Vol.2017-ARC-225 No.41
Vol.2017-SLDM-179 No.41
Vol.2017-EMB-44 No.41
2017/3/10

組込みシステム向けマルウェアMiraiの攻撃性能評価

長柄 啓悟[1,a)]　松原 豊[2]　青木 克憲[2]　高田 広章[2]

概要：近年 IoT 機器が注目され，その脆弱性が明らかになっている．組込み/IoT システムの脆弱性とその対策を検討するため，既存マルウェアである Mirai による攻撃性能を評価した．Mirai とは，組込み Linux 上で動作するコンピュータをボット化し，DDoS 攻撃を行うマルウェアである．2016 年 9 月のセキュリティブログへの攻撃では，史上最大規模である 620 Gbps の攻撃が観測された．公開されているソースコードを元に Mirai の動作概要について把握し，VM 環境上で Mirai の動作環境を構築し，実際の通信や攻撃を確認し，攻撃の性能を計測した．次に，VM 環境と同様にローカルネットワーク下で複数の実機を用いて Mirai 動作環境を構築し動作させ，ボットが動作する実機の CPU やネットワーク帯域について調べ，Mirai の感染や攻撃の対策を検討した．Mirai の攻撃性能は，攻撃を行うボットである組込みボードが 1 台で最大約 248 Mbps のパケットを送信していることや，台数に応じて攻撃量が比例して増加することが確認でき，約 2500 台のボットがあれば，前述した 620 Gbps の大規模攻撃が十分に可能であることを確認した．

1. はじめに

近年 IoT 機器が注目され，多種多様な組込みシステムがネットワークに繋がるようになり，それら機器の脆弱性が明らかになっている．このような組込み機器は，製品の規模やアーキテクチャの多様性，資源の乏しさ，使用期間の長さ，コストの制約からセキュリティが十分に考慮されていないことがある [4]．脆弱性を悪用される具体例として，IoT マルウェアである Mirai が挙げられる．Mirai とは，組込み Linux が動作する，デジタルビデオレコーダーやネットワークカメラ，家庭用ルータなどの IoT 機器を対象にボットを感染させ，DDoS 攻撃を行うマルウェアである．2016 年 9 月のセキュリティブログへの攻撃や同年 10 月の DNS サーバープロバイダである Dyn 社への攻撃では，史上最大規模である 620 Gbps の攻撃が観測された [6]．この規模の攻撃を防ぐのにかかるコストは莫大であり，DNS サーバが攻撃にあうとそれを利用する多くのサービスも同時に使用できなくなってしまう．また，同年 9 月に Mirai の作者がソースコードを公開したことにより，これを利用した亜種が確認され，今後も脆弱な IoT 機器を対象としたマルウェアの増加が懸念される．

このような被害をなくすためには，Mirai がどのように動作するのか解析する必要がある．Mirai が悪用する脆弱性や，その攻撃手法を調査し，Mirai による DDoS 攻撃の危険性を把握し対策しなければならない．

本研究では，公開されているソースコードを元に Mirai の攻撃性能について評価した．予備実験として VM 上で Mirai の動作環境を構築し，実際の通信の様子や攻撃の流れを確認し，攻撃の性能を計測した．次に，ローカルネットワーク下での実機実験として複数の組込みボード（odroid-c2）を用いて，VM と同様に Mirai の動作環境を構築し，攻撃の性能や影響の調査を行なった．さらに，RTOS（TOPPERS/ASP）や組込みシステム向け TCP/IP スタック（Lwip）からなる http サーバが動作する静的な組込みシステムのプロトタイプを対象に攻撃を行った．

