Vol.2021-CSEC-93 No.12
Vol.2021-IOT-53 No.12
2021/5/13

LKMを介したSeccompフィルタの適用によるアクセス制御手法の提案と評価

山内 利宏[1,2]　吉元 亮太[3]

概要：Mirai などのマルウェアによる IoT 機器への攻撃が増加している．IoT 機器は今後も増加すると考えられるため，IoT 機器で利用されるソフトウェアに脆弱性が見つかった場合に，その対策を実行できることが重要である．そこで，Linux カーネルやアプリケーションを改変することなく，脆弱性のあるアプリケーションを保護することを目的にして，発行できるシステムコールを制限する Seccomp の Berkeley Packet Filter (BPF) をプロセスに適用する機能を Loadable Kernel Module（LKM）に実現し，アクセス制御する手法を提案する．本稿では，提案手法の実現方式を示し，攻撃防止実験により，提案手法の有効性を評価し，提案手法の有効性と限界について議論した結果を報告する．

Yamauchi Toshihiro[1,2]　Yoshimoto Ryota[3]

1. はじめに

Mirai [1] などのマルウェアによる IoT 機器への攻撃が増加しており，IoT 機器が脅威にさらされている．IoT 機器を狙った代表的なマルウェアである Mirai は，推測が容易なパスワードを使用する IoT 機器などを対象に，Telnet などを利用して侵入する．Mirai などのマルウェアは，Telnet などで IoT 機器へのログインに成功すると，目的を達成するためにボットなどのマルウェアを IoT 機器にダウンロードさせ，感染する．これにより，IoT 機器をボット化するなどの活動を行う．ボット化した IoT 機器は，感染の拡大化のために，他の IoT 機器に同様の手法で侵入を試みることがある．また，攻撃者は，このようにして作成したボットネットを用いて，DDoS 攻撃を行うことがある．

IoT 機器への攻撃は，既知のログイン情報を用いるものや，簡単なパスワードを用いて侵入するものが多かったが，IoT 機器のソフトウェアの脆弱性を悪用して侵入し感染する事例がある．今後は，後者のソフトウェアの脆弱性を悪用した攻撃が増加すると考えており，ソフトウェアに脆弱性が見つかったとしても，容易に感染しない防御機能をの萎えていることや，脆弱性に対して対策できる IoT 機器向けのセキュリティ機能が必要不可欠である．

そこで，本稿では，脆弱性のあるアプリケーションを保護することを目的にして，Linux カーネルやアプリケーション（以降，AP と略す）を改変することなく，発行できるシステムコールを制限する Seccomp の Berkeley Packet Filter（BPF）をアプリケーションに適用する機能を Loadable Kernel Module（LKM）に実現し，アクセス制御する手法を提案する．また，提案手法の有効性を評価し，いくつかの脆弱性攻撃を防止できることを確認できたことを報告する．

[1] 岡山大学学術研究院
Okayama University
[2] JST さきがけ
JST PRESTO
[3] 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University

2. Seccomp と IoT 機器の特徴

2.1 Seccomp とは

Seccomp（Secure Computing）は，Linux のセキュリティ機能の一つである [2]．Seccomp により，プロセスが発行するシステムコールを制限できる．システムコールが発行された際の処理を BPF で記述し，`prctl()`，または `seccomp()` システムコールを発行することにより，これを発行したプロセスに Seccomp の BPF（以降，Seccomp フィルタと略す）を適用する．Seccomp フィルタは，システムコール発行後に，システムコール処理実行前に処理される．Seccomp フィルタにより，システムコールを実行させず，エラーとしたり，発行プロセスを kill することができる．

Seccomp は，2005 年にリリースされた Linux 2.6.12 から導入されている．このバージョンの Seccomp では，`exit()`，`sigreturn()`，及び開いているファイルディスクリプタへの `read()`，`write()` システムコールのみが発行できるように制限される．また，2012 年にリリースされた Linux 3.5 から．Seccomp フィルタを用いてシステムコールをフィルタリングできる機能が導入されている．

2.2 IoT 機器の特徴と課題

IoT では，従来のパソコン（PC）などの情報機器だけでなく，センサや様々なモノがインターネットに接続され，データが集められ，そのデータを分析することでより高度な処理を実現することが期待されている．PC などの情報機器とは異なり，IoT 機器には，次の特徴がある．

(1) 一度電源を入れると，長時間起動したまま
(2) ソフトウェアの自動更新機能がないものが多い
(3) ソフトウェアの手動での更新が行われないものが多い
(4) 10 年以上の長期間の利用

