通信可視化と動的解析の連携による攻撃解析支援

義則 隆之† 伴 拓也† 宮嵜 仁志† 松井 拓也† 佐藤 両† 岡崎 亮介† 篠田 昭人†
廣友 雅徳‡ 毛利 公美†† 神薗 雅紀‡‡ 白石 善明†

†名古屋工業大学 ‡佐賀大学 ††岐阜大学 ‡‡株式会社セキュアブレイン

あらまし プラグインやブラウザの脆弱性を悪用しマルウェア配付サイトに誘導する Drive-by-Download（DBD）攻撃による被害が増加している．DBD 攻撃への対策の一つとして，クライアントハニーポットを用いて Web サイトをクローリングし収集した通信データからコードを抽出して解析し，悪性サイトを検知するアプローチがある．しかしながら，膨大な観測データの中から疑わしいフローを見つけ，さらにそれに含まれるコードを一つずつ手動で解析することは容易ではない．本稿では通信データから復元した解析対象の選択を支援する通信データ可視化，および解析対象の動的解析を統合した DBD 攻撃解析支援システムを提案する．提案システムのプロトタイプにより，D3M 2012 (Drive-by-Download Data by Marionette 2012) データセットを用いて，提案システムの基本的な機能が動作することを確認した．

A Malware Analysis Support System Combining Visualization and Dynamic Analysis

Takayuki Yoshinori† Takuya Ban† Hitoshi Miyazaki† Takuya Matsui†
Ryo Sato† Ryosuke Okazaki† Akihito Shinoda† Masanori Hirotomo‡
Masami Mohri†† Masaki Kamizono‡‡ Yoshiaki Shiraishi†

† Nagoya Institute of Technology ‡ Saga University
†† Gifu University ‡‡ SecureBrain Corp.

Abstract Recently, the Drive-by-Download (DBD) attack increases threats which make users lead to the web sites distributing malwares. As countermeasure for this attack, we detect malicious web sites from the traffic data collected by crawling the web sites by client honeypots. However, it is infeasible to analyze each malicious code in huge traffic data. In this paper, we propose a system for supporting the analysis of the DBD attacks. The proposed system consists of two novel subsystems. The first subsystem is an interface which visualizes the traffic to focus on the attack flow. The second one is an analysis environment which automates dynamic analysis of the focused malicious codes.

1 はじめに

プラグインおよびブラウザの脆弱性の悪用や，Gumblar などで代表される Web 改ざんにより，マルウェア配付サイトに誘導する Drive-by-Download（DBD）攻撃による被害が拡大している．DBD 攻撃はリダイレクトにより多段に Web サイトを経由してマルウェア配布サイトまで誘導する攻撃フローが特徴である．標的型攻撃の一攻撃手段ともなっており，DBD 攻撃の対策方法の確立が望まれている．DBD 攻撃の被害を受けないために，通信セッション中に攻撃を検知し，防御する研究がなされている．しかしながら，新しい攻撃コードが頻繁に出現する状況においては，ユーザサイドですべての攻撃コードを漏れなくリアルタイムに検知･防御することは容易ではない．

組織として攻撃の防御や被害の抑制を目指すならば，リアルタイム検知・防御だけではなく，クライアントハニーポットを用いて能動的に Web サイトをクローリングすることで悪性サイトを検知するというアプローチ[1]や，ユーザの通信データを保管し，事後にそのデータを分析することで悪性サイトへのアクセスを検知し，対策を講じるというアプローチも疎かにはできない．

クローリングしたハニーポットの通信データあるいはユーザが通信したデータの中から悪性サイトを探し出すには，リダイレクトフローに着目し（図 1 の A），DBD 攻撃で最終的にダウンロードされ実行される検体（図 1 の B）を解析するという作業が行われる．前者の解析対象とする検体の選択，後者の検体を解析してマルウェアと判定する方法には例えば次のような研究がある．

通信データの分析における解析対象コードに至る攻撃フローの絞込みでは，サーバ型のハニーポットが収集したログに基づいた攻撃者の地理的分布を，WHOIS データベースと GeoPlot ソフトウェアを利用して世界地図に可視化する手法[2]などがある．DBD 攻撃に関係するものには，Gumblar に感染した端末を設置し，全ての通信データの送信元アドレスを Google Earth 上にマッピングしサーバ上のファイル操作に着目し可視化を行う手法[3]がある．

マルウェアを判定する方法は一般に静的解析と動的解析に大別される．静的解析はコードを見てマルウェアの挙動を把握する方法である．動的解析は，解析環境内でコードを実行し挙動を監視することで解析を行う方法である．動的解析は，静的解析と比較すると短時間での解析が可能で難読化されたコードにも対応が可能である．例えば，文献[4]のマルウェアが外部と通信する挙動に着目し，仮想的なネットワークを構築した実環境内でマルウェアを動作させ詳細な通信挙動を解析する手法などがあり，文献[5]では動的解析に関係するツールなどがまとめられている．DBD 攻撃に関係するものには，JavaScript の難読化されたコードを動的解析する手法[6]がある．

