THE IEICE TRANSACTIONS ON COMMUNICATIONS (JAPANESE EDITION)

電子情報通信学会 B 論文誌

通　信

VOL. J102-B NO. 8
AUGUST 2019

通信ソサイエティ

一般社団法人 電子情報通信学会

THE COMMUNICATIONS SOCIETY

THE INSTITUTE OF ELECTRONICS, INFORMATION AND COMMUNICATION ENGINEERS

感染持続型 IoT マルウェアの実態調査と実機による概念実証

原　悟史[†,††a)]　田宮　和樹[†]　鉄　穎[†]　渡辺　露文[†,††]
吉岡　克成[†††]　松本　勉[†††]

Persistent IoT Malware: Real-World Observations and Proof-of-Concept with Real Devices

Satoshi HARA[†,††a)], Kazuki TAMIYA[†], Ying TIE[†], Tsuyufumi WATANABE[†,††], Katsunari YOSHIOKA[†††], and Tsutomu MATSUMOTO[†††]

あらまし　Mirai に代表される，組込み機器を攻撃対象とした IoT マルウェアのほとんどは，感染した機器の電源を再起動することで消滅することが知られている．しかしながら，特定の機器を狙った，電源の再起動のみでは消えない “持続的感染” を引き起こす IoT マルウェアの事例も報告されている．このようなマルウェアが流行した場合，被害の深刻化が容易に想定される．本研究では，まず IoT 機器の実機とマルウェア検体を用いて持続的感染の有無の調査を行った．7 種類の IoT 機器と計 45 の IoT マルウェア検体による実験の結果，いずれの機器，マルウェア検体についても持続的感染は確認されなかった．しかし，一つの機器においては，“システム起動スクリプト改変攻撃” により，容易に持続的感染に至る危険な状態であることが分かった．他の二つの機器においても，“ファームウェア攻撃” による，IoT マルウェアの持続的感染が成立することを実証した．これらの実証実験をもとに，持続的感染が成立する要因を分析し，持続的感染防止に効果的なファームウェア構成について議論を行う．

キーワード　組込み機器，IoT，マルウェア，持続的感染

1. まえがき

近年，世界中の様々なモノがインターネットに接続されるようになり，このような状態はモノのインターネット (IoT) と称されている．インターネットに接続する組込み機器は増加の一途をたどり，2015 年の 49 億台から，2020 年には 200 億台超と急増が予測されている [1]．一方で，組込み機器を狙った攻撃は，機器の増加のペースを上回る毎年 4〜5 倍のペースで増加 [2] しており対策が急務である．従来，組込み機器は，機器ごとに専用の OS やプログラムが搭載されることが多かったが，近年は汎用コンピューターと同様に，Linux [3] をはじめとするオープンソースソフトウェア（以下，OSS）を活用する事例が増えている．組込み機器のファームウェアを収集し静的解析を行った大規模調査 [4] によると，86%の IoT 機器の OS は Linux であった．OSS の活用は，開発の期間短縮・コスト削減の観点で非常に効果的である．一方で，OSS はソースコードが公開されており，攻撃者にとっても有益な情報となり得る．更には，OSS に脆弱性がある場合に複数機器にまたがる脆弱性となり得る．このように，OSS はセキュリティ面のリスクを内包している．

IoT を議論するうえで，モノとネットワーク機器を区別するかは解釈が分かれるところである．“モノのインターネット” という言葉より，厳密にモノだけを指す考え方がある一方，文献 [5] のように，セキュリティを語るうえではモノとネットワーク機器をあわせて IoT 機器と称する考え方がある．モノとネットワーク機器は，いずれも本質的にはインターネットに接続された組込み機器であり，マルウェア感染の観点にお

† 横浜国立大学，横浜市
Yokohama National University, 79–1 Tokiwadai, Hodogaya-ku, Yokohama-shi, 240–8501 Japan
†† 富士ソフト株式会社，横浜市
FUJI SOFT INCORPORATED, 1–1 Sakuragi-cho, Naka-ku, Yokohama-shi, 231–8008 Japan
††† 横浜国立大学大学院環境情報研究院/先端科学高等研究院，横浜市
Graduate School of Environment and Information Sciences, Yokohama National University/Institute of Advanced Sciences, Yokohama National University, 79–1 Tokiwadai, Hodogaya-ku, Yokohama-shi, 240–8501 Japan
a) E-mail: hara-satoshi-gr@ynu.jp
DOI:10.14923/transcomj.2018WFP0014

いて共通した攻撃が通用すると考えられる．組込機器を模したハニーポットを用いた攻撃元の調査 [6] によると，モノとネットワーク機器の両方から攻撃を受けることが示されている．組込機器をターゲットとしたマルウェア “Mirai” はモノとネットワーク機器のいずれにも感染することが示されている [7], [8]．このような背景より，本論文では，インターネットに接続された組込み機器を総称して IoT 機器と定義する．

IoT 機器の多くは購入後すぐに利用できるよう設定されて出荷されていることから，管理画面等が初期パスワードのまま公開され放置されているケースが多く，不正侵入やマルウェア感染の原因となっている [9], [10]．2016 年に流行したマルウェア “Mirai” [8] は，Linux を搭載した IoT 機器に対して，汎用的な攻撃が通用することを示す結果となった．このように，十分なマルウェア対策をしていない多くの IoT 機器がマルウェアに感染しており，感染した機器は攻撃者により大規模な DDoS 攻撃等に利用されている [11]．

Mirai に代表される，IoT 機器を攻撃対象とするマルウェアの多くは，感染した機器の電源を再起動することで消滅することが知られている．一方で，特定の機器を対象とした，電源の再起動のみでは消えない “持続的感染” を引き起こすマルウェア SYNFul Knock [12]，機器を故障に至らせるマルウェア Bricker Bot [13] が報告されている．

IoT 機器の用途によっては，常時通電の機器も多数存在しており，このような機器に対しては持続的感染の有無にかかわらず長期間にわたる感染が可能である．一方で，機器の再起動による IoT マルウェアの駆除方法が公的機関より周知されている [5]．将来，駆除方法が一般に広まった状況下において，持続的感染をしないマルウェアは，大規模なボットネットを構築・維持するために短期間で大量の機器を感染させる必要があり，感染拡大時の特徴的な通信の増加を観測する目的として，ダークネットのトラヒック観測 [14] や，ハニーポットによる観測が引き続き効果的である．しかしながら，持続的感染を引き起こすマルウェアは長期にわたりゆっくりと感染を広げることが可能になり，観測が困難になると考えられる．現に，2018 年 5 月に一般に認知されたマルウェア “VPNFilter” は 2016 年頃から活動していると考えられるものの，長期にわたり多くのセキュリティ研究者に認知されることなく 50 万台以上への感染を拡大した [15]．このように持続的感染の脅威は大きく，対策が急務である．

