Vol.2013-DPS-154 No.58
Vol.2013-CSEC-60 No.58
2013/3/15

# マルチユーザ利用を考慮したスマートデバイス向けデータ保護方式の提案

栗原優樹† 市原尚久†

近年，スマートフォンやタブレット端末などのスマートデバイスの業務活用が注目されている．スマートデバイスは常時携行が前提となるために，PC などの情報端末と比べて紛失による情報漏洩の可能性が高く，スマートデバイス内に保存された業務ファイルの保全方法が課題となる．課題解決の方法として業務ファイルを暗号化し，復号する際にデバイス外で保管されている復号鍵を用いるという方法が考えられるが，複数ユーザがデバイスや業務ファイルを共用するケースでは，業務ファイルごとのアクセスストラクチャが異なるために１ユーザが業務ファイルごとに異なる復号鍵を保持しなければならなくなる．そこで本研究では，１つの復号鍵で異なるアクセスストラクチャを持つ業務ファイルへアクセスすることのできる方式を提案する．

# A Proposal of a Data Protection Scheme for Smart Devices Considering Multi-User Environments

YUUKI KURIBARA† NAOHISA ICHIHARA†

Recent years, smart phones and tablet devices are widely used and one uses those smart devices not only on private but also on business. Suppose that one uses a smart device on business and mounts confidential business files on the device, it is an issue how to keep confidentiality of the files since a smart device is more likely to be lost than note-book or desk-top type computers. Trivially, one can keep confidentiality of the files if one encrypts the files and keep its decryption key secretly out of the smart device; however, it needs a different decryption key by the different access structure to decrypt encrypted files which have different access structures each other, when a smart device and contained files are shared by multiple users. This yields quite complicated life cycle management for the keys. In this study, we propose a decryption key generation scheme so that one decrypts the different encrypted files with a single decryption key.

## 1. はじめに

近年，スマートフォンやタブレット端末（以降，スマートデバイス）を用いた業務活用が注目されている．スマートデバイスの業務活用では，例えば地下において建設作業を行う場合など，ネットワークに常時接続することが困難な環境下で利用することも想定されている．そのような環境では，ネットワーク接続をすることなく業務アプリを利用可能とするために，スマートデバイス内に業務ファイルを保存する必要がある．

スマートデバイスは常時携行することが前提となることから, PC などの情報端末と比べて紛失による情報漏洩の可能性が高く，スマートデバイス内に保存された業務ファイルの保全方法が課題となる．課題解決方法として，業務ファイルを AES[1) ] のような共通鍵暗号方式を用いて暗号化し，秘密鍵（暗号化鍵兼復号鍵）をスマートデバイス外に保管しておくという方法が考えられる．スマートデバイスや業務ファイルをシングルユーザで利用するケースでは，すべての業務ファイルを同一の秘密鍵を用いて暗号化および復号することにより１ユーザが保持する秘密鍵を１つに限定することも可能であるが，複数ユーザがスマートデバイスにインストールされている業務アプリや業務ファイルを利用する場合（マルチユーザセッティング）においても同様に１ユーザ１秘密鍵とするためには，鍵生成の方法に工夫が求められる．例えばスマートデバイスをユーザ A，ユーザ B およびユーザ C で共用し，ユーザ A およびユーザ B がアクセス可能な業務ファイル（ファイル甲）と，ユーザ A およびユーザ C がアクセス可能な業務ファイル（ファイル乙）がそれぞれ AES で暗号化されて保存されているケースを想定する場合，ユーザ A はファイル甲の秘密鍵およびファイル乙の秘密鍵をそれぞれ保持する必要がある．つまりマルチユーザセッティングにおいて，AES の秘密鍵そのものをユーザが保持する方式とすると，業務ファイルのアクセスストラクチャ（業務ファイルへのアクセス権を与えられたグループ）ごとに異なる秘密鍵を保持しなければならなくなる．一般に，保持しなければならない秘密鍵数の増大は，その保全や失効管理などに必要な運用コストの増大を招くため，１ユーザ１秘密鍵で異なるアクセスストラクチャを持つ業務ファイルにアクセス可能な方式が望ましい．

