

# CatB Ransomware: A New Threat Exploiting DLL Side-Loading - VMRay

By VMRay Labs

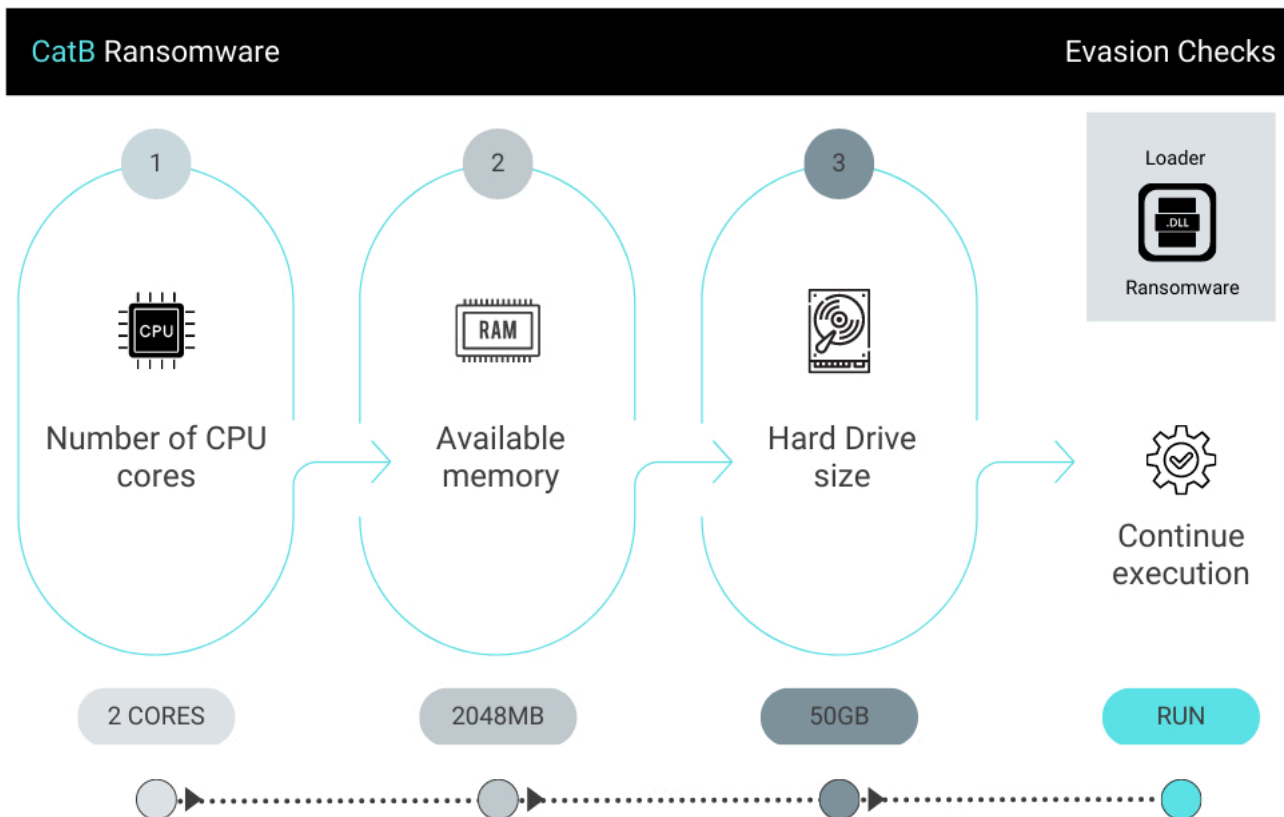
Published: 2023-03-15 · Archived: 2026-04-05 16:51:26 UTC

## Introduction

Just before closing off the year 2022, a new ransomware called [CatB appeared on VirusTotal](#). Compared to other ransomware, this new malware family gets shipped with unique characteristics that make this recent market joiner interesting: Before the ransomware is executed, its loader component performs basic evasion checks to ensure the sample is not running in an analysis environment, such as a sandbox.

