「マルチメディア，分散，協調とモバイル
(DICOMO2019) シンポジウム」 令和元年7月

IoT機器を模したハニーポットの構築

山川 大貴[1]　上原 哲太郎[†1]

概要：脆弱な IoT 機器を乗っ取り，それを踏み台にしたサイバー攻撃の脅威が問題になっているが，我々は IoT 機器を模した仮想計算機に対する攻撃を観察できるハニーポットを運用することで攻撃手法についてデータの収集を行っている．本報告ではその構成法について提案している．また，収集したデータからログイン試行時に使用したユーザ名とパスワード，ログイン成功後に実行されたシェルコマンド，接続元ホストのドメイン情報について分析を行い報告する．分析結果より，様々な IoT 機器で初期設定されているユーザ名とパスワードのリストを用いた辞書攻撃が行われていることを確認できた．また，ハニーポットに接続してきた全ホストの IP アドレスのドメイン情報を取得した結果，特定の国からの通信が多いことが判明した．

Inplementation of a honeypot to simulate IoT devices

DAIKI YAMAKAWA[1]　TETSUTARO UEHARA[†1]

1. はじめに

近年，多くの IoT 製品が人々の生活の中に普及しているが，その一方で，IoT 機器の所有者が必ず情報セキュリティの知識を有しているとは限らず，管理されていない IoT 機器は攻撃者によって侵入され，さらなる攻撃の踏み台にされている．例えば，2016 年 9 月に米国のセキュリティ情報を発信しているブログ “Krebs on Security” が大規模な DDoS 攻撃を受けてダウンした．事故後の分析により，攻撃発生時の最大通信量は 620Gbps(Gigabits per secound) を超えることが判明した．セキュリティ事故の原因は Mirai と呼ばれるマルウェアに感染した IoT 機器によって構築されたボットネットからの通信だった．[2] Mirai は Telnet(23/TCP) が動作している IoT 機器に対して様々な IoT 機器で初期設定されているユーザ名とパスワードを使用しての辞書攻撃を行う．[3] つまり，Mirai を始めとした IoT マルウェアの標的はユーザ名とパスワードが初期設定のまま運用されている IoT 機器であると言える．したがって，先ほどの事例で紹介した IoT 機器がマルウェアによって DDoS 攻撃の踏み台にされている現状を改善する必要がある．その改善策を考えるには IoT 機器がどのような攻撃で踏み台にされているのか現状を理解することが重要である．そのため，本報告では IoT 機器を模したハニーポットの構築方法を提案し，実際に開発・運用することで IoT 機器に対する攻撃の観測データを収集し分析する．

[1] 立命館大学 情報理工学研究科
[†1] 現在，立命館大学 情報理工学部

2. 研究背景

2.1 IoT 機器の現状

総務省の平成 30 年度版情報通信白書 [1] では，2020 年に世界で 403 億台の IoT 機器が普及すると予測している．IoT 製品の例としては，スマートフォンのアプリを利用して遠隔地から家電の稼動状態の確認や電源を操作できる “スマート家電” が開発され，消費者向けに販売されていることが挙げられる．このように，今までインターネットに繋がっていなかった様々なものをインターネットを介して繋げることで IoT 機器は利用者に価値を提供している．しかし，IoT 機器は最新ファームウェアのアップデートやデフォルトパスワードの変更などを行わなければ，攻撃者の標的となってしまうことが知られている．現在，NICT が NICTER プロジェクトの中で大規模なダークネット観測を行っている．NICTER 観測レポート 2017[4] では，宛先ポート番号別のパケット数割合で過半数を占めているのが IoT 機器で使用されているポート番号であることを示して

おり，IoT 機器を狙った攻撃は多く存在していると推測されている．実際に脆弱な IoT 機器は攻撃者に侵入され，さらなる攻撃の踏み台にされた後，第三者に被害を与えている．[2] 原因は脆弱な IoT 機器を運用していることと所有者が自身の IoT 機器のマルウェア感染に気づいていないことであり，このような問題を改善することが必要である．

