Hunting For Stuxbot

Threat Intelligence Report: Stuxbot

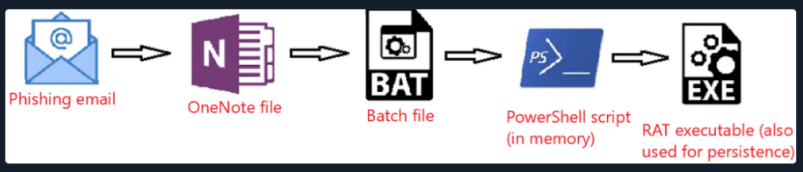
The present Threat Intelligence report underlines the immediate menace posed by the organized cybercrime collective known as "Stuxbot". The group initiated its phishing campaigns earlier this year and operates with a broad scope, seizing upon opportunities as they arise, without any specific targeting strategy – their motto seems to be anyone, anytime. The primary motivation behind their actions appears to be espionage, as there have been no indications of them exfiltrating sensitive blueprints, proprietary business information, or seeking financial gain through methods such as ransomware or blackmail.

- `Platforms in the Crosshairs:` `Microsoft Windows`
- `Threatened Entities:` `Windows Users`
- `Potential Impact:` `Complete takeover of the victim's computer / Domain escalation`
- `Risk Level:` `Critical`

The group primarily leverages opportunistic-phishing for initial access, exploiting data from social media, past breaches (e.g., databases of email addresses), and corporate websites. There is scant evidence suggesting spear-phishing against specific individuals.

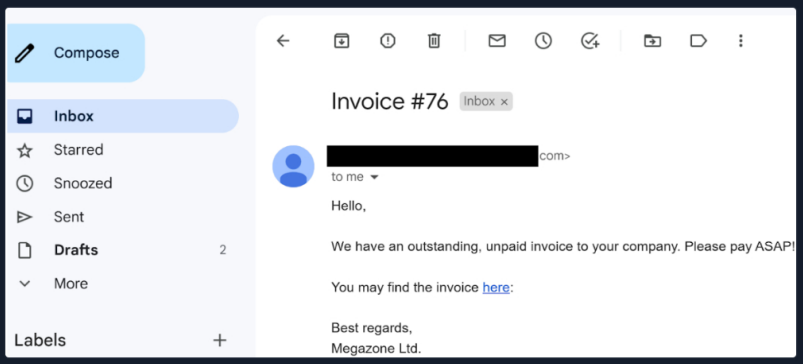
The document compiles all known Tactics Techniques and Procedures (TTPs) and Indicators of Compromise (IOCs) linked to the group, which are currently under continuous refinement. This preliminary sketch is confidential and meant exclusively for our partners, who are strongly advised to conduct scans of their infrastructures to spot potential successful breaches at the earliest possible stage.

In summary, the attack sequence for the initially compromised device can be laid out as follows:



`Initial Breach`

The phishing email is relatively rudimentary, with the malware posing as an invoice file. Here's an example of an actual phishing email that includes a link leading to a OneNote file:



Our forensic investigation into these attacks revealed that the link directs to a OneNote file, which has consistently been hosted on a file hosting service (e.g., Mega.io or similar platforms).

This OneNote file masquerades as an invoice featuring a 'HIDDEN' button that triggers an embedded batch file. This batch file, in turn, fetches PowerShell scripts, representing stage 0 of the malicious payload.

`RAT Characteristics`

The RAT deployed in these attacks is modular, implying that it can be augmented with an infinite range of capabilities. While only a few features are accessible once the RAT is staged, we have noted the use of tools that capture screen dumps, execute Mimikatz, provide an interactive `CMD shell` on compromised machines, and so forth.

? Go to Questions

Table of Contents

Threat Hunting & Threat Intelligence Fundamentals
- Threat Hunting Fundamentals
- The Threat Hunting Process
- Threat Hunting Glossary
- Threat Intelligence Fundamentals

Threat Hunting With The Elastic Stack
- Hunting For Stuxbot

Let's Go Hunting
- Skills Assessment

My Workstation

OFFLINE

Start Instance

∞ / 1 spawns left

Persistence

All persistence mechanisms utilized to date have involved an EXE file deposited on the disk.

Lateral Movement

So far, we have identified two distinct methods for lateral movement:

- Leveraging the original, Microsoft-signed PsExec
- Using WinRM

Indicators of Compromise (IOCs)

The following provides a comprehensive inventory of all identified IOCs to this point.

OneNote File:

- https://transfer.sh/get/kNxU7/invoice.one
- https://mega.io/dl9o1Dz/invoice.one

Staging Entity (PowerShell Script):

- https://pastebin.com/raw/AvHtdKb2
- https://pastebin.com/raw/gj58DKz

Command and Control (C&C) Nodes:

- 91.90.213.14:443
- 103.248.70.64:443
- 141.98.6.59:443

Cryptographic Hashes of Involved Files (SHA256):

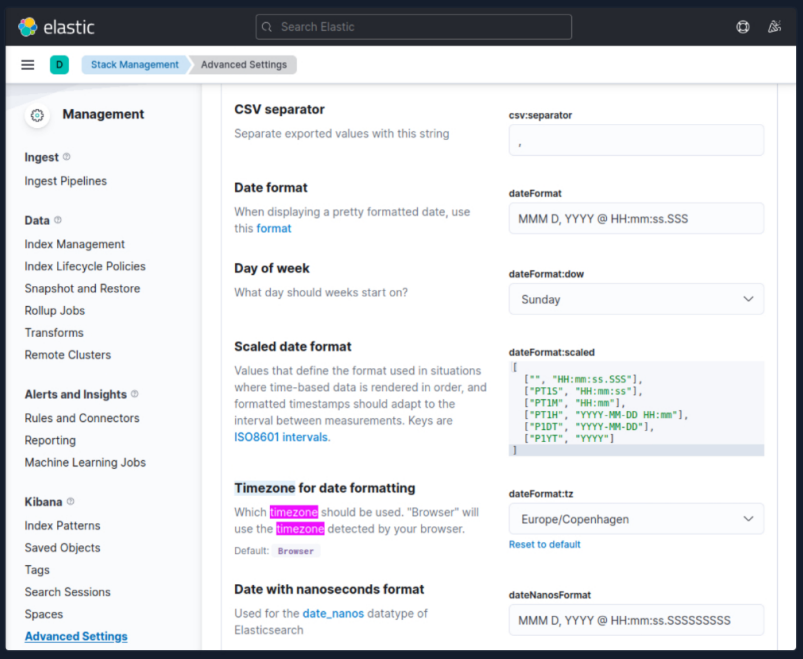
- 226A723FFB4A91D9950A8B266167C5B354AB0DB1DC225578494917FE53867EF2
- C346077DAD0342592DB753FE2AB36D2F9F1C76E55CF8556FE5CDA92897E99C7E
- 018D37CBD3878258C29DB3BC3F2988B6AE688843801B9ABC28E6151141AB66D4

