MWS Cup 2014

事前課題3「マルウェア動的解析」模範回答

3 マルウェア動的解析（10 点）

3.1 設問1

3.1.1 問題文

マルウェアの動的解析は、マルウェアの挙動を監視するために一定時間マルウェアを実行し続ける必要がある。また、仮想化技術を利用して実現される動的解析は、環境の復元等に一定のオーバーヘッドが必要となる。そのため、動的解析は、シグネチャ判定等の静的な解析手法に比べて時間当たりの検体処理数（スループット）が低い。システムをスケールアップ、スケールアウトする以外にこの課題を解決し、スループットを 100 倍にする方法を述べよ。（2 点）

3.1.2 解答

(1) 挙動監視時間の短縮

昨今のマルウェアは標的の破壊を目的とするものよりは、標的からの情報窃取を目的としたものが多い。これらのマルウェアは情報を窃取するために PC の利用者に気づかれにくいように活動するため、システムファイルの変更など PC の動作に異常を来す恐れがある活動はしないと考えられる。このため、100 検体を同時に 1 台の PC で動作させたとしても互いに影響を与えることなく動作する。各検体の挙動についてはプロセス ID で識別することで切り分けることが可能である。ただし、以下のマルウェアの場合には解析が正しくできない場合がある。

- ファイル消去など破壊を目的としたマルウェア
  システムの動作に必要なファイルが破壊されてしまうと他のマルウェアは動作ができなくなってしまう。
- Rootkit
  システムレベルのマルウェアの複数同時実行は相互に影響を与えてしまう危険性が高い。
- DLL 設置などでバイナリプランティングを行うマルウェア
  複数のマルウェアが同じ名前の DLL を設置した場合、有効となるのは 1 つだけである。
- 既存の実行ファイルを書き換えるタイプのマルウェア
  実行ファイル書き換えで有効になるのは最後に書き換えたマルウェアのみである。

また、マルウェアの中には悪意のある活動を行う前に、長時間の Sleep や時間のかかる演算処理を実行することで自動動的解析のタイムアウトを狙うものがある。これらの処理が行われている場合には、以下のような手法を用いることでその時間を短縮できる。

- Sleep API の引数を 0 にする。
- Sleep 等で待機状態にある場合に解析環境の時刻の進みを 100 倍にする。
- 繰り返し実行されているコードを特定し回避する[3-1]。

[3-1] C. Kolbitsch, et al, “The Power of Procrastination: Detection and Mitigation of Execution-Stalling Malicious Code”, ACM CCS 2011.

(2) 環境の復元時間の短縮

マルウェアの動的解析においては、しばしば挙動の監視よりも環境の復元に時間がかかる場合がある。この復元を行わないようにすることで解析速度の向上が見込める。具体的には、Sandboxie[3-2]や Deep Freeze[3-3]等で使われている技術を活用して解析中に発生したディスクやレジストリへの変更を破棄し元の状態に簡単に復旧できるようにすることで復旧にかかる時間をなくすことができる。Sandboxie のようなサンドボックス環境は、(1)で挙げた複数マルウェアの 1 端末での同時解析時にも活用できる。

[3-2] Sandboxie, http://www.sandboxie.com/
[3-3] Faronics, Deep Freeze, http://www.faronics.com/ja/products/deep-freeze/

(3) その他

その他、以下のような手法により解析を高速化できる。ただし、それぞれ単体では 100 倍の高速化は達成できないが、(1)(2)で挙げた手法との組み合わせにより 100 倍高速化は不可能ではないと考える。

- VM のコンパクト化
  ディスク容量を極限まで小さくするなどの工夫により復旧にかかる時間を短縮する。
- VM のオンメモリ化
  仮想環境をディスク含めて全てメモリ上に載せることで高速な処理を実現する。
- マルウェア動作に必要な機能のみのエミュレーション
  解析のために Windows を搭載した VM を用意する手法が一般的だが、必ずしも Windows の全ての機能がマルウェア動作に必要ではないと考える。マルウェア動作に必要な機能のみをソフトウェアでエミュレーションすることで解析を高速化する。

3.2 設問2

3.2.1 問題文

マルウェアの中には解析対策として、仮想マシン検知、解析手法・ツール対策等を実装しているものが存在する。これら解析対策を備えたマルウェアを適切に解析する方法を述べよ。尚、適切な解析とは、解析対策技術の発見・識別、解析対策技術の影響を受けた場合、受けなかった場合双方の解析を意味する。(2 点)

3.2.2 解答

解析対策技術を持つマルウェアは，解析環境を検出できた場合と検出できなかった場合で動作が異なる．したがって，仮想マシン環境，実マシン環境，解析ツールを多数インストールした環境など

の複数の解析環境を用意し，それぞれで動的解析を行い，結果を比較することによって解析対策技術の存在および検出された場合とされなかった場合の挙動を得ることが可能となる．また，Moser ら[5]のように，分岐命令実行時のスナップショットと分岐に使用された条件の書き換えによって，複数の実行パスを強制的に実行することで，検出された場合とされなかった場合の動作を観測する手法も考えられる．

