Vol.2013-CSEC-62 No.60
Vol.2013-SPT-6 No.60
2013/7/19

コード解析を伴わないAndroidマルウェア検出方法の検証

岩本 一樹[1,2,a)]　西田 雅太[1,b)]　和﨑 克己[2,c)]

概要：大幅に増加するAndroidのマルウェアを効率よく検出するために，我々はAndroidアプリのDalvikのバイトコードを静的解析することでマルウェアを検出する方法を提案してきた．しかし我々が提案するバイトコードの静的解析だけでは，完全に新規に作成されたマルウェアやバイトコード以外にマルウェアとしての原因がある場合，たとえばネイティブコードやHTML + CSS + JavaScriptでマルウェアが作成されているときには検出できない．マルウェアの検出率を高めるためには，我々のこれまでの提案と別の方法を組み合わせて，複数の方法でマルウェアの検出を試みる1つのシステムを構築する必要がある．そこでバイトコードを解析せずにマルウェアを検出する方法として我々が今回提案するAndroidアプリを配布するWebサイトのドメインからマルウェアを特定する方法に加えて，これまでに提案されたセカンドアプリを内包するアプリを見つける方法と署名情報を利用する方法について検証を行った．検証の結果，これらの方法でマルウェアを検出できる可能性を確かめることができた．

Evaluation of Android Malware Detection Method without Code Analysis

Kazuki Iwamoto[1,2,a)]　Masata Nishida[1,b)]　Katsumi Wasaki[2,c)]

Abstract: In order to efficiently detect Android malware to increase significantly, we proposed the method to detect Android malware by analyzing Dalvik bytecode statically. But the method we proposed can not detect malware which was created completely new or whose cause is out of bytecode (e.g. Native code or HTML+CSS+JavaScript). For increasing the detection rate, it is necessary to build a malware detection system by combining our previous proposal with other methods. We evaluated the method without analyzing bytecode to detect malware from the domain of web sites which distribute Android apps. We also evaluated the methods proposed by other researchers, which are second app and digital certificates. Finally, we confirmed the possibility of detecting malware by these methods.

1. はじめに

Androidを対象とするマルウェアは2012年に入って大幅に増加している[1]．そのすべてを技術者が解析することは困難であるため，マルウェアを効率よく検出できる方法が必要である．

[1] 株式会社セキュアブレイン 先端技術研究所
Advanced Research Laboratory, SecureBrain Corporation, Chiyodaku, Tokyo, 102-0083
[2] 信州大学大学院総合工学系研究科
Interdisciplinary Graduate School of Science and Technology, Shinshu University, Matsumoto, Nagano, 390-8621
[a)] kazuki_iwamoto@securebrain.co.jp
[b)] masata_nishida@securebrain.co.jp
[c)] wasaki@cs.shinshu-u.ac.jp

我々はこれまでにAndroidのアプリを静的解析することでマルウェアを検出する方法[2]を提案してきた．文献[2]の提案手法では，Androidアプリの Dalvikのバイトコードを逆アセンブルし，制御フロー解析により導かれたグラフを比較する．これまでの結果では，グラフ構造というヒューリスティックな定義ファイルにより既知のマルウェアから未知のマルウェアを検出することができた．しかし制御フロー解析やグラフの比較に多くの計算時間を要する点や，バイトコード以外の部分にマルウェアの原因がある場合に対応できていないといった問題点がある．また，マルウェアを実際に実行して動作を見ることでマルウェアの検出を試みる方法[3], [4], [5]も提案されている．いずれもその方法を実現するためのシステムを構築が必要であり，

実行にも時間がかかるという問題がある．

一方，マルウェアのコードや動作を解析せずに，マルウェアと推定されるアプリを検出する方法 [6], [7] も提案されている．これらの方法は，既存のプログラムの組み合わせで検出システムを構築できるため実装が比較的容易であり，実行時間も短くて済む．これらの検出方法を前述の方法と組み合わせれば，より効率よくマルウェアを検出することが期待できる．

