スマートフォン端末における セキュリティ上の脅威と対策： 権限昇格攻撃と悪性ウェブサイト への誘導に焦点を当てて

山内利宏（やまうちとしひろ）

要　旨

スマートフォンなどのモバイル端末の利用者数が増加しており、モバイル端末を金融取引やキャッシュレス決済のツールとして利用するケースが拡大している。一方で、モバイル端末を対象としたマルウェアは、パーソナル・コンピュータを対象としたマルウェアを上回るペースで増加しており、モバイル端末を対象とした攻撃が増大している。本稿では、モバイル端末のなかでも、金融・決済サービスに今後も利用されると予測されるスマートフォンに焦点を当て、スマートフォンへの脅威とセキュリティ対策の動向について報告する。スマートフォンへの脅威のなかでも、特に、権限昇格の脆弱性や脅威、および悪性ウェブサイトへの誘導について述べる。また、Android（TM）を採用したスマートフォンへの脅威に対する主な対策手法として、システム・コール処理における権限の変更に着目した権限昇格攻撃防止手法、SELinux（Security-Enhanced Linux）による Android でのアクセス制御とポリシ、Android における URL（Uniform Resource Locator）バーの切替間隔に着目した利用者の意図しない悪性ウェブサイトへの遷移の検知手法について述べる。最後に、スマートフォンを利用するうえで、留意すべき点について述べる。

キーワード：悪性ウェブサイト、アクセス制御、権限昇格攻撃、スマートフォン、セキュリティ、Android

本稿は、日本銀行からの委託研究論文である。本稿の作成に当たっては、毛利公一教授（立命館大学）から有益なコメントを頂戴した。ここに記して感謝したい。ただし、本稿に示されている意見は、筆者個人に属し、日本銀行や岡山大学の公式見解を示すものではない。また、ありうべき誤りはすべて筆者個人に属する。Android は、Google LLC の商標である。

山内利宏　岡山大学学術研究院（E-mail: yamauchi@cs.okayama-u.ac.jp）

1. はじめに

スマートフォンなどのモバイル端末の利用者数が増加している。2021 年 1 月に公表された調査結果では、世界中のモバイル端末の利用者数は 1 年間で約 9,300 万人（1.8％）増加し、世界人口の 66.6％に達したと報告されている（DataReportal [2021]）。また、2020 年 12 月に公表された日本国内のモバイル端末の利用実態調査では、モバイル端末利用者 1,200 人を調査した結果、スマートフォン利用率は 98.3％であったと報告されている（情報通信ネットワーク産業協会［2020］）。

さらに、モバイル端末の普及により、モバイル版ウェブ・ブラウザの使用率はパーソナル・コンピュータ（Personal Computer: PC）版ウェブ・ブラウザの使用率を上回っており、これに伴いモバイル端末を対象としたマルウェアが増加している。McAfee, LLC [2018] においては、「PC を対象としたマルウェアのサンプルは、200 万件へ到達するまでに 20 年間を要した一方で、モバイル端末を対象としたマルウェアのサンプルは同じ数に到達するまでに 5 年しか要さなかった」と報告されている。

また、情報処理推進機構の「情報セキュリティ 10 大脅威 2020」[1]（情報処理推進機構［2020a］）のうち、金融分野でのモバイル端末の利用に関する脅威として、個人にかかわる脅威の第 1 位に「スマホ決済の不正利用」、第 2 位に「フィッシングによる個人情報の詐取」、第 4 位に「インターネットバンキングの不正利用」、第 6 位に「不正アプリによるスマートフォン利用者への被害」、第 8 位に「インターネット上のサービスへの不正ログイン」、第 9 位に「偽警告によるインターネット詐欺」が挙げられている。

本稿では、モバイル端末のなかでも、金融・決済サービスに今後も利用されると予測されるスマートフォンに焦点を当て、スマートフォンへの脅威と主なセキュリティ対策の動向について紹介する。特に、2020 年 10 月時点で、全世界で約 73％（StatCounter Global Stats [2020]）のシェアを占めている Android を搭載したスマートフォンを中心に調査した結果を報告する[2]。Android は、Google LLC により開発されており、スマートフォンやタブレット端末を対象としたモバイル向けのオペレーティング・システム（Operating System: OS）である。

脅威としては、権限昇格攻撃、および悪性ウェブサイトへの誘導について取り上

1 1 年間に発生した社会的に影響が大きかったと考えられる情報セキュリティにおける事案から、10 大脅威を決定したものである。2006 年から毎年公開されている。

2 Apple Inc. が開発・提供するモバイル向けのオペレーティング・システム（Operating System: OS）である iOS を搭載した端末については、補論 1 を参照されたい。なお、iOS は、Cisco Systems, Inc. の米国およびその他の国における商標または登録商標であり、ライセンスに基づき使用されている。

げる。権限昇格攻撃とは、ユーザやアプリケーション（Application: AP[3]）が本来ならば与えられていない各種の処理を実行する権限を不正に奪取する攻撃である。権限昇格攻撃は、OS や AP に存在する権限昇格可能な脆弱性（セキュリティ上の欠陥）を悪用する。権限昇格攻撃については、スマートフォンの制御を奪い、利用者情報などの奪取に悪用される権限昇格攻撃の概要と対策の難しさについて述べる。JVN iPedia（アイペディア）[4] で調査したところ、Android に関連する権限を昇格させる脆弱性は多数報告されており、特に機器を制御するソフトウェアであるドライバで脆弱性が多く報告されている。

悪性ウェブサイトへの誘導については、金銭目的で、パスワードや個人情報などを窃取するためのサイト（悪性ウェブサイト）へ利用者を誘導する攻撃が多発している。悪性ウェブサイトへの誘導手法と誘導先のサイトでユーザを欺く手法について述べる。特に、スマートフォンでは、PC 向けの攻撃手法とは異なる手法がとられるため、この点について述べる。

また、これらの攻撃への主な対策手法とその課題として、

① OS 内部の処理における権限の変更に着目した権限昇格攻撃防止手法
② セキュリティを強化したアクセス制御機構による Android でのアクセス制御とポリシ
③ Android におけるブラウザの URL（Uniform Resource Locator）表示領域の URL の切替間隔に着目した、利用者の意図しないウェブサイトへの遷移の検知手法

について述べる。最後に、スマートフォンを利用するうえで、留意すべき点について述べる。

本稿の構成は以下のとおりである。2 節でスマートフォンのハードウェア、ソフトウェア、およびウェブ・アクセスの基本的な処理を説明し、3 節でスマートフォンにおける主な脆弱性や脅威について説明する。4 節では、3 節で述べた脅威に対する主な対策手法と課題について説明する。最後に、5 節では、スマートフォンを利用するうえで、留意すべき点について考察する。

3 アプリケーション全般を以下では「AP」と表記するが、そのうち、スマートフォン向け OS のランタイム上で実行されるものについては、特に、「アプリ」と表記する。

4 国内外の脆弱性対策情報を蓄積し、公開している脆弱性対策情報データベースである。2004 年 7 月より JPCERT コーディネーションセンターと情報処理推進機構により共同で運営されている。URL は https://jvndb.jvn.jp（アクセス日は 2020 年 10 月 20 日）。

2. スマートフォンの構成

スマートフォンの構成について、Android を例に説明する（Android デベロッパー［2020a］）。また、スマートフォンからインターネットにアクセスするためのウェブ・アクセスの仕組みについても簡単に説明する。

(1) ハードウェアの構成

Android は、3 種のプロセッサ[5] を公式にサポートしているが、シェアの大半を占めるのは、Arm[6] である。

スマートフォンには、タッチ・パネル、カメラ、SD カードなどが搭載されている。カメラや SD カードなどのデバイスは、ユーザの身の回りの様子を撮影したり、ユーザの電話帳の連絡先情報やカメラで撮影した動画像などの利用者情報を格納したりするために利用できる。アプリがこれらのデバイスを操作するためには、後述する Android のパーミッション[7] をユーザから取得する必要がある。

(2) ソフトウェアの構成

ソフトウェアは、OS とアプリによって構成される。

イ. OS

図表 1 に Android のソフトウェア構成を示す。Android は、① Linux™カーネル[8]、②ハードウェア抽象化レイヤー（Hardware Abstraction Layer: HAL）、③ネイティブ・ライブラリ（Native C/C++ Libraries）、④ Android ランタイム（Android Runtime）、

