

DNS クエリ分析に基づく Android PUA の識別と亜種分類

畑田 充弘^{1,2} 森 達哉¹

概要: Potentially Unwanted Application (PUA) は従来のマルウェアとは区別され、ユーザのプライバシーやコンピュータのセキュリティの脅威となっている。しかしながら、限られたリソースでインシデント対応を行う CSIRT において、PUA に起因するインシデント対応の優先度は低い。リスクの高いマルウェアへの対応優先度を上げるためにも、PUA を検出・分類できることは有用である。本研究では、集中監視・制御を行いやすい DNS クエリに着目し、業務利用も増えている Android を対象とする。高精度・高速に PUA とマルウェア、正常アプリを識別し、かつ PUA の亜種名までを分類できることを示す。

キーワード: PUA, PUP, Android, 検知, 分類, DNS クエリ

Detecting Android PUAs and classifying its variants with analysis of DNS queries

MITSUHIRO HATADA^{1,2} TATSUYA MORI¹

Abstract: Potentially Unwanted Application (PUA) is distinguished from traditional malware and poses a threat to user's privacy and computer security. However, In the CSIRT that handles incidents with limited resources, incident response due to PUA has lower priority than malware. It is crucial to be able to detect PUAs and classify its variants even in order to raise priority for malware with high risk. In this paper, we focus on DNS queries that facilitate centralized monitoring and control of network, and target Android, which is increasingly used for business. Our experimental results demonstrate that our proposed method performed to distinguish PUAs from malware and benign applications with high accuracy and low computational time, and also to classify the variants of PUA.

Keywords: PUA, PUP, Android, Detection, Classification, DNS query

1. Introduction

Potentially Unwanted Application (PUA) は従来のマルウェアとは区別され、ユーザのプライバシーやコンピュータのセキュリティの脅威となっている [1]。2017 年 6 月に、Check Point 社が、ゾンビ化する危険性のある Fireball というアドウェアが、世界で 2 億 5 千万台のコンピュータに感染していることを報告している [2]。また、同月 Sophos 社

は、Google Play で公開されている 40 種類以上の Android アプリに隠れていたアドウェア “Judy” が、感染デバイス上の広告に対する不正なクリックを大量に発生させていたことを報告している [3]。PUA は、ブラウザ・ツールバーやリモート監視ツールなど、一見ユーザに利便性を提供するアプリケーションではあるが、インストール時に広告の配信やユーザ情報の収集を暗黙的に同意してしまっていることが多く、通常のアプリ配布サイトからダウンロードできることが多い。一方マルウェアは、ブラウザやそのプラグインの脆弱性を悪用して感染させるドライブ・バイ・ダウンロードや、巧妙に細工されたメールの添付ファイルなどにより、ユーザに気付かせずにインストールさせる点で大きく異なる。

¹ 早稲田大学
Waseda University
{m.hatada, mori}@nsl.cs.waseda.ac.jp
² NTT コミュニケーションズ株式会社
NTT Communications Corporation
m.hatada@ntt.com

Computer Security Incident Response Team (CSIRT) におけるインシデント・ハンドリングでは、Security Operation Center (SOC) や IT ヘルプデスクで検知したアラートに対して、対応優先度を判断 (トリージ) し、詳細な分析や隔離、復旧措置を行う。多種多様なアラートが発生する中で、効率的なトリージ、つまり重要性の高いインシデントへの対応優先度を上げ、比較的重要性の低い PUA を起因とするアラートの対応優先度を下げするためには、PUA を検出できる必要がある。また、検出できた PUA が世の中でリスクが高いと話題となっていたり、前述のような報告 [2,3] があると、端末隔離等の緊急措置がいつでもとれるように調査しておく必要があり、そのためには PUA の亜種分類ができることも有用となる。

本研究では、Android 端末を対象として、PUA が送信する DNS クエリに着目する。Domain Name System (DNS) はネットワークの集中監視・制御が行いやすく、PUA による広告取得や収集したプライバシー情報の送信先など、ドメイン名が PUA 特有のものとなることが考えられる。Android 端末は業務で利用するユーザも多く、公式サイト以外からもアプリのインストールが容易であり、Apple 社の iOS に比べてリスクが高いことが考えられる。関連研究には、アンドロイドマルウェアの検知 [4,5] や分類 [6-8]、PUA の分類学 [9,10]、PUA 流通の実態調査 [11,12]、PUA の分析 [13,14] があるが、我々の知る限り Android PUA の DNS クエリに着目し、検知や亜種分類への有用性を評価したものはない。本研究の主要な貢献を以下の通りである。

- 個々の Android アプリが生成する DNS クエリ (名前解決を行う複数の FQDN) を用いて PUA かマルウェア、正規アプリを識別する手法を提案した。
- Android PUA とマルウェア及び正規アプリをそれぞれ 5,640 個用いた大規模なデータセットによる評価の結果、92%以上の精度で PUA を識別することが可能であり、230 種類以上の PUA を 85.3%の精度で亜種分類できることを示した。
- Android PUA と Windows PUA が生成する DNS クエリは共通性が低いことを明らかにした。
- 既存の DNS ブラックリストは Android PUA の識別・亜種分類に有効性が低いことを示した。

