SCIS 2024 2024 Symposium on
Cryptography and Information Security
Nagasaki, Japan, Jan. 23 – 26, 2024
The Institute of Electronics,
Information and Communication Engineers

マルウェアの動的解析を支援するネットワークシミュレータの提案
Network Simulator for Supporting Dynamic Analysis of Malware

原 淳一郎 * 毛利 公一 * 金城 聖 * 瀧本 栄二 * †
Junichiro Hara Koichi Mouri Akira Kanashiro Eiji Takimoto

あらまし C&C サーバ等の外部サーバと通信するようなマルウェアの動的解析において, マルウェア本来の挙動を観測しようとするとインターネット接続が必要である. しかし, それをすると攻撃に加担する可能性がある. そのため, ファイアウォールを詳細に設定して攻撃を遮断する方法や, 外部サーバを含むインターネット環境を実機や仮想化技術を用いて模擬する手法が採られる. ただし, これらの方法は, そういった環境の構築のコストが大きい. 以上の背景から, 本論文では, ネットワークシミュレータ内に模擬 C&C サーバや模擬 DNS サーバなどを構築可能とする手法を提案する. これにより, マルウェアがインターネットと直接接続しなくても本来の挙動の観測ができる環境を, 柔軟に低コストで構築できる. 具体的には, マルウェアの全通信をネットワークシミュレータ内に取り込み, シミュレータ内に構成した模擬サーバが応答することでマルウェアを欺瞞し解析する. 提案手法を用いて, 実際に IoT マルウェア Mirai を動作させ, Mirai の挙動として DoS 攻撃と感染拡大活動の観測ができることを確認した.

キーワード ネットワークシミュレータ, マルウェア動的解析, IoT マルウェア

1 はじめに

マルウェアは, C&C サーバや DNS サーバなど外部サーバとの通信を前提として作成されている. その例として, ランサムウェアや Remote Administration Tool (RAT), Internet of Things (IoT) ボットなどのマルウェアがある. これらのマルウェアを動的解析することは, その挙動を把握する上で重要である. 動的解析により, マルウェア本来の挙動を観測するには, マルウェアが外部サーバと通信する必要がある. しかし, インターネットに接続することは, 攻撃に加担する可能性があるため避ける必要がある. このような問題を解決する既存手法として, ファイアウォールを設置して攻撃を遮断する方法や, 実計算機を用いてマルウェアと通信するサーバを模擬した環境で, マルウェアを解析する手法がある [1]. 後者の方法は, インターネットと接続しないため, 攻撃に加担することなく安全である. しかし, 両手法ともに, ネットワークの詳細な設定を適切にする必要があることや, 実計算機の調達および保守をする必要があるなど, コストが大きい. また, マルウェアの検体を変更すると, ネットワーク構成の変更や, 模擬的な C&C サーバや DNS サーバなどの複数の実計算機のプログラムを変更する必要があり, 手間がかかる.

本論文では, 既存手法の課題を解決するため, ネットワークシミュレータ内に模擬サーバを構築可能とする手法を提案する. 提案手法では, マルウェアの全通信をネットワークシミュレータ内に取り込み, ネットワークシミュレータ内に用意した模擬サーバが応答することでマルウェアを欺瞞し解析可能とする. これにより, マルウェアがインターネットと直接接続しなくても本来の挙動の観測ができる環境を低コストで構築できる. 本手法は, 外部ネットワークを模擬した環境をシミュレーションで用意するため, 柔軟なネットワーク構成を実現する. また, 提案手法を実装したシステムと, IoT マルウェアの Mirai を用いた動的解析を行った結果について述べる.

以下, 本論文では, 2 章でマルウェアの動的解析に関する関連研究について述べ, 3 章でネットワークシミュレータを用いたマルウェアの動的解析を支援するシステムを提案する. また, 4 章では提案システムの実装について述べ, 5 章で Mirai を用いた評価を行う. 最後に 6 章で本論文をまとめる.

2 関連研究

これまでに, 外部通信をするマルウェアの動的解析を行うためにさまざまな研究がされてきた. 本物の外部サー

* 立命館大学 〒 525–8577 滋賀県草津市野路東 1 丁目 1-1. Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu-shi, Shiga, 525–8577 Japan.
† 奈良女子大学 〒 630–8506 奈良県奈良市北魚屋東町. Nara Women's University, Kitauoya-higashimachi, Nara-shi, Nara, 630–8506 Japan.