本研究では，まず IoT 機器の実機とマルウェア検体を用いた感染実験を行った．主にハニーポットを用いてマルウェア検体の収集と攻撃元機器の特定を行い，7 種類の機器と 45 のマルウェア検体を用意した．これらの機器・マルウェア検体を用いた感染実験を行い，いずれの機器，マルウェア検体についても，機器の電源の再起動後にマルウェアのプロセスが消滅，すなわち持続的感染が生じないことを確認した．

続いて，概念実証用の疑似マルウェアを用い，これらの機器に対して 2 種類の攻撃 1) 起動スクリプト改変攻撃 2) ファームウェア攻撃により持続的感染を実証した．起動スクリプト改変攻撃では，前述の 7 種類の機器のうち，1 種類の機器において持続的感染が実現し得ることを実証した．本機器と同様の特徴のファイルシステムをもつ機器を追加で 3 種類用意し，いずれの機器においても持続的感染が実現し得ることを実証した．起動スクリプト改変攻撃において持続的感染が生じなかった 6 種類の機器に対してファームウェア攻撃を実施し，2 種類の機器で持続的感染が実現し得ることを実証した．最後に，これらの実証実験をもとに持続的感染の成立条件を分析したうえで，対策方法について議論する．

本研究により実証した攻撃手法は，Linux の標準的な機能に着目した手法であり，本研究で実証した機器にとどまらず，一般的に通用する攻撃である．特に起動スクリプト改変攻撃は現在流行するマルウェアの感染方法のわずかな変更で実現が可能であり，非常に大きな脅威である．本研究はこのような危機的状況と対策方法を IoT 機器開発者に示し，IoT 機器のセキュリティ改善を促すものである．

2. 関 連 研 究

IoT 機器への攻撃実態を把握するうえで，多くの攻撃を効率的に収集することが求められる．IoT 機器を標的としたマルウェアを引き付け効率的に捕捉する方法として，IoT 機器を模擬した囮システムによるマルウェアの挙動を解析する手法 [6], [16], [17] の提案がされている．また，IoT 機器をターゲットとしたマルウェアは，多くの機器に急速に感染を広げる事例が多い．この特徴を利用して，ダークネットと呼ばれる到達可能かつ未使用の IP アドレス空間に宛てた通信の観測による感染拡大の早期検知 [14] が提案されている．感染拡大防止の観点では，マルウェア感染した機器にリモート再侵入し 23/tcp 宛のスキャン攻撃を止めるこ

とにより感染拡大を防ぐ手法 [18] の提案がされている．

IoT 機器のファームウェアに着目した脆弱性分析手法として，web 上から大量のファームウェアを収集し静的解析を行う手法 [4] や，ファームウェアの収集と動的解析を自動化する手法 [19] が提唱されている．Zhen [20] らは，特定の 1 機器を対象としてファームウェア攻撃を行い，機器のソースコードを改変することにより，既存のプロセスに悪意のあるコードを埋め込む手法を実証している．しかし，同様の攻撃が他の機器に対して通用するかどうかは言及していない．IoT 機器におけるファームウェア更新機能は，ファームウェアを書き換えるという目的は同じであるものの，機器それぞれ独自に実装された機能であるため，他の機器に通用するとは言い難い．

IoT 機器の開発者向けセキュリティガイドラインも多く提唱されており，持続的感染の対策となり得る言及がされている．[21] では更新データの正当性の担保及びファームウェア更新機能の乗っ取り防止の必要性に言及している．[22] ではソフトウェア署名を行うことによる，不正改造されたソフトウェアの動作防止を推奨している．[23] では，ファームウェア更新のセキュリティが不十分な場合のファームウェア改変のリスクが指摘されている．ファームウェアのディジタル署名及び，製品の書込み保護の必要性についても指摘している．[24] では暗号を用いたファームウェア更新の完全性と信頼性の担保について言及している．

一般に IoT 機器は数年単位の長期間で使用されることが多い一方で，セキュリティアップデートの頻度が低いといった傾向があるため，持続的感染の脅威を体系立てて分析し，多層防御を講じることが重要である．しかしながら，現実的には IoT 機器の開発における費用や開発期間の制約から，セキュリティガイドラインを完全に網羅することは困難である．いずれのガイドラインも，持続的感染の脅威と対策に関して体系立てた記載がなく，ガイドラインの取捨選択が困難である．

我々は，文献 [25] において，観測した範囲内で IoT マルウェアの持続的感染が発生していないことを示した．続いて文献 [26] においてファームウェア更新機能を介した IoT マルウェアの持続的感染について検討を行った．本研究では，ファームウェア更新を伴わない持続的感染の可能性について検討を加えたうえで，IoT 機器に対するマルウェアの持続的感染の体系立てを行い，持続的感染の成立条件の分析と対策方法の提唱を行う．

3. 持続的感染の攻撃モデル

持続的感染を実現するためには，下記 3 点を全て満たす必要がある．

1. 機器上にマルウェアのバイナリファイルを保存する
2. 機器の電源再起動後にマルウェアのプロセスが自動起動するよう，機器の起動スクリプトを改変する
3. 上記二つの変更が，機器の再起動後も維持されるようにする

持続的感染が実現するかどうかは，感染後のマルウェアの振る舞いと関係なく，感染時に機器のどの程度の深さまで感染するかに依存する．本条件を満たす攻撃モデルを以下に二つ定義する．

3.1 起動スクリプト改変攻撃

本攻撃モデルを図 1 に示す．動作中の機器に対し何かしらの脆弱性を悪用して管理者権限によるリモートコマンド実行を行うことにより，マルウェアの格納及び起動スクリプトの改変を行う方法である．IoT 機器は，PC やサーバーと違い，機器の電源再起動時に一部の設定情報を除き変更を破棄する設計思想で作られている．機器自体が起動時若しくはシャットダウン時にリカバリ処理を行っている可能性があり，電源再起動後に変更が維持されるかどうかは，機器の設計に依存すると考えられる．

3.2 ファームウェア攻撃

ファームウェア攻撃を説明するにあたり，本論文における “ファームウェア” の定義を行う．“ファームウェア” という用語は，「機器に組み込まれたソフトウェア」の意味で使用される場合と，「機器に組み込まれたソフトウェアを書き換えるためのデータ」の意味で使用される場合がある．本論文では下記のとおり明示的に使い分ける．

ファームウェア

機器に所望の動作をさせるために不揮発メモリに

図 1 起動スクリプト改変攻撃
Fig. 1 Initial script change attack.

