情報処理学会論文誌 Vol.56 No.9 1857–1867 (Sep. 2015)

推薦論文

TaintDroidを用いた利用者情報送信の動的制御手法

小倉 禎幸[1] 山内 利宏[1,a)]

受付日 2014年12月5日, 採録日 2015年6月5日

概要：近年，Android 端末の普及にともない，Android を標的とするマルウェアが増加し，マルウェアへの対策が重要視されている．特に，端末外部へのマルウェアによる利用者情報の漏洩が問題となっている．そこで，この問題に対処するために，TaintDroid を利用し，端末外部への利用者情報の漏洩を防止する手法を実現する．具体的には，本手法は，TaintDroid を利用することで端末内部の利用者情報の伝搬を追跡し，利用者情報が端末外部へ漏洩する場合，利用者の判断に従って AP の動作を動的に制御することで，端末外部への利用者情報の漏洩を防止する．また，AP 間で利用者情報のやり取りがあった場合，利用者情報の漏洩に関わった AP 名を取得し，利用者情報の伝搬経路を把握する．これにより，漏洩要因の各 AP に対処できる．さらに，端末外部に送信される利用者情報をダミーデータに置換し，AP の正常な動作をできるだけ妨げることなく利用者情報の漏洩を防止できる．

キーワード：Android，テイント解析，情報伝搬追跡，情報漏洩の防止

Dynamic Control Method for Sending User Information Using TaintDroid

Yoshiyuki Ogura[1] Toshihiro Yamauchi[1,a)]

Received: December 5, 2014, Accepted: June 5, 2015

Abstract: In recent years, Android malware has been increasing, and countermeasures against them have become an issue. In particular, the leakage of user information by malware has become an important issue. In order to address this problem, we design and implement a method that uses TaintDroid to prevent the leakage of user information from Android device. This method tracks the diffusion of user information in a device and dynamically controls the action of application program (AP) when the leakage of user information is detected. As a result, this method prevents the leakage of user information from the device. In addition, this method obtains the AP name involved in the leakage of user information and understands the diffusion path of user information when APs communicate user information with each other. Therefore, a user can deal with each AP of leakage factors. Furthermore, this method replaces user information that is leaked from a device with a dummy data. As a result, this method prevents the leakage of user information from the device without interfering the process of AP.

Keywords: Android, taint analysis, information diffusion tracking, prevention of information leakage

1. はじめに

近年，Android [1] 端末が普及し，Google Play [2] などのマーケットにおいて多種多様なアプリケーションプログラム（Application Program：以降，AP と略す）が配布され，利用されている．

しかし，配布されている AP の中には，システムの脆弱性を攻撃して管理者権限を不正に取得するマルウェア，ネットワークや SMS 経由で外部サーバと通信して料金を発生させるマルウェア，および利用者情報を外部に漏洩するマルウェアなどが発見され [3]，被害を及ぼしている．特

[1] 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University, Okayama 700–8530, Japan
[a)] yamauchi@cs.okayama-u.ac.jp

本論文の内容は 2014 年 10 月のコンピュータセキュリティシンポジウム 2014 にて報告され，同プログラム委員長により情報処理学会論文誌ジャーナルへの掲載が推薦された論文である．

に，利用者情報を外部に漏洩するマルウェアによる被害が問題となっている [4], [5]．このため，端末外部に利用者情報を漏洩させるマルウェアやマルウェア以外にも悪意はないが不要な利用者情報を取得，送信するような AP への対策が重要視されており，この問題に対処するために，利用者情報の漏洩防止に関する様々な手法が研究されている [6], [7], [8], [9], [10], [11]．しかし，これらの手法には，追跡粒度が粗く誤検知が多い，利用者情報の漏洩に関わった AP の情報を利用者が正確に把握不可，利用者情報の取得制限による AP の動作の妨害，および利用者の判断による AP の動作制御不可という問題のいずれかがある．

そこで，本論文では，これらの問題をすべて解決するため，TaintDroid [9] を用いた細粒度の情報追跡による利用者情報送信の動的制御手法の設計，実現方式，および評価について述べる．本手法は，TaintDroid を利用することで端末内部の利用者情報の伝搬を追跡し，利用者情報が端末外部へ漏洩する場合，利用者の判断に従って AP の動作を動的に制御することで，端末外部への利用者情報の漏洩を防止する．また，AP 間で利用者情報のやり取りがあった場合，利用者情報の漏洩に関わった AP 名を取得し，利用者情報の伝搬経路を把握する．利用者情報の漏洩の可能性がある場合に，把握した利用者情報の伝搬経路を利用者に提供することで，利用者は，利用者情報の漏洩に関わったすべて AP の情報を正確に把握し，漏洩要因の各 AP を特定し，アンインストールなどの対処を実施できる．さらに，本手法では，端末内部での AP による利用者情報の取得を制限せず，利用者情報が端末外部に送信される場合にのみ，端末外部に送信される利用者情報をダミーデータに置換する．これにより，AP の正常な動作をできるだけ妨げることなく利用者情報の漏洩を防止できる．

なお，本論文において，利用者情報は Android のパーミッションを利用して取得できる情報のうち，端末や利用者に関連する情報のことを指す（4.1 節で後述する）．

2. Android における利用者情報の漏洩防止手法

2.1 API に着目した手法

文献 [6], [7] の手法は，API をフックし，利用者が API の利用の可否決定をすることで利用者情報の漏洩を防止している．文献 [6] の手法は，利用者情報にアクセスする API と利用者情報を外部に漏洩する API をフックし，利用者が API の利用の可否決定をすることで利用者情報の漏洩を防止する．また，監視している AP がインテントにより別の AP と通信した場合，通信先の AP も監視する．

しかし，文献 [6], [7] の手法は，API により利用者情報が伝搬しない場合でも追跡漏れを防ぐために伝搬したものとして追跡するため，追跡粒度が粗く，誤検知が発生しやすいという問題がある．また，利用者に提示される情報の内容は AP のパッケージ名や AP が利用する API 名であり，一般の利用者では表示された内容を理解しにくいという問題がある．

2.2 ダミーデータを用いた手法

MockDroid [8] は，AP が利用者情報の代わりにダミーデータを取得するように事前に設定する．これにより，利用者情報へのアクセスを許可していない AP による利用者情報の取得を防止し，利用者情報の漏洩を防止する．しかし，AP が利用者情報を取得する際にダミーデータを取得するため，利用者情報を取得できないことにより AP が強制終了するなどの異常な動作が発生し，AP の動作を妨害する問題がある．