本研究では既に提案された方法の有効性を再検証し，またそれらとは別にコード解析を伴わない方法を 1 つ提案する．

2. 本研究の目的と提案

2.1 本研究の目的

我々は Android のアプリを静的解析し，制御フロー解析の結果のグラフを比較することでマルウェアを検出する方法 [2] を提案した．この提案した方法では解析できなかった 1,292 種類を除く 121,131 種類のアプリのうち，1,987 種類を Black（マルウェア）とした．残りの 119,153 種類は Ivory または White であり，提案した方法ではマルウェアであるとは判断していない．提案した方法で検出を行えるのは Dalvik のバイトコードにマルウェアとしての原因がある場合だけであり，ネイティブコードや HTML + CSS + JavaScript の組み合わせで動作するマルウェア，脆弱性を攻撃するコードなどを検出することはできない．またバイトコードに原因があっても新種やマルウェアのコードが大きく変化した場合には，我々が新たに定義ファイルを追加しなければ提案した方法では検出できない．

ゆえに，119,153 種類の中には我々が見逃しているマルウェア検体があると考えられる．マルウェアを検出するシステムを構築するためには，提案した方法に別の方法を組み合わせて多角的に解析する必要がある．本研究の目的は，我々が見つけることができなかったマルウェアを検出するための別の方法を検証することである．

2.2 本研究の提案

マルウェアを配布する場合には，Google Play をはじめとする通常のアプリを配布するためのサイトで配布されること，もしくはマルウェア作者が用意した通常のアプリを配布サイトを装う Web サイトで配布されることが多く見られる．前者の場合，数多くのアプリの中にマルウェアが存在することになり，管理者が自身のサイトのマルウェアの存在に気がつけばマルウェアは削除されることになる．サイトの管理状況によって，そのサイトでの通常のアプリの数とマルウェアの数の比率は変化するが，世の中全体の比率と大きく乖離することはないと思われる．一方，後者の場合は他に配布しているアプリもマルウェアであると推測でき，サイトが消滅するまでマルウェアの配布は続くことになる．

我々はアプリが配布されていたサイトのドメインまたは IP アドレスを収集し，マルウェアの比率が高いドメインまたは IP アドレスを抽出することで，そのサイトにあるアプリはマルウェアであると推定する方法を提案する．

3. 関連研究

3.1 セカンドアプリ

Android のアプリのファイルの拡張子は.apk であるが，実体は ZIP 形式のアーカイブであり，このアーカイブの中に Dalvik VM の実行形式である classes.dex が含まれている．マルウェア作者はアーカイブに別の悪意のある Android のアプリおよび root 権限を取得するための攻撃コードを追加し，このアプリをインストールして実行するためのコードを classes.dex に追加することができる．この場合には追加した別の悪意のある Android のアプリを変えることで，異なる動作をするマルウェアを作成することができる．

磯原らはこの追加される悪意のある Android のアプリをセカンドアプリと定義し，通常の Android のアプリが別の Android のアプリを含むことはないので，セカンドアプリが存在するときにはそのアプリをマルウェアとして検出する方法 [6] を提案した．磯原らの方法では，アプリ内のファイルが ZIP 形式のアーカイブであり，そのアーカイブが classes.dex と AndroidManifest.xml を含むならば，セカンドアプリ内包型 Android マルウェアとして検出する．

3.2 署名情報

Android のアプリは署名されており，この署名はアプリの作者が同一であることを確認するために用いられている．

西田らはマルウェアであっても同じ証明書が使いまわされていることに着目し，過去に配布されたマルウェアと同じ証明書で署名されているならばマルウェアであると推定する方法 [7] を提案した．

Android のアプリの中の META-INF ディレクトリの公開鍵証明書ファイル（名前は CERT.RSA であることが多い）から署名に用いた証明書データを抽出することができる．西田らの方法では，マルウェアから抽出した証明書データとアプリから抽出した証明書データを比較して，一致したならばそのアプリをマルウェアとして検出する．