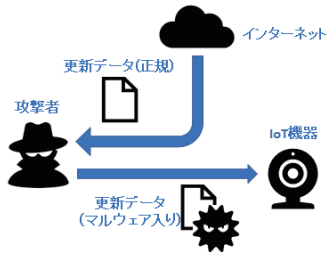

図 2　ファームウェア攻撃
Fig. 2　Firmware attack.

書き込まれているソフトウェア一式を指す．

更新データ

ファームウェアを更新するために，機器に入力するデータを示す．ファームウェアと等価な場合もあるが，ファームウェアの一部のみを更新するものや，アップデーターやチェックサムのような付加データを含む場合もある．

本攻撃モデルを図 2 に示す．マルウェアの格納及び起動スクリプト改変を行った更新データを生成し，機器のファームウェアを書き換える方法である．IoT 機器の多くは購入後にファームウェアを書き換える機能があり，正規の更新データはメーカーのウェブサイトに公開されている場合が多い．

3.3　既知の持続的感染型マルウェアの考察

持続的感染型マルウェア SYNful Knock [12] の攻撃対象の OS は Linux ではなくメーカー独自の OS であるものの，改変した更新データを用いた攻撃であり，ファームウェア攻撃である．一方で，持続的感染型マルウェア VPNFilter [15] の感染方法は本論文執筆時点で明らかになっていない．しかし，プログラム自動起動の仕組みである crontab の設定ファイルを改変し持続的感染を実現していることが分かっている．そのため，起動スクリプト改変攻撃若しくはファームウェア攻撃のいずれかであると推測される．

4.　実 証 実 験

最初に実験 1 では，市販の機器と実際のマルウェア検体を用いて持続的感染の有無の調査を行う．続いて実験 2 では，起動スクリプト改変攻撃における持続的感染の実現性を実証する．最後に実験 3 では，ファームウェア攻撃における持続的感染の実現性を実証する．本実証実験で用いる機材を表 1 に示す．機器 A～G は実験 1～実験 3 で使用する．機器 H～J は実験 2 及び実験 3 で使用する．

表 1　機 材 一 覧
Table 1　Device list.

機器	種類	国	アーキテクチャ
A	IP Camera	台湾	ARM
B	プリンター	アメリカ	ARM
C	ルータ	台湾	MIPS
D	Wi-Fi ストレージ/ポケット Wi-Fi	日本	MIPSEL
E	Wi-Fi ストレージ/ポケット Wi-Fi	日本	MIPSEL
F	Wi-Fi ストレージ/ポケット Wi-Fi	アメリカ	MIPSEL
G	衛星放送受信機	ドイツ	SH
H	モバイル Wi-Fi ルータ	日本	ARM
I	モバイル Wi-Fi ルータ	フランス	ARM
J	モバイル Wi-Fi ルータ	中国	ARM

4.1　実　験　1

まず持続的感染の有無を確認するため，実機及びマルウェア検体を用いた感染実験を実施する．IoT 機器の内部システムはメーカーや機器の種類により異なり，マルウェアの挙動も異なると推測されるため，7 種類の実機と計 45 のマルウェア検体を用いて実証実験を行う．

4.1.1　実 験 方 法

実験手順を以下に示す．

1. 機器内部のファイルシステム (全ファイル名とファイルサイズ) とプロセスを感染前の状態として記録する
2. 事前に入手したマルウェアバイナリファイルを機器に転送し実行する
3. マルウェア実行後，プロセスの増加や攻撃者の C&C サーバーに対する通信を開始するなどにより機器がマルウェア感染状態になったことを確認する
4. 機器の電源を再起動する
5. 手順 1 で記録した感染前の機器のファイルシステムやプロセスと比較し，感染の持続性を評価する

実験環境を図 3 に示す．通信制御・観測用マシンは IoT 機器からインターネットへの通信の制御と観測を行う他，IoT 機器に対するマルウェアバイナリファイルの転送と，機器のファイルシステム及びプロセスの記録も行う．

4.1.2　実験用機器と検体の準備

本項では実験に使用した機器と検体の準備について

表 2　マルウェア検体 (ARM)
Table 2　Specimen malware (ARM).

	md5hash 値	trend micro	Avg	Kaspersky
1	06ffde55395f4508c2a8edd6fb8c2a83	ELF_BASHLITE.SM	a variant of Linux/Gafgyt.C	Linux/Fgt.AB
2	1e1a9668e86dfec845893968760dcdb6	ELF_BASHLITE.DHD	a variant of Linux/Gafgyt.MT	Linux/Fgt
3	4493a098787ef436308ef1ed571a6ab9	ELF_BASHLITE.DIB	a variant of Linux/Gafgyt.SR	Linux/Fgt
4	a021d25c75f2a0dfa6a1103ac812c3e1	ELF_BASHLITE.DIA	a variant of Linux/Gafgyt.QE	Linux/Fgt.BP
5	ac8733aa6c973564643236ee2535083b	検知されず	a variant of Linux/Gafgyt.C	Linux/Fgt.CE
6	1a3480286d6f6b4e7fe42fa11ee1122d	検知されず	a variant of Linux/Gafgyt.C	Linux/Fgt.AB
7	c7c3a9c7a73c9f8dcd160a86be65740b	検知されず	a variant of Linux/Mirai.A	Linux/Fgt.CI
8	59e59d762e1eb8838d33c4e133c6a546	検知されず	検知されず	Linux/Fgt.CI
9	238a67e6f9b129680b618a3c579a8c6c	ELF_MIRAI.A	a variant of Linux/Mirai.A	Linux/Fgt.CI
10	8d5419f45c1074713325a5f414bcb347	検知されず	a variant of Linux/Tsunami.NAL	Linux/Tsunami.BK
11	ae68fa96792645d7e8ef40031330f4f3	ELF_SONEX.SMA	Linux/Dnsamp.B	Linux/MrBlack.A

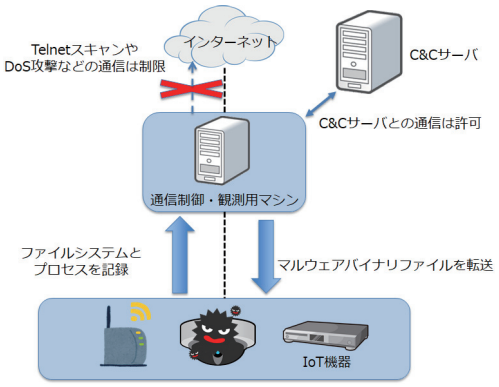

図 3　実験環境
Fig. 3　Experimentation environment.

