情報処理学会論文誌 Vol.54 No.9 2220–2231 (Sep. 2013)

SEAndroidの拡張によるAPの動的制御手法の実現

矢儀 真也[1] 山内 利宏[1,a)]

受付日 2012年11月30日, 採録日 2013年6月14日

概要：Android において，管理者権限を取得して端末を制御するマルウェアや AP のパーミッションを悪用して利用者情報を漏えいさせるマルウェアによる被害がある．これらのマルウェアへの対策として，SEAndroid を利用する方法がある．しかし，SEAndroid の機能を活用するには，ユーザが SEManager を利用して AP のパーミッションを適切に無効化するのが難しいという問題がある．また，SEAndroid による AP 間の通信の制御は，誤検知により必要な通信を拒否する可能性が高いという問題がある．そこで，本論文では，SEAndroid を拡張することにより，ユーザが AP を動的に制御できる手法を提案する．提案手法は，Permission Revocation を拡張し，AP がパーミッションを利用するとき，パーミッションの利用可否を動的に制御する機能を実現する．また，提案手法は，Tag Propagation を拡張し，AP 間の通信を動的に制御する機能を実現する．この機能では，AP が利用者情報を漏えいさせる可能性のある通信をするとき，およびユーザがインストールした AP が最初からインストールされている AP と通信するとき，ユーザに通信の許可を求める．本論文では，SEAndroid の問題点と対処方法を示し，設計と評価結果を報告する．

キーワード：Android，SEAndroid，情報漏えいの防止，パーミッションの制御

Implementation of a Method for Dynamic Control of Application Programs by Extending SEAndroid

Shinya Yagi[1] Toshihiro Yamauchi[1,a)]

Received: November 30, 2012, Accepted: June 14, 2013

Abstract: In Android, there is some damage by malicious software that obtains root privilege to control device or leaks personal information by using permission allowed to application programs (AP). SEAndroid is effective for mitigating damages caused by these malicious software. However, it is difficult for users to use SEAndroid properly. To leverage its function, a user has to revoke permissions by SEManager. Moreover, policy enforcement in SEAndroid might reject required communication. In this paper, we propose a method for dynamic control of AP by extending SEAndroid. In our proposal, when AP uses a permission, our system dynamically asks whether a user allows it or not by extending Permission Revocation function. The proposed method can dynamically control the communication among APs by extending Tag Propagation. Moreover, it also asks when there is a possibility that AP leaks personal information or AP installed by user communicates with trusted AP. This paper describes the problems of SEAndroid. This paper also reports the design and evaluation results.

Keywords: Android, SEAndroid, prevention of information leakage, permission control

1. はじめに

スマートフォン向けのオペレーティングシステムの 1 つとして Android [1] が広く利用されている．Android のユーザは，Google Play [2] からアプリケーションプログラム（Application Program：以降，AP と略す）をスマートフォンに自由にダウンロードして，インストールできる．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，利用者情報を外部に漏えいさせるマルウェア，およびネット

[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

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などが発見され [3]，被害を及ぼしている．

上記のマルウェアによる被害は，管理者権限の問題とパーミッション機構の問題に分類できる．管理者権限を攻撃者が取得した場合，あらゆる操作が可能となる．また，AP を利用するには，AP のインストール時に AP 作成者が要求するパーミッションをユーザが承認する必要がある．しかし，要求されたすべてのパーミッションをユーザが承認する必要があるため，ユーザが不要だと判断したパーミッションを拒否することができない．

Android のセキュリティ上の問題を改善するために，National Security Agency（以降，NSA と略す）[4] は Security Enhanced Android（以降，SEAndroid と略す）[5] を開発している．SEAndroid は Security Enhanced Linux（以降，SELinux と略す）[6] をベースとした強制アクセス制御（Mandatory Access Control：MAC）を持つ．これにより，特権を持つデーモンの動作を制限し，攻撃者による管理者権限の取得を防止できる [7]．また，AP に付与されているパーミッションを AP のインストール後に無効化できる Permission Revocation や，AP に付与したタグに基づいて AP 間の通信を制御する Tag Propagation と呼ばれる機能を持つ．

しかし，Permission Revocation を利用するには，ユーザが SEManager と呼ばれるツールを利用して AP ごとに不要なパーミッションを無効化する必要がある．このため，多くのユーザにとって利用は難しい．また，Tag Propagation は，誤検知により，必要な通信を拒否する可能性が高い．

本論文では，これらの問題に対処するために，SEAndroid を拡張することにより，ユーザが SEAndroid の機能を利用して AP を動的に制御できる手法を提案する．提案手法は，AP がパーミッションを利用するとき，システムがユーザにパーミッションの許可を求める．これにより，ユーザが個々のパーミッションを動的に制御できる．また，AP の種類や AP がアクセスできるリソースの情報を含むタグに応じて AP 間の通信を制御する．制御対象の通信の場合は，ユーザに通信を許可するか否かを尋ねる．これにより，必要な通信を妨害することなく利用者情報の漏えいを防止できる．さらに，複数の AP を用いた情報漏えい防止の評価結果，アラート頻度の評価結果，SEAndroid との比較評価結果，およびオーバヘッドの評価結果について述べる．

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，端末やユーザに関連する情報のことを示す（詳細は，4.3.3 項で述べる）．

2. Android のセキュリティ機構と問題点

2.1 Android のセキュリティ機構

Android は Dalvik VM と呼ばれる仮想マシン上ですべての AP を実行する．AP は，個別のプロセスとして動作するため，他の AP のメモリ空間にアクセスできない．また，AP ごとに固有のユーザ ID が割り当てられており，AP は，他の AP の資源にアクセスできない．なお，AP が他の AP と通信するときはインテントを利用する．

AP がアクセスできる資源は，パーミッションで制御されている．たとえば，AP が電話帳にアクセスするには，READ_CONTACTS パーミッションが必要である．なお，AP が利用できるパーミッションは，AP のインストール時にユーザが承認する．

2.2 問題点

Android のセキュリティ上の問題点として，以下の 5 つがある [8]．

（問題 1）管理者権限の取得に関する問題
（問題 2）開発支援ツールに関する問題
（問題 3）パーミッションの悪用
（問題 4）情報漏洩の検知が不可能
（問題 5）WebKit の悪用

（問題 1）は管理者権限に関する問題であり，（問題 2），（問題 3），（問題 4），および（問題 5）はパーミッション機構に関する問題である．これらの問題は，SEAndroid を利用することで対処できる．