4. 実験

文献 [2] で収集した 121,131 種類のアプリを本研究の実験の対象とする．これらは初期の 13 種類を除いて 2011 年 6 月から 2013 年 3 月までの間に我々が Web サイトを巡回して取得したアプリである．同文献ではこのうち 1,978 種

表 1　科名ごとのマルウェア検体数と割合
Table 1　Number of malware per family name

	Family name	Number	Rate
1	Hamob	828	41.86%
2	Kmin	281	14.2%
3	Plangton	275	13.9%
4	Agent	87	4.4%
5	Loicdos	70	3.54%
6	GinMaster	62	3.13%
7	FakeInst	55	2.78%
8	Opfake	43	2.17%
9	KungFu	31	1.57%
10	Qdplugin	19	0.96%
	Other	227	11.48%

類のアプリを Black（マルウェア）とした．これらはマルウェアの名称では 163 種類，科名（Family name）では 65 種類になる．検体数が多い科名の 10 種類を**表 1** に示す．

4.1　ドメイン

我々はアプリを配布しているサイトのドメインについて調査した．サイトの事実上の管理者が同一であっても，バーチャルドメインなどを用いている場合にはホスト名は異なる．単純にホスト名で比較すると，ホスト名が違うだけの同一サイトを別のサイトとして判定してしまう．そこで本研究では Public Suffix List[*1] を用いて，ホスト名が異なっていても管理者が同じならば同一のサイトとみなす．たとえば「www.example.jp」と「download.example.jp」はともに「example.jp」として扱う．また IP アドレスだけでサイトが運営されているときには，その IP アドレスをドメインとみなす．

1,978 種類のマルウェアは 651 種類の異なるドメイン上のサイトで配布されていた．**表 2** は 10 種類以上のマルウェアを配布していたサイトと同一のドメインにあったマルウェアとして検出していないアプリの数，全体に占めるマルウェアの比率である．ドメインは実際の名前ではなく仮の文字列で表記している．また表 2 のうちマルウェアの比率が高い 5 つのドメインについて，最初に我々が検体を入手した月，最後に検体を入手した月，検体の名前と数を**表 3** にまとめた．

4.2　セカンドアプリ

表 4 は文献 [6] と同様に，APK 形式のアプリを展開して内部に ZIP 形式のアーカイブがある場合には再帰的にアーカイブを展開し，アーカイブが classes.dex と AndroidManifest.xml を含むならばそのアプリをセカンドアプリ内包型 Android マルウェアとして検出した結果である．表 4 の行は内包されていたアプリの拡張子が apk である場合と，拡張子が偽装されている場合，複数のアプリが内包されており拡張子が apk と偽装されている拡張子の両方がある場合の 3 通りに分かれている．

[*1] http://publicsuffix.org/

表 2　ドメインごとのマルウェア検体数とその他のアプリの数
Table 2　Number of malware and other per domain

Domain	Malware	Other	Rate
M00	410	68	85.77%
M01	280	12	95.89%
B02	91	5,044	1.77%
M03	73		100%
B04	51	961	5.04%
B05	44	1,137	3.73%
B06	32	3,917	0.81%
B07	26	5,258	0.49%
B08	26	1,226	2.08%
M09	23		100%
B10	22	2,740	0.8%
B11	20	1,033	1.9%
B12	19	3,343	0.57%
B13	19	1,074	1.74%
B14	18	132	12%
B15	16	812	1.93%
B16	15	907	1.63%
B17	14	188	6.93%
B18	13	3,006	0.43%
B19	12	3,321	0.36%
M20	11		100%
B21	10	145	6.45%

表 3　マルウェア配布ドメインの詳細
Table 3　Detail of malware distributing domain

Domain	Malware	
Date	Name	Number
M00	Hamob.a	30
2012/2 - 2013/3	Hamob.b	6
	Hamob.c	29
	Hamob.d	32
	Hamob.e	287
	Loicdos.a	26
M01	Kmin.b	280
2011/11 - 2012/4		
M03	Plangton.c	71
2012/8 - 2013/1	Plangton.e	2
M09	FakeInst.ed	23
2012/2		
M20	FakeInst.ed	11
2012/2		