Once all checks are passed, the payload containing the ransomware is executed in the context of a trusted Windows service via a [DLL Side-Loading](#) attack. CatB then searches for user-specific files based on a list of extensions that can be encrypted. In contrast to behavior shown by other ransomware families, instead of dropping the ransom note in separate files on various different locations, CatB prepends it to each of the encrypted files. According to the ransom notes content, the files are encrypted with RSA and a key size of 2048.

Even though we haven't observed different samples at the time of writing, within this Spotlight, we are taking a brief look at one ransomware sample to highlight the evasion techniques as well as the DLL Side-Loading attack.



## Analysis of CatB Ransomware Loader

While the full delivery chain of CatB is currently unknown, the infection mechanism itself can be observed within VMRay Platform: A malicious DLL, referred to as the load within this Spotlight, is executed on a system and starts with basic evasion checks to hide its behavior from analysis environments. In total, there are three different techniques implemented.

The first one checks the number of CPU cores that are present on the system by calling the Windows API `GetSystemInfo` to retrieve the respective counter. The loader expects to see at least two cores available to continue its execution (Figure 1). While many sandboxes only feature single-core VMs to lower the required resources for running parallel analyses, modern desktop computers or notebooks typically have multiple CPU cores.

```

mov     [rbp+2D0h+var_10], rax
lea     rcx, [rbp+2D0h+SystemInfo] ; lpSystemInfo
call    cs:GetSystemInfo
cmp     [rbp+2D0h+SystemInfo.dwNumberOfProcessors], 2
jnb     loc_1800013E2
    
```

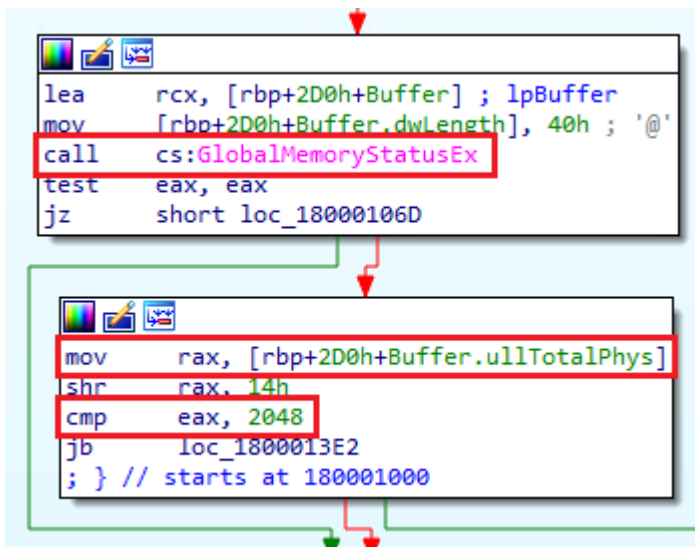
VMRay Platform provides the option to easily adjust the number of CPU cores directly within the system configuration (Figure 2). The modification is transparent to major parts of the VM as it only fakes the number of CPU cores for processes that are relevant to the analysis. This ensures short reaction times on changes observed in threat trends, as well as keep maintaining short analysis times, as increasing the number of faked CPU cores does not have a negative performance impact in the case of parallel running analyses.

## Fake number of CPUs

4

The second performed anti-VM technique checks for the amount of main memory that is available to the system. This is utilized via calling the Windows API GlobalMemoryStatusEx, which returns a struct with various information about physical and virtual memory.

The loader extracts the value of the total available physical memory to verify that the system has at least 2048MB of memory installed (Figure 3). Desktop computers rarely have less, but many sandboxes do to save resources for the execution of parallel analyses.