2. Mirai の概要

本章では，本研究で扱う Mirai の概要について説明す

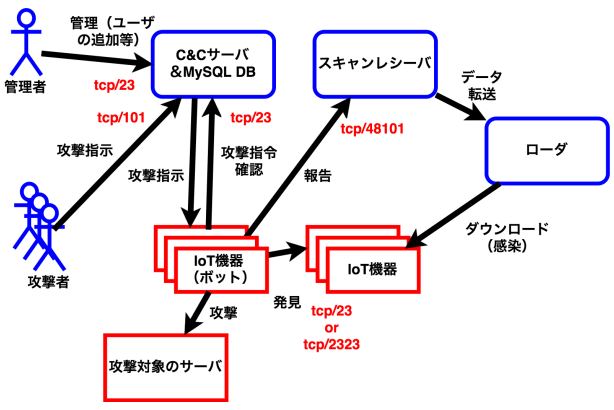

図 1　Mirai の概要

1　名古屋大学工学部電気電子情報工学科
Department of Information Engineering, Nagoya University
2　名古屋大学大学院情報科学研究科
Graduate School of Information Science, Nagoya University
a)　nagara@ertl.jp

る．図 1 のように，Mirai は，MySQL データベースを含む C&C サーバ，ボット，スキャンレシーバやローダから成っている．Mirai の概要について，機能ごとに順に説明する．

2.1 C&C サーバ

C&C サーバは Go 言語によって記述されていて，Go 言語特有の非同期処理である Go ルーチンによって Telnet サーバと API サーバを立てて，ボットやユーザからの接続を待ち，MySQL にユーザや攻撃履歴を記録している．C&C サーバの主な機能は，ボット管理機能，攻撃指示機能，ユーザ管理機能に分けられる．ボット管理機能は，ボットをリストで管理し，攻撃指示をバッファによってボットに伝える機能である．攻撃指示機能は，登録されたユーザが使用できるもので，C&C サーバの 101 番ポートに Telnet 接続してコマンドを入力し攻撃指示を出す機能である．コマンドは「APIKEY|攻撃方法　対象　時間」の形で入力し，攻撃対象は IP アドレスで指定し「,」で区切ることで複数選択可能で，攻撃時間は 1 秒から 1 時間まで選択可能で，表 1 のように攻撃方法は 10 種類ある．APIKEY はユーザを識別するための鍵で初期設定では NULL である．ユーザ管理機能は，管理者が C&C サーバの 23 番ポートに Telnet 接続し，ユーザ名とパスワードでログインし，コマンドによって，ボットの数を数えたり，ユーザを登録したり，攻撃指示機能同様に攻撃を行ったりできる機能である．ユーザ登録時にユーザ名とパスワード，攻撃に使用可能なボット数，最大攻撃可能時間，冷却時間を入力する．最後に，MySQL によるデータベースについて説明する．MySQL にはユーザリストや攻撃記録があり，MySQL のユーザリストには，ユーザ管理機能での登録情報の他に支払日といった情報の欄がある．攻撃記録にはユーザ ID，攻撃開始時間，攻撃時間，コマンド，使用ボット数が記録され，攻撃時に指示が被らないようにするための管理に使われている．

表 1　攻撃の種類

攻撃名	詳細
UDP 攻撃	UDP パケットを大量に送る
VSE 攻撃	ゲームエンジンに対して UDP パケットを大量に送る
DNS リゾルバ攻撃	DNS に存在しないドメイン名の名前解決要求をする
SYN 攻撃	SYN パケットを大量に送る
ACK 攻撃	ACK パケットを大量に送る
STOMP 攻撃	TCP セッション確立後に ACK パケットを大量に送る
GRE IP 攻撃	GRE プロトコルによるパケットを大量に送る
GRE イーサネット攻撃	イーサネットと GRE プロトコルによるパケットを大量に送る
プレーン UDP 攻撃	高速化のために最適化した UDP パケットを大量に送る
HTTP 攻撃	HTTP リクエストを大量に送る