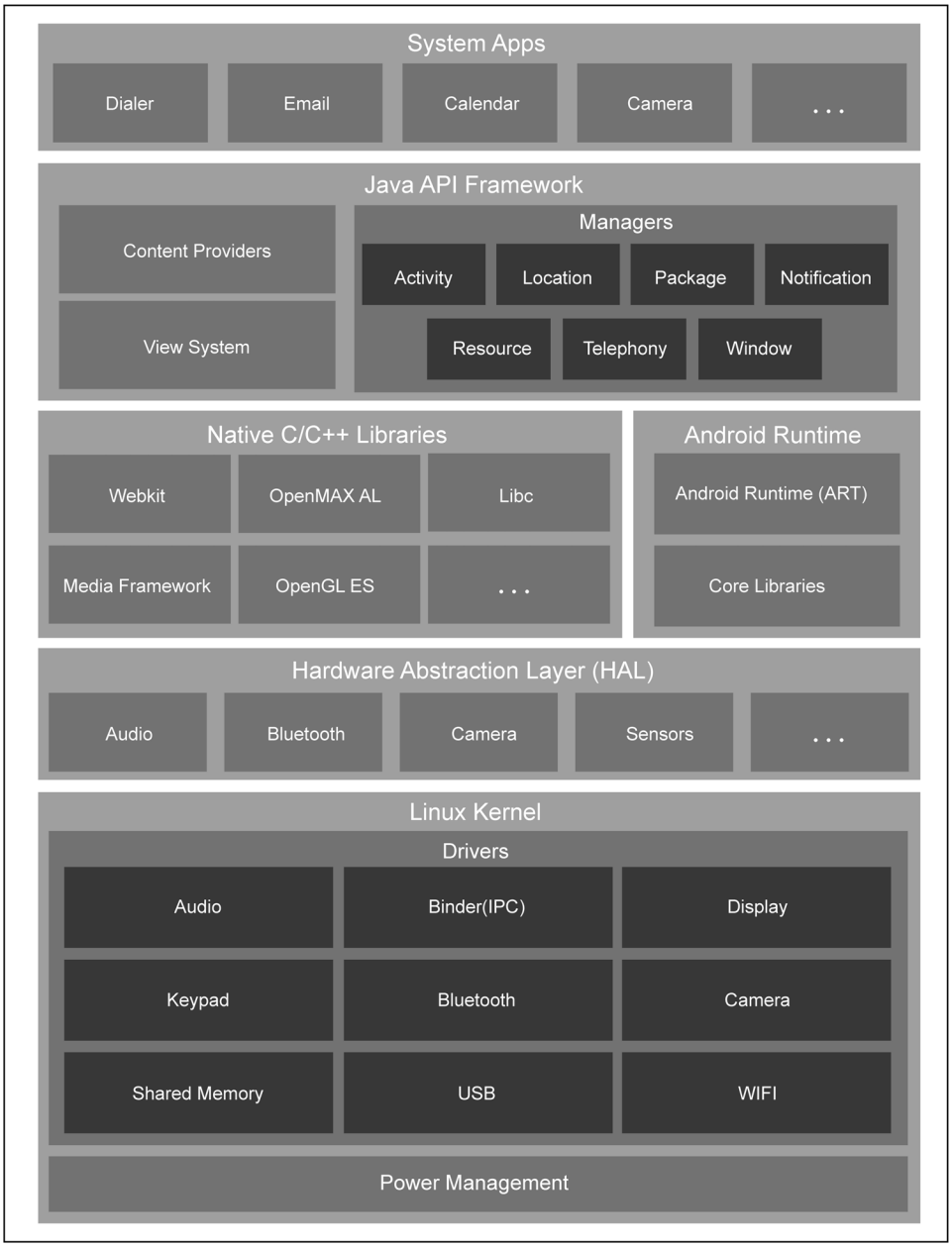
5 Arm®（アーム）、MIPS（ミップス）、x86。Arm は、Arm Limited（またはその子会社）の EU またはその他の国における登録商標である。また、MIPS は、米国およびその他の国における MIPS Technologies, Inc. の登録商標である。

6 Arm Limited により設計されているアーキテクチャ。低消費電力であるため、組込機器のプロセッサで多く利用される。

7 アプリに対して、操作を許可する権限。操作の対象ごとに、パーミッションが設定されている。

8 本稿では、「Linux カーネル」とは、OS の中核となるプログラムを指し、メモリ管理、ファイルの管理、デバイス・ドライバとしての役割、プロセスの管理などを指す。これに対し、「Linux」とは、Linux カーネルのほか、ライブラリや OS を構成するその他のプログラムを含めたものを指す。Linux は、リーナス・トーバルズ（Linus Torvalds）氏の米国およびその他の国における登録商標または商標である。

図表 1　Android のソフトウェア構成



資料：Android デベロッパー［2020b］

⑤ Android フレームワーク（Java API Framework）の 5 つのレイヤーによって構成されている。Android は OS に Linux カーネルを用い、その上にミドルウェア、主要ア

プリを組み合わせたプラットフォームであるが、一般にはOSと呼ばれることが多い。Linuxカーネルは、オープン・ソース・ソフトウェアで開発されているOSであり、コンピュータのハードウェアや周辺機器を管理し、コンピュータの各種の機能をOSとして提供している。HALは、複数のライブラリ・モジュールで構成され、それぞれが特定のタイプ（例：カメラやBluetooth）のハードウェア・コンポーネントに対応するインタフェースを提供する（Androidデベロッパー［2020b］）。Linuxカーネル上のミドルウェアに当たる部分では、ネイティブ・ライブラリと呼ばれるライブラリ・プログラムや、Androidアプリを実行するAndroidランタイム[9]が動作している。その上に、Androidの共通的な要素機能をアプリに提供するAndroidフレームワーク（図表1のJava API Framework）が存在する。Androidフレームワークは、Androidのデータ管理やアクセス制御機能などのさまざまな機能を実現している。Androidには、システム・アプリ（System Apps）[10]として、電子メールやカレンダーなどの主要なアプリが搭載されている。

ロ．アプリ

Androidアプリは、Androidフレームワークが提供するAPI（Application Programming Interface）を介してコンピュータの各種機能を利用することが可能であり、それを用いてアプリのサービスを提供する。AndroidのAPIを利用するには、パーミッションをユーザに許可してもらう必要がある場合がある。例えば、電話帳にアクセスするアプリでは、電話帳にアクセスするパーミッションをユーザが許可している場合、電話帳にアクセスするAPIを用いて、アプリは電話帳から連絡先情報を読み取ることができる。最新のAndroidでは、当該パーミッションを初めて必要とする操作時に、アプリがパーミッションを求めるようになっている。

Androidアプリは、主に、Google Play[11]や携帯キャリアが運営する公式のマーケットで配布されている。公式のマーケットでは、セキュリティ・チェックがされており、一定の安全性が確保されている。一方で、公式以外のサードパーティ・マーケットで配布されるものもあり、こうしたアプリを利用することもできるが、マルウェアなどの不正アプリである可能性もあるため、注意が必要である[12]。

9 Androidにおけるプログラム実行環境のことである。

10 システム・アプリは、Androidを動作させるうえで必要なアプリである。また、システム・アプリは、ユーザ向けのアプリ機能に加えて、デベロッパーのアプリからアクセスできるSMS（Short Message Service）機能などの主要機能を提供する（Androidデベロッパー［2020b］）。

11 Google Playは、Google LLCの商標である。

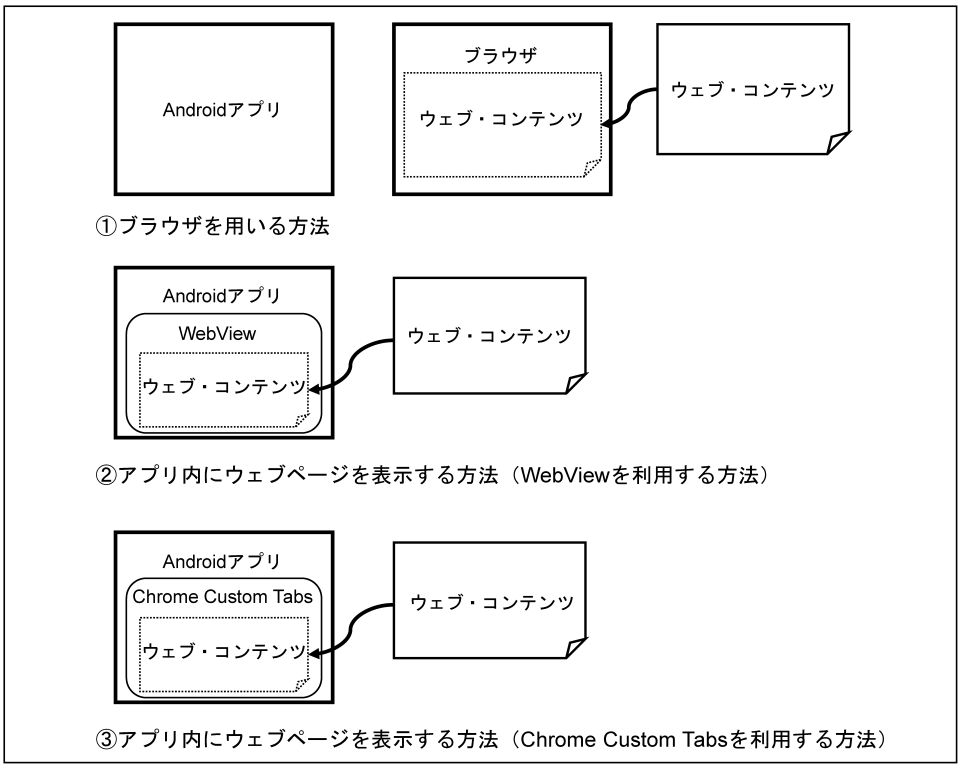
12 マルウェアは、プリインストール・アプリやリパッケージ・アプリとしても配布される可能性がある。これらについては、それぞれ補論2、3を参照されたい。

(3) ウェブ・アクセスの方法

Android 端末におけるウェブ・アクセスの方法には、ブラウザを用いる方法と、アプリ内にウェブ・コンテンツを表示する方法がある。

ブラウザを用いる方法の利点は、高いセキュリティや利便性（パスワード管理など）が提供されることである。Google Chrome™（グーグル・クローム。以降、「Chrome」と呼ぶ）[13] や Mozilla Firefox™（モジラ・ファイアフォックス）[14] などのブラウザは、Google LLC や Mozilla Foundation などにより開発されており、セキュリティにも配慮されている。一方で、後述する WebView（ウェブビュー）は、アプリごとに開発者が異なり、セキュリティに配慮して開発を行うことができるか否かは、開発者に依存してしまう。ブラウザを利用する方法の場合、別のアプリであるブラウザを起動するため（図表 2 の①図を参照）、ウェブ・コンテンツを表示する