以降、第 2 章でデータセットの概要と提案手法を述べ、第 3 章で比較分析結果と識別・分類結果を報告し、本研究における制約と課題を第 4 章で議論する。第 5 章で関連研究を要約し、第 6 章でまとめる。

2. 提案手法

本章では、図 1 に示す提案手法と評価方法の概要を述べる。

表 1 データセット概要

Android アプリ数	453,687
収集期間	2016 年 6 月～8 月
PUA 数 (PUA 亜種数)	5,640 (237)
マルウェア数 (マルウェア亜種数)	5,640 (393)
正規アプリ数	5,640
Windows PUA 数 (PUA 亜種数)	5,640 (511)
収集期間	2016 年 6 月

2.1 データセット

本研究で利用するデータセットの概要を表 1 に示し、データセット準備の手順を以下 1)~7) で説明する。1) 利用者が多い 12 種類のサードパーティーの Android アプリ配布サイトからダウンロードした 453,687 個の無料 Android アプリ (参考文献 [15] のデータセットの一部) について、複数のアンチウイルスソフトが稼働する VirusTotal [16] で検知結果を取得する。サードパーティーの配布サイトは公式 Google Play に比べて、10 倍以上の不正なアプリを配布しているとの報告 [17] もあり、PUA やマルウェアを効率的に収集できることが考えられる。2) VirusTotal の各アンチウイルスソフトの検知名から、PUA と考えられる特定のキーワード (“pua”, “pup”, “adware”, “unwanted”, “ad”, and “/ad”) に一致するアプリを抽出する。ここで、“pup” は “potentially unwanted program” の略であり PUA の別名として知られている。同じ検知名であっても、異なる動作をするアプリは存在するため、同じ検知名の複数のアプリを利用することで、提案手法の信頼性を高めることを想定している。3) アンチウイルスソフトの中から、10 個以上のアプリが同じ検知名であり、検知の種類 (亜種) が最も多いアンチウイルスソフトは *ESET-NOD32* であったため、これを検知名の基準とする。4) 少なくとも亜種毎に 10 個となるよう各亜種から 5%ずつランダムに抽出し、237 種類の 5,640 個の PUA を得た。5) *ESET-NOD32* で検知したアプリのうち、前述のキーワードに該当する PUA を除いて、PUA と同数の 5,640 個の Android マルウェアをランダムに抽出した。6) *ESET-NOD32* が検知しなかったアプリから PUA とマルウェアと同数の 5,640 個のアプリをランダムに抽出した。7) Windows PUA は 3 種類の公開レポジトリから取得し、*ESET-NOD32* の検知名から前述のキーワードに該当する PUA をランダムに 5,640 個抽出した。

2.2 DNS クエリの抽出

Android アプリが通信するホストの FQDN を取得するため、アプリがプログラム内で生成する通信先 URL を全て抽出する。ここでは、シンボリック実行により広範なコー

*1 <https://malwr.com/>
<http://malshare.com/>
<https://virusshare.com/>

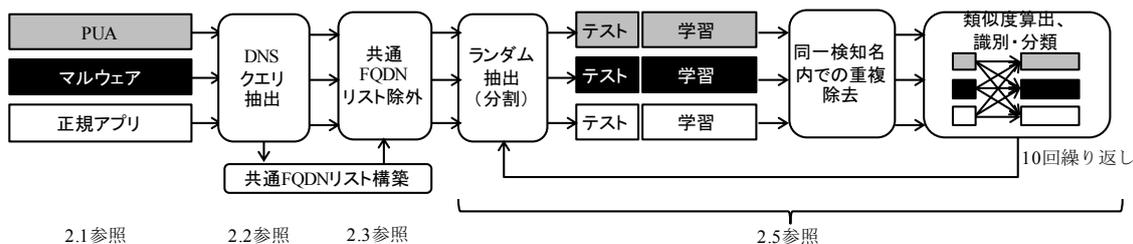


図 1 提案手法と評価方法の概要

ドカパレッジを得て、関数から関数へのデータ伝搬を追跡することができる市販製品を利用した。DNS クエリとして名前解決が行われるのは、抽出した URL のうち FQDN 部分であり、これをアプリ毎にリストアップする。Windows PUA については、Cuckoo sandbox [18] の動的解析で得たパケットキャプチャから Wireshark のコマンドラインツールである *tshark* で DNS クエリを抽出し、名前解決を行っている FQDN をリストアップする。

2.3 共通 FQDN の除去

アプリ特有の通信先 FQDN を重視し、正規アプリ共通で広く利用される通信先 FQDN を除外するために、自然言語の分類精度を高めるために広く用いられる “Domain-specific Stopwords” を FQDN のリストとして構築する。ここでは Document Frequency (DF) を用いる。ある FQDN t が出現するアプリの数を $DF(t)$ とし、総正規アプリ数 N に対する割合 R を下式で定義する。

$$R = \frac{DF(t)}{N}$$

R が大きい FQDN は共通性が高く除外対象となり、 R がある値以上の FQDN を共通 FQDN リストとする。Windows PUA の場合は、Web トラフィックボリュームの多い上位 10,000 のドメイン [19] に該当する FQDN を除外する。

