Threat Thursday: WhisperGate Wiper Targets Government, Non-profit, and IT Organizations in Ukraine

blogs.blackberry.com/en/2022/01/threat-thursday-whispergate-wiper

The BlackBerry Research & Intelligence Team

Earlier this month, a potent wiper named WhisperGate targeted government, non-profit, and IT organizations in Ukraine with what initially appeared to be ransomware. The threat was in fact a destructive wiper malware, carried out as a multi-stage attack. Thus far, it has been seen targeting devices running the Windows operating system; however, the Ukrainian government reported that there might also be a Linux variant of the malware lurking out there too.

WhisperGate runs as a multi-stage attack, beginning by overwriting the Master Boot Record (MBR) and displaying a fake ransom note. The second and third stages involve retrieving the payload from a malicious discord link. The final stage executes a file corruptor against target file types, irrecoverably destroying data.

No encryption has been observed during any of the attack stages. The malware does not appear to have been created for financial gain. Thus far, the sole purpose of WhisperGate seems to be the destruction of data, making it most likely that a threat group is using this malware to disrupt or disable target organizations.

Given the escalating geopolitical events in Ukraine and its surrounding regions, BlackBerry strongly encourages organizations with an elevated risk profile to use the information in this blog to proactively defend against any malicious activity from this group.

Operating System

Windows	MacOS	Linux	Android
Yes	No	No	No

Risk & Impact

Impact	High
Risk	Low

Technical Analysis

Stage 1 – MBR Wiper

The initial stage of this attack involves the malware gaining access to, then overwriting, the Master Boot Record (MBR) to display a fake ransom note the next time the system is booted. The MBR is used to identify how and where an operating system is located so that it can be loaded correctly. Modification of the MBR can result in the system becoming inoperable.

According to Microsoft's initial report, the threat actor uses the Python tool Impacket to perform lateral movement and execute the malware. This is done using the following command.

> *"cmd.exe /Q /c start c:\stage1exe 1> \\127.0.0.1\ADMIN$_[TIMESTAMP] 2>&1" .*

SHA256: a196c6b8ffcb97ffb276d04f354696e2391311db3841ae16c8c9f56f36a38e92

The initial file used by attackers is a Win32 binary called "stage1.exe." When launched with administrative privileges, the malware will load an 8192-byte buffer onto the stack. The process will retrieve a handle to the MBR by calling the "CreateFileW" function, as seen in the image below.

```
buffer = a1;
sub_401FE0(8236u, (int)&dwCreationDisposition, (unsigned int)&dwCreationDisposition);
v1 = alloca(8236);
sub_401990();
qmemcpy(&buffer - 2054, &MBR_data, 8192u);
file_handle = CreateFileW(
                L"\\\\.\\PhysicalDrive0",
                GENERIC_ALL,
                3u,
                (LPSECURITY_ATTRIBUTES)NO_INHERITANCE,
                OPEN_EXISTING,
                0,
                0);
WriteFile(file_handle, &buffer - 2054, 512u, 0, 0);
CloseHandle(file_handle);
return 0;
```

Figure 1 - Buffer and handle being created and written to disk

The malware will then overwrite the MBR with this handle that contains the fake ransom note and a shellcode used to display it on the victim's machine. As can be seen in Figure 2, a file of 512 bytes is written into the MBR in the first sector of the physical drive.

Time ...	Process Name	PID	Operation	Path
16:53:...	A196C6B8FFC...	2152	CreateFile	\Device\Harddisk0\DR0
16:53:...	A196C6B8FFC...	2152	WriteFile	\Device\Harddisk0\DR0
16:53:...	A196C6B8FFC...	2152	CloseFile	\Device\Harddisk0\DR0

Figure 2 – Buffer handle overwriting the MBR with fake ransom note

This buffer contains a ransom note (as seen in Figure 3) that is shown to the victim upon the next system reboot. The note contains a bitcoin wallet address and an ID for the Tox instant message system, along with a demand for $10,000 to recover the data. However, this note appears to be pure subterfuge, since the victim's MBR has been overwritten and partitions already destroyed.

```
Your hard drive has been corrupted.
In case you want to recover all hard drives
of your organization,
You should pay us  $10k via bitcoin wallet
1AVNM68gj6PGPFcJuftKATa4WLnzg8fpfv and send message via
tox ID 8BEDC411012A33BA34F49130D0F186993C6A32DAD8976F6A5D82C1ED23054C057ECED5496
F65
with your organization name.
We will contact you to give further instructions._
```

Figure 3 - Fake ransom note displayed to victim after system reboot

WhisperGate does not force a reboot; instead, it waits for victims to reboot their systems themselves. The delayed reboot allows the malware time to launch additional stages of the attack chain, as detailed below.