そこで本研究では，マルチユーザセッティングにおいて，あるユーザがアクセスすることを許可されているすべての暗号化された業務ファイルを，ひとつのユーザ固有鍵（ユーザ固有の秘密情報）だけで復号することのできる方式を提案する．提案方式では，各業務ファイルがアクセスストラクチャの異なる業務ファイルごとに, 異なる AES の秘密

† 株式会社 NTT データ
NTT DATA CORPORATION

Vol.2013-DPS-154 No.58
Vol.2013-CSEC-60 No.58
2013/3/15

鍵で暗号化されているものとする．ユーザ A がある暗号化された業務ファイルを復号する場合，ユーザ A は自身のユーザ固有鍵（たとえばパスワードなど）を，業務ファイルを利用する業務アプリへ入力する．業務アプリは受け取ったユーザ固有鍵と業務アプリ固有の情報（たとえばパッケージ名など）を鍵生成部（図 1，(1) および (2)）に渡し，鍵生成部において業務ファイルを復号するために必要な AES の秘密鍵（図 1，(3)）が生成される．生成された秘密鍵は復号部に渡され，復号部で復号された業務ファイルが業務アプリへ渡される．鍵生成部で生成される秘密鍵は業務ファイルのアクセスストラクチャごとに異なるが，提案方式では ElGamal 暗号[2] を応用したアルゴリズムを導入することにより，ひとつのユーザ固有鍵から業務ファイルに対応した異なる秘密鍵を生成可能な方式となっている．

## 2. 提案方式

本章では，マルチユーザセッティングにおいて，ユーザが 1 つのユーザ固有鍵のみを用いて，利用が許されているすべての業務ファイルの秘密鍵を生成可能な鍵生成方式について記述する．

### 2.1 鍵生成部の構成と業務ファイルの復号フロー

業務ファイルの秘密鍵を生成する鍵生成部は，ユーザの固有鍵と事前に設定された変数を入力としてシード 2 と呼ぶ値を生成するシード 2 生成部，業務アプリ固有の情報からシード 1 と呼ぶ値を生成するシード 1 生成部，シード 1 とシード 2 からシードと呼ぶ値を生成し AES の秘密鍵を生成するファイル鍵生成部から構成される．ここでシード 2 生成部の入力となる事前に設定された変数は，アクセスストラクチャの異なる業務ファイルごとに設定される固有の変数であり，スマートデバイス内の設定ファイル等に書き込まれているものとする．

業務アプリが業務ファイルへアクセスする際には，まずユーザが業務アプリへユーザ固有鍵を渡し，次に業務アプリがシード 2 生成部へユーザ固有鍵を渡すとともにシード 1 生成部へ業務アプリ固有の情報を渡す．ユーザ固有鍵を受け取ったシード 2 生成部が設定ファイルから該当する変数を読み込み，シード 2 を生成すると同時に，シード 1 生成部は業務アプリ固有の情報からシード 1 を生成する．シード 1 生成部とシード 2 生成部からそれぞれシード 1，シード 2 を受け取ったファイル鍵生成部は，シード 1 とシード 2 を結合した値にパディング処理を施した値を入力とした暗号学的ハッシュ関数の出力を AES の秘密鍵へ加工することで業務ファイル用の秘密鍵を生成し，ファイルを復号する復号部へ生成した秘密鍵を渡す．最後に復号部は復号した業務ファイルを業務アプリへ渡す．生成された秘密鍵は業務アプリが業務ファイルを利用している間，キャッシュに一時保存され，業務ファイルを再び保管する際に同一の秘密鍵によって暗号化されるものとする．

