When Threat Actors Fly Under the Radar: Vatet, PyXie and Defray777

unit42.paloaltonetworks.com/vatet-pyxie-defray777/3

Ryan Tracey, Drew Schmitt November 7, 2020

By Ryan Tracey and Drew Schmitt

November 6, 2020 at 6:15 PM

Category: Malware, Ransomware, Unit 42

Tags: Defray777, PyXie, Vatet

This post is also available in: 日本語 (Japanese)

Last, but Not Least: Defray777

Defray777 is an elusive family of ransomware also known as Ransom X and RansomExx. Although it has recently been covered in the news as a new family, it has been in use since at least 2018 and is responsible for a number of high-profile ransomware incidents -- as detailed in the articles we linked to.

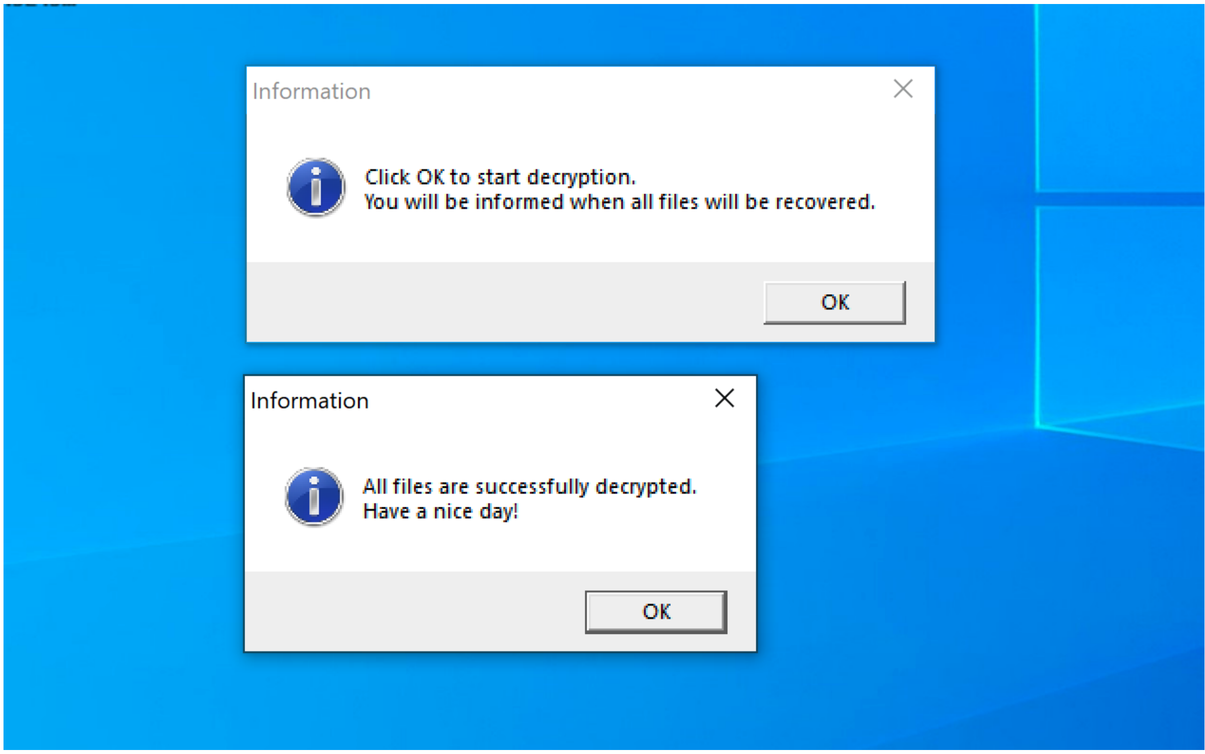
Defray777 runs entirely in memory, which is why there have been so few publicly discussed samples to date. In several recent incidents, Defray777 was loaded into memory and executed by Cobalt Strike, which was delivered by the Vatet loader.