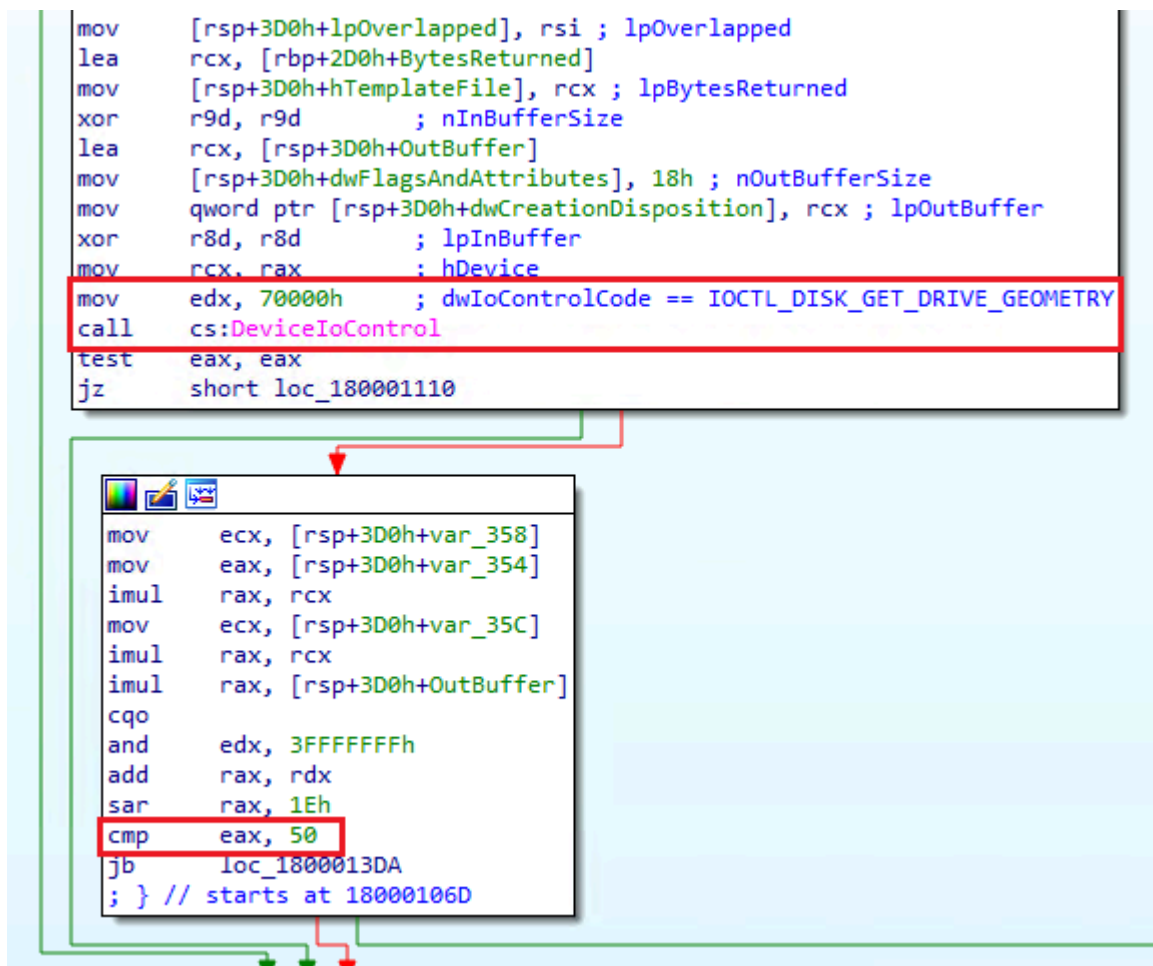
Besides faking CPU cores, VMRay Platform also provides a configuration option to adjust the amount of memory that is seen by relevant processes. Similar to faking the number of CPU cores, increasing the size of available main memory does not have a negative performance impact on analyses as this is also transparent to major parts of the VM.

## Fake RAM size

4096

For the third and final check, the sample extracts the size of the hard drive as most analysis environments provide only very limited space, which is not typical for modern computers. As for all prior checks, there is again a Windows API function called DeviceIoControl available that the loader uses to receive specifications for the drive.

By using IOCTL\_DISK\_GET\_DRIVE\_GEOMETRY as a parameter for the API call, the system returns the geometry data of the physical disk, which can be used to calculate the total size. The loader continues execution only if the drive has a size of at least 50GB (Figure 5). The analysis VMs used by VMRay have disk sizes that are large enough not to get caught by this analysis environment check.