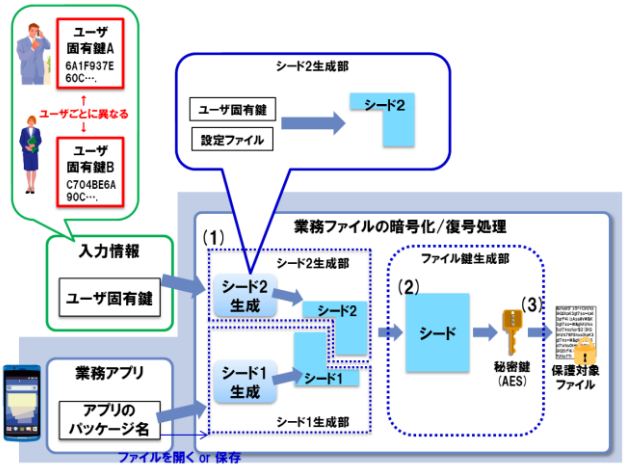


**図 1：提案方式の概要図**

### 2.2 シード 2 の生成方式

いま，すべての業務ファイルはユーザ数 2 のアクセスストラクチャを持つものとし，ある業務ファイルのアクセスストラクチャがユーザ A およびユーザ B で構成される場合を考える．ユーザ A の固有鍵を$\alpha$としユーザ B の固有鍵を$\beta$とする．ただし$p$を十分大きな素数とし，それぞれのユーザ固有鍵は$Z_{p-1}$上でランダムに抽出された要素とする．業務ファイルへのアクセス権を設定する信頼できる第三者は乱数$k_{\alpha\beta}(\in_u Z_{p-1})$を選び，

$$x_{\alpha\beta} = h(\alpha,\theta) + h(\beta,\theta) + k_{\alpha\beta} \bmod p\text{-}1$$

として，$x_{\alpha\beta}$を計算する．ただし，$h$は暗号学的に理想的なハッシュ関数とし，$\theta$は当該業務ファイルを一意に識別する識別子とする．第三者は 3 つの乱数$r,\ t_\alpha,\ t_\beta(\in_u Z_{p-1})$を独立に選び，$y_{\alpha\beta} = g^{x_{\alpha\beta}} \bmod p$として以下を計算する．

$$s_\alpha = h(\alpha,\theta)t_\alpha + h(\beta,\theta) + k_{\alpha\beta} \bmod p\text{-}1,$$
$$s_\beta = h(\beta,\theta)t_\beta + h(\alpha,\theta) + k_{\alpha\beta} \bmod p\text{-}1,$$
$$c_1 = g^r \bmod p, c_2 = m_{\alpha\beta} y_{\alpha\beta}^r \bmod p.$$

ここで$g$は巡回群Gの生成元，巡回群Gは暗号学的に適切に選ばれているものとし，$m_{\alpha\beta}$は群G上でランダムに選ばれたシード 2 の値とする．第三者は生成した以下の値の組を変数の組としてあらかじめ設定ファイルに格納しておく．

$$< s_\alpha, t_\alpha, s_\beta, t_\beta, h, \theta, p, g, c_1, c_2 > .$$

ユーザ A が業務アプリを通じて業務ファイルへアクセスする場合，業務アプリから呼び出されたシード 2 生成部は設定ファイルに格納されている変数の組から$< s_\alpha, t_\alpha, p, g, c_1, c_2 >$を取り出し，入力されたユーザ A の固有鍵$\alpha$を用いて以下の計算によりシード 2 の値$m_{\alpha\beta}$を算出し，ファイル鍵生成部へ渡す．

$$m_{\alpha\beta} = \frac{c_2}{c_1^{\alpha} c_1^{(s_\alpha - \alpha t_\alpha)}} \bmod p.$$

ファイル鍵生成部へ$m_{\alpha\beta}$を渡したあと，シード 2 生成部はユーザ固有鍵αと設定ファイルに格納されている変数$s_\alpha$および$t_\alpha$を用いて$x_{\alpha\beta}$を算出し，あらたな乱数$\check{r}$を用いて以下の値を生成する．

$$\check{c}_1 = g^{\check{r}} \bmod p, \check{c}_2 = m_{\alpha\beta} g_{\alpha\beta}^{\check{r} x_{\alpha\beta}} \bmod p \ .$$

シード 2 生成部は設定ファイルに格納されている$c_1, c_2$を$\check{c}_1, \check{c}_2$へ書き換える．この書き換え処理は，アクセスストラクチャに含まれる各ユーザが業務ファイルを利用する際に必ず実行される共通の処理である．