解析対策技術の発見・識別方法については，既知手法については，シグネチャによるマッチングや使用される API などの監視によって検出が可能である．一方，未知の手法については，解析環境を検出した場合のマルウェアは動作をすぐに停止することが多いことを考えると，実行停止前に実行された分岐命令を逆昇ることで，検出ルーチンを発見できると考えられる．なお，分岐記録は BTS やエミュレータなどを活用することで，取得が可能である．

解析対策を備えたマルウェアを、その影響を受けずに解析するには、解析対策の 1 つ 1 つを回避する対策を施す事が必要である。 以下に、マルウェアで用いられるような解析対策の一例と、それに対する回避策を述べる。

- 仮想マシン対策
  - VMM のバックドア I/O ポートへのアクセスを試みる
    - VMM の設定で I/O ポートを閉じる
  - VM 特有のドライバファイルや補助ツール(例: VMWare Tools, VirtualBox Guest Addition 等)の存在を確認する
    - そのようなものをインストールしない
  - Windows において二次記憶ディスクのシリアル ID をレジストリ等から読み取って VM 特有の文字列があるか確認する
    - シリアル ID を変更する
    - シリアル ID の読み取りを行うようなコードの周辺を確認して、解析対策があればパッチなどで無効化する
  - (仮想マシン対策全般)
    - (コストが許す場合)実マシン上で解析する
- 解析手法・ツール対策
  - プロセスを列挙して解析ツールの実行ファイル名のプロセスが動作していないか確認する
    - 解析ツールの実行ファイル名を書き換えておく
  - ウィンドウを列挙して解析ツールのウィンドウ名のウィンドウが無いか確認する
    - ウィンドウを列挙するような API をフックして偽の値を返す
  - Windows において IsDebuggerPresent 等でデバッガの存在を確認する
    - メモリ上のフラグを書き換えたり、API をフックして偽の値を返す
  - Windows において解析に使われるツール等のモジュール(SbieDll.dll, dbghelp.dll 等)がプロセス内にロードされているか確認する
    - そのようなモジュールをロードしない範囲で解析する
    - GetModuleHandle でモジュールの存在を確認している場合、当該 API をフックして偽の値を返す
  - プログラム中の一部分の経過時間を測定し、経過時間が一定より長ければ解析が行われ

ていると判断する

- ✧ 時刻や経過時間を取得する API や syscall 等をフックして偽の値を返す
- ✧ 時刻や経過時間を取得する API や syscall 等を呼び出すコードの周辺を確認し、解析対策があればパッチなどで無効化する

解析対策の影響を受けた場合の解析を行うには、上記などの解析対策に検知されるような行動をあえて行う事で、解析対策の影響を受けた場合の挙動を解析する事が出来る。

3.3 設問3

3.3.1 問題文

マルウェアは、C&C サーバからのコマンド受信等、特定の動作・条件をトリガーとして悪意の動作を行うものが存在する。こうしたマルウェアについて直接的ないしは間接的に悪意の動作を実行・再現させ、悪意の動作に関する動的解析結果を取得する方法を述べよ。(3 点)

3.3.2 解答

(1) 実環境の模擬

マルウェア解析の環境をインターネットに接続し、マルウェアを動作させると同時にユーザにその環境を使った作業(ネット閲覧、メール閲覧など)をしてもらい、挙動観測を実施する。マルウェアが想定する感染先の環境と同様の状況を作り出すことでマルウェアは動作すると考える。

(2) コマンドやトリガの推測

近年のマルウェアは完全にオリジナルなものは数少なく、多くが既知のマルウェアの亜種であることが知られている。亜種間では C&C サーバのコマンド体系や動作のトリガも似た形になることが推測される。このため、マルウェアの動的解析の前にそれが既知のマルウェアの亜種であるかどうかを確認し、亜種であった場合には既知のマルウェアのコマンドや動作トリガを調べ、そのように振舞う偽 C&C サーバや解析環境を用意して動的解析を行うことで悪意の動作を再現できる。

(3) C&C サーバに接続してコマンド解析

マルウェアの動的解析により接続先の C&C サーバの情報を取得することができる。C&C サーバと直接通信を行い、使用プロトコルおよびコマンドを解析する。ここで得られた結果を用いて偽 C&C サーバを用意することでマルウェアの悪意の動作を再現する。この手法の場合、通信データからプロトコルやコマンドが解析できることが条件となる。暗号技術が使われている場合などには対応しきれない。

(4) コマンド受信後やトリガとなる動作の実施後の処理に強制移動

事前にマルウェアのコードを簡単に調べておく。コマンドに対応する動作などはそれぞれ関数で記述されていると想定されることから、call 命令のジャンプ先を記憶しておき、マルウェアの動的解析時

にそれらのジャンプ先に強制的に EIP をセットする。これによりコマンド受信やトリガとなる動作の実施などがなくとも悪意の動作を実行させられると考える。

(5) 静的解析結果のフィードバック

対象マルウェアの静的解析を行い、C&C サーバとの通信プロトコルやコマンド、動作のトリガなどを明らかにした上で、当該プロトコルやコマンドを実装した偽 C&C サーバ、トリガとなる動作を実施する環境を用意することでマルウェアによる悪意の動作を実現できる。