2.4 比較評価と類似度評価

DNS ブラックリストはマルウェア等による通信を検知・防御するために広く利用されており、2.2 で抽出した FQDN を既存のブラックリストと比較することで、既存のブラックリストが Android PUA の識別・分類に有用であるかを評価する。

さらに、2つの Android アプリの FQDN リスト (X と Y) の類似度を Jaccard 係数を用いて算出する (下式)。

$$J(X, Y) = \frac{|X \cap Y|}{|X \cup Y|} = \frac{|X \cap Y|}{|X| + |Y| - |X \cap Y|}$$

$$0 \leq J(X, Y) \leq 1$$

2つのアプリ間で共通する FQDN がない場合、 $J(X, Y)$ は

0 となり、全ての FQDN が一致する場合 $J(X, Y)$ は 1 となる。この時、FQDN は Second Level Domain 等による部分一致ではなく、FQDN として完全に一致するかどうかを判定している。Android PUA 全ての組み合わせについて J を算出し、検知名毎に J の平均値を求め、同じ PUA と異なる PUA での類似度を評価する。

2.5 Android PUA の識別と亜種分類

PUA, マルウェア, 正規アプリからそれぞれ 10%(564 個) をランダムに抽出してテストデータとし、残りの 90% を学習データとする。DNS クエリが発生しないアプリは本研究の対象外となるため、テストデータ、学習データの両方から除外する。類似度 J の計算コスト削減のため、同じ検知名で完全に FQDN リストが一致するアプリは学習データからは除外する。テストデータに対して学習データ全てとそれぞれ J を算出し、 J が最大となる学習データの種別 (PUA, マルウェア, 正規アプリ) によって、識別結果を得る。同時に、 J が最大となる学習データの検知名によって亜種分類結果を得る。実験結果の信頼性向上のため、各 R において、テストデータのランダム抽出から、識別結果・亜種分類結果を得るまでのステップを 10 回繰り返す。 R に従って除外される FQDN は変動し、除外されるテストデータが生じることで、識別・分類対象となるアプリの対象数も変動する。そこで、識別・分類の評価指標として、精度 (Accuracy) とテストデータ率 (564 アプリのうち、除外されたテストデータを除いた有効なテストデータの割合) を乗算したスコアを用いる。

3. 実験結果

本章では、2.4 及び 2.5 で述べた評価実験の結果を示す。

3.1 FQDN リストの比較評価

表 2 に各種別 ((1)Android PUA, (2)Android マルウェア, (3)Android 正規アプリ, (4)Windows PUA) におけるユニークな FQDN 数, 各アプリの平均 FQDN 数, 種別間で共通する FQDN 数をまとめた。Android PUA とマルウェアは正規アプリに比べて FQDN が多く、ユニーク FQDN の約 30% は Android に共通であることがわかつ

表 2 各種別における DNS クエリの FQDN 数とブラックリスト照合結果

種別	平均 クエリ数	ユニーク FQDN 数	(1)	(2)	(3)	(4)	EasyList (of 13,620)	ad server (of 2,377)	dns-bh (of 33,302)
(1) Android PUA	34.4	5,124	n/a	1,780	1,563	18	79	10	0
(2) Android マルウェア	37.2	5,259	1,780	n/a	1,498	21	65	7	0
(3) Android 正規アプリ	23.7	9,502	1,563	1,498	n/a	22	96	10	0
(4) Windows PUA	5.3	635	18	21	22	n/a	5	1	9

た。そのため、共通 FQDN の除外が有効に働くことが期待できる。また、18 個の FQDN のみが Android PUA と Windows PUA に共通するものであり、そのうち 11 個は *accounts.google.com* 等の Google 関連のものであった。6 個はブラウザツールバーや Amazon S3, Facebook, ニュースサイト等であり、1 個は RFC 6761 [20] で文書用ドメインとして規定されている *example.com* で、テストあるいはデバッグ用のアクセス先としてアプリに実装された可能性が考えられる。

一般に入手可能な 3 種類のブラックリストと各種別 (1)~(4) の一致数から、Android PUA やマルウェア、Windows PUA の検知には既存のブラックリストの有用性には限界があることを示している。EasyList [21] は広告関連のキーワードや FQDN, ad server [22] は広告関連の FQDN がリストになっており、主要な広告除外ツールで利用されているものである。Malware domain blacklist [23] はマルウェアやスパイウェア対策のプロジェクトであり、関連するドメインが掲載されている。これらのリストでもし検知できたとしても、どの PUA やマルウェアによるものかということ判断できる情報は不足しており、インシデント発生時の対応優先度の判断は困難である。

3.1.1 PUA の類似度評価

Android PUA に関して、同一の検知名のアプリ間、異なる検知名のアプリ間における FQDN リストの平均類似度 J の累積度数分布を図 2 に示す。ここで、同一検知名の組合せは 237, 異なる組合せは 27,966 がそれぞれの母数となっており、図中 n/a は比較対象アプリの両方ともに FQDN がない場合の組合せである。紙面の都合上、後述する亜種分類で最もスコアの高かった $R = 0.0020$ の結果のみを掲載している。異なる検知名の場合、66.8%の組合せが $J = 0.0$ となっており、FQDN リストに類似性がなく、97.3%の組合せで $J < 0.1$ となっていることから、異なる検知名の PUA の DNS クエリは類似度が低いことがわかる。一方で、同一検知名の場合は、6.3%のみが $J = 0.0$ であり、65.8%は $J \geq 0.1$ であることから、同じ検知名の PUA の DNS クエリは類似度が高いことがわかる。

