Vol.2012-CSEC-59 No.8
2012/12/14

スマートフォンアプリ向け独自 ID の生成・管理

竹森敬祐[†1]　松井利樹[†1]　磯原隆将[†1]　川端秀明[†1]　渡辺龍[†1]　窪田歩[†1]

[†1] KDDI 研究所

多くのスマートフォン（以降、スマホ）向けアプリケーション（以降、アプリ）に、端末や利用者に結びつく固定的なグローバル ID を外部送信する情報収集モジュールが組み込まれている。これは、スマホのアプリにおいて、既存の Web ブラウザ向け第三者 cookie の仕組みを適用し難い背景からである。主に、利用者の嗜好に合わせたターゲティング広告のための閲覧履歴を管理するキーとして利用されており、プライバシ侵害の懸念が拡がっている。

本稿では、スマホ端末の殆どに実装されている共有メモリを使って、忘れられる権利と名寄せの脅威を排除した、独自 ID（UUID: Unique User Identifier）の生成・管理の手法を提案する。これは、サーバ側で生成した UUID に変数を加えて暗号化した後に、スマホ端末に配信する手法である。この暗号化された UUID を取り出せるのは発行元のサーバのみであることと、利用者の操作で UUID の削除や送信停止の制御を行うことも実現する。

Management Techniques for Universally Unique Identifier for Applications in Smart Phone

Keisuke TAKEMORI[†1]　Toshiki MATSUI[†1]　Takamasa ISOHARA[†1]
Hideaki KAWABATA[†1]　Ryu WATANABE[†1]　Ayumu KUBOTA[†1]

[†1]KDDI R&D Laboratories Inc.

Many applications in a smart phone include information collectors that send global IDs, i.e., device ID and SIM serial number, to service servers. In case of an advertisement module, the global IDs are used for management keys of web-view history, which track a user interests. The information collectors in the applications are difficult to share the third party cookies managed by the web browser.

In this research, we propose the sharing mechanisms for a universally unique identifier (UUID) using a shred memory implemented in the smart phone. It achieves both the right to be forgotten and removing the threats of ID aggregations. The service servers generate UUID and encrypt it with a variable parameter, which is sent to the application. In addition, deletion and suspension mechanisms for the UUID managements are implemented in the information collector.

1. はじめに

スマホ向けの無料アプリを中心に、利用者の閲覧履歴を使った行動ターゲティング広告ビジネスが盛んに行われている。これは、アプリに組み込んだ広告枠を指し、そこに表示された広告のクリック履歴を、利用者を識別する ID をキーに管理しておき、興味を持っていそうな分野の広告を適切に提示するというものである。アプリ開発者が、広告配信事業者の提供する広告モジュールをアプリに組み込むことで実現している。広告配信事業者には広告主からの報酬が、アプリ開発者には広告配信事業者からの配当が支払われるため、利用者は無料でアプリを楽しめるエコシステムが回っている。

この広告配信ビジネスは否定されるものではないが、利用者などからプライバシ侵害を懸念する声が上がっている。その理由は、広告配信事業者側が利用者の嗜好を把握するための閲覧履歴を管理するキーとして、端末識別子（IMEI）や SIM 識別子（IMSI）、電話番号、OS・プラットフォームが生成する ID などを用いている為である。これらはグローバル ID と呼ばれ、端末や契約者に結び付く固定的な ID である。

グローバル ID を使う問題の本質は二つある。まず一つが、固定の ID であるが故に、広告配信事業者側に蓄積された履歴を利用者が無効化できないという問題である。広告配信事業者にとって履歴を集めるほど、個人の嗜好が分かり精度の高い広告配信ができる一方、利用者にとってはプライバシに関わる情報が蓄積していくという不安がある。プライバシを守るためには、簡単に ID を削除／変更できるような利用者関与の機会が提供される必要がある。欧州では、閲覧履歴などを利用者側から無効化できるようにする「忘れられる権利」が認められており、世界的にもこの権利を認める方向で議論が進んでいる。もう一つの問題が「名寄せの脅威」である。端末に複数のアプリ・複数の事業者の情報収集モジュールが組み込まれている状況において、異なる広告配信事業者が同じグローバル ID を使って閲覧履歴を管理することになる。広告配信事業者同士が収集した閲覧履歴を、同じグローバル ID をキーに共有した場合、名寄せができてしまう。こうした名寄せは企業倫理上、許されていない。

そこで本稿では、スマホ端末の殆どに実装されている共有メモリを使って、忘れられる権利と名寄せの脅威を排除した、独自 ID（UUID: Unique User Identifier）の生成・管理の手法を提案する。これは、サーバ側で UUID を生成し、これに変数を加えて暗号化した後に、スマホ端末に配信する手法である。この暗号化された UUID を取り出せるのは、発行元のサーバのみであり、利用者の操作で UUID の削除や送信停止の制御を行うこともできる。

2 章では、スマホやそのアプリに特有な問題と課題を整理する。3 章では、既存の Web ブラウザ向け広告における利用者履歴の管理に用いられる第三者 cookie の仕組みについて復習する。4 章では、スマホ向けアプリ内広告における第三者 cookie に代わる UUID の生成・管理の仕組みについて、考察を経ながら安全な手法の一例を提案する。5 章では、本技術の安全性や残課題について考察する。最後に 6 章で、波及効果と今後の期待を述べて、まとめとする。

