6-3　Android アプリのリスク監視・評価技術

高橋健志　　班　涛

Android 端末を標的としたマルウェアは急増している。それらのマルウェアを検知すべく、本稿では Android アプリのマルウェア判定手法を提案する。提案手法は、従来の静的解析にオンラインマーケットから取得できる APK のメタデータを利用することで、マルウェア検知率を向上する。プロトタイプ実装では、マルウェア判定に加えて脆弱性判定機能も実装し、Android アプリのリスクを総合的に判断できることを示す。

1　まえがき

スマートフォンは現代社会において必要不可欠な生活基盤ツールとなってきている。スマートフォンを用いると、電話やメール、ウェブブラウジングができるのみならず、インターネットショッピングやインターネットバンキング、株取引なども実施することができる。そのスマートフォンの代表格のひとつに Android 端末が存在するが、これが悪意のある第三者の攻撃の対象になっている。

G DATA 社によれば、2015 年は同年第 3 四半期までに、既に 1,575,644 個の Android のマルウェアを検出した [1] [2]。これらのマルウェアの中には、ユーザのファイルを暗号化する Simplocker や端末の PIN ロックを勝手に設定・変更する LockerPIN など、強力なランサムウェアが含まれている。例えば後者の LockerPIN では、PIN を勝手に変えてしまい、その変更した PIN はユーザどころか攻撃者にも送られることはないため、要求される金額を攻撃者に支払っても、PIN を知ることができない。Android のセキュリティ脅威は既に現実かつ差し迫っており、想定される被害も大きく、インシデントが起きる前に十分な対策を講じていく必要がある [2]。

マルウェアへの感染は、マーケットからマルウェアをダウンロードしてインストールしてしまうケースと、通常の PC 向け同様であるがウェブサイトやメールでユーザにアプリをダウンロード・インストールさせるケースが主なものである。どちらも Android 端末の通常利用の範囲で起こり得るケースである。Android Security に常に気を配っているエンジニアであれば、こういったマルウェアなどを取り込まないよう事前に対応することも可能かもしれないが、Android 端末の利用者は裾野が広く、IT に詳しいエンジニアだけが利用しているわけではない。ご家庭によっては、かなり幼い子供からお年寄りまで、非常に幅広い利用者がいる。そういった人々全員に対し、セキュリティを考えていく必要がある。

Android のセキュリティを担保するには様々な技術を要するが、本稿では、特に Android アプリ（以下、アプリ）のマルウェア判定手法を提案する。提案手法では、従来の静的分析に、Web から取得できるアプリの説明文などの各種メタデータを活用している。プロトタイプ実装では、マルウェア判定手法に加え、脆弱性判定機能も実装し、アプリのリスクを総合判定できるシステムを構築した。なお、本稿は著者の論文 [3] 及び文献 [2] [4] の要約であり、詳細についてはこれらの文献を参照されたい。

2　Android アプリ分析の各種アプローチ

APK ファイルを分析する技術には、多数のものが存在するが、大きく分けて、静的解析と動的解析が存在する。静的解析とは APK ファイルの中身を解析していくホワイトボックス手法であり、動的解析とは APK ファイルの中身を解析することなく Android OS 上にて動作させ、その動作を解析していくブラックボックス手法である。どちらも有効な技術であるが、本稿では、静的解析に焦点を絞ることとする。以下に、代表的なアプローチ別に説明する。

2.1　ブラックリストを活用

本アプローチは、ブラックリストに基づきマルウェアを特定する手法である。ブラックリストの例として、マルウェアであることが分かっている APK ファイルのハッシュ値のブラックリストや、危険な通信先 IP アドレス／URL のブラックリスト、マルウェア作成者の証明書のブラックリストなどが存在する。これらの手法は、動作が早く広く利用されているものの、既

に誰かが評価をした結果に基づきブラックリストが作成されていることが前提である。そのため、新たに登場するアプリの信頼性を評価するには、新たにそのアプリを評価しなければならない。