Stage 2 – Downloader

SHA256: dcbbae5a1c61dbbbb7dcd6dc5dd1eb1169f5329958d38b58c3fd9384081c9b78

The second stage of this attack involves the use of a .NET malware. This file is a downloader used to create an HTTP connection with a hardcoded Discord content delivery network link. This link (seen in Figure 4) hosts a JPG file called "Tbopbh.jpg," which contains the malicious payload required to initiate stage 3 of this attack. This file has the same name as the decompiled assembly of the second stage loader.

Figure 4 - Discord download link for third stage malware

Before reaching out to the Discord server, the file utilizes PowerShell (shown in Figure 5) to launch a Base64 encoded command that performs a sleep function for 20 seconds. This delay aids the malware in remaining undetected on the target machine.

```
77          Facade.InitItem(Facade.SetItem(new ProcessStartInfo
78          {
79              FileName = "powershell",
80              Arguments = Facade.SearchItem("-enc UwB0AGEAcgB0AC", text2),
81              WindowStyle = ProcessWindowStyle.Hidden
```

Figure 5 - PowerShell used to perform sleep function for 20 seconds

The loader creates a connection with the Discord server hosted at 162[.]159[.]135[.]233 as shown below, in order to retrieve the next stage.

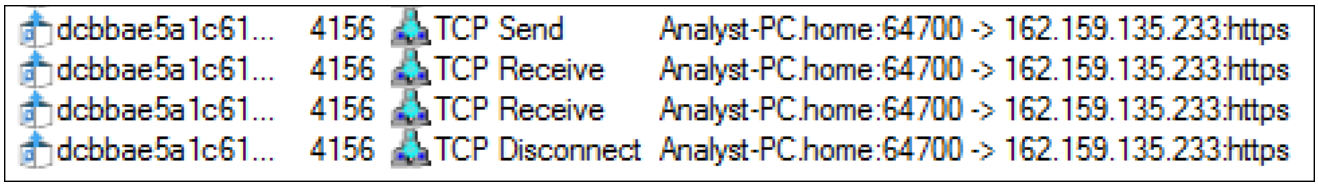

Figure 6 - Stage 2 loader creates TCP connection with Discord server

Stage 3 – Injector

SHA256: 923eb77b3c9e11d6c56052318c119c1a22d11ab71675e6b95d05eeb73d1accd6

The retrieved payload for Stage 3 is a .NET assembly file, which is obfuscated by using the reverse order of each byte in the file. When the byte order of the payload has been reversed, it results in a Win32 DLL (Dynamic Link Library) file called “Frkmlkdkdubkznbkmcf.dll.” The assembly contains three resources, as shown in Figure 7.

