4ZA-02

Windows API コールのログからのマルウェアの動作再現の検討

末吉　真也[†]　福田　洋治[†]　廣友　雅徳[‡]　毛利　公美[*]　白石　善明[††]
近畿大学[†]　佐賀大学[‡]　岐阜大学[*]　神戸大学[††]

1. はじめに

情報セキュリティ対策におけるインシデント対応では，根絶と復旧の過程で，ネットワーク構成や保存されているログに基づき，被害を受けた可能性のある端末や情報を洗い出し，影響の範囲を確定させるといった，証拠の収集，処理が行われる[1)]．

証拠の収集，処理の場面では，保存されたログから，インシデント発生時の状況の再現，追跡が試みられることがあるが，インシデントにマルウェアを用いた攻撃が含まれている場合，現場でマルウェアそのものを扱うことは，情報漏洩などの被害が拡大する危険を伴う．

著者らは，マルウェアを直接扱うことなくその挙動を再現，観測できるようにするために， Windows API コール[2)]のログから，マルウェアの動作を再現，観測を行うツール[3)]（以下，本ツール）を継続して検討している．

本稿では，マルウェアでよく用いられる Win32 API のうち，ファイル・ディレクトリ操作，HTTP 通信の他に，新たにプロセス・スレッド管理，レジストリ操作の API に対応した，試作中のツールの機構，動作実験について述べる．

2. API ログからのマルウェア動作再現ツール

本ツールは Windows API を用いるマルウェアの API コールログを利用して，インシデント発生後でも端末上で当時の挙動を再現できるようにしたものである．本ツールの構成を図 1 に示す．

事前に対象のマルウェアを動的解析して，API コールログが得られていることを前提とし，動作再現を行いたい端末上にツールを設置し実行，API コールログのファイルを与える．

API コールログのファイルから記録された時刻順にログレコードを読み込み，以下の動作を繰り返す．（1）ログ抽出部は，1 つのログレコードから引数の文字列や API 名を抽出する．（2）API 判別部で API と引数の型を判別する．（3）型変換部は，戻り値・参照データ管理部が保持するデータの型を当該 API の引数の型に変換する．（4）API コール部は，当該 API に引数を与えて呼び出し，戻り値や参照データを得ると，それを戻り値・参照データ管理部に渡す．以上の繰り返し動作のうち，（1）の

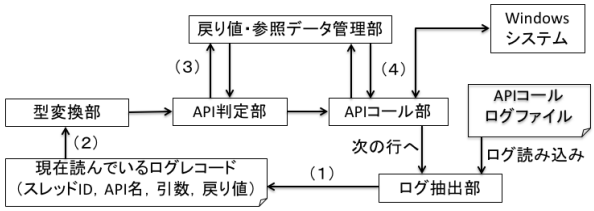

図 1　Windows API を用いるマルウェアの API コールを用いた動作再現ツールの構成

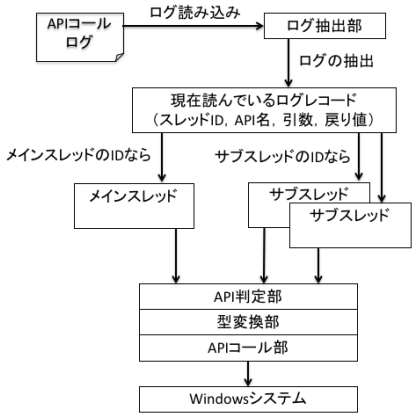

図 2　マルチスレッド処理を再現するための機構

API 判別部において，判別した API が未実装である場合は型変換部に送らず，次のログレコードを読み込む．

著者らはこれまでファイル・ディレクトリ，通信のカテゴリの基本的な Win32API (CopyFile, CreateFile, CreateDirectory, CreateDirectoryEx, DeleteFile, MoveFile, MoveFileEx, WriteFile, InternetReadFile, IntenetOpen, InternetOpenUrl, HttpOpenRequest, HttpSendRequest, HttpQueryInfo, InternetConnect, InternetCrackUrl, CloseHandle, InternetSetOption, InternetCloseHandle) に対応した実装を行ってきた[3)]．