図表 2　WebView の利用の有無による表示方法の違い



13 Google Chrome は、Google LLC の商標または登録商標である。
14 Mozilla Firefox は、Mozilla Foundation の米国及びその他の国における商標または登録商標である。

までの待ち時間が長くなってしまう課題がある。

アプリ内にウェブ・コンテンツを表示する方法としては、2 つの方法が挙げられる。第 1 に、WebView と呼ばれるシステム・コンポーネントを利用することである。このコンポーネントを呼び出すことで、アプリ開発者はアプリ内でウェブ・コンテンツを表示させ、利用者にアプリのサービスを提供できる（図表 2 の②を参照）。本方法は、アプリの画面内でウェブ・コンテンツを表示することができるため、ブラウザを起動するのに比べ、待ち時間を短くできる。課題としては、WebView の機能を呼び出すプログラムを個別に作成する必要があるため、脆弱性を作りこんでしまうリスクがあり、セキュリティ上の問題を引き起こす可能性があることと、Chrome などのブラウザのアクセス履歴などは、別アプリであるために利用できないことである。

アプリ内にウェブ・コンテンツを表示する第 2 の方法として、Chrome Custom Tabs（クローム・カスタム・タブ）がある。これは、アプリ内で Chrome の機能を使って、ウェブ・コンテンツを表示する方法である（図表 2 の③を参照）。Chrome Custom Tabs では、開発者が WebView ほど表示をカスタマイズできないものの、アプリ内のウェブページで、Chrome と同等のセキュリティ機能や保存したパスワードの利用など、呼出元の Chrome の機能を利用して利便性を向上できる利点がある。

このように、ウェブ・アクセスには複数の方法が存在するが、いずれの方法についても、意図しない悪性ウェブサイトへ利用者を誘導する攻撃が可能である。攻撃の詳細については、3 節（3）で紹介する。

3. 既知の主な脆弱性や脅威

(1) 権限昇格攻撃

OS において、ファイルなどの資源を操作する場合、操作するユーザがその資源にアクセスする権限を持つか否かにより、アクセス制御機構がアクセスの可否を制御する。権限には、ルート（特権〈root〉）ユーザだけが持つ権限（ルート権限と呼ばれる）があり、ルート権限を有する場合、資源に対するアクセス制御はバイパスされ、コンピュータ上のほぼすべての操作を行うことが可能になる。このため、権限昇格攻撃では、OS カーネル[15] や AP に存在する権限昇格可能な脆弱性を悪用するコードを実行することにより、攻撃者がルート権限を奪取するものが多い。この

15 OS 一般に当てはまる場合は「OS」あるいは「OS カーネル」、特に Linux を想定して説明する場合は、「Linux」あるいは「Linux カーネル」という用語を用いている。

ように、1 つの権限昇格攻撃がシステム全体のセキュリティを脅かすことにつながるため、こうした脆弱性は、攻撃者にとっては格好の的である。

また、Android において、ルート権限の奪取はルート化と呼ばれる。権限昇格攻撃の脆弱性を悪用した Android ルート化ツールが不正に配布されており、このツールによって多くの端末がルート化されている（小久保ほか［2015］）。また、端末のルート化によって、端末メーカーが独自に開発した AP やライブラリなどの知的財産が漏えいするリスクも指摘されている（小久保ほか［2015］）。

権限昇格攻撃への根本的な対策は、Linux カーネルにあるすべての脆弱性を取り除くことである。しかし、Linux カーネルは、2020 年 1 月時点でコード行数が約 2,780 万行を超えている（LINUX.COM [2020]）。このように、非常に大規模で、多くの開発者がかかわって開発されているため、Linux カーネルの脆弱性をすべて無くすことは困難である。また、Android を採用しているスマートフォンの場合、Linux カーネルに脆弱性が見つかったとしても、端末メーカーが Linux カーネルをアップデートしなければ脆弱性に対処できないが、実際は脆弱性が発見されてからアップデートされるまでの期間が長い場合が多く（山口［2018］）、アップデートが提供されない場合もある。

上記のような理由から、権限昇格攻撃は非常に大きな脅威であり、権限昇格攻撃への対策は重要である。

(2) アクセシビリティ・サービスを利用した情報窃取

アクセシビリティ・サービス（Accessibility Service）とは、身体が不自由な方に補助機能による支援を提供するサービスである。Android アプリは、こうしたアクセシビリティ・サービスを利用することで、画面上に表示された文字の読上げや、音声などによる操作を行うことができる。

アクセシビリティ・サービスは、画面上の状態を観測し、その観測した状態に対してさまざまなサービスを提供する。画面上の状態の観測には、イベントと呼ばれる利用者の端末操作や画面上の動作を観測する機能が用いられる。このイベント機能を用いることで、他のアプリが画面に表示しているテキストの内容を読み取ることができる。アクセシビリティ・サービスは、ユーザが許可したときに限り有効となる。

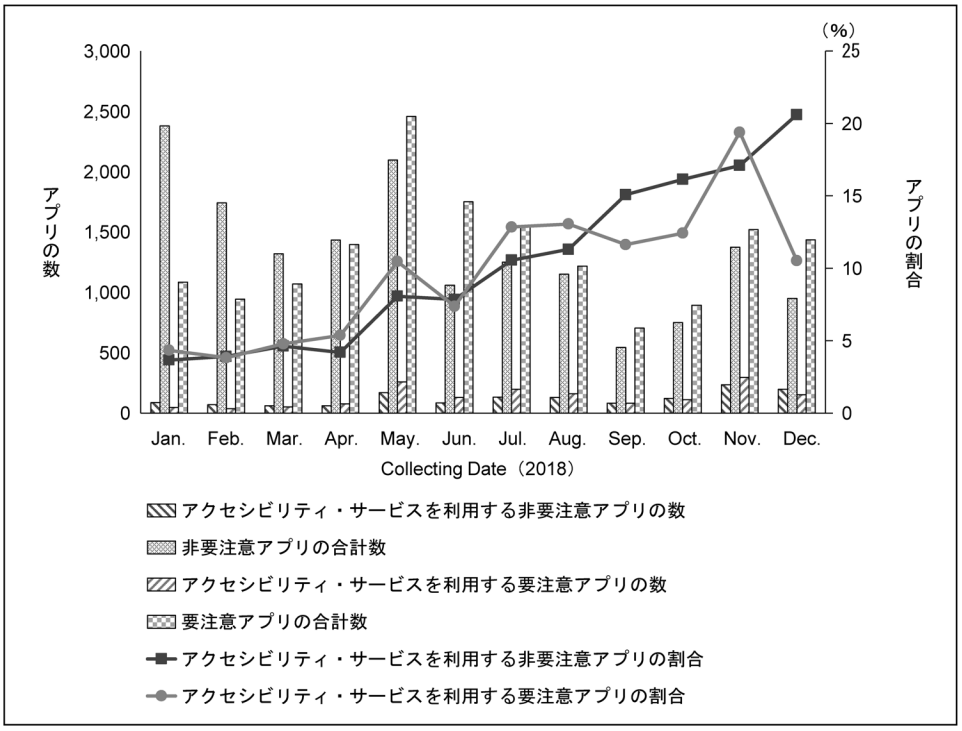
アクセシビリティ・サービスは便利な機能である一方で、その機能を悪用することでセキュリティ上のリスクとなる。実際、ユーザの情報を盗み取る不正なアプリの報告が多い。こうした報告の 1 つとして、Twitter[16] で配布されるアプリのリスク

16 Twitter は、Twitter, Inc. の商標または登録商標である。

を調査した結果がある（図表 3 を参照。Ichioka *et al.* [2020]）。Ichioka *et al.* [2020] によれば、まず、アクセシビリティ・サービスを利用するアプリが増加していることがわかった。また、VirusTotal（ウィルストータル）[17] によるチェックによって、1 個以上のウィルス対策ソフトウェアで悪性のアプリと判定されたアプリ（要注意アプリ）に着目すると、アクセシビリティ・サービスを利用する要注意アプリの割合も増加傾向にあることがわかった。なお、要注意アプリとそれ以外のアプリ（非要注意アプリ）において、アクセシビリティ・サービスを利用するアプリの割合に有意な差は見られなかった。差は見られなくとも、アクセシビリティ・サービスを悪用するマルウェアは増加傾向であることから、その動向に注意する必要がある。

金融サービスにおけるアクセシビリティ・サービスの悪用事例としては、Gustuff（グスタフ）と呼ばれるマルウェアが報告されている（Pisarev [2019]）。Gustuff は、アクセシビリティ・サービスを悪用して盗み取ったユーザ情報を利用して、ユーザの知らないうちに不正に送金を行う。

図表 3　Twitter で配布されるアプリのアクセシビリティ・サービスの利用調査結果



資料：Ichioka *et al.* [2020]

17 VirusTotal は、ファイルやウェブサイトのマルウェア検査を行うサイトである。複数のウィルス対策ソフトウェアの判定結果を利用できる特徴がある。

今後、アクセシビリティ・サービスを悪用するアプリについて注視が必要なものの、アクセシビリティ・サービスをアプリが利用するには、ユーザが明示的に許可する必要があり、不用意にアクセシビリティ・サービスを許可しないことが対策となる。

(3) 悪性ウェブサイトへの誘導

Android 端末を対象とした攻撃に、利用者を意図しない悪性ウェブサイトへ誘導する攻撃が存在する（折戸・佐藤・山内［2019］）。この攻撃では、利用者が遷移元サイトにウェブ・アクセスした際に、自動的もしくは画面のタップなどの操作を契機として、複数のウェブサイト（以降、「経由サイト」と呼ぶ）に連続してリダイレクト（ウェブページの遷移）することによって、目的の悪性ウェブサイトへ誘導する[18]。