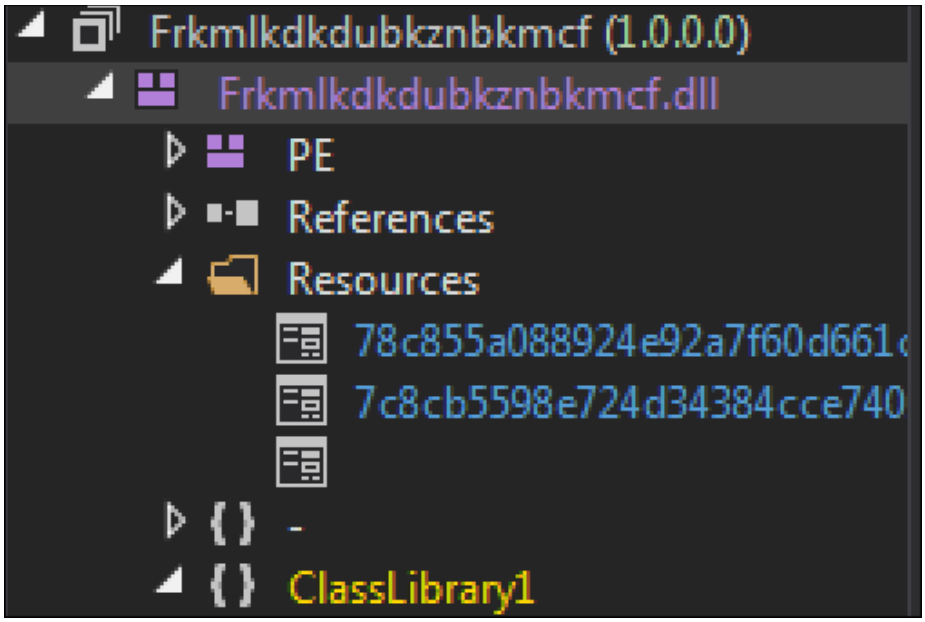

Figure 7 - Assembly containing three resources

The resource “78c855a088924e92a7f60d661c3d1845” is loaded into memory and is decoded by an XOR operation. The result of this is another .NET DLL file called “zx_fee6cce9db1d42510801fc1ed0e09452.dll.” This decoded file contains two additional resources called “AdvancedRun” and “Waqybg.”

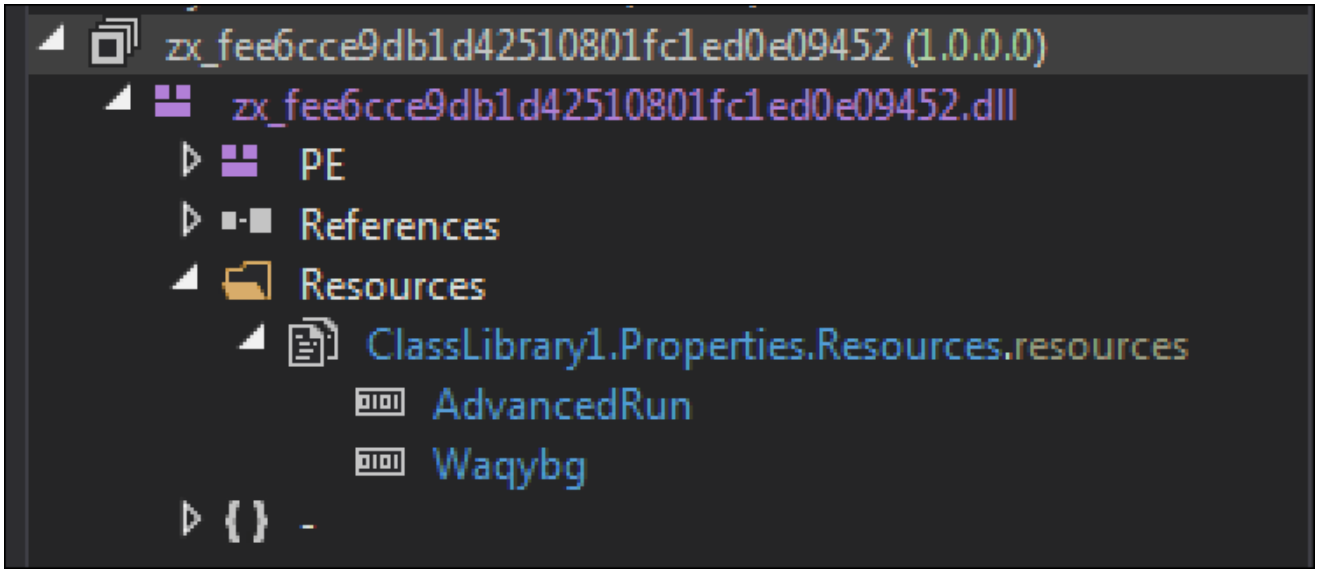

Figure 8 - Additional Resources "AdvancedRun" and "Waqybg"

The resource “AdvancedRun” is used to stop Windows Defender from running, and to delete it from memory. It does this by invoking PowerShell and executing multiple commands.

Stage 4 – File Corruptor

SHA256: 34ca75a8c190f20b8a7596afeb255f2228cb2467bd210b2637965b61ac7ea907

The final stage of this malware is the file corruptor stage. Here the resource “Waqybg” is executed and begins overwriting the start of each targeted file with 1MB of static data. During analysis, this resulted in hard disk space filling up rapidly and the virtual machine (VM) used by our researchers crashing.

The following is the list of the hardcoded file extensions that this malware targets. Once the file is encrypted, the file corruptor will append a random four-byte extension to the file name: for example, “File.1234.”