2. 問題と課題の整理

図 1 に、スマホアプリ向け広告による利用者情報の収集と行動ターゲティング広告の様子を示す。ここで利用者情報とは、端末や利用者に結びつくグローバル ID や、位置やアドレス帖などのセンシティブデータなどを含む[1]。以下、図 1 を用いて、スマホ向けアプリ特有の問題を列挙し、取り組むべき課題を整理する。

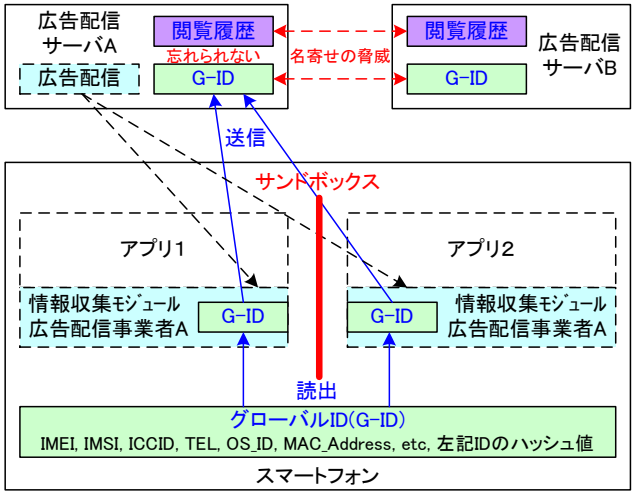

図 1　グローバル ID を用いた閲覧履歴の管理

Figure 1　Page view management using global ID

2.1 忘れられる権利

スマホ OS の特徴として、アプリをサンドボックスと呼ばれる隔離環境で実行させることで、汎用の PC 向け OS に比べてマルウェアの影響を受け難いアーキテクチャを持っている。その反面、従来から行われてきた Web ブラウザ向け広告で用いられている第三者 cookie をアプリ間で共有し難い問題が生じている。このため、行動ターゲティング広告向けの情報収集モジュールの多くは、利用者を特定し、その閲覧履歴を管理するキーとして、全てのアプリから標準的な API を使って取得できるグローバル ID を使う傾向にある。固定値であるグローバル ID をキーに広告配信事業者側に閲覧履歴などが蓄積された場合、利用者が ID を取り替えられないことから、忘れられない問題がある。

2.2 名寄せの脅威

グローバル ID には、固定的な ID である特徴に加えて、世界で唯一の値が割り振られた ID という特性も持つ。

異なる事業者の情報収集モジュールが、同じグローバル ID をキーに閲覧履歴を蓄積していた場合、この ID をキーに閲覧履歴の交換を行えることになる。このため、グローバル ID をキーとした名寄せの脅威も指摘されている。

2.3 オプトアウトの適用

サンドボックスの壁は、任意のアプリに組み込まれた情報収集モジュール間で、状態共有を難しくしている。例えば、利用者がグローバル ID の送信や閲覧履歴の蓄積の停止を制御するオプトアウトについて、あるアプリ内の情報収集モジュールから設定を行ったとしても、他のアプリ内の同事業者の情報収集モジュールにその設定状況を伝えることが難しい。このため、各アプリに含まれる情報収集モジュールごとにオプトアウトの設定をすることになる。この煩雑さ故に、現状の情報収集モジュールの多くが、オプトアウト制御の機能を持っていない。

2.4 透明性の確保

サンドボックスによる安全設計は、端末を操作する利用者やセキュリティ対策ソフトなどから、アプリの挙動が見え難いという問題も生じさせている。現在のスマホ向け OS において、アプリの挙動をモニタするための標準的な API は殆ど提供されていない。

よって、利用者情報を外部に送信するなどのプライバシに関わるアプリの処理について、利用者に事前に説明する透明性の確保が求められている。

2.5 情報収集モジュールの実態調査

KDDI 研究所では、2012 年 4 月に入手したスマホ向けアプリ 100 件について、利用者情報の送信の有無、アプリのプライバシポリシの有無などの調査を行った（表 1）。過去の調査結果は、[2]を参考にされたい。

表 1　アプリや情報収集モジュールによる送信状況

Table 1　Investigation results of information sending by application or information collector.

送信状況	件数
アプリ本体を含め、利用者情報を送信するアプリ数	81/100 件
適切な説明や手続きによる承諾を経て[1]、利用者情報を外部送信しているアプリ数（白）	2/100 件
適切な説明や手続きを経ないまま[1]、利用者情報を外部送信しているアプリ数（灰）	13/100 件
何ら説明のないまま、勝手に利用者情報を外部送信しているアプリ数（黒）	66/100 件

表 1 における、”白”とは、[1]や[3]を参考に、プライバシポリシに記載されるべき 8 項目の有無や事前の承諾（オプトイン）など全てを満たしており、かつ実際に送信される情報との整合性があるものを指す。”灰”とは、8 項目や承

諾過程のいずれかに欠如があるか、実際に送信される情報との不整合が生じているものを指す。”黒”とは、利用者に説明のないまま、勝手に利用者情報を外部に送信しているものを指す。