以上のような通信データからの解析対象コードの選択とマルウェアの判定は基本的には独立してなされている．事後に通信データを分析して悪性サイトへのアクセスを検知し組織として対策を講じるというアプローチをとるにあたっては，これらの手法を一連のシステムとして結合することができれば，被害に遭った場合でも次の対策に速やかに移行できると期待される．

そこで，本稿では解析対象コードの選択を支援する通信挙動の可視化と，解析対象コードの動的解析を統合した一連の処理を自動化する DBD 攻撃解析支援システムを提案する．

2　提案システムのコンセプト

通信データを分析し，DBD 攻撃により悪性サイトに誘導された事実を特定する流れは次のようになる．まず，通信データからダウンロードしたファイルをすべて復元し，そして復元したファイルに危険なコードが含まれていないかを調べる．

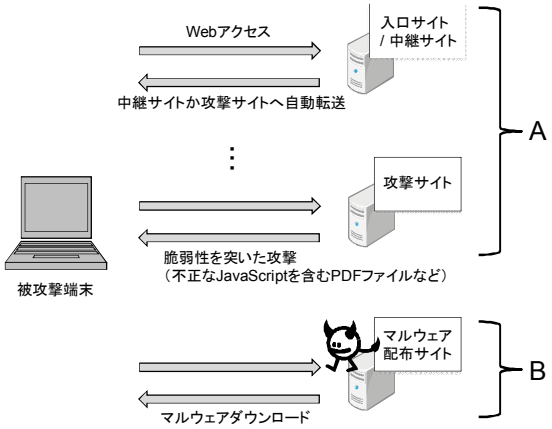

図 1　Drive-by-download 攻撃のフロー

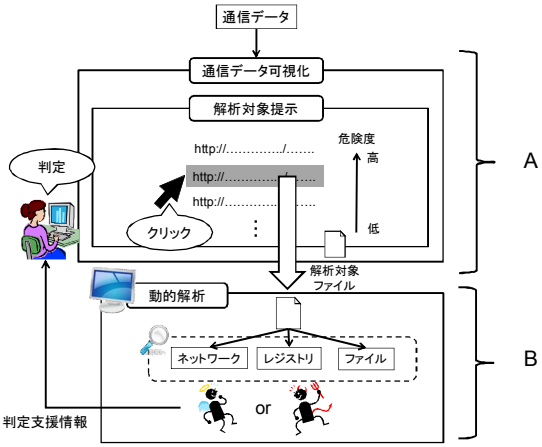

図 2　提案システムのコンセプト

通信データのサイズが大きい場合は，ファイルが多く含まれているということになる．すべてのファイルに対して解析することは現実的ではないので，解析対象のある程度の絞込みができればユーザ（分析者）の支援になる．通信フローの可視化，復元したファイルの危険度などをユーザに提示し，ユーザが解析の優先度が高いと思う順に動的解析環境に投入するのがよいと考えられる．

そこで，図 2 で示すような，通信フローを可視化することでユーザに解析対象ファイルの選択を支援し，選択した解析対象ファイルに対して自動的に動的解析が開始され，その動的解析結果を判定支援情報としてユーザに提示するという一連の流れを途切れることなく行うことを提案システムのコンセプトとする．DBD 攻撃により悪性サイトに誘導された事実を従来の技術や既存の手法を用いて特定しようとすると，可視化に関する研究と動的解析に関する研究のそれぞれの成果を組み合わせる点は手動になる．そのような組み合わせの部分が一連の作業として結合されることが提案システムの特徴である．

3　DBD 攻撃解析支援システム

3.1　システム構成

提案システムの構成は図 3 のようになる．提案システムは 3 つの主要な構成要素から成る．

3.1.1　解析対象コード選択支援部

通信データから解析対象ファイルを選択するための攻撃フローの可視化をし，選択したフローに含まれる検体をマルウェア判定支援部に転送するサブシステムである．解析対象コード部は復元・分類部，分析部，フロー可視化部，ファイル転送部のモジュールからなる．