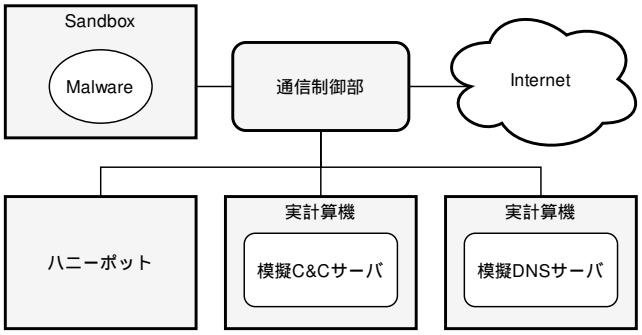

図 1: 既存手法

バを使用しない方法に, 実計算機を用いてマルウェアと通信するサーバを模擬した環境で, 動的解析を行う研究がある [1]. これは, インターネットと接続しない環境で動的解析を行うことが可能である. しかし, 実計算機の調達や保守をする必要があることや, ネットワーク構成の変更に手間がかかるといった問題がある. 本論文では, マルウェアが通信するサーバをネットワークシミュレータ内に構築可能とする手法を提案する. そのため, 実計算機を用意する必要はなく, ネットワーク構成の変更を容易かつ柔軟に行うことが可能である.

マルウェアから悪意のある通信がインターネットへと流出しないようにした上で, 本物の外部サーバと通信させ動的解析を行う研究がある [2]. また, ボットから害のあるプログラムを取り除いた上で外部サーバと通信させた研究 [3] がある. マルウェアの検体を収集する手段であるハニーポットを動的解析に応用した研究もある [4]. これらの手法は, C&C サーバとの接続に成功した場合, 実際に行われた攻撃命令の収集や攻撃の観測が可能である. しかし, インターネットに接続するため, 通信制御を誤った場合, マルウェアからの悪意のある通信が解析環境外へと流出し, 攻撃に加担する可能性がある. また, C&C サーバの挙動に依存するという面もあり, C&C サーバが稼働していない場合には, マルウェアの攻撃を観測できないといった問題がある. 提案手法では, マルウェアの動的解析に必要なネットワークをネットワークシミュレータ内に構築する. これは, 物理的にインターネットと切断するため, 安全である. 加えて, マルウェアの通信には模擬 C&C サーバが応答するため, 本物の C&C サーバの挙動に依存しない解析が可能である.

このように既存手法では, 図 1 に示すように, 通信制御部によりサンドボックス環境と外部ネットワークとの通信を制御する. 一方, 提案手法では, 通信制御部を含め, サンドボックス環境外のネットワークをすべてネットワークシミュレータ内に構築可能とする. これにより, マルウェアの動的解析を支援する環境を安全かつ低コストに構築できる.

3 提案手法

3.1 要件の整理

外部通信をするマルウェアを動的解析し, 詳細な挙動を観測するためには, C&C サーバや DNS サーバなど外部サーバとの通信が不可欠である. 既存手法は, 2 章で示したように, 通信制御部により, マルウェアがインターネットと接続する通信を模擬サーバへと転送する. この手法の問題点は以下である.

- ネットワークの設定を適切にしなければ, 悪意のある通信がインターネットへ流出する危険性がある
- 検体を変更する度に, 複数の実計算機を操作し解析環境を変更する必要がある
- 新たな模擬サーバを導入する際, ネットワーク構成の変更を実環境で行う必要があるためコストが大きい

すなわち, マルウェアの動的解析を支援するシステムは, 以下の特性を持つべきである.

- 安全にマルウェアを動的解析できる環境を構築可能である
- 特定のマルウェアのための動的解析環境を容易に構築可能である
- 支援環境のネットワーク構成は, 柔軟に変更可能である

本章では, これらの要件を満たす提案手法により, 既存手法の問題点を解決可能であることを示す.

3.2 安全な解析環境

インターネットを使用する既存手法では, ファイアウォールを詳細に設定し, 悪意のある通信がインターネットへ流出しないようにする必要がある. しかし, マルウェアの検体に合わせ, ネットワーク構成を変更する必要があるため, 人為的ミスが発生する可能性がある. ゆえに, 安全に動的解析を行うには, インターネットと接続しない環境が望ましい.