表 1 より、81 件ものアプリが利用者情報を外部送信し、そのうち”白”は 2 件しかない状況にある。情報送信を行うアプリ(81 件)の 8 割が”黒”であり、アプリから利用者に知らされることなく利用者情報の送信が行われている現状がある。”灰”や”黒”の殆どが、アプリに組み込んだ情報収集モジュールの挙動を考慮していないか、挙動を完全に把握できていないために生じている。

尚、81 件のアプリからは、70 種類の情報収集モジュールが抽出され、そのうち広告系のモジュールは、65 件（93%/70 件）、クラッシュレポートや起動確認用のモジュールは、5 件（7%/70 件）であった。

2.6 情報収集モジュールの組み込み

70 種類の情報収集モジュールのうち、67 種類は.jar 形式のファイルとして提供されていた。.jar 形式とは、コンパイルされた複数の Java バイトコードや画像などのリソースを一つにまとめ、ZIP 形式で圧縮した java アーカイブファイルである。情報収集モジュールの提供者は、この.jar 形式のファイルを含むソフトウェア開発キット（SDK）を公開し、アプリ開発者は、SDK を利用して、.jar 形式のファイルを、外部ライブラリとしてアプリにインポートして開発を行う。尚、広告用の情報収集モジュールの多くには、webview を使った広告取得用ライブラリが含まれている。

図 2 に、Android OS 向けの情報収集モジュールをアプリにインポートした後に、アプリ開発者がマニフェストファイルに追記するコードを示している。インターネットへのアクセス権限、IMEI や TEL などのグローバル ID を読み取る権限、位置情報を読み取る権限、オプションとしてアドレス帖を読み取る権限などの追記が要求されている。

```
Step 1 (required) –

You must set your application to allow the following permissions:
<uses-permission android:name="android.permission.VIBRATE"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>

Step 1a (optional) -

Include the following permission(s) to take advantage of future planned releases
Notifier and/or increase potential publisher revenue share!

<uses-permission
android:name="android.permission.READ_CONTACTS"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>

Step 2 –

Add the following code to Android.XML
```

図 2　アプリに情報収集モジュールを組み込む手順の例
Figure 2　Example of implementation procedure for information collector in smart phone application.

情報収集モジュールの特性を考慮しないままアプリにインポートし、マニフェストファイルを機械的に編集するという、プライバシへの配慮を怠りがちなアプリ開発者でも、安全に利用者の閲覧履歴を管理できるプライバシ・バイデザインに沿った ID の仕組みが必要とされている。

2.7 進み始めた OS によるサポート機能

iOS6 からは、広告モジュール向けの UUID を発行する API が用意されている[4]。

- ASIdentifierManager.advertisingIdentifier

広告モジュールがこの API を実行すると、UUID が実行元の広告モジュールに返される。端末の設定画面にも、”Ad Tracking を制限する”という項目が追加され、プライバシ保護に向けた機能を OS 側がサポートし始めている。

但し、異なる広告配信事業者の広告モジュールに対して、同じ UUID を返すため、名寄せの脅威が指摘されている。

3. Web ブラウザ向け広告の安全策

そもそも、こうした問題が発生し得るのは、スマホ向けアプリに組み込まれた広告モジュールにおいて、Web ブラウザのクッキ（cookie）のような仕組みを持たせられないためである。Web ブラウザ向け広告では、第三者 cookie と呼ばれる端末や利用者に結び付かないランダムな値を、Web サーバからブラウザに配信して、サイトを跨る閲覧履歴の管理に使っている（図 3）。但し、FireFox と Internet Explore のように、ブラウザ･アプリが異なる場合には、ブラウザ･アプリ間で cookie の共有はできない。

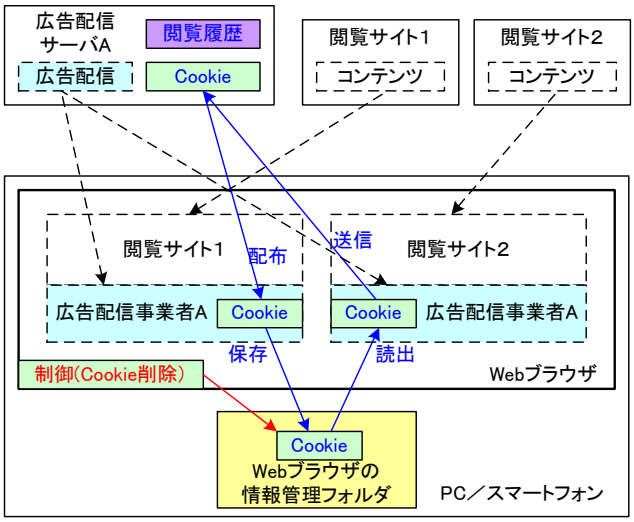

図 3　Web ブラウザ向け行動ターゲティング広告の仕組み
Figure 3　Targeting advertisement scheme for web browser

Cookie は、ブラウザが管理する一時フォルダに保存されるが、利用者側でこれを削除することができる。削除により、広告配信サーバ側で cookie をキーに管理していた閲覧履歴も無効化される。これにより、忘れられる権利が成り立っている。また、Web ブラウザの仕様として cookie は発

行元ドメインにしか返信しない設計になっている。広告配信事業者ごとにランダムな値として生成される cookie を使って名寄せを行うこともできない。同じ広告配信事業者が異なる複数の Web サイトをまたがって利用者の閲覧履歴を管理することはできる。これは、複数の Web サイトに、同じ広告配信事業者の広告枠が組み込まれている場合に、利用者がそれぞれのサイトを閲覧した際に、発行しておいた cookie が広告配信事業者に送られるためである。