2.2 本研究で構築するハニーポットが観測する攻撃モデル

IoT 機器を模したハニーポットを提案するために，どのような攻撃をハニーポットで観測するか決定する必要がある．NICTER 観測レポート 2017[4] より，多くの IoT 機器は telnet(23/TCP)，ssh(22/TCP) を用いて攻撃されていることがわかっているため，開発するハニーポットで観測する通信プロトコルは telnet(23/TCP)，ssh(22/TCP) とした．また，今回は IPA の資料を参考にして攻撃モデルを決定した．[5] 本研究で想定している攻撃モデルを図 1 に示す．まず，攻撃者は 22/TCP や 23/TCP ポートに対してネットワークスキャンを行い，telnet/ssh が動作している IoT 機器を探索する．その後，telnet/ssh が動作している IoT 機器に対して，ログイン試行する (図 1-1)．その際に工場出荷状態の IoT 機器に設定されているデフォルトのユーザ名/パスワードや推測しやすいユーザ名/パスワードを組み合わせた辞書を用いた辞書攻撃を行う．ログインに成功すると，telnet/ssh 経由でシェルコマンドを実行して，マルウェアをダウンロードする環境を整える (図 1-2)．そして，外部のサーバからマルウェアの本体もしくはシェルスクリプトをダウンロードする (図 1-3)．このようにしてダウンロードされたシェルスクリプトにはマルウェアの本体をダウンロード，実行権限の付与，実行するコマンドなどが記述されており，シェルスクリプトを実行することでマルウェアをダウンロードして実行できる．IoT 機器に感染したマルウェアは C&C サーバからの命令を受けて (図 1-4)，他の脆弱な IoT 機器を探すためにネットワークスキャンを行う (図 1-5)．さらに，C&C サーバからの命令により，標的のサーバに対して DoS 攻撃など様々な攻撃を行うことが考えられる (図 1-6)．

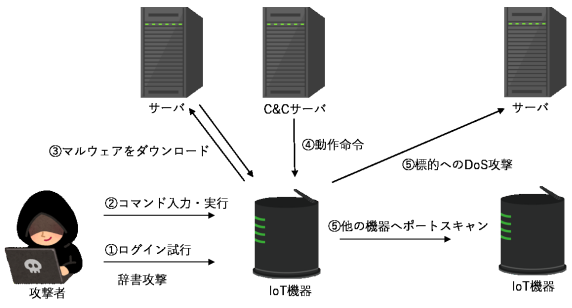

図 1 IoT 機器を狙った攻撃の一例

さらに，感染したマルウェアによっては自身が感染した IoT 機器を他のマルウェアに乗っ取られないように，狙われやすいポートを閉じるなどの対策を講じるものも存在する．そういった挙動をするマルウェアの一つに Hajime があげられる．Hajime[16] は Mirai と同様にランダムに選択したアドレスのポート 23/TCP(telnet) に対してスキャンを行い，様々な IoT 機器においてデフォルトで用いられているユーザ名とパスワードのリストを使ってログイン試行を行う．ログイン成功後，23/TCP などのマルウェアに狙われやすいポートへの通信を遮断するため，Mirai をはじめとする他のマルウェアに感染する可能性は減少する．Hajime は IoT 機器に感染後，“Just a white hat, securing some systems. Important messages will be signed like this! Hajme Author. Contact CLOSED Stay sharp!” というメッセージを約 10 分ごとにターミナル上に表示することから，IoT 機器の保護を掲げていることがわかる．現時点では感染活動のみ行っていて，Hajime が構築したボットネットによる DDoS 攻撃は確認されていない．もし，ハニーポットで Hajime のようなマルウェアが活動した場合，攻撃者をおびき寄せるために開けていたポートを閉じられてしまうため，次の新しい攻撃を観測することができない．そのため，定期的にハニーポットをクリーンな状態にすることが必要である．

2.3 関連研究

中山ら [6] は IoT 機器を模したハニーポットを運用して得られたデータから，ログイン試行時に使用される ID/パスワード情報とログイン成功後に使用されるシェルコマンドの分析を行っている．さらに，ハニーポットを運用することで収集されたマルウェア検体を独自の解析環境で動的解析し，サンドボックス内でマルウェアによるログイン試行と実行するシェルコマンドを確認している．この研究では攻撃者が用いる ID/パスワードのセットリストとログイン成功後の挙動を観測するために，すべてのログイン試行を拒否するハニーポットとすべてのログイン試行を許可するハニーポットの 2 種類を運用している．また，ハニーポットで観測している通信プロトコルは telnet である．佐藤ら [7] は既存のハニーポットである Honeyd と Kippo を組み合わせたシステムを開発し，ssh プロトコルを使った通信を観測している．その後，観測したデータの解析を行い，特にログイン試行時に使われたユーザ名とパスワードについて分析している．Honeyd[8] は仮想ホストをエミュレートし，侵入者に関する情報を収集できるオープンソースの低対話型ハニーポットである．Kippo[9] は ssh サービスのエミュレートに特化したハニーポットであり，ログイン試行時に用いられたユーザ名とパスワード情報，ログイン成功後のコマンド入力を記録できる．