そこで, 本手法では, 1 台の実計算機内に, サンドボックス環境とネットワークシミュレータを用意する (図 2 参照). さらに, ルータや模擬サーバをネットワークシミュレータ内に用意する. ルータは, マルウェアの全通信を取り込み, 模擬サーバはルータを介してマルウェアと通信する. このように, 動的解析に必要なネットワークをネットワークシミュレータ内に構築する. そのため, 本環境は, インターネットと物理的に切断された安全な環境を構築可能である.

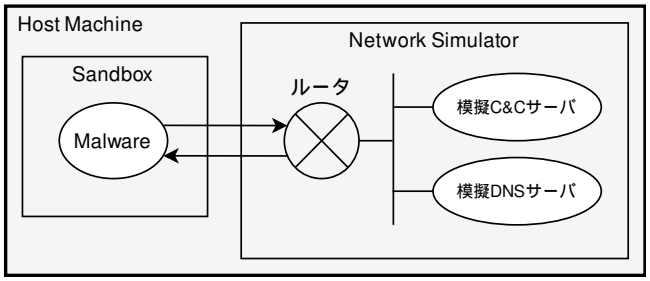

図 2: 提案手法

しかし, 通常のネットワークシミュレータは, 動的解析を行うための機能を備えていないため, 以下の機能をもったネットワークシミュレータの開発が必要である.

- サンドボックス環境からの通信をネットワークシミュレータ内のルータに取り込む
- ルータは, パケットの内容に応じて特定のサーバへとルーティングを行う
- ルーティングされたパケットに対し, 模擬サーバが応答する

まず, サンドボックス環境からの通信をネットワークシミュレータ内のルータに取り込む. これは, サンドボックス環境のデフォルトゲートウェイをネットワークシミュレータ内のルータに設定することにより実現する.

次に, 取り込んだパケットの内容に応じて, 特定の模擬サーバへとルーティングを行う機能が必要である. マルウェアは, 不特定の相手と通信するため, ルータは, 通信プロトコルや宛先ポート番号を元にルーティング先を決める必要がある. これは, ルータのルーティングテーブルに, 静的なルーティングを随時追加することにより実現する. 具体的には, UDP かつ 53 番ポート宛のパケットならば模擬 DNS サーバへと転送するために, そのパケットの宛先 IP アドレスから模擬 DNS サーバの IP アドレスへの静的なルーティングをルーティングテーブルに追加するといったものである.

最後に, ルーティングされたパケットに対し, 模擬サーバが応答する機能が必要である. ルータがルーティングしたパケットは, 模擬サーバの IP アドレスとは異なるため, 通常は破棄される. そのため, TCP/IP レイヤで, 以下のようなパケットを書き換える機能が必要である.

1. パケットが模擬サーバに到着した際, 宛先 IP アドレスと宛先ポート番号を保存する
2. パケットの宛先 IP アドレスと宛先ポート番号を模擬サーバのものに書き換える
3. パケットが模擬サーバから送信される際, 送信元 IP アドレスと送信元ポート番号を保存しておいたものに書き換える

これらの機能により, マルウェアと模擬サーバとの通信を実現する. マルウェアからの通信に対し, ネットワークシミュレータ内に用意した模擬サーバが応答することで, マルウェアを欺瞞し解析可能とする.

3.3 容易かつ柔軟な解析環境

実計算機上に実装した模擬サーバを用いた既存手法では, マルウェアの検体を変更すると, そのマルウェアに合わせたネットワーク構成にする必要や模擬サーバの挙動を変更する必要がある. それには, 複数の実計算機の操作が不可欠である. しかし, 特定の時点でのネットワーク構成や模擬サーバの設定に戻すことは手間がかかる. これは, 解析経験のあるマルウェアを動作させる環境へすることに手間がかかるだけでなく, 第 3 者による再現実験のコストが大きくなる.

提案手法は, 動的解析に必要なネットワークをネットワークシミュレータ内に用意する. ネットワークシミュレータは, 任意のネットワークを容易に作成可能である. 具体的には, ノードを作成し, 通信方式や IP アドレスを設定するのみで, 特定のネットワークを作成可能である. また, ネットワークシミュレータは, シミュレーションシナリオファイルを変更すると, 異なる設定でのシミュレーションが可能である. 本手法は, このようなネットワークシミュレータの特性を活かした解析環境であるため, ネットワーク構成や模擬サーバを容易に作成でき, その設定を柔軟に変更可能である.

4 実装

4.1 概要

提案システムの概要は以下である.