2.3 TaintDroid を用いた手法

TaintDroid は，利用者情報に Taint と呼ばれるタグ（以降，Taint タグと略す）を付与し，その利用者情報が使用された場合に，利用者情報の伝搬を追跡する．Taint タグは，Taint タグが付与されている変数のコピーや配列への挿入などが行われた際に伝搬する．また，TaintDroid は，インターネット送信や外部ストレージへの書き込みなどを監視しており，端末外部に利用者情報が漏洩した場合に，利用者情報に付与された Taint タグ，端末外部に伝搬させた AP のパッケージ名，および利用者情報の伝搬先をログとして記録し，利用者に提示する．利用者は，提示されたログから利用者情報を端末外部に伝搬させた AP がマルウェアか否かを判断する．

しかし，TaintDroid は，利用者情報の伝搬を追跡することを目的としており，利用者情報の漏洩を防止しない．また，端末外部にデータを送信した AP のパッケージ名しかログに記録しない．このため，2 つ以上の AP が連携して利用者情報を漏洩させるような場合，たとえば，利用者情報を取得する AP と外部と通信する AP において，利用者は，外部と通信する AP の情報しか把握することができず，利用者情報の取得に関わった AP を正確に把握できない．

文献 [10], [11] の手法は，TaintDroid を用いて利用者情報の漏洩を防止している．文献 [10] の手法は，AP のインストール時に，AP が取得した情報を端末内でのみ使用するか端末外部への送信を許可するかを利用者が選択する．このように，利用者情報の使用範囲を制限することで，利用者情報の漏洩を防止する．しかし，インストール時以外には，利用者は利用者情報の使用範囲を変更できない．このため，利用者情報が漏洩する際にその動作の制御方法を利用者が動的に選択できない．また，インストール時に設定を行うため，実際に AP がどのようなタイミングでどのような情報を送信するのか利用者は判断できない．このため，利用者は AP に対して適切な判断ができない．

AppFence [11] は，Taint タグが付加されている利用者情

報が端末外部に送信される際，送信をブロックするかまたはダミーデータと入れ替えることで利用者情報の漏洩を防止する．しかし，利用者は AP ごとにダミーデータやポリシの設定をする必要がある．また，ポリシを用いて利用者情報の漏洩を防止しているため利用者情報が漏洩する際，その動作を利用者が動的に制御できない．さらに，端末上でポリシの設定を変更できず，一般の利用者への負担が大きい．

2.4 既存研究の問題点

これまでに述べた手法 [6], [7], [8], [9], [10], [11] には，以下のいずれかの問題が存在する．特に問題 2 については，手法 [6], [7], [8], [9], [10], [11] で，細粒度に AP 間の利用者情報のやり取りに関わった AP の情報を追跡し，その情報を把握しているものはない．

（問題 1）追跡粒度が粗く誤検知が多い
（問題 2）利用者情報の漏洩に関わった AP を利用者は正確に把握不可
（問題 3）利用者情報の取得制限による AP の動作の妨害
（問題 4）利用者の判断による AP の動作制御不可

本論文では，これらすべての問題点を解決する手法の設計，実現方式，および評価について述べる．

3. 設計

3.1 システムへの要求

本章では，本手法の設計について説明する．本手法は，AP による利用者の意図しない利用者情報の漏洩を防止することを目的としている．2 章で示した各問題に対処するために必要な本手法への要求は以下のようになる．

（要件 1）細粒度で利用者情報の伝搬を追跡できること
（要件 2）利用者が利用者情報の漏洩に関わった AP を正確に把握できること
（要件 3）AP が利用者情報を端末内部で使用する限り，AP の動作を妨害しないこと
（要件 4）利用者が AP が端末外部に利用者情報を送信する処理を動的に制御し，必要に応じて利用者情報の漏洩を防止できること

（要望 1）と（要望 2）は，利用者情報の漏洩防止の観点から要件とはならないが，重要な課題である．

（要望 1）利用者の負担を少なくすること
（要望 2）利用者の判断を支援すること

（要望 1）と（要望 2）は，利用者情報の漏洩防止の観点から要件とはならないが，重要な課題である．

3.2 本手法の全体像

本手法の全体像を**図 1** に示す．本手法は，一部に TaintDroid の機能を利用した以下の 5 つの機能からなる．

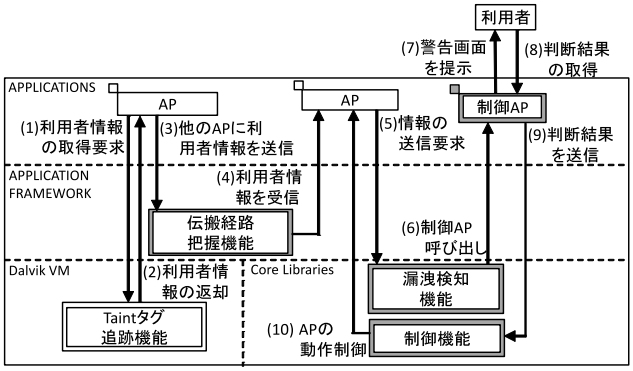

図 1 TaintDroid を用いた利用者情報送信の動的制御手法の全体図
Fig. 1 Overview of the proposed method.

(1) Taint タグ追跡機能（TaintDroid）
AP が利用者情報を取得した際，その取得した利用者情報の伝搬を追跡する．

(2) 伝搬経路把握機能
AP 間で利用者情報がやり取りされた場合，その伝搬経路を把握する．

(3) 漏洩検知機能
AP がデータを端末外部に送信する動作を検知し，送信されるデータに利用者情報が含まれているか否かを判断する．利用者情報が含まれていた場合は，制御 AP を呼び出す．

(4) 制御 AP
端末外部への利用者情報の送信を許可するか否かを端末の利用者に尋ね，利用者の判断結果を取得する．

(5) 制御機能
制御 AP において取得した利用者の判断結果に従って，AP が利用者情報を端末外部に送信する動作を制御する．

3.3 対処

3.3.1 （要件 1）への対処

利用者情報の漏洩を検知するために，AP 内の利用者情報の使用を追跡する必要がある．このため，本手法では，Taint タグ追跡機能（TaintDroid）を利用することで AP 内での利用者情報の使用を追跡する．TaintDroid を用いて利用者情報の伝搬を変数レベルで追跡することにより，API を用いた利用者情報の漏洩検知手法 [6], [7] と比べて追跡粒度の粗さによる誤検知の発生を削減できる．