スマホ向けアプリに組み込まれた多くの広告モジュールは webview を使っているが、サンドボックスによって発行される cookie をアプリ間で共有するのが困難な問題がある。このため、広告配信事業者はどのアプリからでも取得が容易なグローバル ID を使ってしまう傾向にある(図 1)。

4. UUID の生成・管理手法

本章では、情報収集モジュールとして、広告モジュールを例に、安全性の高いとされる UUID の生成・管理手法について考える。以下、4.3 節と 4.4 節に、安全度の高い手法を提案するが、この 2 つのモデルに至るまでに考えられた、安全度はやや劣るが既存システムとの親和性の高い手法についても 4.1 節と 4.2 節に記載しておく。

4.1 共有メモリ利用型

【効果】忘れられる権利 ○、名寄せの脅威 ×

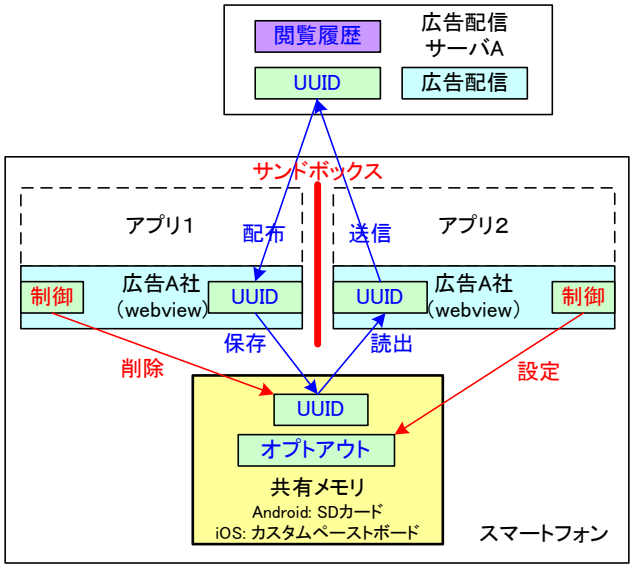

図 4　共有メモリを使う UUID とオプトアウト状態の共有
Figure 4　Sharing mechanism of UUID and opt-out flag using shared memory.

端末メーカの実装に依存するが、昨今の Android 端末には Internal SD カードが標準で組み込まれることが多くなってきた。利用者によって External SD カードを挿入するケースも多い。SD カードは、全てのアプリから読み取り（Read）アクセスできる共有メモリとして活用できる。また、android.permission.WRITE_EXTERNAL_STORAGE のパーミッションを宣言して、利用者に承認されたアプリは、SD カードに保存（Write）アクセスも可能である。iOS については、カスタムペーストボードという機構があり、こちらも全てのアプリから読み取り（Read）、保存（Write）アクセスでき、共有メモリとして活用できる。これらの特性から、SD カードやカスタムペーストボードを共有メモリとして使う手法を図 4 に紹介する。

4.1.1 処理の流れ

(1)広告配信サーバは、端末側の広告モジュールから UUID がセットされていないアクセスを受けると、UUID を生成する。

(2) サーバは生成した UUID を広告コンテンツと共に、端末側の広告モジュールに配布する。

(3) 端末側の広告モジュールは受け取った UUID を SD カードやカスタムペーストボード上の共有メモリに保存する。

(4) ここで、別アプリ内の広告モジュールが起動した際に、共有メモリに保存されている UUID を読み出し、広告配信サーバに送信する。

(5) 広告配信サーバは受け取った UUID から過去の広告閲覧履歴などを参照し、適切な広告コンテンツを選択する。

上記に加えて、利用者の操作で UUID を削除する機能を設ける。また、UUID の送信停止を指示するオプトアウトフラグも共有メモリで管理することで、アプリを跨ぐ同事業者の広告モジュールがオプトアウト状態を共有できる。

4.1.2 考察：忘れられる権利

広告モジュールが表示する枠内に、「i」マークと呼ばれる UUID の削除や送信停止を設定するためのオプトアウトの導線を設けておく（図 5）。

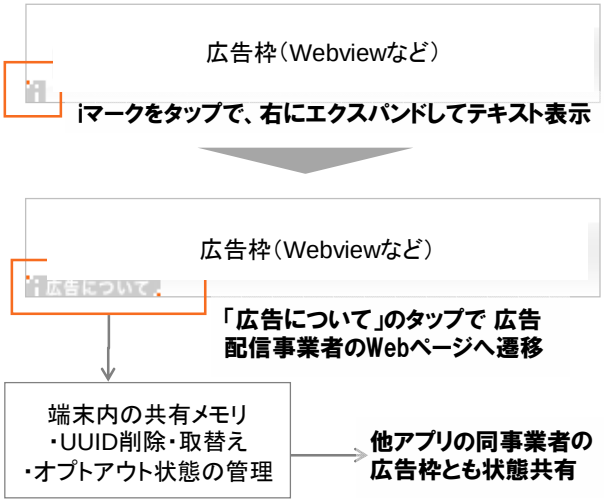

図 5　オプトアウトの導線の実装例
Figure 5　Implementation of opt-out manager.