2.2 ボット

ボットは，C 言語で記述されているため，x86，MIPS，MPSL，ARM，PowerPC，m68000，SuperH といった多くのアーキテクチャに対応している．ボットの主な機能として，防御機能，スキャン機能，DoS 攻撃機能がある．防御機能は，ボットに感染した直後にポートを塞ぎ他のマルウェアからの感染を防いだり，Linux のウォッチドッグを排除したりして，ボットの活動の阻害要因を減らすものである [9]．スキャン機能は，root と admin といった 61 種類のユーザ名とパスワードの組み合わせでログインが可能な IoT 機器を探す機能である．スキャン対象はランダム生成したものであり，ループバックアドレスやアメリカ国防総省などに割り当てられている一部の IP アドレスはあらかじめ除外されている．DoS 攻撃機能は，C&C サーバの指示に従い攻撃を行うものである．攻撃の種類は，UDP フラッド攻撃や DNS リゾルバフラッド攻撃，SYN フラッド攻撃，ACK フラッド攻撃，HTTP フラッド攻撃といった一般的なものの他に，VSE フラッド攻撃や GRE イーサネットフラッド攻撃といった見慣れない攻撃方法もある．STOMP フラッド攻撃 は，非常にシンプルで拡張可能なメッセージパッシングプロトコル である STOMP（Simple Text Orientated Messaging Protocol）によって，TCP セッション確立後に ACK パケットを大量に送りつける攻撃である [6]．GRE IP フラッド攻撃は，トンネルプロトコルである GRE（Generic Routing Encapsulation）による UDP パケットを大量に送りつける攻撃である．プレーン UDP フラッド攻撃は，パケットごとのポート番号の変更やチェックサムの過程を省き，通常のものよりもパケットの送信を高速化した UDP 攻撃である．

2.3 スキャンレシーバ&ローダ

最後に，スキャンレシーバとローダについて説明する．スキャンレシーバはボットからのスキャン結果を受け取り出力する．ボットが 48101 番ポートで Telnet 接続して，IP アドレスとポート番号，ユーザ名，パスワードについて報告を行う．報告を受けてスキャンレシーバは，IP アドレスとポート番号，ユーザ名，パスワードを出力する．ローダはスキャンレシーバの出力データを入力として受け取ることで，自前のダウンローダ，wget や TFTP を使ってボットプログラムを対象の IoT 機器にダウンロードさせ，ボットを実行させる [6]．

Vol.2017-ARC-225 No.41
Vol.2017-SLDM-179 No.41
Vol.2017-EMB-44 No.41
2017/3/10

3. Mirai の攻撃性能評価

3.1 実験目的

Mirai のような IoT マルウェアによる被害をなくすためには，Mirai の詳細を知る必要がある．Mirai が悪用する脆弱性や，その攻撃手法を調査し，Mirai の攻撃の威力を把握し対策しなければならない．そのために，本研究では，表 2 のように，実際のマルウェアの動作環境を再現し，攻撃の様子について確認した．

3.2 実験方法

まず，ソースコードをコンパイルし実行するための VM 環境を構築した．セキュリティの点からマルウェアの解析は，ローカルネットワーク下で行うことが望ましく，VM はこの点において適している．この節では，環境構築として，インストール方法や VM のネットワーク構成について説明する．

使用する VM には oracle VirtualBox ver.5.1.10 を使用し，この VM 上で Linux である ubuntu 16.04 LTS を使用し，これらを複数用いてボットや C&C サーバの通信や攻撃の様子を確認した．Mirai のソースコードは，これが公開されているサイト [10] からダウンロードした．VM について，ネットワークの設定を VM 同士で相互の通信が可能な「NAT ネットワーク」に変更した．VM に割り当てる CPU 数を 1 CPU にし，メモリを 512 MB とした．VM 上の ubuntu の環境を Mirai の実行が可能なものにするために，公式サイトから Go 言語の環境をインストールし，apt-get コマンドにより，my-sql-server，mysql-client，electric-fence，ufw をインストールした．ローカルネットワーク下であるため，実行のための Go 言語環境として，/mirai/cnc/attack.go と/mirai/cnc/database.go で使用する必要なパッケージ [11][12] をそれぞれインストールし適切な場所に名前を設定しておいた．端末で「mysql -u root -p」と入力し MySQL を起動し，MySQL 設定スクリプトである/scripts/db.sql の 2 行目に「use mirai」を加え，このスクリプトを実行した．また，MySQL の画面にて「INSERT INTO users VALUES (NULL, 'root', 'password', 0, 0, 0, 0, -1, 1, 30, '');」と入力することにより，管理者ユーザを登録した．ubuntu のファイアウォールである ufw で，使用するポート番号（23，101，48101）を開放した．このように作成した VM をクローンによって，図 2 のように，ボット，C&C サーバ，スキャンレシーバ，攻撃対象のサーバの 4 種類に分け，ボットにおいてはホストマシンのメモリが許す 5 台まで作成し，それぞれに IP アドレスを手動で割り当てた．ボットが C&C サーバに接続する時には，あらかじめ設定したドメイン名を DNS サーバを利用して名前解決しているため，DNS サーバのない環境でも動作するようにソースコードを書き換えて名前解決の過程を跳ばし，直接 IP アドレスやポート番号を指定するようにした．コンパイルにはソースコード内のビルドスクリプト（/mirai/build.sh）を利用した．