3.3.2 （要件 2）への対処

複数の AP が連携して利用者情報を漏洩させる場合がある．このため，利用者情報を漏洩させた AP が，どういった経路で利用者情報を取得したのかを判断する必要がある．しかし，利用者情報の追跡に用いる TaintDroid では，利用者情報がどの AP を経由して伝搬したのかを把握できない．このため，本手法では，AP 間で利用者情報がやり取りされた際，伝搬経路把握機能において，通信先と通信

元の AP 名，やり取りされた利用者情報，および利用者情報の種類を取得することで，利用者情報の漏洩に関わった AP と利用者情報の伝搬経路を把握する．これにより，複数の AP で連携した利用者情報の漏洩 [12], [13] を防止できる．また，利用者は漏洩要因となった各 AP に対処できる．

3.3.3 （要件 3）への対処

MockDroid などの既存手法では，AP による利用者情報の取得を制限した場合，AP が正常に動作しなくなる場合がある．このため，本手法では，端末内での AP による利用者情報の取得を制限せず，利用者情報が端末外部に送信される場合にのみ，その利用者情報の使用を制限する．これにより，利用者情報を端末内部で使用する限り，AP の異常な動作を防止できる．また，利用者情報が端末外部に送信される際，その利用者情報の使用を制限する場合は，利用者情報をダミーデータに置換して送信する．これにより，外部への情報送信に関係する API を AP に成功として見せることで，できる限り AP が強制終了するなどの異常な動作をしないことを目指す．

3.3.4 （要件 4）への対処

AP による利用者情報の漏洩を防止するために，本手法では，漏洩検知機能において端末外部への利用者情報の送信を検知し，制御 AP を用いて利用者に警告画面を提示する．提示した警告画面において，AP の動作の可否を利用者に尋ね，利用者の判断結果を取得する．その後，制御機能において，利用者の判断結果に従って AP の動作を動的に制御する．これにより，利用者が AP の動作を動的に制御することができる．

3.3.5 （要望 1）と（要望 2）への対処

利用者が難しい操作なしに，利用者情報の漏洩を防止し，利用者が利用者情報の漏洩による危険性を理解する必要がある．このため，本手法では，利用者への警告において，次回以降の提示の有無を AP ごとに設定できるようにする．また，警告画面において一般の利用者が理解しやすい情報を提示する．

4. 実現方式

4.1 Taint タグ追跡機能（TaintDroid）

本機能は，TaintDroid により利用者情報の伝搬を細粒度で追跡する機能である．TaintDroid は，以下に示す変数や配列に Taint タグ（**表 1**）を付与している．本論文では，TAINT_CLEAR を除く Taint タグを付与されたデータを利用者情報とする．

(1) メソッドのローカル変数
(2) メソッド引数
(3) クラスの静的フィールド
(4) クラスのインスタンスフィールド
(5) 配列

Taint タグの付与は，具体的には，32 bit の変数に隣接して 32 bit の TaintTag を付与し，この 2 つの隣接した値を 1 つの 64 bit の値として解釈する．たとえば，SIMRecords.java 内の SIMRecord クラスの setMsisdnNumber メソッドが呼ばれた際，msisdn に TAINT_PHONE_NUMBER という Taint タグが付加される．このように，変数や配列に付与された Taint タグの伝搬を追跡することで，AP が取得した利用者情報を細粒度で追跡できる．

表 1 Taint タグの定義
Table 1 Definition of TaintTag.

Taint タグの種類	Taint タグの値
TAINT_CLEAR	0x00000000
TAINT_LOCATION	0x00000001
TAINT_CONTACTS	0x00000002
TAINT_MIC	0x00000004
TAINT_PHONE_NUMBER	0x00000008
TAINT_LOCATION_GPS	0x00000010
TAINT_LOCATION_NET	0x00000020
TAINT_LOCATION_LAST	0x00000040
TAINT_CAMERA	0x00000080
TAINT_ACCELEROMETER	0x00000100
TAINT_SMS	0x00000200
TAINT_IMEI	0x00000400
TAINT_IMSI	0x00000800
TAINT_ICCID	0x00001000
TAINT_DEVICE_SN	0x00002000
TAINT_ACCOUNT	0x00004000
TAINT_HISTORY	0x00008000

4.2 伝搬経路把握機能

本機能は，AP 間の利用者情報の伝搬経路を把握する機能であり，Taint タグを利用することで実現している．TaintDroid は，Taint タグを伝搬させることで AP が利用する利用者情報を追跡する．しかし，TaintDroid は，利用者情報がどの AP を介して伝搬したかの情報を保持しない．このため，複数の AP が連携し利用者情報を漏洩させた場合，利用者情報の漏洩に関わったすべての AP を特定することができない．

本機能では，インテント処理をフックすることで，AP 間で利用者情報がやり取りされた際に，利用者情報の送信元，送信先の AP のパッケージ名，やり取りされる値，および Taint タグを取得する．ただし，フックする箇所の実装によっては，オーバヘッドの増加や取得したい情報が取得できないなどの問題が発生する．そこで，本研究では，オーバヘッドの増加を抑制でき，必要な情報が取得できる ActivityStack.java 内にフック箇所を実装した．また，取得した Taint タグからどの種類の利用者情報がやり取りされた値に含まれているか判断できる．これらの取得した情報をログに出力することで記録する．これにより，ログに出力された情報から利用者情報の伝搬経路を把握できる．

また，利用者情報が端末外部に送信される際，記録している情報を利用して，利用者情報の漏洩に関わったすべてのAP のパッケージ名を取得できる．利用者情報の伝搬経路に関わる情報は，漏洩検知機能が利用者情報の漏洩を検知した際に，制御 AP に渡され，利用者に提示される．複数の AP で連携した利用者情報の漏洩を防止でき，利用者は漏洩要因となった各 AP に対処することができる．

なお，本機能では，漏洩要因となった各 AP を区別して把握することができる．しかし，各 AP に組み込まれた外部モジュールを AP 本体と区別することはできない．この問題に対処する場合は，インテント処理や外部への送信を行う際にスタックトレースなどを行いスタック内に保持されている情報から外部モジュールと AP 本体を区別するなどの手法 [14] が考えられる．

4.3 漏洩検知機能

本機能は，端末外部への利用者情報の送信を検知し，制御 AP を呼び出す機能であり，Taint タグを利用することで実現している．

AP による利用者情報の漏洩を防止するためには，AP が端末外部に情報を送信する動作を検知する必要がある．Android では，SSL 通信を行う際には OpenSSLSocket.java が，プロセス間通信の際には Posix.java が使用される．それぞれの通信処理を監視し，漏れなく端末外部へ情報を送信する処理を検知するために，OpenSSLSocket.java と Posix.java 内にフック箇所を実装することで，その動作の検知を実現した．