2.2　マルウェアらしさを統計処理に基づき数値化

本アプローチは、マルウェアである可能性（マルウェアらしさ）を数値化する技術である。そのひとつである DroidRisk[5] という方式では、あるアプリが利用している各パーミッションについて、そのリスクの期待値を算出し、その期待値の総和をアプリ全体でのリスク値としている。そして、そのリスク値が一定の閾値に達しているか否かでマルウェア診断を実施している。より具体的には、要求されている各パーミッションがマルウェアに悪用される確率とその際の影響度を算出し、それらを乗算することにより各パーミッションのリスク値を算出する。そして、ひとつの APK が利用するすべてのパーミッションに対し本リスク値を合算し、その値を当該 APK のリスク値としている。

この方式でも、ある程度のマルウェア検出精度を実現しているが、もちろん完璧ではない。実際、あるパーミッションが悪影響を与える期待値は、そのアプリの種類により異なるはずであるが、そういったアプリのコンテクストについては考慮していない。例えば、カレンダーアプリでは電話帳へのアクセスのパーミッション要求があってもおかしくないが、電卓アプリがそのパーミッション要求を実施するのは考えにくい。このようなコンテクストを考慮していないため、やはりマルウェア検知精度に限界がある。

2.3　アプリ説明文と実態機能の乖離を検証

APK のマーケットには、アプリカテゴリやアプリ説明文などのメタデータが存在する。このアプリ説明文と実際の挙動が一致するのかを分析する研究開発も報告されている。一例として、CHABADA[6] では、アプリ説明文からクラスタを生成し、その各クラスタ内に属する APK に対し他の APK とは明らかに異なる特徴を持つ APK を「アプリ説明文と実態機能が乖離しているアプリ」と判定している。

マルウェアは、実際の説明文とは異なる挙動をすることが多いため、これらの手法は軽微な調整を実施することにより、マルウェア検知にも十分利用できる。ただし、マルウェア検知が目的であれば、アプリ説明文と実際の挙動を比較するよりは、直接、アプリの特徴情報を元に機械学習を実施するアプローチの方が検知率は高くなる。

2.4　機械学習を利用

アプリのリスク数値化や、疑わしいパーミッションの利用を監視する手法も有効性が高いものの、マルウェアか否かを 2 値判定する目的に絞れば、機械学習を用いた手法が最も高いマルウェア検知精度を実現する。機械学習手法のひとつであるサポートベクターマシン（SVM）[7] は、対象となるデータ集合に対し、それぞれのデータの特徴をマッピングし、データ集合を 2 分割する境界線を引く技術である。アプリのマルウェア分析にも利用可能であり、入力される APK ファイルの特徴情報に対し、マルウェアとそうでないものの境界線を引くことにより、APK ファイル群全体を 2 分する。上述のリスク数値化技術等とは異なり、マルウェアらしさなどの多段階で表現される評価を実施するのは難しいものの、マルウェアか否かを 2 値評価するというケースでは非常に精度が高い。

この手法は、APK の特徴となる情報を input としているため、よりマルウェアとそれ以外での特徴が出やすいと考えられるパラメータを入力することで、より高いマルウェア判別制度を実現可能となる。例えば、各アプリで利用しているパーミッション要求リストを入力することで、高いマルウェア判定精度を実現できる。

なお、上記のすべての方式の説明において、パーミッションに基づく説明をしているが、実際には分析の対象としてパーミッションよりも API 呼出を用いたほうが通常は分析精度が高い。そのため、上記のそれぞれの方式において、パーミッションの代わりに API 呼出を分析することで、マルウェアの検知精度の向上が実現できる点に留意されたい。簡単のため、本稿では API 呼出に関する検討については省略する。

2.5　脆弱性を検知

ソフトウェアに関する脆弱性情報については、日本の IPA が管理している Japan Vulnerability Note（JVN）[8] や米国の NIST が管理している National Vulnerability Database（NVD）[9] にデータの問い合わせをすることで、情報を収集することが可能である。とはいえ、それらの情報の大半は PC 端末を想定しており、Android に関する情報は限定的なのが現状である。