3.2 PUA の識別

PUA の識別の前処理として行う 2.3 で示した、共通 FQDN の除外例を図 3 に示す。Google や Twitter, Face-

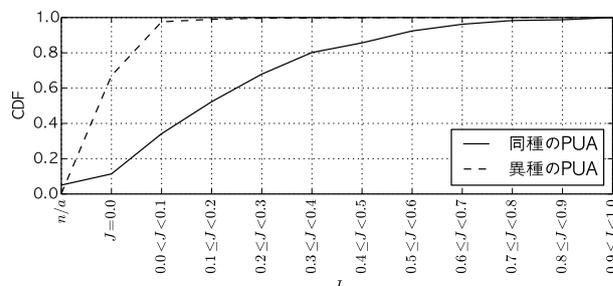


図 2 同種または異種 PUA の類似度 J の累積度数分布

アプリから抽出したFQDN	共通FQDN除外後のFQDN
a.admob.com a.vserv.mobi adlog.flurry.com ads.flurry.com c.admob.com data.flurry.com googleads.gdoubleclick.net in.sb.vserv.mobi market.android.com play.google.com schemas.android.com sites.google.com sns.vserv.mobi www.facebook.com www.googleadservices.com www.gstatic.com www.twitter.com	a.vserv.mobi in.sb.vserv.mobi sns.vserv.mobi

図 3 a variant of Android/Adware. Viser.B から抽出した FQDN と共通 FQDN 除外の例

book 等の FQDN が共通 FQDN として除外されていることが確認できるが、過剰に除外してしまっている可能性もあり、4 で議論する。

表 3 に Android の各種別の識別結果を示す。Android PUA は 92.9%の精度で識別できており、マルウェアと正規アプリにおいても、それぞれ 91.7%, 96.0%と高い精度で識別が可能であることがわかる。ここで、複数の種別で同じ J と識別した場合は、それぞれ“PUA or マルウェア”, “PUA or 正規アプリ”, “マルウェア or 正規アプリ”, “PUA, マルウェア, or 正規アプリ”として表中に記載している。“PUA or malware”と判定した PUA とマルウェアがわずかに他と比較して多い。これらは合計 3%未満に過ぎず、正しい識別結果も含んでいるため、一意に識別できなかったとしても、対応優先度の判断材料として有意であると考えられる。マルウェアの場合のみ、約 0.9%が識別でき

表 3 PUA, マルウェア, 正規アプリの識別精度 ($R = 0.0020$).

	PUA	マルウェア	正規アプリ	PUA or マルウェア	PUA or 正規アプリ	マルウェア or 正規アプリ	PUA, マルウェア or 正規アプリ
PUA	0.9291	0.0318	0.0066	0.0249	0.0040	0.0015	0.0021
マルウェア	0.0368	0.9170	0.0095	0.0182	0.0032	0.0052	0.0013
正規アプリ	0.0129	0.0168	0.9602	0.0016	0.0039	0.0045	0.0000

表 4 マルウェアと誤識別した PUA の上位 5 種類

誤識別数	誤識別した PUA	誤識別されたマルウェア
8	<i>a variant of Android/Domob.G potentially unwanted</i>	<i>probably a variant of Android/Domob.G</i>
8	<i>a variant of Android/AdDisplay.AdsWo.A potentially unwanted</i>	<i>a variant of Android/Domob.F</i>
5	<i>a variant of Android/AdMogo.A potentially unwanted</i>	<i>probably a variant of Android/Domob.G</i>
5	<i>a variant of Android/AdDisplay.AdsWo.A potentially unwanted</i>	<i>probably a variant of Android/Domob.G</i>
4	<i>probably a variant of Android/Adware.Youmi.B</i>	<i>probably a variant of Android/Domob.B</i>

なかったが、これらはテストデータが学習データの FQDN と全く一致しなかったことを意味しており、2.1 で述べたように、PUA のように検知名毎に最低 10 アプリを抽出するのではなく、単純に同数のマルウェアをランダムに抽出したことに起因する。

表 4 は PUA をマルウェアと誤って識別した上位 5 種類の組合せを示している。検知名からも *Domob* ファミリーが主要な誤識別の要因となっていることがわかる。*Domob*^{*3}は中国のモバイル広告ネットワークであり、アプリに組み込むための SDK も提供している。PUA とマルウェアの両方で *Domob* ファミリーは 1,748 個あり、99% は中国の主要なアプリ配布サイトである *Baidu* アプリストアから収集したものであった。アンチウイルスソフトによる命名規則の微妙な違い、例えば *a variant of Android/Domob.G potentially unwanted* と *probably a variant of Android/Domob.G* も誤分類の要因であり、マルウェアを PUA として誤識別した例も同様の傾向が見られた。PUA を抽出する際のキーワードに大きく関連し、4 章で議論する。