The CatB ransomware sample discussed in this Spotlight uses a DLL Side-Loading vulnerability in a trusted Windows service called MSDTC (Microsoft Distributed Transaction Coordinator), which has been known for more than 2 years. DLL Side-Loading is a well-known and easy-to-find exploitation technique that can be used by attackers to achieve code execution in the context of a trusted process as the vulnerable program tries to load one or multiple DLLs via their names from specified locations.

Once all anti-VM checks have been passed, the loader drops a file called “oci.dll” into the Windows System32 directory, which is later on mistakenly loaded by MSDTC (Figure 6).

```

[0126.370] CreateFileW (lpFileName="C:\windows\system32\oci.dll" (normalized: "c:\windows\system32\oci.dll"), dwDesiredAccess=0x40000000, dwShareMode=0x0, lpSecurityAttributes=0x14f738, dwCreationDisposition=0x2, dwFlagsAndAttributes=0x80, hTemplateFile=0x0) returned 0x150
[0126.374] WriteFile (in: hFile=0x150, lpBuffer=0x7ff8f7e09a40*, nNumberOfBytesToWrite=0x29000, lpNumberOfBytesWritten=0x14f7d4, lpOverlapped=0x0 | out: lpBuffer=0x7ff8f7e09a40*, lpNumberOfBytesWritten=0x14f7d4*0x29000, lpOverlapped=0x0) returned 1
[0126.405] RtlAllocateHeap (HeapHandle=0x5a0000, Flags=0x8, Size=0x1000) returned 0x5beb60
[0126.406] CloseHandle (hObject=0x150) returned 1
    
```

During the next step, the loader modifies the MSDTC service to let it start automatically to achieve persistence, even though this might not be important for a ransomware.

To reach an execution with higher permissions on the system, the loader updates the user which is used to run the service to LocalSystem, which is an administrative user (Figure 7). To finally trigger the execution of CatB ransomware, the loader starts the service, which tries to load the previously dropped DLL due to its DLL Side-Loading vulnerability.

```
[0126.414] OpenSCManagerA (lpMachineName=0x0, lpDatabaseName=0x0, dwDesiredAccess=0xf003f) returned 0x5bd6f0
[0126.424] OpenServiceA (hSCManager=0x5bd6f0, lpServiceName="msdtc", dwDesiredAccess=0x12) returned 0x5bd4e0
[0126.426] ChangeServiceConfigA (in: hService=0x5bd4e0, dwServiceType=0xffffffff, dwStartType=0x2, dwErrorControl=0xffffffff, lpBinaryPathName=0x0, lpLoadOrderGroup=0x0, lpdwTagId=0x0, lpDependencies=0x0, lpServiceStartName="LocalSystem", lpPassword=0x0, lpDisplayName=0x0 | out: lpdwTagId=0x0) returned 1
[0126.446] StartServiceA (hService=0x5bd4e0, dwNumServiceArgs=0x0, lpServiceArgVectors=0x0) returned 1
[0129.030] CloseServiceHandle (hSCObject=0x5bd4e0) returned 1
[0129.045] CloseServiceHandle (hSCObject=0x5bd6f0) returned 1
```

Dropping a DLL into a trusted Windows system-based location as well as Side-Loading it into a process is detected via VTIs (VMRay Threat Identifiers) and helps to detect the maliciousness of the loader used by the ransomware (Figure 8).

▼	2/5	Defense Evasion	Loads a dropped DLL
<ul style="list-style-type: none"> <li>• Dropped DLL oci.dll is side-loaded into (Process #16) msdtc.exe. ...</li> <li>• Dropped DLL oci.dll is side-loaded into (Process #21) msdtc.exe. ...</li> </ul>			
▼	1/5	System Modification	Modifies operating system directory
<ul style="list-style-type: none"> <li>• (Process #1) jhafdvir.exe creates file "C:\windows\system32\oci.dll" in the OS directory. ...</li> </ul>			