動作のトリガを明らかにすることでマルウェアが動作する環境を用意でき、また C&C サーバとの通信プロトコルやコマンドを明らかにすることで通信の監視時にどのようなコマンドを受信したかが監視できる。これにより、インターネットに接続した環境でマルウェアを動作させ、実際の攻撃者からのコマンドを受けることで攻撃者の意図の把握が可能になり、同時に外部への攻撃実施など危険な活動を行おうとした際には通信切断などの対処を取ることにより水際で止めることができる。

3.4 設問4

3.4.1 問題文

現行の FFRI Dataset 2014 について追加・変更すべき情報、及びその活用に関する方法について述べよ。Cuckoo Sandbox 及び FFR yarai analyzer Professional 双方の動的解析ログを対象とする。参考までに昨年度の活用事例を下記に示す。(3 点)

■MWS 2013 における活用事例の論文タイトル

・サンドボックス解析結果に基づく URL ブラックリスト生成についての一検討
・自動化されたマルウェア動的解析システムで収集した 大量 API コールログの分析
・類似度に基づいた評価データの選別によるマルウェア検知精度の向上
・Kullback-Leibler 情報量を用いた亜種マルウェアの同定

3.4.2 解答

(1) 複数の解析エンジンの併用による解析精度向上

現行の FFRI Dataset 2014 では、Cuckoo Sandbox、FFR yarai analyzer Professional の解析ログが含まれている。マルウェアの動的解析を行う解析エンジンは研究レベル、製品レベルで数多く存在し、それぞれ解析手法、出力データが異なっている。一つのマルウェアを複数の解析エンジンで解析することにより、様々な視点からの解析結果を取得でき、それらを統合することでマルウェアの解析精度を向上できると考える。このため、FFRI Dataset で別の解析エンジンを用いた解析データが追加されると良いと考える。

(2) 複数種類の解析環境の利用

Cuckoo Sandbox は仮想環境として VirtualBox や KVM を用いることができる(Dataset でどの仮想環境を用いているか不明)。Yarai Analyzer Professional は VMware を仮想環境として用いる。両者の解析結果を比較することにより、仮想環境に応じたマルウェアの挙動の違いを明らかにできる(設問 2 とも関連)。このため、Xen なども含めたより多種の仮想環境で解析した結果が Dataset

に含まれると良いと考える。さらに、仮想環境ではなく物理マシン上でのマルウェア挙動の観測結果があると良い。

(3) 情報共有フォーマットの検討

マルウェアへの対策として、マルウェアに関する情報の共有がある。マルウェアの挙動を示すフォーマットとしてMAECがあるが、マルウェアの挙動について共有するのにMAECで十分か、解析エンジンの出力で足りない情報はないか、検討が十分にはされていないと考える。FFRI Dataset の解析ログを元に情報共有フォーマットの検討ができる。

(4) 開発した解析システムの評価

FFRI Dataset に含まれる検体と同じものを解析した結果と Dataset を比較して開発システムの評価ができる。比較のために Sandbox の環境（仮想環境、OS、アプリ、設定など）が情報として追加されていることが望ましい。

(5) fuzzy hashing 値の追加

fuzzy hashing とは，類似的なデータである場合，ハッシュ値が近い値となる技術である．Cuckoo Sandbox のログには，ssdeep を使用し，ハッシュ値を求めている．しかし，ハッシュ値を算出できていないログが多い(null となっている)．そこで，すべてのマルウェアに fuzzy hashing 値を追加することで，類似しているマルウェアの分類に利用できると考える．文献[11] では，fuzzy hashing で 90%以上で類似したマルウェア同士は，検体名が 40%～80%で類似していた．そこで，マルウェアの分類の制度を向上させるため，fuzzy hashing で分類した後，API コールシーケンスで分類することで，より精度の高い分類ができると考える．API コールシーケンスでの分類は，機械学習などの手法を用いる研究が多く存在し，これらを利用する．これにより，精密なマルウェアの分類ができ，優先して解析すべきマルウェアを特定やマルウェア検知に有益な分類ができると考える．

(6) 脆弱性利用箇所の詳細な情報の追加

マルウェアが脆弱性を利用した箇所についての詳細な情報を追加することで，利用者を支援できると考える．例えば，ある DLL 内のコードの脆弱性を利用した場合，その DLL 名，脆弱性攻撃の発生箇所，コールスタックなどを追加することで，利用した脆弱性を特定できる．これにより，利用した脆弱性の傾向などを把握できる．

(7) どの程度マルウェアを解析できたかのパラメータの追加

解析したマルウェアについて，全体の実行コードの何割解析できているかのパラメータを追加することで，マルウェアの全体像や解析を優先すべきマルウェアの提示を支援できると考える．まず，マルウェアの実行コード領域のサイズを抽出する．その後，マルウェアを実行し，実行できたコード領域を算出することで，マルウェアがどの程度解析できたのか分かる．ただし，パックされているマルウェアの場合，正しい実行コード領域を取得できない可能性があるため，動的にアンパックする手法などを利用することを検討したい．