このため，これらの特徴を考慮した対策が必要である．長時間起動したままであることから，一度マルウェアに感染すると，そのままマルウェアに長期間活動されてしまう．IoT 機器の場合，再起動すると多くのマルウェアは消えてしまうという報告がある [3]．一方で，文献 [3] では，再起動後も持続的な感染を引き起こす持続感染型 IoT マルウェアについて述べられている．今後は，持続感染型 IoT マルウェアが増加することが考えられる．

IoT 機器にはソフトウェアの自動更新機能がないものが少なくない．最近の機器では，自動更新機能に対応しているものが増えてきていると推察されるが，インターネットに接続された機器では，自動更新機能がないものも少なくないため，対策が必要である．また，自動更新機能ではなく，手動での更新機能を備えた IoT 機器もある．この場合，IoT 機器は PC のようにユーザが普段使用する機器でない場合があるため，ファームウェアなどの更新を管理者やユーザが忘れてしまうことが考えられる．

最後に，IoT 機器には，10 年間以上の長期間利用されることが想定されるため，長期間利用してもマルウェアに感染しないようなセキュリティ機構があることが理想である．今後，より多くの IoT 機器がインターネットに接続されることが想定されるため，IoT 機器のセキュリティ機能を向上させる取り組みが必要不可欠である．

本稿では，Linux カーネルを改変することなく，保護したい AP 向けの Seccomp フィルタを作成し，導入することで，不必要なシステムコール発行を防止し，攻撃を防御する手法を提案する．

3. Seccomp を用いたアクセス制御手法

3.1 考え方

2 章で述べたように IoT 機器には，PC とは異なる特徴があるため，本研究では，長期間，ファームウェアのアップデートが行われないような運用をされたとしても，脆弱性の影響を抑制できるセキュリティ機構の実現を目指す．

攻撃者の目的達成を防ぐために，IoT 機器の外部から侵入後に，マルウェアのダウンロードや実行を防ぐことや，OS の脆弱性を悪用して，権限昇格攻撃を行って，IoT 機器を自由に制御できることを防ぐことを目指す．また，提案するセキュリティ機構が導入しやすいことも重要であるため，OS や AP を改変することなく，導入できる手法を検討する．

そこで，本研究では，Linux カーネルを改変することなく，LKM モジュールに Seccomp フィルタファイルを読み込ませ，保護対象のプロセスに Seccomp フィルタを適用する手法を提案する．また，保護対象プロセスに適用する Seccomp フィルタファイルは，自動生成できるようにする．

3.2 提案手法実現の要件

既存の OS や AP を改変せずに，脆弱性を悪用した攻撃を防止することを目的とするための要件を示す．

（要件 1） OS を改変せずに提案方式を適用できること
（要件 2） AP を改変せずに提案方式を適用できること
（要望 1） 対象のソフトウェアに脆弱性があったとしても，攻撃を防止，もしくはその被害を抑制できること
（要望 2） 自動で防止機能を適用できること

3.3 基本方式

提案手法の全体像を**図 1** に示す．まず，保護対象の AP に対して，処理を実行するために必要なシステムコールを取得する．このために，ログ収集プログラムから，対象のプログラムを起動する．ログ収集プログラムは，Seccomp を `SCMP_ACT_LOG` オプションを指定して初期化し，そのプログラムの子プロセスで Seccomp をロードしてからシェルを起動する．次に，対象のプログラムを起動し，親プロセスで子プロセスの終了後に audit のログを抽出することで，起動したプログラムのシステムコールの発行ログを取得する [4]．

次に，取得したログから，発行されていないシステムコールの発行を制限する Seccomp フィルタファイルを生成する．この処理も，プログラムを作成し，自動化した（要望 2）．

さらに，Seccomp フィルタを pid（プロセス識別子）を指定して適用可能にする LKM モジュールを作成した．この LKM モジュールを Linux カーネルに組み込み，この LKM