3. SEAndroid の機能と問題点

3.1 機能

NSA は Android のセキュリティ上の問題を改善するために，SEAndroid を開発している．SEAndroid は，SELinux をベースとして開発されており，SELinux の機能に加えて，Android のパーミッションを制御する機能や AP 間の通信を制御する機能などを持つ．SEAndroid の機能を利用することで，管理者権限の取得や情報漏えいを防止できる．なお，本論文では，2012 年 6 月 4 日現在の SEAndroid を対象とする．

SEAndroid は，カーネルレベルのアクセス制御機構（以降，Kernel MAC と呼ぶ）と Android のフレームワークレベルのアクセス制御機構（以降，Middleware MAC と呼ぶ）を持つ．

Kernel MAC は特権を持つデーモンに固有のラベルを割り当てて，ラベルごとにアクセスできる資源を設定する．これにより，特権を持つデーモンの動作を制限し，攻撃者による管理者権限の取得を防止する．なお，管理者権限を取得できたとしても，Kernel MAC によりアクセスを制限できる．また，Multi Category Security（以降，MCS と略す）を利用して AP ごとに違うカテゴリを設定し，AP 間の隔離を強制する．MCS は，ファイルやディレクトリなどの資源をカテゴリで分類し，プロセスがアクセスできるカテゴリを決定するアクセス制御機構である．

Middleware MAC には以下の 3 つの機能がある．

(1) Install-time MAC
(2) Permission Revocation
(3) Tag Propagation

Install-time MAC は，AP のインストールを許可するか否かをポリシに基づいて判定する機能であり，マルウェアの可能性が高い AP のインストールを防止する．

Permission Revocation は，AP のインストール時にユーザが承認したパーミッションを，インストール後に無効化できる機能である．なお，パーミッションの無効化には SEManager を用いる．

Tag Propagation は，AP にタグを付与して，AP 間の通信によるタグの拡散を管理する機能である．たとえば，利用者情報を取得するパーミッションを持つ AP に sensitive_data というタグを付与する．この AP が他の AP と通信した場合，利用者情報が拡散した可能性があるため，通信先の AP に sensitive_data を付与する．これにより，利用者情報の拡散を管理できる．また，AP 間の通信をタグと関連付けたポリシに基づいて制御する．たとえば，利用者情報を取得するパーミッションを持つ AP に sensitive_data を付与し，外部と通信するパーミッションを持つ AP に sinks というタグを付与する．この 2 つの AP が通信する場合，利用者情報が外部に漏えいする可能性があるため，通信を拒否する．これにより，利用者情報の漏えいを防止できる．なお，タグは，AP のインストール時，および AP 間の通信時に付与される．

3.2 問題点

SEAndroid には，以下の問題がある．

（問題 1）Permission Revocation の利用が困難

Permission Revocation を利用するには，ユーザは，SEManager を利用して AP ごとに不要なパーミッションを無効化する必要がある．しかし，ユーザは SEManager の存在を知らないことや，どのパーミッションをあらかじめ無効化すればよいか分からないという問題がある．このため，多くのユーザにとって，Permission Revocation を利用することは難しい．

（問題 2）Tag Propagation の誤検知

Tag Propagation の誤検知により，必要な通信を実行できないという問題がある．たとえば，カレンダ AP（com.android.calendar）は，自身のプロバイダ（com.android.providers.calendar）と通信する．しかし，カレンダ AP は，利用者情報を取得するパーミッションを持ち，プロバイダはインターネットへアクセスするパーミッションを持つ．このため，Tag Propagation は，利用者情報の漏えいの可能性を検知して通信を拒否する．これにより，必要な通信ができないという問題がある．

（問題 3）最小特権のポリシの作成が困難

SEAndroid は，プロセスなどの動作の主体（以降，サブジェクトと呼ぶ）とファイルなどの動作の対象（以降，オブジェクトと呼ぶ）にそれぞれラベルを割り当て，ラベルに基づいてホワイトリスト方式のアクセス制御を行う．つまり，ポリシで許可している動作は制限しないため，サブジェクトとオブジェクトのすべての組合せについて最小特権を満たすポリシを作成する必要がある．しかし，デーモンにはそれぞれ固有のラベルがあり，AP には最初からインストールされている AP，ウェブブラウザ，およびユーザがインストールした AP の 3 種類のラベルがある．さらに，オブジェクトには約 100 種類のラベルがある．このため，ユーザが適切なポリシを設定するのは，非常に難しい．以上のことから，最小特権のポリシを作成することは困難であるといえる．

本論文では，上記の問題のうち（問題 1）と（問題 2）に対処し，AP を動的に制御できる手法を提案する．

4. SEAndroid の拡張による AP の動的制御手法の設計

4.1 前提条件

提案手法では SEAndroid を利用する．このため，SEAndroid を利用できる Android 4.0.3 以降を対象とする．また，Kernel MAC のポリシは最小特権を実現しているものと仮定する．

4.2 設計方針

(1) SEAndroid を拡張すること
(2) ユーザが AP の動作を制御できること
(3) ユーザによる AP の動作の可否判断を支援すること
(4) 頻繁な問合せによるユーザの負担を少なくすること

提案システムは，SEAndroid の既存の機能を拡張することで SEAndroid の問題点のうち（問題 1），および（問題 2）に対処する．

AP がパーミッションを利用するとき，および AP 間で通信するときに，AP の動作を許可するか否かをユーザに尋ねるように SEAndroid を拡張する．これにより，すべてのユーザが Permission Revocation を利用できるため，（問題 1）に対処できる．また，Tag Propagation を拡張して，AP 間の通信のルールを細分化すること，および通信を許可するか否かをユーザが決定できるようにすることで，Tag Propagation の誤検知を削減し，誤検知による影響を緩和する．これにより，（問題 2）に対処できる．

なお，AP の動作を制御するべきかどうかをユーザが判断できないことが考えられる．そこで，パーミッションの利用時と AP 間の通信時に，ユーザの判断を支援する情報をシステムが自動的に提示する．

また，すべての AP を制御対象とした場合，ユーザへの負担が大きいという問題がある．たとえば，ユーザが電話をかけるたびに，電話をかけることを許可するか否かを

ユーザが選択する必要がある．そこで，ユーザが制御対象の AP を決定できるようにする．さらに，同じアクセスに対する判断をユーザがする必要がないように，ユーザの判断結果を次回以降に反映できるようにする．これにより，ユーザの負担を減らす．

