複数 Android アプリケーションによる情報集約の脅威分析

小野寺 渓太 †　　金岡 晃 †

† 東邦大学
274-8510 千葉県船橋市三山 2-2-1
6515004o@nc.toho-u.ac.jp, akira.kanaoka@is.sci.toho-u.ac.jp

あらまし　本論文では、複数アプリケーションにおけるアプリケーション間通信に着目し、複数アプリケーションが相互に情報のやり取りを行う部分での情報集約の脅威について調査を行った。実際にマーケット上に存在している約 15,000 のアプリケーションを取得、解析をし、アプリケーション間通信とその脅威の有無について調査をした。この調査により、アプリケーション間の通信による脅威が存在する可能性を指摘する。

Threat Analysis on Information Aggregation between Multiple

Keita Onodera†　　Akira Kanaoka†

†Toho University
Miyama 2-2-1, Funabashi, Chiba 274-8510, Japan
6515004o@nc.toho-u.ac.jp, akira.kanaoka@is.sci.toho-u.ac.jp

Abstract　In this paper, we investigate a threat of information aggregation by multiple android application focusing on inter-application communication. We prepare over 15,000 applications from real application market andanalyze them. From our result, we can see there are threats about information aggreagation.

1　はじめに

Android OS が搭載された端末（以下 Android 端末と呼ぶ）に向けてさまざまなアプリケーションが開発されている。Android OS とそれら Android OS 向けアプリケーション（以下 Android アプリケーションと呼ぶ）に関するセキュリティは、Android 端末の増加に伴い避けることができない大きな問題となっている。

PC 向けの OS である Windows OS や Mac OS、サーバ等でも広範に利用される Linux 等と比較して、Android OS に関してはまだ歴史が浅いこともあるからか、セキュリティに関する検討が十分であるとはいいがたい。それは近年の Android に関連するセキュリティの研究の多さが示している。一方で、その普及は爆発的であり、スマートフォンなどの個人向け端末から家電等への進出などさらなる拡大が予想され、その脅威はますます高くなると言える。Android の OS に関連するセキュリティがまだ成熟していない 1 つの例として、現状の Android OS やその OS を搭載した端末、あるいは Android アプリケーションに対するさまざまな調査が行われ、つぎつぎと発表されている。

それは視点を変えれば、いまだに Android に関連するセキュリティは、現状の把握が追い付いていない状況であり、そのセキュリティを確保する新たな手法の研究開発は、より確実な方向性がまだ捉えられずにいるということも言えよう。

Android アプリケーションに対するさまざまな調査の 1 つとして、複数のアプリケーションの協調による情報集約の脅威がある。これまでの調査研究では、アプリケーション単体がどれほど情報漏えいの脅威を持つかについて強く焦点が当てられてきたが、Android 端末には複数のアプリケーションがインストールされることが可能であり、また Android 端末はその汎用性の高さからスマートフォンなどに代表されるように複数のアプリケーションインストールをされる形態での利用が通常となっている。

Android にはアプリケーション間での通信を可能とする仕組みが複数あり、これらが悪用されることで単体のアプリケーションでは起き得なかった新たな情報漏えいが発生する恐れがある。それらをここでは情報集約による漏えいと呼び、本論文ではその脅威に関する調査を行う。

本論文では、情報集約による漏えいについて、その可能性を論じ、実際に Android のマーケット上に存在するアプリケーションを収集し情報集約による現実的な脅威を調査した。その結果、本調査が対象としたアプリケーション群からは悪意のある複数のアプリケーションによる協調攻撃は発見されなかったものの、暗黙的 Intent により悪意のないアプリケーションが脆弱性を利用して情報集約に悪用される脅威の高さがその利用率から示された。

本論文の構成は以下の通りである。まず第 2 章で関連する研究について触れ、第 3 章では複数アプリケーションによる情報集約の結果起きうる情報漏えいの脅威について分析を行う。第 4 章ではその情報集約の現実性調査の方法と対象について述べ、第 5 章ではその調査結果を示して分析と考察を行う。第 6 章では、本調査に関連して得られた知見について述べ、第 7 章でまとめる。

