DarkSide ransomware analysis

By Zawadi Done

Published: 2020-10-05 · Archived: 2026-04-05 17:45:44 UTC

This blog post will try to explain how the ransomware called DarkSide works. Based on my research, this ransomware uses Salsa20 encryption to encrypt files and RSA encryption to encrypt the key used by Salsa20. A new key is created per file based on random bytes.

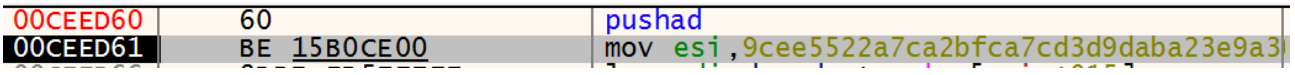
> A new ransomware operation named DarkSide began attacking organizations earlier this month with customized attacks that have already earned them million-dollar payouts.
>
> Starting around August 10th, 2020, the new ransomware operation began performing targeted attacks against numerous companies.
>
> In a “press release” issued by the threat actors, they claim to be former affiliates who had made millions of dollars working with other ransomware operations.
> https://www.bleepingcomputer.com/news/security/darkside-new-targeted-ransomware-demands-million-dollar-ransoms/amp/

UnpackingPermalink

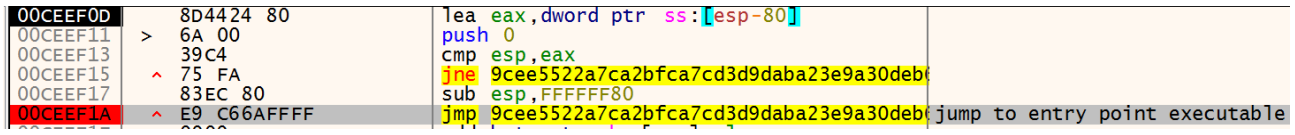
The executable is compressed with UPX

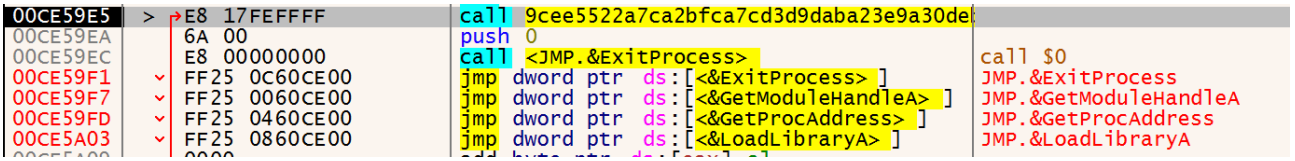
```
file 9cee5522a7ca2bfca7cd3d9daba23e9a30deb6205f56c12045839075f7627297
[...]: PE32 executable (GUI) Intel 80386, for MS Windows, UPX compressed
```

After the first instruction `pushad` I put a breakpoint on the `ESP` register and continue.



The execution breaks on the instruction `lea eax, dword ptr ss:[esp80]`. After the loop is executed it jumps to the entry point of the packed executable.





Once the executable is unpacked, we can analyze the ransomware

Anti-analysisPermalink

To make static analysis harder the ransomware resolves DLL's and API calls dynamically using `LoadLibrary`, `GetProcAddress` and 2 custom functions shown below. In this screenshot, the address of `_wcsicmp` is resolved in memory.

```
push dword ptr ds:[esi-4]
push esi                                    esi:"ntdll"
call 9cee5522a7ca2bfca7cd3d9daba23e9a30deb6205f56c12
push esi                                    esi:"ntdll"
call <JMP.&LoadLibraryA>
mov ebx,eax
push dword ptr ds:[esi-4]
push esi                                    esi:"ntdll"
call 9cee5522a7ca2bfca7cd3d9daba23e9a30deb6205f56c12
```

```
push dword ptr ds:[esi-4]
push esi                                    esi:"_wcsicmp"
call 9cee5522a7ca2bfca7cd3d9daba23e9a30deb6205f56c12
push ecx
push esi                                    esi:"_wcsicmp"
push ebx
call <JMP.&GetProcAddress>
```

PreparationPermalink

The mutex `Global\\3e93e49583d6401ba148cd68d1f84af7` is created to make sure only one copy of the ransomware is running, otherwise the ransomware exits. This is done based on the name of the executable. Then `SetThreadExecutionState` is called to force the system to be in the working state by resetting the system idle timer.

