A Detailed Analysis of The Last Version of REvil Ransomware

Prepared by: Vlad Pasca, Senior Malware and Threat Analyst
Table of contents

Executive summary 2
Analysis and findings 2
Thread activity – sub_1282EA7 function 37
Thread activity – sub_1287677 function 37
Thread activity – sub_1284468 function 41
Thread activity – sub_12841D3 function 44
Running with the -smode parameter 48
Running with the -silent parameter 51
Running with the -path parameter 51
Running with the -nolan parameter 51
Running with the -nolocal parameter 51
Running with the -fast parameter 51
Running with the -full parameter 51
Indicators of Compromise 51
Appendix 52
Executive summary

REvil/Sodinokibi ransomware has been active since 2019, with breaks due to law enforcement. The ransomware can run with one of the following parameters: "-nolan", "-nolocal", "-path", "-silent", "-smode", "-fast", and "-full". The malware comes with an RC4 encrypted configuration, kills a list of targeted processes, and stops some specified services. It also deletes all Volume Shadow Copies using WMI and targets logical drives and network shares.

The sample only renames the files that are supposed to be encrypted due to a bug implemented by the developers. REvil implements a combination of ECC (Curve25519) and Salsa20 algorithms during the encryption process. The ransomware can operate in Safe Mode by specifying a parameter, and it establishes persistence in this case by creating an entry under the RunOnce key.

Analysis and findings

SHA256: 0c10cf1b1640c9c845080f460ee69392bfaac981a4407b607e8e30d2ddf903e8

The ransomware resolves the relevant APIs at runtime by implementing an API hashing function:
REvil decrypts relevant strings at runtime using the RC4 algorithm. The call shown in figure 2 contains a pointer to a buffer that contains the RC4 keys, the encrypted strings, and the offset to the RC4 key:

![Figure 2](image)

The implementation of the RC4 algorithm and a decrypted string are shown below:

![Figure 3](image)

![Figure 4](image)
LoadLibraryA is used to load multiple DLLs into the address space of the process:

Figure 5

GetProcAddress is utilized to obtain the address of multiple exported functions:

Figure 6

The malware forces the system not to display the critical-error-handler message box via a function call to SetErrorMode (0x1 = SEM_FAILCRITICALERRORS):

Figure 7

The REvil configuration is decrypted using the RC4 algorithm:

Figure 8
The table below describes the meaning of each parameter:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk</td>
<td>The ECC public key encoded using Base64</td>
</tr>
<tr>
<td>pid</td>
<td>Bcrypt hash representing the affiliate ID</td>
</tr>
<tr>
<td>sub</td>
<td>Bcrypt hash representing the campaign identifier</td>
</tr>
<tr>
<td>dbg</td>
<td>Boolean value that indicates whether REvil runs in debug mode</td>
</tr>
</tbody>
</table>
| wht      | Three lists of elements that will be skipped:  
|          | ● fld – whitelisted directories  
|          | ● fls – whitelisted files  
|          | ● ext – whitelisted extensions |
| prc      | A list of processes that will be stopped |
| accs     | A list of credentials that will be used to connect to network shares |
| svc      | A list of services that will be stopped and deleted |
| net      | Boolean value that indicates whether REvil communicates with the C2 servers |
| nbody    | The ransom note encoded using Base64 |
| nname    | The ransom note name |
| exp      | Boolean value that indicates whether REvil should perform privilege escalation |
| img      | The text that will be written in the background image, which is encoded using Base64 |
| et       | A value that specified the encryption type:  
|          | ● 0 – fast encryption  
|          | ● 1 – full encryption  
<p>|          | ● 2 – encrypt 1MB of a file and then skip several MBs mentioned in the spsize |</p>
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spsize</td>
<td>A value that specifies the number of MBs that will be skipped when the encryption type is 2</td>
</tr>
<tr>
<td>arn</td>
<td>Boolean value that indicates whether REvil establishes persistence on the system</td>
</tr>
<tr>
<td>rdmcnt</td>
<td>Readme count (set to 0)</td>
</tr>
</tbody>
</table>