ユーザ B が同じ業務ファイルにアクセスする場合は，シード 2 生成部は設定ファイルに格納されている変数の組から$< s_\beta, t_\beta, p, g, c_1, c_2 >$を取り出し，入力された固有鍵βを用いて同様の計算により値$m_{\alpha\beta}$を算出する．

$$m_{\alpha\beta} = \frac{c_2}{c_1^{\beta} c_1^{(s_\beta - \beta t_\beta)}} \bmod p.$$

つぎにアクセスストラクチャがユーザ A とユーザ C で構成される業務ファイルを考える．ユーザ A の固有鍵を同じくαとしユーザ C の固有鍵をγとする．業務ファイルへのアクセス権を設定する信頼できる第三者は乱数$k_{\alpha\gamma}\left(\in_u z_{p-1}\right)$を選び，

$$x_{\alpha\gamma} = h(\alpha, \delta) + h(\gamma, \delta) + k_{\alpha\gamma} \bmod p\text{-}1$$

として$x_{\alpha\gamma}$を計算する．ただし，$\delta$は当該業務ファイルを一意に識別する識別子とする．第三者は 3 つの乱数$\acute{r},\ \acute{t}_\alpha,\ t_\gamma\left(\in_u z_{p-1}\right)$を独立に選び，$y_{\alpha\gamma} = g^{x_{\alpha\gamma}} \bmod p$として以下を計算する．

$$\acute{s}_\alpha = h(\alpha, \delta)\acute{t}_\alpha + h(\gamma, \delta) + k_{\alpha\gamma} \bmod p\text{-}1,$$
$$s_\gamma = h(\gamma, \delta)t_\gamma + h(\alpha, \delta) + k_{\alpha\gamma} \bmod p\text{-}1,$$
$$\acute{c}_1 = g^{\acute{r}} \bmod p, \acute{c}_2 = m_{\alpha\gamma} y_{\alpha\gamma}^{\acute{r}} \bmod p\text{-}1.$$

ここで$m_{\alpha\gamma}$は当該業務ファイルの秘密鍵を生成するために必要なシード 2 の値であり，群G上でランダムに選ばれているものとする．第三者は生成した以下の値の組を変数の組としてあらかじめ設定ファイルに格納しておく．

$$< \acute{s}_\alpha, \acute{t}_\alpha, s_\gamma, t_\gamma, h, \delta, p, g, \acute{c}_1, \acute{c}_2 > .$$

ユーザ A が業務アプリを通じて業務ファイルへアクセスする場合，業務アプリから呼び出されたシード 2 生成部は設定ファイルに格納されている変数の組から$< \acute{s}_\alpha, \acute{t}_\alpha, p, g, \acute{c}_1, \acute{c}_2 >$を取り出し，入力されたユーザ A の固有鍵αを用いて以下の計算によりシード 2 の値$m_{\alpha\gamma}$を算出し，ファイル鍵生成部へ渡す．

$$m_{\alpha\gamma} = \frac{\acute{c}_2}{\acute{c}_1^{\alpha} \acute{c}_1^{(\acute{s}_\alpha - \alpha \acute{t}_\alpha)}} \bmod p.$$

ユーザ C が当該業務ファイルへアクセスする際にも，同様にシード 2 の値$m_{\alpha\gamma}$を算出する．

このような方式とすることで，ユーザ A は自身の固有鍵αを用い，異なるアクセスストラクチャを持つ業務ファイルへアクセスすることが可能になる．アクセスストラクチャを構成するユーザが 3 人以上となる場合も，上記の手法を拡張することで実現できる．

なお，パスワードからユーザ固有鍵を生成する場合は，シード 2 生成部の前処理として，暗号学的ハッシュ関数などを用いてパスワードを$Z_{p-1}$上に写すなどの方法が考えられる．

