Computer Security Symposium 2017
23 - 25 October 2017

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

原 悟史[†1†2]　渡辺 露文[†1†2]　田宮 和樹[†2]　吉岡 克成[†3]　松本 勉[†3]

概要：Mirai に代表される，組込み機器を対象とした 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 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，IoT マルウェアの持続的感染の成立条件を分析し実機による検証を行う．多くの組込み機器では，ファイルシステムは読み取り専用であり，機器上にマルウェアのバイナリをコピーし実行する従来の感染方法では，持続的感染は困難である．そこで，我々は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を，実機を用いて実証するとともに，このような攻撃を防止する方法について考察を行う．

キーワード：組込み機器，IoT，マルウェア，持続的感染

Analysis of factors for persistent infection of IoT malware and their substantiation using real devices

Satoshi Hara[†1†2]　Tsuyufumi Watanabe[†1†2]　Kazuki Tamiya[†2]
Katsunari Yoshioka[†3]　Tsutomu Matsumoto[†3]

Abstract: Most of the IoT malware which target embedded devices, such as Mirai, is known to be disinfected by rebooting the infected device. However, it has been reported that there are IoT malware which infect particular devices persistently and cannot be removed by system reboot. If a pandemic of these malware occur, more significant damage can be expected. In this paper, we analyze the factors for persistent infection of the IoT Malware and verify them with experiments using real device. For many embedded devices, persistent infection is difficult with traditional infection methods that copy and execute malware binaries on the device because of their file system is read-only. Therefore, we demonstrate persistent infection of IoT malware via firmware updating function using real device, and discuss how to prevent such attack.

Keywords: Embedded devices, IoT, Malware, persistent infection

1. はじめに

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．インターネット接続を有する組込み機器(以降では IoT 機器とよぶ)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]．従来，IoT 機器は，機器ごとに独自の 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 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．IoT 機器のファームウェアを対象にした大規模調査[2]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．さらには，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 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，管理画面や Telnet 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[3]. 2016 年に流行したマルウェア"Mirai"[4]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．

我々は，"Mirai"をはじめとする IoT 機器をターゲットとしたマルウェアが，感染した機器の電源を再起動することで，消滅することを複数の機器により実証している[5]．しかし，電源を再起動することで消滅する明確な理由や，消滅しないマルウェアの実現性については十分に議論できていない．特定の機器を対象とした，電源を再起動しても消えないマルウェアの事例[6]，機器を故障に至らせるマルウェアの事例[7]も報告されており，早急な対策が必要であると考えられる．

本研究では，Linux を使用した IoT 機器に対して，電源の再起動で消えない，"持続的"マルウェアの実現性の調査

†1 富士ソフト株式会社
FUJI SOFT INCORPORATED
†2 横浜国立大学
Yokohama National University.
†3 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
Graduate School of Environment and Information Sciences, Yokohama National University / Institute of Advanced Sciences, Yokohama National University

を実施する．文献[5]の調査対象機器の追跡調査を実施し，機器のファイルシステムが読み取り専用であることが，マルウェアの持続的感染を防ぐ効果をもたらしていることを示す．また，逆の視点として以下の要件が整えば，ファームウェア更新機能を介してマルウェアの持続的感染が容易に実現される恐れがあることを指摘し，実機の調査によりこれらの要件が満たされる場合が実際に存在することを示す．更に，特定機器について実際に持続的感染が成立することを実証する．

- 攻撃者はファームウェア更新機能を提供する機器のユーザインタフェースにアクセス可能であること
- 更新データの難読化が行われておらず，内部のファームウェアが容易に解析・改変できること
- ファームウェア更新時に，ファームウェアの正当性の検証が行われていないか，行われていても不十分なため，攻撃者は検証に合格する改変ファームウェアを作成できること