4.3　署名情報

Android のアプリは APK 内部の META-INF ディレクトリに証明書に関するファイルがある．本研究ではこのディレクトリにある CERT.RSA，またはそれが存在しな

表 4　セカンドアプリを内包するアプリの数
Table 4　Number of including application

Extension	Malware	Other	Rate
apk	14	1,855	0.75%
Mismatch	130	241	35.04%
Both	5	60	7.69%
Total	149	2,156	6.46%

表 5　除外したアプリの数
Table 5　Number of excluded application

	Malware	Other	Total
Extraction Error	13	353	366
Verification Error	9	3,010	3,019
media.x509.pem		117	117
platform.x509.pem	1	634	635
shared.x509.pem		389	389
testkey.x509.pem	34	6,092	6,126
Total	57	10,595	10,652

いならば拡張子が RSA または DSA の最初に見つかったファイルを PKCS7 形式の署名情報を含むファイルとみなす．準備したアプリのうち，このファイルが存在するアプリは下記のコマンドで署名情報を出力する．出力された情報のうち「`-----BEGIN CERTIFICATE-----`」と「`-----END CERTIFICATE-----`」で囲まれたテキストを比較の対象とする．

- openssl pkcs7 -inform DER -print_certs -in META-INF/CERT.RSA

我々は署名情報を抽出できないアプリと証明書の検証に失敗したアプリ，文献 [7] と同様に開発者用証明書で署名されたアプリを除外した．**表 5** は除外したアプリの数である．署名情報を抽出できなかったマルウェア検体が 13 種類あるが，これらは classes.dex だけしか入手できなかった文献 [2] の初期の 13 種類の検体である．

表 5 の 57 種類のマルウェア検体を除く 1,921 種類のマルウェア検体から署名情報を抽出したところ，証明書は 281 種類に集約された．マルウェア検体の中で 1 つの検体にしか証明書が使われていないのは 183 種類あり，残り 98 種類は複数回使われていた．同様に表 5 の 10,595 種類のアプリを除く 108,558 種類のマルウェアとして検出していないアプリから署名情報を抽出したところ，証明書は 28,407 種類に集約された．マルウェアとして検出していないアプリの中で 1 つの検体にしか証明書が使われていないのは 17,983 種類あり，残り 10,424 種類は複数回使われていた．証明書の使用回数の内訳を**表 6** に示す．

281 種類の証明書のうち 1 つのマルウェアでしか署名に使用されていない 183 種類を除いた 98 種類の証明書が使用されていた期間を**表 7** に示す．文献 [7] では APK ファイルの中にある AndroidManifest.xml の日時を検体の作成日時とし，最も古い作成日時と最も新しい作成日時の差を証明書の使用期間としている．しかし本研究では我々が初めてその検体を入手した日時を検体の入手日時とし，最も古い入手日時と最も新しい入手日時の差を証明書の使用期間としている．

表 8　証明書の重複率
Table 8　Overlap rate

	Number	Overlap	Rate
Certificate	28,407	78	0.27%
Application	108,558	5,230	4.82%

表 5 の 10,595 種類のアプリを除く 108,558 種類のアプリの中で，マルウェアとして検出していないアプリから抽出した証明書の情報は 28,407 種類あった．これらのうちマルウェアで使われた証明書と重複していた証明書は 78 種類あり，それらの証明書で署名されたマルウェア検体の数は 1,134 種類，マルウェアとして検出していないアプリの数は 5,230 種類であった．証明書の重複率を**表 8** に示す．

5. 再検査

4.1 項ではドメイン M00 で 68 種類，M01 で 12 種類が文献 [2] でマルウェアとして検出していないアプリであった．4.2 項の表 4 ではセカンドアプリを持っている 2,156 種類のアプリが文献 [2] でマルウェアとして検出していないアプリであった．またセカンドアプリとして APK に内包されていたアプリは 779 種類あった．4.3 項では文献 [2] で 5,230 種類のマルウェアとして検出していないアプリがマルウェアで使われた証明書で署名されていた．これらは重複しているアプリがあるので全部で 7,988 種類になる．これらのアプリは文献 [2] では検出していないものの，バイトコードによらない解析によりマルウェアであることが疑われるアプリということができる．