これらに加えて新たにプロセス・スレッド管理のカテゴリの API (CreateEvent, OpenEvent, SetEvent, ResetEvent, CreateThread, SuspendThread, ExitThread, ResumeThread, WaitForSingleObject, Sleep) 10 種類，レジストリ操作のカテゴリの API (RegCreateKeyEx, RegSetValueEx, RegOpenKeyEx, RegQueryValueEx, RegCloseKey) 5 種類に対応する実装を行った．

プロセス・スレッド管理の API への対応に伴い，図 2 に示すようなマルチスレッド処理を再現するための機構を本ツールのログ抽出部に追加した．この機構は，ログに含まれるスレッド ID を用いて，メ

Consideration on Reproduction of Malware Behavior by Using Windows API Coll Logs
[†] Masaya SUEYOSHI, Youji FUKUTA, Kindai University
[‡] Masanori HIROTOMO, Saga University
[*] Masami MOHRI, Gifu University
[††] Yoshiaki SHIRAISHI, Kobe University

インのスレッド，もしくは新しく作られたサブスレッドのどちらで呼び出された API なのかを判別し，対応するスレッドから API 呼び出しを行うことで，擬似的にマルチスレッド処理を再現する．

本ツールはインシデント対応におけるフォレンジック支援のために，API コールログから当時のプログラムの挙動を再現するものであり，端末からの入力や状態によって挙動を変えるようなプログラムには対応していない．

3. 実験・考察

新たに追加した機構の動作を確認するために次のような実験を行った．端末は Windows10 Pro 64bit の PC を使用，API コールログは API Monitor v2 (Alpha-r13) [4] を用いて取得した．(1)マルウェアを想定して，マルチスレッドでインターネット上からテキストをダウンロードしファイルを作成，レジストリへの書き込みを行うという実験用プログラムを用意する．(2)実験用プログラムを Windows の端末上で動作させて，API コールログを取得，ファイルに記録する．(3)API コールログを与えて本ツールを動作させ，API コールログを取得，ファイルに記録された API コールログと一致するか確認する．

実験用プログラムの API コールログを図 3 に，本ツールで再現を行った API コールログを図 4 に示す．WaitForSingleObject を再現するため引数の値が変わっている点と，データのポインタやハンドルのアドレス，CreateFileA と CreateFileW のような環境に依存し変化する箇所を除き一致していることが確認できる．また，API Monitor の仕様で，サブスレッドで呼び出された API は背景が黄色で表示されるが，この点においてもマルチスレッドでの処理が，

API	Return Value
CreateThread (NULL, 0, 0x00061000, NULL, 0, NULL)	0x00000190
RegCreateKeyExW (HKEY_CURRENT_USER, "Software\MyRegistry\RegKey", 0, "", 0, KEY_ALL_ACCE...	ERROR_SUCCESS
RegSetValueExW (0x00000198, "Test_Value", 0, REG_DWORD, 0x006ff9a0, 4)	ERROR_SUCCESS
RegSetValueExW (0x00000198, "Test_Data", 0, REG_SZ, 0x006ff9a8, 8)	ERROR_SUCCESS
RegCloseKey (0x00000198)	ERROR_SUCCESS
WaitForSingleObject (0x00000190, INFINITE)	WAIT_OBJECT_0
InternetOpenW ("exp", INTERNET_OPEN_TYPE_PRECONFIG, NULL, NULL, 0)	0x00cc0004
InternetOpenUrlW (0x00cc0004, "http://drive.google.com/uc?id=1Lg-vsJbN2KvpNfwNPyiOvLcU_...	0x00cc000c
InternetReadFile (0x00cc000c, 0x0343fa54, 32, 0x0343fa50)	TRUE
CreateFileW ("mock_malware", GENERIC_ALL, FILE_SHARE_READ | FILE_SHARE_WRITE, NULL, OP...	0x00000550
WriteFile (0x00000550, 0x0343fa54, 32, 0x0343fa4c, NULL)	TRUE
CloseHandle (0x00000550)	TRUE
InternetCloseHandle (0x00cc000c)	TRUE
InternetCloseHandle (0x00cc0004)	TRUE
GetModuleHandleW (NULL)	0x00060000

