Diavol - A New Ransomware Used By Wizard Spider?

fortinet.com/blog/threat-research/diavol-new-ransomware-used-by-wizard-spider

July 1, 2021

FortiGuard Labs Threat Research Report

Affected Platforms:	Windows
Impact:	Data encryption, Data destruction
Threat Severity:	Critical

Diavol Introduction

At the beginning of June, FortiEDR prevented a ransomware attack that had targeted one of our customers. After successfully stopping the attack, we were able to isolate two suspicious files that, at the time, were not found on VirusTotal: *locker.exe* and *locker64.dll*. In the timeline of the attack, locker.exe was deployed a day before locker64.dll.

While we were able to identify locker64.dll to be a Conti (v3) ransomware, locker.exe appeared to be entirely different. So, let's say hello to a new ransomware family.

In this blog, we'll dive into the inner workings of Diavol and its possible attribution to the criminal group known as Wizard Spider.

First Encounter with Diavol

The ransomware drops a ransom note in a text format in every folder it goes over, as can be seen in figure 1.

Figure 1: The dropped "README_FOR_DECRYPT.txt" ransom note.

According to the note, the authors claim they stole data from the victim's machine, though we did not find a sample that was capable of performing that. This is either a bluff or a placeholder for future capabilities. Browsing to the URL led to us a website, seen in figures 2 and 3, from which we derived the name for the ransomware.

Figures 2 and 3: Diavol's website using Tor browser.

As part of a rather unique encryption procedure, Diavol operates using user-mode Asynchronous Procedure Calls (APCs) without a symmetric encryption algorithm. Usually, ransomware authors aim to complete the encryption operation in the shortest amount of time. Asymmetric encryption algorithms are not the obvious choice as they significantly slower than symmetric algorithms.

Technical Analysis

locker.exe is a 32-bit executable compiled with Microsoft Visual C/C++ Compiler. The timestamp 2021-04-30 15:58:15 on the file supports the hypothesis that this ransomware is relatively new.

Upon execution, Diavol starts by checking the command line arguments:

"*-p*": path to a file with a list of paths to scan first for files.
"*-log*": path to a log file.
"*-m*": mode: *net* or *local*.
net- encrypt network shares only.
local- encrypt local drives only and ignore network shares.
"*-h*": path to a file that contains specific hosts (names and IPs) to enumerate for shares.
"*-s*": IP address that the initial register message will be sent to. Overrides the hardcoded address.

The following command-line parameters were observed in the incident:

-p "C:\b.txt" -m local -log "C:\programdata\log.txt"

The log file lists files that were encrypted.

The authors didn't remove the path to the debugging information file, unfortunately it didn't divulge sensitive or incriminating details about the ransomware authors:

D:\Development\Master\onion\locker.divided\LockMainDIB\Release\LockMainDIB.pdb

While Diavol is not packed nor has any anti-disassembly tricks, it does use an interesting anti-analysis technique to obfuscate its code. Its main routines are kept in bitmap images, which are stored in the PE resource section. Before calling each routine, it copies the bytes from the bitmap to a global buffer that has execute permissions.

Figure 4: Diavol's internals routines stored as bitmaps.

The imports used by each routine are also stored in the resource section under "*OFFSET*", with the same names as the bitmaps.

Figure 5: Diavol's imports data for the REGISTER routines.

Diavol has 14 different routines stored as bitmaps. They are called in the following order:

Figure 6: Diavol's execution flow for invoking routines from the resource section.

· Create an identifier for the victim:

The *GENBOTID routine* creates a unique identifier of the infected machine. It is composed of the following:

<NetBIOS_computer_name> + <username> + "_W" + <OS major version in hex> + <OS minor version in hex> + <OS build number in hex> + "." + <random_GUID_bytes in hex>

· Initialize configuration:

The *SHAPELISTS* routine copy the hardcoded configuration from the PE's .data section. The configuration begins with the "*STATIC_DATA*" string and holds many unicode strings:

- Base64 encoded RSA public key.
- Server address for the initial registration.
- Group ID for the initial registration.
- List of excluded file extensions, file names and paths.
- List of process names to terminate.
- List of service names to stop.

- List of paths to enumerate files.
- List of filenames to delete.
- Ransom note (in reverse).

