L-002

IoT 機器におけるセキュア OS の適用可否と保護機能の評価
Evaluation on Applicability and Protection of LSM-based MAC Systems in IoT Devices

三木　雅登[1)]　　山内　利宏[2)]
Masato Miki[1)]　Toshihiro Yamauchi[2)]

1　はじめに

IoT（Internet of Things）機器の普及 [1] に伴い，IoT 機器を標的とする攻撃が増加している．しかし，IoT 機器では搭載メモリ量が限られ [2]，アンチウイルスソフトウェアの実行時にメモリが不足する可能性が高い [3]．また，IDS やファイアウォールでは，環境の変化に伴い通信内容が変化する IoT 機器を効果的に保護できない [3]．

IoT 機器では Linux が OS として搭載されることが多い [4] ものの，LSM（Linux Security Modules）ベースのセキュア OS を含む Linux のセキュリティ機能の多くは IoT 機器に導入されていない [5]．

そこで，本研究では，IoT 機器の保護において LSM ベースのセキュア OS が有効か否かを明らかにするため，IoT 機器におけるセキュア OS の適用可否と保護機能を調査した．具体的には，IoT 機器におけるセキュア OS の適用可否に関わる条件としてファイルシステムごとに既存のセキュア OS が適用可能か否かを調査した．

また，適用したセキュア OS が Mirai の手法を模擬した攻撃からシステムを保護できるか調査した結果を報告する．

2　IoT 機器におけるセキュリティ機能

2.1　IoT 機器の保護に必要なセキュリティ機能

IoT 機器は CPU 負荷や搭載メモリ量の削減が求められ [2]，メモリ使用量が多いセキュリティ機能は IoT 機器で動作しない場合が多いと推察する．このため，IoT 機器の開発上の制約を満たすセキュリティ機能が必要である．

セキュア OS である SELinux は，設定の変更によりメモリ使用量を 1MB 未満に削減できる [6]．一方，IoT 機器向けのアンチウイルスソフトウェア Commtouch Antivirus は，128MB のメモリを消費する [3]．このため，セキュア OS のメモリ使用量は少ないと推察する．また，セキュア OS は，処理の遅延が小さい [7]．

Mirai はボットネットの構築を目的に IoT 機器に侵入する．IDS やファイアウォールは侵入後の活動を制御できないものの，セキュア OS は MAC により侵入後の活動を制限でき，攻撃の被害を抑制できる可能性がある．

2.2　LSM ベースのセキュア OS

LSM とは，Linux に組み込まれたセキュリティフレームワークであり，カーネルのセキュリティチェック機能の拡張に用いる．セキュア OS は，強制アクセス制御（Mandatory Access Control，以降，MAC）と最小特権を実現する機能を備えた OS を指す．

セキュア OS は，MAC によりプロセスを最小の権限で実行できる．多くの OS で採用されている任意アクセス制御（以下，DAC）は root 権限を持つユーザが全ての資源にアクセスできる．一方，MAC はアクセス可能な資源，および操作項目をセキュリティポリシによって定める．セキュリティポリシは root 権限を持つユーザにも適用され，root 権限を持つ場合でも，悪意のあるプロセスの操作を制限できる．主な LSM ベースのセキュア OS を以下に示す．

1. SELinux [8]：ラベルベースのセキュア OS であり，ファイルやプロセスごとにタイプというラベルを与え，セキュリティポリシに定義がないアクセスを全て拒否する TE（Type Enforcement）によりシステムを保護する．
2. Smack [9]：ラベルベースのセキュア OS であり，TE でシステムを保護する．
3. TOMOYO Linux [10]：パス名ベースのセキュア OS であり，プロセスの実行履歴を基に，実行を許可する操作をセキュリティポリシに記録できる．
4. AppArmor [11]：パス名ベースのセキュア OS であり，プロファイルという設定ファイルをプログラムごとに定義してセキュリティポリシの設定を行う．

3　IoT 機器におけるセキュア OS の適用可否

3.1　セキュア OS の適用可否に関する調査内容

一部のセキュア OS の適用可否は，ファイルシステムに依存する．SELinux は，ファイルシステムにおける xattr のサポートの有無により保護単位が変化する [12] ものの，それ以外のセキュア OS における保護単位の変化は不明である．また，IoT 機器で利用されるファイルシステムでの実際の動作も不明である．このため，調査対象のセキュア OS について，IoT 機器で利用されるファイルシステムにおける適用可否を調査した．