本研究では，telnet プロトコルだけではなく ssh プロトコルも同時に観測可能な IoT 機器を模したハニーポットの

開発を行う．また，ssh プロトコルの通信を観測するために，佐藤らの研究で使われている Kippo をベースにして開発された HonSSH[10] を用いた．Kippo は Kippo 自身がハニーポットとして動作するため，ハニーポットで動作する OS や CPU アーキテクチャを選択することができない．しかし，本研究で開発するハニーポットは IoT 機器を模倣する必要がある．HonSSH は ssh ゲートウェイとして動作することを目的として開発されたソフトウェアであるため，攻撃者とハニーポットの間に設置することが出来，任意のアーキテクチャのハニーポットを容易に設置することができる．結果として，同一の IP アドレスに対し telnet と ssh で同一のシステムが応答するため，外部からスキャンを仕掛ける攻撃者にとってハニーポットであると気づかれる可能性が低くなると期待できる．

3. 設計

本章では，想定している攻撃モデルとハニーポットを開発して取得するデータを示すことで，開発するハニーポットのシステム要件を明らかにする．ハニーポットで観測する攻撃モデルは図 1 であり，telnet/ssh プロトコルの通信を同時に取得可能なハニーポットを開発する．また，開発したハニーポットを運用して取得するデータはログイン試行時に使用されたユーザ名とパスワードのリスト，パケットキャプチャファイル，コマンド実行ログとする．

3.1 IoT 機器を模したハニーポットの提案

ハニーポットでは，攻撃者から IoT 機器に見えるように組み込み向けのアーキテクチャを使って開発する必要がある．しかし，実機を使用してハニーポットを構築すると，マルウェア感染した際にハニーポットを感染前の状態に戻す手間がかかってしまう．そこで，ハニーポットを容易に復元できるように QEMU[11] を利用して開発することにした．QEMU とはオープンソースのマシンエミュレータであり，様々な種類の CPU アーキテクチャをエミュレートできる．これにより，本研究で開発するハニーポットはさまざまな IoT 機器を容易に模倣できる．ハニーポットに対する通信は，telnet に関しては通信路上でそのデータをキャプチャすることで記録し，観察できる．暗号化される ssh については，HonSSH を用いて一度通信の途中で通信を復号して記録し，再度暗号化することでハニーポットに対して透過的な記録を実現している．このようにして，通信データの記録機構に手を加えずとも QEMU 上で任意の CPU アーキテクチャと OS を模倣させることで容易に多種多様な IoT 機器を模したハニーポットを実現することができる．

本研究で開発したハニーポットのシステム構成を図 2 に示す．外部から各 VM に対して telnet または ssh プロトコルを用いた通信が来た場合，その通信は iptables によって QEMU 上のハニーポットにフォワーディングされるように設定している．また，telnet/ssh プロトコルを用いた通信を観測するために tshark と HonSSH を用いた．tshark はネットワークプロトコルアナライザ Wireshark の CUI 版であり，telnet プロトコルを用いた通信を含むすべての通信データを取得するために使用した．ssh プロトコルを用いた通信内容は暗号化されているため，本研究では各 VM のネットワークインタフェース (eth0) とハニーポットの間に ssh ゲートウェイである HonSSH を立てることで，ssh プロトコルでやり取りされる攻撃者とハニーポット間の暗号化された通信内容の取得を実現する．

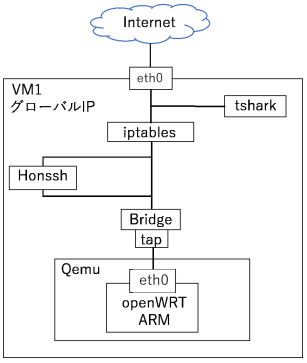

図 2 ハニーポットのシステム構成

次に開発したハニーポットの全体図を図 3 に示す．

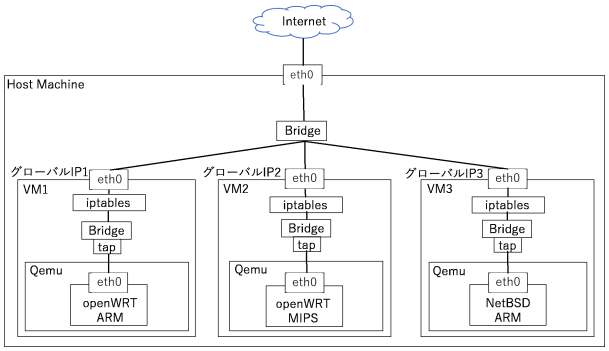

図 3 ハニーポットの全体図

今回は 3 つのハニーポット運用して通信データを取得することにした．また，各ハニーポットに採用した OS・CPU アーキテクチャの組み合わせと対応するハニーポット名を以下に示す．

- honeypot1: OpenWRT/ARM
- honeypot2: OpenWRT/MIPS
- honeypot3: NetBSD/ARM

今回，ハニーポットに採用する CPU アーキテクチャと OS は IoT 機器に広く搭載されており，かつ攻撃者の標的になりやすいものを選択した．OpenWRT[12] は組み込み機器を対象とする Linux ディストリビューションであり，主にルータの OS として利用されている．現在，ルータは