表 2 実験環境

参照	C&C サーバ	攻撃ノード	攻撃対象
3.3	VM	VM	VM
3.4	odroid-c2	odroid-c2	ホストコンピュータ
3.5	odroid-c2	odroid-c2	GR-PEACH

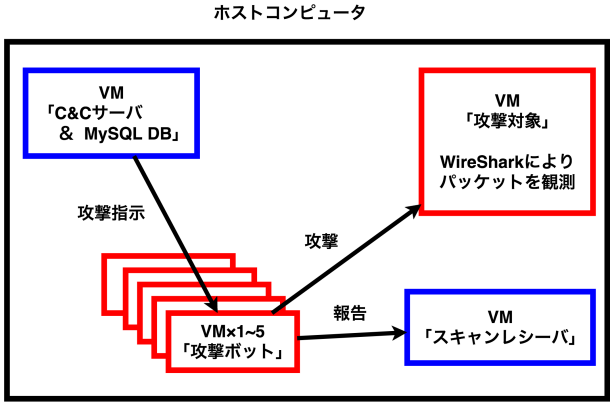

図 2 Mirai 実行のための VM 環境

3.3 VM における攻撃性能評価

この節では，VM における Mirai の攻撃の様子や，その性能評価について述べる．実機での実験を行う前に，予備実験として Mirai が動作することを VM 環境で確認する．

DDoS 攻撃の詳細を調べるために，ボットから攻撃対象に攻撃を行い，攻撃対象の Wireshark で送られてくる攻撃パケットを確認した．攻撃方法は攻撃時間を 10 秒，攻撃対象を 1 台として，各攻撃方法ごとに各 10 回行った．UDP フラッド攻撃においては，ボットの攻撃台数を 1 台から 5 台まで 1 台ずつ変化させ，攻撃パケットがどのように変化するのか確認した．また，攻撃を行なっているボットの CPU 使用率の様子を top コマンドで確認した．

実際に DDoS 攻撃を行なった結果が図 3 や表 3 になる．

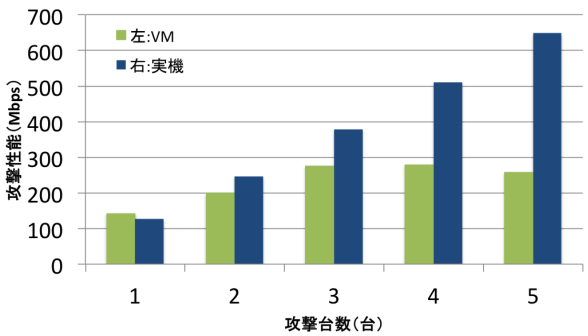

図 3 VM と実機において攻撃台数を変化させた時の UDP フラッド攻撃の性能

Vol.2017-ARC-225 No.41
Vol.2017-SLDM-179 No.41
Vol.2017-EMB-44 No.41
2017/3/10

表 3　各攻撃の性能（攻撃を行う VM が 1 台の場合）

攻撃方法	パケット長（byte）	パケット数（個/s）	転送速度（Mbps）
UDP	556	26513	142.22
VSE	69	27266	15.05
SYN	76	36774	22.35
ACK	568	35384	160.78
GRE IP	580	38665	179.40
GRE ETH	594	38638	183.61
UDP F	556	41827	186.05
HTTP	350	36787	10.89