モバイル端末における攻撃は、誘導先のウェブサイトで利用者を欺くことで、広告収入の獲得や個人情報の奪取を目的としている（Levinson [2012]）。広告収入の獲得を目的とした攻撃では、偽の画面を表示し、ユーザの不安をあおることで偽のウィルス対策ソフトウェアをインストールさせるといったことが行われる。また、個人情報の奪取を目的とする場合は、偽の懸賞当選を表示し、懸賞に当選したと思わせて、個人情報などを入力させるといった手段が用いられている（Aravindhan *et al.* [2016]、デジタルアーツ［2019］、利穂ほか［2019］）。

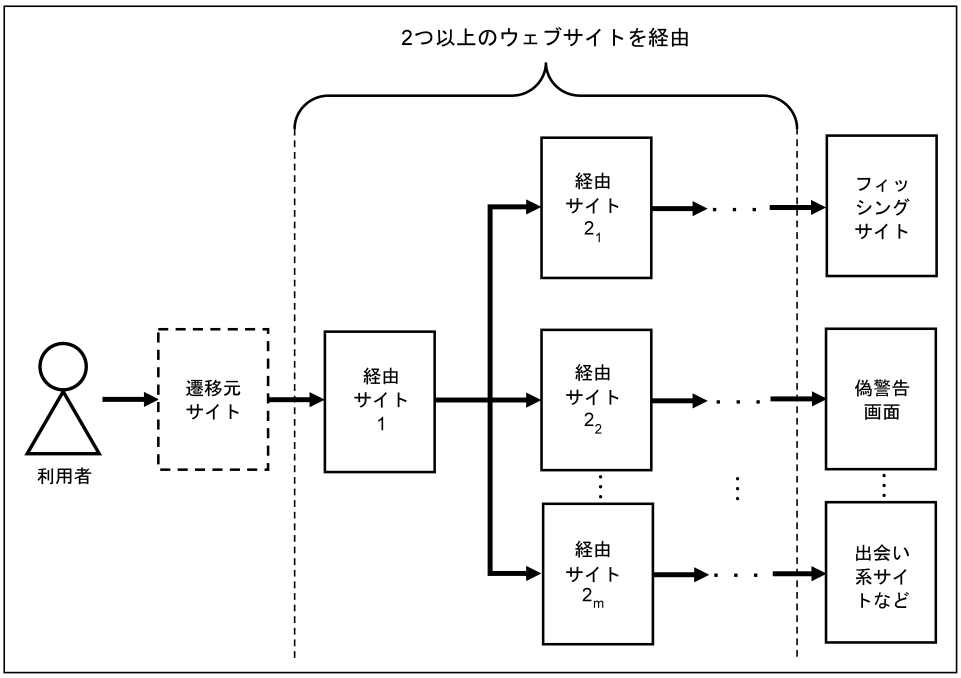
利用者を悪性サイトへ誘導する際の特徴について、折戸・佐藤・山内［2019］と今村ほか［2018］によって得られた調査結果を紹介する。これらの調査結果による Android における遷移元サイトから悪性ウェブサイトへの遷移の流れを図表 4 に示す。

悪性ウェブサイトへの遷移は、①複数のサイトを短時間で経由すること（以降、「連続リダイレクト」と呼ぶ）によってもたらされる、② JavaScript（ジャバスクリプト）コードなどにより、経由サイトのコンテンツの表示が完了する前に発生している、③リダイレクト時に毎回異なる URL を生成し、閲覧履歴を残さないリダイレクト方法が利用されることがあるという 3 つの特徴を持つことがわかった。

このように、毎回異なる URL を生成することで、ブラック・リストに登録されたり、悪性ウェブサイトへの誘導を検知されたりすることを回避していると推測で

18 自動的に連続してリダイレクトさせるために、経由サイトへのリダイレクトを引き起こすようなコードを HTML（Hypertext Markup Language）ファイルに埋め込んでおくという方法がある。また、ウェブ・サーバが HTTP（Hypertext Transfer Protocol）レスポンスのロケーション・ヘッダにリダイレクト先の URL を設定して、ブラウザに返却する場合がある。この場合、ロケーション・ヘッダの URL には、経由サイトの URL が設定される。

図表 4　意図しないウェブサイトへの遷移



資料：折戸・佐藤・山内［2019］、今村ほか［2018］

きる。また、閲覧履歴を残さないリダイレクト方法によって、ブラウザの「戻る」ボタンで前のページに戻ることをできなくすることで、遷移させた悪性ウェブサイトの閲覧を強制させる目的もあるように思われる。

4.　主な対策手法と課題

3 節で述べた権限昇格攻撃によるルート権限の奪取への対策として、権限昇格攻撃防止手法と SELinux（Security-Enhanced Linux）によるアクセス制御について述べる。また、悪性ウェブサイトへの誘導に対する対策技術として、URL バーの切替間隔に着目した検知手法について述べる。

(1)　権限の変更に着目した権限昇格攻撃防止手法

OS カーネルの脆弱性を悪用されると、カーネルのメモリの値が改変されて（本

来意図されていない）メモリ領域の書換えが生じてしまう。その結果、カーネルのメモリ上に格納されている権限に関する情報も変更され、権限昇格が起きる。以下では、これに対処する代表的な方法として、APがシステム・コール（APがOSの機能を呼び出す機構）によってカーネルの権限を更新する処理に着目する手法（赤尾・山内［2016］、Yamauchi *et al.* [2021]、福本・山内［2020］）を紹介する。また、上記の手法を、Armプロセッサのハードウェア機能であるArm TrustZone®（トラストゾーン）[19]において実現する手法（吉谷・山内［2020］）についても説明する。

イ．設計と実装

Linuxカーネルにおいては、APが実行されると、さまざまなプロセス（特定のデータをメモリから読み出したり書き込んだりするなど）が起動される。個々のプロセスには、権限（プロセス権限）が設定されており、その変更は、プロセス権限を変更する役割を持ったシステム・コールによってのみ実現される。また、Linuxカーネルの脆弱性を悪用する権限昇格攻撃では、本来ならばプロセスの権限を変更しないシステム・コールの処理中にプロセスの権限が変更される。

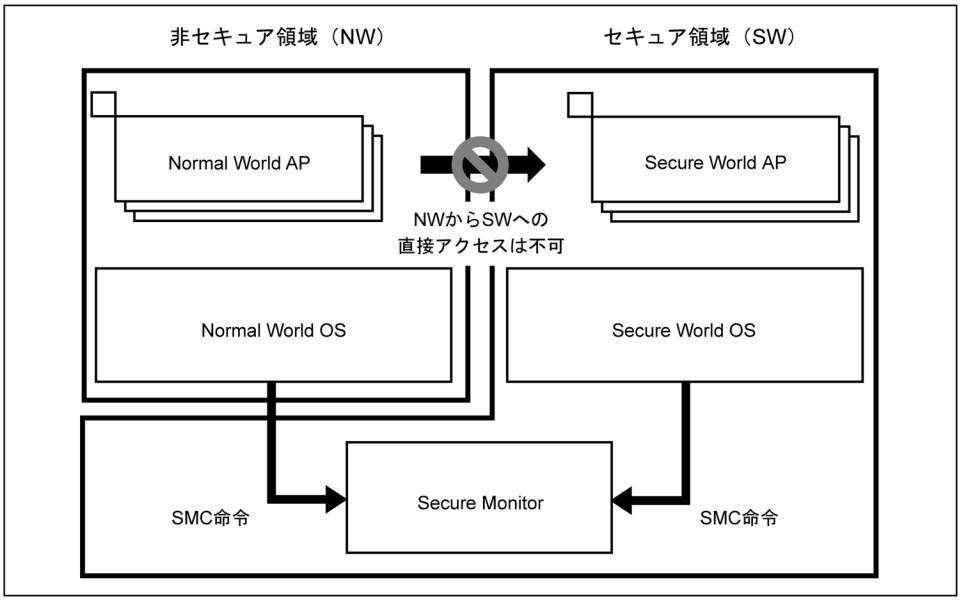
そこで、赤尾・山内［2016］では、これらの特徴に着目し、プロセス権限を変更する役割を持たないシステム・コール処理の前後において、そのシステム・コールが変更しえない権限が変更されていることを検知する手法を提案している。提案手法の処理の流れを説明する。

① 提案手法を実現したセキュリティ機能が、攻撃者が権限昇格のために実行するプロセスがシステム・コールを発行した直後に、プロセスの権限情報を別に保存する。
② システム・コールの処理が実行される。
③ 提案手法を実現したセキュリティ機能が、システム・コール処理前に保存した権限情報から現時点までの権限情報の変更をチェックする。
④ 権限情報の変更内容が正常でない場合、提案手法を実現したセキュリティ機能が、権限昇格攻撃が行われたと判断し、権限情報を元の情報に復元し、攻撃を検知したことを利用者に通知する。

上記の提案手法の利点は不正な権限の変更にのみ着目しているため、今後、未知のカーネルの脆弱性が見つかった場合でも、権限を不正に変更する攻撃に利用するものであれば、攻撃を検知して防止できることである[20]。また、理論上はカーネル

19 TrustZoneは、Arm Limited（またはその子会社）のEUまたはその他の国における登録商標である。
20 本手法は、カーネルのメモリを監視する手法であるため、他のカーネル・データの改ざんの検知にも有効である。Yamauchi *et al.* [2021]では、この手法がセキュアOSの1つであるSELinuxを無効化する攻撃を検知し、防止できることを示している。

図表 5　TrustZone の構成



資料：吉谷・山内［2019］

のメモリをすべて監視することが可能であるが、その分だけシステム・コール発行時にカーネルのメモリを監視する処理のオーバーヘッド[21] が増えるため、セキュリティ向上への寄与が大きいメモリに限定して監視することが望ましい[22]。Yamauchi *et al.* [2021] では、実際に報告された 5 つの脆弱性に対して権限昇格攻撃を行い、上記手法によってすべての攻撃を検知できたと報告している。