記す．まず，実験に使用するための機器では何かしらの手段で OS コマンドのリモート実行ができる必要がある．そこで，Telnet 接続が可能な機器の情報を当研究室のハニーポット [6], [16], [17] を用いて収集した．具体的には，ハニーポットで 23/tcp 宛の攻撃を観測した際に，攻撃元の IP アドレスに対してスキャンすることで攻撃元ホストの Telnet のバナー情報を取得し，マルウェアに感染したと考えられる機器の製品名や型番を特定した．そして，同機種を幾つか購入し実際に Telnet でログイン可能なことを確認した機器を実験に使用した．本手法により，表 1 の機器 A～機器 G の 7 種類の機器を用意した．

実験に使用した検体についても，主にハニーポットを利用して準備を行った．まず，ハニーポットを用いて攻撃者が実行するマルウェアをダウンロードするシェルコマンドを観測する．その後，別マシンでこのコマンドを再現し攻撃者のダウンロードサーバーからマルウェアをダウンロードした．また，一部の検体についてはマルウェア解析 Web サイト「VirusTotal」[27] から検体をダウンロードした．機器の CPU アーキテクチャごとに，ARM [28] で動作するものを 11 検体 (表 2)，MIPS [29] で動作するものを 12 検体 (表 3)，MIPSEL [29] で動作するものを 12 検体 (表 4)，SH(Super H) [30] で動作するものを 10 検体 (表 5) の計 45 検体を使用した．マルウェア検体 1～7，12～18，24～30，36～43 の 29 検体については，我々が運用する IoT ハニーポットで取得したものであり，残りの 16 検体は「VirusTotal」から取得した．いずれの検体も 2016 年 9 月から 12 月にかけて収集を行った．

4.1.3　実験結果

本節では感染実験の結果を表 6～表 12 に示す．結果表中の「○」は機器の電源再起動により，悪性プロセス若しくはマルウェアのバイナリファイルが残留したことを示す．「×」は機器の電源再起動により，悪性プロセス若しくはマルウェアのバイナリファイルが削除されたことを示す．

機器 A の実験結果を表 6 に示す．検体 7 のマルウェアは動作しなかったが，動作した他の検体ではいずれ

表 3　マルウェア検体 (MIPS)
Table 3　Specimen malware (MIPS).

	md5hash 値	trend micro	Avg	Kaspersky
12	4ee0f67ed642ea4b1b81ce15e6063e29	ELF_BASHLITE.SM	a variant of Linux/Gafgyt.C	Linux/Fgt.AB
13	9133cb25d9df542a131c23043d0b3e87	ELF_BASHLITE.DHD	a variant of Linux/Gafgyt.MT	Linux/Fgt
14	6b70af7e7b23a4308d8cc998e7a719d4	ELF_BASHLITE.DIB	a variant of Linux/Gafgyt.SR	Linux/Fgt
15	33f9b8c338b23a5101c8368a6cf55890	ELF_BASHLITE.DIA	a variant of Linux/Gafgyt.QE	Linux/Fgt.BP
16	97688fd70bfcf344a9ab1ea1cd7c3b6a	検知されず	a variant of Linux/Gafgyt.C	Linux/Fgt.CE
17	e8b7ba10d79ba9803391996dc548233c	BASHLITE	a variant of Linux/Gafgyt.C	Linux/Fgt.AB
18	933364b18742a0bbd7b9b5ca1b0adea6	検知されず	a variant of Linux/Mirai.A	Linux/Fgt.CI
19	be01698c9dfa07c0e4f9d93b89ef3b82	検知されず	検知されず	Linux/Fgt.CI
20	9ba4401c2a4faa8975175498dc1fbfd4	ELF_MIRAI.A	a variant of Linux/Mirai.A	Linux/Fgt.CI
21	cfec82edb6a96d43667741ab62d63a44	検知されず	a variant of Linux/Tsunami.NAL	Linux/Tsunami.CT
22	5afdcceb2fc5fc1c15d7fdbef674c6a5	検知されず	a variant of Linux/PNScan.A	Linux/Generic_c.ZB
23	811351fd15c2c4c355543c4bac3d7cc9	ELF_SONEX.SMC	Linux/Dnsamp.A	Linux/MrBlack.D

表 4　マルウェア検体 (MIPSEL)
Table 4　Specimen malware (MIPSEL).

	md5hash 値	trend micro	Avg	Kaspersky
24	7a1bace1b58a1619fe4122ee8cfce3a5	ELF_BASHLITE.SM	a variant of Linux/Gafgyt.C	Linux/Fgt.AB
25	c214ddb267d339229f71875cc1839261	ELF_BASHLITE.DHD	a variant of Linux/Gafgyt.MT	Linux/Fgt
26	7474bec1a0427efe1a46d79293a1b706	ELF_BASHLITE.DIB	a variant of Linux/Gafgyt.SR	Linux/Fgt
27	dcd9d22066509902647e4a40741a9dcd	ELF_BASHLITE.DIA	a variant of Linux/Gafgyt.QE	Linux/Fgt.BP
28	6541826fd6428a99ef77a8188073192c	検知されず	a variant of Linux/Gafgyt.C	Linux/Fgt.CE
29	65cdb4c8a76dd0cc75f0aa15f13590fd	検知されず	a variant of Linux/Gafgyt.C	Linux/Fgt.AB
30	f6e7d8574469ed5f76f23945efaeb0cc	検知されず	a variant of Linux/Mirai.A	Linux/Fgt.CI
31	d55e3cccf128eb0f6b1ec176fdce2573	検知されず	検知されず	Linux/Fgt.CI
32	5849bb9ceefee5ef295e7e966d0ba2b5	ELF_MIRAI.A	a variant of Linux/Mirai.A	Linux/Fgt.CI
33	e0f1b98d6778f0eba97be0a3cafc4d52	検知されず	a variant of Linux/Tsunami.NAL	Linux/Tsunami.CT
34	ac93f826c3031859f4a71529794fe7f9	検知されず	Linux/Dofloo.A	Linux/Dofloo.A
35	856f14251f643bac62b9193c54449472	検知されず	Linux/PNScan.A	Linux/Generic_c.AKS

も機器の電源の再起動により，マルウェアのバイナリとプロセスはともに消滅した．

機器 B の実験結果を表 7 に示す．この機器では，検体 4, 7, 8 のマルウェアは動作しなかった．動作した他の検体ではいずれも機器の電源の再起動により，マルウェアのプロセスは消滅したが，バイナリは残留した．検体 9 では，マルウェアのバイナリが残っていなかったが，これはマルウェアの実行後に自身のバイナリファイルを削除するという挙動によるものである．