During our research, we discovered multiple decryptors for this ransomware family, going back as early as 2018. Reviewing decryptors from 2018 until present shows that there has been consistency in the ransomware's encryption and decryption methodology, as well as the use of Themida for packing their decryptors. Table 10 shows a list of Defray777 decryptors discovered in AutoFocus, with a list of organizations that suffered ransomware attacks. This shows that Defray777 has been consistently active since 2018.

Date	Victim
12/7/2018	Education Organization
2/4/2019	Healthcare Organization
3/1/2019	Technology Organization
3/15/2019	Education Organization
8/8/2019	Healthcare Organization
8/25/2019	Education Organization
8/28/2019	Transportation and Logistics Organization
9/3/2019	Legal Organization

9/6/2019	Education Organization
9/26/2019	Healthcare Organization
10/30/2019	Government Organization
11/1/2019	Healthcare Organization
2/4/2020	Technology Organization
2/10/2020	Government Organization
3/16/2020	Food Organization
10/17/2020	Finance Organization

Table 10. Defray777 ransomware attacks listed by date and victim.



Click OK to start decryption.
You will be informed when all files will be recovered.

All files are successfully decrypted.
Have a nice day!

Figure 21. Defray777 decryptor.

We have examined several recent Defray777 samples, including one sample that was obtained directly from memory during a recent incident. Our in-depth analysis resulted in the findings outlined below.

Decrypted Strings

The string decryption process is the same as we saw with PyXie. The following strings were decrypted from a recent Defray777 sample:

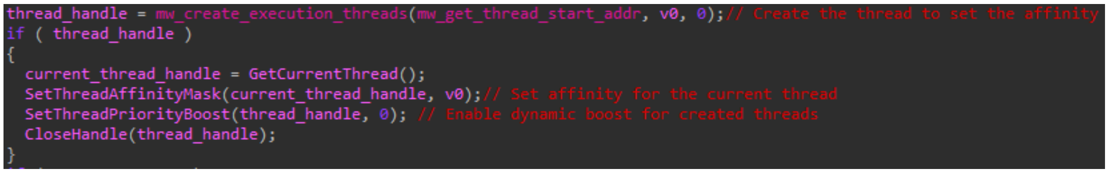
Already active [%s] +%u (%u) files done [%s] [%u KB/s]
Started (PID: %u; Workers: %u; AES-%s) [%s]
Complete (+%u (%u) files done) [%s]
Work time: %d:%02d:%02d
Unable to get computer name
CryptoGuard
kernel32.dll
ConvertStringSecurityDescriptorToSecurityDescriptorW
advapi32.dll
IsWow64Process
SystemDrive
KiUserExceptionDispatcher

Table 11. Defray777 encrypted strings.

Prioritizing Defray777 on the Impacted System

While deep diving on a recovered Defray777 sample, we found that Defray777 exhibits the following notable characteristics regarding the prioritization of threads and processes:

- During execution, the ransomware uses SetProcessPriorityBoost to prioritize the threads of the Defray777 process.
- Defray777 additionally focuses on creating and prioritizing threads for encryption by calling SetThreadAffinityMask and SetThreadPriorityBoost.
- Defray777 uses multithreading to improve ransomware performance.



```
thread_handle = mw_create_execution_threads(mw_get_thread_start_addr, v0, 0);// Create the thread to set the affinity
if ( thread_handle )
{
  current_thread_handle = GetCurrentThread();
  SetThreadAffinityMask(current_thread_handle, v0);// Set affinity for the current thread
  SetThreadPriorityBoost(thread_handle, 0); // Enable dynamic boost for created threads
  CloseHandle(thread_handle);
}
```

Figure 22. Prioritization of Defray777 threads during execution.

Killing “Undesirable” Processes

As part of the execution workflow, Defray777 creates threads that will be responsible for the killing of processes that the threat actors deem to be “undesirable.” The execution continues by getting a listing of processes using CreateToolhelp32Snapshot before iterating through all active processes (with the exception of itself) and killing all “undesirable” processes. Defray777 specifically targets process that can be opened with the desired access of SYNCHRONIZE | PROCESS_QUERY_INFORMATION | PROCESS_VM_WRITE | PROCESS_VM_READ | PROCESS_VM_OPERATION | PROCESS_CREATE_THREAD.