1. サンドボックス環境とネットワークシミュレータが通信可能な環境を構築する
2. マルウェアの全通信を ns-3 内に取り込み, パケットの内容に応じて特定のサーバへルーティングを行う
3. ルーティングされたパケットの内容を書き換え, 模擬サーバの IP アドレスとポート番号にする

これらを実装したシステムにより, マルウェアと模擬サーバとの通信を実現し, マルウェアを欺瞞した動的解析を可能とする.

提案システムの実装に用いるネットワークシミュレータには, ns-3[5] を使用する. 提案手法では, ホストを経由し, サンドボックスとネットワークシミュレータが通信する必要がある. ns-3 は, リアルタイムシミュレーション

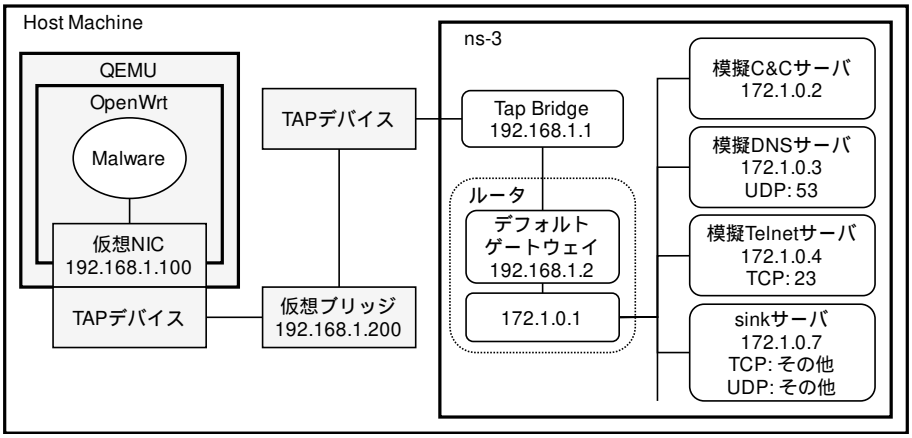

図 3: 提案システムのネットワーク構成

表 1: ソフトウェアのバージョン

ソフトウェア	バージョン
Ubuntu	22.04
ns-3	ns-3.38
QEMU	6.2.0
OpenWrt	18.06.1

およびホストとの通信が可能である．さらに, オープンソースのネットワークシミュレータであるため, ソースコードを改変し, 動的解析に応用することが可能である．

また, 提案システムに Mirai を動作させる機能を追加した．IoT マルウェアは, インターネットと接続し, 外部サーバと通信することを前提として作成されている．Mirai は, C&C サーバを始め DNS サーバなど, いくつかの外部サーバと通信する．加えて, すでに解析がなされていることや, ソースコードが公開されている [6] [1]ため, 外部サーバの仕様を把握可能である．今回は, マルウェアの動的解析に提案システムを利用できることを示すため, 適切な挙動をする模擬サーバを作成可能な Mirai を選定した．

4.2 提案システムの実装

4.2.1 解析環境のネットワーク構築

サンドボックス環境とネットワークシミュレータが通信可能な環境を構築した．提案システムのネットワーク構成を図 3 に示す．また, 実装に使用したソフトウェアのバージョンを表 1 に示す．まず, ホストには, Ubuntu を使用し, サンドボックス環境には, QEMU[7] を用いて組み込み Linux の 1 つである OpenWrt[8] を使用した．次に, ns-3 とホストとの通信には, ns-3 の Tap Bridge 機能 [9] の UseBridge Mode を使用した．Tap Bridge を使用したノードは, ホストの TAP デバイスがパケットを受信した際に, 自分自身から同じパケットを送信する．自分がパケットを受信した際は, ホストの TAP デバイスのコールバック機能を使用し, 同じパケットを TAP デバイスから送信させる．ns-3 内には, ルータとルータに接続した模擬サーバを構築した．これらの説明は次節で行う．

4.2.2 ルーティング機能

提案システムでは, マルウェアの全通信をネットワークシミュレータ内に取り込む必要がある．これは, OpenWrt のデフォルトゲートウェイを ns-3 内のルータにすることで実現した．本節では, 取り込んだ通信を内容に応じて特定の模擬サーバへとルーティングを行う機能について述べる．