多くの人のインターネット利用において重要な役割を果たしているが，ルータのファームウェアを更新していない脆弱な機器も存在している．攻撃者は容易に利用者の通信内容やルータに接続しているデバイスの情報などを得ることができるため，このような脆弱性が知られた機器が標的になりやすいと考えられる．[13] NetBSD[14] を選択した理由は BSD ライセンスであることと多くのアーキテクチャ上で動作するため，組み込み機器で広く使われているためである．CPU アーキテクチャは組み込み機器向けのアーキテクチャである ARM と MIPS を選択した．したがって，QEMU を利用して ARM と MIPS の CPU アーキテクチャをエミュレートし，選択した OS を動作させることで IoT 機器を模倣した．また，各ハニーポットで動作させるサービスは telnet(23/TCP)，ssh(22/TCP) とする．攻撃者は telnet と ssh サービスを利用する際にユーザ名とパスワードを入力するが，あえて攻撃者に推測されやすいユーザ名とパスワードとして，root/root を設定した．QEMU を動作させるホストマシンには IP アドレスを割り当てていないので，外部からネットワーク越しに直接アクセスすることが困難であり，万一にもシステム全体が乗っ取られないようにできる．ホストマシンのネットワークインターフェース (eth0) をプロミスキャスモードにして，ハニーポット宛ての通信を全て通過させる．そして，ハニーポットが動作する各 VM にグローバル IP アドレスを割り当てることで，ネットワークの外部からはハニーポットだけが見えるようにした．

3.2 マルウェア感染時の対応

ハニーポットはわざと脆弱な機器を攻撃者から見えるように設置することで，攻撃者の挙動を観測することを目的としている．そのため，ハニーポットを運用し始めると，攻撃者が侵入を試みて成功した際にマルウェアをダウンロード・実行することが考えられる．第 1 章，2 章で述べたとおり，IoT 機器に感染するマルウェアは感染後，IoT 機器をボット化し外部のサーバなどに DoS 攻撃などを行う．そのため，ハニーポットを設置することにより，第三者のサーバに対しての DDoS 攻撃に加担しないような対策を考える必要がある．本研究では iptables の設定でハニーポットからネットワーク外部に対しての通信量を制限することで DDoS 攻撃に加担する可能性を低減させている．また，2 章で述べたようにハニーポットをマルウェア感染した状態で放置すると次の新しい攻撃を観測できない．そこで，本研究では 24 時間ごとに QEMU を再起動することでハニーポットを定期的にクリーンな状態に戻し，次の攻撃を観測する．

4. 提案したハニーポットの運用

本章では，3 章で提案したハニーポットを運用して得られたデータを示す．また，ハニーポットに対して telnet，ssh プロトコルを用いて接続を試みた外部ホストの国別割合を各プロトコルごとに示す．その後，取得したデータからログイン試行時に使用されるユーザ名，パスワード情報とログイン成功時に使用されたシェルコマンドの分析を行った．

4.1 収集したデータの概要

収集したデータの概要を図 4 に示す．また，図 4 の列ラベルについて下記に説明する．

- honeypot_name: ハニーポット名
- all_packet_count: 総パケット数
- telnet_packet: telnet プロトコルを用いたパケット数
- ssh_packet: ssh プロトコルを用いたパケット数
- time: ハニーポットでの観測時間

図 4 より，各ハニーポットに届いた通信を通信プロトコルごとに見てみると，honeypot3 は telnet プロトコルを用いた通信が他のハニーポットに比べて極端に少ないことがわかった．honeypot3 に届いた通信を詳細に調査すると，すべてログイン試行時にユーザ名を入力するフォームが出力される前に入力している通信であった．そこで，実際にローカル環境にハニーポットを立てて，telnet 接続を試みるとユーザ名の入力フォームが表示されるまでの時間が OpenWRT より NetBSD の方が遅いことが判明した．このことから，NetBSD と判明した時点で接続を止めているか，入力フォームが表示されるまでの時間が長いことから接続するのを止めていることが考えられる．また，入力フォームが出力される前にユーザ名を入力している通信データを確認できたことから，送信元ホストは機械的に接続していることがわかる．

honeypot_name	all_packet_count	telnet_packet	ssh_packet	time
honeypot1	404271	11099	173730	303:53:43
honeypot2	484698	35281	184529	304:47:43
honeypot3	390734	735	178068	304:18:07

図 4 収集したデータの概要

4.2 telnet プロトコルを用いた通信

telnet プロトコルを用いてハニーポットに接続してきた送信元 IP アドレスの総数は 1114 個であり，総 IP アドレスの国別割合を図 5 に示す．図 5 より，中国からの通信が最も大きな割合を占めていることが分かった．