機器 C の実験結果を表 8 に示す．実験に用いた 12 検体全てにおいて，機器の電源の再起動により，マルウェアのバイナリとプロセスはともに消滅した．

機器 D の実験結果を表 9 に示す．検体 32 のマルウェアは動作しなかったが，動作した他の検体ではい

表 5　マルウェア検体 (SH)
Table 5　Specimen malware (SH).

	md5hash 値	trend micro	Avg	Kaspersky
36	65aaa70a0986597196cb7416fab7de9b	ELF_BASHLITE.SM	a variant of Linux/Gafgyt.C	Linux/Fgt.AB
37	02f72a165304074e9e437800c35154da	ELF_BASHLITE.DHD	a variant of Linux/Gafgyt.MT	Linux/Fgt
38	4b49b439ff434f0aad0c36efd22378a9	ELF_BASHLITE.DIB	a variant of Linux/Gafgyt.SR	Linux/Fgt
39	e62aa5444b24a36848d012c67ee3bcf9	ELF_BASHLITE.DIA	a variant of Linux/Gafgyt.QE	Linux/Fgt.BP
40	9f9cc8c71822ca872de2c28e9c41f44a	検知されず	a variant of Linux/Gafgyt.C	Linux/Fgt.CE
41	e892b8e008aeb50f2634d7c602b8fb77	検知されず	a variant of Linux/Gafgyt.C	Linux/Fgt.AB
44	c44e92168d07a637fa8943dd95f2a462	検知されず	a variant of Linux/Mirai.A	Linux/Fgt.CI
42	e867f3fc4bffe17bfd8cdcbfbfb28dc5	検知されず	検知されず	Linux/Fgt.CI
43	a490bb1c9a005bcf8cfe4bdffe7b991f	ELF_MIRAI.A	a variant of Linux/Mirai.A	Linux/Fgt.CI
45	cc33124ee92cc9b9b120cd54fee94d9b	検知されず	a variant of Linux/Tsunami.NAL	Linux/Tsunami.CT

表 6　実験 1 結果 (機器 A)
Table 6　Experiment 1 result (Device A).

検体	マルウェアのバイナリの残留	マルウェアのプロセスの残留
検体 1	×	×
検体 2	×	×
検体 3	×	×
検体 4	×	×
検体 5	×	×
検体 6	×	×
検体 7	動作せず	動作せず
検体 8	×	×
検体 9	×	×
検体 10	×	×
検体 11	×	×

表 7　実験 1 結果 (機器 B)
Table 7　Experiment 1 result (Device B).

検体	マルウェアのバイナリの残留	マルウェアのプロセスの残留
検体 1	○	×
検体 2	○	×
検体 3	○	×
検体 4	動作せず	動作せず
検体 5	○	×
検体 6	○	×
検体 7	動作せず	動作せず
検体 8	動作せず	動作せず
検体 9	×	×
検体 10	○	×
検体 11	○	×

ずれも機器の電源の再起動により，マルウェアのバイナリとプロセスはともに消滅した．

機器 E の実験結果を表 10 に示す．検体 26 のマル

表 8　実験 1 結果 (機器 C)
Table 8　Experiment 1 result (Device C).

検体	マルウェアのバイナリの残留	マルウェアのプロセスの残留
検体 12	×	×
検体 13	×	×
検体 14	×	×
検体 15	×	×
検体 16	×	×
検体 17	×	×
検体 18	×	×
検体 19	×	×
検体 20	×	×
検体 21	×	×
検体 22	×	×
検体 23	×	×

表 9　実験 1 結果 (機器 D)
Table 9　Experiment 1 result (Device D).

検体	マルウェアのバイナリの残留	マルウェアのプロセスの残留
検体 24	×	×
検体 25	×	×
検体 26	×	×
検体 27	×	×
検体 28	×	×
検体 29	×	×
検体 30	×	×
検体 31	×	×
検体 32	動作せず	動作せず
検体 33	×	×
検体 34	×	×
検体 35	×	×

表 10　実験 1 結果 (機器 E)
Table 10　Experiment 1 result (Device E).

検体	マルウェアのバイナリの残留	マルウェアのプロセスの残留
検体 24	×	×
検体 25	×	×
検体 26	動作せず	動作せず
検体 27	×	×
検体 28	×	×
検体 29	×	×
検体 30	×	×
検体 31	×	×
検体 32	×	×
検体 33	×	×
検体 34	×	×
検体 35	×	×

表 11　実験 1 結果 (機器 F)
Table 11　Experiment 1 result (Device F).

検体	マルウェアのバイナリの残留	マルウェアのプロセスの残留
検体 24	×	×
検体 25	×	×
検体 26	×	×
検体 27	×	×
検体 28	×	×
検体 29	×	×
検体 30	×	×
検体 31	×	×
検体 32	×	×
検体 33	×	×
検体 34	×	×
検体 35	×	×

表 12　実験 1 結果 (機器 G)
Table 12　Experiment 1 result (Device G).

検体	マルウェアのバイナリの残留	マルウェアのプロセスの残留
検体 36	×	×
検体 37	動作せず	動作せず
検体 38	×	×
検体 39	×	×
検体 40	動作せず	動作せず
検体 41	×	×
検体 42	×	×
検体 43	×	×
検体 44	×	×
検体 45	×	×

ウェアは動作しなかったが，動作した他の検体ではいずれも機器の電源の再起動により，マルウェアのバイナリとプロセスはともに消滅した．

機器 F の実験結果を表 11 に示す．実験に用いた 12 検体全てにおいて，機器の電源の再起動により，マルウェアのバイナリとプロセスはともに消滅した．

機器 G の実験結果を表 12 に示す．検体 37 と 40 は動作しなかったが，動作した他の検体ではいずれも機器の電源の再起動により，マルウェアのバイナリとプロセスはともに消滅した．

4.2 実　験　2

実験 1 では，いずれの機器・マルウェアの組み合わせにおいても，持続的感染は発生しなかった．実験 2 では，これらの機器に対して起動スクリプト改変攻撃を行い，持続的感染の実現性を実証する．

4.2.1 実 験 方 法

持続的感染は感染後のマルウェアの振る舞いと関係なく，感染時に機器のどの程度の深さまで感染するかに依存する．そのため，機器上でプロセスを起動するだけの疑似マルウェアを用意し実験を行う．下記の実験手順による感染により，各機器で持続的感染に至るかどうかを検証する．実験手順を以下に示す．