ロ．TrustZone を利用した権限昇格攻撃防止手法の実現方式

本節（1）イ．の手法は、Linux カーネル内部に実現しているため、Linux カーネルに脆弱性がある場合、無効化される可能性がある。このため、Linux カーネル内部のセキュリティ機構を無効化する攻撃に対して耐性を持つセキュリティ機構の実現方式が求められている。本節では、TrustZone により、この手法自体を保護する実現方式について述べる（吉谷・山内［2020］）[23]。

Android 端末で採用されている Arm プロセッサでは、TrustZone と呼ばれる安全な OS と AP 実行環境が利用できる。図表 5 に TrustZone の構成を示す。TrustZone

21 追加した処理に要する時間。

22 システム・コールは OS に処理を依頼するたびに実行されるので、実行回数が多く、オーバーヘッドが小さいことが望ましい。

23 本節で紹介する権限昇格攻撃防止手法を TrustZone で保護する研究では、オープン・ソースとして開発されている TrustZone 向け実装の 1 つである OP-TEE を用いている。

は、メモリ領域を非セキュアな領域とセキュアな領域に分離する。非セキュア領域では、Linux や Android などの汎用的 OS と AP が動作する。これに対し、セキュア領域では、セキュア領域用の OS と AP が動作する。

セキュア領域のコードやデータには、非セキュア領域から直接アクセスすることはできない。非セキュア領域からセキュア領域に処理を移行するには、両領域間の切替えの処理を行うソフトウェア（セキュア・モニタと呼ばれる）に処理を移行させる必要がある（須崎［2020］）。また、切替えを行うための専用の命令はセキュア・モニタ・コール（Secure Monitor Call: SMC）命令と呼ばれる。セキュア領域と非セキュア領域は、異なる物理メモリ領域が割り当てられており、領域間の切替えには、セキュア領域で動作するセキュア・モニタを介するため、非セキュア領域からセキュア領域の物理メモリにアクセスできない。このため、攻撃者が非セキュア領域の AP や OS に対する攻撃に成功した場合でも、セキュア領域のコードやデータを攻撃者による漏えいや改ざんから守ることができる。この仕組みにより、セキュア領域の AP は、デジタル著作権の管理（Digital Rights Management: DRM）や鍵管理、認証処理など、高い秘匿性が求められる処理をより安全に実行できる。

本節（1）イ．で紹介した権限昇格攻撃の防止手法を、TrustZone を用いて保護する方式（吉谷・山内［2020］）の処理の流れを図表 6 に示す。セキュア領域に実装した処理は、非セキュア領域におけるシステム・コールによって、プロセスの処理がカーネルに遷移した直後（システム・コール・サブルーチンの前）に、そのプロセスの権限情報をセキュア領域に保存する。また、システム・コール・サブルーチンの後に、セキュア領域に保存した権限情報と、現在の権限情報を比較する処理をセキュア領域で実行するため、これらを攻撃者による改ざんから保護することができる。

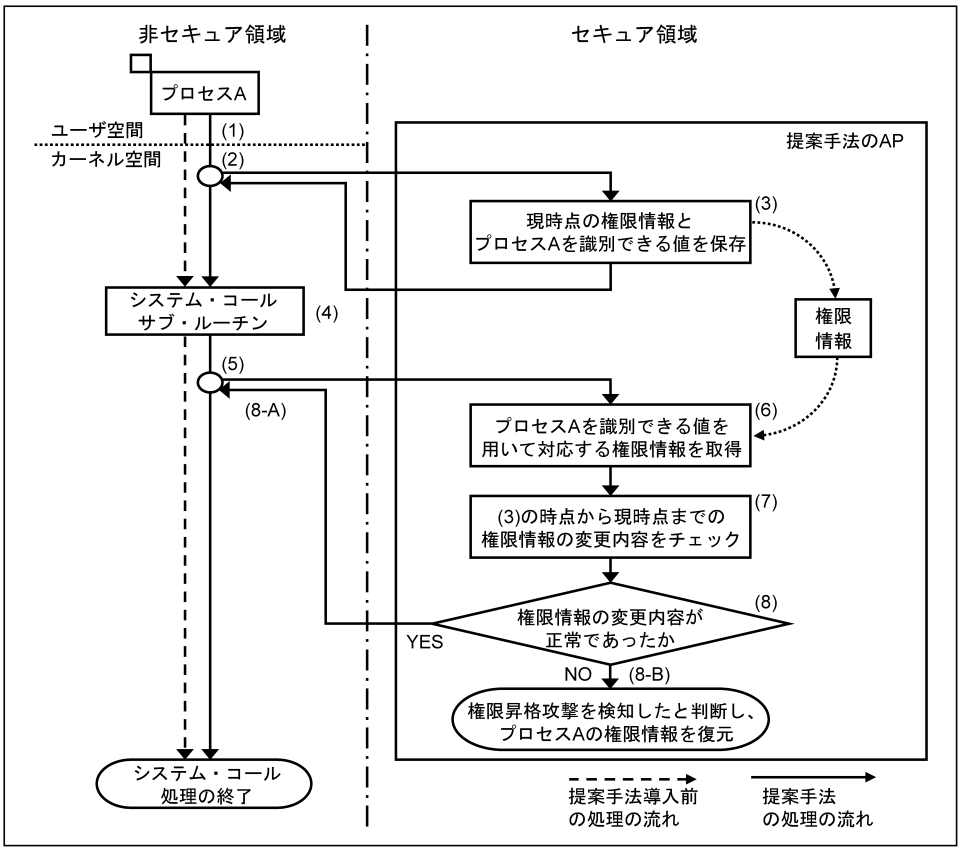
セキュリティ機能の保護に TrustZone を利用した場合、セキュリティ機能の呼出しに非セキュア領域とセキュア領域の切替えを伴うため、処理のオーバーヘッドが増加する。もっとも、エミュレータで性能の評価を行った結果、そのオーバーヘッドはわずかであり、システム・コールを多く呼び出す AP でなければ、性能に与える影響が大きくないことが示されている（吉谷・山内［2020］）。

（2） SELinux による Android でのアクセス制御とポリシ

イ． SELinux の概要

Linux では、任意アクセス制御と呼ばれるアクセス制御を標準で採用している。任意アクセス制御では、ファイルの所有者は、ファイルごとに、①読込み、②書込み、③実行のパーミッションを任意に設定することができる。任意アクセス制御

図表 6　セキュア領域上での権限昇格攻撃防止手法の処理の流れ

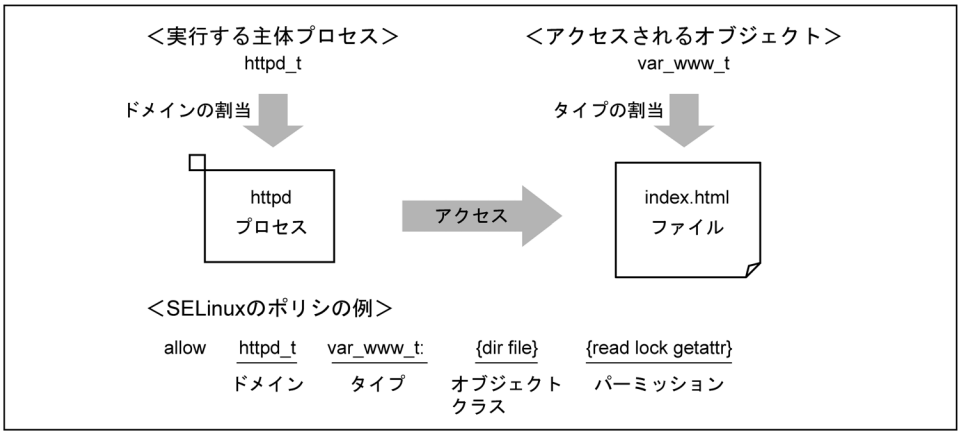


資料：吉谷・山内［2020］

は、ファイル所有者がパーミッションを設定するため、コンピュータ管理者のアクセス制御ポリシをコンピュータ内で必ずしも徹底できないという問題がある。また、ルート権限は、任意アクセス制御で設定されたパーミッションの設定をバイパスして、すべてのファイルにアクセスできるため、権限昇格攻撃によりファイルが奪われ悪用されるという問題がある。

SELinux で採用されている強制アクセス制御とは、実行する主体であるプロセスと、アクセスされるファイルなどのオブジェクトに対して、それぞれ「ドメイン」と「タイプ」と呼ばれるラベルを設定し、主体プロセスとオブジェクトのラベルに対して、どのようなパーミッションが許可されるのかを「ポリシ」として設定し、それに従ってアクセス制御する。図表 7 では、ウェブ・サーバのプロセスに “httpd_t” というドメイン、ウェブ・コンテンツのファイルに “var_www_t” というタイプを設定している。ポリシの例では、ドメイン httpd_t が、var_www_t タイプの

図表 7　SELinux の強制アクセス制御におけるポリシの例



資料：中村・山内［2010］

ディレクトリ（dir）とファイル（file）に対し、read、lock、getattr を許可することを示している。この場合、read はファイルの読込み、lock は他のアプリからアクセスされるのを防ぐためのロック、getattr は属性情報の取得をそれぞれ可能にする。強制アクセス制御では、特権ユーザであっても、ポリシに従ってアクセス制御が強制される。Linux では先に述べたように、ファイルの読込み、書込み、実行のパーミッションがそれぞれ準備されているが、SELinux では 13 個のパーミッションに細分化されている。

SELinux は、特権ユーザであってもポリシに従ってアクセス制御を行うため、特権ユーザが実行できる権限を細分化することに活用できる。例えば、権限昇格攻撃やルート化などに必要な操作を制限することができ、OS の特権が攻撃者などに奪取された場合においても、攻撃者の操作を制限するのに有効である。こうした SELinux のポリシは、Android のオープン・ソース・ソフトウェア開発プロジェクトである Android Open Source Project（AOSP）により開発されている。

