DEFEATING RANSOMWARE BY HOOKING SYSTEM CALLS ON WINDOWS OS

Filip Touš

Bohumil Hrabal Grammar School Nymburk (4)

E-mail: smfitous@gmail.com

Abstract: This paper explains why ransomware needs to use the Windows API to encrypt files and how this can be utilized to protect sensitive data from ransomware. Critical API functions are examined on a low level and a generic method to monitor and possibly block their usage through system call hooks is presented. This approach is then demonstrated with a custom kernel mode driver which can keep protected files safe from any user mode malware. It is then compared to current ransomware protection in Windows 10.

Keywords: ransomware, Windows API, system call, hooking

1 ÚVOD

Ransomware je typ malwaru, který na napadených zařízeních blokuje přístup k souborům jejich zašifrováním nebo celému počítači (například přepsáním master boot record sektoru na disku). Po infikování a šifrování je uživateli zobrazena zpráva o výkupném, které obvykle musí být zaplaceno ve virtuální měně jako Bitcoin nebo Ethereum. Klasický krypto ransomware (tedy ten, který šifruje jednotlivé soubory) sleduje typický vzorec chování:

1. infekce PC,
2. nalezení a šifrování souborů,
3. zobrazení zprávy požadující zaplacení pro dešifrování dat.

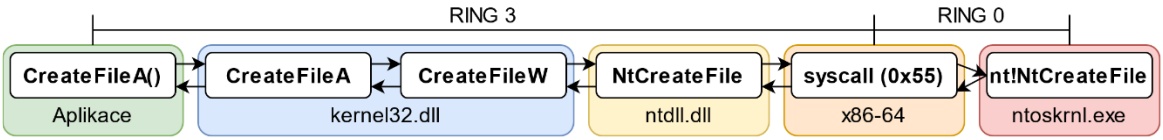
Tento druh malwaru se na OS Windows neobejde bez použití Windows API pro provedení zmíněných kroků [1]. To dokládá i analýza několika známých vzorků ransowmaru (WannaCry, CTB-Locker, Revenge a další) v online nástroji VirusTotal. Díky tomu byly v minulosti navrženy mechanismy pro detekování ransomwaru statickou nebo behaviorální analýzou volání specifických API funkcí [2] [3]. Cílem softwaru navrženého v rámci této práce je však kompletně ransomwaru zablokovat přístup k citlivým souborům a zastavit tak ransomware dříve, než je schopen napáchat škody.

2 WINDOWS API

Funkce pro I/O a manipulaci se soubory jsou součástí Win32 API. Téměř všechny vyžadují poskytnutí handlu na soubor jako parametr, a to s adekvátními právy (např. WriteFile vyžaduje vytvoření handlu s parametrem dwDesiredAccess „GENERIC_WRITE“). Výjimky jsou uvedeny dále v textu. Handle k souboru vrací funkce CreateFile, alternativně starší OpenFile. Přesněji se jedná o funkce s příponou -W. CreateFileA (druhá z používaných přípon) jen konvertuje cestu k souboru z ANSI do Unicode a volá CreateFileW, podobně funguje mnoho funkcí Win32 API.

Převzetím kontroly nad těmito funkcemi (CreateFileW a OpenFileW) by teoreticky šlo libovolně blokovat přístup k souborům a chránit je tak před ransomwarem. Tyto vysokoúrovňové funkce jsou ale ve skutečnosti jen wrappery pro funkce Native API s předponou Nt (NtCreateFile, NtOpenFile). K těm často neexistuje oficiální dokumentace. Ačkoliv jsou v systémové hierarchii o úroveň níže oproti Win32 API, ani hookování těchto funkcí není spolehlivé, neboť stále běží v uživatelském prostředí (ring 3) a tyto hooky lze obcházet. Z tohoto důvodu je nutné převzít kontrolu nad voláním API funkcí až na kernelové úrovni (ring 0), k čemuž je na Windows zapotřebí vytvořit ovladač.