「i」マークの導線を経て UUID の削除が指定されると、広告モジュールが共有メモリに保存されている UUID を削除する。これにより、広告配信サーバへアクセスする際に、UUID を持たないアクセスとして、広告配信サーバ側から

新たな UUID が発行される。過去の UUID で蓄積されていた閲覧履歴は無効になる。また、送信停止というオプトアウトが指定されると、UUID の送信を停止する。以後の広告モジュールから広告配信サーバへのアクセスには、UUID の送信停止状態を通知するフラグを埋め込み、新たな UUID の発行も停止させる。

4.1.3 考察：名寄せの脅威

UUID を共有メモリに保存する本方式は、他社の広告モジュールからもアクセスできる。保存した UUID を複数の広告配信事業者が用いてしまうと、蓄積された情報の交換を行えるようになり、名寄せの脅威が生じる。

4.1.4 考察：既存システムとの親和性

広告モジュールの実装においては、標準的な API などを用いて開発でき、特別なプログラミングは不要である。

広告配信サーバ側での UUID の発行についても、Web ブラウザ向け cookie 発行機構と同じものを流用できる。

よって、既存の広告配信システムとの親和性は高い。

4.2 Content Provider 利用型：Android 端末向け

【効果】忘れられる権利 ○、名寄せの脅威 ×

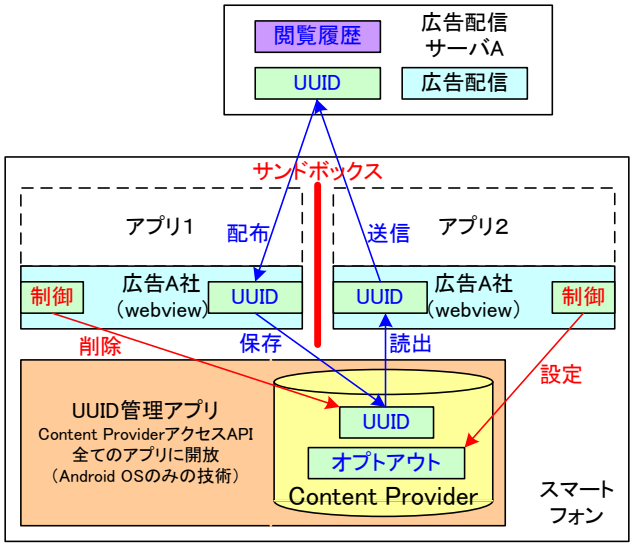

図 6　Android OS 向けの Content Provider を使った UUID とオプトアウト状態の共有（iOS は対象外）

Figure 6　Sharing mechanism of UUID and opt-out flag using content provider for Android OS.

Android 端末では、アプリが情報を管理・共有するための Content Provider と呼ばれるデータベース（DB）が備えられている。この Content Provider を UUID の保存・共有に使う手法が考えられる（図 6）。

Content Provider に格納するデータは、Content Provider を有するアプリでのみ使えるようにしたり、同じ開発者署名を持つアプリ間で使えるようにしたり、全てのアプリから使えるように、設定できる。様々な開発者が作成するアプリに埋め込まれた広告モジュールにおける情報共有の場合、開発者によって施される署名も多様になるため、アクセス制限を施せない。よって、Content Provider に格納した UUID やオプトアウトフラグを全てのアプリに対して開放する実装になる。

4.2.1 処理の流れ

事前準備：UUID やオプトアウト状態を Content Provider で管理する専用のアプリを通信事業者や端末メーカなどが端末にプリインストールしておくか、もしくは利用者が別途入手してインストールしておく。

UUID の発行や端末への配布、端末からの送信などの基的な仕組みは、4.1 節と同じである。

（1）端末は受け取った UUID を、UUID 管理アプリの Content Provider に保存する。

（2）ここで、別アプリにある同広告配信事業者のモジュールが起動した際に、UUID 管理アプリの Content Provider にアクセスし、保存されている UUID を読み出す。

4.2.2 考察：忘れられる権利

4.1 節と同じく「i」マークを設けておき、UUID の削除が指定されると、Content Provider に保存されている UUID を削除する。また、送信停止が指定されると、UUID の送信を停止するフラグを Content Provider に書き込んでおく。

4.2.3 考察：名寄せの脅威

UUID を管理する Content Provider は、全てのアプリからアクセスできる。保存した UUID を広告用 ID として複数の事業者が用いてしまうと、蓄積された情報の交換を行えるようになり、名寄せの脅威が生じる。

4.2.4 考察：既存システムとの親和性

Content Provider は Android プラットフォームの基本機能である。そのため、広告モジュールのプログラミングのみで対処可能である。

広告配信サーバ側での UUID の発行についても、Web ブラウザ向け cookie の発行機構と同じものを流用できる。

よって、既存の広告配信システムとの親和性は高い。

但し、UUID 管理アプリを、端末メーカや通信キャリアがプリインストールしておく場合や、利用者が別途インストールする場合には、その手間を要する。

4.3 提案：暗号化された UUID の共有

【効果】忘れられる権利 ○、名寄せの脅威 ○

前記の二つの方式は、いずれも名寄せの脅威を排除できない。そこで、広告配信事業者のサーバ側で発行する UUID に暗号化を施し、他の広告配信事業者の広告モジュールが UUID を取り出せないようにする手法を提案する（図 7）。

