逆アセンブルを用いたマルウェアの解析

戸田晃太・安藤桂汰

研究背景|マルウェアとは？？

マルウェアとは悪意のあるソフトウェアの総称
　→ランサムウェア、トロイの木馬、キーロガー等

企業や政府も被害を受けている
　・JBS(アメリカ最大の食肉企業)
　・イギリスの国民保健サービス

被害は収束していない
　→マルウェアの挙動の理解が望まれる
　→マルウェアの解析を試みた

研究手法|逆アセンブラについて

マルウェアは実行形式で配布されており、私たちは機械語(0と1の羅列)のみしか入手できない

⇩

機械語を人間が読めるアセンブリ言語に変換するツール(リバースエンジニアリングツール)を用いてマルウェアを解析する方法を検討した

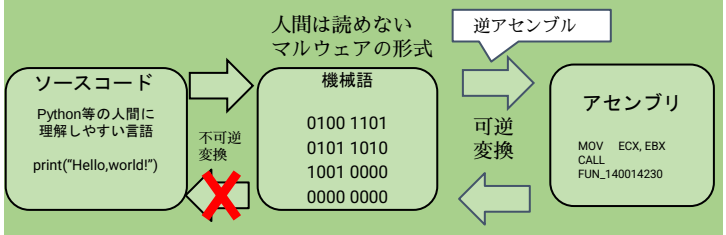

図1　逆アセンブルの流れ

研究手法|自作プログラムの解析

実際のマルウェアは難読化が施されており逆アセンブルでの解析が難しかった

⇩

自作のプログラムを解析することによってソースコードとアセンブリ言語の対応を明らかにすることを試みた

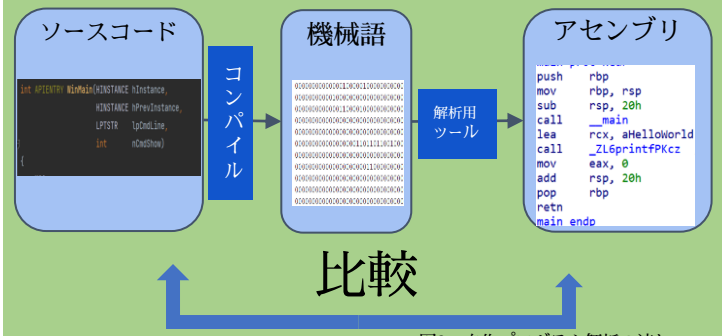

図2　自作プログラム解析の流れ

研究手法|解析環境と検体について

解析用ツールとして、IDAというリバースエンジニアリングツールを用いた。
また、解析対象であるマルウェアはMalwareBazaarやVirusBayなどのマルウェアサンプルの共有サイトから入手した。
解析はすべてVirtualBoxを用いて作成した仮想環境上で隔離された状況で行った。

結果|Hello Worldの解析

1. 関数のプロローグ…ベースポインタ、スタックポインタの位置を変更し、初期化を行っている
2. 引数の定義…MessageBoxAに渡す引数を指定している
3. Windows APIの呼びだし
4. エピローグ…ベースポインタ、スタックポインタの位置を元に戻している

```
push    rbp
mov     rbp, rsp
sub     rsp, 20h
call    __main                  ①
mov     r9d, 0          ; uType
lea     r8, Caption     ; "hello world!"
lea     rdx, Text       ; "title"
mov     ecx, 0          ; hWnd      ②
mov     rax, cs:__imp_MessageBoxA
call    rax ; __imp_MessageBoxA     ③
mov     eax, 0
add     rsp, 20h
pop     rbp                         ④
retn
main endp
```

図3　Hello Worldの逆アセンブル結果

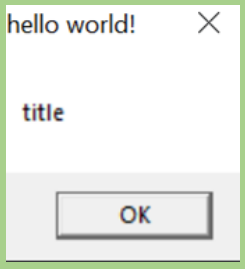

図4 Hello Worldの実行結果

```
int main() {
    MessageBoxA(NULL,"title","hello world!",0);
    return 0;
}
```

図5 C++のソースコード

上の結果から、Hello Worldを出力するだけの単純なソースコードであっても、逆アセンブルをすると冗長で難解なものになるとわかる
また、C++と逆アセンブル結果の間には、次のような違いが見られる

・関数の宣言、終了処理の仕方が異なる
・NULLと0の区別がされていない
・アセンブリ言語ではデータの受け渡しをいちいちレジスタを介して行っている。
・どちらにおいてもWindows APIが使われている

特にWindows APIは様々な動作において使用されていて、逆アセンブリ言語を用いた解析のためにはWindows APIに関する深い理解が必要だと分かった。

結論と今後の展望

結論

・C++のソースコードとアセンブリ言語の対応がわかった。
・WindowsAPIがアセンブリ言語の理解に重要である。

今後の展望

・実際のマルウェアを解析する。(難読化されているため、解読が困難)
・マルウェアの挙動を理解した上で、その被害を減少させる。

参考文献

Ghidraホームページ<https://ghidra-sre.org>
IDA開発会社ホームページ<https://hex-rays.com>
Monnapa K A著,石川朝久訳,『初めてのマルウェア解析』,北原憲・中津留勇監修,オライリー・ジャパン,2020年
矢崎雅之,『ハッカー・プログラミング大全　攻撃編』,第2版,データハウス,2008年
大崎博之,『独習アセンブラ新版』,翔泳社,2021年