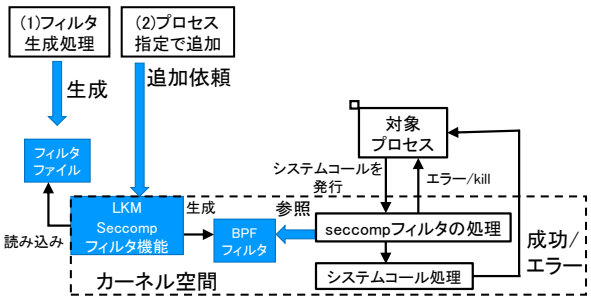

図 1 提案方式の全体像

モジュールの機能を procfs を利用して呼び出し可能にする（要件 1）．

保護対象のプロセスに Seccomp フィルタを適用するため，対象プログラムから保護対象プロセスを起動後，保護対象プロセスの pid とあらかじめ生成した Seccomp フィルタファイルを指定して，端末から LKM モジュールが procfs で作成したファイルに書き込みを行う．書き込みの後に，提案手法の LKM モジュールの関数が呼び出され，対象プロセスに対して Seccomp フィルタが登録される（要件 2）．この処理は，保護対象プロセスは，Seccomp フィルタで制限されていないシステムコールのみ発行可能となる（要望 1）．

制限されたシステムコールが発行された場合には，当該プロセスを終了させることや，単にシステムコールをエラーとして返すことができる．対象のソフトウェアの処理によって，検知時の処理を検討する必要がある．

4. 評価実験

4.1 評価内容と項目

本実験の目的は，ソフトウェアの脆弱性を利用した攻撃に対して，Seccomp フィルタを利用してシステムコールの呼び出しを制限することで，攻撃を防御することができるか否か評価することである．

評価対象の AP として，IoT 機器で使用されている実績があり，かつ脆弱性が確認されているバージョンを用いて，攻撃実験を行った．攻撃に用いたソフトウェア名とバージョンを以下に示す．

(1) ntp 4.2.8p11
(2) Samba 3.5.0
(3) ImageMagick 6.7.8-10

これらのソフトウェアが実際の IoT 機器で用いられているか否か，文献 [5] の調査結果を確認した．文献 [5] の調査では，2000 年から 2019 年までに配布された 13 ベンダのファームウェア 5,712 個，GPL ソースコード 2,379 個から，各ソフトウェアのバージョン情報を抽出した結果を報告している．全体で，1,510 機種の IoT 機器を分析した結果である．この結果から，対象としたソフトウェアは，多くの IoT 機器で利用されていることを確認した．また，文献 [5] の調査結果について，上記 3 つのソフトウェアを調査したところ，評価に用いたものと同じバージョンか，近いバージョンを含むファームウェアを確認した．

表 1 ホスト OS の環境

ホスト OS	Windows 10, 64-bit (Build 18363) 10.0.18363
仮想マシンソフトウェア	VMware(R) Workstation 15 Player 15.5.2 build-15785246

表 2 Seccomp フィルタを利用した Ubuntu の環境

OS	Ubuntu 16.04.7 LTS
アーキテクチャ	x86_64
カーネル	`Linux ubuntu 4.15.0-123-generic #126~16.04.1-Ubuntu SMP Wed Oct 21 13:48:05 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux`

表 3 攻撃を実行した Kali Linux の環境

OS	Kali Linux 2020.3
アーキテクチャ	x86_64
カーネル	`Linux kali 5.7.0-kali1-amd64 #1 SMP Debian 5.7.6-1kali2 (2020-07-01) x86_64 GNU/Linux`
Metasploit	5.0.101-dev

評価の手順について述べる．まず，脆弱性のあるソフトウェアに対して正常処理の場合に呼び出されるシステムコールを記録する．次に，正常な処理で呼び出されたシステムコール以外のシステムコールの呼び出しを制限する Seccomp フィルタを生成する．生成した Seccomp フィルタを提案手法の LKM を介して，実行中のソフトウェアのプロセスに適用し，攻撃を行った．攻撃が成功したか否かを実行結果から判断し，攻撃の成否を結果としてまとめた．

また，攻撃防止効果を比較するため，文献 [5] で調査対象とした 4 つのセキュリティ機能（RELRO，SSP，NX Bit，PIE）とも比較した．

4.2 提案手法による Seccomp フィルタ適用による防止実験

4.2.1 実験環境

実験には，Windows 上で仮想マシンソフトウェアの VMware を動作させ，被攻撃側のゲスト OS に Ubuntu，攻撃側のゲスト OS に Kali Linux を使用した．ホスト OS の環境を**表 1**，Seccomp フィルタを利用した Ubuntu の環境を**表 2**，攻撃を実行した Kali Linux の環境を**表 3** に示す．