The binary implements a JSON parser that will parse the configuration:

![Diagram 1](image1.png)

**Figure 9**

The MultiByteToWideChar API is used to convert the encoded ECC public key to a UTF-16 string:

![Diagram 2](image2.png)

**Figure 10**
CryptStringToBinaryW is utilized to decode the public key (0x1 = CRYPT_STRING_BASE64):

The nbody parameter that contains the ransom note content is also decoded using the same approach (0x1 = CRYPT_STRING_BASE64):
Lastly, the ransomware decrypts the img field (0x1 = CRYPT_STRING_BASE64):

The malicious executable opens the "SOFTWARE\LFF9miD" registry key via a call to RegOpenKeyExW (0x80000002 = HKEY_LOCAL_MACHINE, 0x1 = KEY_QUERY_VALUE):

REvil decrypts a .onion address and a string using the RC4 algorithm:
The rdtsc instruction is used multiple times to retrieve the processor time stamp:

![Figure 19](image1)

The malware uses a custom function to generate 0xC0 bytes:

![Figure 20](image2)

![Figure 21](image3)
The resulting buffer of the above operation is highlighted in figure 22:

![Figure 22](image22)

The above buffer is modified two times, and the binary will use 32 bytes from the result:

![Figure 23](image23)
The process generates a 32-byte Curve25519 secret key based on the above result. Using the private key, the binary generates the corresponding 32-byte Curve25519 public key, where the basepoint is 09 followed by all zeros:

The capa tool confirms that the algorithm used to generate the above key is indeed Curve25519:
Using the same approach, REvil generates a second Curve25519 public key based on a different Curve25519 private key:

The ransomware computes a shared secret based on the secret key used to generate the above public key and the attacker’s public key from the configuration:
Using the same method as before, the malicious process generates a buffer and then computes 32 bytes:

![Figure 32](image)

The executable creates the "SOFTWARE\LFF9miD" registry key by calling the RegCreateKeyExW routine (0x80000002 = HKEY_LOCAL_MACHINE, 0x2 = KEY_SET_VALUE):

![Figure 33](image)

REvil stores the attacker’s public key that was decoded in a registry value called “miz” (0x3 = REG_BINARY):

![Figure 34](image)
REvil stores the first generated Curve25519 public key in a registry value called "od4U" (0x3 = REG_BINARY):

The second Curve25519 private key is AES encrypted using the shared key. The IV was randomly generated, and the ransomware computes the CRC32 of the encrypted buffer. The result is stored in a registry value called "U7ykk":

The malicious binary encodes the above buffer using Base64 (0x40000001 = CRYPT_STRING_NOCRLF | CRYPT_STRING_BASE64):
The `CryptBinaryToStringW` function is utilized to encode the attacker's public key back to Base64 format (0x40000001 = `CRYPT_STRING_NOCRLF | CRYPT_STRING_BASE64`):

The binary retrieves the processor name using the `cpuid` instruction:
The GetWindowsDirectoryW routine is used to obtain the path of the Windows directory:

REvil extracts the volume serial number for the C drive by calling the GetVolumeInformationW API:

The ransomware computes the CRC32 hash of the processor name. The implementation is displayed below:
The malware constructs a UID by combining the CRC32 hash and the volume serial number:

The executable opens the "SOFTWARE\LFF9miD" registry key via a function call to RegOpenKeyExW (0x80000002 = HKEY_LOCAL_MACHINE, 0x1 = KEY_QUERY_VALUE):
RegQueryValueExW is utilized to retrieve the type and data for a non-existent registry value called "IhnG91T":

REvil generates a random extension consisting of 5-10 alphanumeric characters. There is a comparison between the “decrypt_everything” string and “msu” (one of the whitelisted extensions):

The binary creates a registry value called "IhnG91T", which contains the generated extension that will be appended to the encrypted files (0x1 = REG_SZ):
The malicious file retrieves the name of the current user by calling the GetUserNameW routine:

The GetComputerNameW API is used to extract the NetBIOS name of the local computer:

The process opens the "SYSTEM\CurrentControlSet\services\Tcpip\Parameters" registry key via a call to RegOpenKeyExW (0x80000002 = HKEY_LOCAL_MACHINE, 0x1 = KEY_QUERY_VALUE):

The ransomware verifies whether the computer is part of a Windows domain by checking the "Domain" registry value. This value is supposed to be the DNS domain name:
RegOpenKeyExW is utilized to open the "Control Panel\International" registry key (0x80000001 = HKEY_CURRENT_USER, 0x1 = KEY_QUERY_VALUE):

The current user's language is extracted using the RegQueryValueExW function:

The RegOpenKeyExW routine is used to open the "SOFTWARE\Microsoft\Windows NT\CurrentVersion" key (0x80000002 = HKEY_LOCAL_MACHINE, 0x1 = KEY_QUERY_VALUE):
The Windows product name is retrieved from the Windows registry:

The malware retrieves a list of sessions on the local computer using the WTSEnumerateSessionsW API:

The executable obtains the primary access token of the logged-on user specified by a session extracted above:
REvil scans for available drives and determines the drive type using `GetDriveTypeW`. It expects the return value to be 0x2 (DRIVE_REMOVABLE) and 0x3 (DRIVE_FIXED):

![Figure 61](image1)

The ransomware performs a call to `GetDiskFreeSpaceExW` in order to retrieve information about the amount of space/free space on a disk:

![Figure 62](image2)

The drive name, its type, and the data extracted above are encoded using Base64 (0x40000001 = CRYPT_STRING_NOCRLF | CRYPT_STRING_BASE64):

![Figure 63](image3)

The `GetNativeSystemInfo` API is utilized to obtain information about the current system:

![Figure 64](image4)

The binary is looking for a registry value called "cN86rtdI" under "SOFTWARE\LFF9miD", which doesn't exist at this time:
Revil writes all the information collected so far in a JSON form:

The executable generates a 32-byte Curve25519 private key that is used to compute the corresponding public key:

The above buffer that contains information about the system is encrypted using the XOR operator (the key changes regularly):
The ransomware computes the CRC32 hash of the encrypted buffer and appends the value to it. The encrypted data is written to a registry value called "cN86rtdl" via a function call to RegSetValueExW (0x3 = REG_BINARY):

All registry values that were created by REvil are shown in figure below:

The encrypted buffer from the "cN86rtdl" registry value is encoded in Base64 format using CryptBinaryToStringW (0x1 = CRYPT_STRING_BASE64):
The following parameters are decrypted using RC4: "-nolan", "-nolocal", "-path", "-silent", "-smode", "-fast", and "-full".

The malicious executable extracts the command-line string for the process:

```
Figure 72
```

The CommandLineToArgvW routine is used to obtain an array of pointers to the command line arguments:

```
Figure 73
```

The malware verifies whether the current user is SYSTEM by calling the SHTestTokenMembership function (0x12 = `SECURITY_LOCAL_SYSTEM_RID`):

```
Figure 74
```

The binary creates a mutex called "Global\8D87239A-846D-CD1A-F9C2-8B6763B3B04F" in order to ensure that only one copy of the ransomware is running at a single time:
The SHEmptyRecycleBinW API is utilized to empty the Recycle Bin on all drives (0x7 = SHERB_NOCONFIRMATION | SHERB_NOPROGRESSUI | SHERB_NOSOUND):

The file retrieves a pseudo handle for the current process:

The priority class for the process is changed to 0x8000 (ABOVE_NORMAL_PRIORITY_CLASS) using the SetPriorityClass API:

REvil prevents the system from entering sleep or turning off the display while it's running (0x80000001 = ES_CONTINUOUS | ES_SYSTEM_REQUIRED):
The process enables the SeDebugPrivilege privilege in the access token using RtlAdjustPrivilege (0x14 = SeDebugPrivilege):