3.2　調査対象のセキュア OS とファイルシステム

3.2.1　セキュア OS

調査対象のセキュア OS を表 1 に示す．調査対象のセキュア OS は，“major” という区分に分類されるセキュア OS とする．この区分のセキュア OS は，ファイルやプロセスへの MAC を提供する．

調査対象のセキュア OS は，ラベルベースとパス名ベースのセキュア OS に分けられる．ラベルベースのセキュア OS は，アクセス制御にラベルという属性を用い，これを xattr に格納する．xattr をサポートしないファイルシステムでは，ファイル単位の MAC が適用さ

1）岡山大学大学院自然科学研究科，Graduate School of Natural Science and Technology, Okayama University
2）岡山大学学術研究院自然科学学域，Faculty of Natural Science and Technology, Okayama University

表 1　調査対象のセキュア OS

セキュア OS	制御方式	アクセス制御の特徴
SELinux	ラベルベース	許可したアクセス以外は全て拒否
Smack	ラベルベース	許可したアクセス以外は全て拒否
TOMOYO Linux	パス名ベース	許可したアクセス以外は全て拒否
AppArmor	パス名ベース	プロファイルの有無で制御，かつ，存在する場合はその内容で制御

表 2　調査対象のファイルシステム

ファイルシステム	xattr のサポート	読み書きの可否
Cramfs	無	読み込み専用
romfs	無	読み込み専用
JFFS2	有	読み書き可能
SquashFS	有	読み込み専用

れない [12]．しかし，IoT 機器で利用されるファイルシステムにおいて，ファイル単位の MAC を適用可能か否か明らかでない．パス名ベースのセキュア OS は，アクセス制御にパス名を用い，xattr のサポートがなくても適用できる．

3.2.2　ファイルシステム

調査対象のファイルシステムを表 2 に示す．調査対象として，IoT 機器での利用が多く確認されているものを選択した．文献 [13] で，調査された IoT 機器のファームウェアでは，調査対象のファイルシステムが全体を占めている．

3.3　セキュア OS の適用可否の調査方法

3.3.1　セキュア OS の適用可否を判断する条件

調査対象のファイルシステムにおいて，セキュア OS を適用可能か否かを明らかにする．このために，ファイルシステムごとに各セキュア OS でファイルやプロセスへの MAC が有効に動作するか否かを調査した．

3.3.2　実験環境の構築

実験環境を表 3 に示す．Raspberry Pi 3 B+，および Ubuntu Server 20.04.3 LTS は，IoT 機器のデバイスとして広く利用されている．

3.3.3　セキュア OS の適用可否の評価方法

セキュア OS が有効に動作するか否かを確認するため，各ファイルシステム内ファイルの読み込みが制限されるか確認した．読み込み対象のファイルは，セキュア OS により読み込みが制限されるファイル（以下，制限ファイル）と制限されないファイル（以下，非制限ファイル）に分けた．アクセス制御に DAC が関与しないよう，全てのユーザに各ファイルの読み込みを許可した．また，各セキュア OS でセキュリティポリシを修正し，ファイルの読み込みを制御する．

ファイルの読み込みが非制限ファイルで成功し，かつ，制限ファイルで失敗した場合，ファイル単位の MAC が有効とする．そうでない場合，ファイル単位の MAC が無効とする．

表 3　セキュア OS の適用可否の実験環境

デバイス名	Raspberry Pi 3 B+
OS	Ubuntu Server 20.04.3 LTS (64bit) （TOMOYO Linux 以外） Ubuntu Server 20.04.3 LTS (32bit) （TOMOYO Linux）
カーネルバージョン	5.4.0-1047-raspi （TOMOYO Linux 以外） 5.4.83-v7+ （TOMOYO Linux）

表 4　各ファイルシステムにおけるセキュア OS の MAC の保護単位（プロセスの制御単位は非考慮）

セキュア OS	Cramfs	romfs	JFFS2	SquashFS
SELinux	ファイルシステム	ファイルシステム	ファイル	ファイル
Smack	ファイルシステム	ファイルシステム	ファイル	ファイル
TOMOYO Linux	ファイル	ファイル	ファイル	ファイル
AppArmor	ファイル	ファイル	ファイル	ファイル