Software navržený v rámci práce funkce hookuje až na úrovni systémových volání (system call) ve Windows. Jedná se o rozhraní umožňující aplikacím používat funkce operačního systému, klasicky skrze výše zmíněná API. V praxi funkce jako NtCreateFile vyvolají system call pomocí softwarového přerušení (v x86-64 instrukce syscall). Následně je dočasně zvýšena úroveň oprávnění CPU a kontrola je předána na předem definované místo v kernelu (konkrétně v ntoskrnl.exe, ve Správci úloh se jedná o proces „System“). Před vykonáním syscall instrukce je do EAX registru uložen index systémového volání / funkce OS, která má být volána a parametry jsou uloženy na zásobník. Po splnění požadavku a návratu z funkce OS je úroveň oprávnění CPU nastavena zpět na uživatelské prostředí. Celý proces ilustruje obrázek 1:



Obrázek 1: Cesta volání z CreateFileA do kernelu

Jak bylo zmíněno výše, některé API funkce na první pohled nevyžadují vytvoření handlu přes CreateFile nebo OpenFile, například CopyFile, DeleteFile, MoveFile, ReplaceFile nebo SetFileAttributes, protože jejich signatura neobsahuje handle parametr. Analýza takovýchto funkcí v disassembleru ale odhaluje, že odpovědnost za vytvoření handlu řeší sami a ve výsledku vždy dojde k systémovému volání NtCreateFile nebo NtOpenFile před provedením změn. Někdy skrze funkce vyšší úrovně (například CopyFileW volá CreateFileW skrze BasepCopyFileExW), v některých případech dochází rovnou k volání nativních funkcí (DeleteFileW volá NtOpenFile). To platí pro Win32 i Native API.

Jedinou skutečnou výjimkou je nativní funkce NtDeleteFile, ke které Microsoft ani neposkytuje kompletní informace v PDB. Nutno podotknout, že funkce vyšší úrovně DeleteFile nevolá NtDeleteFile, soubory jsou místo toho označeny k vymazání pomocí NtSetInformationFile. NtDeleteFile umožňuje soubor vymazat ihned (nečeká na uzavření všech handlů k souboru) a podle statické analýzy nevolá NtCreateFile nebo NtOpenFile i přesto, že jako parametr akceptuje i pouhou cestu k souboru. Některý ransomware manipuluje se soubory přímo, jindy je vytvořena kopie, která je poté zašifrována a původní soubor smazán. Navržený software se tedy zajímá celkem o tři systémová volání:

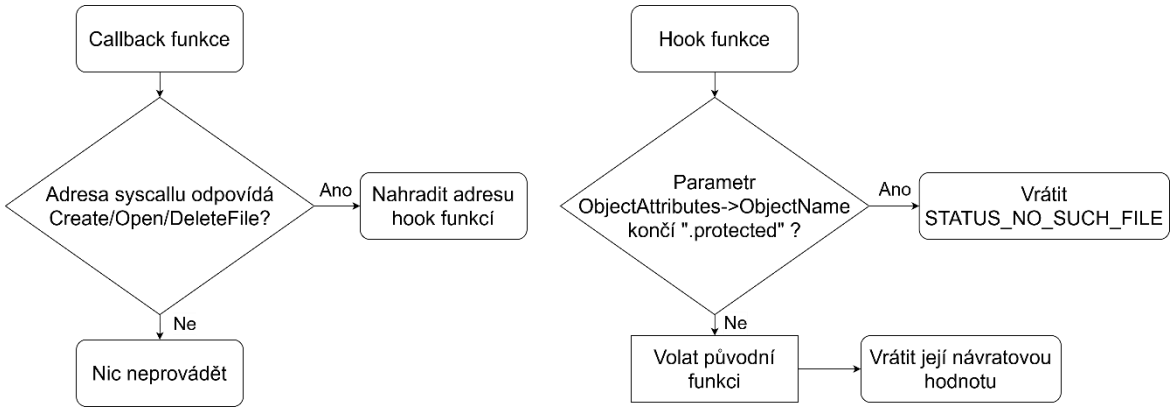
- NtCreateFile
- NtOpenFile
- NtDeleteFile

3 BLOKOVÁNÍ PŘÍSTUPU K SOUBORŮM

K hookování těchto systémových volání byla použita knihovna libfinityhook (součást InfinityHook). Software je sestaven jako primitivní ovladač (od Windows 10 verze 1903 takto musí být sestaveny všechny ovladače, které nejsou vázány na specifický hardware). Když dojde k systémovému volání, volá vlastní callback funkci, ve které je možné nahradit adresu cílené funkce v ntoskrnl.exe vlastní hook funkcí ještě před tím, než ji OS volá. Pokud adresa odpovídá nt!NtCreateFile, software ji nahradí adresou své vlastní funkce HookNtCreateFile (analogicky hookuje i další dvě funkce). Prvotní funkce ovladače (DriverEntry) získává adresy funkcí v ntoskrnl.exe pomocí MmGetSystemRoutineAddress a inicializuje callback funkci.