表 9 はこの 7,988 種類の疑わしいアプリと，文献 [2] でマルウェアとして検出していない 119,153 種類のアプリから 7,988 種類のアプリを除いた 111,165 種類のアプリからランダムに 10,000 種類を抽出して G Data，Kaspersky，Symantec の Windows 版アンチウイルス製品で再度検査した結果である．いづれか 1 つのアンチウイルス製品で検出されたアプリをマルウェアとみなすと，疑わしいアプリの中からは新たに 545 種類のアプリが，またマルウェアとして検出していないアプリ 10,000 種類の中からは 50 種類のアプリがマルウェアとして検出されたことになる．なお表 9 は 2013 年 6 月 5 日から 2013 年 6 月 11 日時点での結果である．

次節より疑わしいアプリ 7,988 種類をアンチウイルス製品で検査した結果の検証を行う．

表 6　証明書の使用回数
Table 6　Reused certificates

	Malware			Other		
	Certificate	Sample	Rate	Certificate	Sample	Rate
$n = 1$	183	183	9.53%	17,983	17,983	16.57%
$2 \leq n \leq 10$	85	284	14.78%	8,981	31,655	29.16%
$11 \leq n \leq 50$	10	207	10.77%	1,211	24,795	22.84%
$51 \leq n \leq 100$	1	71	3.7%	150	10,510	9.68%
$101 \leq n \leq 500$	1	280	14.58%	73	12,709	11.71%
$501 \leq n \leq 1,000$	1	896	46.64%	3	2,151	1.98%
$1,001 \leq n$				6	8,755	8.06%

表 7　証明書の使用期間
Table 7　Duration certificates

Day	Malware				Other			
	Certificate	Rate	Sample	Rate	Certificate	Rate	Sample	Rate
$d < 1$	21	21.43%	115	6.62%	1,494	14.33%	6,017	6.64%
$1 \leq d < 30$	5	5.10%	32	1.84%	329	3.16%	1,391	1.54%
$30 \leq d < 60$	7	7.14%	89	5.12%	407	3.9%	1,838	2.03%
$60 \leq d < 90$	1	1.02%	2	0.11%	564	5.41%	2,542	2.81%
$90 \leq d < 180$	16	16.33%	321	18.47%	1,549	14.86%	7,519	8.3%
$180 \leq d < 360$	17	17.35%	130	7.48%	2,464	23.64%	13,712	15.14%
$360 \leq d$	31	31.63%	1,049	60.36%	3,617	34.7%	57,556	63.54%

表 9　アンチウイルス製品の検出結果
Table 9　Result of antivirus products

		Suspicious	Other
Checked		7,988	10,000
Detected	G Data	520	42
	Kaspersky	179	23
	Symantec	45	6
Malware		545	50
Rate		6.82%	0.5%

表 10　セカンドアプリを内包するアプリの再検査結果
Table 10　Result of including application

Extension	Checked	Detected	Malware + Detected	Rate
apk	1,855	164	178	9.52%
Mismatch	241	66	196	52.83%
Both	60	4	9	13.85%
Total	2,156	234	383	16.62%

表 11　セカンドアプリとして内包されるアプリの再検査結果
Table 11　Result of included application

Extension	Number	Detected	Rate
apk	530	12	2.26%
Mismatch	244	14	5.73%
Both	5	1	20%
Total	779	27	3.47%

5.1　ドメイン

ドメイン M00 の 68 種類のうち新たにマルウェアとみなされたアプリは 1 種類，ドメイン M01 の 12 種類のうち新たにマルウェアとみなされたアプリは 2 種類であった．ドメイン M01 の 2 種類中 1 種類は拡張子が偽装されたセカンドアプリを内包している．

5.2　セカンドアプリ

表 10 はセカンドアプリを内包するアプリのうちマルウェアとして検出していないアプリを再検査した結果である．表 10 の Checked は表 4 の Other に対応する．