3.4　セキュア OS の適用可否の調査結果

各ファイルシステムにおける各セキュア OS の MAC の保護単位の評価結果を表 4 に示す．

1. SELinux
   JFFS2 と SquashFS では，非制限ファイルの読み込みに成功し，制限ファイルの読み込みに失敗したことから，SELinux が適用できることを確認した．一方，Cramfs と romfs では，実験用ディレクトリと内部のファイルのラベルが全てラベル `romfs_t` となり，全ファイルの読み込みに失敗した．これより，ファイルシステム単位のアクセス制御であることを確認した．また，ラベル `romfs_t` はマウント時に付与され，読み込み専用のファイルシステムでは変更できない．
2. Smack
   SELinux と同様の結果が得られたものの，Cramfs と romfs では実験用ディレクトリと内部のファイルのラベルが全てラベル_（以下，ラベル `floor`）となり，全ファイルの読み込みに成功した．
3. TOMOYO Linux
   全てのファイルシステムにおいて，非制限ファイルの読み込みに成功し，制限ファイルの読み込みに失敗した．このため，TOMOYO Linux は，Cramfs，romfs，JFFS2，および SquashFS で適用できる．
4. AppArmor
   TOMOYO Linux と同様の結果が得られたため，AppArmor は，Cramfs，romfs，JFFS2，および SquashFS で適用できる．

3.5　IoT 機器におけるセキュア OS の適用可否の考察

調査の結果より，ラベルベースのセキュア OS とパス名ベースのセキュア OS では，各ファイルシステムにおける適用可否が異なる．

1. ラベルベースのセキュア OS
   実験結果より，ファイル単位の MAC を動作させるためには，ファイルシステムにおける xattr のサポートが必要である．ファイルシステム単位の MAC のみが動作する場合，単一のファイルシステ

表 5　セキュア OS の保護機能の実験環境

デバイス	標的 IoT 機器	攻撃者サーバ兼 標的サーバ用デバイス
デバイス名	表 3 に同じ	Endeavor MR4900
OS	表 3 に同じ	Fedora 34 Workstation
カーネルバージョン	表 3 に同じ	5.15.6-100.fc34.x86_64

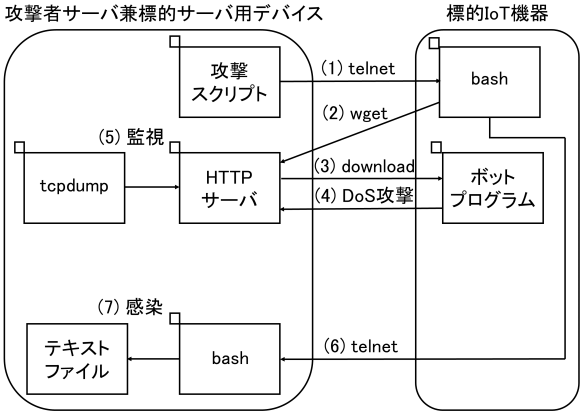

図 1　攻撃実験の流れ

ムで構成されるシステムでは，システムの動作に必要なファイルと不要なファイルで異なるアクセス制御ができないため，システムの動作と攻撃からの保護を両立できない可能性が高い．

2. パス名ベースのセキュア OS
   xattr のサポートの有無に関わらず，実験対象のファイルシステム全てでファイル単位の MAC が有効に動作した．これは，パス名ベースのセキュア OS がファイルシステムにおける xattr のサポートを必要としないためであり，任意のファイルシステムで適用できる可能性が高いことを示している．

4　IoT 機器におけるセキュア OS の保護機能の評価

4.1　セキュア OS の保護機能に関する調査の内容と方法

既存のセキュア OS が Mirai の攻撃からシステムを保護できるか調査するため，攻撃実験を行った．攻撃実験に用いた環境を表 5 に示す．攻撃実験は，インターネットに接続しない環境で行った．また，図 1 に攻撃実験の流れを示す．図 1 に示す攻撃手法は，Mirai のソースコード [14] を参考にした．

攻撃者サーバ兼標的サーバ用デバイス（以下，攻撃／標的サーバ）に HTTP サーバを導入する．HTTP サーバは，DoS 攻撃用のボットプログラム（以下，ボット）配布用のマルウェア配布サーバ（以下，ローダ）と DoS 攻撃の標的サーバを兼ねる．標的 IoT 機器で各セキュア OS を有効化し，攻撃実験の成否を確認することで，各セキュア OS がシステムを保護可能か否かを調査した．攻撃の流れの詳細を以下に示す．

1. 標的 IoT 機器へのリモートログイン
   攻撃スクリプトを実行して，`telnet` でリモートログインする．ログインするユーザは，root ユーザ root，Ubuntu Server 20.04.3 LTS の標準の管理者ユーザ ubuntu，および一般ユーザ user とする．

表 6　標準設定での各セキュア OS における攻撃の成否

セキュア OS	リモートログイン	DoS 攻撃	他ホスト感染
SELinux	失敗	未観測	未観測
Smack	成功	成功	成功
TOMOYO Linux	成功	成功	成功
AppArmor	成功	成功	成功