ルータに実装したルーティング機能のフローチャートを図 4 に示す．まず, ルータに届いたパケットのイーサネットヘッダを確認することで, サンドボックス環境から届いたパケットかどうかを判断する．これは, マルウェアの中には, 送信元 IP アドレスを偽造する IP スプーフィングを行うものがあるためである．次に, 4.3 節で示すように, Mirai は, 名前解決により模擬サーバの IP アドレスを取得した場合, 模擬サーバの IP アドレスに通信を試みる．この場合は, 静的なルーティングを作成する必要がないため, 172.1.0.0/16 のネットワーク宛のパケットに対しては, 通常のルーティングのみを行うようにした．最後に, 宛先 IP アドレスを確認し, ルータのルーティングテーブルになければ, 通信プロトコルや宛先ポート番号を確認し, 静的なルーティングをルータのルーティングテーブルに追加する．このとき, 宛先ポート番号に対応する模擬サーバが存在する場合は, その模擬サーバへのルーティング規則を作成し, なければパケットの受信のみを行う sink サーバへのルーティング規則を作成する．宛先ポート番号と模擬サーバの対応付けは, シミュレーション開始時に行う．例えば, UDP の 53 番ポート宛の

[1] 初期化に失敗する部分を一部改修

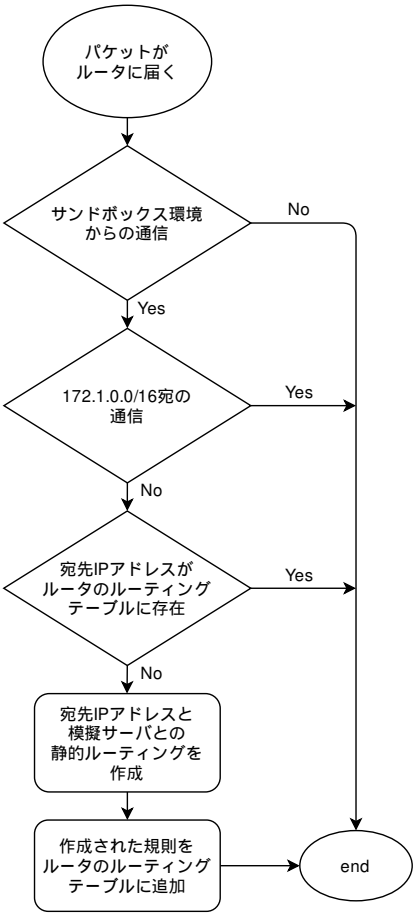

図 4: ルーティング機能のフローチャート

パケットならば模擬 DNS サーバへルーティングを行い, TCP の 23 番ポート宛のパケットならば模擬 Telnet サーバへルーティングを行う. この機能により, マルウェアと模擬サーバとの通信経路を確立することを可能とした.

4.2.3 模擬サーバにおけるパケット書き換え

ルータから転送されたパケットの宛先 IP アドレスと宛先ポート番号は, 模擬サーバの IP アドレスおよびポート番号とは異なる. そこで, 模擬サーバにパケットが到着した際に, 宛先 IP アドレスと宛先ポート番号を模擬サーバのものに書き換える機能を実装した.

ns-3 では, 各ノードが同じクラスをインスタンス化したインターネットスタックを持っている. 提案システムでは, インターネットスタックの TCP/IP レイヤのソースコードに, 自身のノードが模擬サーバの場合, パケットを書き換える機能を実装した. パケット書き換えに用いる情報は以下である.

- 模擬サーバの IP アドレス
- 模擬サーバのポート番号
- 受信した際の IP アドレス
- 受信した際のポート番号

表 2: Mirai の攻撃の種類

攻撃名
UDP フラッド
VSE フラッド
DNS フラッド
SYN フラッド
ACK フラッド
高度な ACK フラッド
GRE - IP を用いた UDP フラッド
GRE - Eth を用いた UDP フラッド
高度な UDP フラッド
Slow HTTP

これらの値を必要な際に入出力可能な変数を作成し, TCP/IP レイヤのソースコードからアクセス可能とした. 指定したノード以外に対しては, 何もしないため, すべてのノードで改変した TCP/IP レイヤのソースコードを利用できる. この機能により, マルウェアからの通信に対し模擬サーバが応答することを可能とした.

4.3 模擬サーバの実装

提案システムに Mirai を動作させるための機能を追加した. Mirai は, 数種類の外部サーバと通信するため, 模擬サーバも必要なだけ作成した. 本節では, Mirai が UDP フラッド攻撃や感染拡大活動を行う際に使用する C&C サーバ, DNS サーバ, Telnet サーバ, レポートサーバを模擬した機能について述べる.