· Register with the C&C server and update configuration:

The *REGISTER* payload uses the WinINet API to send a request to a server and returns the response status code. Diavol issues the C&C server a POST request to the *hxxp://<server_address>/BnpOnspQwtjCA/register* URL with the following headers:

User-Agent: "Agent"
Content-Type: application/x-www-form-urlencoded; charset=UTF-8

The body of the request is*:*

cid=<unique_victim_id>&group=<group_id_from_config>ip_local1=111.111.111.111&ip_local2=222.222.222.222&ip_external=2.16.7.12

If the server returns status success, the ransomware will attempt to get an updated configuration from the C&C server. Since the "*-s*" command-line parameter was not provided, and as the configured address is 127.0.0.1 (localhost), the ransomware didn't register itself or retrieve configuration updates.

The *FROMNET* routine sends a request to the server using the WinINet API as well and returns the data from the response. To get an updated configuration and override the hardcoded values, the ransomware sends HTTP GET requests using the same headers as before, to:

hxxp://173[.]232[.]146[.]118/Bnyar8RsK04ug/<unique_victim_id>/qqq123/<object_name>

The following objects are supported:

- */key* - base64 encoded RSA public key.
- */services* - list of services to stop.
- */priority* - list of paths to scan for files for the first time.
- */ignore* - list of file extensions, filenames and paths to exclude.
- */ext* - list of file extensions to include.
- */wipe* - list of filenames to delete if they are found while enumerating the filesystem.
- */landing* - ransomware note.

Inspecting the network traffic to 173.232.146.118 showed that the HTTP Cookie header contains the string "*diavol_session*".

Figure 7: Network traffic to Diavol's C&C server.

· Stop services and processes:

To maximize its effect on the target machine and to encrypt as many files as possible, the ransomware terminates running processes that can lock access to valuable files, such as databases, office applications, financial\accounting software, web servers, and virtual machines.

SERVPROC terminates services using the Service Control Manager (SCM) API. This API requires administrator permissions, which suggests the attackers are aware of this requirement and have taken appropriate steps beforehand.

The sample attempts to stop the following services:

sqlservr.exe, *sqlmangr.exe*, *RAgui.exe*, *QBCFMonitorService.exe*, *supervise.exe*, *fdhost.exe*, *Culture.exe*, *RTVscan.exe*, *Defwatch.exe*, *wxServer*
GDscan.exe, *QBW32.exe*, *QBDBMgr.exe*, *qbupdate.exe,axlbridge.exe*, *360se.exe*, *360doctor.exe*, *QBIDPService.exe*, *wxServer.exe*, *httpd.exe*, *fd*
java.exe, *wdswfsafe.exe*.

In addition, the malware developers don't check to see if the API calls are successful.

Figure 8: Disassembly of SCM API calls to stop services.

KILLPR use *CreateToolhelp32Snapshot*, *Process32First* and *Process32Next* APIs to enumerate the running processes in the system.

The sample attempts to terminate the following list of processes:

DefWatch, *ccEvtMgr*, *ccSetMgr*, *SavRoam*, *dbsrv12*, *sqlservr*, *sqlagent*, *Intuit.QuickBooks.FCS*, *dbeng8*, *sqladhlp*, *QBIDPService*, *Culserver*, *RTV*
vmware-usbarbitator64, *vmware-*
converter, *VMAuthdService*, *VMnetDHCP*, *VMUSBArbService*, *VMwareHostd*, *sqlbrowser*, *SQLADHLP*, *sqlwriter*, *msmdsrv*, *tomcat6*, *QBCFMonitorServicechrome.exe*, *outlook.exe*, *chrome.exe*.

The authors of Diavol made some mistakes in their hardcoded configuration. For starters, *SERVPROC* gets the list of the process names to terminate **instead** of service names to stop, and vice versa. The services list appears to include items that have nothing to do with services, like "winword.exe". In the processes list, only the last three items look like actual process names, while even one of them is wrong ("*QBCFMonitorServicechrome.exe*" looks like a concatenation of "*QBCFMonitorService*" and "*chrome.exe*").

· Initialize encryption key:

RSAINIT initializes the RSA public key used for encryption with the standard WinCrypt API.

· Find all drives to encrypt:

ENMDSKS gets all the local drives in the system using *GetLogicalDriveStrings* API and checks to ensure they are not in the exclusions list. This routine will be skipped if the mode command-line parameter ("*-m*") is set to "net".

The default list of the excluded file extensions, file names and paths is:

**.exe*, **.sys*, **.dll*, **.lock64*, **readme_for_decrypt.txt*, **locker.txt*, **unlocker.txt*, *%WINDIR%*, *%PROGRAMFILES%*, *%PROGRAMW6432%*, *%TEM*

*SMBFAST*and *SMB* routines enumerate available network shares for access.