攻撃実験の結果をまとめたものを**表 4** に示し，以降でその結果について説明する．

4.2.2 ntp 4.2.8p11 への攻撃

脆弱性のある ntp 4.2.8p11 への攻撃を行った．このソフ

表 4　Seccomp フィルタによる防御の可否

ソフトウェア	防御
ntp 4.2.8p11	失敗
Samba 3.5.0	成功
ImageMagick 6.7.8-10	成功

```
$ ~/lkm_seccomp/logshell
$ ./ntpq -4 [127.0.0.1]
ntpq> exit
$ exit
```

図 2　ntp のフィルタ作成時のコマンド

```
$ ./ntpq -4 [AAAAAAAAAA...(中略)...AAAAA]
Name or service not known
Segmentation fault (core dumped)
```

図 3　ntp への攻撃時のコマンド

トウェアには，Stack-based buffer overflow の脆弱性があり，IPv4 または IPv6 のコマンドラインパラメータの引数として長い文字列を与えることによって，攻撃者がコードの実行またはより高い権限への昇格を起こすことができる．脆弱性の CVE 番号は CVE-2018-12327 である．

提案手法の LKM を使い，攻撃の防御を行った．フィルタの作成時に実行するコマンドを**図 2** に示す．ここでは，ntpq を実行し，対話モードの ntpq ですぐに exit を入力して終了している．ここで作成したフィルタをシェルに適用して**図 3** のコマンドを実行した．

ここで，直接 300 文字の A を引数に渡すと，Segmentation fault で終了するので，バッファオーバーフローが発生したと考えられ，攻撃は防ぐことができていない．しかし，渡される文字列が実行できるコードである場合，そのコードの実行が制限される可能性がある．

4.2.3　Samba 3.5.0 への攻撃

Metasploit を利用し，脆弱性のある Samba 3.5.0 への攻撃を行った．このソフトウェアには，任意のモジュールをロードできる脆弱性があり，遠隔の攻撃者が任意のコードを実行することができる．脆弱性の CVE 番号は CVE-2017-7494 である．

提案手法の LKM を使い，攻撃の防御を行った．Seccomp フィルタを作成するため，Samba を動作させるサーバ用の仮想マシンと Samba にアクセスするためのクライアント用の仮想マシンを使用した．また，ここでは，サーバで `smbd` をデーモンではなく対話モードで起動し，クライアントでは，ファイル一覧を取得し，サーバの `test` ディレクトリに移動し，サーバの `test/test_remote` ファイルを取得し，クライアントの `test_local` をアップロードする．このようにして，ファイル一覧の取得，ディレクトリの移動，ファイルのアップロード，ダウンロードができる Seccomp フィルタを作成した．

```
msf5 exploit(linux/samba/is_known_pipename) > run

[*] Started reverse TCP handler on 192.168.126.145:4444
[*] 192.168.126.130:445 - Using location \\192.168.126.130\sambashare\ for the path
[*] 192.168.126.130:445 - Retrieving the remote path of the share 'sambashare'
[*] 192.168.126.130:445 - Share 'sambashare' has server-side path '/sambashare
[*] 192.168.126.130:445 - Uploaded payload to \\192.168.126.130\sambashare\sitMMwOU.so
[*] 192.168.126.130:445 - Loading the payload from server-side path /sambashare/sitMMwOU.so using \\PIPE\/sambashare/sitMMwOU.so...
[-] 192.168.126.130:445 -   >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
[*] 192.168.126.130:445 - Loading the payload from server-side path /sambashare/sitMMwOU.so using /sambashare/sitMMwOU.so...
[*] Sending stage (3012516 bytes) to 192.168.126.130
[-] 192.168.126.130:445 -   >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
```

図 4　Samba 3.5.0 への攻撃時の攻撃側の端末

攻撃実験では，`smbd` をデーモンとして実行し，起動されたプロセスに Seccomp フィルタを適用した．Seccomp フィルタを適用された Samba 3.5.0 に攻撃を試みた際の攻撃側の端末の表示を**図 4** に示す．ここでコマンドを入力しても実行されず，攻撃は失敗していることが分かる．

攻撃を防止した際に，発行を防止したシステムコールを調査した．攻撃の際に Seccomp フィルタが呼び出しを防ぐシステムコールを調べるため，通常の通信の場合と攻撃を受けた場合に呼び出されたシステムコールの差分を取った．攻撃の際にのみ呼び出されるシステムコールは，`readv`, `recvfrom`, `setsid`, `clock_gettime`, 及び `eventfd2` であった．