表 3 は攻撃に使われる VM が 1 台の場合の各攻撃方法によるおよそのパケット長，1 秒あたりのパケット数，転送速度になる．この実験ではローカルネットワーク下で直接 IP アドレスを指定し，DNS サーバを利用しなかったことから DNS リゾルバフラッド攻撃を行うことができなかった．STOMP フラッド攻撃は環境が異なる理由からか正常に動作しなかった．図 3 は UDP フラッド攻撃において，VM の攻撃台数を 1 台から 5 台まで変化させた時の転送速度の様子である．攻撃を行う VM での CPU 使用率は，top コマンドを確認したところ最大値が約 95 %であった．CPU 使用率は，スキャン機能を除いた場合通常時は約 1 %であり，攻撃開始とともに上昇し最大値で止まり攻撃時間終了とともに元に戻った．

表 3 の結果より攻撃性能として，150 Mbps 前後の数値が確認された．それぞれの攻撃では，通信の確立の処理や，プレーン UDP フラッド攻撃を除いて，それぞれのポート番号やデータの中身をパケットごとにランダムに変更してパケットを作成する処理がある．結果の数値は，回線の理論上の転送量の最大値である 1 Gbps と比べると約 15 %しかなく，この最大値による攻撃性能の制限はみられず，実際に攻撃中は VM の CPU 使用率が 95 %と大きく割り当てられていたため，性能は攻撃を行うコンピュータの CPU に依存しているといえる．そのため，VM の台数を増やしすぎると，ホストコンピュータにおいて VM に割り当てる CPU やメモリが足りなくなり，図 3 のように攻撃性能が台数が 3 台の時点で頭打ちになり，さらに増やすと攻撃性能が落ちてしまったと考えられる．Mirai の攻撃時の CPU 使用率が高いことから，CPU 使用率を監視することで Mirai の感染を確認できると考えられる．

3.4　実機における攻撃性能評価

この節では，実機として組込みボード（odroid-c2）を用いた環境で，Mirai の攻撃の性能について確認した．実機を用いた環境で実験を行うことで，実際の Mirai の動作により近い性能や影響を確認できる．

まず，実験環境や実験方法について述べる．ボットに感染した IoT 機器のモデルとして，ARM プロセッサを搭載したシングルボードコンピュータである Hardkernel 社の odroid-c2 を使用した．また，C&C サーバにも同一の組込みボードを利用した．このボードのイーサネットは 1 Gbps である．組込みボードにはそれぞれ ubuntu の 16.04 LTS Xenial Xerus をインストールした．この ubuntu に VM の時と同様に MySQL などの必要なものをインストールし，ファイアウォールを無効にするように設定し実行環境を整えた．Mirai の攻撃対象として，ホストコンピュータを使用し，そのコンピュータ上で Wireshark による計測を行った．HTTP 攻撃の場合においてのみ，攻撃対象を VM の Apache によるサーバとした．それぞれに IP アドレスを手動で設定し，機器同士を LAN ハブ（伝送速度 1 Gbps）でつなぎ，セキュリティを考慮しインターネットに接続せず，図 4 のようにローカルネットワークを構築した．

UDP フラッド攻撃を行う組込みボードを 1 台から 5 台に変化させ，攻撃性能を計測した結果が図 3 になる．1 台の場合の転送速度は 127.18 Mbps で，2 台の場合の転送速度は 247.04 Mbps で，3 台の場合の転送速度は 378.75 Mbps で，4 台の場合の転送速度は 510.60 Mbps で，5 台の場合の転送速度は 648.54 Mbps と，台数に比例して攻撃性能が増加している．この時，攻撃を行う組込みボード上での Mirai の CPU 使用率が 95 %を超えていることを top コマンドにより確認した．