Defray777 excludes all processes that contain the system file path in their full image path. Additionally, the ransomware will exclude the following processes from being killed during execution:

powershell.exe	rundll32.exe
wefault.exe	explorer.exe
vmnat.exe	

Table 12. Excluded processes.

Stopping System Services

During execution, Defray777 stops the following services from running:

Acronis VSS Provider	MSExchangeADTopology	MSSQLSERVER	SQLAgent$PRACTT
AcronisAgent	MSExchangeAntispamUpdate	MSSQLServerADHelper	SQLAgent$PROD
AcronixAgent	MSExchangeEdgeSync	MSSQLServerADHelper100	SQLAgent$PROFXE

AcrSch2Svc	MSExchangeES	MSSQLServerOLAPService	SQLAgent$SBSMON
Antivirus	MSExchangeFBA	MySQL57	SQLAgent$SHAREP
ARSM	MSExchangeFDS	MySQL80	SQLAgent$SOPHOS
AVP	MSExchangeIS	NetMsmqActivator	SQLAgent$SQL_200
BackupExecAgentAccelerator	MSExchangeMailboxAssistants	nginx	SQLAgent$SQLEXP
BackupExecAgentBrowser	MSExchangeMailboxReplication	ntrtscan	SQLAgent$SYSTEM
BackupExecDeviceMediaService	MSExchangeMailSubmission	OracleClientCache80	SQLAgent$TPS
BackupExecJobEngine	MSExchangeMGMT	OracleServiceXE	SQLAgent$TPSAMA
BackupExecManagementService	MSExchangeMTA	OracleXETNSListener	SQLAgent$VEEAMS
BackupExecRPCService	MSExchangeProtectedServiceHost	PDVFSService	SQLAgent$VEEAMS
BackupExecVSSProvider	MSExchangeRepl	POP3Svc	SQLBrowser
bedbg	MSExchangeRPC	ReportServer	SQLsafe Backup Se
DbxSvc	MSExchangeSA	ReportServer$SQL_2008	SQLsafe Filter Servi
DCAgent	MSExchangeSearch	ReportServer$SYSTEM_BGC	SQLSafeOLRServic
EhttpSrv	MSExchangeServiceHost	ReportServer$TPS	SQLSERVERAGEN
ekrn	MSExchangeSRS	ReportServer$TPSAMA	SQLTELEMETRY
Enterprise Client Service	MSExchangeThrottling	RESvc	SQLTELEMETRY$E
EPSecurityService	MSExchangeTransport	sacsvr	SQLWriter
EPUpdateService	MSExchangeTransportLogSearch	SamSs	SstpSvc
EraserSvc11710	msftesql$PROD	SAVAdminService	svcGenericHost
EsgShKernel	MSOLAP$SQL_2008	SAVService	swi_filter
ESHASRV	MSOLAP$SYSTEM_BGC	SDRSVC	swi_service
FA_Scheduler	MSOLAP$TPS	SepMasterService	swi_update
IISAdmin	MSOLAP$TPSAMA	ShMonitor	swi_update_64
IMAP4Svc	MSSQL$BKUPEXEC	Smcinst	Symantec System R
KAVFS	MSSQL$ECWDB2	SmcService	TmCCSF
KAVFSGT	MSSQL$PRACTICEMGT	SMTPSvc	tmlisten
kavfsslp	MSSQL$PRACTTICEBGC	SNAC	TrueKey
klnagent	MSSQL$PROD	SntpService	TrueKeyScheduler
macmnsvc	MSSQL$PROFXENGAGEMENT	Sophos Agent	TrueKeyServiceHelp
masvc	MSSQL$SBSMONITORING	Sophos AutoUpdate Service	UI0Detect
MBAMService	MSSQL$SHAREPOINT	Sophos Clean Service	Veeam Backup Cata Service
MBEndpointAgent	MSSQL$SOPHOS	Sophos Device Control Service	VeeamBackupSvc
McAfeeEngineService	MSSQL$SQL_2008	Sophos File Scanner Service	VeeamBrokerSvc
McAfeeFramework	MSSQL$SQLEXPRESS	Sophos Health Service	VeeamCatalogSvc
McAfeeFrameworkMcAfeeFramework	MSSQL$SYSTEM_BGC	Sophos MCS Agent	VeeamCloudSvc
McShield	MSSQL$TPS	Sophos MCS Client	VeeamDeploymentS
McTaskManager	MSSQL$TPSAMA	Sophos Message Router	VeeamDeploySvc
mfefire	MSSQL$VEEAMSQL2008R2	Sophos Safestore Service	VeeamEnterpriseMa