## Analysis of CatB Ransomware

Finally, the actual payload containing CatB ransomware is executed in the context of the trusted Windows service MSDTC. An interesting observation here is that the ransomware performs exactly the same anti-VM checks, as done by the loader (Figure 9).

```
[0129.254] GetSystemInfo (in: lpSystemInfo=0xc9f7c0 | out: lpSystemInfo=0xc9f7c0*(dwOemId=0x9, wProcessorArchitecture=0x9, wReserved=0x0, dwPageSize=0x1000, lpMinimumApplicationAddress=0x10000, lpMaximumApplicationAddress=0x7fffffff, dwActiveProcessorMask=0xf, dwNumberOfProcessors=0x4, dwProcessorType=0x21d8, dwAllocationGranularity=0x10000, wProcessorLevel=0x6, wProcessorRevision=0x5504))
[0129.254] GlobalMemoryStatusEx (in: lpBuffer=0xc9f780 | out: lpBuffer=0xc9f780) returned 1
[0129.254] CreateFileW (lpFileName="\\\\.\\PhysicalDrive0" (normalized: "\\device\\harddisk0\\dr0"), dwDesiredAccess=0x0, dwShareMode=0x3, lpSecurityAttributes=0x0, dwCreationDisposition=0x3, dwFlagsAndAttributes=0x0, hTemplateFile=0x0) returned 0x1d8
[0129.254] DeviceIoControl (in: hDevice=0x1d8, dwIoControlCode=0x70000, lpInBuffer=0x0, nInBufferSize=0x0, lpOutBuffer=0xc9fa20, nOutBufferSize=0x18, lpBytesReturned=0xc9f770, lpOverlapped=0x0 | out: lpOutBuffer=0xc9fa20*, lpBytesReturned=0xc9f770*=-0x18, lpOverlapped=0x0) returned 1
```