2 関連研究

Android アプリケーションの脅威を対象とした研究のほとんどが単一 Android アプリケーションを対象としており、複数のアプリケーションを対象としている研究は少ない。

単一のアプリケーションによる情報漏えいの脅威に関する研究は、セカンドアプリと呼ばれる正規のアプリケーションを解凍し悪性アプリケーションを追加し、再圧縮して作成される Android マルウェアを検知する研究 [1] や、端末情報取得 API を実行する際に行われる遠隔手続きを呼び出しによる情報取得に着目し、遠隔手続き呼び出しにおける手続き実行時に呼び出されたアプリケーションを検出し、記録する手法を提案した研究 [2] がある。

一方、複数のアプリケーションによる情報漏えいの脅威に関する研究では、端末にインストールされているアプリケーション同士が持つリンクを通した情報漏えいの脅威をグラフ化し、グラフ化されたデータをベースに情報漏えいを防ぐ LinkDroid [3] や、アクセスを許可したアプリケーションにプライバシー情報を渡すことで起こる情報漏えいを防ぐためにプライバシー情報をデバイス内で処理をし保護することを目的とした π Box を提案した研究 [4] などがある。しかしいずれもその現実的な脅威は限られたアプリケーションを対象にのみ調査されており大規模であるとはいえない。

また広告系ライブラリに対象を絞った Ham らの研究 [5] や Stevens らの研究 [6] もあるが、いずれも特定領域を扱ったものである。

複数アプリケーション間の脅威についてはさまざまな面から注目を浴びているものの大規模に精査されてはおらず、それらの必要性が伺える。

3 情報集約による情報漏えいの脅威

複数のアプリケーションが同一端末に存在していることで、アプリケーション同士がデータの受け渡しを行う直接的な情報集約方法や、共通の外部ライブラリを利用する間接的な情報の集約方法がある。

直接的な情報集約を行う手法として Intent によるアプリケーション間通信や指定したアプリケーションが起動された時に取得された同じリ

ソースやクラスローダを含む Context オブジェクトを返す createPackageContext() がある。

Intent によるアプリケーション間通信には、明示的な Intent と暗黙的な Intent の 2 種類がある。明示的な Intent はやり取りされるデータや動作を指定して通信する、また同一のアプリケーション間での通信を行う方法である。一方、暗黙的な Intent はやり取りされるデータや動作を指定し、その動作を行うアプリケーションをユーザに選ばせる方法である。

ソースコード内での Intent の利用では通信を行うアプリケーションやデータを指定するが、直接アプリケーションのパッケージ名や暗号化していないデータを記載することは少なく、多くの場合はクラスのインスタンスが直接渡される。

createPackageContext() ではアプリケーションのパッケージ名と flags と呼ばれる 2 つの引数を指定する。flags には 3 種類あり、アプリケーションのコードをロードされるときに、クラスのインスタンスを行うため getClassLoder() を使用する CONTEXT_INCLUDE_CODE (0x01)、要求される Context 上の全てのセキュリティ制限を無視し、常に行われるロードを許可する CONTEXT_IGNORE_SECURITY (0x02)、特定の機能を無効にすることができる CONTEXT_RESTRICTED(0x04) がある。

間接的な情報集約を行うライブラリとして広告を表示する広告ライブラリ、アプリケーションが他のサービスを利用する際にそのサービスの認証をさせる認証ライブラリがある。広告ライブラリは特定のアプリケーションを実行した際に、外部にあるサーバやデータベースにアクセス情報を送信し、受け取ったサーバやデータベースから広告情報を取得し、画面上に表示するライブラリである。認証ライブラリは Web サイトの情報を SNS を通して発信するさいに、発信しようとしているユーザがサービスのアカウント情報やログイン情報を持っているのかを通常のログインをさせ、認証させるライブラリである。