そのため、自ら脆弱性の存在を判断する技術が併せて必要となる。様々な方法があるが、APK ファイルをリバースエンジニアリングにより分析し、プログラムが脆弱なコードを含んでいないかをチェックすることができる。ここで、危険なコーディングの基準が必要となるが、例えば、各種のコーディングガイド遵守違反に対し、危険と判断するなどの方法が考えられる。

例えば、一般社団法人日本スマートフォンセキュリティ協会（JSSEC）から「Android アプリのセキュア設計・セキュアコーディングガイド」[10] が発行されているが、ここで定められたガイドラインに違反したコードを発見した際には脆弱性があると判断するなどの方法が考えられる。既に本ガイドブックの中に従うべきルールが明記してあるため、これらのルール順守状況をチェックするツールを構築するだけでも、ある程度の脆弱性チェックは自動で実施可能となる。

3　提案手法

提案手法は、Web から収集したアプリのメタデータを活用する。このメタデータには、アプリの説明文やカテゴリ情報などが含まれるが、現状ではこの 2 種類の情報のみを活用している。具体的なアルゴリズムとして、2 種類の手法を提案する。

まず、DroidRisk を拡張する手法 $DR_{category}$ を提案する。$DR_{category}$ では、アプリのメタデータごとに、DroidRisk を実施する手法である。DroidRisk は、統計処理によりマルウェア判定を実施する手法であるが、そのアプリのコンテクストごとに統計処理を実施することにより、その精度を実現するのが $DR_{category}$ である。また、カテゴリの代わりに、アプリの説明文からクラスタを自動生成し、そのクラスタ別に DroidRisk を実施する $DR_{cluster}$ も構築したが、ここでは省略する。

次に、機械学習手法である SVM を拡張する手法 $SVM_{cluster}$ を提案する。SVM は各種パラメータからアプリの特徴を抽出し、それに基づきマルウェアか否かで 2 つのグループに分類する手法である。従来の SVM 手法では、APK ファイルに含まれる情報のみに基づき特徴が構成されていたが、提案方式では Web にあるメタデータという、性質の異なる情報を特徴抽出に活用している。$SVM_{cluster}$ では、アプリの説明文から生成したクラスタ情報と静的分析結果から SVM に基づく機械学習を実施した。

上記の 2 つの方式ともに、既存方式の拡張であり、性能改善を実現するものであるが、結果的に、$SVM_{cluster}$ の性能が最も優れていたため、後述のプロトタイプ実装では現時点では主に $SVM_{cluster}$ を採用している。なお、これらの方式の詳細については、文献 [3] を参照されたい。

4　プロトタイプの構築

本節では、$SVM_{cluster}$ のプロトタイプとして構築したアプリのリスク評価システムを紹介する。本プロトタイプでは、Android 端末にインストールされているすべてのアプリを監視し、その一つひとつに対してリスク評価を実施する。リスク評価は、脅威と脆弱性の両面から実施しており、脅威評価では、そのアプリがマルウェアであるか否かを 3 段階評価している。ここで 3 段階評価とは、信号を意識した「赤」「黄」「無点灯」評価であり、そのアプリがほぼ確実にマルウェアであると考えられる際には「赤」評価を、マルウェアの可能性があると考えられる際には「黄」評価を、現在の評価エンジンでは特にリスクを検知できない際には「無点灯」評価を実施する。脆弱性評価では、そのアプリに深刻な脆弱性があるかどうかを同様に信号形式にて 3 段階評価している。

脅威と脆弱性の評価エンジンは独立しており、また各評価エンジンには自由に各種の分析手法を組み込むことができる形になっている。これは、現在のリスク分析手法が将来にわたって最適なものではないと考えているのと同時に、現時点においてもひとつの分析手法では完璧なリスク評価ができないためである。図 1 に、現在の我々の「Android アプリリスク分析」アプリについて、そのスナップショットを示す。