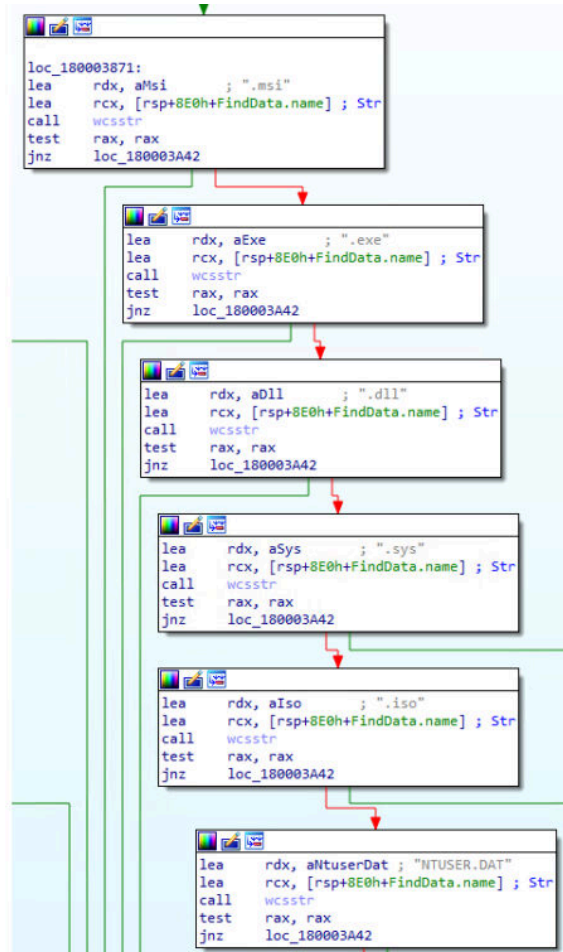
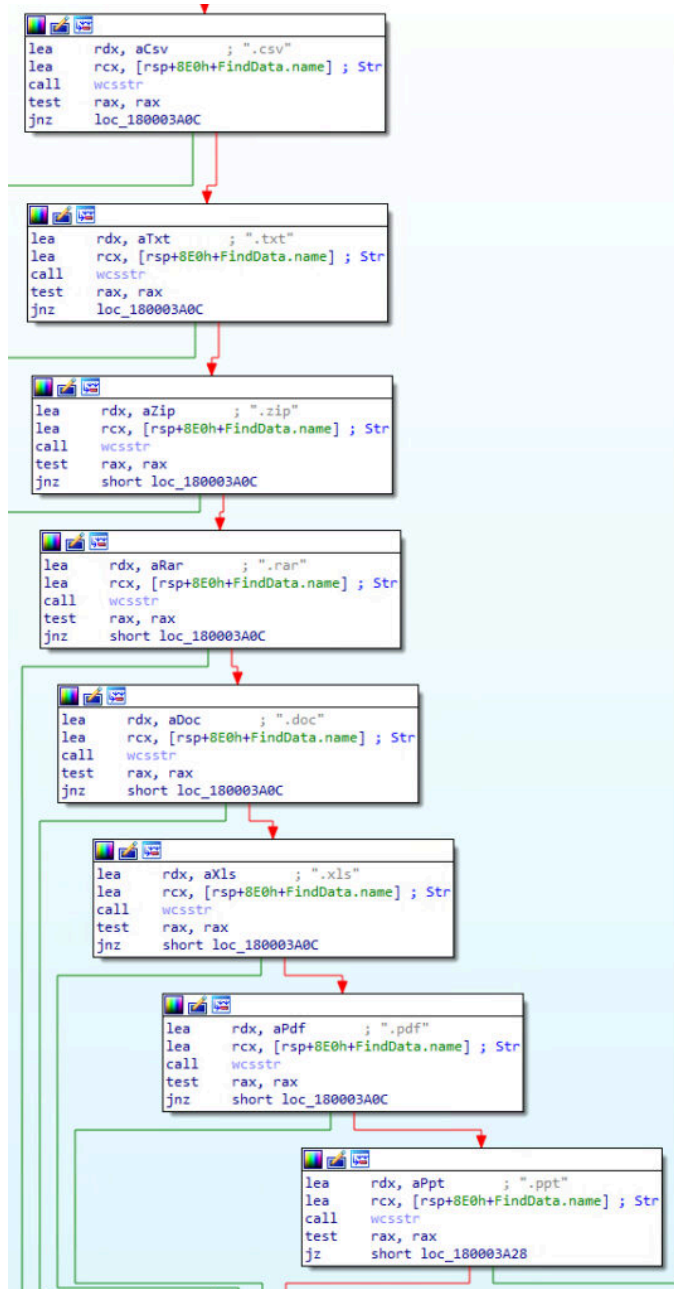
Once CatB passes all the checks for the second time, the ransomware tries to find interesting files to encrypt to make them unavailable to the user. VMRay reveals the behavior of searching for files across multiple drives in its function log (Figure 10).

```
[0129.294] FindFirstFileExW (in: lpFileName="I:\\*.*)" (normalized: "i:\\*.*)" , fInfoLevelId=0x0, lpFindFileData=0xc9ebd0, fSearchOp=0x0, lpSearchFilter=0x0, dwAdditionalFlags=0x0 | out: lpFindFileData=0xc9ebd0) returned 0xfffffffffffff
[0129.294] GetLastError () returned 0x3
[0129.294] GetLastError () returned 0x3
[0129.295] SetLastError (dwErrCode=0x3)
[0129.306] FindFirstFileExW (in: lpFileName="H:\\*.*)" (normalized: "h:\\*.*)" , fInfoLevelId=0x0, lpFindFileData=0xc9ebd0, fSearchOp=0x0, lpSearchFilter=0x0, dwAdditionalFlags=0x0 | out: lpFindFileData=0xc9ebd0) returned 0xfffffffffffff
[0129.395] GetLastError () returned 0x3
[0129.395] GetLastError () returned 0x3
[0129.395] SetLastError (dwErrCode=0x3)
[0129.408] FindFirstFileExW (in: lpFileName="G:\\*.*)" (normalized: "g:\\*.*)" , fInfoLevelId=0x0, lpFindFileData=0xc9ebd0, fSearchOp=0x0, lpSearchFilter=0x0, dwAdditionalFlags=0x0 | out: lpFindFileData=0xc9ebd0) returned 0xfffffffffffff
[0129.409] GetLastError () returned 0x3
[0129.409] GetLastError () returned 0x3
[0129.409] SetLastError (dwErrCode=0x3)
[0129.421] FindFirstFileExW (in: lpFileName="F:\\*.*)" (normalized: "f:\\*.*)" , fInfoLevelId=0x0, lpFindFileData=0xc9ebd0, fSearchOp=0x0, lpSearchFilter=0x0, dwAdditionalFlags=0x0 | out: lpFindFileData=0xc9ebd0) returned 0xfffffffffffff
[0129.421] GetLastError () returned 0x3
[0129.421] GetLastError () returned 0x3
[0129.421] SetLastError (dwErrCode=0x3)
[0129.430] FindFirstFileExW (in: lpFileName="E:\\*.*)" (normalized: "e:\\*.*)" , fInfoLevelId=0x0, lpFindFileData=0xc9ebd0, fSearchOp=0x0, lpSearchFilter=0x0, dwAdditionalFlags=0x0 | out: lpFindFileData=0xc9ebd0) returned 0xfffffffffffff
[0129.651] GetLastError () returned 0x3
[0129.651] GetLastError () returned 0x3
[0129.651] SetLastError (dwErrCode=0x3)
[0129.658] FindFirstFileExW (in: lpFileName="D:\\*.*)" (normalized: "d:\\*.*)" , fInfoLevelId=0x0, lpFindFileData=0xc9ebd0, fSearchOp=0x0, lpSearchFilter=0x0, dwAdditionalFlags=0x0 | out: lpFindFileData=0xc9ebd0) returned 0xfffffffffffff
[0129.659] GetLastError () returned 0x3
[0129.659] GetLastError () returned 0x3
[0129.659] SetLastError (dwErrCode=0x3)
[0129.659] FindFirstFileExW (in: lpFileName="C:\\Users\\*.*)" (normalized: "c:\\users\\*.*)" , fInfoLevelId=0x0, lpFindFileData=0xc9ebd0, fSearchOp=0x0, lpSearchFilter=0x0, dwAdditionalFlags=0x0 | out: lpFindFileData=0xc9ebd0) returned 0x440350
```