こういった広告ライブラリや認証ライブラリはそれぞれのアプリケーションが適用するものであり、利用者が端末にインストールした複数のアプリケーションが同じライブラリを利用する可能性がある。その際、ライブラリの提供者は 1 つの端末より複数のアプリケーションから情報を得ることができ、情報の集約により単一のアプリケーションより深い情報を得ることが可能になる。

4 情報集約の現実性調査

本調査では Google Play Store[1] から 2015 年 1 月 24 日から 2015 年 1 月 28 日までに取得した 907 個と Opera Mobile Store[2] から 2014 年 9 月 3 日から 2014 年 12 月 2 日までに取得した 15,069 個の計 15,976 個の APK(Application PacKage) を使用する。

調査内容は外部ライブラリを対象とした調査と Intent によるアプリケーション間通信と createPackageContext() の利用調査の 2 つとする。

使用したツールは「apktool」「dex2jar」「Java Decompiler」 である

- apkTool により AndroidManifest や Dex ファイルを取得
- dex2jar で Dex ファイルを jar ファイル群に変換
- Java Decompiler で jar ファイル群を Java ソース群に逆コンパイル

外部ライブラリを対象とした調査では逆コンパイルして得た Java ソース群に対して、ソースファイルより import 文に記載のあるパッケージを抽出し、抽出したパッケージからホワイトリスト (Package Index — Android Developers[3] に記載されているパッケージ名) の除外や重複パッケージ名を排除をし、外部アプリとのやり取りをするパッケージがないか、広告系ライブラリや認証系ライブラリの利用状況を確認する。認証系ライブラリは代表的なものとして Twitter, Facebook に限定し、Twitter からは Twitter4J, Facebook からは Facebook Android SDK のライブラリを対象とする。また広告系ライブラリ

[1] https://play.google.com/
[2] http://apps.opera.com/
[3] http://developer.android.com/reference/packages.html

は AppBrain の 「Ad networks」 にある上位 10 件で調査した。

Intent によるアプリケーション間通信の調査では逆コンパイルして得た Java ソース群に対して、ソースファイルより Intent 部を抽出し、暗黙的 Intent と明示的 Intent のそれぞれに対し外部アプリケーションの呼び出しを行っているかを調査する。暗黙的 Intent は Intent クラスの setType メソッドを呼びだしているかを調査する。明示的 Intent は Intent クラスの setClassName メソッドを呼び出しているかの調査と、ComponentName クラスを用いたパッケージ指定をしているかの調査の 2 つを行う。

5 調査結果と考察

外部ライブラリ 外部ライブラリの調査では、Google Play から取得したアプリケーション群のうち、逆コンパイルに成功した 899 個のアプリケーションについて調査を行った。通信をする外部ライブラリをいくつか発見することができた。その中に一つが com.navitime.*である。

com.navitime は NAVITIME が提供している API 群で一部の機能では位置情報を提供する API から現在地を取得し、サーバへ送信し、経路や最寄り駅の検索や住所情報の取得が可能である。法人を対象に API の提供を行っており、API の設定は取引相手との疎通で NAVITIME 側が初期設定をする。

このほか amazon や KDDI の提供している通信系ライブラリを多数発見することができた。amazon は法人だけでなく個人の開発者にも API を提供しているが、API の資格の入手や設定内容が定められ、最終的には開発したアプリケーションを提出をする。さらに Amazon のアカウントが US である必要性やひと月に 1000 個までの商品の情報しか通信で得ることができないといった制限がある。KDDI の提供する API は Google の提供しているマーケットと通信し、アプリケーションの管理を行う API でマーケットに直接通信され、ストアアプリケーションの起動を促すものである。

Intent によるアプリケーション間通信 Intent によるアプリケーション間通信の調査を、OperaMobileStore から取得した 2745 のアプリケーションに対して行った。この調査では、ApkTool の実行により得た AndroidManifext.xml からアプリケーションのパッケージ名を取得し、Java Decompiler によって取得したソースファイルのうちパッケージ名部分についてソースコードの解析を行った。