4.2.4　ImageMagick 6.7.8-10 への攻撃

Metasploit を利用し，ImageMagick 6.7.8-10 への攻撃を行った．このソフトウェアには，コードが埋め込まれた画像ファイルに対して処理を行わせることで，任意のコードを実行できる脆弱性がある．脆弱性の CVE 番号は CVE-2016-3714 である．

提案手法の LKM を使い，攻撃の防御を行った．使用する Seccomp フィルタは，`convert` コマンドで png ファイルを jpg ファイルに変換した際に呼び出されるシステムコールのみが許可されるものである．被攻撃側の端末で Seccomp フィルタを適用し，`convert` コマンドでコードが埋め込まれた画像ファイルを読み込もうとすると，**図 5** のように表示される．

2 行目の Bad system call (core dumped) から，許可され

```
1 $ convert msf.png test.jpg
2 Bad system call (core dumped)
```

図 5 ImageMagick への攻撃時の被攻撃側の端末

```
1 msf5 exploit(unix/fileformat/
      imagemagick_delegate) > run
2 [*] Started reverse TCP handler on
      192.168.126.145:4444
3 [+] msf.png stored at /home/kali/.msf4/local/
      msf.png
4 [*] Exploit completed, but no session was
      created.
```

図 6 ImageMagick への攻撃時の攻撃側の端末

表 5 バイナリのセキュリティ機能による防御の可否

ソフトウェア	RELRO	SSP	NX bit	PIE
ntp 4.2.8p11	失敗	成功	失敗	失敗
Samba 3.5.0	失敗	失敗	失敗	失敗
ImageMagick 6.7.8-10	失敗	失敗	失敗	失敗

ていないシステムコールを呼び出そうとし，Seccomp フィルタによって終了されることが分かる．攻撃側の端末では，**図 6** のように出力され，攻撃が成功しないことがわかる．

攻撃の際に Seccomp フィルタが呼び出しを防ぐシステムコールを調べるため，通常の通信の場合と攻撃を受けた場合に呼び出されたシステムコールの差分を取った．攻撃の際にのみ呼び出されるシステムコールは，`poll`，`select`，`shutdown`，`getsockopt`，`unlink`, `getrusage`, `mknod`, `clock_gettime`, `newfstatat`, `unlinkat`, 及び `pipe2` であった．

4.3 提案手法と Linux の 4 つのセキュリティ機能との比較

バイナリファイルに適用可能な 4 つのセキュリティ機能の防止実験の結果を**図 5** に示す．提案手法により適用した Seccomp フィルタで防止できた Samba と ImageMagick については，バイナリのセキュリティ機能では防止できていなかった．このため，提案機能を既存のセキュリティ機能と併用することで，よりセキュアな環境を構築できる可能性がある．

一方で，ntp については，提案手法では防止できなかった．SSP で攻撃を防止できていることから，スタックでのバッファオーバフロー攻撃によるものと考えられる．提案手法は，許可されたシステムコールを利用された攻撃の場合には，防止できないものの，バッファオーバフロー攻撃で制御を奪われた後に発行できるシステムコールを制限できるため，プログラムによっては効果があると推察できる．

```
1 $ sudo insmod lkm.ko
2 insmod: ERROR: could not insert module lkm.ko:
      Operation not permitted
```

図 7 LKM ロード禁止時に LKM をロードしようとした際の出力

5. 提案手法の適用に関する制限についての議論

提案手法を利用するには，Seccomp による Seccomp フィルタをサポートし，かつ LKM モジュールの組み込みが可能な Linux カーネルが必要である．

Seccomp フィルタをサポートする Seccomp は，Linux 3.5 から導入されており，このバージョン以降の Linux カーネルであれば導入可能である．ただし，文献 [5] の調査では，IoT 機器のファームウェアからバージョンが判明した Linux カーネルのうち，Linux 3.5 以上のカーネルは，約 23.7% (54/229) であったと報告されている．このため，現状ではあまり採用されていないものの，今後は採用率が向上することが推察できる．

また，LKM の組み込みを可能とするように，`insmod` コマンドの実行が可能でなければならない．`insmod` コマンドは, root 権限を持つユーザのみが実行することができる．したがって, 攻撃によって root 権限を取得された際に, `insmod` コマンドによってカーネル空間で動作するプログラムを実行される可能性が生まれ, セキュリティの脅威が生じる．また，カーネルパラメータ `kernel.modules_disabled` の値を 1 にすることで LKM のロードを禁止することができる．提案手法の LKM 導入後に LKM ロードを禁止した際に，新たに LKM をロードしようとしたときの出力を**図 7** に示す．ロードの際，`insmod: ERROR: could not insert module lkm.ko: Operation not permitted` と表示され，ロードが失敗していることが分かる．このように，提案手法の LKM ロード後に，他の LKM のロードを防止することが対策として考えられる．