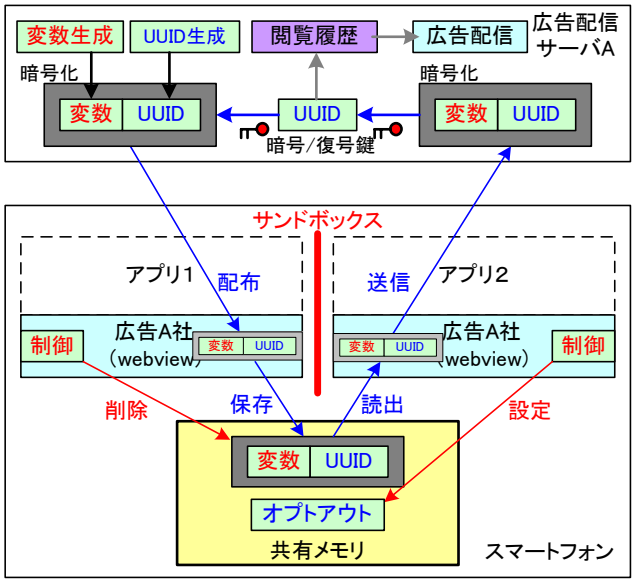

図 7　暗号化された UUID の共有

Figure 7　Encrypted UUID sharing mechanism.

広告コンテンツを配信するたびに、UUID に時刻などの変数を加えて暗号化を施すことで、配信・保存されるデータを毎回変化させる。この変化する暗号化データについて、中にある UUID を取り出せない他の広告事業者のモジュールは、利用者特定のための ID として利用できなくなる。尚、暗号された UUID を格納する場所として、Android の場合は SD カードでも Content Provider でも構わない。iOS の場合は、カスタムペーストボードになる。

4.3.1 処理の流れ

（1）広告配信サーバは、端末側の広告モジュールから UUID がセットされていないアクセスを受けると、UUID を生成する。

（2）この UUID に時刻などの変数を加える。

（3）広告配信サーバ側で安全に管理された鍵で、（2）で生成した値を暗号化する。

（4）暗号化されたデータを広告コンテンツとともに、端末側の広告モジュールへ配布する。

（5）広告モジュールは受け取った暗号化データ（変数＋UUID）を、共有メモリに保存する。

（6）ここで、別アプリ内の広告モジュールが起動した際に、共有メモリに保存されている暗号化データ（変数＋UUID）を読み出し、広告配信サーバに送信する。

（7）広告配信サーバは受け取ったデータを復号し、変数と UUID を取り出し、UUID のみを抽出する。

（8）取り出した UUID に関連付けられた過去の閲覧履歴を参考に、適切な広告コンテンツを選択する。

（9）（2）－（5）の処理に戻り、再び UUID に変数を加えて暗号化し、これと選択した広告コンテンツを端末側の広告モジュールに配布する。

4.3.2 考察：忘れられる権利

4.1 節の手法と基本的には同じであり、利用者から UUID の削除が指定されると、広告モジュールから共有メモリに保存されている暗号化（変数＋UUID）を削除する。これにより、広告配信サーバへアクセスする際に、UUID を持たないアクセスとして、広告配信サーバ側から新たな UUID が発行され、過去の UUID で蓄積されていた閲覧履歴は無効化される。

4.3.3 考察：名寄せの脅威

共有メモリに保存される暗号化（変数＋UUID）は、広告配信サーバにアクセスするたびに変化する。UUID は例え同事業者の広告モジュールであったとしても、端末側で復号して取り出すことができない。よって、他のアプリからこの値を用いようとしても、ID としての役割をなさず、名寄せの脅威を排除できる。

4.3.4 考察：既存システムとの親和性

広告モジュールの実装においては、標準的な API などを用いて開発でき、特別なプログラミングは不要である。

広告配信サーバ側における UUID の発行において、既存の cookie 発行機構に、変数の付記や、暗号化／復号化の処理を加える必要がある。

よって、既存の広告配信システムとの親和性はやや劣る。

4.4 提案：暗号化された UUID の共有

【効果】忘れられる権利 ○、名寄せの脅威 ○

4.3 節の手法では、共有メモリに保存したオプトアウトフラグが Write 権限を持つアプリから簡単に書き換えられてしまう問題がある。そこで、同一の広告配信事業者からしかアクセスできない暗号化フォルダを作り出し、オプトアウトフラグをそこで管理する手法を追加する（図 8）。

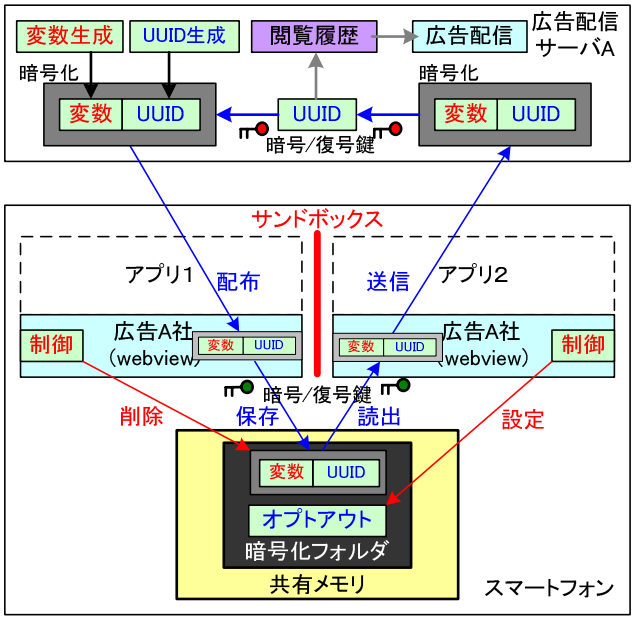

図 8　暗号化フォルダによるオプトアウト状態の管理

Figure 8　Sharing mechanism of opt-out flag using encrypted shared folder.