Intent の利用率を表 1 に、暗黙的 intent の利用率を表 2 に、明示的インテントと外部パッケージの利用率を表 3 にそれぞれまとめた。なお、表 1 のデータは内部 Activity の呼び出しを含んだ数となっている。

表 1: アプリケーションにおける Intent の利用率

利用数	利用率
1798	65.50%

表 2: 暗黙的 Intent の利用率

利用数	利用率
852	31.04%

表 3: 明示的 Intent+外部パッケージの利用率

名称	利用数	利用率
setClassName	408	14.86%
ComponentName	124	4.52%

明示的 Intent で呼び出されたパッケージのうち上位 5 つについてを表 4 に示す。

暗示的 Intent で呼び出された Mime Type のうち上位 5 つについてを表 5 に示す。

認証系ライブラリ 認証系ライブラリの調査では、Opera Mobile Store から取得したアプリケーション群のうち、逆コンパイルに成功した 15,042 個のアプリケーションについて調査を行った。その結果を表 6 に示す。

facebook の SDK は 1671 件の利用があり、全体からみた利用率としては 11.11 %となってい

表 4: 明示的 Intent で呼び出されたパッケージの割合

パッケージ名	利用アプリ数	割合
com.jb.gokeyboard	87	3.17%
com.adobe.air.AIRService	65	2.37%
com.facebook.katana	22	0.80%
android.lgt.appstore	17	0.62%
com.adobe.air.AndroidGcmRegistrationService	15	0.55%

表 5: 暗示的 Intent で呼び出された MimeType の割合

MimeType	利用アプリ数	割合
text/*	596	21.71%
text/plain	517	18.83%
image/*	323	11.77%
plain/text	173	6.30%
message/rfc822	118	4.30%

表 6: 認証系ライブラリ利用状況

名称	利用数	利用率
facebook	1671	11.11%
twitter	385	2.56%

る。また Twitter4J を用いたアプリケーションが 385 件あり、全体からみた率としては 2.56 %となっている。

広告系ライブラリ　広告系ライブラリの調査では、Opera Mobile Store から取得したアプリケーション群のうち、逆コンパイルに成功した 15,042 個のアプリケーションについて調査を行った。その結果を表 7 に示す。

広告系ライブラリでは AdMob の利用が最も多く 20.79 %を占めた。

認証・広告ライブラリの利用率が表 6、表 7 に示されているが、広告ライブラリの利用率の高さに注目したい。複数のアプリケーションを 1 つの端末に入れている場合、端末には 1 つ以上の広告ライブラリが利用されている可能性は高くなり、インストールされたアプリケーションの数によっては同一のライブラリが別のアプリケーションで利用されているケースも十分に起きうる。この際、双方のアプリケーションがそれぞれ異なるパーミッションを持ち、広告ライブラリがそれらパーミッションを利用した情報収集を行っていることがある場合、広告ライブラリ提供者には、単一のパーミッションで得られる情報を超えた情報取得が可能になる。

表 7: 広告系ライブラリ利用状況

名称	利用数	利用率
Admob	3127	20.79%
Charboost	659	4.38%
MoPub	340	2.26%
Millennial Media	1238	8.23%
InMobi	795	5.29%
AdColony	303	2.01%
Tapjoy	362	2.41%
Unity Ads	7	0.05%
Appsflyer	26	0.17%
Vungle	36	0.24%

6　関連調査結果

今回逆コンパイルできない APK ファイルがいくつか存在した。これは有料ソフトウェアの盗難防止やコンピュータウイルスの存在を知られないようにするといった目的で解析を妨げる手法がとられる。この手法を難読化と呼ぶ。難読化する方法として逆コンパイルを困難にするものから、逆コンパイルが可能であっても、開発者以外がプログラムを解読しにくくしたものま