また，他の攻撃方法について実機で行なった場合の攻撃

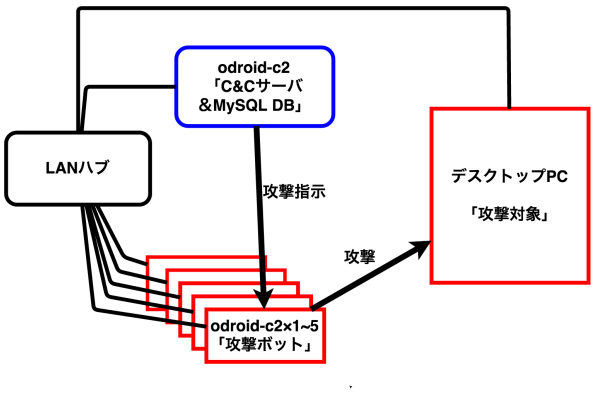

図 4　Mirai 実行のための実機環境

表 4　攻撃台数が 1 台と 5 台の場合の各攻撃の転送速度（ Mbps）

攻撃方法	1 台	5 台
UDP	127.18	648.54
VSE	36.47	184.26
SYN	23.04	113.46
ACK	98.50	612.74
GRE IP	198.75	963.53
GRE ETH	200.67	966.12
UDP PLAIN	248.97	959.91
HTTP	2.22	11.04

性能をまとめたものが表 4 になる．この表から攻撃台数に比例して攻撃性能が推移していることが分かる．プレーン UDP フラッド攻撃においては，攻撃台数が 2 台の時は 462.43 Mbps，3 台の時は 749.74 Mbps と，攻撃台数に比例して攻撃性能が推移していたが，4 台の時は 959.74 Mbp であり，LAN ハブの回線の最大値（1 Gbps）が原因で，4 台目以降は測定できる攻撃性能が頭打ちになってしまったと考えられる．そのため，回線による制限を無視すれば，4 台以上の攻撃台数の場合も攻撃台数に比例した分のパケットが送ることが可能だと考えられ，実際のマルウェアの DDoS 攻撃の威力は，攻撃に使用される IoT 機器の性能と台数によると予測できる．

次に，1 台の組込みボードによる HTTP フラッド攻撃を，VM の Apache によるサイトに対して行った結果について説明する．攻撃性能の値としては 2.22 Mbps と他の攻撃方法と比較して小さいが，リクエスト 1 つに対して TCP プロトコルと HTTP プロトコルを合わせて，約 10500 バイト分のパケットのレスポンスがあり，測定した数値以上の負荷がサーバにかかっているとみられる．HTTP リクエストは大きく分けると，クライアントからの 3 ウェイハンドシェイクによる TCP 接続の確立，クライアントからの HTTP プロトコルの GET メソッドリクエスト，サーバからの HTTP プロトコルのレスポンス，サーバからの TCP 接続の切断の 4 段階に分けることができる．Apache に対する攻撃の流れとしては，攻撃開始時 TCP 接続が行われた後，HTTP プロトコルによる GET メソッドリクエストとレスポンスが繰り返され，攻撃終了時 TCP 接続が切断されていた．

実機における実験では，Mirai の攻撃の中で最大のものはプレーン UDP フラッド攻撃であり，その数値は 247.84 Mbps であり，攻撃ボットの台数に比例して攻撃性能が増加していることが確認された．この攻撃ボットが約 2500 台集まって攻撃すれば単純計算で 620 Gbps に達することになり，実際に 2016 年に観測された事例も可能だと考えられる．組込みボードの 1 Gbps の回線の転送量の最大値と比べると VM における実験同様に攻撃の転送速度は数十パーセントと小さい．実機での攻撃性能評価の実験においても，前章の VM における実験同様に，攻撃ボットの CPU 使用率が限界に達していたので，攻撃性能は Mirai の動作する実機の CPU の性能に依存している可能性が高い．このことからも IoT 機器の CPU 使用率を監視することでマルウェアの感染を発見することができると考えられる．なお，ubuntu には CPU 使用率を監視するウォッチドッグという機能があり，これは一定時間ごとにファイルに 1 バイトの書き込みを行い，その書き込みに異常があれば機器を再起動するというものである．Mirai は感染直後にこの機能を停止させるコードを送り無効化するため，マルウェアによって無効にされない対策が必要である．

3.5 静的な組込みシステムのプロトタイプに対する攻撃性能評価

この節では，静的な組込みシステムのプロトタイプ（図 5）に対する攻撃性能評価について述べる．このプロトタイプのボードは GR-PEACH であり，OS は TOPPERS/ASP（Release 1.9.2）であり，TCP/IP スタックは LwIP である．Apache のデフォルトページではなく、http サーバとしてサービスを提供しているサーバを攻撃し，その影響や攻撃の様子を確認した．また，その結果を Apache の場合と比較した．