ロ．SEAndroid の概要

現在の Android では、強制アクセス制御を採用した SELinux を Android 向けに拡張した SEAndroid（Security-Enhanced Android）が標準で採用されている。現在リリースされている Android では SEAndroid が有効化されている。SEAndroid のポリシは、端末メーカーが設定したものが搭載されており、任意アクセス制御のように、ユーザが自由に設定できない。また、強制アクセス制御により、Android のアプリがスマートフォンの資源に不必要にアクセスできないように制御している。Android 4.3 で、初めて SEAndroid が導入され、Android 5.0 から、SEAndroid のすべての機能が有効な状態で導入されている（Android Open Source Project [2020]）。

端末メーカーは、このポリシを自社のデバイス向けにカスタマイズして、SEAndroid のポリシとしている。Chen *et al.* [2017] では、AOSP の異なる Android バージョン間のポリシ、および端末メーカー 7 社のカスタマイズされたポリシを分析した結果として、誤設定により潜在的な問題があることが指摘されている。SEAndroid の導入により、セキュリティを大幅に向上できるものの、その有効性を活かすためには、カスタマイズ時にセキュリティ上の誤設定を防止することが求められる。

(3) URL バーの切替りに着目したウェブサイトへの遷移の検知手法

Windows[24] などの PC 向けの OS には、ウィルス対策ソフトウェアなどによる、他の AP の通信を監視する機能がある。一方、Android の場合は、すべてのアプリがアプリごとに隔離された環境で実行されるほか、他のアプリの通信や挙動を監視する機能もない。そのため、3 節（3）で述べた悪性ウェブサイトへのアクセスを防止するに当たり、ウェブ・アクセス状況を監視するには、PC 向けの OS とは別の手法を用いる必要がある。こうした方法の 1 つに、アクセシビリティ・サービスを利用し、ブラウザの URL バーの切替間隔を把握することで、利用者の意図しない悪性ウェブサイトへの連続リダイレクトを検知する手法がある（折戸ほか［2020a］）。

イ. 利用者の意図しない悪性ウェブサイトへのリダイレクトを検知する手法

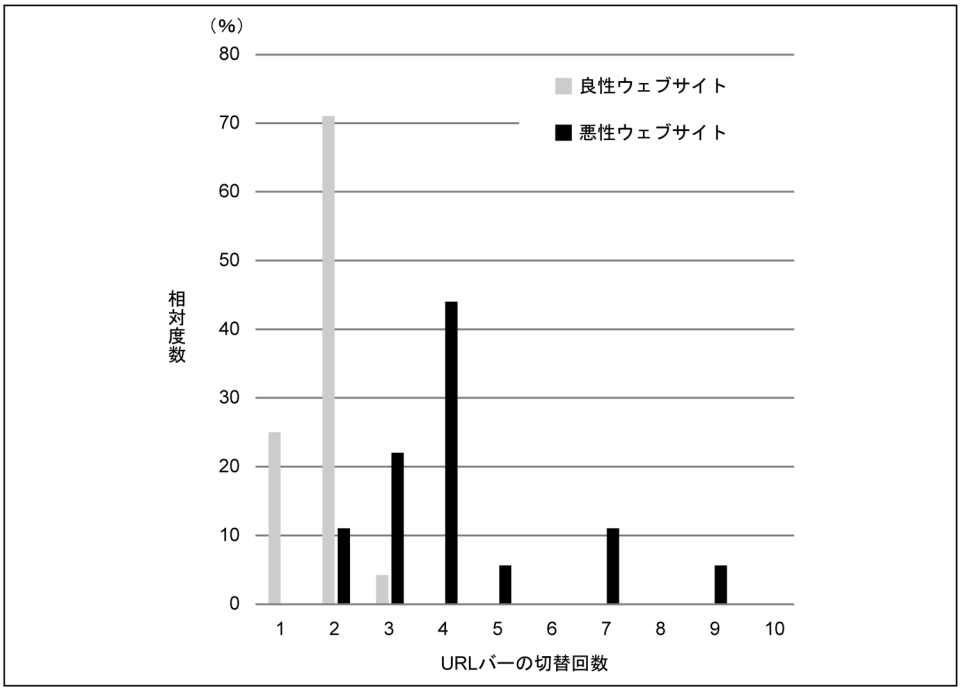
3 節（3）で説明したように、利用者の意思に反して悪性ウェブサイトに遷移させる手法として、利用者が画面のいずれかの部分をタップすると、それを契機に悪性ウェブサイトに誘導させるというものがある。このとき、連続リダイレクトによって、意図しないサイトにユーザをアクセスさせる。これに対処する手法として、折戸ほか［2020a］では、短い時間間隔での連続したウェブ・アクセスを検知する手法を提案している。また、この手法を Android 向けに実現するために、3 節（2）で説明したアクセシビリティ・サービスの機能を利用して、ブラウザの URL バーを監視するというものである。

ロ. リダイレクトの回数による傾向の違い

連続リダイレクトは、悪性ウェブサイトへの誘導における特徴の 1 つであるものの、良性ウェブサイトにおいても、連続リダイレクトを用いる場合がある。例えば、パスワード認証の際に、複数のサイトを経由するリダイレクトが発生する場合がある。このため、連続リダイレクトを悪性ウェブサイトへの遷移の検出に用いるには、悪性ウェブサイトに特有の特徴を利用する必要がある。

24 Windows は、Microsoft Corporation の、米国およびその他の国における商標または登録商標である。

図表 8　リダイレクト時の URL バーの切替回数



資料：折戸ほか［2020a］

折戸ほか［2020a］では、連続リダイレクト発生時の URL バーの切替回数を、良性ウェブサイトと悪性ウェブサイトにアクセスするそれぞれの場合について比較評価している。この調査結果の一部を図表 8 に示す。

図表 8 の縦軸の相対度数は、アクセスした際に、URL バーの切替回数（連続リダイレクト回数）が横軸の値となったウェブサイト数の（全ウェブサイト数に占める）割合である。

図表 8 から、良性ウェブサイトの連続リダイレクト回数は、ほとんどが 2 回以下であることがわかる。一方、悪性ウェブサイトの場合、2 回以上であり、その多くが 3 回以上であることがわかる。このことから、連続リダイレクトの回数の分布の違いにも着目することで、悪性ウェブサイトへの連続リダイレクトを識別できる可能性があることがわかる。例えば、一定の時間内に 3 回以上の連続リダイレクトが確認された場合には、悪性ウェブサイトへの遷移の可能性があるなどの警告を表示するといった手法によって、ユーザに注意喚起を促すことが考えられる。

ハ．連続リダイレクトの回数に着目した検知手法の評価

折戸ほか［2020a］における調査を基に決定した閾値を用いて、実際のサイトに

アクセスしたときの評価結果を紹介する。折戸ほか［2020b］の評価結果は、良性ウェブサイトと悪性ウェブサイトへのアクセスにおける連続リダイレクトの遷移時間に大きな差があることを示している。具体的には、3 回以上の URL バーの切替りが 6 秒以内に観測できた場合に悪性ウェブサイトと判断可能というものである。ただし、単純に連続リダイレクトの遷移時間だけで判断した場合には誤検知が多く発生するため、この手法では、①別のウェブサイトへのリンクをタップする操作は検知の対象外とする、②「戻る」操作によって連続して発生するウェブページの遷移は検知の対象外とする、③良性ウェブサイトは検知の対象外とするという対処を行い、誤検知数を減らしている。この手法により、誤検知を抑えつつ、悪性ウェブサイトへの遷移に対して、事前に警告を表示できており有用であることを示している。

(4) セキュリティ・アプリによる悪性ウェブサイトへの誘導の検知手法の評価

悪性ウェブサイトへの誘導を検知する方法として、Twitter のツイートなどから収集した URL を評価し、悪性ウェブサイトの遷移元をブラック・リストに登録する手法が提案されている（石原ほか［2019］）。こうした手法では、比較的新しい悪性ウェブサイトへの対応が可能であることから、Google Safe Browsing（以降、「GSB」と呼ぶ）[25]、および Google Play において利用者評価の高い 2 つのセキュリティ・アプリ[26]（以降、「アプリ 1」、「アプリ 2」と呼ぶ）と比較して、高い精度で検知できることがわかっている。

評価結果[27]から、25 件の悪性ウェブサイトへの誘導について、アプリ 1 が 1 件検知したのみで、他の 2 つ（GSB とアプリ 2）は検知できなかったのに対し、提案手法では、25 件すべてを検知できたと報告されている。これは、石原ほか［2019］の手法では、Twitter でツイートされた新しい悪性ウェブサイトを評価して、ブラック・リスト構築に用いるため、比較的新しい悪性ウェブサイトにも対応できるためである。一方、GSB とアプリ 1、2 については、ブラック・リストに登録するウェ

25 Google LLC により運用されているサービスである。GSB に対応しているブラウザは、安全性をチェックしたい URL を GSB に登録されているリストと照合し、悪性のウェブサイトとして登録されている場合、そのウェブサイトへのアクセスを防止できる。

26 商用のアプリと無料のアプリの 2 つであり、いずれのアプリも調査の時点（2019 年 7 月 7 日）のインストール数は、500 万件を超えている。

27 評価は、新しい悪性ウェブサイトを用いて評価するため、Twitter のストリーミング API（Twitter, Inc. が公開している API で、ツイートされたデータをリアルタイムで取得することができる）を用いる、モバイル向けの悪性ウェブサイト探索手法（石原ほか［2019］）により、2019 年 7 月 7 日に収集した 5 つの遷移元サイトを利用し、同年 8 月 7 日に実験されたものである。