Our analysis shows that the ransomware searches for files that are related to users and maintains two lists for this purpose. One list with the extensions of file types that will be encrypted (Figure 11, left), and another, containing extensions that are excluded and, therefore, will not be modified (Figure 11, right).

Based on the exclusion list, we can assume that the malware developer does not want to damage the system or does not see any advantage in encrypting system files, installers, or disk images. Ransomware operators typically do not want to destroy, they want to hold files hostage, and the operating system still needs to work for the victim

to see the ransom note.

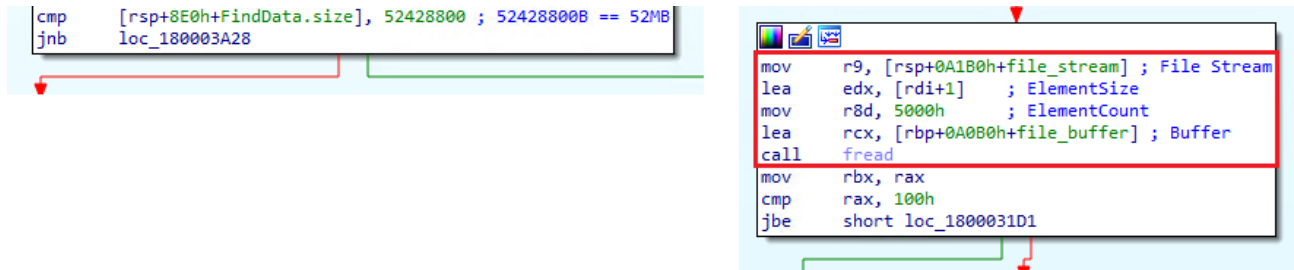


However, even though the ransomware maintains a list of file extensions used to search for files that will be encrypted, it will also encrypt files with extensions that are not part of that list.