Hunting For Stuxbot With The Elastic Stack

Navigate to the bottom of this section and click on `Click here to spawn the target system!`

Now, navigate to `http://[Target IP]:5601`, click on the side navigation toggle, and click on "Discover". Then, click on the calendar icon, specify "last 15 years", and click on "Apply".

Please also specify a `Europe/Copenhagen` timezone, through the following link `http://[Target IP]:5601/app/management/kibana/settings`.



The Available Data

The cybersecurity strategy implemented is predicated on the utilization of the Elastic stack as a SIEM solution. Through the "Discover" functionality we can see logs from multiple sources. These sources include:

- Windows audit logs (categorized under the index pattern windows*)
- System Monitor (Sysmon) logs (also falling under the index pattern windows*, more about Sysmon here)
- PowerShell logs (indexed under windows* as well, more about PowerShell logs here)
- Zeek logs, a network security monitoring tool (classified under the index pattern zeek*)

Our available threat intelligence stems from March 2023, hence it's imperative that our Kibana setup scans logs dating back at least to this time frame. Our "windows" index contains around 118,975 logs, while the "zeek" index houses approximately 332,261 logs.

The Environment

Our organization is relatively small, with about 200 employees primarily engaged in online marketing activities, thus our IT resource requirement is minimal. Office applications are the primary software in use, with Gmail serving as our standard email provider, accessed through a web browser. Microsoft Edge is the default browser on our company laptops. Remote technical support is provided through TeamViewer, and all our company devices are managed via Active Directory Group Policy Objects (GPOs). We're considering a transition to Microsoft Intune for endpoint management as part of an upcoming upgrade from Windows 10 to Windows 11.

The Task

Our task centers around a threat intelligence report concerning a malicious software known as "Stuxbot". We're expected to use the provided Indicators of Compromise (IOCs) to investigate whether there are any signs of compromise in our organization.

The Hunt

> The sequence of hunting activities is premised on the hypothesis of a successful phishing email delivering a malicious OneNote file. If our hypothesis had been the successful execution of a binary with a hash matching one from the threat intelligence report, we would have undertaken a different sequence of activities.

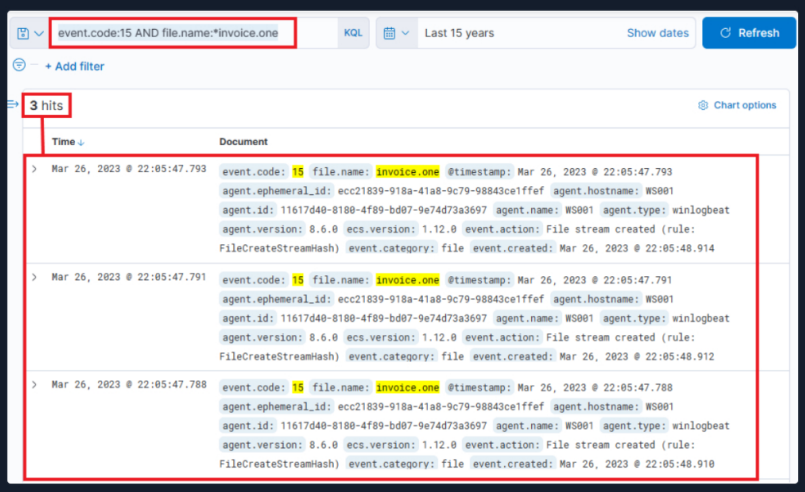
The report indicates that initial compromises all took place via "invoice.one" files. Despite this, we must continue to conduct searches on other IOCs as the threat actors may have introduced different delivery techniques between the time the report was created and the present. Back to the "invoice.one" files, a comprehensive search can be initiated based on Sysmon Event ID 15 (FileCreateStreamHash), which represents a browser file download event. We're assuming that a potentially malicious OneNote file was downloaded from Gmail, our organization's email provider.

Our search query should be the following.

Related fields: winlog.event_id or event.code and file.name

Hunting For Stuxbot

```
event.code:15 AND file.name:*invoice.one
```



While this development could imply serious implications, it's not yet confirmed if this file is the same one mentioned in the report. Further, signs of execution have not been probed. If we extend the event log to display its complete content, it'll reveal that MSEdge was the application (as indicated by `process.name` or

`process.executable`) used to download the file, which was stored in the Downloads folder of an employee named Bob.

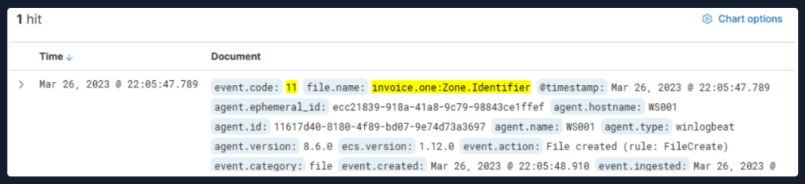
The timestamp to note is: `March 26, 2023 @ 22:05:47`

We can corroborate this information by examining Sysmon Event ID 11 (File create) and the "invoice.one" file name. This method is especially effective when browsers aren't involved in the file download process. The query is similar to the previous one, but the asterisk is at the end as the file name includes only the filename with an additional Zone Identifier, likely indicating that the file originated from the internet.