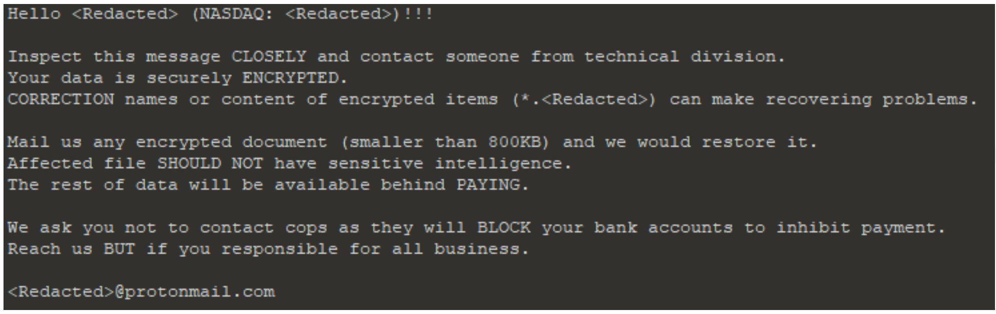
mfemms	MSSQL$VEEAMSQL2012	Sophos System Protection Service	VeeamHvIntegration
mfevtp	MSSQLFDLauncher	Sophos Web Control Service	VeeamMountSvc
MMS	MSSQLFDLauncher$PROFXENGAGEMENT	sophossps	VeeamNFSSvc
MongoDB	MSSQLFDLauncher$SBSMONITORING	SQL Backups	VeeamRESTSvc
mozyprobackup	MSSQLFDLauncher$SHAREPOINT	SQLAgent$BKUPEXEC	VeeamTransportSvc
MsDtsServer	MSSQLFDLauncher$SQL_2008	SQLAgent$CITRIX_METAFRAME	W3Svc
MsDtsServer100	MSSQLFDLauncher$SYSTEM_BGC	SQLAgent$CXDB	wbengine
MsDtsServer110	MSSQLFDLauncher$TPS	SQLAgent$ECWDB2	WRSVC
MSExchangeAB	MSSQLFDLauncher$TPSAMA	SQLAgent$PRACTTICEBGC	Zoolz 2 Service

Table 13. Services stopped by Defray777.

File Encryption

Based on a recent Defray777 sample recovered from memory, the ransomware will get a listing of all logical drives on the system using a call to GetLogicalDriveStringsW before iterating through each drive to encrypt files using the following process:

- To begin, Defray777 checks for whether the processor feature PF_XMMI64_INSTRUCTIONS_AVAILABLE is present on the impacted system.
  If enabled, Defray777 knows that SSE2 is supported and more complex mathematical operations are possible.
- Defray777 will also determine if the processor is capable of using AES-NI for improved encryption performance.
- As encryption begins, a ransom note will be created in each directory where files will be encrypted.
  - The name of the ransom note will vary. However, from our research, the ransom notes most commonly contain a combination of exclamation points, the string “README,” and a reference to the victim name.
  - Example: !!!_IMPACTED_Client_README_!!!.txt
- The file contents will be encrypted using an on-the-fly generated AES key that gets encrypted with RSA-4096 and stored in the file footer in a 512-byte block.
- The encrypted file will be renamed by appending an extension that consists of a unique victim identifier and a randomized eight-digit hexadecimal number.
  Example: .v1ct1m-1bc461ac



```
Hello <Redacted> (NASDAQ: <Redacted>)!!!

Inspect this message CLOSELY and contact someone from technical division.
Your data is securely ENCRYPTED.
CORRECTION names or content of encrypted items (*.<Redacted>) can make recovering problems.

Mail us any encrypted document (smaller than 800KB) and we would restore it.
Affected file SHOULD NOT have sensitive intelligence.
The rest of data will be available behind PAYING.

We ask you not to contact cops as they will BLOCK your bank accounts to inhibit payment.
Reach us BUT if you responsible for all business.

<Redacted>@protonmail.com
```

Figure 23. Recent example of a Defray777 ransom note.

Specifically, the encryption mechanism consists of the following steps:

- Dynamically generate a 32-byte AES key.
- Encrypt the file with AES-256 in ECB mode using 16-byte blocks.
- Encrypt the AES key with RSA-4096 and append the 0x200 byte cipher text to the end of the encrypted file.

Encryption Exclusions

During the encryption process, Defray777 aims to encrypt as many files as possible without impacting the system’s core functionality. To accomplish this, Defray777 uses a set of excluded folders, files and file extensions that will not be encrypted during execution.

Excluded Folders:

\windows\system32\	\windows\syswow64\	\windows\system\
\windows\winsxs\	\appdata\roaming\	\appdata\local\
\appdata\locallow\	\all users\microsoft\	\inetpub\logs\

:\boot\	:\perflogs\	:\programdata\
:\drivers\	:\wsus\	:\efstmpwp\
:\$recycle.bin\	:\EFSTMPWP\	crypt_detect
cryptolocker	ransomware	

Table 14. Folders excluded from encryption by Defray777.

Excluded files:

iconcache.db	thumbs.db	ransomware	ransom
debug.txt	boot.ini	desktop.ini	autorun.inf
ntuser.dat	ntldr	ntdetect.com	bootfont.bin
bootsect.bak			

Table 15. Files excluded from encryption by Defray777.

It is also important to note that Defray777 adds the name of the ransom note into the excluded files list.

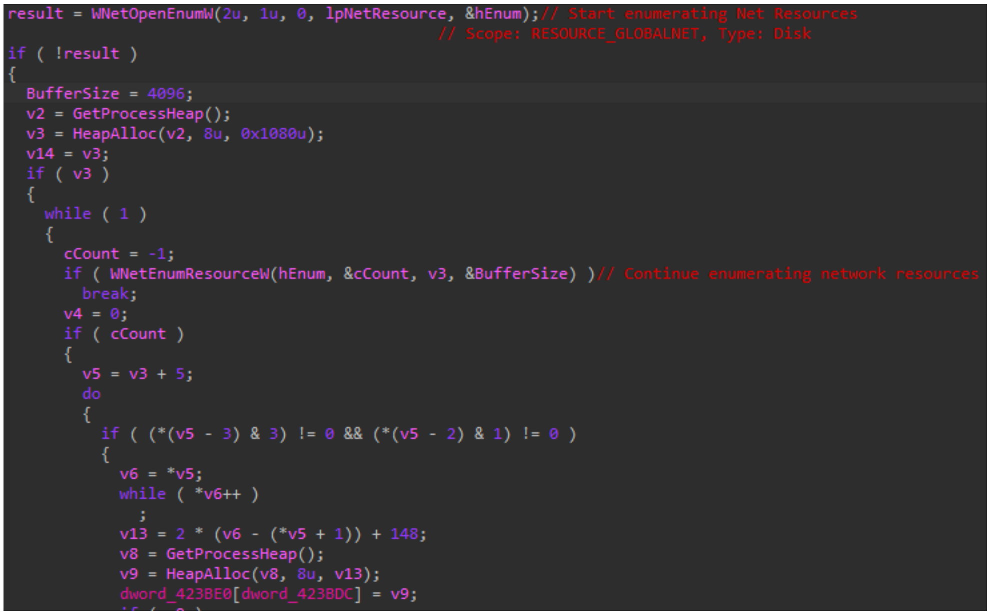
Excluded extensions:

.ani	.cab	.cpl	.cur	.diagcab
.diagpkg	.dll	.drv	.hlp	.icl
.icns	.ico	.iso	.ics	.lnk
.idx	.mod	.mpa	.msc	.msp
.msstyles	.msu	.nomedia	.ocx	.prf
.rtp	.scr	.shs	.spl	.sys
.theme	.themepack	.exe	.bat	.cmd
.url	.mui			

Table 16. Extensions excluded from encryption by Defray777.

Searching for Unmapped File Shares

During execution, Defray777 uses WNetOpenEnumW and WNetEnumResourceW to search for file shares that may contain files that could be encrypted. This tactic has been seen amongst other ransomware variants in the wild to encrypt files that are accessible via unmapped file shares.



```
result = WNetOpenEnumW(2u, 1u, 0, lpNetResource, &hEnum);// Start enumerating Net Resources
                                                        // Scope: RESOURCE_GLOBALNET, Type: Disk
if ( !result )
{
  BufferSize = 4096;
  v2 = GetProcessHeap();
  v3 = HeapAlloc(v2, 8u, 0x1080u);
  v14 = v3;
  if ( v3 )
  {
    while ( 1 )
    {
      cCount = -1;
      if ( WNetEnumResourceW(hEnum, &cCount, v3, &BufferSize) )// Continue enumerating network resources
        break;
      v4 = 0;
      if ( cCount )
      {
        v5 = v3 + 5;
        do
        {
          if ( (*(v5 - 3) & 3) != 0 && (*(v5 - 2) & 1) != 0 )
          {
            v6 = *v5;
            while ( *v6++ )
              ;
            v13 = 2 * (v6 - (*v5 + 1)) + 148;
            v8 = GetProcessHeap();
            v9 = HeapAlloc(v8, 8u, v13);
            dword_423BE0[dword_423BDC] = v9;
```

Figure 24. Defray777 enumerating network resources.

Anti-Forensic Measures

After all files are encrypted on the system, Defray777, like many other ransomware variants, implements common anti-forensics measures to remove as much evidence of the intrusion as possible and make it extremely difficult for the system to be recovered without a backup. Although these commands are common amongst other ransomware variants, Defray777 runs commands post-encryption, which means that when security tools alert or take action against Defray777, the files have already been encrypted.

Commands executed by Defray777:

```
cipher.exe /w:[DRIVE]
fsutil.exe usn deletejournal /D [DRIVE]
wbadmin.exe delete catalog -quiet
bcdedit.exe /set {default} recoveryenabled no
bcdedit.exe /set {default} bootstatuspolicy ignoreallfailures
schtasks.exe /Change /TN "\Microsoft\Windows\SystemRestore\SR" /disable
wevtutil.exe cl Application
wevtutil.exe cl System
wevtutil.exe cl Setup
wevtutil.exe cl Security
wevtutil.exe sl Security /e:false
```

Table 17. Anti-forensic commands executed by Defray777.

Registry keys modified:

```
\Software\Microsoft\Windows NT\CurrentVersion\SystemRestore\DisableConfig
\Software\Microsoft\Windows NT\CurrentVersion\SystemRestore\DisableSR

\SOFTWARE\Policies\Microsoft\Windows NT\SystemRestore\DisableSR

\SOFTWARE\Policies\Microsoft\Windows NT\SystemRestore\DisableConfig
```

Table 18. Registry Keys Modified by Defray777.

Defray777's Port to Linux

During the course of our research, we found that Defray777 ransomware has been ported over to Linux. Before Defray777, ransomware that impacted both Windows and Linux operating systems was limited to being written in Java or scripting languages such as Python. These ransomware variants would be considered cross-functional since they were written in a single language that must be installed and supported by both operating systems. Defray777's port to Linux ensures that the ransomware has standalone executables for each platform with no external dependencies.