本機能では，フック箇所において，AP が端末外部への情報の送信を行う際，送信される値に付加されている Taint タグを取得する．たとえば，OpenSSLSocket.java 内では，端末外部に送信されるデータの書き込み処理を行うメソッド（write）が実行された際，その引数として渡された値を取得する．また，取得した値に付与されている Taint タグを取得する．その後，取得した Taint タグから送信を要求した情報に利用者情報が含まれているか否かを識別し，含まれていた場合，利用者情報の漏洩と判断し，取得した Taint タグから利用者情報の種類を特定する．

また，このとき，利用者情報を端末外部に送信する動作を行っていた AP のパッケージ名を取得する．取得したパッケージ名を用いて，伝搬経路把握機能が保持している情報から取得したパッケージ名に該当する情報を検索する．

該当するパッケージ名が存在しない場合は，利用者情報を端末外部に送信する動作を行った AP のみを利用者情報の漏洩に関わった AP とする．該当するパッケージ名が存在した場合，伝搬経路把握機能で保持している AP 間でやり取りされた際の利用者情報の送信元，送信先の AP のパッケージ名，やり取りされる値，および Taint タグを取得する．これにより，複数の AP が連携し利用者情報を漏洩させた場合でも利用者情報の漏洩に関わったすべての AP を特定できる．

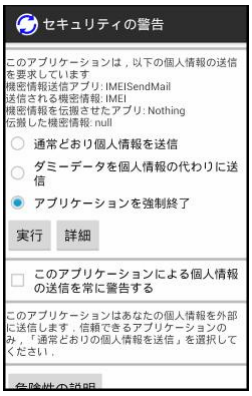

図 2 警告画面
Fig. 2 Warning screen.

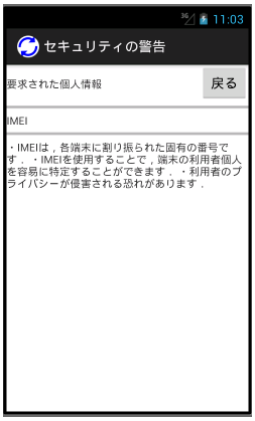

図 3 利用者情報の危険性の説明
Fig. 3 Explanation of the danger of user information.

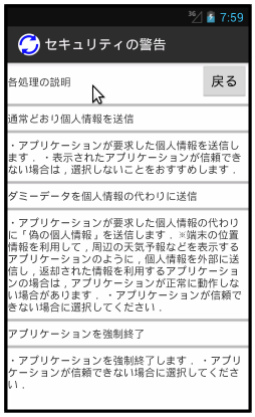

図 4 処理内容の詳細
Fig. 4 Details of the contents of processing.

さらに，フック箇所において，送信先（IP アドレス，ドメイン）や漏洩時の日時も取得することができる．取得したこれらの情報は，制御 AP を呼び出し，本機能と伝搬経路把握機能で保持している情報を制御 AP に送信する．

4.4 制御 AP

4.4.1 表示する情報

制御 AP は，利用者に警告画面を提示し，利用者の判断結果を取得する AP である．本 AP では，漏洩検知機能から送信された情報を利用者に分かりやすい形式に変更し，警告画面として利用者に提示する．警告画面の例を**図 2**，**図 3**，および**図 4** に示し，利用者の判断を支援するために利用者に提示する情報を以下に示す．

(1) 利用者情報を外部に送信する AP 名
(2) 利用者情報を取得した AP と利用者情報を外部に送信する AP が別の場合（複数の AP が連携している場合），すべての AP 名
(3) AP が送信する利用者情報の種類（電話番号など）

(4) APが取得した利用者情報の危険性の説明
(5) APに対する警告画面の表示の有無の選択項目
(6) APに対する処理内容の選択項目
(7) APに対する処理内容の詳細
(8) 送信先の提示（IPアドレス，ドメイン）
(9) 漏洩時の日時

利用者の判断を支援する情報について，既存研究では，利用者に提供する情報としてパッケージ名やUIDを提示する場合が多い．しかし，これらの情報では利用者は，APが利用者情報を端末外部に送信する動作を許可するか否かの判断が難しい．このため，本手法では，利用者の判断を支援する情報としてAP名と送信される利用者情報を提示する．また，提示する情報量が多い場合，利用者が判断しにくくなる．このため，表示する警告画面は図2のように1画面のみで警告の内容と処理選択ができるようにした．

さらに，利用者情報が端末外部に漏洩した場合の危険性を確認できることが利用者の判断を支援するうえで重要である．このため，図2の「危険性の説明」ボタンから，図3のようにAPが取得した利用者情報が漏洩することによる危険性を確認できるようにする．なお，利用者情報が漏洩することによる危険性は，これらの情報を利用者が必要だと判断した場合に提供できるように，画面遷移を用いて提供する．

次に，利用者の負担を減らす工夫について述べる．本手法では，利用者情報が端末外部に送信されようとするたびに利用者に許可を求める．しかし，許可を求める頻度が高くなると利用者の負担が増加する．このため，利用者に許可を求める頻度を抑制する必要がある．本手法では，図2のチェックボックスで表示されている項目のように，利用者がAPごとに警告の有無を選択できるようにする．これにより，警告画面の表示頻度を抑制し，利用者の負担を減らす．

さらに，利用者情報を漏洩させたAPに対する処理選択において，利用者が処理選択を判断しやすいことが重要である．このため，警告画面におけるAPに対する処理選択の際，図2のラジオボタンで表示されている項目のように，選択項目を3つに限定し，簡単な選択で処理内容を決定できるようにする．また，APに対する処理内容の詳細は，図2の「詳細」ボタンから図4のように確認できるようにする．なお，APに対する処理内容の詳細は，これらの情報を利用者が必要だと判断した際に提供できるように，画面遷移を用いて提供する．

このように，既存研究と異なり本手法では，既存手法では保持していないAP間のデータのやり取りや利用者情報の伝搬に関する細粒度の情報を利用することで，利用者の判断を支援することができる．