このプロトタイプは，カメラに映っている映像を http サーバ上で提供する機能を持つ．これに対して UDP フラッド攻撃と HTTP フラッド攻撃を行い，そのパケットの様子を Wireshark で確認した．HTTP フラッド攻撃では，攻撃対象のプロトタイプの http サーバに合わせて，サーバ内に存在する URL や存在しない URL を指定して攻撃を行なった．

http サーバが動作する静的な組込みシステムのプロトタイプを攻撃対象にした場合では，攻撃ボットである組込みボードが 1 台で 10 秒間 UDP フラッド攻撃を行ったところ，攻撃を受ける 10 秒間程サイトが提供する映像が固まり，その後正常に映像を提供するという様子が確認できた．攻撃ボットを 5 台に増加しても影響は同様であった．このプロトタイプのイーサネットが 100 Mbps であるため，これを超える 127 Mbos の UDP 攻撃によってサービスが停止したとみられる．攻撃ボット 1 台による HTTP フラッド攻撃を行なった場合では，提供する映像が少し遅くなるという影響があった．攻撃台数を増加させるほど，映像の遅れは大きくなり，攻撃台数が 4 台になったところで，攻撃ボット 1 台による UDP フラッド攻撃と同様に，映像が 10 秒間程停止した．

存在する URL に対する攻撃において，HTTP フラッド攻撃を行う実機上から Wireshark で確認したところ，HTTP リクエストにおいて 3 ウェイハンドシェイクにより TCP

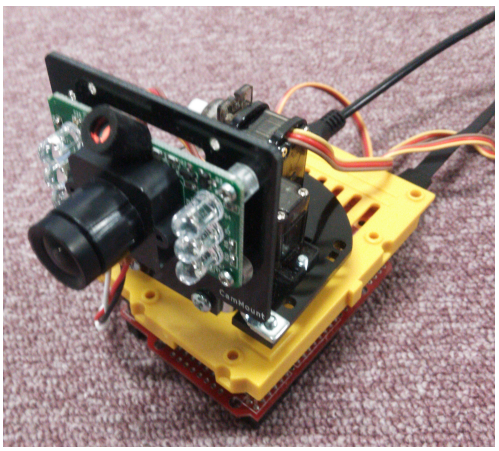

図 5 静的な組込みシステムのプロトタイプ

接続を確立して HTTP プロトコルによる GET メソッドリクエストを行う様子が確認できたが，http サーバがレスポンスを返した後，再び攻撃が行う GET メソッドリクエストに対してサーバ側が拒否をし，TCP 接続を切断して，再び攻撃側が TCP 接続を行うという様子が確認された．また，存在しない URL を使った HTTP リクエストによる攻撃を行なった時はエラーを返すレスポンスがあり，TCP 接続が切断されるという様子が確認された．

HTTP フラッド攻撃に関しては，攻撃対象が VM の Apache と静的な組込みシステムのプロトタイプで，それぞれ攻撃の流れが異なっていた．後者の場合，攻撃側は HTTP リクエストにおいて TCP 接続を行う処理が多く必要となり，攻撃としての HTTP リクエストのパケットの転送速度が低くなっているとみられる．これは，サーバの実装の差異によるものだと考えられ，攻撃対象によって，この攻撃方法が与える影響は変わると予想される．それでも，映像の処理の遅れの様子を鑑みると，約 4 台の攻撃ボットによる HTTP フラッド攻撃は，1 台の攻撃ボットによる UDP フラッド攻撃に相当するとみられ，TCP 接続を HTTP リクエスト数回ごとに行わなければならないものの，http サーバ側が HTTP リクエストに対してレスポンスを行う処理があることの負荷は確実に存在していて，パケット総数でみた攻撃性能よりは実際の攻撃の威力があると考えられる．