模擬 C&C サーバ Mirai は, C&C サーバと接続し, 定期的に信号を送受信するハートビートを行う. さらに, 攻撃命令を受信すれば, 表 2 に示すような攻撃を行う. そのため, 作成した模擬 C&C サーバには, Mirai から送信されるハートビートに返答する機能と, 攻撃命令の送信を行う機能が必要である. そこで, Mirai からのハートビート「\x00\x00\x00\x00」に対し, 同様の文字列を返す機能を実装した. また, Mirai が可能な攻撃 10 種類すべてに対して攻撃命令を作成し, ハートビートが送られてきたタイミングで攻撃命令を送信できるようにした.

模擬 DNS サーバ Mirai は, C&C サーバや感染可能な端末の情報を報告するレポートサーバの名前解決に DNS サーバを利用する. ゆえに, 作成する模擬 DNS サーバには, ドメイン名に応じて, 模擬 C&C サーバや模擬レポートサーバの IP アドレスを返す機能を実装した. 具体的には, C&C サーバのドメイン名「cnc.changeme.com」に対し, 模擬 C&C サーバの IP アドレス「172.1.0.2」を返す機能である.

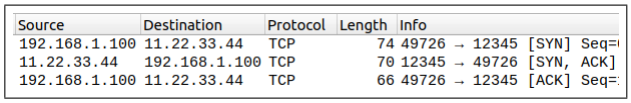

図 5: 提案システムの動作例

模擬 Telnet サーバ　今回使用した Mirai は, 感染拡大活動の際, 他デバイスへの接続方法として, Telnet による接続を試みる. Telnet は, コネクションを確立した後, クライアントとサーバ間でオプションの要求および確認を互いに行う. Mirai は, 1 回のみオプションの要求に対し返答するため, それに合わせた仕様にした. Telnet サーバは, オプションを確認した後, ユーザ名とパスワードの入力を促す必要がある. その後, 受信したメッセージに対しシェルを模擬した文字列を送信する必要もある. 模擬 Telnet サーバには, これらの機能を実装した. この際, 模擬したシェルは, Mirai が感染対象としている OS の 1 つである OpenWrt のシェルである. 例えば, 「sh」という文字列に対し, 「\n\nBusyBox v1.28.3 () built-in shell (ash)\n\n」という文字列を返すなどの機能を実装した.

模擬レポートサーバ　Mirai は, レポートサーバに, 感染可能な端末の IP アドレスやパスワードなどの情報を報告する. ゆえに, 模擬レポートサーバには, 受信したメッセージをログとして出力する機能のみを実装した.

5　評価

本章では, 提案手法の評価について述べる. 提案手法を実装したシステムの動作確認を行った. また, マルウェアの動的解析を支援可能なシステムであることを明らかにするために, Mirai の動的解析を行い, その挙動として DoS 攻撃と感染拡大活動の観測を行った.

5.1　提案システムの動作確認

提案システムの動作確認では, ルーティング機能とパケット書き換え機能が正しく動作しているかを評価する. これは, OpenWrt から不特定の IP アドレスとポート番号宛に送信された通信に対し, 模擬サーバが応答可能かを確認することにより行う. 模擬サーバには, TCP または UDP のパケットに対し受信のみを行う sink サーバを使用する. ただし, TCP の通信では, 必要に応じて ACK パケットを送信する. また, 通信の様子は, Wireshark[10] を用いて, ホストのブリッジにおいてパケットキャプチャをし観測する.

OpenWrt(192.168.1.100) 内で, nc コマンドを用いて, IP アドレス 11.22.33.44 の 12345 番ポート宛に TCP パケットを送信したときの OpenWrt と sink サーバとの通信の様子を図 5 に示す. OpenWrt から解析環境のネットワークに存在しない IP アドレス 11.22.33.44 の 12345 番ポート宛に送信された TCP のパケットに対し, 3way-handshake によりコネクションが確立できていることが分かる. これは, パケットがルータにより sink サーバに転送され, sink サーバが通信を受理したことを示している. この結果から, ルーティング機能とパケット書き換え機能は, 正しく動作していることが分かる.

5.2　Mirai の DoS 攻撃の観測

Mirai は, 4.3 節で示したように, 10 種類の攻撃を行うことが可能である. 本節では, その内の 1 つである UDP フラッド攻撃の観測について述べる.