ログイン試行時に使用されたユーザ名とパスワードのうち，上位 10 個のリストを図 6 に示す．ハニーポットにあらかじめ設定したユーザ名とパスワードが root/root であるため，図 6 において最上位に位置している．上位に位置している他のユーザ名を見てみると，管理者を意味する administrator の略称である “admin”やゲストアカウント

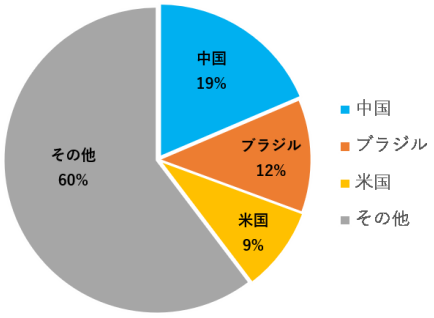

図 5　送信元 IP アドレスの国別割合

を意味する “guest” といった単語の入力が多いことがわかる．“admin” や “guest” といったユーザ名は様々な IoT 機器の初期状態で設定されていることが多いユーザ名である．他にも，“888888” が KEIAN 社の IP カメラの初期パスワードとして設定されていることを確認している．[15]

ranking	user_name	input_count
1	root	1839
2	enable	723
3	admin	381
4	default	99
5	shell	49
6	guest	42
7	support	42
8	telnetadmin	34
9	user	31
10	telecomadmin	23

ranking	password	input_count
1	root	599
2	888888	315
3	system	276
4	admin	256
5	0	253
6		107
7	sh	46
8	linuxshell	46
9	default	44
10	tsgoingon	35

図 6　ユーザ名とパスワードのリスト

honeypot2 では 2018 年 12 月 13 日以降から 1 つの IP アドレスからの通信が増加した．該当 IP アドレスと honeypot2 の間で行われた通信内容のうち特徴的なものを図 7，8，9 に示す．通信内容が長かったため，3 つに分割して順に表示している．図 7 で入力されているコマンドについて説明する．

(1) sh コマンドを用いてシェルを起動

(2) busybox を使って “ECCHI” と名付けられたプログラムを実行しようとしているが存在しないためエラー文が出力されている．

busybox は UNIX 系ユーティリティツールを一つの小さな実行ファイルにまとめたものである．組み込み機器向けに開発されているので，組み込み機器にとって不必要なオプションを省いた様々なコマンドが格納されている．そのため，各コマンドごとにインストールするよりもファイルサイズを小さくできる利点があり，限られた資源しか使えない組み込み機器において広く利用されている．

```
root@OpenWrt:/# .........P.........root
root@OpenWrt:/# .[Jroot00000000
00000000enable
shell
sh

/bin/ash: root00000000: not found
root@OpenWrt:/# enable
/bin/ash: enable: not found
root@OpenWrt:/# /bin/busybox ECCHI
shell
/bin/ash: shell: not found
root@OpenWrt:/# sh


BusyBox v1.23.2 (2017-08-29 19:48:18 JST) built-in shell (ash)

root@OpenWrt:/# /bin/busybox ECCHI
ECCHI: applet not found
root@OpenWrt:/# /bin/busybox ps; /bin/busybox ECCHI
/bin/busybox ps; /bin/busybox ECCHI
  PID USER       VSZ STAT COMMAND
    1 root      1300 S    /sbin/procd
    2 root         0 SW   [kthreadd]
    3 root         0 SW   [ksoftirqd/0]
    4 root         0 SW   [kworker/0:0]
    5 root         0 SW<  [kworker/0:0H]
    7 root         0 SW   [rcu_sched]
    8 root         0 SW   [rcu_bh]
    9 root         0 SW   [migration/0]
   10 root         0 SW<  [khelper]
   11 root         0 SW   [kworker/u2:1]
   25 root         0 SW<  [writeback]
   32 root         0 SW<  [bioset]
   33 root         0 SW   [kworker/0:1]
   34 root         0 SW<  [kblockd]
  100 root         0 SW   [kswapd0]
  103 root         0 SW   [fsnotify_mark]
  135 root         0 SW<  [deferwq]
  137 root         0 SW   [kworker/u2:2]
  266 root       784 S    /sbin/ubusd
  277 root       664 S    /sbin/askfirst /bin/ash --login
  442 root       936 S    /sbin/logd -S 16
  461 root      1384 S    /sbin/netifd
  532 root      1296 S    /usr/sbin/telnetd -F -l /bin/login.sh
  558 root      1296 S    /usr/sbin/ntpd -n -S /usr/sbin/ntpd-hotplug -p 0.ope
  583 root      1032 S    /usr/sbin/dropbear -F -P /var/run/dropbear.1.pid -p
  757 root      1300 S    /bin/ash --login
  765 root      1296 S    sh
  767 root      1296 R    /bin/busybox ps
ECCHI: applet not found
```