3.3 PUA の亜種分類

図 4 に、 R に対する Android PUA 及び Android マルウェアの亜種分類の精度とテストデータ率、スコアの変化を示し、表 5 にその詳細なデータを示す。PUA, マルウェアともに R を大きくすると分類精度も高くなるが、テストデータ率は低くなる傾向がわかる。スコアが最も高くなった時の分類精度は、PUA の場合は $R = 0.0020$ で 85.3%, マルウェアの場合は $R = 0.0205$ で 86.9% であった。PUA に比べて全般的にマルウェアの亜種分類精度が高く、マルウェアの方が亜種によって生成する DNS クエリに差異があることが考えられる。1 アプリの分類にかかる処理時間は、Dell PowerEdge R210 II (CPU: 2.3 GHz Intel Xeon E3-1220Lv2, Memory: 32GB 1333MHz UDIMM) にインストールした Ubuntu 14.04.5 LTS 上の Python 2.7 による

実装で、 $R = 0.0020$ の時に PUA では 0.034 s, $R = 0.0205$ の時にマルウェアでは 0.042 s であった。テストデータはランダム抽出したものの、 $R = 0.0020$ の時に PUA の 237 種類のうち 234 種類がテストデータに含まれており、98.7% をカバーしている。169 種類は分類精度が 70% を超え、7 種類のみが 30% 未満の分類精度となっている。 $R = 0.0205$ の時にマルウェアは 393 種類のうち 261 種類 (66.4%) がテストデータに含まれており、PUA と比較してカバー率は低いが、各亜種毎に 10 アプリ以上となるよう PUA と同様にデータセットを準備することでカバー率の向上は期待できる。

2.3 で述べた共通 FQDN を除外しない場合に比べて、亜種分類の精度は $R = 0.0020$ の時に PUA では 9.3%, マルウェアでは 1.9% 向上しており、特に PUA の場合は共通 FQDN の除外が有効であることがわかる。また、分類にかかる計算時間が削減されていることもわかる。表中の“候補含有率”は、同一の類似度となった複数の分類候補があった場合に、正しい分類結果を含む割合を示している。“候補含有率”は PUA, マルウェアともに 2.5% 以下であり、該当するアプリはほぼ異なる DNS クエリを生成するため、誤分類の要因となることが考えられる。

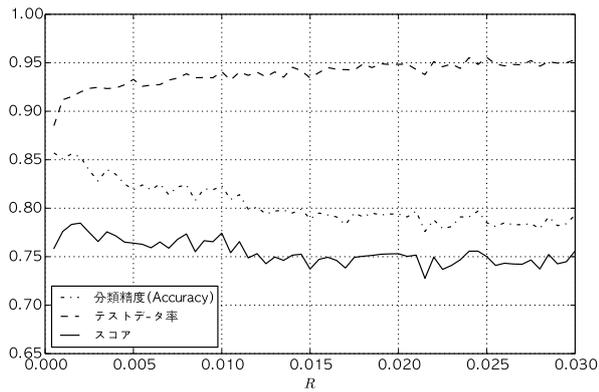
4. 考察

本研究における制約と適用例の課題を考察する。

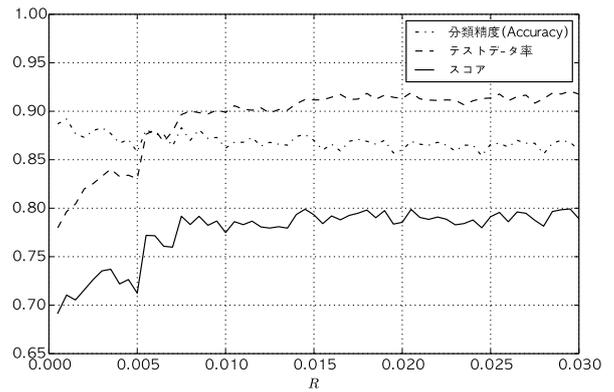
4.1 制約

プラットフォーム: 本研究では Android プラットフォームが主な対象であり、Apple の iOS や Windows Mobile など他のプラットフォームは対象外としている。プラットフォームのカバレッジを高めるためには、それぞれのアプリストアから広くアプリを収集し、プラットフォームに応じた手法でアプリが生成する DNS クエリの抽出から行う必要がある。DNS クエリ抽出以降のアプローチについては提案手法が適用できるものと考えている。

*3 <http://www.domob.cn/>



(a) PUA



(b) マルウェア

図 4 亜種分類精度と有効テストデータ率に基づくスコア

表 5 亜種分類の精度
(a) PUA

R (除外 FQDN 数)	精度	J の平均	J の標準偏差	候補含有率	学習データ数	テストデータ数	処理時間 [s]	スコア
0.0010 (685)	0.851	0.635	0.182	0.024	7606.4	514.3	0.032	0.776
0.0020 (373)	0.853	0.639	0.180	0.025	7991.1	518.8	0.034	0.785
0.0050 (175)	0.819	0.650	0.180	0.019	8292.6	526.1	0.037	0.764
0.0100 (91)	0.823	0.658	0.186	0.018	8631.2	530.6	0.040	0.774
0.0150 (62)	0.789	0.671	0.184	0.015	8806.6	527.0	0.044	0.737
0.0200 (53)	0.794	0.666	0.187	0.018	8861.7	534.9	0.043	0.753
0.0205 (52)	0.791	0.669	0.184	0.014	8902.8	535.0	0.043	0.750
0.0250 (42)	0.785	0.680	0.183	0.012	9004.1	538.9	0.046	0.750
0.0300 (38)	0.793	0.679	0.184	0.008	9018.0	537.6	0.045	0.756
除外無 (0)	0.760	0.735	0.169	0.004	9978.6	545.2	0.060	0.735