Mirai による UDP フラッド攻撃の様子を図 6 に示す. 1 番目のパケットは, 模擬 C&C サーバ (172.1.0.2) から Mirai(192.168.1.100) へ攻撃命令を送信したものである. ここで送信した攻撃命令の内容は, IP アドレス 1.2.3.4 に 10 秒間 UDP フラッド攻撃を行う趣旨のものである. それに対し, Mirai は, ACK パケットを返した後, 3 番目以降のパケットに示すように, 攻撃命令の通りの UDP フラッド攻撃を行っていることが分かる. 同様に, 他のすべての攻撃についても攻撃を観測できた.

5.3　Mirai の感染拡大活動の観測

今回使用した Mirai は, 一部の IP アドレスを除くランダムな IP アドレスに Telnet でのログインを試みる攻撃を行う. Mirai は, 活動している間, 絶えず数多くのパケットを送信し, Telnet ログインを試みる. 今回は, そのパケットの中から代表して, ルータに最初に届いた 23 ポート宛のパケットのみを模擬 Telnet サーバにルーティングするよう設定した.

5.3.1　ログインの観測

Mirai の総当たり攻撃の観測を行うために, 模擬 Telnet サーバへのログインを許可しないように設定した. まず, Mirai と模擬 Telnet サーバは, TCP のコネクションを確立した後, Telnet のオプションを互いに確認した. 次に, 模擬 Telnet サーバは, ログインのユーザ名とパスワードを入力するよう Mirai に促した. その結果, Mirai は, 時間を置き, 図 7 に示すように何度もユーザ名とパスワードを変えてログインを試みた. このように, Mirai の総当たり攻撃を観測した.

5.3.2　レポートサーバへの報告

模擬 Telnet サーバへのログインを許可すると, Mirai はいくつかのコマンドを模擬 Telnet サーバへ送信した. Mirai が「/bin/busybox MIRAI」と送信したのに対し, 模擬 Telnet サーバは, 「MIRAI: applet not found」という OpenWrt のシェルの反応を模した文字列を送信した. その結果, Mirai は, この Telnet サーバが有効であ

Time	Source	Destination	Protocol	Length	Info
10.043396…	172.1.0.2	192.168.1.100	TCP	80	23 → 43066 [ACK] Seq=1 Ack=8 Win=131072 Len=14
10.044804…	192.168.1.100	172.1.0.2	TCP	66	43066 → 23 [ACK] Seq=8 Ack=15 Win=29200 Len=0 T
10.057177…	192.168.1.100	1.2.3.4	UDP	554	65084 → 477 Len=512
10.059431…	192.168.1.100	1.2.3.4	UDP	554	65084 → 477 Len=512
10.060161…	192.168.1.100	1.2.3.4	UDP	554	65084 → 477 Len=512
10.060797…	192.168.1.100	1.2.3.4	UDP	554	65084 → 477 Len=512
10.061221…	192.168.1.100	1.2.3.4	UDP	554	65084 → 477 Len=512
10.061633…	192.168.1.100	1.2.3.4	UDP	554	65084 → 477 Len=512
10.062036…	192.168.1.100	1.2.3.4	UDP	554	65084 → 477 Len=512
10.062432…	192.168.1.100	1.2.3.4	UDP	554	65084 → 477 Len=512

図 6: UDP フラッド攻撃の様子

```
Telnet-Server: -Login Information-
    UserName: root
    Password: xc3511

Telnet-Server: -Login Information-
    UserName: root
    Password: klv123

Telnet-Server: -Login Information-
    UserName: root
    Password: 888888

Telnet-Server: -Login Information-
    UserName: root
    Password: vizxv

Telnet-Server: -Login Information-
    UserName: 666666
    Password: 666666

Telnet-Server: -Login Information-
    UserName: admin
    Password: admin
```

図 7: 総当たり攻撃の様子

ると判断し, 図 8 に示すように, Telnet サーバの情報をレポートサーバに報告した. この際, Mirai は, レポートサーバの名前解決を行うため, 模擬 DNS サーバによって模擬レポートサーバの IP アドレスを返した. Mirai は, レポートサーバへと以下の情報を送信した.

- 模擬 Telnet サーバの IP アドレス
- 模擬 Telnet サーバのポート番号
- ログインしたユーザ名とその長さ
- ログインしたパスワードとその長さ

ゆえに, Mirai が Telnet サーバにログインし, その情報をレポートサーバへと送信する感染拡大活動の一連の流れを観測できた.