`prctl` システムコールによる Seccomp フィルタの適用では，システムコールの呼び出しを行ったプロセス以外に対して適用することができない．本研究で使用した LKM は，`prctl` システムコールとそこで呼び出される関数に対して，適用するプロセスの `task_struct` を渡すことができるように変更した関数を LKM 内に定義した．よって，Linux カーネルの対応する関数の変更に応じて，LKM の関数も変更する必要があり，提案手法では，Linux カーネルのバージョン毎に LKM を作成しなければならない．

6. 関連研究

文献 [6] は 2018 年に人気であった 28 台のホームルータを対象に，ASLR，NX bit，RELRO，および SSP の適用率を調査している．この調査により，PC と比較し，ホーム

ルータは ELF ファイルに十分なセキュリティ機能を適用していないことを示している．また，全体的に ARM を使用しているホームルータの方が，セキュリティ機能の適用率が高かったと述べられていた．この研究では，Seccomp を利用したアクセス制御方式を提案しており，既存の方式と共存してよりセキュアな環境を構築できる可能性がある．

文献 [7] では，初期化フェーズとサービスを提供するフェーズで，必要なシステムコールが異なることに着目し，それぞれのフェーズで最小限のシステムコールのみ許可することで，アタックサーフェスを削減することを実現している．この研究では，必要のないシステムコール発行をブロックするために，Seccomp を利用しており，フェーズの変わり目で，Seccomp フィルタを切り替える処理を AP 側で行う必要がある．一方，提案手法は，AP を改変することなく，外部のプログラムから pid 指定で対象 AP の Seccomp フィルタを指定できることが異なる．

7. おわりに

IoT 機器を想定して，Seccomp フィルタを自動生成し，OS や AP を改変することなく，LKM モジュールを介して，Seccomp フィルタを保護対象のプロセスに適用し，システムコールを制限する手法を提案した．提案手法は，LKM モジュールのロード後，procfs を介して操作することで，対象プロセスの pid を指定して Seccomp フィルタを設定できる．

提案手法の実現方式を示し，攻撃実験を行った結果を報告した．2 つの事例で，攻撃により発行されたシステムコールの実行を防止し，攻撃を防止できていることを示した．また，Linux の 4 つのセキュリティ機能との比較を行い，提案手法の特徴について示した．

残された課題として，多様なプログラムでの評価や性能評価がある．

謝辞 本研究の一部は, JST, さきがけ, JPMJPR1938, および JSPS 科研費 JP19H04111 の助成を受けたものです．

参考文献

[1] Manos Antonakakis, Tim April, and Michael Bailey et al. Understanding the mirai botnet. In *26th USENIX Security Symposium (USENIX Security 17)*, 2017.
[2] Linux programmer’s manual seccomp(2). `http://man7.org/linux/man-pages/man2/seccomp.2.html` (accessed 2019–12–26).
[3] 井上貴弘, 原悟史, 榊博史, 岡田晃市郎, 塩治榮太朗, 秋山満昭, 佐々木貴之, 田辺瑠偉, 吉岡克成, 中尾康二, 松本勉. 適応的サンドボックスによる持続感染型 iot マルウェアの解析. 電子情報通信学会技術研究報告, 第 120 巻, pp. 90–95, 2021.
[4] 松下瑛佑, 山内利宏. Seccomp を利用した iot 機器のセキュリティ機能の向上手法の一検討. 2020 年電子情報通信学会総合大会 情報・システム講演論文集 2, p. 118, 2020.
[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. 情報処理学会シンポジウムシリーズコンピュータセキュリティシンポジウム 2020 (CSS2020) 論文集, 第 2020 巻, pp. 875–882, 2020.
[6] Parker Thompson and Sarah Zatko. Build safety of software in 28 popular home routers. In *Cyber-ITL*, 2018.
[7] Seyedhamed Ghavamnia, Tapti Palit, Shachee Mishra, and Michalis Polychronakis. Temporal system call specialization for attack surface reduction. In *29th USENIX Security Symposium (USENIX Security 20)*, pp. 1749–1766, 2020.