ブサイトが、悪性ウェブサイトへの誘導を伴うものにフォーカスしていない場合や、Twitter でツイートされた比較的新しい URL に対応できていなかったためと推察している。

石原ほか［2019］の評価は、5 つの遷移元サイトを対象とした検知の可否を確認したものであり、その他の遷移元サイトにおいても同様の結果が得られるかどうかは明らかになっておらず、さらなる研究が求められる。

5. 結びに代えて：スマートフォン利用時の注意点

本稿では、Android を例として、スマートフォンの構成やウェブ・アクセス方法について述べ、スマートフォンの既知の脅威として、特に、権限昇格攻撃、アクセシビリティ・サービスを用いた情報窃取、および悪性ウェブサイトへの誘導に焦点を当てて、最新の対策手法の研究動向を紹介した。

まず、権限昇格攻撃に関して、Android 端末におけるルート化のリスクについて説明した。対策としては、①権限昇格攻撃を検知するために、管理者権限の不正な変更を検知する手法、② TrustZone を利用してその検知手法をセキュア領域に実現する方法、③ Android に標準で採用されている SELinux を Android 向けに拡張した SEAndroid を紹介した。次に、アクセシビリティ・サービスを用いた情報窃取に関して、アクセシビリティ・サービスを用いて情報を窃取するマルウェアが発見されていることを説明し、アクセシビリティ・サービスを不用意に許可しないことが対策になることを述べた。最後に、悪性ウェブサイトへの誘導に関して、攻撃の仕組みを説明したうえで、対策として、連続リダイレクト回数に着目した検知方式を紹介し、一部のセキュリティ・アプリにおける検知精度の比較評価について説明した。

4 節で示したように、提案手法はいずれもそれだけで十分な効果があるというわけではない。攻撃の足掛りとなる不正な AP のインストールを防ぐ、OS の脆弱性を極力解消するなどの運用による対策も欠かせない。スマートフォンは、金融取引やキャッシュレス決済のツールとして、ますます重要になってくるため、その物理的セキュリティは重要である。スマートフォンを紛失しないことは当然であるが、目を離した隙に、情報の窃取や不正アプリのインストールなどの被害を受けないように、画面のロックなど適切なセキュリティ機能の設定が重要である。

また、本稿でも述べたように、スマートフォンにインストール済みのアプリ、もしくはインストールするアプリが信頼できるものであることが重要である。このため、インストールするアプリは、公式のマーケットから必要なものだけをインストールし、プリインストールされたアプリでも、不必要なものはアンインストールするなど、必要最小限の信頼できるアプリを利用することが重要である。

多くのサービスは、スマートフォンのアプリやウェブを介して提供される。このとき、サービス提供者側では、スマートフォンの利用者が本当にそのスマートフォンを所有する利用者であるのか、ユーザ認証を適切にできることも重要である。サービスを受ける利用者側としては、スマートフォンを第三者に操作させないように管理することや、端末のルート化を行わないことが求められる。また、端末への攻撃や認証情報などの窃取を避けるために、最新のバージョンに更新するなどの基本的な対策が重要である。さらに、多要素認証が可能なアプリやサービスでは、積極的に利用することが推奨される。端末に指紋認証機能があれば、パスワードの代わりに利用することも有用と考えられる。

ウェブ上で提供されるサービスを利用するうえでは、ウェブサイトの信頼性や、掲載されている情報の信頼性を見極めることが重要である。本稿で述べた悪性ウェブサイトへの誘導だけでなく、偽装したメールによるフィッシング・サイトや詐欺サイトなどへの誘導も行われており、慎重にサービスを利用することが重要である。特に、アプリの利用時に不必要なパーミッションなどを要求される場合は、慎重に判断をして、安易に許可しないことが必要と考えられる。また、これ以外にもアクセシビリティ・サービスは、他のアプリの情報を取得できることから、安易に許可しないことや、自分が意図していない画面操作で、警告画面やウェブサイトが表示された場合についても、悪意のあるアプリによる誘導の結果である可能性を考慮した方がよいと考える。

参考文献

赤尾洋平・山内利宏、「システムコール処理による権限の変化に着目した権限昇格攻撃の防止手法」、コンピュータセキュリティシンポジウム 2016 論文集、情報処理学会、2016 年、542〜549 頁

石原　聖・折戸凜太郎・佐藤将也・山内利宏、「モバイル向け悪性 Web サイトの探索によるブラックリスト構築手法」、コンピュータセキュリティシンポジウム 2019 論文集、情報処理学会、2019 年、1025〜1032 頁

今村祐太・折戸凜太郎・Kritsana Chaikaew・Celia Manardo・Pattara Leelaprute・佐藤将也・山内利宏、「Android における WebView の Web アクセス観測機構を利用した悪性 Web サイトの脅威分析と対策の提案」、コンピュータセキュリティシンポジウム 2018 論文集、情報処理学会、2018 年、137〜144 頁

折戸凜太郎・石原　聖・佐藤将也・梅本　俊・中嶋　淳・山内利宏、「Android における URL バーの切り替わり間隔に着目した利用者の意図しない Web サイトへの遷移の検知手法の評価」、2020 年暗号と情報セキュリティシンポジウム発表論文、電子情報通信学会、2020 年 a

――――・佐藤将也・梅本　俊・中嶋　淳・山内利宏、「URL バーの切り替わり間隔に着目した意図しない Web サイト遷移の検知手法の改善と実証実験データによる評価」、コンピュータセキュリティシンポジウム 2020 論文集、情報処理学会、2020 年 b、9〜16 頁

――――・――――・山内利宏、「Android における URL バーの切り替わり間隔に着目した利用者の意図しない Web サイトへの遷移の検知手法」、コンピュータセキュリティシンポジウム 2019 論文集、情報処理学会、2019 年、1017〜1024 頁

金井文宏・庄田祐樹・橋田啓佑・吉岡克成・松本　勉、「Android アプリケーションの自動リパッケージに対する耐性評価」、『情報処理学会論文誌』56(12)、情報処理学会、2015 年、2275〜2288 頁

小久保博崇・古川和快・兒島　尚・武仲正彦、「Android/Linux 脆弱性についての一考察」、2015 年暗号と情報セキュリティシンポジウム発表論文、電子情報通信学会、2015 年

情報処理推進機構、「情報セキュリティ 10 大脅威 2020」、情報処理推進機構、2020 年 a（https://www.ipa.go.jp/security/vuln/10threats2020.html、2020 年 10 月 20 日）

――――、「共通脆弱性評価システム CVSS v3 概説」、情報処理推進機構、2020 年 b（https://www.ipa.go.jp/security/vuln/CVSSv3.html、2020 年 12 月 18 日）

情報通信ネットワーク産業協会、「『2020 年度モバイル通信端末の利用実態調査』〜スマートフォン利用者 100％に近づく！ 今後の端末買替え動向は？ 5G いよいよスタート 認知度は高い！ 今後に期待!?〜」、情報通信ネットワーク産業協会、2020

年（https://www.ciaj.or.jp/pressrelease2020/6280.html、2021 年 1 月 31 日）
須崎有康、「Trusted Execution Environment の実装とそれを支える技術」、『電子情報通信学会 基礎・境界ソサイエティ Fundamentals Review』第 14 巻第 2 号、電子情報通信学会、2020 年、107～117 頁
デジタルアーツ、「正規のウェブサイトを改ざんし偽警告や偽当選サイトへ誘導する攻撃を確認」、デジタルアーツ、2019 年（https://www.daj.jp/security_reports/190613_1/、2019 年 8 月 8 日）
中村雄一・山内利宏、「セキュリティポリシー設定簡易化手法」、『情報処理』51(10)、情報処理学会、2010 年、1268～1275 頁
福本淳文・山内利宏、「KVM 上のゲスト OS における権限の変更に着目した権限昇格攻撃防止手法」、『情報処理学会論文誌』61(9)、情報処理学会、2020 年、1507～1518 頁
山口恵祐、「Android スマホの OS アップデート、なぜ遅い？ ソニーモバイルが解説」、ITmedia、2018 年 8 月 20 日
吉谷亮汰・山内利宏、「権限昇格攻撃防止手法における ARM TrustZone を利用した権限の保護」、コンピュータセキュリティシンポジウム 2019 論文集、情報処理学会、2019 年、581～588 頁
———・———、「64-bit ARM 環境における権限の変更に着目した権限昇格攻撃防止手法」、『情報処理学会論文誌』61(9)、情報処理学会、2020 年、1531～1541 頁
利穂虹希・折戸凜太郎・佐藤将也・山内利宏、「Android を対象とした利用者の意図しない Web サイトの分類」、コンピュータセキュリティシンポジウム 2019 論文集、情報処理学会、2019 年、1011～1016 頁
Android デベロッパー、「デベロッパーガイド」、Google LLC、2020 年 a（https://developer.android.com/guide?hl=ja、2020 年 11 月 13 日）
———、「プラットフォームアーキテクチャ」、Google LLC、2020 年 b（https://developer. android.com/guide/platform?hl=ja、2020 年 10 月 20 日）
Android Open Source Project、「Android における Security-Enhanced Linux」、Android Open Source Project、2020 年（https://source.android.com/security/selinux、2020 年 10 月 20 日）
Aravindhan, Ragunathan, Ramachandran Shanmugalakshmi, K. Ramya, and Selvan Chinnaiyan, “Certain Investigation on Web Application Security: Phishing Detection and Phishing Target Discovery,” Proceedings of International Conference on Advanced Computing and Communication Systems 2016, IEEE, 2016, pp. 502–510.
Chen, Haining, Ninghui Li, William Enck, Yousra Aafer, and Xiangyu Zhang, “Analysis of SEAndroid Policies: Combining MAC and DAC in Android,” Proceedings of Annual Computer Security Applications Conference, Association for Computing Machin-

ery, 2017, pp. 553–565.