5.4 考察

本評価では, Mirai が行う通信をネットワークシミュレータに引き込み模擬サーバへ転送し, 各模擬サーバにおいて必要に応じて IP アドレスの書き換えを行った. これにより, Mirai を欺瞞し, Mirai の通信と動作の解析ができることを確認した. したがって, 外部との通信を行

```
Telnet-Server: -Receive-
  size: 19
  str : /bin/busybox MIRAI
  hex : 2f62696e2f62757379626f78204d4952414900

Telnet-Server: -Receive-
  size: 2
  str :

  hex : 0d0a

Telnet-Server: -Send-
  size: 24
  str : MIRAI: applet not found

  hex : 4d495241493a206170706c6574206e6f7420666f756e64 a

Telnet-Server: -Send-
  size: 13
  str : root@root:~$
  hex : 726f6f7440726f6f743a7e2420

DNS-Server: -Receive-
  size: 37
  str : ◆reporchangemecom
  hex : 1fe401000001000000000000067265706f72740863686 16e6
7656d6503636f6d0000010001

DNS-Server: -Send(Raw)-
  size: 53
  data: 1fe4810000010001000000000672657 06F72740863686 16E6
7656D6503636F6D0000010001c00c0001000100000e100004ac010005

Report-Server: -Accept-
Report-Server: -Receive-
  size: 1
  str :
  hex : 00

Report-Server: -Receive-
  size: 16
  str : ^◆◆6root1234
  hex : 5ea6a236001704726f6f740431323334
```

図 8: レポートサーバへの報告

うマルウェアの動的解析を提案手法で可能であることが明らかとなった.

一方で, UDP フラッド攻撃など大量のトラフィックが生成される場合において, ns-3 の処理速度がパケット生成速度に追い付かず, パケットロスが発生する場合があった. マルウェア解析において攻撃トラフィックをすべて受信する必要はないと考えられるが, パケットロス数を低減する措置が必要であると考えられる.

6 おわりに

本論文では, ネットワークシミュレータを用いたマルウェアの動的解析を支援するシステムを提案した. また, 提案手法を実装し, Mirai の動的解析を行うことが可能

なことを実証した．提案手法を実現したシステムは，マルウェア本来の挙動の観測ができる環境を，低コストで構築可能である．加えて，ネットワークシミュレータを利用しているため，ネットワーク構成を柔軟に変更可能である．

今後は，多種のマルウェアの動的解析を支援可能なシステムを作成するために必要なことを検討していく．そのために，提案システムをどのように拡張すれば良いかを検討し，必要となる機能があれば実装する．

参考文献

[1] 鉄 穎, 楊 笛, 保泉 拓哉, 中山 颯, 吉岡 克成, 松本 勉,“IoT マルウェアによる DDoS 攻撃の動的解析による観測と分析,” 情報処理学会論文誌, pp. 1321-1333 (2018)

[2] Ying-Dar Lin, Tzung-Bi Shih, Yu-Sung Wu, and Yuan-Cheng Lai,“Secure and transparent network traffic replay, redirect, and relay in a dynamic malware analysis environment,” SECURITY AND COMMUNICATION NETWORKS, pp. 626-640 (2014)

[3] 河口 綾摩, 空閑 洋平, 中村 修,“Mirai 型 DDoS ボットネットの監視環境の構築,” マルチメディア, 分散, 協調とモバイル (DICOMO2017) シンポジウム (2017)

[4] Miyoung Kim, Misun Kim, and Youngsong Mun,“Design and Implementation of the Honey-Pot System with Focusing on the Session Redirection,” Computational Science and Its Applications - ICCSA, pp. 262-269 (2004)

[5] ns-3, https://www.nsnam.org/ (参照 2023/12/2)

[6] Jerry Gamblin, Mirai のソースコード, https://github.com/jgamblin/Mirai-Source-Code (参照 2023/12/2)

[7] QEMU, https://www.qemu.org/ (参照 2023/12/2)

[8] OpenWrt, https://openwrt.org/ (参照 2023/12/2)

[9] ns-3, Tap Bridge Model, https://www.nsnam.org/docs/release/3.9/doxygen/group___tap_bridge_model.html (参照 2023/12/2)

[10] Wireshark, https://www.wireshark.org/ (参照 2023/12/2)