[復元･分類部] 通信データから復元可能な全てのファイルを復元する．分析部から受け取ったリダイレクト先 URL をリダイレクト元の URL と対応付ける．

[分析部] 復元されたファイルがリダイレクトを行うコードを含んでいないか調べ．含んでいればリダイレクト先 URL を復元･分類部に渡す．

[フロー可視化部] 復元・分類部から整理された URL を受け取り可視化する．

[ファイル転送部] UI 管理部からユーザの指定した動的解析対象ファイルの名前を受け取り，該当するファイルをマルウェア判定支援部に転送する．

3.1.2　マルウェア判定支援部

解析対象コード選択支援部で得られた攻撃候補を動的解析するサブシステムである．ユーザによるマルウェア判定を支援できるように解析結果を可視化する．マルウェア判定支援部はファイル受信部，動的解析用端末，挙動可視化部のモジュールからなる．

[ファイル受信部] ファイル転送部からファイルを受信する．

[動的解析用端末] ファイル受信部でストレージに格納された当該ファイルを実行する．挙動監視部を内包する．

[挙動監視部] API 呼び出し，レジストリ操作，ファイル操作，ネットワーク挙動を監視する．

[挙動可視化部] 動的解析部が生成した解析結果を可視化する．

3.1.3　システム管理部

ユーザの入出力と解析対象コード選択支援部とマルウェア判定支援部の管理をするサブシステムである．UI 管理部と動的解析管理部のモジュールからなる．

[UI 管理部] ユーザの入力を受け付ける．可視化部で提示された動的解析対象の通信フローをユーザが選択したらファイル転送部へ処理を渡す．

[動的解析管理部] 設定ファイルを読み込む．動的解析端末のソフトウェアやパラメータの設定する．

3.2　システムの動作

提案システムの動作の流れを以下に示す．

1. UI 管理部が通信データの入力を受け付け，通信データを復元・分類部に渡す
2. 復元・分類部は通信データに含まれるすべてのファイルを復元しストレージに格納する
3. 分析部はストレージ内に存在する復元されたファイルを分析し，リダイレクト URL を復元・分類部に返す
4. 復元・分類部は分析部から受け取った結果を基に通信データに含まれる URL を整理してフロー可視化部に渡す
5. フロー可視化部は整理された URL と IP アドレスから通信フローを可視化インタフェースに描画する
6. UI 管理部がユーザの入力から動的解析対象のファイル名を取得し，ファイル名をファイル転送部に送る
7. ファイル転送部は受け取ったファイル名と対応するファイルをストレージから取り出し，マルウェア判定支援部に転送する
8. ファイル受信部はファイルを受信し，動的解析端末に渡す
9. 動的解析管理部は設定ファイルを読み込み，動的解析端末に設定情報を渡す
10. 動的解析端末は動的解析対象のファイルを実行する．挙動監視部が当該ファイルの挙動を監視する
11. 挙動監視部が出力した結果を挙動可視化部に渡す
12. 挙動可視化部は受け取った結果を読み込み，可視化する

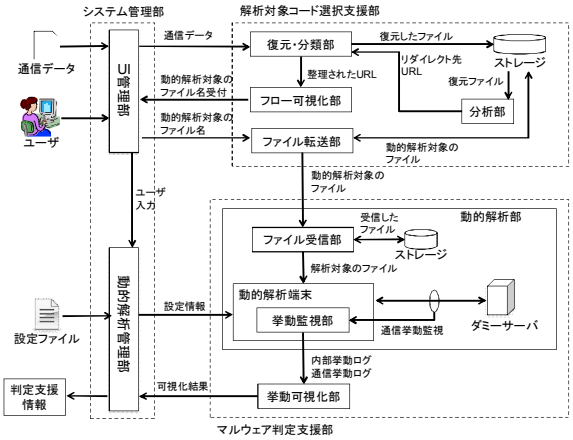

図 3　提案システムの構成

4　評価用システムの試作

以下では，通信データに含まれている解析対象候補のユーザへの提示から解析対象ファイルの選択，そして当該ファイルの自動実行と動的解析結果のユーザへの提示までの一連の流れが進むことを確認するための評価に用いるシステムを説明する．

提案システムでは解析支援の精度を向上させるために，種々のアルゴリズムや手法をモジュールとして用いることを想定している．4.2 および 4.3 で示す解析対象コード選択支援部とマルウェア判定支援部はいずれもサブシステムの一つの実現方

法であり，他のモジュールとの入れ替えや併用を考えている．

4.1 評価用システムの構成

解析対象の評価用システムの構成を図 4 に示す．

[**復元・分類部**] 通信データのファイル（Pcap ファイル）からファイルを復元する機能は WinPcap[7]の Java のラッパーである jNetPcap 1.3.b4[8]を利用して実装した．

[**分析部**] リダイレクトの最後のフローを検知することを考え，アンチマルウェアソフトの ESET NOD32 Antivirus 49]を用いる．

[**フロー可視化部**] Java の外部ライブラリの Java 3D 1.5.1 を用いて実装した．

[**ファイル転送部**] Java 1.7.0_05 で実装した．

[**ファイル受信部**] Java 1.7.0_05 で実装した．

[**動的解析端末**] Windows Virtual PC で，ゲスト OS は Windows 7 Professional 64bit を用いて構築した．