で幅広く存在する。今回の調査ではツールを使用して逆コンパイルできないアプリケーションを難読化しているものとした。今回は難読化されているアプリケーションの数がそれほど多くなかったため、その中でも比較的個数が収集できていて比較しやすいと考えた、Opera Mobile Store で入手した難読化されたアプリケーションを考察の対象にした。最大ページ数が 9000 ページ台であった期間に 1728 ページまで巡回して入手した 72 個のアプリケーションと、それを入手したページ番号の関係を図 1 のグラフで表した。グラフを見ると最初のページ周辺と 1400 ページ前後で密度が高い。一方 700 ページ前後や 1100 ページ台では発見されなかった。1400 ページ前後の密度が高い理由は不明であるが、最初のページ付近に表示されるような人気のアプリケーションは、比較的難読化率が高くなる傾向にあるとグラフから推測した。

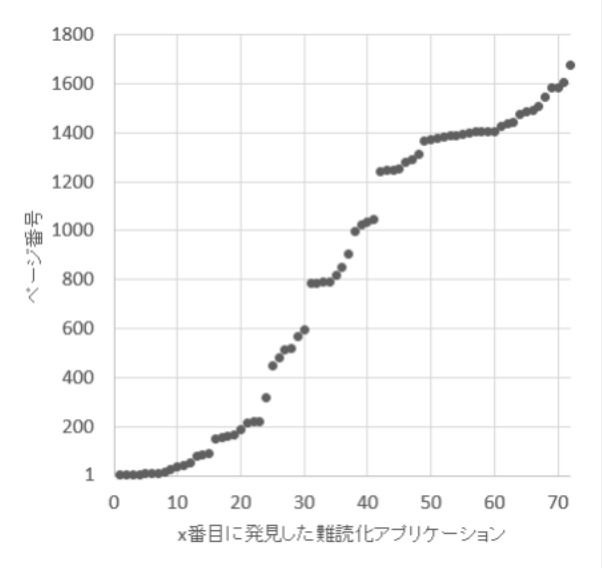

図 1: Opera Mobile Store で入手した難読化アプリケーションのページ番号密度

7 まとめ

本論文では、情報集約による漏えいについて、その可能性を論じ実際に Android のマーケット上に存在するアプリケーションを収集し情報集約による現実的な脅威を調査した。その結果、本調査が対象としたアプリケーション群からは悪意のある複数のアプリケーションによる協調攻撃は発見されなかったものの、暗黙的 Intent により悪意のないアプリケーションが脆弱性を利用して情報集約に悪用される脅威の高さがその利用率から示された。明示的 Intent についても高い利用がされるタイプが示され、それらに脆弱性があった場合は大きな脅威となることも示された。

情報を送信する側がどれだけの情報を送信可能か、そして情報を受信する側がどれだけの情報を受け入れられるかにより、脅威は大きく変化する。さらなる調査を行うことで、その脅威をより深くしることが今後さらに重要となると考える。

参考文献

[1] 磯原隆将, 川端秀明, 竹森敬祐, 窪田歩, 可児潤也, 上松晴信, 西垣正勝”セカンドアプリ内包型 Android マルウェアの検知” コンピュータセキュリティシンポジウム 2011

[2] 西本祐揮, 堀良彰, 櫻井幸一”動的解析を用いた Android における端末情報の取得検知手法” 情報処理学会研究報告 IPSJ SIG Technical Report

[3] Huan Feng, Kassem Fawaz, and Kang G. Shin, University of Michigan ”LinkDroid: Reducing Unregulated Aggregation of App Usage Behaviors” In Proc. of the 24th USENIX Security Symposium (USENIX Security 2015)

[4] Sangmin Lee, Edmund L. Wong, Deepak Goel, Mike Dahlin, and Vitaly Shmatikov ”π Box: A Platform for Privacy-Preserving Apps” In Proc. of the 22rd USENIX Security Symposium (USENIX Security 2013)

[5] S. Han, J. Jung, D. Wetherall. A study of third-party tracking by mobile apps in the wild. UW-CSE-12-03-01, 2011

[6] R. Stevens, C. Gibler, J. Crussell, J. Erickson, and H. Chen, “Investigating user privacy in Android ad libraries,” in Mobile Security Technologies (MoST) 2012, 2012