1. 疑似マルウェアのバイナリファイルを機器に転送し実行する
2. マルウェア実行後，疑似マルウェアのプロセスが開始し感染状態となったことを確認する
3. 機器の起動時に疑似マルウェアのプロセスが自動起動するよう，起動スクリプトを改変する
4. 機器の電源を再起動する
5. 再起動後，疑似マルウェアのプロセスが開始したことを確認する

管理者権限によるリモートコマンド実行の手段として，本実験では機器のもつ telnet/SSH のパスワードの脆弱性を使用する．改変対象の起動スクリプトは，Linux で一般的に使われる “Init Script” [31] とする．機器が何かしらの復旧機能を有していない場合，手順 3. により持続的感染が生じると考えられる．

4.2.2 実 験 結 果

実験結果を表 13 に示す．機器 B のみ再起動後に疑似マルウェアのプロセスが残留している状態であり，持続的感染が生じている．一方，他の機器においては，疑似マルウェアのバイナリや起動スクリプト改変といった，機器に対する変更が復旧されていることが分かる．

各機器のファイルシステムを調査したところ，表 14 のとおりであった．機器 A はファイルシステムが特定できなかったものの，挙動より読み取り専用ファイルシステムと考えられる．持続的感染とファイルシステムの関連を明確にするため，読み書き可能ファイルシステムを有する機器を表 1 の機器 H～機器 J のとおり用意し同様に実験を行ったところ，表 13 に示すと

表 13　実験 2 結果
Table 13　Experiment 2 result.

機器	疑似マルウェアのバイナリの残留	起動スクリプト改変の残留	疑似マルウェアのプロセスの残留
A	×	×	×
B	○	○	○
C	×	×	×
D	×	×	×
E	×	×	×
F	×	×	×
G	×	×	×
H	○	○	○
I	○	○	○
J	○	○	○

表 14　ファイルシステム
Table 14　Filesystem.

機器	ファイルシステム
A	不明
B	UBIFS（読み書き可能）
C	squashfs（読み取り専用）
D	squashfs（読み取り専用）
E	squashfs（読み取り専用）
F	squashfs（読み取り専用）
G	cramfs（読み取り専用）
H	YAFFS2（読み書き可能）
I	YAFFS2（読み書き可能）
J	YAFFS2（読み書き可能）

おり，機器 H〜機器 J 全てにおいて持続的感染を確認した．本結果により持続的感染を引き起こすかどうかはファイルシステムの性質に依存するといえる．読み書き可能ファイルシステムを有する機器に対しては，現状のマルウェアの感染方法に加え，起動スクリプトを改変する処理を加えるのみで持続的感染を引き起こす可能性が高く，非常に危険な状況といえる．

4.3 実　験　3

実験 2 において持続的感染を生じなかった機器に対して，ファームウェア攻撃による持続的感染の実現性を実証する．

4.3.1 実 験 方 法

本実験では，実験 2 と同じ疑似マルウェアを使用する．実験手順を以下に示す．

1. メーカーの HP より，正規の更新データを入手する
2. 正規の更新データの中からファイルシステムを抽出・展開する
3. ファイルシステム上に疑似マルウェアを格納する
4. 機器の起動時に疑似マルウェアのプロセスが自動起動するよう，ファイルシステム上の起動スクリプトを改変する
5. 改変したファイルシステムをもとに，更新データを再構築する
6. 改変した更新データを機器に書き込んだうえ，機器の電源を再起動する
7. 再起動後，疑似マルウェアのプロセスが開始したことを確認する

4.3.2 実 験 結 果

本実験の結果，2 種類の機器で持続的感染を確認した．他の 1 種類の機種では，機器への書き込みは行っていないものの，更新データの改変まで成功した．

以下に機器ごとの結果を示す．機器 A はメーカー HP より更新データを入手することができず，実験を行えなかった．機器 C,G は正規の更新データを入手でき，またファイルシステムの抽出も実施できた．しかし，ファームウェア更新機能における，更新データの正当性判定処理を抽出できなかったため，更新データの再構築を行えなかった．機器 D,E は手順通り実行することにより，持続的感染を生じることを確認した．機器 F は更新データを改変し，機器に受け入れられるようチェックサムの改変まで行っており，持続的感染を実現できる可能性が高い．しかし，機器の入手性の問題があるため，改変した更新データの機器への書き込みは行わなかった．

5. 考　　察

4. の実証実験では，実際にマルウェアの持続的感染が実現されることを，実機を用いて実証した．起動スクリプト改変攻撃とファームウェア攻撃のいずれにおいても，Linux の標準的なファイルシステム及びシステム起動スクリプトに着目した持続的感染を実証しており，他機器へも同様の攻撃が通用する可能性が高いと言える．以下では，持続的感染の成立条件及び対策について考察を行う．

5.1 起動スクリプト改変攻撃

下記の条件を全て満たす場合に，起動スクリプト改変攻撃による持続的感染が成立する．

(a) 機器上にマルウェアのバイナリファイルを保存できること
(b) 機器の電源再起動後にマルウェアのプロセスが自動起動するよう，機器の起動スクリプトを改変できること
(c) 上記二つの変更が，機器の再起動後も維持される

ようにすること

(a)(b) の対策として，攻撃者にリモート接続をさせないことが挙げられる．具体的には，下記の対策が効果的である．

(1) リモート接続を許可しない
(2) アカウントロック機能を導入する
(3) 機器の初回仕様時にパスワード変更を義務付ける

(c) の対策として，読み取り専用ファイルシステムの活用が挙げられる．ファイルシステム全体に適用することが望ましいが，機器の機能上困難な場合は，最低限として起動スクリプトやタスクスケジューラーの領域に適用すべきである．

5.2 ファームウェア攻撃

下記の条件を全て満たす場合に，ファームウェア攻撃による持続的感染が成立する．

(A) 機器の更新データを入手可能であること
(B) 機器の更新データを解析・改変可能であること
(C) 改変後の更新データが機器に受け入れられること
(D) ファームウェア更新機能を攻撃者が利用できること

(A) の対策として，更新データの隠蔽が挙げられる．具体的には，web 上での更新データの配布をせず，機器自体が暗号化された通信経路を用いて更新サーバーから更新データを取得するという方法が挙げられる．しかしながら，オフライン環境でファームウェア更新が行えないなどの制約も発生し，対応が困難な場合も多いと考えられる．