[**挙動監視部**] ファイル，レジストリ操作の監視に ProcessMonitor[10]を用いた．API 呼び出し監視には Immunity Debugger[11]用に開発されたプラグインである BlackManta[12]を利用した．

[**挙動可視化部**] 挙動監視部から解析結果を受け取り挙動可視化の結果を作成する．Java 1.7.0_05 で実装した．

4.2 解析対象コード選択支援部の実装

解析対象コードの選択支援のために用いる，通信データ（Pcap ファイル）に含まれる通信挙動を可視化するサブシステムを評価用に実装した．

4.2.1 解析対象コードの選択を支援する可視化の流れ

マルウェア候補抽出の流れは以下のようになる．

1. Pcap ファイルを入力し，jNetPcap を利用して，パケットの中に含まれるすべてのファイルを復元し，ストレージに保存する
2. NOD32 のリアルタイムファイルシステム保護により，マルウェアを含むファイルの隔離が行われログが生成される
3. XML 出力したログを URL Classifier に入力し，Pcap ファイルに含まれるパケットのうちマルウェアの候補となるものを抽出する
4. Flow Visualizer に抽出したパケットを入力し，それを元に通信フローを可視化する

以下，4.2.2 節で Restorer，4.2.3 節で URL Classifier，4.2.4 節で Flow Visualizer の動作を説明する．

4.2.2 Restorer

Restorer では，jNetPcap を用いて以下のような手順で Pcap ファイルからすべてのファイルを復元した．

1. Pcap ファイル中のすべてのパケットを配列 P_{All} に読み込む
2. 読み込んだパケットの中から HTTP ヘッダの GET メソッドを含むパケットのみを取り出し配列 P_{GET} に格納する
3. P_{GET} からパケット P を 1 つずつ取り出し，3-1 から 3-3 を繰り返す
   - 3-1 P_{All} の中から P を検索する
   - 3-2 P_{All} の中から P 以降にある P の GET メソッドで得られた Response のパケットのデータを検索し，該当パケットを配列 P_{Res} に格納する
   - 3-3 P_{Res} のパケットをシーケンス番号順に並べ，データ部分を結合し，ファイルを出力する

この試作では，2 で作成した P_{GET} をいくつかに分割し異なるプロセスで 3 の処理を行うことでファイル復元の並列処理を行い，高速化を図っている．

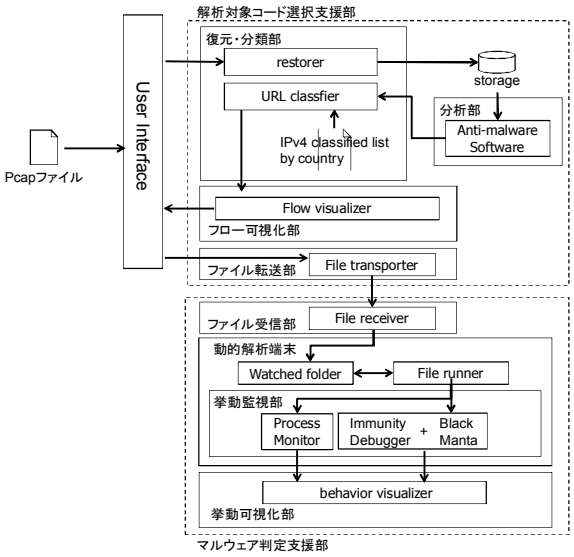

図 4 評価用システムの構成

4.2.3 URL Classifier

URL Classifier は，NOD32 に隔離されるファイルを持つ IP で，かつ，ハニーポットからのアクセス頻度が高い IP を含むパケットの集合を出力する．

以下のような手順で機能を実現している．

1. Pcap ファイルから通信フローの先頭パケットを取り出した集合を G_{URL} とする
2. Pcap ファイルからすべてのファイルを復元する．NOD32 のリアルタイムファイルシステム保護で隔離されたファイルの元の通信フローの先頭パケットの集合を G_{AV} とする
3. G_{URL} に含まれるパケットのヘッダから Destination IP を重複なく取り出し，リスト L_{IP} に格納する
4. L_{IP} から要素 D_{IP} を一つずつ取り出し，4-1,4-2 を繰り返す
   - 4-1 G_{URL} のうち Destination IP が DIP と一致するパケットをグループ化し，時刻順に並べたものを G_{IP_URL} とする
   - 4-2 G_{IP_URL} と G_{AV} の共通部分を新たに G_{IP_URL} とする
   - 4-3 すべての G_{IP_URL} を要素数の順に並べたときの中央にくる G_{IP_URL} の要素数を c（中央値）とする
5. 要素数が c 以上の G_{IP_URL} を要素とする集合を