Related fields: winlog.event_id or event.code and file.name

Hunting For Stuxbot

```
event.code:11 AND file.name:invoice.one*
```



It's relatively easy to deduce that the machine which reported the "invoice.one" file has the hostname WS001 (check the `host.hostname` or `host.name` fields of the Sysmon Event ID 11 event we were just looking at) and an IP address of 192.168.28.130, which can be confirmed by checking any network connection event (Sysmon Event ID 3) from this machine (execute the following query `event.code:3 AND host.hostname:WS001` and check the `source.ip` field).

If we inspect network connections leveraging Sysmon Event ID 3 (Network connection) around the time this file was downloaded, we'll find that Sysmon has no entries. This is a common configuration to avoid capturing network connections created by browsers, which could lead to an overwhelming volume of logs, particularly those related to our email provider.

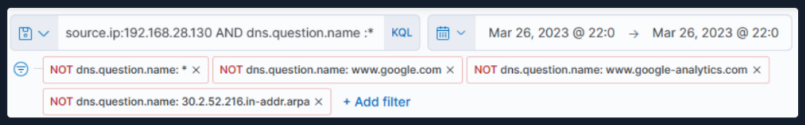
This is where Zeek logs prove invaluable. We should filter and examine the DNS queries that Zeek has captured from WS001 during the interval from `22:05:00` to `22:05:48`, when the file was downloaded.

Our `Zeek query` will search for a source IP matching 192.168.28.130, and since we're querying about DNS queries, we'll only pick logs that have something in the `dns.question.name` field. Note that this will return a lot of common noise, like google.com, etc., so it's necessary to filter that out. Here's the query and some filters.

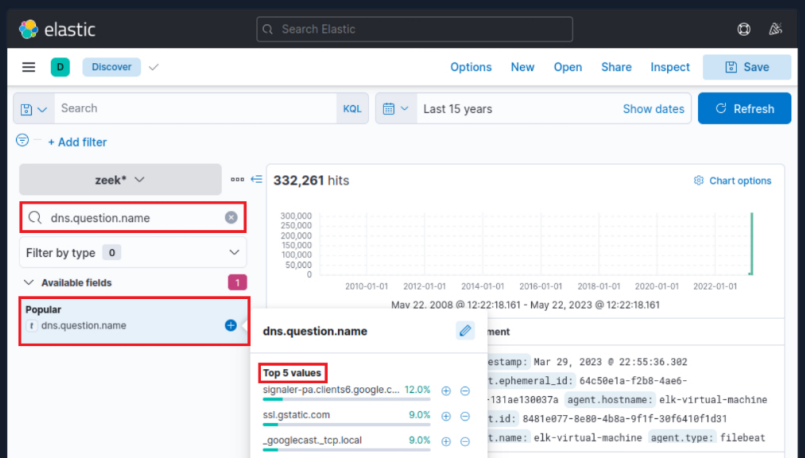
Related fields: source.ip and dns.question.name

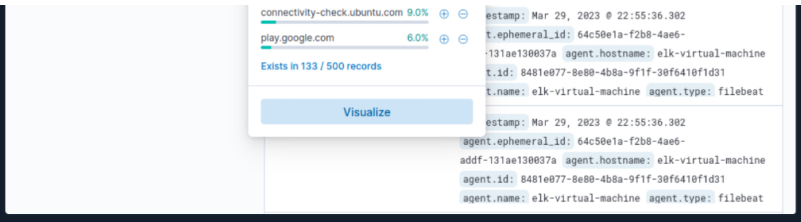
Hunting For Stuxbot

```
source.ip:192.168.28.130 AND dns.question.name:*
```

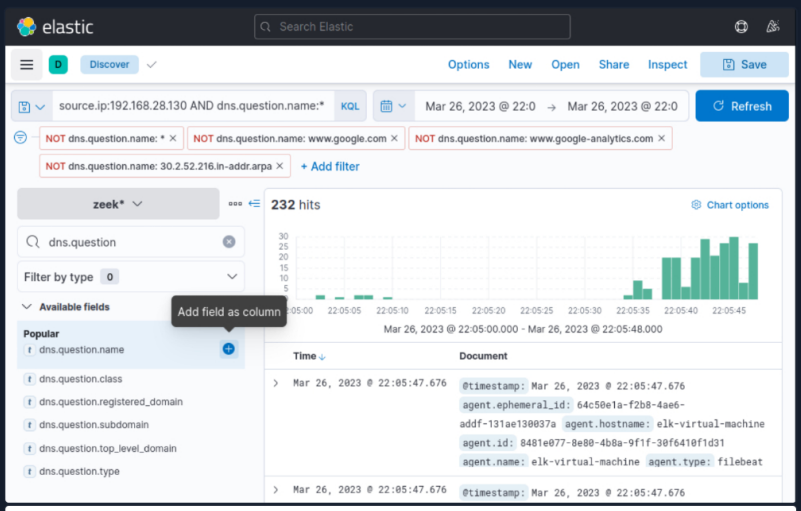


We can easily identify major sources of noise by looking at the most common values that Kibana has detected (click on a field as follows), and then apply a filter on the known noisy ones.





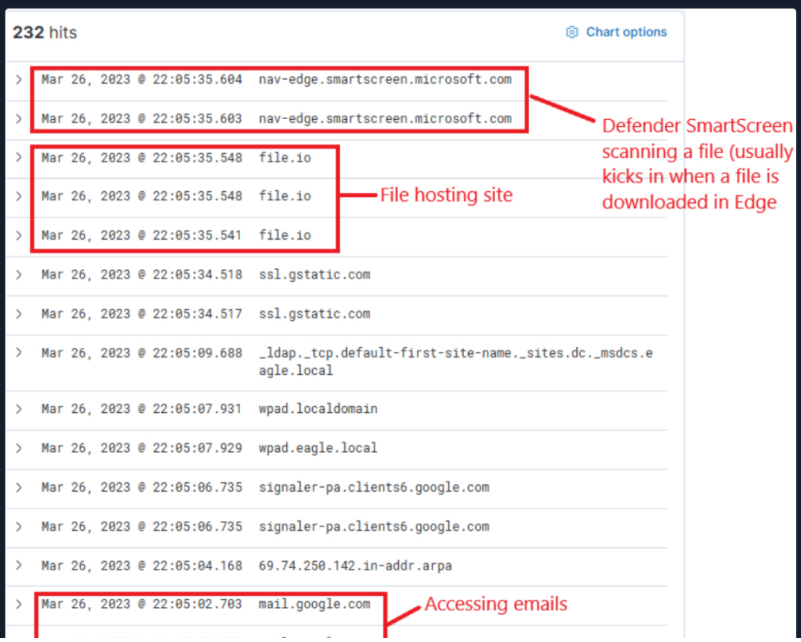
As part of our search process, since we're interested in DNS names, we'd like to display only the `dns.question.name` field in the result table. Please note the specified time `March 26th 2023 @ 22:05:00` to `March 26th 2023 @ 22:05:48`.