また，これらの調査結果をもとに，本研究では"持続的"マルウェアの防止について考察する．

2. 関連研究

これまで，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の提案[8][9]がされている．IoT 機器を標的としたマルウェアに関して，Telnet を介した感染が報告されている[10]．文献[5]では，マルウェアに感染した IoT 機器の電源の再起動により，マルウェアのプロセスを駆除できることを示した．一方で，一部機器では再起動によりマルウェアのプロセスは駆除できるものの，マルウェアのバイナリが残り続けることも示されている．しかし，これらの理由に踏み込んだ研究は行われていない．

そこで，本研究では電源の再起動によりマルウェアが駆除できる理由を解明し，電源の再起動では消えない，マルウェアの"持続的感染"の可能性について攻撃モデルの実証を行った．

3. 攻撃モデルの定義

IoT 機器へのマルウェアの持続的感染の実証実験を行うにあたり，攻撃対象機器と攻撃方法のモデル化を行う．

3.1 OS

IoT 機器の OS を大別すると，OS 機能を持たない「OS なし」，対象機器専用に作られた「独自 OS」，様々な機器にポーティングして使用される「汎用 OS」に分類される．IoT 機器のファームウェアを対象にした大規模調査[2]によると，86%の IoT 機器の OS は汎用 OS の 1 種である Linux であることから，本研究では汎用 OS を検討対象とする．

3.2 ファイルシステム

汎用 OS の多くは，機能の一部としてファイルシステムを有する．IoT 機器に感染するマルウェアの持続的感染の可否について検討する上で，当該機器のファイルシステムの種類が重要である．ファイルシステムは大きく分けて，UBIFS[11]のような読み書き可能なものと，squashfs[12]や cramfs[13]のような読み取り専用のものがある．読み書き可能なファイルシステムは事実上汎用コンピューターと同様の条件であり，マルウェア本体をファイルシステムにコピーして，機器が再起動したら，マルウェアが自動実行するようにシステム設定を書き換えればマルウェアの持続的感染が実現可能である．これは汎用コンピューターと同じで新たに議論する意義がないことから，本研究ではこのようなファイルシステムを持つ IoT 機器は検討対象としない．

本研究にあたり，Wi-Fi ルーターを対象として，2012 年 1 月から 2017 年 5 月に日本国内で発売された機器の OS およびファイルシステムを調査した．メーカーのホームページで確認が取れた範囲で Linux を使用した機器は 4 メーカー37 機種存在し，うち 18 機種のファイルシステムは squashfs であった．

3.3 ファームウェア更新機能

読み取り専用のファイルシステムを持つ機器に対してマルウェアの持続的感染が実現できるかどうかが，本研究の議論の対象となる．ただし，真にシステムが読み取り専用であれば，マルウェアの持続的感染は実現不可能なはずであるので，何らかの付加的条件が必要となる．具体的には，機器がファームウェア更新機能を有しているという条件が考えられる．

読み取り専用のファイルシステムを持つ機器のファームウェア更新処理は，ファームウェアをインストールするためのプログラム(以下，インストーラー)により，更新前のファイルシステムを更新後の新しいファイルシステムで上書きすることで行われる．新しいファイルシステムはネットワーク越しに更新データとして取得するかポータブルメディア等を介して取得する．また，インストーラーには以下のような実現形態がある．

3.3.1 ファームウェア同梱型インストーラー

更新データの中に，更新後のファイルシステムに加えてインストーラーが組み込まれており，機器側はこのプログラムを無条件に実行するだけで更新が行われる．インストーラーは更新データの中に含まれるため，インストーラー自体を改ざんされる恐れがある．

3.3.2 機器内蔵型インストーラー

更新データには，更新後のファイルシステムしか入って

おらず，それをインストールする機能は機器本体側に実装される．機器内蔵型インストーラーは，以下の 3 つに分類される．

単純型

インストーラーを含めたファイルシステムを丸ごと書き換える．インストーラーが更新データの中に含まれるため，インストーラー自体を改ざんされる恐れがある．

マルチドライブ型

ファイルシステムの領域が，インストーラーのみの領域(A 面)とそれ以外の領域(B 面)の 2 面に分かれている．インストーラーは B 面のみを書き換える．更新データにインストーラーを含める必要がなく，インストーラー自体の改ざんを防止できる．

