PowerShell and Windows commands.
- **Decoding Base64 Encoded Commands:**
    - Decoding the Base64 encoded string in the scheduled task was a bit tricky. It required the use of external tools like CyberChef to decode and understand what commands were being executed by the attacker.
- **Identifying the Right Tool Usage:**
    - There was some initial difficulty in pinpointing which executable was used for credential dumping. Familiarity with common tools like Mimikatz helped narrow down the search, but the attacker's use of a disguised executable (`memotech.exe`) added complexity.

Conclusion

This lab demonstrated the effectiveness of using Wazuh to detect and analyze malicious activities on a compromised system. By leveraging Wazuh's capabilities, we were able to:

- Identify the initial access vector used by the attacker.
- Detect a malicious scheduled task created for persistent C2 communication.
- Find the credentials of a newly created account used to maintain access.
- Uncover the executable used to dump credentials from the system.

The lab also highlighted the importance of understanding different system commands and how attackers may use common tools in disguise. This exercise enhanced our skills in using Wazuh for threat detection, incident response, and understanding real-world attack patterns.

References

- Wazuh Documentation: https://documentation.wazuh.com
- TryHackMe Monday Monitor Challenge: https://tryhackme.com
- CyberChef Tool: https://gchq.github.io/CyberChef/
- Microsoft Docs - Command-Line Reference: https://docs.microsoft.com/en-us/windows-server/administration/windows-commands
- MITRE ATT&CK Framework: https://attack.mitre.org