図 7　特徴的なログ (1)

次に，図 8 で入力されているコマンドについて説明する．“.nippon” という文字列を使用していることから，攻撃者は自身の接続先ホストが日本に存在していることを認識していると推測できる．

(1) busybox で cat プログラムを実行し，cat プログラムの引数に “/proc/mounts” を入力することで入力時に使用している全マウントの一覧を表示している．

(2) busybox で echo プログラムを実行し，引数に出力したいディレクトリを指定した後，出力結果を “.nippon” と名付けられた隠しファイルに書き込んでいる．

(3) busybox で cat プログラムを実行し，cat プログラムの引数に “.nippon” を入力することで，“.nippon” の内容を表示している．

(4) busybox で rm プログラムを実行し，作成した “.nippon” を削除している．

攻撃者が “.nippon” に書き込もうとしているディレクトリに書かれている情報について説明する．

- /proc ディレクトリ配下：システムハードウェアや現在実行中のプロセスについての情報
- /sys ディレクトリ配下：デバイスドライバ関連の情報
- /tmp ディレクトリ配下：一時ファイルを格納しているディレクトリ
- /dev ディレクトリ配下：デバイスファイルが格納されている

- /dev/pts ディレクトリ配下：telnet や ssh で接続しているターミナルを示す
- /sys/kernel/debug ディレクトリ配下：debugfs というファイルシステムがマウントされている場所

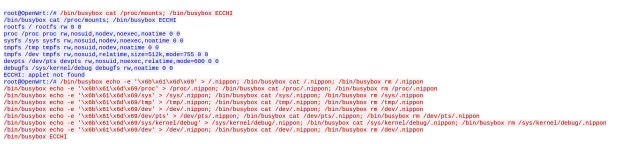

図 8　特徴的なログ (2)

最後に，図 9 で入力されているコマンドについて説明する．

(1) rm コマンドで “.t” “.sh” “.human” というファイルが削除されている
(2) busybox で cp プログラムを実行し，引数に “/bin/echo dvrHelper” と入力することで，該当ファイルをコピーしてコマンド実行時のディレクトリにある “dvrHelper” というファイルに書き込んでいる
(3) busybox で chmod プログラムを実行し，“dvrHelper” に実行権限を与えている．
(4) busybox で cat プログラムを実行し，echo の実行プログラムを表示している．

dvrHelper は Mirai の亜種で DDoS 対策を回避する機能を持っているマルウェアである．[17] 図 7，8，9 を通して，攻撃者の目的はハニーポットのシステム情報を取得して “ECCHI”，“dvrHelper” を実行することであったと考えられる．また，busybox を使って “ECCHI” を実行しようとする挙動は Hajime が IoT 機器に侵入した際の挙動と類似している．[16]

```
root@OpenWrt:/# rm /.t; rm /.sh; rm /.human
rm /tmp/.t; rm /tmp/.sh; rm /tmp/.human
rm /dev/.t; rm /dev/.sh; rm /dev/.human
rm /dev/.t; rm /dev/.sh; rm /dev/.human
cd /
/bin/busybox cp /bin/echo dvrHelper; >dvrHelper; /bin/busybox chmod 777 dvrHelper; /bin/busybox ECCHI
rm /.t; rm /.sh; rm /.human
rm: can't remove '/.t': No such file or directory
rm: can't remove '/.sh': No such file or directory
rm: can't remove '/.human': No such file or directory
root@OpenWrt:/# rm /tmp/.t; rm /tmp/.sh; rm /tmp/.human
rm: can't remove '/tmp/.t': No such file or directory
rm: can't remove '/tmp/.sh': No such file or directory
rm: can't remove '/tmp/.human': No such file or directory
root@OpenWrt:/# rm /dev/.t; rm /dev/.sh; rm /dev/.human
rm: can't remove '/dev/.t': No such file or directory
rm: can't remove '/dev/.sh': No such file or directory
rm: can't remove '/dev/.human': No such file or directory
root@OpenWrt:/# rm /dev/.t; rm /dev/.sh; rm /dev/.human
rm: can't remove '/dev/.t': No such file or directory
rm: can't remove '/dev/.sh': No such file or directory
rm: can't remove '/dev/.human': No such file or directory
root@OpenWrt:/# cd /
root@OpenWrt:/# /bin/busybox cp /bin/echo dvrHelper; >dvrHelper; /bin/busybox ch

mod 777 dvrHelper; /bin/busybox ECCHI
ECCHI: applet not found
root@OpenWrt:/# /bin/busybox cat /bin/echo
/bin/busybox ECCHI
/bin/busybox cat /bin/echo
```

図 9　特徴的なログ (3)