4.4.2 処理の流れ

制御APは，漏洩検知機能から送信された情報のうちAPのパッケージ名から対応するAP名を取得する．パッケージ名から対応するAP名を取得する処理の流れを以下に示す．

(1) 端末にインストールされているAPのパッケージ名とアプリ名を取得・保持
(2) 漏洩検知機能から送信されたAPのパッケージ名を取得
(3) (2)で取得したパッケージ名をキーとして(1)で取得した情報から該当するアプリ名を取得

上記の処理で取得したAP名と利用者情報の種類を警告画面に提示し，APが端末外部へ利用者情報を送信する動作を許可するか否かを利用者に尋ね，利用者の判断結果を取得する．その後，制御機能に取得した判断結果を送信する．

なお，AP名を取得する際に，対応するAPのアイコンも取得できる．APのアイコンを利用することで，利用者により分かりやすい警告画面の提示が可能だと考える．

4.5 制御機能

制御機能は，制御APで取得した利用者の判断結果に従って，APが利用者情報を端末外部に送信する動作を制御する機能である．本機能は，漏洩検知機能の箇所に実装している．APの処理の制御には，APの処理を制御しない場合（送信を許可）と，APを強制的に終了させる制御方法以外に，利用者の意図しない利用者情報の漏洩を防止し，かつAPを継続して利用できる制御方法（送信を拒否）が必要である．このため，本手法では，以下を3つの制御方法を利用者に提供する．

(1) 送信許可
　利用者情報の端末外部への送信を許可する場合に選択する．利用者情報は，APの動作により端末外部に送信される．
(2) ダミーデータの送信
　利用者情報の端末外部への送信を許可せず，APの動作を継続したい場合に選択する．利用者情報が端末外部に送信される際，送信される利用者情報の代わりに，対応するダミーデータを端末外部に送信する．これにより，APの動作を終了せず，利用者情報の漏洩を防止できる．
(3) 送信拒否
　利用者情報の端末外部への送信を許可しない場合に選択する．送信拒否は，APが端末外部に利用者情報を送信する動作を，例外の発生を通知するシグナルを発生させることで中断し，利用者情報の漏洩を防止する．

なお，ダミーデータは，以下の2つの方法のどちらかで置換する．1つ目の方法は，null文字列のデータに置換するものである．これは，端末外部に送信される利用者情報

表 2 ゲスト OS の評価環境
Table 2 Evaluation environment of guest OS.

ディストリビューション	Ubuntu 10.04.4 LTS
カーネル	Linux ubuntu 2.6.32-42-generic
仮想 CPU 数	4
メモリ	4 GB

表 3 Android Emulator の評価環境
Table 3 Evaluation environmental of Android Emulator.

Android	Android 4.1
TaintDroid	TaintDroid 4.1 (updated Dec 6, 2012)

が，カメラ，マイク，およびログの場合に使用する．これは，端末外部に送信される利用者情報がログなどの情報の場合，ログの中身をすべて，または一部置換する処理を行うと，置換処理にかかる時間が増加するためである．2 つ目の方法は，固定の値を返すものである．これは，端末外部に送信される利用者情報が，端末の位置情報，電話番号，および，IMEI（International Mobile Equipment Identifier）などの場合に使用する．

ただし，ダミーデータを送信することにより，AP が利用者の意図しない動作をする場合がある．たとえば，利用者の位置情報を取得し，その場所の天気予報を表示する AP が外部にダミーデータを送信したとする．このとき，AP は，利用者の正確な位置情報を取得できないため，利用者がいる場所の天気予報を表示できない．

4.6 期待される効果

本手法の実現により，以下の効果が期待できる．

(1) 誤検知の削減
(2) 利用者情報の漏洩に関わったすべての AP 名の取得
(3) AP の動作を妨げない利用者情報の漏洩の防止
(4) 利用者情報の漏洩の防止

5. 評価

5.1 評価項目

評価では，VM 上で Android Emulator を用いた．ゲスト OS と Android Emulator の評価環境を**表 2** と**表 3** に示す．以下に評価項目と評価の目的を示す．

（評価 1）誤検知の検証

利用者情報を取得と端末外部との通信を行う AP において，実際に取得した利用者情報を端末外部に送信する場合と，送信しない場合において本手法の動作を確認し，誤検知が削減できているか否か評価した．

（評価 2）利用者情報の漏洩防止

利用者情報を漏洩させる AP を用いて，AP による利用者情報の漏洩を検知し，利用者が AP による情報の送信を拒否することで，AP による不正な利用者情報の漏洩を防止できるか否か評価した．また，端末外部に送信される利用者情報をダミーデータに置換できるか否か評価した．

（評価 3）実アプリを用いた追跡精度の比較

実際のアプリを用いて，TaintDroid と本手法での追跡精度を比較し，評価した．また，本手法で利用者情報が端末外部に送信される際の検知タイミングと処理を評価した．

（評価 4）複数の AP が連携した場合の利用者情報の漏洩防止

複数の AP で連携した利用者情報の漏洩を防止する実験を行い，利用者が利用者情報の伝搬経路を把握できるか否か評価した．

（評価 5）AP の実行から漏洩検知までの処理時間の測定

テスト AP を作成し，Android，TaintDroid，および本手法における AP の実行から動作終了までの処理時間を測定し，オーバヘッドを比較評価した．

5.2 誤検知の検証

利用者情報として IMEI を取得したのちに，端末外部との通信を行う 2 つの AP を用いて誤検知の検証を行った．1 つ目の AP（IMEISendMail）は，取得した IMEI を端末外部に送信する．2 つ目の AP（MailApp）は，IMEI を取得後，取得した IMEI を含まないデータを端末外部に送信する．

図 2 は，1 つ目の AP（IMEISendMail）を動作させた際の警告画面である．IMEISendMail が取得した IMEI を端末外部に送信しようとしていることが分かり，利用者情報の漏洩を検知できていることが分かる．一方，2 つ目の AP（MailApp）を動作させた場合は，警告画面は表示されない．

一方，文献 [6] の手法において，同じ AP を動作させた場合は，2 つの AP の両方で利用者情報送信の可能性があると判断され，警告画面が表示される．これは，文献 [6] の手法が API を用いて端末外部への情報の送信を検知しており，実際にその情報の中に利用者情報を含むかどうかの判定ができないためである．

以上のことから，本手法では，端末外部への利用者情報の送信を検知できており，また，利用者情報が端末外部に送信される場合にのみ警告を表示できてことが分かる．このことから，文献 [6] の手法のように API を用いて利用者情報の漏洩を防止する手法と比べて誤検知を削減できていることが分かる．

5.3 利用者情報の漏洩防止

利用者情報を取得したのちに，取得した利用者情報を端末外部に送信する AP を用いて利用者情報の漏洩を防止する例を示す．また，端末外部に送信される利用者情報をダミーデータに置換することで，利用者情報の漏洩を防止する例を示す．なお，AP は誤検知の検証で用いた IMEISendMail を用いた．

図 5 AP を送信許可させた場合の送信内容
Fig. 5 Sending content in normal operation.