図 3　実験用プログラムの API コールログ

API	Return Value
CreateThread (NULL, 0, 0x00ff7e90, NULL, 0, NULL)	0x000000b0
RegCreateKeyExA (HKEY_CURRENT_USER, "Software\MyRegistry\RegKey", 0, "", 0, KEY_ALL_ACCESS,	ERROR_SUCCES
RegSetValueExA (0x000001a4, "Test_Value", 0, REG_DWORD, 0x00cffb38, 4)	ERROR_SUCCES
RegSetValueExA (0x000001a4, "Test_Data", 0, REG_SZ, 0x00cffb38, 8)	ERROR_SUCCES
RegCloseKey (0x000001a4)	TRUE
WaitForSingleObject (0x000000b0, 5000)	
InternetOpenA ("exp", INTERNET_OPEN_TYPE_PRECONFIG, NULL, NULL, 0)	0x00cc0004
InternetOpenUrlA (0x00cc0004, "http://drive.google.com/uc?id=1Lg-vsJbN2KvpNfwNPyiOvLcU_-...	0x00cc000c
InternetReadFile (0x00cc000c, 0x01004c28, 32, 0x01004648)	TRUE
CreateFileA ("mock_malware", GENERIC_ALL, FILE_SHARE_READ | FILE_SHARE_WRITE, NULL, OPE...	0x0000055c
WriteFile (0x0000055c, 0x01004c28, 32, 0x01004648, NULL)	TRUE
CloseHandle (0x0000055c)	TRUE
InternetCloseHandle (0x00cc000c)	TRUE
InternetCloseHandle (0x00cc0004)	TRUE
GetModuleHandleW (NULL)	0x00ff6e90

図 4　本ツール動作時の API コールログ

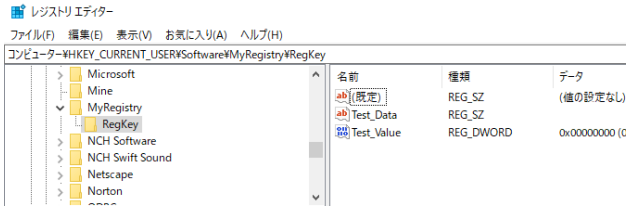

図 5　本ツールで作成されたレジストリの内容

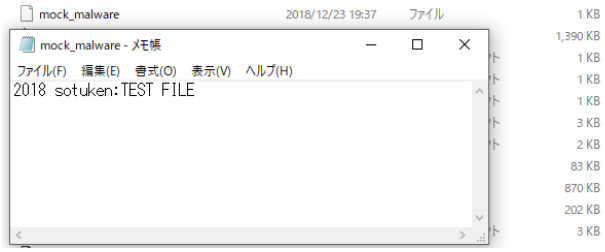

図 6　本ツールで作成されたファイルの内容

時系列順に再現できていることがわかる．

またレジストリへの書き込み動作の結果を Windows 標準ツールであるレジストリエディターを用いて図 5 のように確認した．レジストリに書き込んだ値の内容が，読み込ませたログ上にないため，仮の値である null や 0 となっているが，指定したレジストリへ書き込むという動作は再現できていることが確認できる．さらにファイルの作成が意図したとおり再現されることも図 6 のように確認した．

4. おわりに

マルウェアの動作から得られた API コールログから，ログの時系列順に API を実行し，マルウェアそのものを扱わずにその動作を再現する，マルウェアの動作再現ツールについて検討し，プロセス・スレッド管理，レジストリ操作のカテゴリの API に対応する実装を行った．

実装した機構の動作を確認する実験を行い，ツールに追加した機構が意図したとおり動作すること，新たに対応したプロセス・スレッド管理，レジストリ操作のカテゴリの API の呼び出しが可能であることを確認した．

今後の課題として，実際のマルウェアの API コールログからの動作再現の実験を行い，再現率を評価すること，ツールに再現動作をプレイバックする機能を追加することなどが挙げられる．

参考文献

1) IPA，インシデント対応へのフォレンジック技法の統合に関するガイド，入手先 〈https://www.ipa.go.jp/files/000025351.pdf〉(参照 2018-12-24)．
2) MSDN：Windows API リスト，入手先〈https://msdn.microsoft.com/ja-jp/windows/hh240557〉(参照 2018-12-24)．
3) 港和人ほか：Windows API コールログからのマルウェアの動作再現について，情報処理学会 第 80 回全国大会講演論文集，6W-01，pp. 527-528 (2018)．
4) Rohitab.com：API Monitor，入手先〈http://www.rohitab.com/apimonitor〉(参照 2018-12-24)．