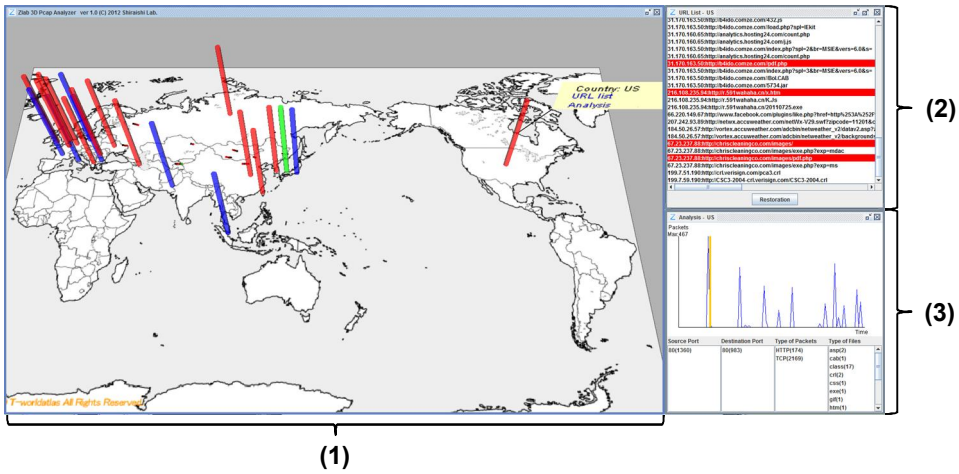

図 5　3 つの可視化ウィンドウ

G_{RET} とし，G_{RET} を出力する

4.2.4　Flow Visualizer

URL classifier が出力した情報を可視化するビューワを試作した．ビューワの表示画面は図 5 のようになる．

図 5 (1)は，文献[13]を参考に試作して設計している．3D で世界地図を表示し，ハニーポットの位置とハニーポットと通信した国にポールを立て，ポール間にパケットを表す線分がアニメーションで行き来することで，ハニーポットと当該国間の通信フローを表現している．ポールの色は，ハニーポットの位置のポールを緑, NOD32 に検出された URL を含み，かつ，ハニーポットからのアクセス頻度が多い国のポールを赤，それ以外の国のポールを青にした. 3D のカメラの位置はマウスで操作できる．左ボタンのドラッグでカメラの焦点を中心に回転，右ボタンのドラッグで平行移動，マウスホイールの回転で拡大，縮小ができる．

ポールをクリックすると吹き出しが表示され，その国のトップレベルドメインと URL List のリンク，Analysis のリンクが表示される．URL List のリンクをクリックするとその国と対応し(2)の画面を表示し，Analysis のリンクをクリックするとその国に対応した(3)の画面を表示する．

図 5 (2)は，ある国の IP とそれに対応した URL のリストである．ハニーポットがどのような IP, URL にアクセスしているのかを国ごとに表示する．そのうちの分析部で検出された URL を赤色に反転させ強調した．URL を選択し，Restoration ボタンを押すと該当するファイルをファイル転送部に渡す．既にファイルを復元した URL が区別できるように，復元済みの URL の背景色は灰色にしている．

図 5 (3)は，文献[14]を参考に，ある国の通信フローの特徴を表した．(3)のウィンドウを拡大したものを図 6 に示す．グラフは，単位時間あたりのパケットの総量を示し，どのタイミングでどれだけのパケットをやり取りしているかを表示した．グラフの時間軸上を動く黄色の棒は，図 5 (1)上に

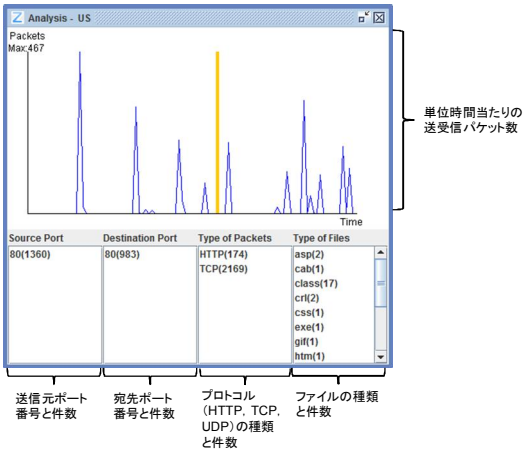

図 6　通信フローの特徴の可視化（図 5 の(3)の拡大）

アニメーションされるパケットと連動する．Source Port, Destination Port はそれぞれある国のホストが利用した送信元ポート番号，宛先ポート番号をリストで示し，括弧内の数字でその件数を示した．Type of Packet は HTTP，TCP，UDP のどれが使われているか，括弧内の数字はその発生件数を表している．Type of Files はある国のホストがハニーポットに送ったファイルの拡張子をリスト表示し，括弧内の数字でその件数を示している．