図 1 の左図は、あるアプリのリスクの総合評価であり、脅威、脆弱性の評価に基づく総合リスク評価がなされている。この脅威、脆弱性に関する評価結果の信号をタップすると、それぞれその評価の裏付けとなる詳細情報が出てくる構成になっている。図 1 の中央図が脅威評価結果であり、図 1 の右図が脆弱性評価結果となる。

まず脅威評価結果を見ると、複数の評価基準を実装している。現時点では、DroidRisk 評価（$=DR_{category}$）、SVM に基づくマルウェア評価（$=SVM_{clustger}$）、ブラックリスト URL チェックなどを実装している。SVM に基づくマルウェア評価もしくはブラックリスト URL チェックに引っかかる際には脅威評価は「赤」となり、そうでなくとも DroidRisk 評価に引っかかる際には脅威評価は「黄」となる。この例では DroidRisk と SVM の両方の手法に引っかかっているため、脅威評価は「赤」となっている。同様に脆弱性評価結果を見ると、JVN に登録されている脆弱性情報はないものの、コーディングガイド違反をしている項目に赤信号評価がなされている。

資源が有限である各組織にとって、すべてを自前で準備するのは人的リソースの観点から難しい。しかしながら、脅威分析アルゴリズムのように自動分析できる技術を活用したり、JVN などのオープンな情報を再利用したり、また、ほかの組織と連携することにより、効果的な技術、そしてツールを作っていけると考えている。実際、我々は脆弱性分析については、台湾