232 hits

Time ↓	dns.question.name
Mar 26, 2023 @ 22:05:47.676	ad-delivery.net
Mar 26, 2023 @ 22:05:47.676	ad-delivery.net
Mar 26, 2023 @ 22:05:47.553	crt.usertrust.com
Mar 26, 2023 @ 22:05:47.237	track.venatusmedia.com
Mar 26, 2023 @ 22:05:47.236	track.venatusmedia.com
Mar 26, 2023 @ 22:05:47.193	52.208.241.202.in-addr.arpa

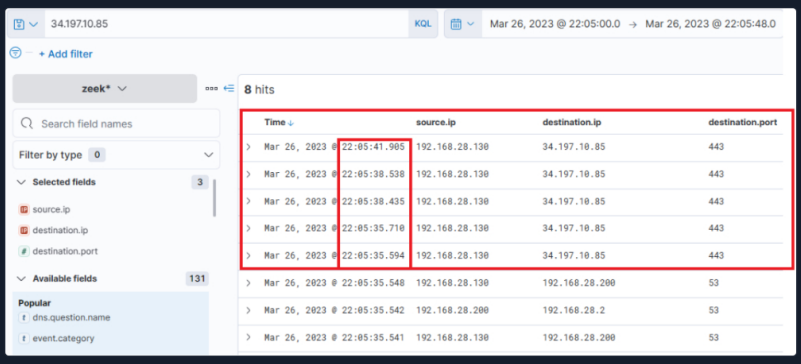
Scrolling down the table of entries, we observe the following activities.



From this data, we infer that the user accessed Google Mail, followed by interaction with "file.io", a known hosting provider. Subsequently, Microsoft Defender SmartScreen initiated a file scan, typically triggered when a file is downloaded via Microsoft Edge. Expanding the log entry for file.io reveals the returned IP addresses (`dns.answers.data` or `dns.resolved_ip` or `zeek.dns.answers` fields) as follows.

`34.197.10.85`, `3.213.216.16`

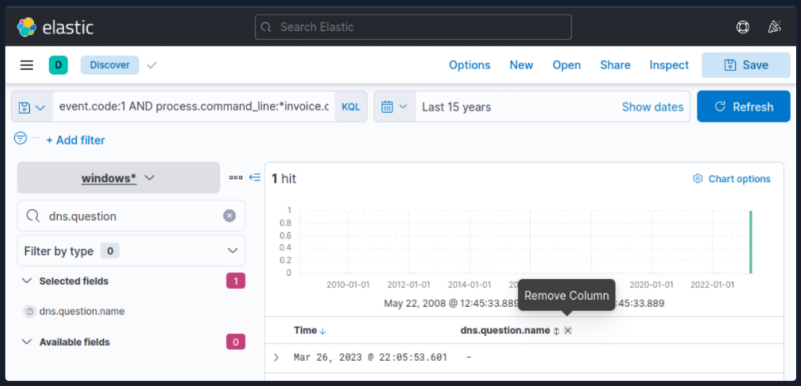
Now, if we run a search for any connections to these IP addresses during the same timeframe as the DNS query, it leads to the following findings.



This information corroborates that a user, Bob, successfully downloaded the file "invoice.one" from the hosting provider "file.io".

At this juncture, we have two choices: we can either cross-reference the data with the Threat Intel report to identify overlapping information within our environment, or we can conduct an Incident Response (IR)-like investigation to trace the sequence of events post the OneNote file download. We choose to proceed with the latter approach, tracking the subsequent activities.

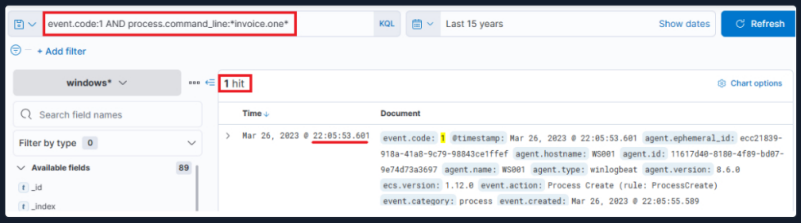
Hypothetically, if "invoice.one" was accessed, it would be opened with the OneNote application. So, the following query will flag the event, if it transpired. **Note**: The time frame we specified previously should be removed, setting it to, say, 15 years again. The `dns.question.name` column should also be removed.



Related fields: winlog.event_id or event.code and process.command_line

Hunting For Stuxbot

```
event.code:1 AND process.command_line:*invoice.one*
```



Indeed, we find that the OneNote file was accessed shortly after its download, with a delay of roughly 6 seconds. Now, with OneNote.exe in operation and the file open, we can speculate that it either contains a malicious link or a malevolent file attachment. In either case, OneNote.exe will initiate either a browser or a malicious file. Therefore, we should scrutinize any new processes where OneNote.exe is the parent process. The corresponding query is the following. Sysmon Event ID 1 (Process creation) is utilized.

Related fields: winlog.event_id or event.code and process.parent.name

Hunting For Stuxbot

```
event.code:1 AND process.parent.name:"ONENOTE.EXE"
```

The results of this query present three hits. However, one of these (the bottom one) falls outside the relevant time frame and can be dismissed. Evaluating the other two results:

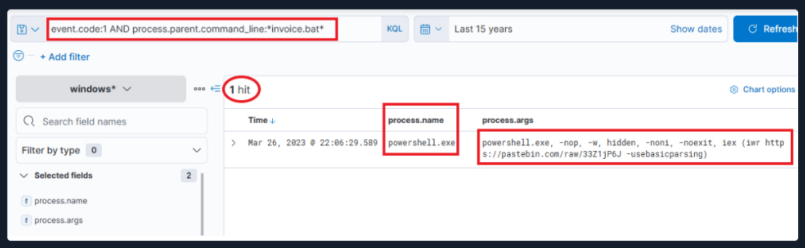
- The middle entry documents (when expanded) a new process, OneNoteM.exe, which is a component of OneNote and assists in launching files.
- The top entry reveals "cmd.exe" in operation, executing a file named "invoice.bat". Here is the view upon expanding the log.