4.3 設計

4.3.1 基本構成

本手法では AP によるパーミッションの利用を動的に制御する（以降，パーミッション制御機能と呼ぶ）．パーミッション制御機能は，AP が危険なパーミッションを利用するときに，利用の許可をユーザに求める．この際，ユーザの判断を支援する情報をシステムが自動的に提示する．これにより，ユーザがパーミッションを制御できるため，3.2 節の（問題 1）に対処できる．なお，ユーザにパーミッションの利用の許可を求めるとき，AP のコンポーネントの 1 つであるアクティビティが必要となる．このため，情報の提示やパーミッションの利用の許可をユーザに求める AP（以降，制御 AP と呼ぶ）を利用する．

また，AP ごとに付与しているタグを利用して，AP 間の通信を制御する（以降，通信制御機能と呼ぶ）．通信制御機能は，利用者情報が漏えいする可能性がある通信をするとき，およびユーザがインストールした AP が最初からインストールされている AP と通信するときに，通信の許可をユーザに求める．3.2 節の（問題 2）は，最初からインストールされている AP 間の通信を sensitive_data と sinks の 2 つのタグのみで制御することで発生する．このため，AP の種類に応じたタグを追加して通信を制御することで誤検知を削減する．また，誤検知が起こった場合でも，ユーザの判断により通信を許可することができる．

提案システムの全体像を**図 1** に示す．図 1 の拡張した SEAndroid は，従来の SEAndroid にパーミッション制御機能と通信制御機能を追加したものである．なお，パーミッション制御機能と Permission Revocation の差異は，パーミッションを制御するタイミング，制御対象のパーミッション，および制御対象の AP である．また，通信制御機能と Tag Propagation の差異は，タグの種類，制御対象の通信，タグが拡散するルール，およびタグを付与するタイミングである．

これらの機能により，SEAndroid に詳しくないユーザでも危険なパーミッションを制御できる．また，従来の問題である誤検知による通信拒否の影響を緩和する．さらに，利用者情報の漏えいの防止に加えて，最初からインストールされている AP の安全性を向上させることができる．

4.3.2 パーミッション制御機能

パーミッション制御機能は，AP によるパーミッションの利用を動的に制御する機能である．本機能は SEAndroid の既存の機能である Permission Revocation を拡張することにより実現する．本機能の処理の流れを**図 2** に示し，以下で説明する．

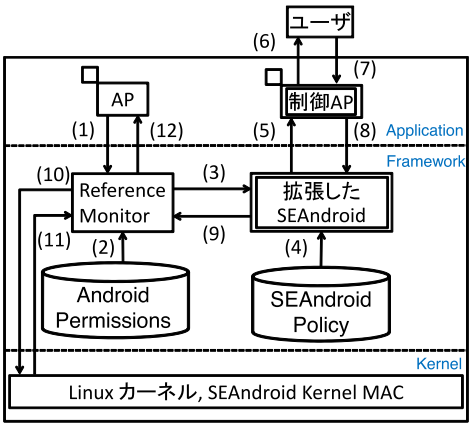

図 1 提案システムの全体像
Fig. 1 System overview.

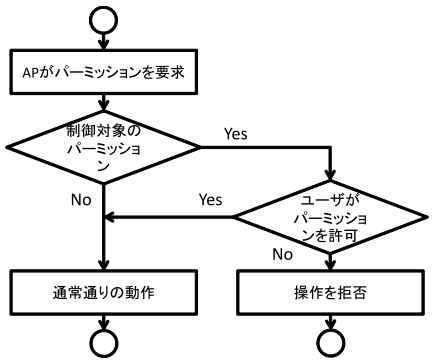

図 2 パーミッション制御機能の処理の流れ
Fig. 2 Control permission flow.

(1) AP がパーミッションを要求
(2) システムが AP のタグ，要求されたパーミッション，および過去の判断結果を参照
(3) 制御対象の場合，ユーザにパーミッションの利用の許可を制御 AP を利用して提案
(4) ユーザが許可した場合，パーミッションの利用を許可
(5) ユーザが拒否した場合，パーミッションの利用を拒否

本機能の設計として，制御対象のパーミッション，制御対象の AP，およびユーザに提示する情報を以下に記述する．

（設計 1）制御対象のパーミッション

すべてのパーミッションを制御対象とした場合，AP がパーミッションを要求するたびにユーザに許可を求めるため，ユーザの負担が大きい．そこで，制御対象のパーミッションを検討した．制御対象のパーミッションを**表 1** に示す．なお，Android 4.0.4_r1.1（API Level 15）を対象として検討した．

制御対象のパーミッションを検討した基準は，パーミッションの保護レベルが dangerous であること，および利用者情報を取得するパーミッション以外であることの 2 つである．

表 1 制御対象のパーミッション
Table 1 Controlled permission.

SIGNAL_PERSISTENT_PROCESSES
SET_ALWAYS_FINISH
SET_DEBUG_APP
SET_PROCESS_LIMIT
WRITE_CONTACTS
WRITE_HISTORY_BOOKMARKS
WRITE_SOCIAL_STREAM
ADD_VOICEMAIL
WRITE_CALENDAR
WRITE_PROFILE
SEND_SMS
CALL_PHONE
WRITE_SMS
BLUETOOTH
INTERNET
USE_SIP
NFC
AUTHENTICATE_ACCOUNTS
MANAGE_ACCOUNTS
USE_CREDENTIALS
WRITE_EXTERNAL_STORAGE
PROCESS_OUTGOING_CALLS
MODIFY_AUDIO_SETTINGS
CHANGE_WIFI_MULTICAST_STATE
CLEAR_APP_CACHE
PERSISTENT_ACTIVITY
DISABLE_KEYGUARD
BLUETOOTH_ADMIN
CHANGE_CONFIGURATION
MOUNT_UNMOUNT_FILESYSTEMS
CHANGE_WIMAX_STATE
SET_TIME_ZONE
MOUNT_FORMAT_FILESYSTEMS
WRITE_SYNC_SETTINGS
SYSTEM_ALERT_WINDOW
WRITE_SETTINGS
CHANGE_NETWORK_STATE
SET_ANIMATION_SCALE
REORDER_TASKS
CHANGE_WIFI_STATE
SUBSCRIBED_FEEDS_WRITE