.HTML .HTM .PHTML .PHP .JSP .ASP .PHPS .PHP5 .ASPX .PHP4 .PHP3 .DOC .DOCX .XLS .XLSX .PPT .PPTX .PST .MSG .EML .TXT .CSV .RTF .WKS .WK1 .PDF .DWG .JPEG .JPG .DOCM .DOT .DOTM .XLSM .XLSB .XLW .XLT .XLM .XLC .XLTX .XLTM .PPTM .POT .PPS .PPSM .PPSX .HWP .SXI .STI .SLDX .SLDM .BMP .PNG .GIF .RAW .TIF .TIFF .PSD .SVG .CLASS .JAR .SCH .VBS .BAT .CMD .ASM .PAS .CPP .SXM .STD .SXD .ODP .WB2 .SLK .DIF .STC .SXC .ODS .3DM .MAX .3DS .STW .SXW .ODT .PEM .P12 .CSR .CRT .KEY .PFX .DER .OGG .JAVA .INC .INI .PPK .LOG .VDI .VMDK .VHD .MDF .MYI .MYD .FRM .SAV .ODB .DBF .MDB .ACCDB .SQL .SQLITEDB .SQLITE3 .LDF .ARC .BAK .TAR .TGZ .RAR .ZIP .BACKUP .ISO .CONFIG

Once the file corruptor has completed overwriting the targeted files, it will execute the following ping command to remove itself from the machine.

“cmd.exe /min /C ping 111.111.111.111 -n 5 -w 10 > Nul & Del /f /q \”[Filepath]\”

Conclusion

The multi-stage attack method and stealth techniques of WhisperGate make it a unique piece of very destructive malware. Through analysis, it becomes apparent that the goal of the malware is to cause disruption at the targeted organization, rather than pursuit of any financial gain.

As this is still an ongoing investigation and BlackBerry’s analysis continues, we will share more information as soon as it becomes available. This will likely not be the last we hear about WhisperGate – especially if the possible existence of a Linux version of the wiper is confirmed. Stay tuned for further updates in next week’s blog.