SMBFAST will only be called if the "-h" parameter is present. This routine scans for accessible network shares on specific hosts.

SMB scans for accessible network shares on hosts found in the ARP table. Note that this routine will be skipped if the mode command-line parameter ("*-m*") is set to "local".

· Find files to encrypt:

FINDFILES traverses the file and directories in a given path. This routine is invoked multiple times:

- Depending on the "-p" command-line or hardcoded configuration.
- According to *ENMDSKS* results.
- According to *SMBFAST* results.
- According to *SMB* results.

First, the routine checks to see if the given file or folder is not in the exclusion list. Then the routine determines whether the current item is in the list of files to delete. If not, it queues an APC on the current thread that gets the path to the item as an argument.

Figure 9: Disassembly of the file system traversal.

· Prevent recovery by deleting shadow copies:

VSSMOD drops and executes *wscpy.exe* in the current working directory, depending on the OS version (Windows Server 2003 or Vista and newer) and architecture (32 or 64-bit). Deleting the shadow copies snapshots is performed using the *IVssBackupComponents* COM object to call the *DeleteSnapshots* method.

The dropped binaries are also kept in plaintext in the resource section under "*TEXT*". These binaries contained a PDB reference source file that was used to compile the malware binary:

D:\Projects\Repository\LockCry.divided\WipeShadowCopies64\RelNoCRT\WipeShadowCopies64.pdb

D:\Projects\Repository\LockCry.divided\WipeShadowCopies64\x64\RelNoCRT\WipeShadowCopies64.pdb

D:\Development\Master\onion\locker.divided\WipeShadowCopies64\RelNoCRT_Win2003\WipeShadowStorageWin2003_32.pdb

D:\Development\Master\onion\locker.divided\WipeShadowCopies64\x64\RelNoCRT_Win2003\WipeShadowStorageWin2003_64.pdb

· Encryption:

As mentioned before, *FINDFILES* will queue an APC object with a file or directory path. For the APC to get executed, the main function calls *SleepEx* API in its final steps to set the thread to an alertable state.

The APC routine checks if the argument is directory and will create the "*README_FOR_DECRYPT.txt*" ransom note regardless of whether the files in the directory were encrypted. If the argument is a file, the *ENCDEFILE*routine will be called.

Figure 10: Disassembly of the APC routine checking the parameter type.

Unlike most other ransomwares, Diavol does not use any symmetric encryption but only RSA to encrypt the files.

ENCDEFILE checks the file size. If it is less than 2,000,000 bytes, then only up to the first 11,700 bytes will be encrypted. If the file size is equal or greater than 2,000,000 bytes, then just the first 1,170,000 bytes will be encrypted.

Each block of 11,700 bytes is split into ten chunks of 117 bytes each, and each chunk is encrypted using *CryptEncrypt* API. 117 plaintext bytes then become 128 bytes of ciphertext. Therefore, 11,700 bytes become 12,800 bytes following the encryption. Diavol overwrites the file and writes 11,700 encrypted bytes at the chunk's original offset and appends the 1,100 remaining bytes to the end of the file.

Finally, *ENCDEFILE* calls *MoveFile* API and appends a "*.lock64*" extension to the filename.

Once all APCs are dequeued and completed, the thread returns following the call to *SleepEx*.

· **Change the desktop wallpaper**:

Before the process terminates, *CHNGDESK* is invoked.

First, it captures the desktop window and sets the background color to black. It then writes "*All your files are encrypted! For more information see "README-FOR-DECRYPT.txt*" with *DrawText* API to a bitmap image and saves it as "*encr.bmp*" in the public pictures folder. Finally, it changes the desktop wallpaper to the new image using the *SystemParametersInfo*API with the SPI_SETDESKWALLPAPER flag.

Figure 11: Desktop wallpaper after being changed.

Connections to Conti and Egregor

As Diavol was deployed in conjunction with the Conti ransomware in this attack, albeit on different machines, we tried to see if there's any correlation between them.

Starting with the command-line parameters, the ones used by Diavol are nearly identical to those of Conti and used for the same functionality: log file, encrypt local drives or network shares, and scan specific hosts for network shares. In addition, Diavol and Conti both operate similarly with asynchronous I/O operations when queuing the file paths for encryption.

There also might be a link between Diavol and Egregor ransomware. Some lines in the ransom note are identical, as can be seen in figure 12, although this is not reliable as it could simply be a red herring that Diavol's authors planted.