保護レベルは，normal，dangerous，signature，および signatureOrSystem の 4 つがある．normal は，危険性が低いパーミッションである．dangerous は，危険性が高いパーミッションであり，AP のインストール時にユーザが承認する必要がある．signature は，パーミッションを定義している AP と利用する AP が同じ署名を持つ場合のみ利用できるパーミッションである．signatureOrSystem は，システムイメージに含まれている AP と同じ署名を持っている場合に利用でき，サードパーティの AP では利用されないパーミッションである．これらのことから，危険性が高い dangerous を対象とした．

なお，利用者情報を取得するパーミッションを制御対象外とした．これは，提案システムは AP が利用者情報を外部に漏えいさせる可能性があるときに制御するので，利用者情報を取得する際の制御は不必要と考えたためである．このことから，ユーザへの負担を考慮して対象外とした．また，提案システムは AP がパーミッションを利用するときに当該パーミッションの利用可否を制御するため，AP のアップデートにより新しくパーミッションが追加された場合でも追加されたパーミッションの利用可否を制御できる．

（設計 2）制御対象の AP

すべての AP を制御対象とした場合，ユーザの負担が大きい．そこで，制御対象の AP を検討した．

AP を最初からインストールされている AP（以降, trusted AP と呼ぶ），ユーザがインストールした AP のうちユーザが信頼できる AP（以降，user_trusted AP と呼ぶ），およびユーザがインストールした AP のうちユーザが信頼できない AP（以降，untrusted AP と呼ぶ）の 3 種類に分類する．

提案システムは，user_trusted AP については利用者情報の外部への漏えいを防止する．また，untrusted AP については利用者情報の外部への漏えいを防止することに加えて，表 1 のパーミッションを制御する．なお，untrusted AP のみ表 1 のパーミッションを制御する理由と，user_trusted と untrusted の判断基準は以下のとおりである．

user_trusted AP と untrusted AP は，AP の配布元のサイトを信頼できるか否かと表 1 のパーミッションの利用可否を制御したいか否かによりユーザが決定する．これは，信頼できるサイトから取得した AP の場合，ユーザへの負担軽減のため，表 1 のパーミッションの利用可否の問合せを行わなくてもよいと考えたからである．たとえば，電気通信事業者が管理しているサイトのように，マルウェアか否かをチェックされたうえで公開されているマーケットから AP を取得した場合，配布元のサイトは信頼できるため，ユーザは AP が利用できるパーミッションの利用可否を制御したい場合は，untrusted AP，そうでない場合は user_trusted AP として利用する．一方，信頼できないサイトから AP を取得した場合，上記のマーケットよりも危険性が高いため，untrusted AP として利用する．なお，ユーザが user_trusted AP と untrusted AP の選択を誤った場合でも，利用者情報の漏えいをどちらの選択でも防止できるため，安全性への影響は小さいといえる．

以上のことから，ユーザは AP の配布元のサイトを信頼できるか否かと表 1 のパーミッションを制御したいか否かに応じて，表 1 のパーミッションの利用可否を制御するか否かを選択できることに加えて，trusted AP は制御対象と

しないことにより，ユーザの負担を少なくできるといえる．

（設計 3）ユーザに提示する情報

untrusted AP が表 1 のパーミッションを利用するとき，システムがパーミッションの利用の許可をユーザに求める．しかし，パーミッション名だけではユーザは許可を求められているパーミッションにどのような危険性があるかを判断できないという問題がある．

そこで，AP のパッケージ名，ホーム画面に表示される AP のアイコン，AP が利用するパーミッション名，およびパーミッションの内容を表示することでユーザの判断を支援する．なお，ここで提示する情報は，制御対象のパーミッションに共通する情報のため，通信を行うパーミッションの場合には，通信先の情報や通信内容を表示することが有用であると考えられる．たとえば，INTERNET パーミッションの場合，通信先の情報として，IP アドレス（ホスト名）とポート番号の表示があり，ソケットなどの情報からこれらの情報を取得することが考えられる．

4.3.3 通信制御機能

通信制御機能は，AP 間の通信を動的に制御する機能である．本機能は，SEAndroid の既存の機能である Tag Propagation を拡張することにより実現する．本機能の処理の流れを**図 3** に示し，以下で説明する．

(1) AP が通信の要求
(2) システムが AP のタグ，および過去の判断結果を参照
(3) 制御対象の通信の場合，ユーザに通信を許可するか否かを制御 AP を利用して提案
(4) ユーザが許可した場合，通信を許可
(5) ユーザが拒否した場合，通信を拒否

本機能の設計として，タグの種類，制御対象の通信，タグを付与するタイミング，およびユーザに提示する情報を以下に記述する．

（設計 4）タグの種類

本機能では，AP に付与しているタグをもとに，AP 間の通信を制御するか否かを決定する．そこで，タグの種類を検討した．

タグは，管理用のタグ（**表 2**）と情報漏えい防止用のタグ（**表 3**）の 2 種類に分類する．

管理用のタグは，trusted，user_trusted，untrusted，および launcher の 4 つである．trusted は trusted AP に付与する．user_trusted は user_trusted AP に付与する．untrusted は untrusted AP に付与する．launcher はホーム画面である com.android.launcher に付与する．これらのタグを利用することで，AP の信頼度に基づいた制御ができる．なお，ランチャは，標準では com.android.launcher である．しかし，サードパーティのランチャを利用することを考慮して，launcher で管理する．

情報漏えい防止用のタグは，sensitive_data と sinks の 2 つである．sensitive_data は，利用者情報の取得に関するパーミッションを対象とする．sensitive_data に関するパーミッションを**表 4** に示す．パーミッションを選択した基準は，保護レベルが dangerous であり，利用者情報を取得できることである．sinks は，利用者情報を外部に漏えいする可能性があるパーミッションを対象とする．sinks に関するパーミッションを**表 5** に示す．情報漏えい防止用のタグを利用して，利用者情報の漏えいを防止する．

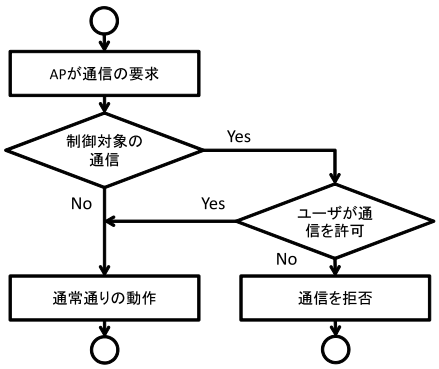

図 3 通信制御機能の処理の流れ
Fig. 3 Control communication flow.