Now we can establish a connection between "OneNote.exe", the suspicious "invoice.one", and the execution of "cmd.exe" that initiates "invoice.bat" from a temporary location (highly likely due to its attachment inside the OneNote file). The question now is, has this batch script instigated anything else? Let's search if a parent process with a command line argument pointing to the batch file has spawned any child processes with the following query.

Related fields: winlog.event_id or event.code and process.parent.command_line

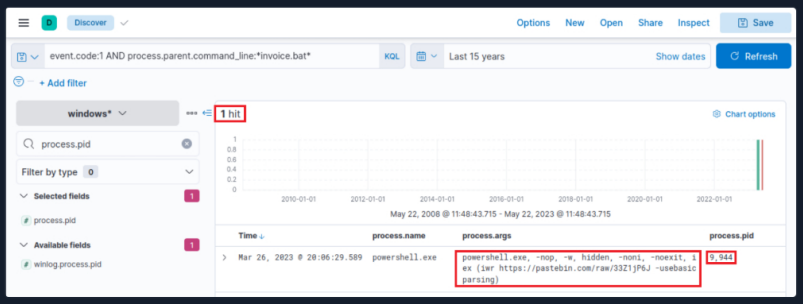
Hunting For Stuxbot

```
event.code:1 AND process.parent.command_line:*invoice.bat*
```



This query returns a single result: the initiation of PowerShell, and the arguments passed to it appear

conspicuously suspicious (note that we have added `process.name`, `process.args`, and `process.pid` as columns)! A command to download and execute content from Pastebin, an open text hosting provider! We can try to access and see if the content, which the script attempted to download, is still available (by default, it won't expire!).



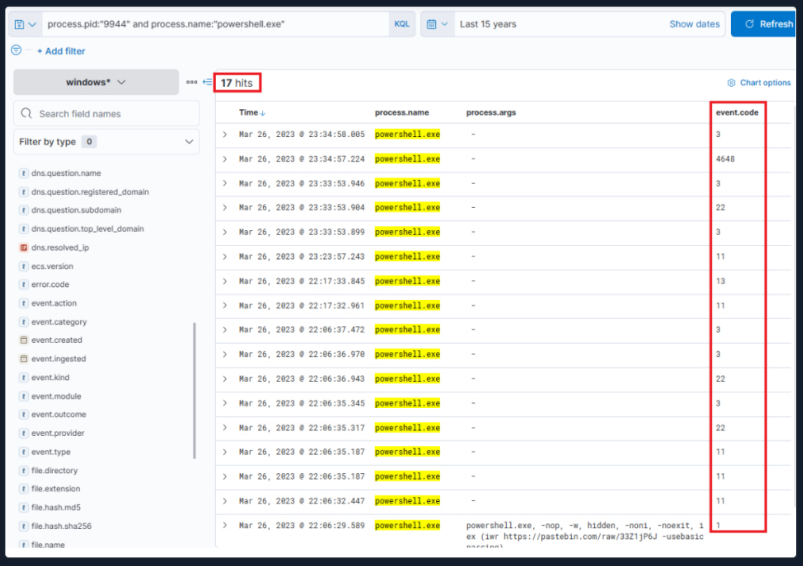
Indeed, it is! This is referred to in the Threat Intelligence report, stating that a PowerShell Script from Pastebin was downloaded.

To figure out what PowerShell did, we can filter based on the process ID and name to get an overview of activities. Note that we have added the `event.code` field as a column.

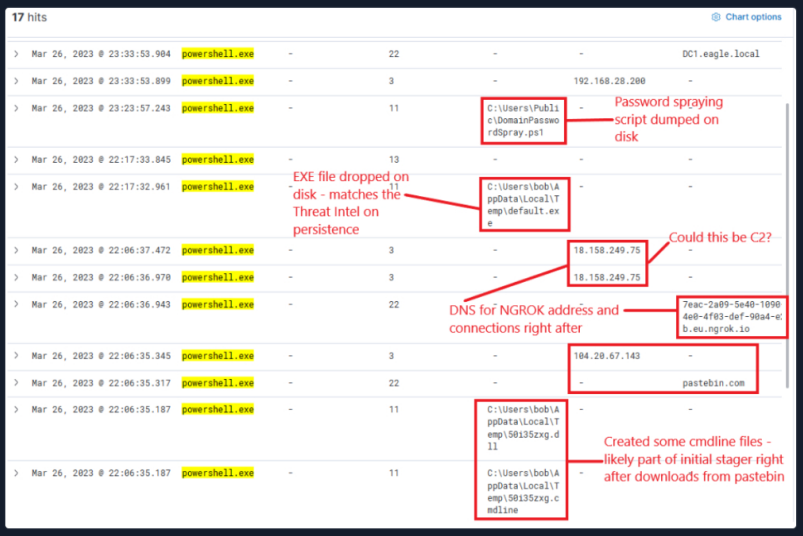
Related fields: process.pid and process.name

Hunting For Stuxbot

```
process.pid:"9944" and process.name:"powershell.exe"
```