マルチ OS 型

ファイルシステムの領域が，インストール専用の OS(A 面)と，機器の通常動作用の OS(B 面)の 2 面から構成される．ファームウェア更新時は，A 面の OS を起動し，B 面のみを書き換える．機器の通常動作時は B 面の OS が起動する．更新データにインストーラーを含める必要がなく，インストーラー自体の改ざんを防止できる．

3.4 攻撃成立条件

マルウェアの持続的感染を実現する方法としては，以下が考えられる

A) ファームウェア更新機能を使用してマルウェア入りのファームウェアをインストールする

B) ファームウェア更新機能を持つプログラムに脆弱性がある場合，これを乗っ取り，ファームウェア更新機能も乗っ取ることで，マルウェア入りのファームウェアをインストールする

A)の場合，以下が攻撃の成立条件となる．

i). ファームウェア更新機能を攻撃者が利用できること
ファームウェア更新機能は WebUI として実装されていることが多い．この場合，WebUI にログインし更新機能を利用できることが条件となる

ii). マルウェア入りファームウェアを用意できること
更新はファイルシステム全体を上書きすることで行われるため，攻撃者はマルウェア入りファームウェアを用意する必要がある．そのため，機器を解析して取得したり，機器メーカーのホームページ等から取得するといった方法で，攻撃者が当該機器用のファイルシステムを入手できることが条件となる

iii). マルウェア入りの偽ファームウェアが機器に受け入れられること
機器のインストーラーが，更新用の新しいファイルシステムの正当性検証を行っている場合は，その検証に合格するような更新データを用意することが条件となる

B)の場合は，脆弱性に依存するが，攻撃者がファームウェア更新機能を乗っ取っていることから，上記 i)相当の条件は既に満たしている．そのため，ii),iii)を満たしていれば，攻撃が成功する．

4. 先行研究の考察

先行研究[5]におけるマルウェアの感染方法は，機器のファイルシステム上にマルウェアのバイナリを転送し実行する手法である．そのため，3 章で示したとおり，「読み書き可能」のファイルシステムを持つ機器には持続的感染の可能性があるが，「読み取り専用」のファイルシステムを持つ機器への持続的感染はしないと考えられる．

当該研究の実験使用機器および結果を表 1 に示す．本研究で新たに各機器のファイルシステムとインストーラー型を調査した結果を併記する．

表 1 機材一覧

機器	種類	電源OFFによるマルウェア駆除	ファイルシステム	ファームウェアのインストーラー
A	IP Camera	プロセス・バイナリともに消滅	不明	不明
B	プリンター	プロセスのみ消滅 バイナリは残る	UBIFS (※読み書き可能)	不明
C	ルータ	プロセス・バイナリともに消滅	squashfs (※読み取り専用)	本体内蔵型
D	WiFiストレージ /ポケットWiFI	プロセス・バイナリともに消滅	squashfs (※読み取り専用)	ファームウェア同梱型
E	WiFiストレージ /ポケットWiFI	プロセス・バイナリともに消滅	squashfs (※読み取り専用)	ファームウェア同梱型
F	WiFiストレージ /ポケットWiFI	プロセス・バイナリともに消滅	squashfs (※読み取り専用)	ファームウェア同梱型
G	衛星放送受信機	プロセス・バイナリともに消滅	cramfs (※読み取り専用)	本体内蔵型

「読み取り専用」のファイルシステムを持つ機器 C， D, E，F においては，いずれのマルウェア検体においても，持続的感染は発生していないことが示されている．

一方，「読み書き可能」ファイルシステムを持つ機器 B は，電源の再起動によりマルウェアの"プロセス"を駆除できたことが示されているが，"マルウェアバイナリファイル"については，駆除できないことが示された．プロセスを駆除できた理由として，以下の 2 つの要因が推測される．