Services

To make sure certain services are not running the following services are stopped using `ControlService - SERVICE_CONTROL_STOP` and `DeleteService`. Deleting a service is not useful if an organization pays the ransom and wants to go back into production quickly. As a system administrator, I wouldn't be happy about this.

- vss
- sql
- svc$
- memtas
- mepocs
- sophos
- veeam
- backup

```
push eax
push 1
push dword ptr ss:[ebp-8]
call dword ptr ds:[<&ControlService> ]
test eax,eax
je 9cee5522a7ca2bfca7cd3d9daba23e9a30deb6205
push dword ptr ss:[ebp-8]
call dword ptr ds:[<&DeleteService> ]
push dword ptr ss:[ebp-8]
call dword ptr ds:[<&CloseServiceHandle> ]
jmp 9cee5522a7ca2bfca7cd3d9daba23e9a30deb620
```

Shadow Copies

Using `CreateProcessW` the following Powershell script is executed which deletes Shadow Volume Copies.

```
powershell -ep bypass -c \"(0..61)|%{$s+=[char][byte]('0x'+'4765742D576D694F626A6563742057696E33325F536861646F
```

> When deobfuscated, we can see that this PowerShell command is used to delete Shadow Volume Copies on the machine before encrypting it.
>
> ```
> Get-WmiObject Win32_Shadowcopy | ForEach-Object {$_.Delete();}
> ```
>
> https://www.bleepingcomputer.com/news/security/darkside-new-targeted-ransomware-demands-million-dollar-ransoms/amp/

Processes

To make sure certain processes are not running a list of processes are terminated (https://pastebin.com/WWSQxhcq.