(b) マルウェア

R (除外 FQDN 数)	精度	J の平均	J の標準偏差	候補含有率	学習データ数	テストデータ数	処理時間 [s]	スコア
0.0010 (685)	0.892	0.669	0.185	0.015	7597.5	449.2	0.031	0.710
0.0020 (373)	0.873	0.674	0.185	0.023	7978.6	462.4	0.033	0.716
0.0050 (175)	0.858	0.692	0.182	0.014	8279.5	468.3	0.036	0.712
0.0100 (91)	0.862	0.722	0.161	0.007	8636.7	507.0	0.040	0.775
0.0150 (62)	0.870	0.730	0.158	0.001	8829.1	514.2	0.043	0.793
0.0200 (53)	0.860	0.727	0.163	0.007	8899.0	515.1	0.043	0.785
0.0205 (52)	0.869	0.727	0.165	0.005	8900.7	518.4	0.042	0.799
0.0250 (42)	0.866	0.740	0.158	0.005	8964.5	515.3	0.044	0.791
0.0300 (38)	0.860	0.737	0.160	0.004	9007.3	517.5	0.044	0.789
除外無 (0)	0.854	0.798	0.151	0.001	9934.0	522.5	0.061	0.791

PUA 抽出条件: アンチウイルスソフトによる検知名のキーワードをもとに PUA の抽出を行ったが、ベンダ特有の命名規則に大きく依存する。また PUA ではないアプリを誤って抽出してしまう可能性もあり得る。複数のアンチウイルスソフトの検知名を集約することによって共通的な検知名を割り当てる技術 [24] の適用によって、適切なラベルに基づく識別及び亜種分類の精度向上の可能性がある。
FQDN 抽出: 2.2 で述べた通り、本研究ではアプリがアクセスする可能性のある URL を全て抽出し FQDN をリスト化しているが、アプリによってはユーザ操作などの条件次第で実際にアクセスするかどうかが変わってくるのが考えられる。Android アプリの動的解析によって、実際にア

クセスする URL のみを抽出することで、識別及び亜種分類の精度は低下する可能性もある。その際、典型的なユーザ操作を再現する動的解析技術そのものも課題と考えられる。

共通 FQDN 除外: R の減少、つまり除外する FQDN 数の増加に伴うテストデータ数の減少は、対象とする亜種数のカバレッジと識別及び亜種分類精度のトレードオフとなる。3.2 で述べたように、共通 FQDN の除外によって、いくつかの PUA の種類特有の FQDN も除外してしまう可能性がある。PUA の識別や亜種分類の精度をさらに向上させるためには、公式 Google Play 上の正規アプリも多数含めた共通 FQDN リストの作成が有効となる可能性がある。

また、共通 FQDN リストの作成に Document Frequency を用いたがより適切なアルゴリズムの採用も有効となる可能性がある。

4.2 適用例

提案手法をネットワーク監視における PUA 検知に適用するためには、DNS クエリの送信元 IP アドレスをもとに、各端末毎の DNS クエリに分離し、一定時間内における DNS クエリのセットに分割してテストデータとする必要がある。PUA やマルウェア、正規アプリ、ユーザが利用するブラウザ等、様々な DNS クエリが含まれる可能性があるが、FQDN リストの類似度をもとに判定する方式のため、調査したい端末が感染しているかもしれない PUA やマルウェアの候補を提示することが可能である。

他の適用例としては、“Malicious DNS query” といったアラートの通知を CSIRT が受けた場合に、名前解決しようとした FQDN を本提案手法で構築したデータベースに照会することで、PUA かマルウェアかを識別し、PUA であれば、CSIRT は他に発生している優先度の高いアラートに対応するといった効果的なトリアージが可能となる。継続的なアプリの収集・解析と蓄積が、識別・亜種分類の精度維持にとって不可欠であるが、発見されないような様々な方法で隠蔽されている Windows のマルウェアとは異なり、Android アプリはアプリ配布サイトからダウンロードすることが容易であるため、マルウェアに比べてデータ蓄積の障壁は低いことが考えらえる。

5. 関連研究

Android に限らずマルウェアの解析、検知、分類に関する研究は幅広く行われ、ここ数年、PUA の脅威への関心が高まりに応じて PUA の流通に関する大規模な実態調査も行われている。Android PUA に特化して、Android マルウェアや正規アプリとの識別や亜種分類のために、DNS クエリの有効性や精度等について報告するのは、我々が調べた限り本研究が初めてである。以下、Android マルウェアや PUA の関連研究を要約する。