4.3　マルウェア判定支援部の実装

デバッガによる解析や仮想マシンでの実行を検知すると動作を停止するマルウェアが存在し，それを応用したマルウェアの活動抑止手法[15]がある．また，実行毎に挙動を変動させるプログラムはマルウェアのみが有する機能であると仮定し，検査対象の実行ファイルを複数回実行して得られた API ログを比較することで実行毎の挙動の差異を判断しマルウェアの検知を行う手法[16]がある．

以上のように，複数の条件下でアンチデバッギ

ング機能を利用してマルウェアと判定する動的解析手法は比較的容易に実現できると考え，動的解析端末を次のように構築した．

4.3.1 マルウェア判定を支援する動的解析の流れ

次のように動的解析を行う．

1. Watched folder に攻撃候補の検体ファイルが格納される
2. File runner が Watched folder 内の検体を実行する
3. ProcessMonitor と，ImmunityDebugger および BlackManta が検体の挙動を解析してそれぞれが結果を出力する
4. Behavior Visualizer は出力結果を基にマルウェアの挙動を可視化する

4.3.2 動的解析端末の処理の流れ

動的解析端末の処理の流れは以下のようになる．

1. 解析対象コード選択支援部から抽出されたマルウェア候補が監視フォルダに投入する
2. 一定時間毎に監視フォルダ内のマルウェア候補の有無を確認し，ファイルが存在すればそれを挙動監視部に渡す

4.3.3 挙動監視部

挙動監視部は以下のような手順で動的解析を行う．

1. ProcessMonitor，Immunity Debugger を起動する
2. Immunity Debugger で BlackManta を起動し監視対象の API を設定する
3. ProcessMonitor と BlackManta が動作している状態でマルウェア候補を実行する
4. マルウェア候補が動作を完了する，完了しない場合は一定時間経過したら ProcessMonitor, BlackManta を停止しログを生成する

BlackManta はレジストリ操作の API を監視しファイル操作の API は監視しないなどの API の役割ごとに監視可否を設定できる．BlackManta は Python で実装されている．プラグインのコードを編集することで監視する API の追加削除などの任意の機能を追加できる．本稿では評価システムを実装するに当たり，解析結果をファイル出力するコードを追加した．

4.3.4 挙動可視化部

4.3.3 の挙動監視部から出力結果から図 7 のように動的解析結果を可視化する Behavior Visualizer を試作した．2 つの円を描画し，それらをファイル操作（File），レジストリ操作（Registry），ネットワークアクセス（Network）という 3 つの描画ブロックに分割して操作対象を表現する．

内周上には解析対象のファイルが呼び出した API 名を表示する．File にはファイル操作の API，Registry にはレジストリ操作の API，Network にはネットワークアクセスの API をマッピングし，File，Registry，Network の外周上の各ブロックも API 毎に均等に分割し，どこに何を置くかをあらかじめ決めておく．検体によっては BlackManta で捕捉

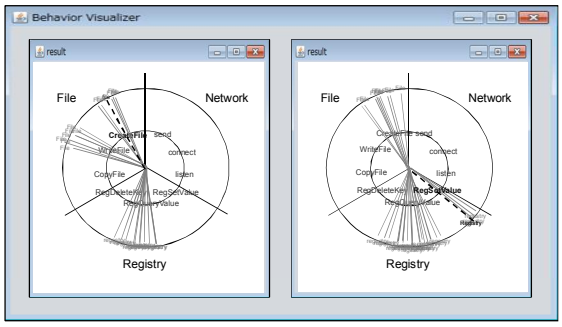

図 7　Behavior Visualizer の可視化画面

した API をすべてマッピングすると膨大な数になるため，API を危険性の高さでソートし上位に該当するものを優先的にマッピングした．各グループに 3 つの API をマッピングする．

外周上には解析対象のファイルが操作したファイルパスを表示する．Registry と Network も同様の処理となる．Network は外周上にアクセス先のドメイン名もしくはドメイン名が解決できなければ IP アドレスをマッピングする．これらはどの API を用いてアクセスされたかを直感的に把握できるように，API ごとにグルーピングされたブロック状にマッピングする．解析結果に含まれる API，パスをすべてマッピングしたら，中央の点とパスをグレー色の線分で結ぶ．その後，タイムスタンプの最も古いものから順に 1〜3 を繰り返す．

1. 中央の点を始点に当該 API のグループに属する当該パスに向かって線分を伸ばしていく
2. 内円と交わるときに当該 API 名を黒色に変更する
3. 線分がパスに到着するとパス名を黒色に変更する