(B) の対策として，更新データの暗号化が挙げられる．暗号の不適切な使用は脆弱性の原因になり得るため，使用方法には十分な配慮が必要である．暗号の不適切な使用の具体例の一つとして，XOR 暗号の使用が挙げられる．XOR 暗号は既知平文攻撃により容易に暗号キーを推定できる問題がある [32]．平文の更新データでは，ファイルシステムの空き領域や，更新データのサイズ調整の領域が 0 埋めなど容易に推測可能な bit 配列であることが多く，既知平文攻撃が通用する恐れがある．また，このように平文の更新データは偏りのあるデータの可能性があり，ECB モードのようなブロック暗号では平文漏洩のリスクがある．更新データの暗号化は，ECB モード以外の暗号モードを使用するべきである．

(C) の対策として，更新データにメーカーの電子署名を付与したうえで，ファームウェア更新機能で署名検証を行う方法が挙げられる．

(D) の対策として，ファームウェア更新機能をリモートから実行できないよう制限する方法が考えられる．しかしながら，ソーシャルエンジニアリングのように，機器の管理者自身に悪意のある更新データを書き込ませる方法も考えられ，対策としては不十分である．

5.3 研究倫理的考察

当該研究はメンロレポート [33] に規定される研究倫理原則に基づき実施した．IoT マルウェアの持続的感染の有無は，対策を検討する上で重要な要素であるにもかかわらず，これまで十分な研究が実施されておらず，正確な情報が提供されていないのが現状である．特に 2016 年に大流行した Mirai やその亜種が持続的感染機能を有していないことから，一般に IoT マルウェアが持続的感染しないという誤った認識に基づき対策が検討される恐れがある．本研究は，IoT マルウェアの持続的感染の可能性について，機器の性質をふまえて体系的に検討するとともに複数の具体的対策方法を示すものであり，今後出現する可能性がある持続的感染型の IoT マルウェアへの対策に資すると考える．一方，本研究成果の悪用の影響を最小化するため，具体的な機器の情報の匿名化を行うことで直接的な悪用を防ぐとともに，情報処理推進機構・JPCERT/CC に当該研究成果に関する情報提供の実施，及び本論文の執筆にあたり新たに持続的感染の可能性が明らかとなった機器のメーカーへの情報提供を実施する予定である．1 機器に関しては，情報処理推進機構を窓口として JPCERT/CC 及び機器のメーカーへ情報提供済である．残る 5 機器に関しても，本論文の公表の 90 日前までに情報提供を実施する予定である．このように本研究により得られる恩恵は，その悪用による潜在的な危害を大きく上回ると考える．

6. むすび

本研究では，Linux を使用した IoT 機器において，7 種類の IoT 機器と計 45 の IoT マルウェア検体による感染実験を行い，これらの機器・マルウェアにおいて持続的感染が発生していないことを示した．続いて，“起動スクリプト改変攻撃” と “ファームウェア攻撃” の 2 種類の手法により，これらの機器に持続的感染が実現可能であることを示した．特に，“起動スクリプト改変攻撃” では持続的感染能力をもたないマルウェア

に対して，わずかに変更を加えるのみで持続的感染が生じる危険性を示した．これらの実証をもとに，“起動スクリプト改変攻撃” と “ファームウェア攻撃” のそれぞれにおいて持続的感染の成立条件を分析したうえで，持続的感染の対策について考察を行った．

本研究の課題として，感染実験に使用したマルウェア検体の網羅性が挙げられる．本論文の執筆時点では本研究で使用したマルウェア検体の亜種による攻撃が依然として多く観測されており，この点は現状の脅威を反映しているものと言える．しかしながら，現状の観測環境で VPNFilter のような新種のマルウェアの検体の収集や感染活動の観測を行えない点に問題がある．今後はハニーポットを改善のうえ，攻撃の観測及び感染実験の網羅性を高めていきたい．

謝辞　本研究の一部は文部科学省国立大学改革強化推進事業の支援を受けて行われた．本研究成果の一部は，国立研究開発法人情報通信研究機構（NICT）の委託研究「Web 媒介型攻撃対策技術の実用化に向けた研究開発」の支援により得られた．

文　献

[1] “Gartner Says 6.4 Billion Connected “Things” Will Be in Use in 2016, Up 30 Percent From 2015,” http://www.gartner.com/newsroom/id/3165317 (参照 2017–7–24).

[2] “IoT Malware Activity Already More Than Doubled 2016 Numbers,” https://threatpost.com/iot–malware–activity–already–more–than–doubled–2016–numbers/126350/ (参照 2017–10–30).

[3] Linux Foundation, https://www.linuxfoundation.org/ (参照 2018–5–28).

[4] A. Costin, J. Zaddach, A. Francillon, and D. Balzarotti, “A Large-Scale Analysis of the Security of Embedded Firmwares,” 23rd USENIX Security Symposium, pp.94–110, 2014.

[5] 独立行政法人情報処理推進機構セキュリティセンター，“情報セキュリティ10 大脅威 2018,” 2018.

[6] Y.M. Pa Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, “IoTPOT: Analysing the rise of IoT compromises,” USENIX/WOOT’15, 2015.

[7] M. Antonakakis, T. April M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J.A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, “Understanding the Mirai Botnet,” 26th USENIX Security Symposium, 2017.

[8] “JVNTA#95530271 Mirai 等のマルウェアで構築されたボットネットによる DDoS 攻撃の脅威,” https://jvn.jp/ta/JVNTA95530271/ (参照 2017–7–24).

[9] C. Ang and S. Salvatore, “A quantitative analysis of the insecurity of embedded network devices: Results of a wide–area scan,” 26th Annual Computer Security Applications Conference: Proceedings: Austin, Texas, USA: 6-10 Dec. 2010.

[10] 中里純二，島村隼平，衛藤将史，井上大介，中尾康二，“nicter によるネットワーク観測および分析レポート～組み込みシステムに感染するマルウエア,” 信学技報，ISEC2013-48, 2013.

[11] “Security issues within the IoT: The problem with DDoS – トレンドマイクロセキュリティブログ,” http://blog.trendmicro.com/security–issues–within–iot–problem–ddos/ (参照 2016–12–11).

[12] “SYNful Knock – A Cisco router implant – Part I,” https://www.fireeye.com/blog/threat–research/2015/09/synful_knock_–_acis.html (参照 2017–7–11).

[13] “IoT 機器を「使用不能」にするマルウェア,「BrickerBot」,” http://blog.trendmicro.co.jp/archives/14757 (参照 2017–7–24).

[14] 笠間貴弘，井上大介，“大規模ダークネット観測と能動的スキャンによるマルウェア感染 IoT 機器の分類,” 情処学論，vol.58, no.9, pp.1388–1398, 2017.