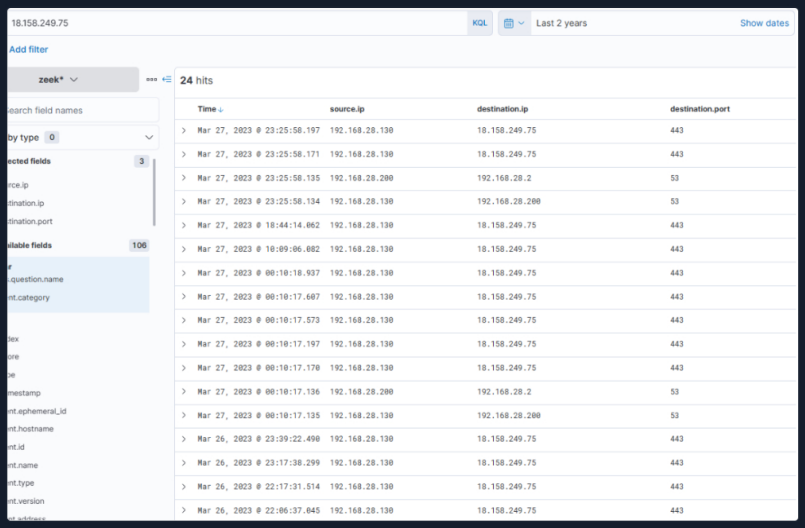
Immediately, we can observe intriguing output indicating file creation, attempted network connections, and some DNS resolutions leverarging Sysmon Event ID 22 (DNSEvent). By adding some additional informative fields (`file.path`, `dns.question.name`, and `destination.ip`) as columns to that view, we can expand it.



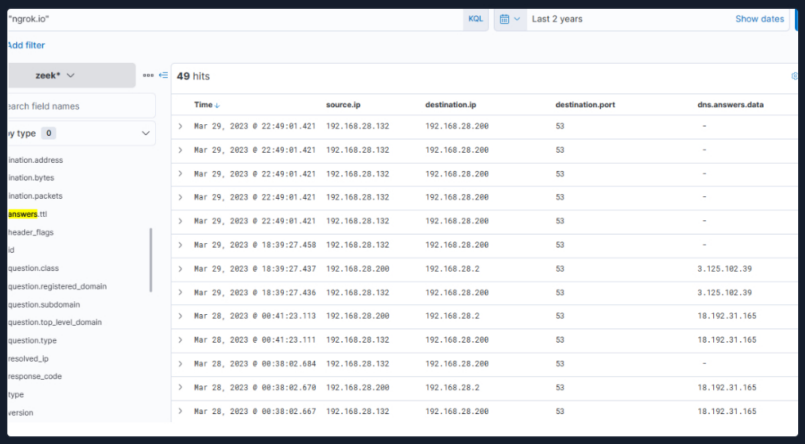
Now, this presents us with rich data on the activities. Ngrok was likely employed as C2 (to mask malicious traffic to a known domain). If we examine the connections above the DNS resolution for Ngrok, it points to the destination IP Address 443, implying that the traffic was encrypted.

The dropped EXE is likely intended for persistence. Its distinctive name should facilitate determining whether it was ever executed. It's important to note the timestamps – there is some time lapse between different activities, suggesting it's less likely to have been scripted but perhaps an actual human interaction took place (unless random sleep occurred between the executed actions). The final actions that this process points to are a DNS query for DC1 and connections to it.

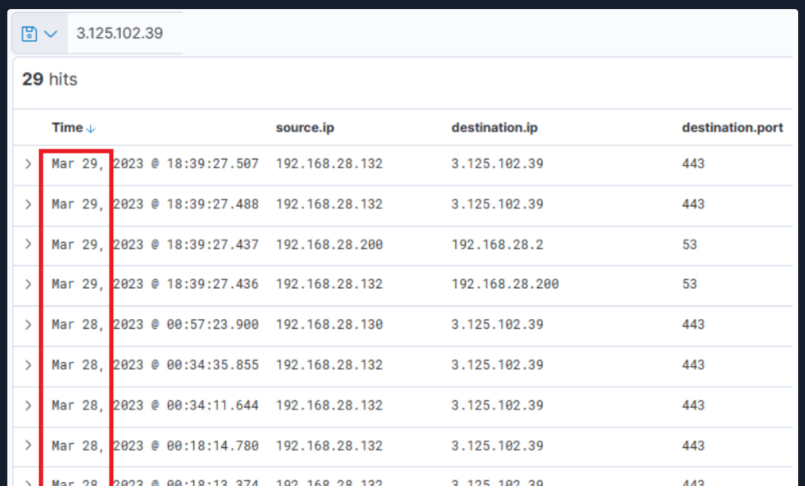
Let's review Zeek data for information on the destination IP address `18.158.249.75` that we just discovered. Note that the `source.ip`, `destination.ip`, and `destination.port` fields were added as columns.



Intriguingly, the activity seems to have extended into the subsequent day. The reason for the termination of the activity is unclear... Was there a change in C2 IP? Or did the attack simply halt? Upon inspecting DNS queries for "ngrok.io", we find that the returned IP (`dns.answers.data`) has indeed altered. Note that the `dns.answers.data` field was added as a column.



The newly discovered IP also indicates that connections continued consistently over the following days.



Mar 28, 2023 @ 00:18:13.353	192.168.28.132	3.125.102.39	443
Mar 28, 2023 @ 00:18:13.241	192.168.28.132	3.125.102.39	443
Mar 28, 2023 @ 00:18:12.783	192.168.28.132	3.125.102.39	443
Mar 28, 2023 @ 00:18:12.723	192.168.28.132	3.125.102.39	443
Mar 28, 2023 @ 00:18:12.641	192.168.28.200	192.168.28.2	53
Mar 28, 2023 @ 00:18:12.640	192.168.28.132	192.168.28.200	53
Mar 28, 2023 @ 00:11:27.043	192.168.28.130	3.125.102.39	443
Mar 27, 2023 @ 23:23:31.933	192.168.28.130	3.125.102.39	443

Thus, it's apparent that there is sustained network activity, and we can deduce that the C2 has been accessed continually. Now, as for the earlier uploaded executable file "default.exe" – did that ever execute? By probing the Sysmon logs for a process with that name, we can ascertain this. Note that the `process.name`, `process.args`, `event.code`, `file.path`, `destination.ip`, and `dns.question.name` fields were added as columns.