While searching for files, CatB checks each file's size; if it is below 52MB and part of the extensions listed in Figure 11 (left) the complete file is encrypted. Otherwise, if the size of the file exceeds the limit of 52MB or does not have a proper extension and is not part of the list in Figure 11 (right), only the first 20480 bytes are encrypted (Figure 12).

We can assume that the ransomware developer limits the file size due to the amount of performance that is needed for encrypting large files. If the encrypted process takes too long, systems can be switched off before all of their

files have been encrypted.



Compared to other ransomware families, CatB prepends the ransom note to each encrypted file without modifying its extension, instead of dropping a separate file and renaming the encrypted one. Victims might not notice what happened to their files as common applications associated with the file type can fail to open them if the header, bytes used to recognize them, are gone. Looking at the content of an encrypted file reveals the ransom note (Figure 13).

```

1
2 ??? What happend???
3 !!! Your files are encrypted !!!
4
5 *All your files are protected by strong encryption with RSA-2048.*
6 *There is no public decryption software.*
7
8
9 ##### Program and private key, What is the price? The price depends on how fast you can pay to us.#####
10 1 day : 50 Bitcoin
11 2 day : 60 Bitcoin
12 3 day : 90 Bitcoin
13 4 day : 130 Bitcoin
14 5 day : permanent data loss !!!!
15
16 Btc Address: bc1qakuel0s4nyge9rxjylsqdxmn9vvyhc2z6k27gz
17 !!! After received, we will send program and private key to your IT department right now.!!!
18
19 *Free decryption As a guarantee, you can send us up to 3 free decrypted files before payment.*
20 email: catB9991@protonmail.com
21
22 !!! Do not attempt to decrypt your data using third-party software, this may result in permanent data loss.!!!
23 !!! Our program can repair your computer in few minutes.!!!
24
25 20496[REDACTED]

```

VMRay’s VTIs are capable of detecting the modification of multiple user files within the analysis environment, even though the malicious behavior was performed in the context of a trusted system service (Figure 14).

In addition, the VMRay Platform comes with a continuously updated set of YARA rules.

	5/5	User Data Modification	Modifies content of user files
		(Process #16) msdct.exe modifies the content of multiple user files. ***	
	5/5	YARA	Malicious content matched by YARA rules
		Rule "CatB" from ruleset "Ransomware" has matched on a memory dump for (process #1) jhafdvir.exe. ***	
		Rule "CatB_Loader" from ruleset "Ransomware" has matched on a memory dump for (process #1) jhafdvir.exe. ***	
		Rule "CatB_FunctionStrings" from ruleset "Ransomware" has matched on the function strings for (process #16) msdct.exe. ***	
		Rule "CatB" from ruleset "Ransomware" has matched on the dropped file "C:\windows\system32\oci.dll". ***	
		Rule "CatB_FunctionStrings" from ruleset "Ransomware" has matched on the function strings for (process #21) msdct.exe. ***	

## Conclusion

Malware families like CatB underline the importance of using an evasion-resistant sandbox. This ensures that samples with anti-VM checks are analyzed properly and reveal their malicious behavior. VMRay provides extensive and easy-to-configure options to bypass a variety of different anti-VM checks.

VMRay's analysis report, as well as the function logging capability, assists malware analysts and researchers in quickly diving deep into the ransomware's behavior and having a detailed look into its capabilities.

## References

<https://minerva-labs.com/blog/new-catb-ransomware-employs-2-year-old-dll-hijacking-technique-to-evade-detection>

<https://www.fortinet.com/blog/threat-research/ransomware-roundup-catb-ransomware>

## IOCs

### Hashes:

Sample (Loader):

3661ff2a050ad47fdc451aed18b88444646bb3eb6387b07f4e47d0306aac6642

### Hashes

Dropped Payload (Ransomware)

35a273df61f4506cdb286ecc40415efaa5797379b16d44c240e3ca44714f945b

### Ransom Note:

Bitcoin Wallet Address

bc1qakuel0s4nyge9rxjylsqdxnn9nvyhc2z6k27gz

### Ransom Note:

Email Contact

catB9991@protonmail.com

## Patrick Staubmann

Threat Researcher