Uvnitř hook funkcí navržený software rozhoduje o zablokování nebo povolení požadavku. Pro testování jednoduše blokuje přístup ke všem souborům s příponou „protected“. Alternativně je možné rozhodovat na základě umístění souboru pro ochranu celých složek. Tyto informace funkce obdrží v parametru ObjectAttributes. Software však není limitovaný jen funkcí samotnou a může volat jiné funkce, což umožňuje implementovat komplexnější nastavení ochrany. V případě, že je zachycena snaha o vytvoření handlu k chráněnému souboru (nebo jeho smazání), je kompletně popřena

existence souboru (vrácená hodnota STATUS_NO_SUCH_FILE). Pokud nedojde k zablokování požadavku, software volá původní, pravou funkci v ntoskrnl.exe. Celý proces popisuje obrázek 2. Software pro ukázku blokuje i žádosti o read-only přístup, neboť i krádež dat představuje riziko pro firmy a možný peněžní zisk pro hackery [4]. Je ale možné jen manipulovat s parametrem DesiredAccess a zamezit pouze přepisování nebo mazání souboru a čtení vždy povolovat.



Obrázek 2: Vývojový diagram ovladače

Po načtení vytvořeného ovladače je jakákoliv manipulace z uživatelského prostředí s chráněnými soubory zablokována. Software byl otestován na Windows 10 s vypnutým Windows Defenderem proti ransomwaru WannaCry. I po zobrazení zprávy o výkupném zůstaly soubory obsahující v názvu řetězec „protected“ nepoškozené (pro vynucení šifrování od WannaCry byla podmínka v ovladači upravena, aby skutečná přípona souborů mohla být „.docx“).

Navržená metoda ochrany je na rozdíl od behaviorální detekce podezřelého používání API funkcí výpočetně i implementačně nenáročná a je 100% úspěšná při ochraně souborů před jakýmkoliv malwarem. Metoda v [3] průměrně ztratila 10 souborů před detekováním ransomwaru. Implementace v [2] hookovala API funkce přímo v paměti procesu v uživatelském prostředí. Tyto hooky lze odstranit a případně i kompletně obejít manuálním systémovým voláním pomocí assembly kódu. V obou případech analýza sekvencí API funkcí a chování programů spotřebovávala mnoho výpočetního času. V [3] došlo ke zvýšení latence operací se soubory v řádu milisekund. Metoda navržená v této práci nezpůsobuje žádné měřitelné zpomalení při tvorbě handlů a ostatní funkce nejsou ovlivněny (kromě NtDeleteFile, kterou by legitimní software používat neměl).

4 POROVNÁNÍ S CONTROLLED FOLDER ACCESS

Controlled folder access („Řízený přístup ke složkám“, dále jen CFA) je vestavěná součást protekce před viry ve Windows 10. Jejím účelem je chránit data před ransomwarem nebo jinými hrozbami. Uživatel může specifikovat složky, nebo celé disky, ke kterým CFA zablokuje přístup všem nepovoleným aplikacím.

CFA má však několik zásadních problémů, tím prvním je fakt, že uživatel nemůže vypnout ochranu výchozích složek. Konkrétně se jedná například o složky „Dokumenty“ všech uživatelů. Jelikož spousta legitimních aplikací stále ukládá data do těchto míst, nemá uživatel jinou možnost než povolovat tyto aplikace a tím jim udělovat kompletní přístup ke všem chráněným složkám a diskům.

Některé systémové aplikace / procesy jsou povoleny vždy, Windows je označuje jako „přátelské aplikace“. Jednou z nich je explorer.exe (Průzkumník Windows), aby uživatel nezablokoval přístup ke složkám „sám sobě“. Mezi přátelské aplikace ale patří například i notepad.exe (Poznámkový blok). V grafickém rozhraní pro povolení aplikace je také napsáno, že „Řízený přístup ke složkám bude povolovat většinu aplikací, aniž byste je sem museli přidat“. Uživatel tak ani nad tímto nemá absolutní kontrolu.