図 6 ダミーデータを送信した場合の送信内容
Fig. 6 Sending content in dummy data.

図 2 は，IMEISendMail を動作させた際の警告画面である．利用者は，図 2 の警告画面において以下を行うことで AP の動作を動的に制御できる．

(1)「アプリケーションを強制終了」を選択し，「実行」ボタンを押下した場合，AP はエラーにより強制終了
(2)「通常どおり個人情報を送信」を選択し，「実行」ボタンを押下した場合，端末外部に利用者情報を送信
(3)「ダミーデータを個人情報の代わりに送信」選択し，「実行」ボタンを押下した場合，利用者情報をダミーデータに置換した内容を送信

なお，**図 5** と**図 6** は，それぞれ (2) と (3) の制御方法で，情報を端末外部に送信した場合の受信メールの内容である．図 5 と図 6 の受信メールの内容にあるように，送信された IMEI の値が，000000000000000 から 012345678901234 に置換できていることが分かる．

以上のことから，本手法では，端末外部に利用者情報が漏洩する場合に，その動作を検知し，利用者の判断により AP の動作を動的に制御することで，利用者情報の漏洩を防止できている．また，利用者情報をダミーデータに置換することで，利用者情報の漏洩を防止できている．

5.4 実アプリを用いた TaintDroid と本手法との追跡精度の比較

Google Play 内の各カテゴリ（27 個）の無料アプリトップ 20 までの AP（合計 540 個）を取得し，その中からランダムに 100 個の AP を選択した．なお，これらの AP を取得した日付は，2014 年 8 月 19 日である．本手法を用いることで，選択した 100 個の AP が端末外部に利用者情報を送信する動作を検知できるか評価した．

表 4 は，選択した 100 個の AP を TaintDroid と本手法のそれぞれで動作させた結果を示している．なお，TaintDroid と本手法の追跡精度は，同じであり，同一の結果となったため，両方の結果を 1 つにまとめて載せている．

利用者情報の送信を検知した際に送信されていた利用者情報は，IMSI（International Mobile Subscriber Identity）と IMEI である．IMSI は，利用者情報の送信を検知したすべての AP で送信されていた．また，利用者情報を送信するタイミングは，AP の起動時である．なお，IMEI を送信する AP は少なく，利用者情報の送信を検知した AP（69 個）の内 9 個である．

表 4 実アプリの動作結果
Table 4 Result of AP execution.

AP の動作内容	該当する AP の個数
利用者情報の送信あり（検知あり）	69 個
利用者情報の送信なし（検知なし）	6 個
正常に動作しない	17 個
インストール不可	8 個

IMSI と IMEI 以外の利用者情報の送信に関して，GPS などのセンサ情報の取得は，エミュレータ上で AP がセンサ情報を取得できない．このため，GPS などのセンサ情報に関する利用者情報の送信は検知できていない．また，AP がセンサ情報を取得できないことにより，AP が正常に動作しない場合があった．

本評価において，TaintDroid と本手法を比較した結果，TaintDroid と本手法の追跡精度は，同じである．これは，本手法が TaintDroid を使用しているためであり，本評価において，本手法の実装により TaintDroid の機能が失われていないことが分かる．また，検知のタイミングについて，本手法は，AP が利用者情報を外部に送信するタイミングでその動作を検知し，動的に利用者が送信の実行可否を制御できる．さらに，本手法は，利用者情報を追跡する際に利用者情報の伝搬に関わった AP 名などの TaintDroid が保持していない情報を保持している．

なお，本手法において，AP が利用者情報を外部に送信する動作を強制的に終了した場合，その後の処理は AP 側の実装に依存する．たとえば，本評価で利用者情報の送信を検知した 69 個の AP において，AP が端末外部に利用者情報を送信する動作を強制的に終了させた場合，60 個の AP で再度利用者情報の送信を行う動作が実行された．このように，本手法を用いた場合でも，何度も利用者情報の送信を行う動作が実行される場合があることが分かった．

5.5 複数の AP が連携した場合の利用者情報の漏洩防止

2 つの AP が連携した場合の利用者情報の漏洩を防止する例を示す．1 つ目の AP（ReadContacts）は IMEI を取得したのち 2 つ目の AP（SendMail）に取得した情報を送信する．2 つ目の AP は，送信された情報（IMEI）を端末外部に送信する．

図 7 は，上記の AP を動作させた際の警告画面である．利用者情報の漏洩に 2 つの AP が関わっていることが分かる．なお，今回は，2 つのアプリが連携した場合の例を示

図 7 複数の AP が連携した場合の警告画面
Fig. 7 Warning screen in multiple APs cooperation.

表 5 処理時間の比較（単位：ms）
Table 5 Comparison of processing time (ms).

測定タイミング	Android	TaintDroid	本手法
AP 起動から動作終了まで	4,353.0	4,435.2	13,225.0
AP 起動から動作終了までの処理時間の内訳			
AP 起動から画面表示	94.3	132.4	128.0
情報の取得	7.3	8.8	9.5
メッセージ送信処理	4,248.6	4,291.6	13,084.8
メッセージ送信処理の処理時間の内訳			
送信するテキストの設定	44.0	47.2	52.3
コネクト処理	2,579.0	2,344.4	4,983.5
メッセージの送信	1,733.0	1,845.2	7,978.3
その他	56.2	54.8	59.3

したが，2 つ以上のアプリが連携した場合も同様に利用者情報の漏洩に関わったすべての AP の AP 名が，図 7 の警告画面に表示される．

以上のことから，本手法では，複数の AP が連携した場合であっても利用者情報の漏洩に関わった AP をすべて把握できており，利用者は利用者情報の漏洩に関わった AP を正確に把握できる．