YARA Rule

The following YARA rule was authored by the BlackBerry Research & Intelligence Team to catch the threat described in this document:

```
rule Stage_1

{
   meta:
      description = "Detects Stage 1 WhisperGate"
      author = "BlackBerry Threat Research"
      date = "2022-01-17"
      license = "This Yara rule is provided under the Apache License 2.0
(https://www.apache.org/licenses/LICENSE-2.0) and open to any user or organization, as long as
you use it under this license and ensure originator credit in any derivative to The BlackBerry
Research & Intelligence Team"

      strings:
         // Regex for ransom note in MBR
         $re1 = /You should pay us \$[0-9]{1,3}k via bitcoin wallet/
```

```
        // MBR signature
        $b1 = {55 AA}
        // Int 13 Bios interrupt
        $b2 = {CD 13}
        // Extended write interrupt code
        $b3 = {B4 43}

        // Strings in MBR replacement data
        $s1 = "Your hard drive has been corrupted."
        $s2 = "tox ID"
        $s3 = "We will contact you to give further instructions."

        // String used in CreateFileW call to open main physical drive
        $s4 = "\\\\.\\PhysicalDrive0" wide

        // The hardcoded data it uses to overwrite the MBR

    condition:
        // MZ header
        uint16(0) == 0x5a4d and

        ((all of ($b*) and 2 of ($s*)) or (all of ($s*) and $re1))
}

import "pe"

rule Stage2
{
    meta:
        description = "Detects WhisperGate Stage 2"
        author = "BlackBerry Threat Research"
        date = "2022-01-17"
        license = "This Yara rule is provided under the Apache License 2.0
(https://www.apache.org/licenses/LICENSE-2.0) and open to any user or organization, as long as
you use it under this license and ensure originator credit in any derivative to The BlackBerry
Research & Intelligence Team"

    strings:
        // Regex for any attachment link coming from the same discord channel
        $re1 = /https:\/\/cdn\.discordapp\.com\/attachments\/928503440139771947\/[0-9]{18}\/[^<>:"\/\|?
*]*\b/ wide

        // 3 strings combined to make powershell command that evaluates to "Start-Sleep -s 10"
        $s1 = "powershell" wide
        $s2 = "-enc UwB0AGEAcgB0AC" wide
        $s3 = "0AUwBsAGUAZQBwACAALQBzACAAMQAwAA==" wide

        // The method name being looked for in stage 3
        $s4 = "Ylfwdwgmpilzyaph" wide

        // Used with .Replace("x", "") for reflection on WebClient to get DownloadData method
        $s5 = "DxownxloxadDxatxxax" wide

    condition:
        // MZ header
        uint16(0) == 0x5a4d and
        // is a .NET binary
        pe.data_directories[pe.IMAGE_DIRECTORY_ENTRY_COM_DESCRIPTOR].size != 0 and
```

```
    $re1 and 2 of ($s*)
}

rule Stage3
{
  meta:
    description = "Detects WhisperGate Stage 3"
    author = "BlackBerry Threat Research"
    date = "2022-01-17"
    license = "This Yara rule is provided under the Apache License 2.0
(https://www.apache.org/licenses/LICENSE-2.0) and open to any user or organization, as long as
you use it under this license and ensure originator credit in any derivative to The BlackBerry
Research & Intelligence Team"

  strings:
    // The same as a string from stage2, not sure what it is
    $s1 = "Ylfwdwgmpilzyaph"
    // Names of resources
    $s3 = "78c855a088924e92a7f60d661c3d1845"
    $s4 = "7c8cb5598e724d34384cce7402b11f0e"

  condition:
    // MZ header
    uint16(0) == 0x5a4d and
    // is a .NET binary
    pe.data_directories[pe.IMAGE_DIRECTORY_ENTRY_COM_DESCRIPTOR].size != 0 and

    all of them
}

import "pe"

rule Stage3_reversed
{
  meta:
    description = "Detects WhisperGate Stage 3 Reversed"
    author = "BlackBerry Threat Research"
    date = "2022-01-17"
    license = "This Yara rule is provided under the Apache License 2.0
(https://www.apache.org/licenses/LICENSE-2.0) and open to any user or organization, as long as
you use it under this license and ensure originator credit in any derivative to The BlackBerry
Research & Intelligence Team"

  strings:
    // This string was also in stage2 so defo want to look into it
    // but this is the reversed version because Stage3 is backwards
    $s1 = "hpayzlipmgwdwflY"
    // Names of resources backwards
    $s3 = "5481d3c166d06f7a29e429880a558c87"
    $s4 = "e0f11b2047ecc48343d427e8955bc8c7"

  condition:
    // MZ header backwards at the end of the file
    uint16(filesize-2) == 0x4d5a and

    all of them
}
```

Indicators of Compromise (IoCs)

Files

a196c6b8ffcb97ffb276d04f354696e2391311db3841ae16c8c9f56f36a38e92 (Stage 1)
dcbbae5a1c61dbbbb7dcd6dc5dd1eb1169f5329958d38b58c3fd9384081c9b78 (Stage 2)
923eb77b3c9e11d6c56052318c119c1a22d11ab71675e6b95d05eeb73d1accd6 (Stage 3)
9ef7dbd3da51332a78eff19146d21c82957821e464e8133e9594a07d716d892d (Stage 3 Reversed)
34ca75a8c190f20b8a7596afeb255f2228cb2467bd210b2637965b61ac7ea907 (File Wiper)
35feefe6bd2b982cb1a5d4c1d094e8665c51752d0a6f7e3cae546d770c280f3a (Decoded Resource)

Bitcoin Wallet Address

1AVNM68gj6PGPFcJuftKATa4WLnzg8fpfv

ToxID

8BEDC411012A33BA34F49130D0F186993C6A32DAD8976F6A5D82C1ED23054C057ECED5496F65

Discord Link

https[:]//cdn.discordapp[.]com/attachments/928503440139771947/930108637681184768/Tbopbh.jpg

IP Address

162[.]159[.]135[.]233

BlackBerry Assistance

If you're battling this malware or a similar threat, you've come to the right place, regardless of your existing BlackBerry relationship.

The BlackBerry Incident Response team is made up of world-class consultants dedicated to handling response and containment services for a wide range of incidents, including ransomware and Advanced Persistent Threat (APT) cases.

We have a global consulting team standing by to assist you providing around-the-clock support, where required, as well as local assistance. Please contact us here: https://www.blackberry.com/us/en/forms/cylance/handraiser/emergency-incident-response-containment

About The BlackBerry Research & Intelligence Team

The BlackBerry Research & Intelligence team examines emerging and persistent threats, providing intelligence analysis for the benefit of defenders and the organizations they serve.

Back