表 11 はセカンドアプリとして内包されているアプリを拡張子別に再検査した結果である．

5.3　署名情報

78 種類の証明書のうちアプリの数が多い 10 種類について**表 12** に示す．表 12 の検出率に関するデータは証明書が使われていたアプリの数，そのうち文献 [2] でマルウェアとした数，アンチウイルス製品が検出した数，新たにマルウェアとしたアプリも含めてマルウェアの全体に占める比率である．それらに加えてその証明書のアプリが配布されていたドメインの数，最も多くのアプリを配布していたドメインでのアプリの数，文献 [2] で我々が作成した定義ファイルの名前を示す．

また 78 種類の証明書のうちアンチウイルス製品で検出されたアプリをマルウェアとみなしたときに，署名されているすべてのアプリがマルウェアとなった証明書は**表 13** の 13 種類である．表 13 の最上位の証明書は表 12 では第

表 12　重複して使用されていた証明書（上位 10 種類）
Table 12　Certificates used by malware and others (Top 10)

	Sample	Malware	Detected	Rate	Domain	1st domain	Name
1	1,506	4	10	0.93%	576	121	AnSmCon.a Kiser.a Kiser.f Twofor.a
2	1,157	9	2	0.95%	8	1,131	YcChar.a
3	1,143	896	5	78.83%	585	478	Hamob.a Hamob.b Hamob.c Hamob.d Hamob.e Loicdos.a
4	286	1		0.35%	44	76	MTracker.a
5	**284**	**1**	**283**	**100%**	**7**	**147**	**Nandrobox.c**
6	234	2		0.86%	118	36	SMSreg.b
7	209	2		0.96%	125	24	Kiser.h Plangton.a
8	183	1		0.55%	148	25	FaceNiff.a
9	163	18		11.04%	19	150	Plangton.a
10	137	1		0.73%	11	110	Qdplugin.a

表 13　すべてのアプリがマルウェアになった証明書
Table 13　Certificates of malware after reanalyzing

Malware	Detected	Name
1	**283**	**Nandrobox.c**
9	4	Boxer.a SMSreg.bs SMSreg.o
10	2	Agent.a
3	3	Anti.b Anti.c
1	3	Biige.a
1	2	AccuTrack.a
1	1	Bosm.b
1	1	Hispo.c
1	1	KungFu.aj
1	1	Nyleaker.a
1	1	SMSreg.bw
1	1	SpyPhone.a
1	1	TigerBot.a

5 位の証明書である．

6. 考察

6.1 ドメイン

実験の結果，表 2 のドメイン M01 で若干のマルウェアとみなせなかったアプリがあったものの，我々が提案したマルウェアの比率が高いドメインにあるアプリはマルウェアである確率が高いという予想は正しかった．しかしドメインに関する実験の結果だけを見ると，表 2 のドメイン M00 では再検査後も 67 種類がマルウェアではなかった．この 67 種類はすべて表 12 の第 3 位の証明書で署名されている．これについては署名情報の考察とあわせて後で述べる．

6.2 セカンドアプリ

セカンドアプリの検証では再検査後の表 10 で示す結果を見ても，セカンドアプリをもつアプリがすべてマルウェアとは言えなかった．「android asset apk install」というようなキーワードの組み合わせで検索すると，セカンドアプリをインストールする方法を説明した Web サイトを見つけることができる．このことからアプリが内包するアプリをインストールする方法はマルウェアに固有の方法ではないと思われる．

またファイル名に alipay という文字列を含むセカンドアプリを持つアプリが 800 種類あり，800 種類のセカンドアプリには重複があるのでセカンドアプリは 21 種類に集約される．この名前に alipay を含む 21 種類のアプリは中国のオンライン決済システムのアプリであった．

以上のことから，文献 [6] で提案されているようにセカンドアプリが存在することを以ってマルウェアであるとは言えない．また表 11 よりセカンドアプリ自体がマルウェアであることは少なかった．しかし表 10 よりセカンドアプリの拡張子が偽装されているときにはマルウェアである確率が高いと言えるので，セカンドアプリに注目してマルウェアを検出するための有益な情報を得ることは可能である．