5.6 処理時間の測定比較

テスト AP を作成し，Android，TaintDroid，および本手法における AP の起動から動作終了までの処理時間を測定し比較した．

作成したテスト AP は，実行後，ボタン処理などの人による操作を行わず，電話番号を取得し，その後，端末外部に電話番号を送信するものである．

評価結果を**表 5** に示す．本手法は，TaintDroid と比べて AP 起動から動作終了までの処理時間において 9 秒弱処理が遅くなっている．また，AP 起動から動作終了までの処理時間の内訳を確認すると，9 秒弱のオーバヘッドのほとんどがメッセージ送信処理で発生していることが分かる．さらに，メッセージ送信処理の処理時間の内訳を確認すると，コネクト処理とメッセージ送信時にオーバヘッドが発生していることが分かる．これは，AP が利用者情報を送信する際，その動作を検知し制御しているためである．また，メッセージ送信時の処理時間には，本手法によって提示された警告画面上での操作による処理時間（1〜2 秒程度）が含まれている．

本手法で発生するオーバヘッドは，利用者が体感できるものである．しかし，AP が利用者情報を端末外部に送信する際にのみ発生するものであり，許容できる範囲のものであると考える．また，AP ごとに次回以降の警告をなくすことができ，この場合，警告画面の表示によるオーバヘッドを軽減できる．

6. 関連研究

端末外部に利用者情報を漏洩させるマルウェアに対処するために，AP の解析手法や利用者情報の漏洩防止手法が研究されている．AP の解析手法として，DroidScope [15] や文献 [16] の手法が提案されている．DroidScope は，仮想化技術を用いた Android マルウェア解析手法であり，Java コンポーネントとネイティブコンポーネント間での利用者情報の伝搬追跡やいくつかの解析ツールを解析者に提供している．一方，本手法では，利用者情報の漏洩防止を目的としており，利用者情報の漏洩を検知するために，動的解析手法である TaintDroid を利用している．このため，本手法は DroidScope と比べて，利用者情報の伝搬追跡範囲は狭いが実際の Android 端末上で動作させることが可能であり，実際の Android 端末の情報が必要な AP の動作の検出が可能であるという特徴がある．

また，文献 [16] の手法は，TaintDroid を拡張し，利用者情報の対象範囲と追跡範囲を拡張した手法であり，従来の TaintDroid では追跡できなかったデータベースを介した利用者情報の送信を検出できる．一方，本手法は文献 [16] の手法と異なり，TaintDroid を拡張することによる利用者情報の対象範囲と追跡範囲の拡張は行っていない．しかし，インテント処理をフックすることで，AP 間の利用者情報の伝搬経路を把握し，TaintDroid の機能では把握できない AP 間のやり取りを把握できる．これにより，複数の AP が連携した利用者情報の漏洩であっても，漏洩要因であるすべての AP を把握し，対処できる．なお，文献 [16] の手法のように TaintDroid を拡張した手法と本手法を併用することで，利用者情報の伝搬追跡範囲を拡張することが可能である．

さらに，TaintDroid を利用した手法として，AppFence [11] がある．提案手法と同様に TaintDroid を利用しているため，AppFence の利用者情報の追跡粒度は同じである．また，AppFence は，利用者情報を取得した際にダミーデータに変換する Data shadowing と，取得した利用者情報を外部に送信ときに遮断する Exfiltration blocking を実現している．一方，提案手法は，AppFence にはない機能とし

て，利用者情報の漏えいに関わったAPを正確に把握する機能，実際に利用者情報が端末外部に漏洩する際にその情報をダミーデータに置換する機能，および利用者情報を外部に送信するタイミングでの利用者へ情報を提示しその動作可否を制御する機能を実現している．

次に，利用者情報の漏洩防止手法として，文献 [17] や文献 [18] の手法が提案されている．文献 [17] の手法は，APによる利用者情報へのアクセスや端末外部への利用者情報の送信を即時に利用者に通知し，制御する．これは，Androidにおける APIの使用をフックし，利用者承認の確認機構を実装することで実現している．一方，本手法は，TaintDroidを利用し，変数レベルで利用者情報の伝搬を追跡している．このため，利用者情報が AP間をどういった経路で伝搬したのかを把握し，利用者情報が実際に端末外部に漏洩する場合にのみ利用者に警告を提示できる．したがって，文献 [17] の手法と比較して本手法の方が追跡粒度は細かく，誤検知による警告頻度は低いといえる．また，文献 [17] の手法では，APIの使用を利用者が拒否した場合に，APに返却する値として nullを返却している．このため，APが強制終了するなどの問題が発生する可能性がある．一方，本手法では，端末内部での APによる利用者情報の取得を制限せず，利用者情報が端末外部に送信される場合にのみ，APの動作を制御している．これにより，利用者情報を端末内部で使用する限りは，APが異常動作を起こすことを防止する．また，端末外部に送信される利用者情報をダミーデータに置換して送信することにより，外部への情報送信 APIを APに成功として見せることで，できる限り APが強制終了するなどの異常な動作を削減している．

本手法の最初の提案 [19] の後で提案された文献 [18] の手法では，Androidにおけるプライバシ漏洩を防止する手法が提案されている．この手法は，APのバイトコードを書き換えることで，情報フローの追跡，利用者情報の漏洩検知，およびポリシによる制御を実現している．この手法では，情報フローの追跡は，Taint追跡ではなくパラメータを付加した Sourceと Sinkを用いて行っている．また，情報フローを追跡するために，APのインストール時に APごとにバイトコードを書き換える．このため，この手法では，1つの AP内に閉じた情報フローしか追跡できない．一方，本手法では，Android全体での利用者情報の流れを追跡している．このため，複数の APが連携した利用者情報の漏洩であっても，漏洩要因であるすべての APを把握し，対処できる．

また，文献 [18] の手法の利用者に提示される警告ダイアログでは，漏洩する利用者情報の種類，情報の送信先，および次回以降に同一の警告があった場合の通知の有無の3つの内容が提示され，利用者に APの動作を許可するか否かの判断を仰いでいる．一方，本手法で利用者に提示される情報は，4.4.1項で述べた9個の情報であり，文献 [18] の手法で表示される情報をすべて含んでいる．さらに，本手法では，文献 [18] の手法とは異なり，APの動作に対して行える選択肢として許可するか否かの二択だけでなく送信される利用者情報をダミーデータに置換し，APの動作を終了せず，利用者情報の漏洩を防止する選択肢を利用者が選択できる．

7. おわりに

既存研究の問題点への対処として，TaintDroidを用いた細粒度の情報追跡による利用者情報送信の動的制御手法の設計，実現方式，および評価について述べた．本手法では，TaintDroidを用いて利用者情報の伝搬を追跡し，端末外部に利用者情報が漏洩する場合，利用者の判断に従って APの動作を動的に制御する．これにより，端末外部への利用者情報の漏洩を防止できることを示した．また，AP間で利用者情報のやり取りがあった場合，利用者情報の伝搬経路を把握し，利用者に提示することで，利用者は利用者情報の漏洩に関わった APの情報を正確に把握し，利用者情報の漏洩に関わったそれぞれの APに対処できることを示した．さらに，端末外部に送信される利用者情報をダミーデータに置換することで，APの正常な動作を妨害することなく利用者情報の漏洩を防止できることを示した．