表 2 管理用のタグの一覧
Table 2 List of tag for management.

trusted	最初からインストールされている AP
user_trusted	ユーザがインストールした AP のうちユーザが信頼できる AP
untrusted	ユーザがインストールした AP のうちユーザが信頼できない AP
launcher	AP を起動するランチャ（標準では com.android.launcher）

表 3 情報漏えい防止用のタグの一覧
Table 3 List of tag for prevention of information leak.

sensitive_data	利用者情報の取得に関連するパーミッションに対応
sinks	利用者情報を外部へ流出する可能性があるパーミッションに対応

（設計 5）制御対象の通信

制御対象の通信は，通信元の AP が sensitive_data を持ち，通信先の AP が sinks を持つ通信，および通信元の AP が untrusted もしくは user_trusted を持ち，通信先の AP が trusted を持つ通信の 2 つとする．ただし，前者については，通信元と通信先の AP の両方が trusted を持つ場合，信頼できる動作と判断し，制御対象から除外する．これらの通信を制御することで，複数の AP で連携した利用者情報の漏えい [9], [10] の防止，および trusted AP の安全性を向上させることができる．また，通信元と通信先を意識することで誤検知を削減できる．

（設計 6）タグを付与するタイミング

SEAndroid は AP のインストール時に，AP に付与されているパーミッションに基づいてタグを付与する．このため，利用者情報にアクセスしていない状態でも利用者情報の拡散，および漏えいの可能性を検知するという問題がある．そこで，タグを付与するタイミングを検討した．

user_trusted と untrusted は AP の起動時にユーザが選択する．これにより，パーミッションを制御するか否かを AP の起動時に決定できる．ユーザがインストールした AP は launcher を持つ AP から起動されるため，launcher を持つ AP からの通信をフックすることで，AP の起動時に AP の種類を決定できる．

sensitive_data はタグに関連付けられているパーミッションの利用時に付与する．これにより，利用者情報の拡散の誤検知を削減できる．

上記以外のタグは，AP のインストール時に付与する．なお，タグは SEManager を利用して変更できるようにする．

（設計 7）ユーザに提示する情報

sensitive_data を持つ AP が sinks を持つ AP と通信するとき，またはユーザがインストールした AP が trusted AP と通信するときに，システムがユーザに通信の許可を求める．この際，通信元（先）AP のパッケージ名，通信元（先）AP のアイコン，通信の内容，および通信の危険性を表示することでユーザの判断を支援する．

表 4 sensitive_data に関連付けるパーミッション
Table 4 Permissions related to sensitive_data.

READ_SOCIAL_STREAM
READ_USER_DICTIONARY
READ_PROFILE
READ_HISTORY_BOOKMARKS
READ_LOGS
READ_CONTACTS
READ_CALENDAR
ACCESS_COARSE_LOCATION
ACCESS_FINE_LOCATION
READ_SMS
READ_ATTACHMENT
RECEIVE_SMS
RECEIVE_WAP_PUSH
RECEIVE_MMS
READ_PHONE_STATE
CAMERA
RECORD_AUDIO
GET_TASKS

表 5 sinks に関連付けるパーミッション
Table 5 Permissions related to sinks.

SEND_SMS
INTERNET
WRITE_EXTERNAL_STORAGE
BLUETOOTH

4.4 期待される効果

(1) AP の動作を動的制御できること
(2) 利用者情報の漏えいを防止
(3) 誤検知による通信の拒否の影響を緩和
(4) trusted AP の安全性の向上

パーミッションの利用時と AP 間の通信時にシステムは自動的にユーザに許可を求める．このため，ユーザは危険なパーミッションの利用や AP 間の通信などの AP の動作を制御できる．また，利用者情報が漏えいする可能性がある通信を検知するため，利用者情報の漏えいを防止できる．さらに，Tag Propagation による通信の誤検知を減らすことに加えて，通信の誤検知が起こったとき，ユーザの判断で通信を許可するか否かを決定できる．これにより，誤検知による影響を緩和できる．加えて，ユーザがインストールした AP が trusted AP と通信するときに，ユーザに通信の許可を求める．このときに，ユーザが適切に判断することで trusted AP の安全性を向上させることができる．

5. 評価

5.1 評価項目

以下に評価項目と評価の目的を示す．なお，評価では，VM 上で動作する Android Emulator を用いた．ホスト OS とゲスト OS の評価環境をそれぞれ**表 6** と**表 7** に示す．

（評価 1）情報漏えいの防止

複数の AP で連携した利用者情報の漏えいを防止する実験について述べる．これにより，提案システムの有用性を示す．

（評価 2）アラートの回数

AP の種類により，提案システムがユーザにどの程度のアラートを出すかを評価する．これにより，ユーザの負担を考察する．

表 6 ホスト OS の評価環境
Table 6 Environmental evaluation of host OS.

ディストリビューション	Ubuntu 10.04.4 LTS
カーネル	Linux ubuntu 2.6.32-41-generic-pae
CPU	Intel Xeon W3520 2.67 GHz 4 コア
メモリ	6 GB
仮想化ソフトウェア	VMware Player 4.0.1

表 7 ゲスト OS の評価環境
Table 7 Environmental evaluation 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

図 4 情報漏えいの防止
Fig. 4 Preventing information leak.

（評価 3）SEAndroid との比較

提案システムの機能を従来の SEAndroid と比較することで，SEAndroid との違いを明確にする．

（評価 4）オーバヘッド

提案システムのオーバヘッドを測定し，性能への影響を考察する．

5.2 情報漏えいの防止

2 つの AP で連携した情報漏えいを防止する例を示す．1 つ目の AP（read.contacts）は電話番号を取得するパーミッションを持ち，電話番号を取得したのちに 2 つ目の AP（send.mail）に送信する．2 つ目の AP はインターネットに接続するパーミッションを持ち，電話番号を外部に送信する．

図 4 は AP 間の通信を制御している実行結果である．read.contacts が取得した電話番号（15555215554）が send.mail に送信されようとしていることが分かる．ユーザは図 4 の no のボタンをクリックすることで電話番号の漏えいを防止できる．なお，AP 間の通信を許可した場合でも，send.mail が外部と通信するときに利用者情報の漏えいを検知できる．

以上のことから，提案システムは，外部に利用者情報を漏えいさせるために必要な処理である利用者情報を取得する API，外部と通信する API を利用できる AP との通信処理，および外部と通信する API をフックできている．また，これらの API と通信処理を情報漏えいのために利用するマルウェアや AP の情報漏えいの可能性を検出できると推察できる．

5.3 アラートの回数

trusted AP のアラートの回数を**表 8** に，user_trusted AP，および untrusted AP のアラートの回数を**表 9** に示す．なお，利用した AP は最初からインストールされている AP を 5 種類，Android SDK のサンプルプログラムの中から制御対象のパーミッションを含むものを 5 種類，およびマーケットで配布されている AP のうち，カテゴリの上位 20 位以内の AP を 10 種類選択したものである．

表 8 trusted AP のアラートの回数
Table 8 Alert count of trusted AP.