4.3　ssh プロトコルを用いた通信

ssh プロトコルを用いてハニーポットに接続してきた送信元 IP アドレスの総数は 1430 個であり，総 IP アドレスの国別割合を図 10 に示す．図 10 より，中国からの通信が最も大きな割合を占めていることが分かった．

ログイン試行時に使用されたユーザ名とパスワードの

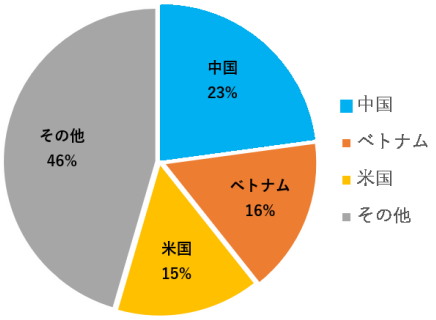

図 10　送信元 IP アドレスの国別割合

うち，上位 10 個のリストを図 11 に示す．ハニーポットにあらかじめ設定したユーザ名とパスワードが root/root であるため，図 11 において最上位に位置している．上位に位置している他のユーザ名とパスワードを見てみると，“ubnt” は Ubiquiti Networks 社 [18] が製造している製品に初期状態で設定されているユーザ名とパスワード [19] であり，“pi” は Rasberry pi 向け OS である Raspbian の初期ユーザ名であることを確認している．

ranking	user_name	input_count
1	root	26933
2	admin	2329
3	user	302
4	oracle	294
5	test	201
6	ubnt	187
7	pi	171
8	support	152
9	usuario	137
10	git	135

ranking	password	input_count
1	root	12771
2	admin	518
3	123456	480
4	password	380
5	1234	350
6	12345	260
7	ubnt	249
8	default	175
9	111111	174
10	support	160

図 11　ユーザ名とパスワードのリスト

5.　考察

本研究で提案したハニーポットを運用したことにより，IoT 機器に対する攻撃を観測することができた．Mirai や Hajime といった IoT 機器に感染するマルウェアはログイン試行時に辞書攻撃を行う．そこで，telnet/ssh のログイン試行時に使用された単語のリストを見てみると，Mirai や Hajime が保持している辞書に登録されている単語が多

く含まれており，それらの単語は様々なIoT製品で初期設定されているユーザ名とパスワードであった．つまり，実際に初期設定のまま運用されているIoT機器が多いので，攻撃者は管理されていないIoT機器を標的にし，初期設定されているユーザ名とパスワードを用いた辞書攻撃が多いのではないかと推測できる．また，IoTマルウェアに踏み台にされた機器がハニーポットに攻撃しているということは機器の所有者がマルウェア感染に気づいていないと考えられる．本来，ユーザは自身のIoT機器がマルウェア感染した場合，第三者に対して攻撃が行われる前にマルウェアを駆除すべきである．しかし，ユーザが必ずセキュリティの知識を持っているとは限らず，IoT機器がインターネットに接続されていることで攻撃を受ける可能性を認識していない可能性がある．そのため，自身が所有しているIoT機器が明確に異常な動作をしない限りマルウェア感染に気づくことができないため，IoTマルウェアの感染を早期検知できるようなシステムが必要である．最後に，ハニーポットに対して様々なIoT機器で初期設定されたユーザ名とパスワードを用いたログイン試行が多かったことから，ユーザは所有している機器に初期設定されたユーザ名とパスワードを変更することでマルウェア感染の確率を軽減できると考える．

6. まとめと今後の研究

本研究では，脆弱なIoT機器を踏み台にして攻撃を行った事例を挙げ，IoT機器の踏み台対策を考える必要があることを示した．しかし，対策を考える際にIoT機器が実際にどのように攻撃されているのかを理解する必要があるので，IoT機器を模したハニーポットを開発した．開発するハニーポットは攻撃者から一般的なIoT機器に見せる必要があるため，IoT機器で広く採用されており，かつ攻撃者の標的となりやすいOSとCPUアーキテクチャの組み合わせを選定した．ハニーポットは攻撃者に侵入されやすいように設定を行うのでマルウェア感染することが考えられ，実機を用いてハニーポットを作成すると感染前の状態に戻す手間がかかる．そのため，エミュレータを用いることで簡単に構築することができ，再利用しやすいハニーポットを提案した．そして，組み込み向けのCPUアーキテクチャをエミュレートし，そのアーキテクチャ上で選定したOSを動かすことでハニーポットを実現した．通信データの取得にはtsharkとHonSSHを用いて，telnetとsshプロトコルの通信データを同時に取得できるようにした．その結果，パケットキャプチャファイル，ログイン試行時に使用されたユーザ名とパスワード，コマンド実行ログを取得することができた．取得したデータを分析した結果，様々なIoT機器の初期設定時に使用されているユーザ名とパスワードを用いてログイン試行が行われていたことが判明した．また，ログイン試行時に使用されたユーザ名とパスワード，コマンド実行ログからIoTマルウェアに踏み台にされたと推測される機器からの通信があることを確認した．さらに，ハニーポットに接続してきた全IPアドレスのドメイン情報を調査した結果，telnet/sshプロトコルともに中国から接続している機器が最も多いことがわかった．今後はハニーポットで動作させるOSの種類を増やすことで多様なデータの取得，攻撃者のコマンド実行ログを再現してマルウェア検体の取得を試みる．

参考文献

[1] 総務省:平成30年度版情報通信白書，総務省 (オンライン) 入手先 ⟨http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h30/pdf/n1100000.pdf⟩ （参照 2019-05-07）.

