7ZC-06

軽量デバッガを用いたマルウェア動作妨害機構の設計と実装

志倉大貴† 西村俊和† 瀧本栄二‡
立命館大学† 広島工業大学‡

1. はじめに

一般的なマルウェア対策として，セキュリティソフトの導入や，当該ソフトの検知精度向上を目的としたマルウェア解析が行われている．しかし，マルウェア対策技術の進展にも関わらず，被害は後を絶たない．このことから，マルウェアの感染を防ぐだけでなく，感染後の被害を抑えるというアプローチも必要と考えられる．

松本ら[1]は，マルウェアの耐解析機能の 1 つであるアンチデバッグを逆用し，感染したマルウェアの動作を妨害する手法を提案している．アンチデバッグはデバッガの存在を検知する機能である．マルウェアはデバッガの存在を検知すると動作を停止してデバッガによる解析を妨害する．当該手法はこの特性を逆用し，デバッガが存在しているように偽装することでマルウェアの正常な動作を妨害する．文献[1]は，デバッガ検知を行うための Application Programming Interface（API）である IsDebuggerPresent API をフックし，常に真を返すことで実現する．この研究の主な目的は，当該手法を用いることによってマルウェアの動作を妨害することが可能であることの実証であり，多様なアンチデバッグの手法のうち，１つについて検証したに過ぎない．したがって，その他の API や API を用いないアンチデバッグは対象外である．

本研究では，Windows 環境において実際にデバッガとして認識されるプロセスを生成してターゲットプロセスにアタッチする機構を設計・実装することで，デバッガ検知に用いられる API だけでなく，API を用いないアンチデバッグに対してもマルウェアの動作妨害が有効な手法を提案する．以下，本稿では，提案手法とシステム構成について述べる．

2. 提案手法

本稿では，マルウェアがシステムに感染した場合における被害防止を目的とした機構を提案する．提案手法は，監視対象であるターゲットプロセスが起動されると，起動イベントを捕捉する．さらに，ターゲットプロセスの実行が開始されるまでに軽量デバッガを起動・アタッチする．これにより，ターゲットプロセスがマルウェアであった場合に，アンチデバッグを行ったタイミングでデバッガを検知して動作を停止させることができる．本章では機構の構成と軽量デバッガ，プロセス監視ドライバ，デバッガ生成プロセスの機能について述べる．

2.1. 構成と動作概要

本機構の構成図を図 1 に示す．本機構は，軽量デバッガ，プロセス監視ドライバおよびデバッガ生成プロセスの 3 つで構成される．本機構の基本動作は，以下の手順のとおりである．

① ターゲットプロセスの実行イメージがメモリにロードされたタイミングで，プロセス監視ドライバがプロセス生成処理をフックする．
② プロセス監視ドライバはデバッガ生成プロセスにターゲットプロセスの Process ID（PID）を通知する．
③ デバッガ生成プロセスは PID を引数として軽量デバッガを生成する．
④ 軽量デバッガは引数の PID を持つプロセスにアタッチし，デバッガ生成プロセスに通知する．
⑤ 通知を受けたデバッガ生成プロセスはプロセス監視ドライバに処理完了を通知する．
⑥ 通知を受けたプロセス監視ドライバは制御をカーネルに返し，ターゲットプロセスは軽量デバッガにアタッチされた状態で実行を開始する．

文献[1]では，提案方式を実現するために 4 つの要件を示している．本研究では，要件 1 と要件 2 は軽量デバッガを用いてターゲットプロセスにアタッチすること，要件 3 はメモリロード時にフックし，アタッチ終了後にフックから復帰すること，要件 4 はプロセス生成時のメモリロードを起点に動作を開始することで対応できる．

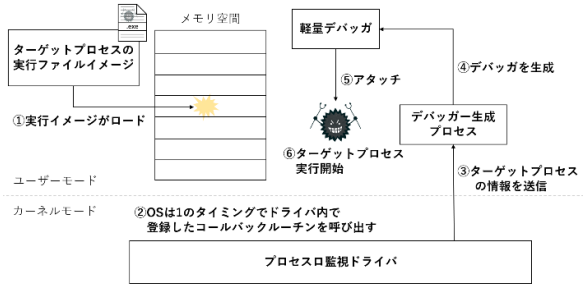

図 1：提案手法の構成図

Design and implementation of the malware deactivator with the light-weight debugger
†Daiki SHIKURA, †Toshikazu NISHIMURA, Ritsumeikan University
‡Eiji TAKIMOTO, Hiroshima Institute of Technology