尚、本手法は広告モジュールにプログラムの難読化が施されていることを前提にする。但し、ソフトウェアレベルでの難読化の強度は高いとは言えないことに注意されたい。

4.4.1 処理の流れ

UUID の発行や暗号化して配信／返信する基本的な仕組みは 4.3 節と同じである。以下に、暗号化フォルダを使って、発行された UUID やオプトアウトフラグを管理する手順を説明する。

(1) 広告モジュールは、起動するたびに端末バインドな暗号化／復号化を行う鍵を作り出し、これを用いて受け取った UUID やオプトアウト状態を、共有メモリに保存する。

(2) ここで、別アプリ内の広告モジュールが起動した際も、(1)と同じアルゴリズムで端末バインドな鍵を作り出し、この鍵を用いて暗号化フォルダに保存されている UUID やオプトアウトフラグの読み取りを行う。

4.4.2 考察：忘れられる権利

4.1 節の方式と基本的には同じであり、利用者から UUID の削除が指定されると、暗号化フォルダで管理される UUID を削除することで、広告配信サーバに蓄積された閲覧履歴が無効化される。

4.4.3 考察：名寄せの脅威

4.3 節の方式と同じであり、暗号化データ（変数＋UUID）が同事業者の広告モジュールが広告配信サーバにアクセスするたびに変化するため、名寄せのための ID として用いることができない。

4.4.4 考察：既存システムとの親和性

広告モジュールに必要な要素として、暗号化フォルダを生成し安全にアクセスする仕組みを秘匿化する難読化の技術である。この技術は各事業者が秘匿して持つものであるが、暗号化フォルダを生成するバイナリファイルを SDK として広告配信事業者に提供するスキームも可能である。

広告配信サーバ側の処理は、4.3 節と同じである。

よって、既存の広告配信システムとの親和性は 4.3 節の方式よりもやや劣る。

5. 考察と残課題

5.1 忘れられる権利の実現

UUID は、端末や利用者に結びつかないランダムな値であり、利用者が任意のタイミングで削除できる機能を設けることで、サーバ側に UUID をキー蓄積された閲覧履歴が無効化される。

しかし、UUID を削除するだけでは過去の UUID とそれにリンクして蓄積された閲覧履歴は、サーバ側に残っている。スマホがマルウェアに感染している場合や、通信をキャプチャされるなどで UUID が漏洩した場合に、その UUID を他の端末からサーバに送ることで、過去のプライバシに関連する情報が漏洩する懸念がある。忘れられる権利としては十分とは言えない。そこで、より安全性を考慮すると、端末上から UUID を削除するだけでなく、サーバ側にも削除対象になった UUID にリンクする閲覧履歴を削除する機能を設ける必要がある。

5.2 名寄せの脅威の排除

4.3 節と 4.4 節の方式は、広告配信サーバ側で管理された暗号鍵を利用して暗号化された UUID を配信する方式であり、その強度は AES や Kcipher2 などの採用される暗号アルゴリズムに帰着する。

従来の Web ブラウザ向け cookie では、マルウェアに感染していない状態や、ユーザによる誤操作がない状態で、正しく cookie が発行元ドメインに返信される仕組みであった。これに比べて 4.3 節、4.4 節の方式は、たとえ端末がマルウェア感染している場合でも、UUID の漏洩による名寄せの脅威を排除できる強度がある。

5.3 透明性の確保

スマホはアプリの挙動が見えにくい特性があるが故に、たとえ安全性の高い UUID とはいえども、UUID の送信の事実、送信先、利用目的、オプトアウト制御の手法などを、利用者に事前説明することが望ましい。

そこで、情報収集モジュールをアプリ開発者へ提供する際には、扱う情報や目的などをアプリのプライバシポリシとして利用者に知らせるように依頼する。これにより、情報収集モジュールを組み込んだアプリの透明性を高めることができる。

5.4 Android OS の場合の必要なパーミッション

従来からの Android ID を除くグローバル ID を読み取る情報収集モジュールは、

- Android.permission.READ_PHONE_STATE

が必要であった。利用者は、IMEI、IMSI、電話番号などを読み取る権限を承諾する気持ち悪さがあった。

4.1 節～4.4 節の手法では、上記のパーミッションに代わって、SD カードに UUID を保存する権限が必要になり、

- Android.permission.WRITE_EXTERNAL_STORAGE

がマニフェストファイルに追加される。このパーミッションは、SD カードへの書き込み権限を与えるのみであり、前者の電話番号などを含むグローバル ID の読み取り権限よりも、不安要素は小さい。

5.5 残課題：UUID とオプトアウトフラグの勝手な削除

UUID やオプトアウト状態を共有メモリに保存する 4.1 節－4.4 節の手法では、他のアプリから勝手に削除される懸念がある。特に、オプトアウトフラグを削除されると、UUID の送信停止状態が解除されてしまい、再び UUID と閲覧履歴が蓄積されてしまう。