(1). 機器の再起動時に，自身を自動起動するようなシステム設定の改変をマルウェアが行っていない

(2). システム設定のアクセス権が適切に設定されており，マルウェアによる改変を防止した

5. 実証実験

続いて，3 章で示したモデルに該当する機器として，機器 E を対象にファームウェア更新機能を介したマルウェアの持続的感染の検証を行う．機器 E は以下の特徴を持つ．

- OS： 汎用 OS (Linux)
- ファイルシステム： 読み取り専用(squashfs)
- WebUI を介したファームウェア更新機能を有する
- インストーラーはファームウェア同梱型である
- WebUI は WAN 側からログイン可能であり，WebUI のデフォルトのユーザーID/パスワードは全製品で共通であり，説明書に明記されているため，攻撃者は容易にログイン可能である
- メーカーのホームページにファームウェアが公開されている

5.1 ファームウェアの構成

Binwalk[14]を用いて，機器 E のファームウェアの構成を調査した結果を図 1 に示す．ファームウェアは以下の 3 部構成であった．

シェルスクリプト

テキスト形式で記載される bash スクリプトである．ファームウェア更新のシーケンスを担う．

Linux ファイルシステム

インストール時に使用するファイルシステムであり，ROM への書き込み機能を持つ．

更新データ

ファイルシステム形式(squashfs)であり，ROM に書き込まれるデータである．Linux ファイルシステムの内部に含まれる．

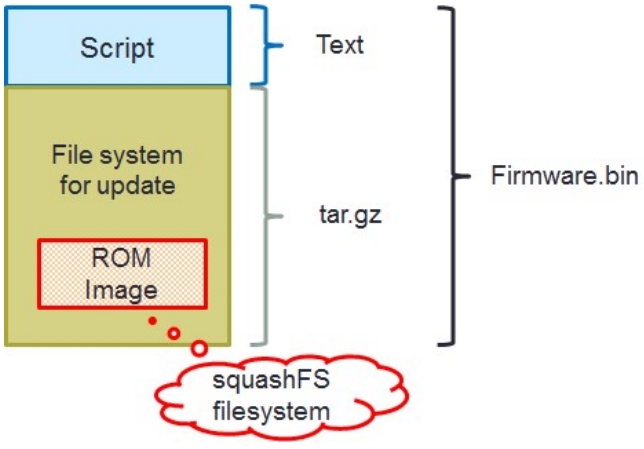

図 1 機器 E ファームウェアフォーマット

「シェルスクリプト」と，「Linux ファイルシステム」は更新データを ROM に書き込む際に使用されるものであり，これらをあわせて広義のインストーラーであると言える．

5.2 持続的マルウェア感染の実証

メーカーのホームページより正規のファームウェアを入手のうえ，疑似悪性プログラムの埋め込みと当該プログラムが機器の起動時に自動実行するようにシステム設定の書き換えを行った．当該機器は汎用ファイルシステムである squashfs を使用しており，squashfs 向けツールである sasquatch[15]，squashfs-tools を用いた解凍・再構築が容易であった．

WAN 側から当該機器の WebUI へログインし，改変したファームウェアへの更新が成功することを確認した．さらに機器の再起動を繰り返しても，疑似悪性プログラムが毎回起動することを確認した．

また，更新データをでたらめな内容に改変してファームウェア更新を行うことにより，機器が起動できなくなることをあわせて確認した．

5.3 インストーラー改変の実証

インストーラーの改変の実証として，シェルスクリプト部に任意の OS コマンドを埋め込み，ファームウェア更新時に当該コマンドが実行されることを確認した．この改変によりマルウェアを実行することが可能であるが，5.2 節で説明した持続的感染ではないため，機器の再起動時にはマルウェアは消滅する．

6. 考察

3.4 章に示した攻撃成立条件を満たすことで，実際にマルウェアの持続的感染が実現されることを，実機を用いて証明した．以下では，持続的感染を防止する方法について考察を行う．

i). ファームウェア更新機能を攻撃者が利用できることに対する対策

被攻撃対象モデルでは，WAN 側から WebUI にアクセスが可能であった．また，デフォルトのユーザーID/パスワードは誰でも知り得る状態であった．対策方法として，以下が挙げられる．