![Figure 80](image)

A new thread that will execute the sub_12841D3 function is created by the malware:

![Figure 81](image)

The OpenSCManagerW routine is used to establish a connection to the service control manager on the local machine. The database name was previously decrypted using RC4 (0x4 = SC_MANAGER_ENUMERATE_SERVICE):

![Figure 82](image)

The binary extracts all active services via a call to EnumServicesStatusExW (0x30 = SERVICE_WIN32, 0x1 = SERVICE_ACTIVE):
There is a comparison between a service name and the targeted list of services ("svc" field):

The executable opens a targeted service using OpenServiceW (0x10020 = DELETE | SERVICE_STOP):

The service is stopped via a function call to ControlService (0x1 = SERVICE_CONTROL_STOP):

Finally, the targeted service is deleted by the ransomware:
The `CreateToolhelp32Snapshot` function is used to take a snapshot of all processes in the system (0x2 = `TH32CS_SNAPPROCESS`):

The malware retrieves information about the first process from the snapshot using the `Process32FirstW` routine:

The enumeration continues by calling the `Process32NextW` API:

A targeted process ("prc" field) is opened using `OpenProcess` (0x1 = `PROCESS_TERMINATE`):

REvil kills a targeted process via a function call to `TerminateProcess`:
A new thread that will execute the sub_1284468 function is created by the ransomware:

The binary enables the SeTakeOwnershipPrivilege privilege in the access token using RtlAdjustPrivilege (0x9 = SeTakeOwnershipPrivilege):

The executable creates an input/output (I/O) completion port that is not yet associated with a file handle (0xFFFFFFFF = INVALID_HANDLE_VALUE):

GetSystemInfo is used to retrieve information about the current system:
The **CreateThread** API is utilized to create 2 (the number of processors) that will handle files encryption:

The new threads’ priority is set to 0x2 (**THREAD_PRIORITY_HIGHEST**) using `SetThreadPriority`:

**REvil** enumerates all drives and extracts their type using `GetDriveTypeW`:

The malicious process allocates and initializes a security identifier (SID):
The `SetEntriesInAclW` API is used to create a new access control list (ACL):

The DACL of a drive is modified using the `SetNamedSecurityInfoW` API (0x1 = `SE_FILE_OBJECT`, 0x4 = `DACL_SECURITY_INFORMATION`):

A ransom note called "<REvil extension>-readme.txt" is created in every targeted directory via a call to `CreateFileW` (0x40000000 = `GENERIC_WRITE`, 0x2 = `CREATE_ALWAYS`):
The ransom note is populated using `WriteFile`:

The ransomware enumerates the files using the `FindFirstFileW` and `FindNextFileW` APIs:

All directories identified on a drive are compared with the following strings: "program files", "program files (x86)", and "decrypt_everything".

A file extension is extracted by calling the `PathFindExtensionW` routine.
The malware opens a file using `CreateFileW` (0x80000000 = `GENERIC_READ`, 0x1 = `FILE_SHARE_READ`, 0x3 = `OPEN_EXISTING`):

The malicious binary moves the file pointer to the last 0xE8 bytes in the file (0x0 = `FILE_BEGIN`):

REvil reads the last 0xE8 bytes from a file, which we believe that are common to all encrypted files (see figure 110). It computes the CRC32 hash of the extracted buffer and compares it with a 4-byte value.

The process obtains file system attributes for a file or directory by calling the `GetFileAttributesW` routine:
New attributes are set for a file using `SetFileAttributesW` (0x80 = `FILE_ATTRIBUTE_NORMAL`):

There is a bug implemented by the malware’s author that prevents the files encryption. Firstly, the ransomware adds the ransomware extension to files using `MoveFileW`:

Whether the call is successful, the return value is supposed to be nonzero. The bug that was introduced doesn’t allow the execution to break out of the loop, and there is a second try to rename a file. However, because the file was successfully renamed, the process raises the `ERROR_FILE_NOT_FOUND` error.