[2] US-CERT: Alert(TA16-288A) Heightened DDoS Threat Posed by Mirai and Other Botnets, US-CERT(Online) 入手先 ⟨https://www.us-cert.gov/ncas/alerts/TA16-288A⟩ （参照 2019-05-07）.

[3] Github: Mirai-Source-Code，Github(オンライン) 入手先 ⟨https://github.com/jgamblin/Mirai-Source-Code⟩ （参照 2019-05-07）.

[4] 国立研究開発法人情報通信研究機構: プレスリリース NICTER 観測レポート 2017 の公開，国立研究開発法人情報通信研究機構 (オンライン) 入手先 ⟨https://www.nict.go.jp/press/2018/02/27-1.html⟩ （参照 2019-05-07）.

[5] 独立行政法人情報処理推進機構 (IPA): 顕在化したIoTのセキュリティ脅威とその対策，独立行政法人情報処理推進機構 (オンライン) 入手先 ⟨https://www.ipa.go.jp/files/000062277.pdf⟩ (参照 2019-05-07）.

[6] 情報処理学会: 中山 楓ほか，IoT機器へのTelnetを用いたサイバー攻撃の分析，情報処理学会論文誌 Vol.58 No.9，pp.1399-1409(Sep.2017) （参照 2019-05-07）.

[7] 情報処理学会: 佐藤 聡ほか，筑波大学におけるハニーポットを用いた不適切なSSHアクセスの収集とその解析，情報処理学会研究報告，2014-IoT-25(17)，pp.1-6，2014（参照 2019-05-07）.

[8] Developments of the Honeyd Virtual Honeypot 入手先 ⟨http://www.honeyd.org/⟩ （参照 2019-05-07）.

[9] Github: Kippo - SSH Honeypot，Github(Online) 入手先 ⟨https://github.com/desaster/kippo⟩ (参照 2019-05-07）.

[10] Github: HonSSH，Github(Online) 入手先 ⟨https://github.com/tnich/honssh⟩ （参照 2019-05-07）.

[11] QEMU: QEMU the FAST!,processor emulator, QEMU(Online) 入手先 ⟨https://www.qemu.org/⟩ （参照 2019-05-07）.

[12] OpenWRT: OpenWrt Wireless Freedom，OpenWRT(オンライン) 入手先 ⟨https://openwrt.org/⟩ (参照 2019-05-07）.

[13] McAfee Blog: 新たなマルウェア VPNFilter，50万台以上のルータに感染，McAfee Blog(オンライン) 入手先 ⟨https://blogs.mcafee.jp/new-vpnfilter-malware-infects-routers⟩ （参照 2019-05-09）.

[14] NetBSD: NetBSD The NetBSD Project，NetBSD(オンライン) 入手先 ⟨https://www.netbsd.org/⟩ （参照 2019-05-07）.

[15] 恵安株式会社: 取扱説明書 有線/無線 LAN 対応ネットワークカメラ C7823WIP Series，恵安株式会社 (オンライン) 入手先 ⟨https://www.keian.co.jp/img/2015/09/c7823wip_manual.pdf⟩ （参照 2019-05-08）

[16] RapidityNetworks: Hajime:Analysis of a decentralized internet worm for IoT devices，RapidityNetworks(Online) 入手先 ⟨https://security.rapiditynetworks.com/publications/2016-10-16/hajime.pdf⟩（参照 2019-05-07）.
[17] TRENDMICRO: トレンドマイクロセキュリティブログ，ネットワークカメラをボット化する「PERSIRAI」拡散と追随するその他のマルウェア，TRENDMICRO(オンライン) 入手先 ⟨https://blog.trendmicro.co.jp/archives/15201⟩（参照 2019-05-09）.
[18] Ubiquiti Networks，Ubiquiti Networks(Online) 入手先 ⟨https://www.ui.com/⟩（参照 2019-05-07）.
[19] Ubiquiti Networks:UniFi User Guide,Ubiquiti Networks (Online) 入手先 ⟨https://dl.ubnt.com/guides/UniFi/UniFi_Controller_V4_UG.pdf⟩（参照 2019-05-08）.