(1). WAN 側から WebUI へアクセスをできなくすることで，攻撃の機会を減らす

(2). 機器の使用前にユーザーID/パスワードの変更を促す

(3). 機器の出荷時に，個体ごとに異なるユーザーID/パスワードを設定する

ii). マルウェア入りファームウェアを用意できることに

対する対策

被攻撃対象モデルではLinux向けの汎用のファイルシステムを使用していたこと，ファームウェアの難読化がされていなかったことから，汎用ツールを用いて容易に解析・改変が可能であった．難読化等でファームウェアの解析のコストを高める対策や限定的なファームウェア公開が有効と考えられる．

iii). 偽のファームウェアが機器に受け入れられることに対する対策

被攻撃対象モデルでは，ファームウェアの正当性を確認せず更新を行う仕組みであった．対策として，ファームウェア更新時に更新データを検証する仕組みが必要である．典型的にはディジタル署名が考えられる．しかし，インストーラーの手法によっては，更新データ内にインストーラーが含まれるため，インストーラー自体を改ざんされる恐れがあり，対策が不十分である．そのため，インストーラー自体の改変がより困難な「マルチドライブ型」もしくは「マルチOS型」にするべきである．

7. まとめと今後の課題

本研究では，Linuxを使用したIoT機器は，読み取り専用のファイルシステムを用いることで，攻撃者もしくはマルウェア感染デバイスが，被攻撃機器にマルウェアをダウンロードする攻撃に対して，持続的感染を防止する効果がある一方で，ファームウェア改変による感染は持続的感染を引き起こす可能性があることを示した．実機を用いた実証では，改変したファームウェアを用いてファームウェア更新を実施することにより，悪性プログラムの持続的感染が実現できることや機器を故障に至らせることを示した．さらにインストーラーがファームウェアに含まれる場合，インストーラーの改変による攻撃の可能性を実証した．これらの攻撃の成立条件を分析し，対策方法の提案を行った．本研究成果のうち特定機器の解析結果については，情報処理推進機構(IPA)への報告を行う予定である．

今後の課題としては，持続的感染の成立を防ぐ手法の実証を行いたい．本研究では市販の製品を用いて実証を試みたが，機器を破損した際の復旧を行えない問題がある．市販の評価基板を用いて被攻撃対象モデルを実装のうえ，実証を行いたい．

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．

参考文献

[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] Andrei Costin, Jonas Zaddach, Aurélien Francillon, and Davide Balzarotti, Eurecom, A Large-Scale Analysis of the Security of Embedded Firmwares, 23rd USENIX Security Symposium(2014)

[3] "ルータや IoT 機器に危険をもたらす管理用機能の放置", http://blog.trendmicro.co.jp/archives/14195 , (参照 2017-7-24).

[4] "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).

[5] 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

[6] "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).

[7] "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).

[8] Yin Minn Pa Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, C. Rossow, “IoTPOT: Analysing the Rise of IoT Compromises ”, USENIX/WOOT’15, 2015

[9] 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”, 電子情報通信学会信学技報 vol. 115 no. 488，ICSS2015-47，pp. 1-6，2016

[10] 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

[11] "UBIFS", http://www.linux-mtd.infradead.org/doc/ubifs.html , (参照 2017-7-24).

[12] "squashfs", http://squashfs.sourceforge.net/ , (参照 2017-7-6).

[13] "filesystems/cramfs.txt", http://archive.linux.or.jp/JF/JFdocs/kernel-docs-2.6/filesystems/cramfs.txt.html , (参照 2017-7-24).

[14] "Binwalk", https://code.google.com/p/binwalk/ , (参照 2017-07-20).

[15] "sasquatch", https://github.com/devttys0/sasquatch , (参照 2017-7-31).