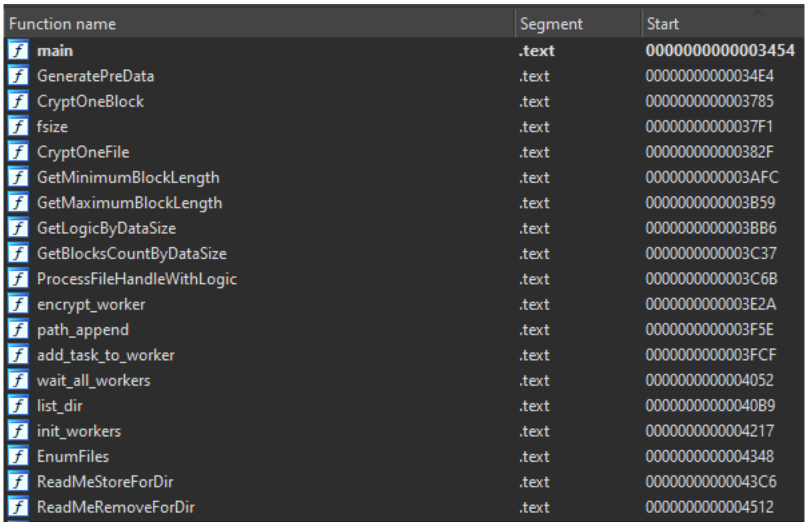
A ZIP archive was uploaded to a public malware repository on Oct. 17, 2020 that contained a Windows executable that was identified as a Defray777 decryptor. Additionally included in this ZIP archive was an ELF binary named decryptor64. Analysis of this binary determined it to be another Defray777 decryptor that had been ported to Linux.

Armed with the idea that there may be Linux versions of Defray777 in the wild, we began hunting in AutoFocus and quickly uncovered an ELF version of the ransomware encryptor.

Reviewing this sample further indicated that it was uploaded in August 2020. As of early October 2020, there appear to be zero detections by antivirus (AV) in VirusTotal for the Linux version of Defray777.

A deeper review of the Linux and Windows variants of Defray777 determined that the encryption and decryption processes used were nearly identical. In fact, by generating our own RSA key pair and modifying the binaries, we were able to confirm that the encryptors and decryptors for both operating systems were interchangeable.

Unlike the Windows versions, the developers didn't seem to put any effort into protecting the Linux samples. To our surprise, the binaries we analyzed still had their symbols intact, which made reversing them quite a bit easier.



Function name	Segment	Start
main	**.text**	**0000000000003454**
GeneratePreData	.text	00000000000034E4
CryptOneBlock	.text	0000000000003785
fsize	.text	00000000000037F1
CryptOneFile	.text	000000000000382F
GetMinimumBlockLength	.text	0000000000003AFC
GetMaximumBlockLength	.text	0000000000003B59
GetLogicByDataSize	.text	0000000000003BB6
GetBlocksCountByDataSize	.text	0000000000003C37
ProcessFileHandleWithLogic	.text	0000000000003C6B
encrypt_worker	.text	0000000000003E2A
path_append	.text	0000000000003F5E
add_task_to_worker	.text	0000000000003FCF
wait_all_workers	.text	0000000000004052
list_dir	.text	00000000000040B9
init_workers	.text	0000000000004217
EnumFiles	.text	0000000000004348
ReadMeStoreForDir	.text	00000000000043C6
ReadMeRemoveForDir	.text	0000000000004512

Figure 25. Named functions listing from ELF version of Defray777.

One of the biggest differences between the Windows and Linux variants is the logic that determines which files to encrypt. The Windows version will recurse the file system and encrypt anything that isn't explicitly excluded. In contrast, the Linux variant will only encrypt directories specified in a command line argument.

Continue reading: Linking Vatet, PyXie and Defray777