よって、こうした削除アプリをマルウェアと判定して駆除するマルウェア対策ソフトによる補強が望まれる。特に、4.4 節のように、たとえ共有メモリといえども、読み取りを拒否する仕様の暗号化フォルダを削除するアプリは、マルウェアと判定すべきである。

5.6 残課題：なりすましへの懸念

配信や送信される UUID をネットワーク上でキャプチャされて、他の端末から再送された場合、なりすましを行える。これは、Web ブラウザ向けの第三者 cookie であっても、ネットワーク上でのキャプチャによるなりすましが生じる問題と同じである。この対策として、UUID の送受信に、https などの暗号化通信路を利用することで、ネットワーク上でのキャプチャによるなりすましを防止できる。

マルウェア感染した端末から UUID が漏洩することで、なりすまされる懸念がある。これについては、マルウェア対策ソフトで、UUID を漏洩するアプリを駆除する補強が望まれる。

5.7 残課題：UUID とオプトアウトフラグの守るべきレベルの議論

Web ブラウザ向け第三者 cookie は、ブラウザの仕様として発行元ドメインにしか返信されないが、PC がマルウェア感染してしまえば第三者 cookie は漏洩し、これを取得した他の広告配信事業者から同じ値の cookie を配信することもできる。利用者もファイル操作の中で、第三者 cookie を自由にコピーや外部送信できる。前提は、マルウェア感染していない状態においては、Web ブラウザ向け第三者 cookie は安全ということになっている。

4.1、4.2 節では、既存の Web ブラウザ向け広告配信の仕組みをそのまま活用できる親和性に優れる反面、他の事業者の情報収集モジュールも UUID やオプトアウトフラグを共有できる問題があった。広告配信事業者のモジュールが保存した UUID やオプトアウトフラグを、勝手に利用する他の情報収集モジュールもマルウェアとみなすことができた場合、Web ブラウザの第三者 cookie の仕組みと同程度の安全度と言えるのではないか。

一方で、共有メモリという全てのアプリから読み出しできる領域に UUID とオプトアウトフラグを保存するという行為に対して、他のアプリが勝手に利用してもマルウェアとは判定しきれないという見解もありうる。そこで、4.4 節では、UUID やオプトアウトフラグを共有メモリに保存する際に、アプリ内に含まれる同一事業者の情報収集モジュールからしか読み出せない仕組み持たせた。この対策を越えて、読み出そうとするアプリは、マルウェアと判定されるべきである。

5.8 残課題：処理速度の測定

4.3 節と 4.4 節の 2 つの手法では、広告配信サーバ側での暗号化／復号化の処理負荷を測定する必要がある。また、4.4 節の手法では、端末側においても UUID やオプトアウトフラグを暗号化フォルダで管理するための鍵の生成、暗号化／復号化の処理負荷を測定する必要がある。これらは、今後の課題とする。

6. 今後への期待

現在、アプリ内情報収集モジュールによるグローバル ID の勝手な送信が問題視される中で、法的側面を重視する事業者、主に日本の広告配信事業者は、プライバシ侵害の恐れのあるグローバル ID による嗜好に関わる情報の蓄積を避ける傾向にある。日本の広告モジュールの多くは、嗜好を加味しないランダム広告になっている。このため、ターゲティングを適切に行えないが故に、広告主からの報酬が少なくなり、アプリ開発者への配当も減る。これにより、アプリ開発者から選ばれない傾向にある。アプリ開発者が組み込む情報収集モジュールの多くは、海外製のプライバシ保護策のない高配当なモジュールばかりになっており、利用者のプライバシ侵害への不安が高まるという悪循環に陥っている。また、アプリ配信 Market におけるアプリ審査に着目すると、多数のアプリに情報収集モジュールが組み込まれて利用者情報が外部送信される中で、悪意の情報漏洩と、情報収集モジュールによる正当な情報送信を見分けることが難しい問題がある。本方式の普及により、アプリ本体や情報収集モジュールがグローバル ID を送信する正当な理由が成り立たなくなり、グローバル ID を送信する行為の低下に繋がるのではと考えている。以下、普及による期待を列挙しておく。

- 利用者のプライバシ保護の促進
- アプリの意図と合わないパーミッションを要求する不安を感じるアプリの減少
- 安全な広告ビジネスの活性化
- アプリの審査者によるマルウェア検知の精度の向上

今後は、本技術の普及に努めていき、スマホのアプリ市場全体の信頼性の向上に繋げることで、皆様に安心・安全なスマホ・アプリの利活用のシーンを提供していきたい。

参考文献

1) 総務省,“スマートフォン プライバシ イニシアティブ”, http://www.soumu.go.jp/menu_news/s-news/01kiban08_02000087.html
2) 竹森敬祐,“スマートフォンからの利用者情報の送信”、総務省 HP、2012 年 1 月. http://www.soumu.go.jp/main_content/000143966.pdf
3) MCF,“スマートフォンの利用者情報に関する連絡協議会”, http://www.mcf.to/press_comment/pdf/spsc_press_20121004.pdf
4) Apple, ASIdentifierManager, http://developer.apple.com/library/ios/#documentation/AdSupport/Reference