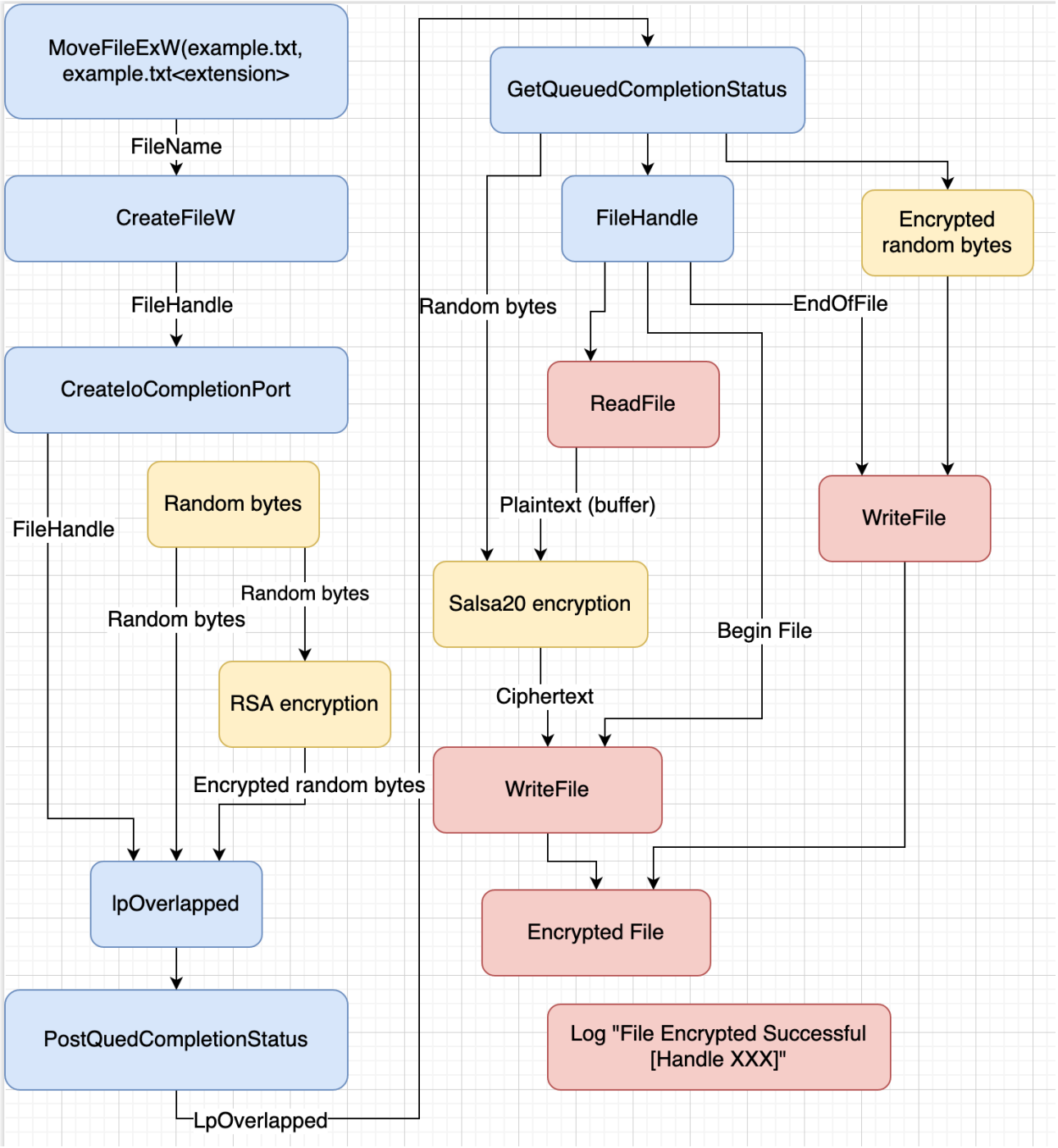
```
push dword ptr ss:[ebp-22C]
push 0
push 1
call dword ptr ds:[<&OpenProcess> ]
mov dword ptr ss:[ebp-8],eax
cmp dword ptr ss:[ebp-8],0
je 9cee5522a7ca2bfca7cd3d9daba23e9a30deb6205f5
push 0
push dword ptr ss:[ebp-8]
call dword ptr ds:[<&TerminateProcess> ]
push dword ptr ss:[ebp-8]
call dword ptr ds:[<&CloseHandle> ]
```

EncryptionPermalink

The encryption routine skips a few files, file extensions and directories (https://pastebin.com/WWSQxhcq).

Encryption flowchartPermalink

The encryption routine of the ransomware is shown below.



Debugging modePermalink

I don't know why but it seems the authors have forgotten to disable debugging functionality in their code or maybe they are using this to verify that the files are encrypted. (XXX = file name). This file was in the same directory as the executable.

```
 [INF] Start Encrypting All Files
[INF] Emptying Recycle Bin
[INF] Uninstalling Services
[INF] Deleting Shadow Copies
[INF] Terminating Processes
[INF] Encrypt Mode - FAST
[INF] Encrypting Local Disks
[INF] Started 8 I/O Workers
[INF] Start Encrypt [Handle 492] \\?\C:\XXX
[INF] File Encrypted Successful [Handle 492]
[INF] Start Encrypt [Handle 640] \\?\C:\XXX
[INF] File Encrypted Successful [Handle 640]
[INF] Start Encrypt [Handle 640] \\?\C:\XXX
[...]
```

IOCPermalink

SHA256 - `9cee5522a7ca2bfca7cd3d9daba23e9a30deb6205f56c12045839075f7627297`

ReferencesPermalink

https://www.bleepingcomputer.com/news/security/darkside-new-targeted-ransomware-demands-million-dollar-ransoms/amp/

https://tria.ge/200828-r31s5nvvm2/behavioral1

Source: https://zawadidone.nl/darkside-ransomware-analysis/