Some have reported a link between Wizard Spider, the threat actor behind Conti, and Twisted Spider, the threat actor behind Egregor. Allegedly, these gangs cooperate on various operations. They are also both notoriously known for double ransoming their victims (data theft and encryption).

Figure 12: Diavol's ransom note above Egregor's ransom note.

Summary

In this blog, we presented a complete analysis of the new ransomware family - Diavol.

Currently, the source of the intrusion is unknown. The parameters used by the attackers, along with the errors in the hardcoded configuration, hint to the fact that Diavol is a new tool in the arsenal of its operators which they are not yet fully accustomed to.

As the attack progressed, we found more Conti payloads named *locker.exe* in the network, strengthening the possibility the threat actor is indeed Wizard Spider. Despite a few similarities between Diavol, Conti, and other related ransomware, it's still unclear, however, whether there's a direct link between them. And, there are a couple of major differences from attacks previously attributed to Wizard Spider & Co., namely:

- No checks and balances to ensure the payload will not execute on Russian victims.
- No clear evidence of double extortion in the environment was found.

Fortinet Solutions

FortiEDR detects and blocks the Diavol and Conti ransomwares attacks out-of-the-box without any prior knowledge or special configuration. It does this using its post-execution prevention engine to identify malicious activities, such as encrypting files or wiping the shadow copy, and then blocking them in real-time, as can be seen in figures 13 and 14:

Figure 13: FortiEDR blocking Diavol ransomware.

Figure 14: FortiEDR blocking Conti ransomware.

The FortiGuard AntiVirus service is supported by, and the FortiGuard AntiVirus engine is included in Fortinet's FortiGate, FortiMail, FortiClient, and FortiEDR solutions. FortiGuard AntiVirus has coverage in place for the following samples:

85ec7f5ec91adf7c104c7e116511ac5e7945bcf4a8fdecdcc581e97d8525c5ac (Diavol, locker.exe) W32/Malicious_Behavior.VEX

426ba2acf51641fb23c2efe686ad31d6398c3dd25c2c62f6ba0621455a3f7178 (Conti v3, locker64.dll) W64/BazarLoader.AD!tr

4bfd58d4e4a6fe5e91b408bc190a24d352124902085f9c2da948ad7d79b72618 (Conti, locker.exe) W32/Conti.F!tr.ransom

All network IOCs have been added to the FortiGuard WebFiltering blocklist.

In addition, as part of our membership in the Cyber Threat Alliance, details of this threat were shared in real time with other Alliance members to help create better protections for customers.

Appendix A: MITRE ATT&CK Techniques

ID	Description
T1027	Obfuscated Files or Information
T1082	System Information Discovery
T1071.001	Application Layer Protocol: Web Protocols
T1489	Service Stop
T1562.001	Impair Defenses: Disable or Modify Tools
T1135	Network Share Discovery
T1490	Inhibit System Recovery
T1559.001	Inter-Process Communication: Component Object Model
T1486	Data Encrypted for Impact
T1485	Data Destruction

Appendix B: IOCs

File Hashes (SHA256)

85ec7f5ec91adf7c104c7e116511ac5e7945bcf4a8fdecdcc581e97d8525c5ac (Diavol, locker.exe)
426ba2acf51641fb23c2efe686ad31d6398c3dd25c2c62f6ba0621455a3f7178 (Conti v3, locker64.dll)
4bfd58d4e4a6fe5e91b408bc190a24d352124902085f9c2da948ad7d79b72618 (Conti, locker.exe)

File Names

locker.exe
locker64.dll
wscpy.exe
encr.bmp
README_FOR_DECRYPT.txt

File Paths

%PUBLIC%\Pictures\encr.bmp

IPs

173[.]232[.]146[.]118

URLs

hxxp://<server_address>//BnpOnspQwtjCA/register
hxxp://173[.]232[.]146[.]118/Bnyar8RsK04ug/

Domains

r2gttyb5vqu6swf5[.]onion

Learn more about Fortinet's FortiGuard Labs threat research and intelligence organization and the FortiGuard Security Subscriptions and Services portfolio.

Learn more about Fortinet's free cybersecurity training, an initiative of Fortinet's Training Advancement Agenda (TAA), or about the Fortinet Network Security Expert program, Security Academy program, and Veterans program. Learn more about FortiGuard Labs global threat intelligence and research and the FortiGuard Security Subscriptions and Services portfolio.