Related field: process.name

Hunting For Stuxbot

```
process.name:"default.exe"
```

process.name:"default.exe"

68 hits

Chart options

Mar 27, 2023 @ 22:58:04.493	default.exe	-	3	-	3.125.102.39	-
Mar 27, 2023 @ 22:58:04.463	default.exe	-	22	-	-	7eac-2a09-5e40-1090-4e0-4f03-def-90a4-e.b.eu.ngrok.io
Mar 27, 2023 @ 18:44:14.684	default.exe	-	5	-	-	-
Mar 27, 2023 @ 09:44:01.496	default.exe	-	3	-	3.125.223.134	-
Mar 27, 2023 @ 09:44:01.484	default.exe	-	22	-	-	7eac-2a09-5e40-1090-4e0-4f03-def-90a4-e.b.eu.ngrok.io
Mar 27, 2023 @ 00:17:01.805	default.exe	-	11	C:\Users\Public\SharpHound.exe	-	-
Mar 27, 2023 @ 00:12:44.594	default.exe	-	13	-	-	-
Mar 27, 2023 @ 00:12:43.663	default.exe	-	11	C:\Users\bob\AppData\Local\Temp\svchost.exe	-	-
Mar 27, 2023 @ 00:10:19.033	default.exe	-	3	-	18.158.249.75	-
Mar 27, 2023 @ 00:10:18.596	default.exe	-	3	-	18.158.249.75	-
Mar 27, 2023 @ 00:10:18.566	default.exe	-	22	-	-	7eac-2a09-5e40-1090-4e0-4f03-def-90a4-e.b.eu.ngrok.io
Mar 27, 2023 @ 00:10:18.246	default.exe	C:\Users\bob\AppData\Local\Temp\default.exe	1	-	-	-

Indeed, it has been executed – we can instantly discern that the executable initiated DNS queries for Ngrok and established connections with the C2 IP addresses. It also uploaded two files "svchost.exe" and "SharpHound.exe". SharpHound is a recognized tool for diagramming Active Directory and identifying attack paths for escalation. As for svchost.exe, we're unsure – is it another malicious agent? The name implies it attempts to mimic the legitimate svchost file, which is part of the Windows Operating System.

If we scroll up there's further activity from this executable, including the uploading of "payload.exe", a VBS file, and repeated uploads of "svchost.exe".

At this juncture, we're left with one question: did SharpHound execute? Did the attacker acquire information about Active Directory? We can investigate this with the following query (since it was an on-disk executable file).

Related field: process.name

Hunting For Stuxbot

```
process.name:"SharpHound.exe"
```

4 hits

Time ↓	process.name	process.args
Mar 27, 2023 @ 00:19:30.147	SharpHound.exe	-
Mar 27, 2023 @ 00:19:30.119	SharpHound.exe	Sharphound.exe, -c, all

>	Mar 27, 2023 @ 00:17:58.409	SharpHound.exe -
>	Mar 27, 2023 @ 00:17:58.000	SharpHound.exe sharphound.exe, -collectionmethod, all

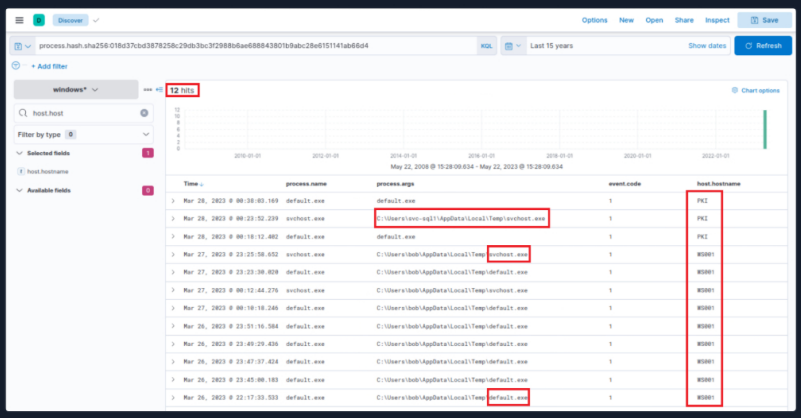
Indeed, the tool appears to have been executed twice, roughly 2 minutes apart from each other.

It's vital to note that Sysmon has flagged "default.exe" with a file hash (`process.hash.sha256` field) that aligns with one found in the Threat Intel report. This leads us to question whether this executable has been detected on other devices within the environment. Let's conduct a broad search. Note that the `host.hostname` field was added as a column.

Related field: process.hash.sha256

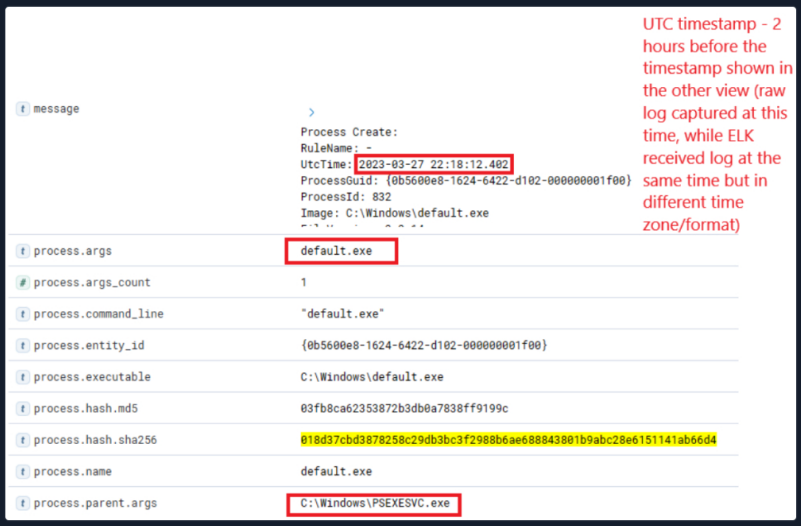
Hunting For Stuxbot

```
process.hash.sha256:018d37cbd3878258c29db3bc3f2988b6ae688843801b9abc28e6151141ab66d4
```



Files with this hash value have been found on WS001 and PKI, indicating that the attacker has also breached the PKI server at a minimum. It also appears that a backdoor file has been placed under the profile of user "svc-sql1", suggesting that this user's account is likely compromised.

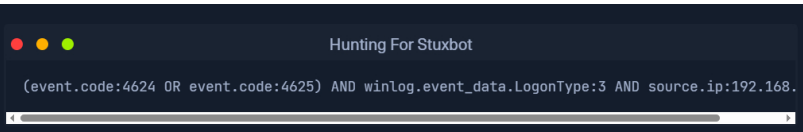
Expanding the first instance of "default.exe" execution on PKI, we notice that the parent process was "PSEXESVC", a component of PSExec from SysInternals – a tool often used for executing commands remotely, frequently utilized for lateral movement in Active Directory breaches.