Všechny povolené aplikace jsou zneužitelné k získání přístupu ke všem chráněným složkám. Útočník má několik možností:

- Provádět změny přes již spuštěný (a povolený) proces, například vytvořením nového vlákna nebo pomocí thread hijackingu.
- Skrytě spustit povolenou aplikaci a ihned nad procesem převzít kontrolu.
- Nahradit .exe soubor svým vlastním (pokud není také chráněn) a soubor spustit.

Všemi třemi metodami se podařilo ochranu obejít. CFA nekontroluje, zda bylo se souborem nebo procesem manipulováno. Stačí, aby kompletní cesta (včetně názvu) ke spouštěnému souboru byla v seznamu povolených aplikací. Ten je navíc volně přístupný v registru jako názvy hodnot specifického klíče, k jehož přečtení není zapotřebí administrátorských privilegií. Malware tak má k dispozici mnoho jednoduchých a 100% funkčních možností k obcházení CFA. Pokud se malwaru podaří získat administrátorská práva, může dokonce manipulovat přímo s jeho nastavením pomocí Powershell příkazů Add-MpPreference a Set-MpPreference. Tyto změny vejdou v platnost až po restartu systému, ale v obou případech není uživatel o změnách informován.

Navržený software těmito problémy netrpí a bez přístupu ke kernelu ho nelze nijak obejít nebo s ním manipulovat. Jeho aktuální implementace však nedovoluje povolovat specifickým aplikacím přístup ke chráněným souborů. Tato varianta po vzoru CFA (a komerčních řešení jako Avast Ransomware Shield) by pro spolehlivé zabezpečení vyžadovala nejen ověřování pravosti .exe souborů, ale i integrity spuštěného procesu, všech jeho vláken a funkcí (a ukazatelů na ně) v paměti. Přísná ochrana použitá v této implementaci by mohla být vhodná pro protekci záloh nebo velmi citlivých souborů. S chráněnými soubory je možné manipulovat skrze ovladač, ten může případně poskytovat zabezpečené rozhraní pro komunikaci z uživatelského prostředí.

5 ZÁVĚR

V práci byly představeny funkce Windows API, které ransomware musí použít k získání přístupu k jednotlivým souborům před jejich šifrováním. Bylo vysvětleno, jak lze jejich používání monitorovat nebo blokovat z kernelu pomocí hookování systémových volání. Na základě toho byl vytvořen vlastní ovladač, který touto metodou blokuje přístup ke všem souborům s koncovkou „protected“ pro demonstraci navrženého ochranného systému. Ten úspěšně zabránil poškození těchto souborů ransomwarem WannaCry. Použitá metoda byla porovnána s aktuální ochranou proti ransomwaru ve Windows 10 (Controlled folder access), kterou lze na rozdíl od navrženého systému obcházet i z uživatelského prostředí. Ochrana souborů hookováním systémových volání je také oproti dříve navrženým metodám detekce ransomwaru na základě používání Windows API výpočetně i implementačně nenáročná a 100% úspěšná při ochraně souborů.

REFERENCE

[1] Hampton, N., Baig, Z., Zeadally, S.: Ransomware behavioural analysis on windows platforms. In: Journal of Information Security and Applications. 2018, s. 44-51. ISSN 22142126

[2] Ki, Y., Kim, E., Kim, H.: A Novel Approach to Detect Malware Based on API Call Sequence Analysis. In: International Journal of Distributed Sensor Networks. 2015, ISSN 1550-1477

[3] Scaife, N., Carter, H., Traynor, P., Butler, K.: CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data. In: 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS). 2016, s. 303-312. ISBN 978-1-5090-1483-5. ISSN 1063-6927

[4] Porter, J.: Cyberpunk 2077 studio falls victim to ransomware attack, data leak threatened: Hackers claim to have accessed source code for Cyberpunk 2077 and Witcher 3. In: The Verge [online]. Vox Media, LLC [cit. 2021-02-22]. Dostupné z: https://www.theverge.com/2021/2/9/22274035/cd-projekt-hack-source-code-cyberpunk-2077-witcher-3-encrypt-data-ransom