パッケージ名	アラートの回数
com.android.browser	0
com.android.calendar	0
com.android.camera	0
com.android.phone	0
com.android.search	0

表 9 user_trusted AP と untrusted AP のアラートの回数
Table 9 Alert count of user_trusted AP and untrusted AP.

パッケージ名（Android SDK のプログラムは com.example.android. を省略）	保護レベルが dangerous のパーミッションの種類数	アラートの回数	
		user_trusted	untrusted
accelerometerplay	2	0	0
hcgallery	2	1	1
multires	1	0	0
musicplayer	2	0	再生する曲を変更した回数
toyvpn	1	0	1
com.cookpad.android.activities	4	0	1
com.fox_kowai	1	0	1
com.google.android.gm	9	2	4
com.google.android.stardroid	4	0	1
com.kayac.koshien	4	2	2
com.nhncorp.lineweather	6	1	1
com.nhn.android.endic	4	0	1
jp.co.ignis.battery.dapan	9	0	2
jp.co.johospace.jorte	9	0	1
jp.co.ponos.battlecats	4	0	3

表 8 より，trusted AP はアラートを出さないことが分かる．これは，ユーザがインストールした AP が trusted AP と通信しない限り trusted AP はアラートを出さないためである．これにより，ユーザは従来どおり trusted AP を利用できることが分かる．

表 9 は，ユーザがインストールした AP を user_trusted と untrusted で動作させた際に，提案システムが提示するアラートの回数をそれぞれ示している．なお，本評価は，一般的なアプリケーション動作を手動で模擬させることにより行った．表 9 の user_trusted AP に関するアラートは，trusted AP との通信と利用者情報の漏えい防止によるもの

である．hcgallery のアラートは，hcgallery が trusted AP に画像を送信したときに提示されたアラートである．また，com.google.android.gm（以降，Gmail と記す）のアラートは，com.google.android.gsf.login との通信によるものと，メールの送信によるものである．com.google.android.gsf.login は INTERNET パーミッションを持っており，利用者情報を外部に送信する可能性を検知し，アラートを表示した．さらに，メールの送信時には，Gmail が利用者情報を外部に送信する可能性を検知し，アラートを表示した．

untrusted AP のアラートは，user_trusted AP のアラートに加えて，制御対象のパーミッションの利用によるものである．musicplayer のアラートは INTERNET パーミッションの利用に関するアラートである．musicplayer は再生する曲を変更するボタンをクリックするたびに，INTERNET パーミッションを必要とする．これは，INTERNET パーミッションは，ソケットを作成するために必要なパーミッションであり，musicplayer は再生する曲を変更するボタンをクリックするたびに INTERNET パーミッションを要求してソケットを作成し，外部と通信しているためである．また，jp.co.ponos.battlecats というゲーム AP において，AP の起動時に INTERNET パーミッションを 3 回必要とした．これは，外部サーバから jar ファイルをダウンロードして広告を表示するためにソケットを 3 回作成したためである．このことから，広告のダウンロード処理によって，ユーザに複数回のアラートを提示する可能性があることが分かった．

表 9 より，本評価で利用した AP は，利用できるパーミッションのすべてを利用しているわけではなく，多くても数回程度のアラートしか出さないことが分かる．また，選択した結果を保持し，同じアラートを出さなくすることで，さらにアラート回数を減らすことができるため，ユーザへの負担は少ないといえる．

また，評価した AP のほとんどが必要とするパーミッションである INTERNET パーミッションの利用を拒否した際の挙動を調査したところ，ほとんどの AP は，インターネットへの接続のみを拒否し，AP のその他の機能は利用できた．しかし，jp.co.ponos.battlecats に不具合が発生することを確認した．具体的には，広告をダウンロードする処理がタイムアウトして，ゲームを開始できないという不具合があることを確認した．このことから，一部の AP に関しては不具合が生じるため，ユーザはこのことに留意して利用する必要がある．

5.4 SEAndroid との比較

5.4.1 比較内容

提案システムの機能を従来の SEAndroid と比較した結果を**表 10** に示し，以下で説明する．

表 10 SEAndroid と提案手法の機能の比較
Table 10 Comparing SEAndroid with proposal method.

比較項目	SEAndroid	提案手法
利用者情報の拡散防止	ポリシで強制	制御の細分化，ユーザが動的制御
パーミッションの制御	事前に設定	ユーザが動的制御
通信制御の誤検知	多い	少ない
利用者情報の漏えい防止	2 つ以上の AP が連携した場合	AP 数に依存しない
trusted AP との通信	制御しない	制御する

5.4.2 利用者情報の拡散防止

SEAndroid は sensitive_data と sinks のそれぞれのタグが通信元の AP か通信先の AP のどちらかに含まれる場合は通信を拒否する．それ以外の通信は許可して通信元の AP のタグ，および通信先の AP のタグは互いに拡散して 2 つの AP のタグは同じになる．しかし，通信先の AP が sensitive_data を持っていた場合，通信元の AP に利用者情報を送信していないにもかかわらず通信元の AP に sensitive_data のタグを拡散させるという問題がある．提案手法は 2 つのタグに加えて管理用のタグに基づいて通信を制御する．また，通信先の AP から通信元の AP にタグを拡散させないこと，および通信を制御する場合，ユーザは AP 間の通信を許可するか否かを選択できる点が SEAndroid と異なる．