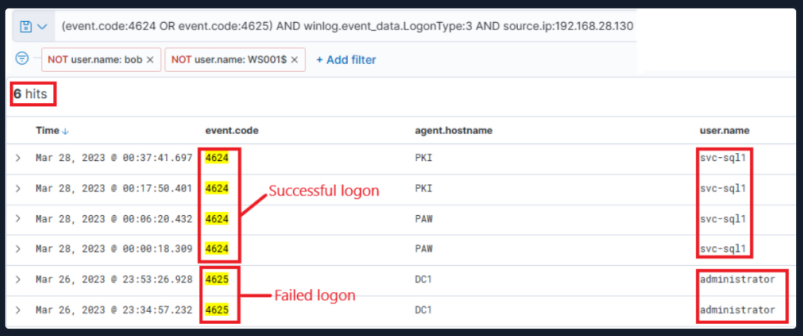


Further down the same log, we notice "svc-sql1" in the `user.name` field, thereby confirming the compromise of this user.

How was the password of "svc-sql1" compromised? The only plausible explanation from the available data so far is potentially the earlier uploaded PowerShell script, seemingly designed for Password Bruteforcing. We know that this was uploaded on WS001, so we can check for any successful or failed password attempts from that machine, excluding those for Bob, the user of that machine (and the machine itself).

Related fields: winlog.event_id or event.code, winlog.event_data.LogonType, and source.ip

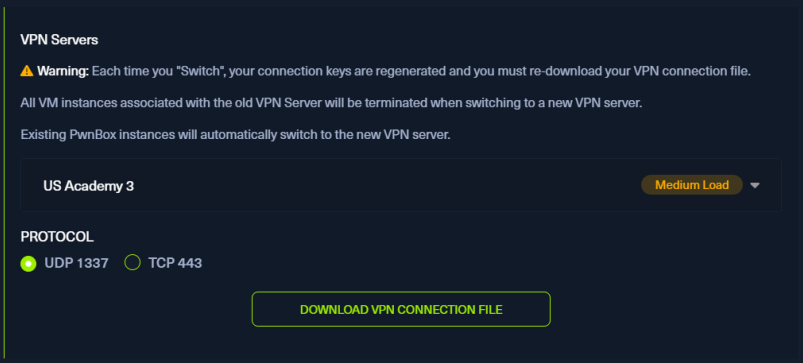


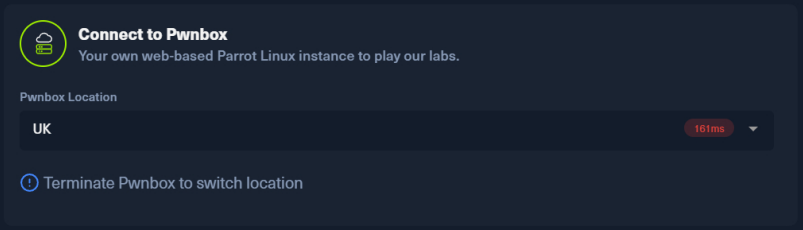


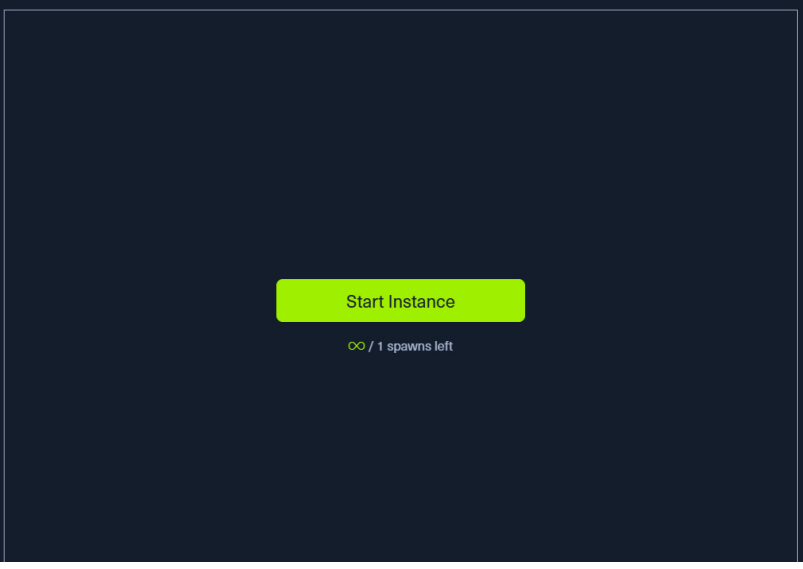
The results are quite intriguing – two failed attempts for the administrator account, roughly around the time when the initial suspicious activity was detected. Subsequently, there were numerous successful logon attempts for "svc-sql1". It appears they attempted to crack the administrator's password but failed. However, two days later on the 28th, we observe successful attempts with svc-sql1.

At this stage, we have amassed a significant amount of information to present and initiate a comprehensive incident response, in accordance with company policies.

Please allow 3-5 minutes for Kibana to become available after spawning the target of the questions below.







Waiting to start...

Enable step-by-step solutions for all questions

Questions

Answer the question(s) below to complete this Section and earn cubes!

Download VPN Connection File

Target(s): Click here to spawn the target system!

+ 2 Navigate to http://[Target IP]:5601 and follow along as we hunt for Stuxbot. In the part where default.exe is under investigation, a VBS file is mentioned. Enter its full name as your answer, including the extension.

XceGuhkzaTrOy.vbs

Submit

+ 1 Stuxbot uploaded and executed mimikatz. Provide the process arguments (what is after .\mimikatz.exe, ...) as your answer.

lsadump::dcsync /domain:eagle.local /all /csv, exit

Submit

+ 1 Some PowerShell code has been loaded into memory that scans/targets network shares. Leverage the available PowerShell logs to identify from which popular hacking tool this code derives. Answer format (one word): P____V___

Powerview

Submit Hint

Previous Next

Mark Complete & Next