Android マルウェアの検知: MADAM [4] はホストベースの Android マルウェア検知システムを提案しており、アプリのメタデータやユーザ動作、SMS の送信、システムコールなど様々なレベルの異なる特徴を検知に利用している。PUA を含む 7 種類の悪性挙動パターンが定義されており、特徴と悪性挙動パターンが関連付けられ、96% 以上の検知率を達成している。CREDROID [5] は VirusTotal を利用した Android マルウェアの検知手法であり、アプリがアクセスする URL のレピュテーション (Web of Trust (<https://www.mywot.com/>)) や、送信するデータ、通信プロトコルによってスコアを算出する。49 ファミリー 1,260 サンプルの調査で、約 63% のアプリはネットワーク通信を

行うことも報告されている。

Android マルウェアの分類: DroidScribe [6] は実行時の動作による Android マルウェアの分類手法を提案しており、ネットワークやファイルアクセス、ファイル実行等を特徴量として利用している。SVM を拡張した Conformal Prediction によって、5,246 の Android マルウェアを 94% の精度で分類している。DroidSieve [7] は Android マルウェア検知とファミリー識別のシステムを提案している。埋め込まれているリソースとネイティブコードが検知のための静的な特徴として有用であり、検知とファミリー識別を 99% の精度で行っている。Monet [8] は Android 上でマルウェアを検知する軽量なシステムであり、アプリのロジック構造と実行時の動作に基づくシグネチャによって検知を行う。3,723 のマルウェアと 500 の正規アプリを評価に用い、マルウェアの種類を約 99% の精度で検知している。

PUA の分類学: Zhou ら [9] は文献 [5] と同じデータセットをシステムチックに分析し、86% は正規アプリをパッケージしたものであり、93% はボットと同じ機能を有していることを報告している。また、4 種類のアンチウイルスソフトの検知率は最高で 79.6%、最低で 20.2% であり、モバイルアプリにおけるより良いアンチウイルスソフトの必要性を 2012 年に提唱した研究でもある。Svajcer ら [10] は、セキュリティベンダや評価機関が準拠することを想定して、モバイルアプリにおける PUA の分類基準の定義を紹介している。

PUA の流通: Kotzias ら [11] は pay-per-install (PPI) サービスを通じた PUA の流通状況の大規模調査を行った。調査対象の 390 万ホストのうち 54% に PUA がインストールされており、65% の PUA はさらに他の PUA をインストールしていたことを報告している。また 25% の PUA は 23 種類の PPI サービスによって流通していることも明らかにした。Thomas ら [12] は主要な 4 つの PPI サービスが毎週 160 種類のソフトウェアを配信していることを報告している。そのうち 59% は少なくとも 1 つのアンチウイルスソフトで “unwanted” と認識され、ランサムウェアの挙動や、アンチウイルスソフトまたは仮想環境を検知・回避する機能を有していることも示している。また、Google Safe Browsing は PPI サービスによるダウンロード時に 350 万件のアラートを挙げていることも報告している。

Analysis of PUAs: Andow ら [13] はモバイルグレイウェアのトリアージのために、テキスト分析と静的コード解析を用いた軽量なヒューリスティック検知手法を提案している。インストール時や実行時の動作に基づいて、グレイウェアの 9 種類のカテゴリを定義しており、Google Play 上で 100 万アプリから数十アプリにグレイウェアを絞り込むことを達成している。Chen ら [14] は、正規ライブラリとしてリパッケージされマルウェアとしても伝搬して組み込まれている潜在的に危険なライブラリ (PhaLib:

Potentially Harmful Libraries) を、Android と iOS の両プラットフォームに渡って分析している。Android では 130 万以上のアプリの中で 117 種類、iOS では 14 万アプリの中で 46 種類の PhaLib を確認したことを報告している。

6. まとめ

従来のマルウェアとは区別され、近年では脅威として認識されている PUA について、Android プラットフォームを対象に識別・亜種分類に DNS クエリが有用であることを示した。マルウェアや正規アプリと PUA を 92% 以上の精度で識別し、230 種類以上の PUA を 85.3% の精度で亜種分類することに成功した。また、識別・亜種分類の処理速度は 1 アプリあたり 0.034 s であった。提案手法において、Document Frequency に基づく Android アプリの共通 FQDN 除外は、識別・亜種分類においてマルウェアよりも PUA の方がより有効であることを確認できた。また、Android PUA と Windows PUA では DNS クエリに共通性が低いことも示した。提案手法のインシデント対応における適用例を考察し、効率的なトリアージに有用である可能性を示した。