The binary posts an empty I/O completion packet to the IOCP created earlier:

It’s interesting that even if the ransomware fails at reading the file content and sending it to the encryption threads, it creates 2 (which is the number of processors) more such threads (`sub_1282EA7` function).

A new thread that will run a different function is created by the malware:
The process waits until the above thread finishes via a function call to WaitForSingleObject:

The GetQueuedCompletionStatus routine is utilized to dequeue an I/O completion packet from the IOCP:

Thanks to the bug implemented by REvil in this binary, the completion packet is always empty, and then the encryption couldn't occur.

The ransomware starts enumerating all currently connected network resources (0x1 = RESOURCE_CONNECTED):
The enumeration continues by calling the WNetEnumResourceW API:

For every network share that can be accessed, the malware creates a ransom note inside every folder (0x40000000 = GENERIC_WRITE, 0x2 = CREATE_ALWAYS):

The WriteFile function is utilized to populate the ransom note:

The files are enumerated using the FindFirstFileW and FindNextFileW APIs:
WNetCancelConnection2W is used to cancel the existing network connection to a network share:

The binary uses the credentials from the "accs" field to connect to a network resource (see figure 125). We believe these credentials were specific to the impacted company.

We continue with the analysis of the main thread.

The process obtains the path of the executable via a function call to GetModuleFileNameW:
REvil deletes itself only after a reboot (0x4 = MOVEFILE_DELAY_UNTIL_REBOOT):

The ransom note is displayed in figure below.

The algorithm that would be used to encrypt files is Salsa20:
Thread activity – sub_1284468 function

The executable initializes the COM library for use by the thread:

The CoCreateInstance API is used to create an IWbemContext Interface with the {674B6698-EE92-11D0-AD71-00C04FD8FDFF} CLSID:
The ransomware retrieves information about the current system by calling the GetNativeSystemInfo routine:

![Figure 132](image1)

The binary uses the IWbemContext::SetValue method in order to create/overwrite a context value:

![Figure 133](image2)

The process creates a WbemLocator object with the {4590f811-1d3a-11d0-891f-00aa004b2e24} CLSID:

![Figure 134](image3)

The malicious file connects to the local "ROOT\CIMV2" namespace and retrieves a pointer to an IWbemServices object:

![Figure 135](image4)
The CoSetProxyBlanket function is utilized to set the authentication information that will be used to make calls on a proxy (0xA = \texttt{RPC\_C\_AUTHN\_WINNT}, 0x3 = \texttt{RPC\_C\_AUTHN\_LEVEL\_CALL}, 0x3 = \texttt{RPC\_C\_IMP\_LEVEL\_ impersonate}): 

\[ \text{Figure 136} \]

The malware executes the following query in order to pull a list of shadow copies stored on the local machine:

\[ \text{Figure 137} \]

The ID of the shadow copy is extracted using the Get method:

\[ \text{Figure 138} \]

Each shadow copy is deleted via a function call to IWbemServices::DeleteInstance:
Thread activity – sub_12841D3 function

CoInitializeSecurity is utilized to set the default security values for the process (0x3 = RPC_C_IMP_LEVEL_IMPERSONATE):

The binary creates a WbemLocator object with the {4590f811-1d3a-11d0-891f-00aa004b2e24} CLSID:

The process connects to the local “ROOT\CIMV2” namespace and retrieves a pointer to an IWbemServices object:
CoSetProxyBlanket is utilized to set the authentication information that will be used to make calls on a proxy (0xA = `RPC_C_AUTHN_WINNT`, 0x3 = `RPC_C_AUTHN_LEVEL_CALL`, 0x3 = `RPC_C_IMP_LEVEL_IMPERSONATE`):

REvil creates an `IUnsecuredApartment` Interface with the `{49bd2028-1523-11d1-ad79-00c04fd8fdff}` CLSID:

The executable runs a query to extract the new process events:
Another query is used to retrieve the service modification events:

The ransomware obtains a pseudo handle for the process using GetCurrentProcess:

The current thread waits until the process is in the signaled state:
There is a call to IWbemClassObject::Get that extracts the TargetInstance property:

![Figure 149](image1)

The ransomware retrieves the user name and domain name under which a process is running using the GetOwner function:

![Figure 150](image2)

Using a similar function call as presented in figure 149, the malware extracts the username, the user domain, and the process name.