[15] “New VPNFilter malware targets at least 500K networking devices worldwide,” https://blog.talosintelligence.com/2018/05/VPNFilter.html (参照 2018–5–26).

[16] 鈴木将吾，インミンパパ，江澤優太，鉄　穎，中山　颯，吉岡克成，松本　勉，“組込み機器への攻撃を観測するハニーポット IoTPOT の機能拡張,” 信学技報，ICSS2015-47, 2016.

[17] 中山　颯，鉄　穎，楊　笛，田宮和樹，吉岡克成，松本勉，“IoT 機器への Telnet を用いたサイバー攻撃の分析,” 情報処理学会コンピュータセキュリティシンポジウム 2016，セッション 3E1，2016.

[18] 田辺瑠偉，鈴木将吾，イン ミン パパ，吉岡克成，松本　勉，“マルウェア感染ホストへのリモート再侵入により感染拡大を阻止する手法,” 情処学論，vol.57, no.9, pp.2021–2033, 2016.

[19] D.D. Chen, M. Egele, M. Woo, and D. Brumley, “Towards automated dynamic analysis for Linux-based embedded firmware,” Network and Distributed System Security Symposium, 2016.

[20] Z. Ling, J. Luo, Y. Xu, C. Gao, K. Wu, and X. Fu, “Security vulnerabilities of Internet of things: A case study of the smart plug system,” IEEE Internet of Things Journal, vol.4, no.6, pp.1899–1909, Dec. 2017.

[21] IoT 推進コンソーシアム，総務省，経済産業省，“IoT セキュリティガイドライン [2016].”

[22] 独立行政法人情報処理推進機構，“IoT 開発におけるセキュリティ設計の手引き [2017].”

[23] Cloud Security Alliance, Future–proofing the Connected World, “13 Steps to Developing Secure IoT Products [2016].”

[24] U.S. Department of Homeland Security, “Strategic Principles for Securing the Internet of Things (IoT) [2016].”

[25] 田宮和樹，中山　颯，江澤優太，鉄　穎，呉　俊融，楊笛，吉岡克成，松本　勉，“IoT マルウェア駆除と感染防止に関する実機を用いた実証実験，” 暗号と情報セキュリティシンポジウム 2017，セッション 3E1-5, 2017.
[26] 原　悟史，渡辺露文，田宮和樹，吉岡克成，松本　勉，“IoT マルウェアの持続的感染の成立要因の分析と実機による検証，” Computer Security Symposium 2017, Oct. 2017.
[27] VirusTotal, https://www.virustotal.com/ja/ (参照 2016–12–11).
[28] ARM, https://www.arm.com/ja/ (参照 2016–08–02).
[29] MIPS Processors – Imagination Technologies, https://imgtec.com/mips (参照 2016–08–02).
[30] SuperH RISC engine ファミリ – RENESAS, https://www.renesas.com/ja–jp/products/microcontrollers–microprocessors/superh.html (参照 2016–12–11).
[31] How To Configure a Linux Service to Start Automatically After a Crash or Reboot – Part 2: Reference, https://www.digitalocean.com/community/tutorials/how–to–configure–a–linux–service–to–start–automatically–after–a–crash–or–reboot–part–2–reference (参照 2018–5–18).
[32] ISO, “ISO/IEC TS 29167-15 Information technology — Automatic identifidation and data capture techniques – Part 15: Crypto suite XOR security services for air interface comunications,” 2017.
[33] “The Menlo Report (Aug. 2012),” https://www.dhs.gov/sites/default/files/publications/CSD–MenloPrinciplesCORE–20120803_1.pdf (参照 2019–1–11).

（平成 2018 年 10 月 23 日受付，2019 年 2 月 24 日再受付，4 月 23 日早期公開）

原　悟史　（学生員）

2005 年 3 月明治大学大学院理工学研究科機械工学専攻博士課程前期修了，修士（工学）．同年 4 月富士ソフト株式会社に入社．組込機器の開発を経て，現在，組込機器のセキュリティ研究に従事．2017 年 4 月より横浜国立大学大学院環境情報学府情報メディア環境学専攻博士課程後期に在学中．

田宮　和樹

2019 年 3 月，横浜国立大学大学院環境情報学府情報メディア環境学専攻博士課程前期修了，修士（工学）．在学中，IoT セキュリティの研究に従事．

鉄　　穎

2018 年 6 月横浜国立大学大学院環境情報学府情報メディア環境学専攻博士課程後期修了．博士（情報学）．情報セキュリティ，特にネットワーク攻撃観測・分析等のネットワークセキュリティ研究に従事．2018 年 8 月よりトヨタ自動車（株）で自動車の安全とセキュリティに関する研究に従事．

渡辺　露文　（学生員）

1999 年 3 月武蔵工業大学大学院工学研究科修士課程修了，同年 4 月より富士ソフト株式会社にて IT インフラエンジニアとしてシステム構築や，社内 IT インフラの企画・構築・運用，IPv6 調査研究を経て，現在はセキュリティの研究に従事．2017 年 4 月より横浜国立大学大学院環境情報学府情報メディア環境学専攻博士課程後期に在学中．

吉岡　克成　（正員）

2005 年より（独）情報通信研究機構にてインシデント対策センター NICTER の研究開発に従事．2008 年より横浜国立大学にてサイバーセキュリティ研究開発を開始．2009 年文部科学大臣表彰，2016 年産学官連携功労者表彰総務大臣賞．総務省「国際連携によるサイバー攻撃の予知即応技術の研究開発」他，プロジェクトに多数参画．博士（工学）．

松本　　勉　（正員）

1986 年 3 月東京大学大学院工学系研究科電子工学専攻博士課程修了，工学博士．同年 4 月横浜国立大学講師．2001 年 4 月より同大学院環境情報研究院教授．先端科学高等研究院情報・物理セキュリティ研究ユニット代表を兼務．暗号アルゴリズム・プロトコル，ネットワーク/ソフトウェア/ハードウェアセキュリティ，バイオメトリクス，人工物メトリクス，計測セキュリティ，自動車セキュリティ等の「情報・物理セキュリティ」の研究教育に 1981 年より従事．1982 年にオープンな学術的暗号研究を目指した「明るい暗号研究会」を 4 名で創設．2005 年-2010 年国際暗号学会 IACR 理事．CRYPTREC 暗号技術検討会座長．日本学術会議連携会員．産業技術総合研究所研究顧問．第 32 回電子情報通信学会業績賞，第 5 回ドコモ・モバイル・サイエンス賞，第 4 回情報セキュリティ文化賞，2010 年文部科学大臣表彰・科学技術賞（研究部門）受賞．