表 7　修正した設定での各セキュア OS における攻撃の成否

セキュア OS	リモートログイン	DoS 攻撃	他ホスト感染
Smack	失敗	一部成功	失敗
TOMOYO Linux	失敗	失敗	失敗
AppArmor	失敗	失敗	失敗

2. ボットのダウンロード要求の送信
   `wget` を用いてローダからボットを入手するよう指示する．
3. ボットのダウンロード
   リクエストを受けたローダは，ボットを標的 IoT 機器に送信する．
4. DoS 攻撃
   ボットの実行を DAC で許可する．ボットを実行して，標的サーバに DoS 攻撃を模擬した HTTP パケットを 10,000 件送信することにより，サーバに負荷を与える．
5. DoS 攻撃の観測
   `tcpdump` を利用して，DoS 攻撃による HTTP パケットの送信を観測する．
6. 他ホストへの感染
   攻撃／標的サーバに `telnet` で一般ユーザ exp としてリモートログインする．
7. 感染した他ホストでの活動
   攻撃／標的サーバ単体で自身を攻撃し続けることを防ぐため，図 1 の (2)，(3)，(4)，および (6) を行う代わりに，カレントディレクトリに感染の証拠となるテキストファイルを作成する．

4.2　セキュア OS の保護機能に関する調査結果

実験では，図 1 に示す攻撃の流れにおける (1) から (7) までのいずれかの段階で攻撃が失敗するか分析した．この結果，標準設定のセキュア OS ごとに攻撃を防ぐ段階が異なることを確認した．標準設定の各セキュア OS における攻撃の成否を表 6 に示す．また，標準設定の場合，攻撃全てが成功した Smack，TOMOYO Linux，および AppArmor について，攻撃に関わるプログラムの実行を禁止するよう設定を変更して再実験した．この結果，表 7 に示すように，攻撃の成否に変化が見られた．

4.2.1　SELinux

図 1 の (1) において，`telnet` によるリモートログインが失敗した．これにより，標的 IoT 機器で行う全ての操作が不可能になる．これは，標準設定の SELinux が攻撃実験における全ての攻撃からの保護に有効であることを示す．

4.2.2　Smack

標準設定では，全ての攻撃が成功した．これは，標準設定において，プロセスやファイルに与えられるラベルのほとんどがラベル `floor` であり，互いにアクセスする

ことが許可されているためである．

リモートログインに関わる in.telnetd の実行のみ禁止した場合，図 1 の (1) が失敗し，残りも全て失敗した．wget の実行のみ禁止した場合，図 1 の (2) の失敗により，DoS 攻撃が失敗した．他ホスト感染に関わる telnet の実行のみ禁止した場合，図 1 の (6) の失敗により，他ホスト感染が失敗した．chmod の実行のみ禁止した場合，ユーザ root 以外では図 1 の (4) の失敗により，DoS 攻撃が失敗したものの，ユーザ root では全ての攻撃が成功した．これは，root 権限が MAC における権限を強化したことが原因であると推察できる．一方，ネットワークに関連する攻撃が失敗したことから，ネットワークに関連する処理において，MAC の権限の強化が発生しない可能性が考えられる．

4.2.3 TOMOYO Linux

標準設定では，全ての攻撃が成功した．これは，標準のセキュリティポリシがない TOMOYO Linux で，アクセス制御を実施しない初期化された設定が使用されるためである．in.telnetd，wget，および telnet のそれぞれを実行のみ禁止した場合，攻撃の成否の変化は，Smack と同様の結果になった．chmod の実行のみ禁止した場合，ユーザに関わらず，図 1 の (4) の失敗により，DoS 攻撃が失敗した．

4.2.4 AppArmor

標準設定では，全ての攻撃が成功した．これは，プロファイルのないプログラムの操作が全て許可される仕様により，標準設定ではプロファイルが用意されていない攻撃に関わるプログラムをアクセス制御しなかったためである．攻撃に関わるプログラムの実行を禁止するよう修正した場合，TOMOYO Linux と同様に，全ての攻撃が失敗した．

4.2.5 IoT 機器におけるセキュア OS の保護機能に関する考察

標準設定では，SELinux が他のセキュア OS と比較してシステムを保護できる可能性が高い．SELinux 以外の全てのセキュア OS は，ネットワークに関する操作を保護できる．しかし，Smack はユーザ root による chmod が許可された．このため，各セキュア OS は，ネットワークに関連する処理を同等に保護できるものの，Smack は権限昇格後の操作に脆弱である．Smack では，Linux Capability におけるケーパビリティ CAP_MAC_ADMIN が root 権限を持つユーザに付与される．Smack における root 権限を持つユーザの操作の許可は，ケーパビリティ CAP_MAC_ADMIN により Smack の MAC をバイパスしたためと推察する．このため，攻撃者が root 権限を取得した場合，Smack は，TOMOYO Linux と AppArmor より脆弱である可能性が高い．