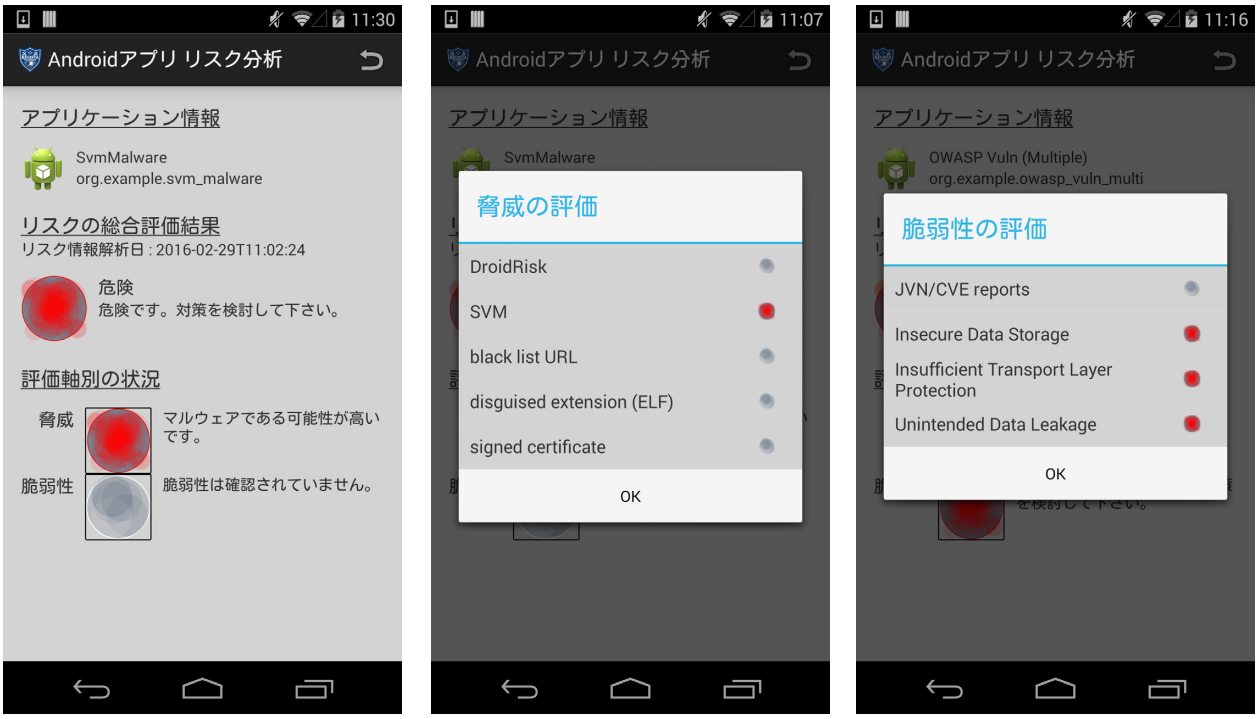

図 1　「Android アプリリスク分析」アプリのスナップショット

の Institute for Information Industry（III）と連携しており、彼らは台湾や日本だけでなく、主要各国のコーディングガイドを調査し、そこから 6 千を超えるコーディングルールを抽出している。これらのルールに照らし合わせて脆弱性チェックを実施することも、彼らと連携することで可能となっている。

5　おわりに

アプリの分析技術は研究開発業界では成熟期に入ってきたと感じる部分もあるが、常に変化し続ける環境を考慮すると、継続的に発展が望まれる。また、これらの技術が実社会に用いられる際には、まだまだ解決しなければならない問題も存在する。例えば、実運用では評価結果の正確性をオペレーションで担保しなければならない。また、データセットの収集・分析の際に法律による制限を超えることができない点や、何をもってマルウェアと定義するかの基準決定が難しい点など [4] も考慮していく必要がある。

謝辞

本研究を実施するにあたり、様々なご支援を頂いた中尾康二主管研究員及び平和昌研究所長に深く感謝する。

【参考文献】

1　G Data, “G DATA RELEASES MOBILE MALWARE REPORT FOR THE THIRD QUARTER OF 2015,” 17 12 2015. [Online]. Available: https://www.gdata-software.com/g-data/newsroom/news/article/g-data-releases-mobile-malware-report-for-the-third-quarter-of-2015.
2　高橋健志，“Android セキュリティ――アプリからユーザー視点までイチから解説，” 10 3 2016. [Online]. Available: http://www.atmarkit.co.jp/ait/articles/1603/10/news011.html.
3　T. Takahashi, T. Ban, T. Mimura, K. Nakao, “Fine-Grained Risk Level Quantification Schemes based on APK Metadata,” The 2015 International Data Mining and Cybersecurity Workshop, 2015.
4　高橋健志, 涛班, “Android アプリの「マルウェア判定」「脆弱性検知」技術，” atmark IT, 28 3 2016. [Online]. Available: http://www.atmarkit.co.jp/ait/articles/1603/28/news002.html.
5　Y. Wang, J. Zheng, C. Sun, S. Mukkamala, “Quantitative security risk assessment of android permissions and applications,” Proceedings of the 27th International Conference on Data and Applications Security and Privacy XXVII, 2013.
6　A. Gorla, I. Tavecchia, F. Gross, A. Zeller, “Checking App Behavior Against App Descriptions,” ICSE'14: Proceedings of the 36th International Conference on Software Engineering, 2014.
7　Wikipedia, “Support vector machine,” [Online]. Available: https://en.wikipedia.org/wiki/Support_vector_machine. [Last access: 25 4 2016].
8　JPCERT/CC and IPA, “Japan Vulnerability Notes,” [Online]. Available: http://jvn.jp/. [Last access:1 2014].
9　National Institute of Standards Technology, “National Vulnerability Database Version 2.2,” [Online]. Available: http://nvd.nist.gov/. [Last access:1 2014].
10　一般社団法人日本スマートフォンセキュリティ協会 (JSSEC)，“Android アプリのセキュア設計・セキュアコーディングガイド，” 2 2016. [Online]. Available: https://www.jssec.org/dl/android_securecoding.pdf.

高橋健志　（たかはし　たけし）
サイバーセキュリティ研究所
サイバーセキュリティ研究室
主任研究員
博士（国際情報通信学）
サイバーセキュリティ、通信プロトコル

班　涛　（ばん　とう）
サイバーセキュリティ研究所
サイバーセキュリティ研究室
主任研究員
博士（工学）
機械学習、ネットワークセキュリティ