2.2. 軽量デバッガ

本機構で用いるデバッガは，その存在をマルウェアに検知させることが目的であるため，解析を目的とした機能を必要としない．また，ターゲットプロセスの数だけ生成されるため，必要とする計算機資源を可能な限り削減する必要がある．そこで，提案手法では，デバッガとして動作するための最低限の機能のみを有する軽量デバッガを作成した．

軽量デバッガはデバッグイベントを捕捉するが，即座にターゲットプロセスに制御を戻すことで軽量化している．また，アンチデバッグの手法の１つにある OllyDbg[2]ウインドウの存在を検知する手法を考慮して同名かつ非表示モードのウインドウを生成する．

2.3. プロセス監視ドライバ

プロセス監視ドライバはカーネルモードで動作する．本ドライバの役割は，ターゲットプロセスの起動を検知してその PID をデバッガ生成プロセスに通知することである．起動の検知は，プロセスの実行イメージがメモリ上にロードされたタイミングで行う．これは，実行イメージのロードから実行開始するまでの間に軽量デバッガのアタッチを完了する必要があるためである．そのため，本ドライバは検知からアタッチ完了までプロセス生成処理を遅延させる．

2.4. デバッガ生成プロセス

デバッガ生成プロセスはユーザーモードで動作する常駐型プロセスである．本プロセスはプロセス監視ドライバから PID の通知を受け取ると，その PID を引数として軽量デバッガの生成を行う．生成した軽量デバッガがターゲットプロセスにアタッチされたことを確認すると，プロセス監視ドライバに処理完了を通知する．

3. 評価

本章では，アンチデバッグに対する有効性の検証と機構を適用したときのオーバヘッド計測の結果について述べる．

3.1. アンチデバッグへの有効性

アンチデバッグの分類[3]を基に，Windows API と Checking Manually，Timing Check，Debugger Detection の 4 つの手法について調査した．そのうち，11 種類についてアンチデバッグプログラムを作成して動作検証を行った．結果，10 種類については有効であることを確認できた．動作抑制ができなかった手法は GetTickCount API を用いたアンチデバッグである．これは特定の処理の時間を計測し，閾値より大きいとデバッガ検知とする手法である．既存デバッガはブレークポイントなどによって処理時間が増加する場合がある．しかし，本機構の軽量デバッガはそのような機能がないため当該手法について対応できない．対策として，文献[1]と同様に API フックで値を改竄することで対応可能だと考える．

3.2. 性能評価

提案手法によるシステムへの影響を調査するため，オーバヘッドと CPU 使用率・メモリ使用量への影響に関する性能評価を行った．なお，ここでのテストプログラムの動作内容は printf 関数で文字列を出力するのみとしている．

3.2.1. オーバヘッド計測

提案手法によって生じるオーバヘッドは大きく分けて以下の２点で考えることができる．

(A) プロセスの生成処理をフックし，軽量デバッガのアタッチが完了するまでのオーバヘッド
(B) 軽量デバッガのアタッチが完了してからテストプログラムの実行終了までのオーバヘッド

計測の結果，機構の未適用時における(A)は 0.488ms，(B)は 2.441ms，適用時における(A)は 8.849ms，(B)は 7.679ms となった．(A)の増加原因は，(A)で行われる各処理が逐次実行をするために同期処理を行っていることである．(B)の増加原因は，軽量デバッガが発生したデバッグイベントを受け取るためである．

3.2.2. CPU 使用率とメモリ使用量への影響

計測対象は軽量デバッガと x64dbg[4]とし，テストプログラム動作時の各デバッガにおける CPU 使用率とメモリ使用量を Windows のタスクマネージャーで計測した．結果，軽量デバッガにおいて前者は 0%で一定，後者は 0.7MB で一定であった．x64dbg において前者は 6.0%~7.7%，後者は 32.5MB~32.9MB であった．これらより，軽量デバッガは x64dbg よりも計算機資源の消費量が少ないと考えられる．

4. おわりに

本稿では，アンチデバッグに着目し，感染したマルウェアの動作を妨害することで被害を防ぐ機構の設計と実装について述べ，その評価を行った．今後は，アンチデバッグを行うマルウェアの割合と，使用されているアンチデバッグの種類を調査し，提案手法の有効性を検証する．

参考文献

[1]松木隆宏，新井悠，寺田真敏，土居範久，“マルウェアの耐解析機能を逆用した活動抑止手法の提案，”情処論，**50**，(9)，2118-2126，2009.
[2]“OllyDbg，”https://www.ollydbg.de(参照 2021-12-26).
[3] “Unprotect Project，” http://unprotect.tdg-t.org/index.php/Antidebugging (参照 2021-12-26).
[4]“x64dbg，”https://x64dbg.com(参照 2021-12-26).