5 おわりに

本稿では，IoT 機器の保護において LSM ベースのセキュア OS が有効か否かを明らかにするため，IoT 機器におけるセキュア OS の適用可否と保護機能を調査した．適用可否を調査するため，IoT 機器で利用されるファイルシステムにおける各セキュア OS の MAC の保護単位を確認した．保護機能を調査するため，Mirai と同様の手法で攻撃実験を行い，各セキュア OS がシステムを保護できるか確認した．調査の結果，xattr をサポートしないファイルシステムではラベルベースのセキュア OS を適用できないことが明らかとなった．また，Smack，TOMOYO Linux，および AppArmor は，標準設定で Mirai に基づく手法の攻撃からシステムを保護できないことが明らかとなった．

謝辞

本研究の一部は，JST さきがけ JPMJPR1938 の助成を受けたものです．

参考文献

[1] Philip Sparks: The route to a trillion devices, available from ⟨https://community.arm.com/cfs-file/__key/telligent-evolution-components-attachments/01-1996-00-00-00-01-30-09/Arm-_2D00_-The-route-to-a-trillion-devices-_2D00_-June-2017.pdf⟩ (accessed 2022-1-11).

[2] 桑田雅彦：IoT における制約，コンテキスト，及びリスクを意識した適応的なアクセス制御方式の構想の提案，第 15 回情報科学技術フォーラム（FIT2016）講演論文集，Vol.4，pp.109–114（2016）．

[3] Yu, T., Sekar, V., Seshan, S., Agarwal, Y., Xu, C.: Handling a Trillion (Unfixable) Flaws on a Billion Devices: Rethinking Network Security for the Internet-of-Things, *Proc. 14th ACM Workshop on Hot Topics in Networks*, No.5, pp.1-7 (2015).

[4] iot.eclipse.org: IoT Developer Survey 2019 Results, available from ⟨https://iot.eclipse.org/community/resources/iot-surveys/assets/iot-developer-survey-2019.pdf⟩ (accessed 2022-1-5).

[5] 白石周碁，福本淳文，吉元亮太，塩治榮太朗，秋山満昭，山内利宏：ソフトウェア解析とベンダインタビューによる IoT 機器のセキュリティに関する大規模実態調査，コンピュータセキュリティシンポジウム 2020（CSS2020）論文集，Vol.2020，pp.875–882（2020）．

[6] Nakamura, Y., Sameshima, Y., Yamauchi, T.: Reducing Resource Consumption of SELinux for Embedded Systems with Contributions to Open-Source Ecosystems, *Journal of Information Processing*, Vol.23, No.5, pp.664-672 (2015).

[7] Zhang, W., Liu, P., Jaeger, T.: Analyzing the Overhead of File Protection by Linux Security Modules, *Proc. 2021 ACM Asia Conference on Computer and Communications Security*, pp.393 –406 (2021).

[8] SELinux Wiki, available from ⟨http://selinuxproject.org/page/Main_Page⟩ (accessed 2022-1-5).

[9] The Smack Project: Home, available from ⟨http://schaufler-ca.com⟩ (accessed 2022-1-5).

[10] TOMOYO Linux: TOMOYO Linux ホーム, available from ⟨https://tomoyo.osdn.jp/⟩ (accessed 2022-4-28).

[11] Ubuntu Wiki: AppArmor, available from ⟨https://wiki.ubuntu.com/AppArmor⟩ (accessed 2022-1-5).

[12] SPDL 仕様書 ver 2.1.：9 SELinux がサポートしないファイルシステムへのアクセス制御: allowfs, 入手先 ⟨http://seedit.sourceforge.net/doc/2.1/spdl_spec_jp/node10.html⟩（参照 2022-1-5）．

[13] Liu, K., Yang, M., Ling, Z., Yan, H., Zhang, Y., Fu, X., Zhao, W.: On Manually Reverse Engineering Communication Protocols of Linux-Based IoT Systems, *IEEE Internet of Things Journal*, Vol.8, No.8, pp.6815-6827 (2021).

[14] GitHub: jgamblin/Mirai-Source-Code: Leaked Mirai Source Code for Research/IoC Development Purposes, available from ⟨https://github.com/jgamblin/Mirai-Source-Code⟩ (accessed 2022-1-5).