5.4.3 パーミッションの制御

SEAndroid はユーザが SEManager を利用して，AP によるパーミッションの利用を事前に設定することでパーミッションを制御する．しかし，SEManager の設定は，それぞれの AP が利用できるパーミッションの一覧の中から不要なパーミッションを探して無効化する必要があることに加えて，パーミッションの内容が記述されていないため，パーミッションを無効化することにより，どのような影響が生じるかが分からないという問題がある．

提案手法は untrusted AP が制御対象のパーミッションを利用するときにユーザが動的に制御する．この際，4.3.2 項の（設計 3）に記述している情報をシステムが自動的に提示するため，ユーザは，SEAndroid の Permission Revocation よりも容易にパーミッションの利用可否を判断できる．

5.4.4 通信制御の誤検知

SEAndroid は利用者情報にアクセス可能な AP に sensitive_data のタグを付与している．このため，利用者情報にアクセスしていない場合でもタグが付与されるという問題がある．また，sensitive_data と sinks を持つ AP は trusted AP にもあり，AP の機能を利用できないという問題がある．たとえば，com.android.calendar や com.android.phone は通信制御の誤検知により，機能を利用することができない．提案手法は利用者情報にアクセスした際に sensitive_data を付与する．このため，SEAndroid よりも誤検知が少ない．

また，タグの拡散を通信元の AP から通信先の AP に拡散するように変更している．これにより，誤ったタグの拡散を削減でき，通信制御の誤検知を削減している．さらに，trusted AP 間の通信は制御対象としていないこと，および誤検知が起きた場合でもユーザが AP 間の通信を制御できるため誤検知による影響を緩和できることが SEAndroid と異なる．

5.4.5 利用者情報の漏えい防止

SEAndroid は sensitive_data と sinks の通信を拒否することで複数の AP で連携した利用者情報の外部への漏えいを防いでいる．しかし，1 つの AP が sensitive_data と sinks を持つ場合，利用者情報の外部への漏えいを防ぐことができないという問題がある．提案手法は，1 つの AP が利用者情報を取得したのちに外部と通信する場合でも利用者情報の漏えいを防ぐことができる．この点において，提案システムは SEAndroid よりも優れているといえる．

5.4.6 trusted AP との通信

SEAndroid は AP をタグで区別していないため，trusted AP の安全性に問題がある．たとえば，untrusted AP がウェブブラウザに攻撃者のサイトにアクセスするインテントを送信した場合，ウェブブラウザを利用してマルウェアをダウンロードするという問題 [11] がある．提案システムは AP を 3 種類に分類して，ユーザがインストールした AP と trusted AP の通信を制御している．これにより，この攻撃による影響を緩和できるため，trusted AP の安全性を高めているといえる．

5.5 オーバヘッド

提案システムのオーバヘッドを**表 11** に示す．制御 AP の起動時間は，パーミッションの利用や AP 間の通信の許可をユーザに求めるときに発生するオーバヘッドである．ボタン入力後の処理時間は，ユーザが制御 AP のボタンをクリックしてから提案システムの処理が終了するまでのオーバヘッドである．ユーザに問合せ済みのパーミッションの処理時間は，ユーザが制御 AP の画面に含まれる「次回以降，結果を反映する」というチェックボックスにチェックを入れて Yes または No を選択したパーミッションを，次回以降に利用する際に発生するオーバヘッドである．制御対象以外のパーミッションの処理時間は，AP が制御対象以外のパーミッションを利用する際に発生するオーバヘッドである．

表 11 提案システムのオーバヘッド
Table 11 System overhead.

測定項目	処理時間
制御 AP の起動時間	2,482 (ms)
ボタン入力後の処理時間	267 (ms)
ユーザに問合せ済みのパーミッションの処理時間	2,415 (μs)
制御対象以外のパーミッションの処理時間	248 (μs)

表 11 より，評価環境ではユーザに情報を提示する際に約 2.5 秒のオーバヘッドがあることが分かった．しかし，表 8 と表 9 を参照すると，制御 AP を起動する頻度は少ないため，それほど大きなオーバヘッドではないといえる．また，ユーザに問合せ済みのパーミッションの処理時間と制御対象以外のパーミッションの処理時間は，どちらも短く，それほど大きなオーバヘッドではないといえる．

6. 関連研究

利用者情報の漏えいを防止する研究として，文献 [12], [13], TaintDroid [14]，AppFence [15]，および MockDroid [16] がある．文献 [12] は，AP に利用者情報に対する参照ポインタのみを渡し，利用者情報に関してはセキュリティマネージャ側で処理することで安全な利用者情報の取扱いを可能にするシステムを提案している．文献 [13] は，パーミッションの組合せや API の使用順番などをもとに利用者情報が漏えいするパターンを作成し，外部との通信を制御する手法を提案している．この手法は，Linux カーネルを変更してパケットを監視することで実現されている．TaintDroid は利用者情報を保持する変数を追跡する機構であり，利用者情報に関連付けられている変数が外部に漏えいする場合，検知する．AppFence は TaintDroid を拡張した機構であり，テイントが付与されている利用者情報が外部に漏えいするときにブロックもしくはダミーデータを外部に送信することで利用者情報の漏えいを防止する．MockDroid 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とで利用者情報の漏えいを防止する．提案手法は，利用者情報を追跡し，外部への漏えいを防止するという観点では AppFence と類似している．AppFence は TaintDroid を利用しており，追跡の粒度が細かいことが特徴である．このため，パーミッションに基づいたタグを利用して利用者情報を追跡している提案手法よりは追跡の粒度が細かいといえる．しかし，AppFence はユーザが AP ごとにダミーデータの設定をする必要があり，ユーザの手間が大きいという問題がある．提案手法は，ユーザが事前に設定する必要がないため，AppFence よりユーザの負担が少ないといえる．また，提案手法は AP が外部に利用者情報を送信するときに動的に制御するため，利用者情報を外部に送信する必要があるときに送信できる点が AppFence と異なる．

AP 間の通信を制御する研究として，文献 [9]，および IPC Inspection [17] がある．文献 [9] は，AP 間の通信の特徴をもとに作成した独自のポリシに基づいて，通信を制御する．この手法は，Android のフレームワークレベルの制御とカーネルレベルの制御を併用することで，インテント経由の通信だけでなく，ファイルシステムやネットワーク経