3.6 対策に関する議論

Mirai への対策としては，製品開発において，開発やメンテナンス用のための Telnet 接続用の 23 番ポートや 2323 番ポートを開放したままにしたり，認証に使用するユーザ名やパスワードが製品で同一で変更しなくても製品を使用できたりするといった脆弱性をなくすようにすることが挙げられる．また，Mirai は，ソースコードが公開されており，誰でもコンパイルし実行できるため，ユーザ名とパスワードの組み合わせに他のパターンを使用したり，ログインした機器を利用して情報を盗んだりするような Mirai の亜種が出現する可能性が考えられる．

すでに感染した IoT 機器また感染の可能性のある機器に関しては，Mirai がメモリ上で動作することから，再起動してパスワードを変更することで感染を防ぐことができる [6]．

インターネットから家庭のルータへつながる回線の最大値は 1 Gbps 程で，ルータはこの数値分の攻撃を受ける可能性があり，これは既存の脅威である．本研究の 3 章の 5 節で述べた Mirai による IoT 機器への攻撃の最大負荷は，その機器のイーサネットの性能に依存するため，DoS 攻撃による一般的な家庭の IoT 機器への最大負荷はおよそ 100 Mbps 程だと予想される．そのため，IoT 機器はこの数値の攻撃に耐えうる設計が必要となる．攻撃を受けると IoT 機器は，攻撃を受けている間サービスを停止するが攻撃がやめばサービスを提供を再開しシステムの破損はみられなかったため，IoT 機器への攻撃ではウォッチドッグによって対応が可能だと考えられる．また，攻撃に対して，ルータで監視したり，エッジで切り離したりするという対応も考えられ，この場合においてスタンドアローンでの安全性の確保が必要である．

4. おわりに

本研究では，マルウェア Mirai の概要やその攻撃性能について調査し評価した．マルウェア Mirai の概要の調査やその攻撃性能の評価では，VM や実機を用いて Mirai を動作させることでその攻撃の詳細について確認した．その結果，Mirai の攻撃が十分に脅威的であることが確認でき，IoT 機器におけるセキュリティの重要性がみられた．STOMP 攻撃や DNS 攻撃の動作確認や CPU 使用率監視の具体的な方法の検討，他のマルウェアや亜種の調査が今後の課題といえる．

謝辞 本研究の一部は JSPS 科研費 16K21097 の助成を受けて行われた．

参考文献

[1] 新井悠，岩村誠，川古谷裕平，青木一史，星澤裕二：アナライジング・マルウェア，オライリー・ジャパン（2010）．
[2] IPUSIRON：ハッカーの学校，株式会社データハウス（2015）．
[3] Adam Shostack：threat modeling designing for security（2014）．
[4] 日経 BP マーケティング：IoT セキュリティ〜インシデントから開発の実際まで〜，日経 BP イノベーション ICT 研究所編．日経 BP 社（2016）．
[5] IPA：IoT 開発におけるセキュリティ設計の手引き，IPA（2016）．
[6] 齋藤衛, IIJ Technial WEEK 2016 セキュリティ動向 2016. http://www.iij.ad.jp/company/development/tech/techweek/pdf/161111_01.pdf（accessed 2017/1/27）．
[7] 高田広章：IoT 時代のセーフティ＆セキュリティ．http://deos.or.jp/event/files/20160608_deos-1606.pdf（accessed 2017/1/16）．
[8] 松原豊：組込みシステムセキュリティ入門と最新動向．http://www.chubu.meti.go.jp/b34jyoho/shiryo/20151116securitykoshukai/security-workshop-talk-public.pdf（accessed 2017/1/16）．
[9] MMD-0056-2016 - Linux/Mirai, how an old ELF ... - Malware Must Die!.http://blog.malwaremustdie.org/2016/08/mmd-0056-2016-linuxmirai-just.html（accessed 2017/1/27）．
[10] jgamblin/Mirai-Source-Code - GitHub．入手先 https://github.com/jgamblin/Mirai-Source-Code（accessed 2016/10/27）．
[11] Go MySQL Driver is a MySQL driver for Go's (golang) database/sql package．入手先 https://github.com/go-sql-driver/mysql（accessed 2016/11/9）．
[12] Parse line as shell words．入手先 https://github.com/mattn/go-shellwords．（accessed 2016/11/9）．