5　評価実験

5.1　D3M データセット

D3M 2012 (Drive-by-Download Data by Marionette 2012)[17] は，NTT セキュアプラットフォーム研究所の高対話型の Web クライアントハニーポット(Marionette)で収集したマルウェア検体，攻撃通信データ，マルウェア通信データの 3 つを収録した Web 感染型マルウェアの観測データ群である．今回は，そのうちの 2 つのマルウェア検体と攻撃通信データを用いて提案システムの評価を行った．

5.2　評価方法と結果

D3M 2012 に含まれる検体のうち，セキュリティベンダーから提供されている解析レポートを取得でき，BlackManta での監視の仕方によって振舞いが変わる 2 つの検体に注目して，提案システムが一連の流れを通して動作するかを確認した．利用する検体とそれが含まれる攻撃通信データを表 1 に，提案システムの実行環境を表 2 に示す．検体の実行は仮想環境となるので，あらかじめ 2 検体が仮想環境でも正常に動作することを確認して

いるものであった．各検体の特徴は以下のようになる．

検体 A　アンチデバッキング機能有り，監視する API の違いによる挙動変化無し

検体 B　アンチデバッキング機能無し，監視する API の違いによる挙動変化有り

実験の流れ

1. Pcap ファイルを評価用システムが読み込み，可視化結果をシステムが出力する．
2. 可視化された URL リストからユーザが検体を選択し，システムが Pcap ファイルから検体を抽出する．
3. 抽出した検体を動的解析用端末に移し，システムが動的解析を行う
4. システムが動的解析結果を可視化する．

今回の実験では検体の監視時間を 60 秒に設定したが，監視時間は検体によって変更可能とする．

実験の流れ 1 および 2 について

Pcap ファイルに含まれる通信データを可視化すると，検体の URL と対応する国上に赤色のポールが描画された．ポールをクリックすると検体の URL が赤く反転していることを確認した．（図 5）それぞれの URL をクリックすると，検体がストレージから取り出された．

実験の流れ 3 および 4 について

検体 A　ImmunityDebugger が検体 A を読み込むと，ImmunityDebugger の動作が停止した旨のエラーメッセージが表示され検体も動作しなかった（図 7 左）．ImmunityDebugger を動作させずに検体 A を実行し，ProcessMonitor で観測した（図 7 右）ものと挙動が異なることがわかる．

検体 B　ファイル操作の API（CreateFileA, ReadFile, ReadFileEx, WriteFile, WriteFileEx, DeleteFileA, MoveFileExA）を監視対象にして API 呼び出しを監視すると，CreateFileA という API で特定のファイル（C:¥Windows¥System32¥actived.dll）のみに繰り返しアクセスするという挙動を示した（図 8 左）．続いて CreateFileA 関数を監視対象から除外したファイル操作の API（ReadFile, ReadFileEx, WriteFile, WriteFileEx, DeleteFileA, MoveFileExA）を監視対象とすると，複数回 C:¥Windows¥System32¥actived.dll にアクセスした後に異なる挙動を示した（図 8 右）．

以上のように，評価システムは通信データの可視化で提示された解析対象コードのリストから検体を選択し，動的解析の結果の出力までを一連の流れでできることを確認した．それぞれの検体の解析に要する時間は約 190 秒を超えない範囲（表 3）であった．

6　まとめ

本稿では通信データから DBD 攻撃の解析対象コードの選択を支援する通信挙動可視化インタフェース，および検体の動的解析を自動化する環境を統合した DBD 攻撃解析支援システムを提案した．解析対象コード選択支援部とマルウェア判定支援部を簡易な実装方法で用意し，一連の動作ができる評価用システムを構築した．D3M 2012 データセットを入力し，それに含まれる挙動の分かっている 2 検体を用いて評価用システムを動作させたところ，一連の流れで動作することと，この 2 検体に限れば約 190 秒を超えない時間で動的解析の結果が出力できることを確認した．

4.2 章で述べた評価用システムの解析対象コード選択支援部は，DBD 攻撃におけるマルウェア配布サイトの直前の通信を抽出する簡易な分析部の実装になっている．DBD 攻撃の特徴を可視化して発見しやすくするための分析部とその他のモジュ

表 1　検体のハッシュ値と攻撃通信データ

	ハッシュ値	検体を含む攻撃通信データ
検体 A	33ce026a56df8d5f0a9e9d9eb314fa5193de6532	url_20120321.pcap
検体 B	fdaf6b9ab4f66940317f86b5c2cd0fc957706c58	url_20120323.pcap

表 2 実行環境

OS	Windows 7 Professional 64bit
CPU	Intel® Core™ i5-2540M CPU @2.60GHz
RAM	2.00GB