由の通信を制御できることが特徴である．IPC Inspection はインテント経由の AP 間通信を制御する方式であり，通信先の AP が通信元の AP より多くの権限を持つ場合，通信先の AP の権限を通信元の AP の権限まで削減する方式である．これにより，複数の AP で連携して AP の権限を昇格する攻撃である confused deputy attacks に対処する．インテント経由の AP 間通信を制御するという観点では，提案手法は IPC Inspection と類似している．しかし，IPC Inspection は，通信元の AP の権限を通信先の AP が引き継ぐため，通信先の AP の機能が制限されるという問題がある．提案手法は，AP の動作の制御はユーザが行い，システムがユーザの判断を支援する．これにより，AP の機能を制限しない点が異なる．

AP の動作を動的に制御する研究として，文献 [8] と [18] がある．文献 [8] 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えいする API をフックして，ユーザが API の利用の可否決定をすることで利用者情報の漏えいを防止する．なお，ユーザに提供する情報は，ユーザ ID，パッケージ名，利用者情報を送信する API，時刻，および利用者情報を取得した API の一覧である．文献 [18] は AP が注意すべき動作をした際に，リアルタイムにユーザに通知する機能を持つ．通知はステータスバーで行い，注意すべき動作の履歴をユーザに提供する．ユーザが通知領域をタップした場合は AP の動作を許可し，何もしなかった場合は AP の動作を拒否する．提案手法は AP が制御対象のパーミッションを利用したとき，および制御対象の通信をしたときに AP を動的に制御する．このため，文献 [8], [18] とは制御する対象が異なる．また，提案システムがユーザに提示する内容は（設計 3），および（設計 7）で示したものである．提案手法は，AP のアイコンや通信の内容などの多くのユーザが分かると考えられる情報のみを提示している．このため，関連研究より提案手法の方が Android に詳しくないユーザにとって，利用しやすいといえる．

7. おわりに

SEAndroid の問題点として，Permission Revocation の利用が困難であること，Tag Propagation の誤検知があること，および最小特権のポリシの作成が困難であることを述べた．これらの問題に対処する方法として，SEAndroid の拡張による AP の動的制御手法を提案した．提案手法では untrusted AP がパーミッションを利用するとき，sensitive_data を持つ AP が sinks を持つ AP と通信するとき，およびユーザがインストールした AP が trusted AP と通信するときにユーザに許可を求めることで，AP を動的制御できること，利用者情報の漏えいを防止できること，誤検知による通信の拒否の影響を緩和できること，および trusted AP の安全性を向上させることができることを示した．

評価では，AP 間の通信を制御することで，情報漏えいを防止できることを示した．また，タグに基づいて通信を制御することと通信を制御するときにユーザに許可を求めることにより，SEAndroid による通信の誤検知を緩和できることを示した．さらに AP のアラートの回数はあまり多くないため，ユーザの負担は少ないことを示した．また，従来の SEAndroid と提案システムを比較することで違いを明確にした．さらに，提案システムのオーバヘッドについて述べ，潜在的なオーバヘッドは少ないことを示した．なお，本システムを利用することでアプリ内の広告モジュールの動作のみを制限した場合，アプリ提供と開発のエコシステムが回らなくなる可能性に留意する必要がある．

今後の課題として，利用者情報の漏えいの誤検知の削減，外部と通信するパーミッションにおける提示手法の実現，およびパーミッションを必要としないユーザ入力情報やクリップボード経由の情報漏えいなど提案手法で検知できない情報漏えいへの対処がある．

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] Google: Android, available from ⟨http://www.android.com/⟩ (accessed 2012-11-30).
[2] Google: Google Play, available from ⟨https://play.google.com/store⟩ (accessed 2012-11-30).
[3] Zhou, Y. and Jiang, X.: Dissecting Android Malware: Characterization and Evolution, *Proc. 33rd IEEE Symposium on Security and Privacy*, Oakland (2012).
[4] National Security Agency: Welcome to the National Security Agency, available from ⟨http://www.nsa.gov/⟩ (accessed 2012-11-30).
[5] NSA: SEAndroid - SELinux Wiki, available from ⟨http://selinuxproject.org/page/SEAndroid⟩ (accessed 2012-11-30).
[6] NSA/CSS: Security-Enhanced Linux, available from ⟨http://www.nsa.gov/research/selinux/⟩ (accessed 2012-11-30).
[7] The Case for Security Enhanced (SE) Android: Android Builders Summit 2012 (2012). available from ⟨https://events.linuxfoundation.org/images/stories/pdf/lf_abs12_smalley.pdf⟩ (accessed 2012-11-30).
[8] 奥田健嗣，中務 亮，山内利宏：Android における情報伝搬の追跡と情報漏洩防止手法の提案，情報通信システムセキュリティ研究会（ICSS），電子情報通信学会研究報告，Vol.111, No.495, pp.5–10 (2012).
[9] Bugiel, S., Davi, L., Dmitrienko, A., Fischer, T., Sadeghi, A.R. and Shastry, B.: Towards Taming Privilege-Escalation Attacks on Android, *Proc. 19th Annual Network and Distributed System Security Symposium* (*NDSS*) (2012).
[10] 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 and Distributed System Security Symposium* (*NDSS*) (2011).
[11] Nils: The Risk you carry in your Pocket, available from

⟨https://media.blackhat.com/bh-ad-10/Nils/Black-Hat-AD-2010-android-sandcastle-slides.pdf⟩ (accessed 2012-11-30).
[12] 上松晴信，可児潤也，名坂康平，川端秀明，磯原隆将，竹森敬祐，西垣正勝：Android OS におけるマスカレーディングポイントを用いたプライバシー保護，情報処理学会研究報告，Vol.2012-CSEC-57, No.18, pp.1–6 (2012).
[13] 葛野弘樹：Android アプリケーションに対する情報フロー制御機構の提案，コンピュータセキュリティシンポジウム（CSS2011）論文集，Vol.2011, No.3, pp.155–160 (2011).
[14] 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, *9th USENIX Symposium on Operating Systems Design and Implementation* (*OSDI*) (2010).
[15] 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* (*CCS*) (2011).
[16] 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*, pp.49–54 (2011).
[17] Felt, A.P., Wang, H.J., Moshchuk, A., Hanna, S. and Chin, E.: Permission re-delegation: Attacks and defenses, *20th USENIX Security Symposium* (2011).
[18] 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.138–145 (2012).

矢儀 真也

2011 年岡山大学工学部情報工学科卒業．2013 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，2012 年度情報処理学会論文賞受賞．電子情報通信学会，ACM，USENIX 各会員．