DataReportal, “Digital 2021: Global Digital Overview Report,” Kepios Pte. Ltd., 2021 (available at https://datareportal.com/reports/digital-2021-global-overview-report、2021 年 1 月 30 日).

Deshotels, Luke, Costin Carabas, Jordan Beichler, Răzvan Deaconescu, and William Enck, “Kobold: Evaluating Decentralized Access Control for Remote NSXPC Methods on iOS,” Proceedings of the 41st IEEE Symposium on Security and Privacy 2020, IEEE, 2020. pp. 1056–1070.

Gamba, Julien, Mohammed Rashed, Abbas Razaghpanah, Juan Tapiador, and Narseo Vallina-Rodriguez, “An Analysis of Pre-Installed Android Software,” Proceedings of the 41st IEEE Symposium on Security and Privacy 2020, IEEE, 2020, pp. 1039–1055.

Ichioka, Shuichi, Estelle Pouget, Takao Mimura, Jun Nakajima, and Toshihiro Yamauchi, “Accessibility Service Utilization Rates in Android Applications Shared on Twitter,” Proceedings of World Conference on Information Security Applications 2020, Lecture Notes in Computer Science, 12583, Springer, 2020, pp. 101–111.

Levinson, Meridith, “Mobile Malware: Beware Drive-by Downloads on Your Smartphone,” CIO FROM IDG, 2012 (available at https://www.cio.com/article/2397969/mobile-malware--beware-drive-by-downloads-on-your-smartphone.html、2019 年 8 月 8 日).

LINUX.COM, “Linux in 2020: 27.8 Million Lines of Code in the Kernel, 1.3 Million in System,” Linux Foundation, 2020 (available at https://www.linux.com/news/linux-in-2020-27-8-million-lines-of-code-in-the-kernel-1-3-million-in-systemd/、2020 年 10 月 20 日).

McAfee, LLC, “McAfee Mobile Threat Report Q1,” McAfee, LLC, 2018 (available at https://www.mcafee.com/enterprise/en-us/assets/reports/rp-mobile-threat-report-2018.pdf、2020 年 7 月 17 日).

Pisarev, Ivan, “Gustuff: Weapon of Mass Infection,” Group-IB, 2019 (available at https://blog.group-ib.com/gustuff、2020 年 10 月 20 日).

StatCounter Global Stats, “Mobile Operating System Market Share Worldwide,” StatCounter Global Stats, 2020 (available at https://gs.statcounter.com/os-market-share/mobile/、2020 年 11 月 13 日).

Tang, Zhushou, Ke Tang, Minhui Xue, Yuan Tian, Sen Chen, Muhammad Ikram, Tielei Wang, and Haojin Zhu, “iOS, Your OS, Everybody’s OS: Vetting and Analyzing Network Services of iOS Applications,” Proceedings of USENIX Security Symposium 2020, USENIX Association, 2020, pp. 2415–2432.

Yamauchi, Toshihiro, Yohei Akao, Ryota Yoshitani, Yuichi Nakamura, and Masaki Hashimoto, “Additional Kernel Observer: Privilege Escalation Attack Prevention Mech-

anism Focusing on System Call Privilege Changes," *International Journal of Information Security*, 20, Springer, 2021, pp. 461–473.

補論 1.　iOS についての脅威や課題

Android は、オープン・ソース・ソフトウェアとして開発されているため、セキュリティ機能の研究や脆弱性調査などの研究が盛んに行われている。一方、iOS はオープン・ソース・ソフトウェアではないため、ソース・コードを基にした研究はできないものの、最近では、リバース・エンジニアリングなどの手法によって、脆弱性の調査などの研究が行われている。

本節では、iOS でも、オープン・ソース・ソフトウェアとして開発されている Android と同様に多くの脆弱性が発見されていることを調査した結果を基に述べる。また、最新の研究で報告されている、ネットワークに関係したプログラムでの脆弱性について、簡単に説明する。

(1)　公表されている脆弱性の件数

図表 A-1 に、JVN iPedia[28] において、“Android” と “iOS” をキーワードとして、脆弱性情報の数を調査した結果を示す。キーワードとして各 OS 名を含むものを調査しているため、OS そのものの脆弱性以外にも、アプリやドライバなどの脆弱性を含んだ数である[29]。図表 A-1 から、両 OS ともに、OS やアプリなどに多数の脆弱性が報告されていることがわかる。Android の方が脆弱性の報告数はかなり多いものの、iOS でも深刻度の高い脆弱性が多く報告されており、セキュリティを高める必

図表 A-1　“Android” と “iOS” のキーワードを含む脆弱性情報の数

年	“Android” を含む脆弱性の数	“iOS” を含む脆弱性の数
2016	779（513）	349（180）
2017	1,213（953）	320（139）
2018	711（603）	172（82）
2019	785（422）	133（67）
2020（9 月 30 日まで）	382（198）	88（41）

備考：括弧内は、CVSSv3 の深刻度が緊急または重要の脆弱性の数。

28 国内外の脆弱性対策情報を蓄積し、公開している脆弱性対策情報データベースである。2004 年 7 月より JPCERT コーディネーションセンターと情報処理推進機構により共同で運営されている。URL は https://jvndb.jvn.jp（アクセス日は 2020 年 10 月 20 日）。

29 図表 A-1 の CVSS（Common Vulnerability Scoring System）は、情報システムの脆弱性に対するオープンで汎用的な評価方法であり、ベンダーに依存しない共通の評価方法を提供している。CVSS を用いると、脆弱性の深刻度を同一の基準のもとで定量的に比較できるようになる（情報処理推進機構［2020b］）。JVN iPedia では、CVSS v3 基本値が 9.0～10.0 の脆弱性の深刻度を「緊急」、7.0～8.9 の脆弱性の深刻度を「重要」と分類している。

要があることがわかる。

(2) 脆弱性に関する研究事例

iOS では、常駐してバックグラウンドで処理を行うプログラムであるシステム・デーモンにより、さまざまなサービスが提供されている。また、サードパーティ・アプリが、セキュリティ上の機密情報に直接アクセスできないように、ファイルへのアクセス制御などを利用している。特に、システム・デーモンは重要なプログラムであるため、適切にアクセス制御がなされ、保護されている。しかし、システム・デーモンへのプロセス間通信により、サードパーティ・アプリが機密情報にアクセスできる場合がある。Deshotels *et al.* [2020] では、フレームワーク Kobold（コボルド）を提案している。Kobold は、プロセス間通信を利用するシステム・プログラムを対象として、脆弱性の有無やシステムを停止させる問題の有無を調査する。

Deshotels *et al.* [2020] は、権限要件を持たないメソッドを 139 件発見し、不正なプライバシー・データの漏えいやマイクの起動、およびシステム・デーモンのクラッシュの問題を発見している。スマートフォン用の OS は、アプリの実行を隔離する機能を有しているものの、その設定が適切になされているかを検証することやその技術が必要であることを示唆している。

また、Tang *et al.* [2020] では、ネットワーク接続を受け付けるスマートフォン・アプリは、ユーザをセキュリティやプライバシーの脅威にさらすことから、iOS アプリのネットワーク・サービスのセキュリティを調査し、分析している。分析の結果、1,300 個のアプリからネットワーク・サービスの脆弱性の特徴を明らかにし、人気のあるアプリにおいて、11 個の脆弱性を発見している。また、11 個の脆弱性の特徴を基に、168,951 個の iOS アプリを調査した結果、92 個のアプリにおいて、リモート接続を受け付ける特定のサードパーティ・ライブラリの使用が共通の原因であることを示している。この文献から、ネットワーク接続を受け付けるアプリにおけるセキュリティの危険性が示唆されている。

補論 2.　プリインストール・アプリの脅威

Android では、オープン・ソース化されているため、端末メーカーは、プリインストール・アプリと合わせて、カスタム化された Android を出荷できるようになっており、他社との差別化を図ることができる。しかし、Android にプリインストールされているソフトウェアの状況、特にそのようなカスタマイズがセキュリティやプライバシーに与える影響については、調査事例は多くない。

Gamba *et al.* [2020] では、200 以上のメーカーの Android 端末のプリインストール・アプリについて、初の大規模調査を実施している。この文献の著者らは、調査結果から、大部分のプリインストール・アプリは、潜在的に有害か好ましくない挙動を示すことを明らかにしている。また、プリインストール・アプリや内蔵されているサードパーティ・ライブラリによって、ユーザの活動を追跡する事例を確認している。Android プラットフォームのオープン・ソースの性質とそのサプライ・チェーンの複雑さのため、さまざまな種類や規模の組織が、自社のソフトウェアを Android のカスタム・ファームウェアとして埋め込んでいることが原因とみられている。

このことから、プリインストールされているソフトウェアについても、セキュリティ上の潜在的なリスクを理解して使う必要があることがわかる。

補論 3.　リパッケージ・アプリの脅威

リパッケージ・アプリとは、公式に配布されているアプリを無断で改変し、元のアプリの機能をそのままに若干の不正な機能を追加したアプリである。開発者は、アプリ自体にデジタル署名を行って、アプリを配布できる。リパッケージされると、開発者以外がもう一度署名するため、デジタル署名を確認することで、リパッケージされたか否かを確認することができる。

リパッケージ・アプリを配布する目的は、広告ライブラリを挿入することなどによる金銭目的のものや、情報収集を目的としたものなどがある。こうしたリパッケージは自動化されており、自動リパッケージにより不正なコードを挿入しても、元のアプリの機能は損なわれないケースが多いことが示されており、ユーザが攻撃を検知しにくいという性質がある（金井ほか［2015］）。