REvil kills all running processes specified in the "prc" field from the configuration using the Terminate function:

![Figure 151](image3)
The services specified by the “svc” field that can be found on the system are stopped using StopService:

Running with the -smode parameter

The current user’s password is changed to “k$U0MFks1V” by the malware:

The ransomware enables the Automatic Log-on by modifying the “AutoAdminLogon”, “DefaultUserName”, and “DefaultPassword” values under the “SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon” registry key:
The GetModuleFileNameW API is used to obtain a path of the executable file of the process:

The binary opens the RunOnce registry key (0x80000002 = HKEY_LOCAL_MACHINE, 0x2 = KEY_SET_VALUE):
It establishes persistence by creating a registry value called "aG951f": Figure 159

The malware configures Windows to boot in Safe Mode (0x5 = SW_SHOW): Figure 160

A new registry value called "uTHnGD" is created under the RunOnce key. The process purpose is to disable the boot in Safe Mode: Figure 161

The process enables the SeShutdownPrivilege privilege in the access token using RtlAdjustPrivilege (0x13 = SE_SHUTDOWN_PRIVILEGE): Figure 162
The NtShutdownSystem function is utilized to shut down the system:

![Call stack for NtShutdownSystem](image)

**Running with the -silent parameter**

In this case, the ransomware doesn’t create the threads that run sub_12841D3 (responsible for stopping the processes/services) and sub_1284468 (Volume Shadow Copies deletion).

**Running with the -path parameter**

The malware only encrypts the directory passed as a parameter.

**Running with the -nolan parameter**

The network shares are skipped by the ransomware because the sub_1287677 function is not executed.

**Running with the -nolocal parameter**

Whether it’s running with this parameter, the binary doesn’t encrypt the local drives.

**Running with the -fast parameter**

The process only encrypts the first MB of the file.

**Running with the -full parameter**

In this case, the whole file is encrypted.
Indicators of Compromise

Mutex
Global\8D87239A-846D-CD1A-F9C2-8B6763B3B04F

REvil Ransom Note
{EXT}-readme.txt

Processes spawned
bcdedit /set {current} safeboot network

Registry Keys
Key: HKLM\SOFTWARE\LFF9miD
Value: miz
Value: od4U
Value: U7ykk
Value: IhnG91T
Value: cN86rtdI
Key: HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
Value: AutoAdminLogon
Value: DefaultUserName
Value: DefaultPassword
Key: HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce
Value: *aG951f
Value: *uTHnGD
Appendix

List of processes to be killed

outlook thebat oracle sqbcoreservice mydesktopservice wordpad encsvc infopath sql visio powerpnt mspub thunderbird agntsvc xfsvccon synctime winword dbsnmp ocautoupds onenote msaccess tbirdconfig mydesktopqos ocomm isqlplussvc firefox ocssd steam excel dbeng50

List of services to be stopped

sophos sql mepocs memtas svc$ backup veeam vss

Whitelisted directories

mozilla perflogs msocache $recycle.bin “system volume information” “tor browser” windows programdata appdata boot “application data” $windows.$~bt “program files” windows.old “program files (x86)” google intel $windows.$~ws

Whitelisted files

autorun.inf ntuser.dat.log ntuser.ini boot.ini iconcache.db bootfont.bin ntuser.dat thumbs.db bootsect.bak ntldr desktop.ini

Whitelisted extensions

ics cur icl lnk hta idx diagpkg exe sys msi mpa shs nomedia ani diagcab ps1 scr cpl bin msstyles ocx msu nls themepack 386 wpx icns lock diagcfg cmd mod bat prf msc key cab rtp com hlp ldf rom spl deskthemepackdll msp drv theme adv ico