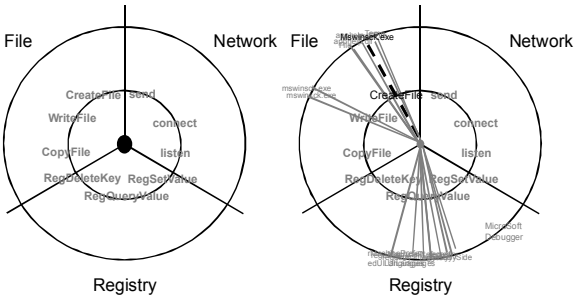

図 7　検体 A の挙動の可視化

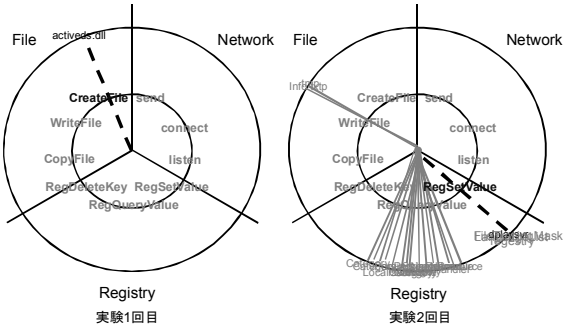

図 8　検体 B の挙動の可視化

表 3　評価用システムの実行時間

検体	実行時間[ms]
検体 A	190479
検体 B	170401

ールとの連携について引き続き検討していく．4.3 のマルウェア判定支援部についても，より有用な方法を組み込むように検討していく．

参考文献

[1] Akiyama, M., Iwamura, M., Kawakoya, Y., Aoki, K., and Itoh, M., “Design and Implementation of High Interaction Client Honeypot for Drive-by-Download Attacks” IEICE Transactions on Communications, Vol.93-B, No 4, pp.1131-1139, 2010

[2] Visoottiviseth, V., Jaralrungroj, U., Phoomrungraungsuk, E., and Kultanon, P., “Distributed Honeypot Log Management and Visualization of Attacker Geographical Distribution”, JCSSE, 11-13 May 2011

[3] 金子 博一，松木 隆宏，新井 悠，“通信トラフィックの分析による Gumblar 感染 PC の可視化”，IEICE Technical Report，IA2010-1，ICSS2010-1，2010

[4] Inoue, D., Yoshioka, K., Eto, M., Hoshizawa, Y., Nakao, K., “Malware Behavior Analysis in Isolated Miniature Network for Revealing Malware's Network Activity”, IEEE International Conference on Communications, 2008

[5] Egele, M., Scholte, T., Kirda, E., Kruegel, C., “A Survey on Automated Dynamic Malware Analysis Techniques and Tools”, ACM Computing Surveys, 44, 2, Article 6, February 8, 2012

[6] 神薗雅紀，西田雅太，星澤裕二，“動的解析を利用した難読化 JavaScript コード解析システムの実装と評価”，マルウェア対策人材育成ワークショップ (MWS2010)，2A3-1，2010

[7] Riverbed Technology, “WinPcap”, http://www.winpcap.org/ (2012/08/20)

[8] Sly Technologies, “jNetPcap”, http://jnetpcap.com/ (2012/08/20)

[9] Canon IT Solutions, “ESET NOD32 Antivirus 4”, http://canon-its.jp/product/eset/ (2012/08/20)

[10] Windows Sysinternals, “Process Monitor”, http://technet.microsoft.com/en-us/sysinternals/bb896645.aspx (2012/08/20)

[11] “IMMUNITY DEBUGGER”, http://immunityinc.com/products-immdbg.shtml (2012/08/20)

[12] “Immunity Python Scripts”, http://tuts4you.com/download.php?view.2939 (2012/08/20)

[13] 情報通信研究機構,“リアルトラフィックの可視化ツール“NIRVANA”を開発”, http://www.nict.go.jp/press/2011/06/02-1.html (2012/08/20)

[14] 高田 哲司, 小池 英樹, “人間による Honeypot の攻撃元ログ調査を支援する User Interface の提案”，マルウェア対策人材育成ワークショップ (MWS 2008)，1014，2008

[15] 松木 隆宏, 新井 悠, 寺田 真敏, 土居 範久, “セキュリティ無効化攻撃を利用したマルウェアの検知と活動抑止手法の提案”，情報処理学会論文誌, Vol.50, No 9, pp.2127-2136, 2009

[16] 笠間 貴弘, 吉岡 克成, 井上 大介, 松本 勉, “実行毎の挙動の差異に基づくマルウェア検知手法の提案”，コンピュータセキュリティシンポジウム(CSS2011)，3B3-4，2011

[17] MWS2012 実行委員会，研究用データセット MWS 2012 Datasets について，http://www.iwsec.org/mws/2012/about.html#datasets (2012/08/20)