評価では，本手法を用いることで，APIを用いた利用者情報の漏洩防止手法と比較して誤検知が削減できることを示した．また，利用者情報の漏洩に関わったすべての APの情報を利用者に提示でき，利用者の判断に従って APの動作を動的に制御できることを示した．さらに，追跡精度と処理のオーバヘッドを評価した結果を示した．

謝辞 本研究の一部は，栢森情報科学振興財団平成23年度研究助成による．

参考文献

[1] Android, available from ⟨http://www.android.com/⟩ (accessed 2013-11-29).
[2] Google Play, available from ⟨https://play.google.com/store⟩ (accessed 2013-11-29).
[3] Zhou, Y. and Jiang, X.: Dissecting Android Malware: Characterization and Evolution, *Proc. 33rd IEEE Symposium on Security and Privacy* (*Oakland 2012*) (2012).
[4] Symantec：日本の Androidユーザーから利用者情報を盗み出す “The Movie” マルウェア，入手先 ⟨http://www.symantec.com/connect/ja/blogs/android-movie⟩ （参照 2013-11-29）.
[5] 産経ニュース，スマホアプリで76万件分の利用者情報流出か「全国電話帳」インストールに注意，入手先 ⟨http://sankei.jp.msn.com/affairs/news/121006/crm12100617380008-n1.htm⟩ （参照 2013-04-10）.
[6] Sakamoto, S., Okuda, K., Nakatsuka, R. and Yamauchi, T.: DroidTrack: Tracking and Visualizing Information Diffusion for Preventing Information Leakage on Android, *Journal of Internet Services and Information Security* (*JISIS*), Vol.4, No.2, pp.55–69 (2014).

[7] 林 里香，後藤厚宏：Android アプリケーション利用の安全性を高めるアプリケーション動作の「見える化」，コンピュータセキュリティシンポジウム 2012（CSS2012）論文集，Vol.2012, No.3, pp.130–137 (2012).
[8] Beresford, A.R., Rice, A., Skehin, N. and Sohan, R.: MockDroid: Trading privacy for application functionality on smartphones, *Proc. 12th Workshop on Mobile Computing Systems and Applications* (*HotMobile*) (2011).
[9] Enck, W., Gilbert, P., Chun, B.G., Cox, L.P., Jung, J., McDaniel, P. and Sheth, A.N.: TaintDroid: An information-flow tracking system for realtime privacy monitoring on smartphones, *Proc. 9th USENIX Symposium on Operating Systems Design and Implementation* (*OSDI*) (2010).
[10] 梶原直也，堀 良彰，櫻井幸一：情報フロー追跡を用いた Android 端末における情報送信制御，2013 年暗号と情報セキュリティシンポジウム（SCIS2013）論文集，電子媒体 (2013).
[11] Hornyack, P., Han, S., Jung, J., Schechter, S. and Wetherall, D.: These aren't the droids you're looking for: retrofitting android to protect data from imperious applications, *Proc. 18th ACM Conference on Computer and Communications Security* (2011).
[12] Bugiel, S., Davi, L., Dmitrienko, A., Fischer, T., Sadeghi, A. and Shastry, B.: Towards Taming Privilege-Escalation Attacks on Android, *Proc. 19th Annual Network and Distributed System Security Symposium* (*NDSS*) (2012).
[13] Schlegel, R., Zhang, K., Zhou, X., Intwala, M., Kapadia, A. and Wang, X.: Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones, *Proc. 18th Annual Network & Distributed System Security Symposium* (*NDSS '11*), pp.17–33 (2011).
[14] 田原裕暉，渡邊華奈子，大月勇人，瀧本栄二，川端秀明，竹森敬祐，毛利公一：Android アプリ内の利用者情報を収集するモジュール特定方式，2015 年暗号と情報セキュリティシンポジウム（SCIS2015）論文集，電子媒体 (2015).
[15] Yan, L.K. and Yin, H.: DroidScope: Seamlessly Reconstructing the OS and Dalvik Semantic Views for Dynamic Android Malware Analysis, *Proc. 21st USENIX Conference on Security Symposium* (2012).
[16] 名雲孝昭，秋山満昭，針生剛男：ContentProvider を用いた利用者情報送信の動的解析手法に関する検討，情報処理学会研究報告，Vol.2013-CSEC-60, No.52, pp.1–6 (2013).
[17] 川端秀明，磯原隆将，竹森敬祐，窪田 歩，可児潤也，上松晴信，西垣正勝：Android における細粒度アクセス制御機構，情報処理学会論文誌，Vol.54, No.8, pp.2090–2102 (2013).
[18] Zhang, M. and Yin, H.: Efficient, context-aware privacy leakage confinement for android applications without firmware modding, *Proc. 9th ACM Symposium on Information Computer and Communications Security* (*ASIACCS '14*), pp.259–270 (2014).
[19] 小倉禎幸，山内利宏：細粒度の情報追跡による機密情報送信の動的制御手法，情報処理学会研究報告，Vol.2013-CSEC-62, No.20, pp.1–7 (2013).

推薦文

Android 端末における利用者情報送信の動的制御手法に関する同著者の先行発表の提案結果に基づき，その実現を行い，十分な評価を行っている点は有用性が高いものとして評価できるため，推薦したい．

（コンピュータセキュリティシンポジウム 2014
プログラム委員長 井上大介）

小倉 禎幸

2013 年岡山大学工学部情報工学科卒業．2015 年同大学大学院自然科学研究科博士前期課程修了．コンピュータセキュリティに興味を持つ．

山内 利宏 （正会員）

1998 年九州大学工学部情報工学科卒業．2000 年同大学大学院システム情報科学研究科修士課程修了．2002 年同大学院システム情報科学府博士後期課程修了．2001 年日本学術振興会特別研究員（DC2）．2002 年九州大学大学院システム情報科学研究院助手．2005 年岡山大学大学院自然科学研究科助教授．現在，同准教授．博士（工学）．オペレーティングシステム，コンピュータセキュリティに興味を持つ．2010 年度 JIP Outstanding Paper Award, 2012 年度情報処理学会論文賞各受賞．電子情報通信学会，ACM，USENIX，IEEE 各会員．