謝辞 本研究の一部は JSPS 科研費 16H02832 の助成を受けたものです。データセットの収集に協力いただいた石井悠太氏と水野翔氏に感謝します。

参考文献

- [1] “Threat intelligence report - Unwanted software.” https://www.microsoft.com/security/portal/enterprise/threatreports_october_2015.aspx.
- [2] “FIREBALL - The Chinese Malware of 250 Million Computers Infected.” <http://blog.checkpoint.com/2017/06/01/fireball-chinese-malware-250-million-infection/>.
- [3] “Punching down the Judy Android adware: a SophosLabs analysis.” <https://nakedsecurity.sophos.com/2017/06/05/punching-down-the-judy-android-malware-a-sophoslabs-analysis/>.
- [4] A. Saracino, D. Sgandurra, G. Dini, and F. Martinelli, “MADAM: Effective and Efficient Behavior-based Android Malware Detection and Prevention,” *IEEE Transactions on Dependable and Secure Computing*, vol. PP, no. 99, pp. 1–1, 2016.
- [5] J. Malik and R. Kaushal, “CREDROID: Android Malware Detection by Network Traffic Analysis,” in *Proceedings of the 1st ACM Workshop on Privacy-Aware Mobile Computing (PAMCO '16)*, pp. 28–36, Jul. 2016.
- [6] S. K. Dash, G. Suarez-Tangil, S. Khan, K. Tam, M. Ahmadi, J. Kinder, and L. Cavallaro, “Droidscribe: Classifying android malware based on runtime behavior,” in *Proceedings of the 2016 IEEE Security and Privacy Workshops (SPW 2016)*, pp. 252–261, May 2016.
- [7] G. Suarez-Tangil, S. K. Dash, M. Ahmadi, J. Kinder, G. Giacinto, and L. Cavallaro, “Droidsieve: Fast and accurate classification of obfuscated android malware,” in *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy (CO-DASPY '17)*, pp. 309–320, Mar. 2017.
- [8] M. Sun, X. Li, J. C. S. Lui, R. T. B. Ma, and Z. Liang, “Monet: A user-oriented behavior-based malware variants detection system for android,” *IEEE Transactions on Information Forensics and Security*, vol. 12, pp. 1103–1112, May 2017.
- [9] Y. Zhou and X. Jiang, “Dissecting Android Malware: Characterization and Evolution,” in *Proceedings of the 2012 IEEE Symposium on Security and Privacy (S&P 2012)*, pp. 95–109, May 2012.
- [10] V. Svajcer and S. McDonald, “Classifying PUAs in the Mobile Environment,” in *Virus Bulletin Conference*, Oct. 2013.
- [11] P. Kotzias, L. Bilge, and J. Caballero, “Measuring PUP Prevalence and PUP Distribution through Pay-Per-Install Services,” in *Proceedings of the 25th USENIX Security Symposium (USENIX Security 16)*, pp. 739–756, Aug. 2016.
- [12] K. Thomas, J. A. E. Crespo, R. Rasti, J.-M. Picod, C. Phillips, M.-A. Decoste, C. Sharp, F. Tirelo, A. Tofigh, M.-A. Courteau, L. Ballard, R. Shield, N. Jagpal, M. A. Rajab, P. Mavrommatis, N. Provos, E. Bursztein, and D. McCoy, “Investigating Commercial Pay-Per-Install and the Distribution of Unwanted Software,” in *Proceedings of the 25th USENIX Security Symposium USENIX Security 16*, pp. 721–739, Aug. 2016.
- [13] B. Andow, A. Nadkarni, B. Bassett, W. Enck, and T. Xie, “A Study of Grayware on Google Play,” in *Proceedings of the 2016 IEEE Security and Privacy Workshops (SPW 2016)*, pp. 224–233, May 2016.
- [14] K. Chen, X. Wang, Y. Chen, P. Wang, Y. Lee, X. Wang, B. Ma, A. Wang, Y. Zhang, and W. Zou, “Following Devil’s Footprints: Cross-Platform Analysis of Potentially Harmful Libraries on Android and iOS,” in *Proceedings of the 37th IEEE Symposium on Security and Privacy (S&P 2016)*, pp. 357–376, May 2016.
- [15] Y. Ishii, T. Watanabe, M. Akiyama, and T. Mori, “Clone or Relative?: Understanding the Origins of Similar Android Apps,” in *Proceedings of the 2016 ACM on International Workshop on Security And Privacy Analytics (IWSPA '16)*, pp. 25–32, Mar. 2016.
- [16] “VirusTotal - Free Online Virus, Malware and URL Scanner.” <https://www.virustotal.com/>.
- [17] Y. Kikuchi, H. Mori, H. Nakano, K. Yoshioka, T. Matsumoto, and M. van Eeten, “Evaluating Malware Mitigation by Android Market Operators,” in *9th Workshop on Cyber Security Experimentation and Test (CSET 16)*, pp. 1–8, Aug. 2016.
- [18] “Cuckoo Sandbox: Automated Malware Analysis.” <https://www.cuckoosandbox.org/>.
- [19] “Alexa Top Sites.” <http://www.alexa.com/topsites>.
- [20] “Special-Use Domain Names.” <https://tools.ietf.org/html/rfc6761>.
- [21] “EasyList.” <https://easylist-downloads.adblockplus.org/easylist.txt>.
- [22] “Blocking with ad server and tracking server hostnames.” <https://pgl.yoyo.org/as>.
- [23] “DNS-BH - Malware Domain Blacklist.” <http://www.malwaredomains.com/>.
- [24] M. Sebastián, R. Rivera, P. Kotzias, and J. Caballero, “AVClass: A Tool for Massive Malware Labeling,” in *Proceedings of the 19th International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2016)*, pp. 1–22, Sep. 2016.