6.3 署名情報

署名情報では表 6 の証明書の重複や表 7 の証明書の使用期間では文献 [7] と大きな違いはなかった．しかし表 8 ではアプリでの重複率が文献 [7] に比べて高くなっており，1,921 種類のマルウェア検体のうち 1,134 種類でマルウェアとして検出していないアプリと証明書が重複していた．これをそのまま解釈すれば，文献 [7] で提案されているような署名情報による推定は成り立たなくなる．

この原因は主に表 12 の第 3 位の証明書にある．第 3 位の証明書で署名されたアプリは 585 種類の異なるドメインで配布されているが，最も多くの 478 種類のアプリを配布しているサイトは表 2 のドメイン M00 であった．このドメイン M00 はユーザから提供された Web コンテンツに広告を表示するコードを付加して Android のアプリを作成して再配布することでユーザに利益を提供するサービスを行っている．そのためアプリの作者が異なっていても，アプリはこのサイトの共通の証明書で署名される．表 9 のア

表 14　修正した証明書の重複率
Table 14　Revised overlap rate

	Number	Overlap	Rate
Certificate	28,391	62	0.22%
Application	106,505	3,177	2.98%

ンチウイルス製品では 247 種類中 5 種類だけをマルウェアとみなした．表 9 のアンチウイルス製品での検出数に大きな違いがあるように，何をマルウェアとみなすかの基準は異なる．我々はこの付加されるコードをアドウェアであると考えている．文献 [2] でマルウェアとして検出していない 247 種類もマルウェアとみなせば，あるいはマルウェアとして検出した 896 種類をマルウェアではないとすれば，証明書の重複は 1,134 種類から 238 種類に減少する．

表 12 の第 1 位の証明書は多くのドメインで配布されており，特定のドメインに偏ってはいなかった．また我々の定義ファイルの名前も異なっており，再検査の結果で新たにマルウェアとみなしたアプリの数も少ない．アプリの内容も一貫性がなく，またその数から単一の開発者によって作成されているとは考えられない．このことより第 1 位の証明書は特殊な用途に用いられていると推測して我々が調査したところ，この証明書は Android License Verification Library（LVL）をクラックするツールに含まれている証明書であることを確認した．

このような特殊な用途の証明書は表 5 の証明書と同様に除外すべきである．またはこのような用途の場合，この証明書で署名されたアプリに対しては何らかの警告を与えるのが妥当かもしれない．

表 12 の第 2 位の証明書も同様の状況であるが，これらの多くは表 2 のドメイン B07 で配布されていた．何らかの特殊な事情があるとは推測できるが，我々はその原因を特定できなかった．この証明書も除外すべき証明書かもしれない．

第 5 位の証明書では当初は 1 つのアプリのみをマルウェアとして検出していたが，再検査の結果ですべてがマルウェアであるとみなしたことで解決した．

表 12 の第 1 位の証明書で署名されたアプリを除外し，第 3 位と第 5 位の証明書で署名されたアプリ，表 13 の証明書で署名されたアプリをマルウェアとみなした場合，重複率は**表 14** になる．表 8 に比べて重複率は下がったものの文献 [7] に比べると依然として高い．さらに特殊な用途の証明書を特定できれば，重複率は下がると思われる．

以上のことから，文献 [7] で提案されているように署名情報からマルウェアを推定することは可能である．しかし文献 [7] では言及されていなかった特殊な用途の証明書があることがわかった．文献 [7] の手法を用いるならば，特殊な用途の証明書を考慮する必要がある．

7. 今後の課題

本研究の実験でドメイン，セカンドアプリ，署名情報からマルウェアである可能性が高いアプリを絞り込むことは可能であった．ドメインに関する情報はアプリの内部にある文字列と照合して怪しい動作をすると思われるアプリを見つけ出すことや，単純にそのドメインへのアクセスをフィルタリングするなどに応用できる．また署名情報では LVL のクラックツールが使用している証明書を見つけるなど，文献 [7] では言及されていない特殊な証明書の存在が明らかになった．署名情報を検証することでこのような事例の検出を行うためには，より多くの署名情報を収集する必要がある．特にコードやデータの解析でクラックされたアプリか否かを判定することは難しいので署名情報は有用である．

ドメイン単独では表 3 のドメインの数が少なかったため情報を十分に活用することはできなかったが，証明書の利用状況を推測するためにドメインを参照することでドメインの情報を活かせた．また表 3 ではマルウェアを配布する Web サイトが数ヶ月にわたって存続していることもわかったので，悪意のあるサイトを見つけた場合には継続して監視する必要がある．一方で，セカンドアプリではドメインや証明書などの情報を参照してより多くの情報を引き出すような実験・考察を行うことができなかった．セカンドアプリと他の情報との連携も今後の課題となる．

いづれの方法もコードを解析する方法に比べて容易に実装が可能でありアプリの検査にかかる時間も短い．しかしマルウェアの作者もこれらの方法に対抗することは容易である．今のところはマルウェアの作者が対抗策をとっていないだけであり，いづれは使えなくなる可能性はあるので，これらの方法が有効に活用できるか検証し続ける必要がある．

文献 [2] でマルウェアとして検出しておらずかつ本研究の検証の対象とならなかったアプリ 10,000 種類のうち 50 種類は表 9 のアンチウイルス製品での再検査の結果ではマルウェアであった．このことから全体では約 555 種類のアプリがまだ検出できていないと推測できるため，これらを検出できる方法を検討する必要がある．しかし再検査にアンチウイルス製品を用いたが，マルウェアを検出する基準はそれぞれ異なっており我々の基準とも異なるので，アンチウイルス製品の検出結果は参考にとどめて再度我々が解析する必要がある．事実，これまでにアンチウイルス製品では検出されるが，我々はマルウェアであると判断しなかったアプリもある．

また我々が提案したマルウェア検出方法 [2] と本研究の検証結果を組み合わせることで，マルウェアを検出するシステムを構築したいと考えている．

参考文献

[1] Chen, X., Dirro, T., Greve, P., Li, H., Paget, F., Schmugar, C., Shah, J., Sherstobitoff, R., Sommer, D., Sun, B. and Wosotowsky, A.: McAfee Threats Report: First Quarter 2013, McAfee Labs (online), available from ⟨http://www.mcafee.com/us/resources/reports/rp-quarterly-threat-q1-2013.pdf⟩ (accessed 2013-06-11).

[2] 岩本一樹，和﨑克己：制御フロー解析により生成されたグラフ比較による Android マルウェア検出方法の提案，研究報告コンピュータセキュリティ（CSEC） 5 (2013).

[3] Shabtai, A., Kanonov, U., Elovici, Y., Glezer, C. and Weiss, Y.: Andromaly": a behavioral malware detection framework for android devices, *Journal of Intelligent Information Systems*, Vol. 38, No. 1, pp. 161–190（オンライン），入手先 ⟨http://www.springerlink.com/index/10.1007/s10844-010-0148-x⟩ (2012).

[4] Burguera, I., Zurutuza, U. and Nadjm-Tehrani, S.: Crowdroid: behavior-based malware detection system for Android, *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*, SPSM '11, New York, NY, USA, ACM, pp. 15–26 (online), DOI: 10.1145/2046614.2046619 (2011).

[5] 澤谷雪子, 川端秀明, 磯原隆将, 竹森敬祐, 窪田　歩：Android マルウェアの挙動に基づく検知ルール自動化生成手法，暗号と情報セキュリティシンポジウム（SCIS 2012），pp. 1–7 (2012).

[6] 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 論文集，Vol. 2011, No. 3, pp. 708–713 (2011).

[7] 西田雅太，神薗雅紀，星澤裕二：署名情報を利用した Android マルウェアの推定手法の提案，コンピュータセキュリティシンポジウム 2012 論文集，Vol. 2012, No. 3, pp. 28–35 (2012).