### 2.3 シード 2 生成方式の安全性に関する考察

本節では，正規ユーザではないアクターがなんらかの理由・手段によってスマートデバイスを取得し，実装されているソフトウェアやデータに対して静的解析を行ったとしても，アクターはシード 2 の値を復元することが困難であることを示す．シード 2 の復元が困難であれば，暗号化された業務ファイルを復号するための秘密鍵の復元も困難であるため，提案方式はスマートデバイスの紛失などによる情報漏洩への対策として有効な方式となっていることがわかる．以下，シード 2 生成方式の安全性に関する考察のラフスケッチを記述する．

いま，異なるアクセスストラクチャを持つ業務ファイル甲および業務ファイル乙が暗号化された状態でスマートデバイス内に格納されているものとする．業務ファイル甲のアクセスストラクチャはユーザ A とユーザ B で構成され，業務ファイル乙のアクセスストラクチャはユーザ A とユーザ C で構成されるものとする．なお，以降本節で用いる記号は，2.2 節で定義されたものと同一とする．

ここで，設定ファイルを含むスマートデバイス内のすべてのリソースへアクセスすることが可能であり，静的な解析により業務ファイル甲に対応するシード 2 の値$m_{\alpha\beta}$に関する一部情報の復元を試みる敵を想定する．

敵は設定ファイルにアクセス可能であるため，以下の情報を得ることができる．

$$< s_\alpha, t_\alpha, s_\beta, t_\beta, h, \theta, p, g, c_1, c_2 > ,$$
$$< \acute{s}_\alpha, \acute{t}_\alpha, s_\gamma, t_\gamma, h, \delta, p, g, \acute{c}_1, \acute{c}_2 > .$$

$\theta, \delta, p, g$ を固定し，$h$を理想的な暗号学的ハッシュ関数であるとすると，2.2 節より$< s_\alpha, t_\alpha, s_\beta, t_\beta, c_1, c_2 >$と$< \acute{s}_\alpha, \acute{t}_\alpha, s_\gamma, t_\gamma, \check{c}_1, \acute{c}_2 >$は敵にとって互いに独立に生成されているように見える．つまり業務ファイル乙に対応する設定ファイル内の変数は，敵に対し$m_{\alpha\beta}$に関する情報をなんら与えていない．したがって敵は以下の情報から$m_{\alpha\beta}$に関する情報を得る必要がある．

$$s_\alpha = h(\alpha, \theta)t_\alpha + h(\beta, \theta) + k_{\alpha\beta} \bmod p\text{-}1,$$
$$s_\beta = h(\beta, \theta)t_\beta + h(\alpha, \theta) + k_{\alpha\beta} \bmod p\text{-}1,$$
$$c_1 = g^r \bmod p, c_2 = m_{\alpha\beta} y_{\alpha\beta}^r \bmod p.$$

$h(\alpha, \theta), h(\beta, \theta), k_{\alpha\beta}, m_{\alpha\beta}$は敵にとって未知の互いに独立なランダム変数であることにより，敵は上記の情報から$m_{\alpha\beta}$に関する情報をなんら得ることはできない．

なお提案方式は，正規ユーザがスマートデバイス OS の root 権限を奪取するなどの手段 [3] により，スマートデバイ

Vol.2013-DPS-154 No.58
Vol.2013-CSEC-60 No.58
2013/3/15

スに実装されているソフトウェアやデータに対して静的解析を行った場合においても，自身がアクセスストラクチャに含まれない業務ファイルにアクセスすることが困難な方式となっている．たとえば上記のユーザ A は，自身のユーザ固有鍵と設定ファイルに格納されている変数から$h(\beta,\theta)$と$h(\gamma,\delta)$を計算することができるが，$h$が理想的な暗号学的ハッシュ関数であるとすれば，これらの値からユーザ B の固有鍵である$\beta$やユーザ C の固有鍵である$\gamma$を特定することは困難である．また$h(\beta,\theta)$や$h(\gamma,\delta)$は業務ファイルに紐づけられた値である（ファイル識別子がハッシュ関数の入力となっている）ため，他の業務ファイルに対応する変数とは独立の値となっており，不正アクセスのために利用することができない．

本研究の関連研究として，谷内による暗号を用いたファイルのアクセス制御方式[4]が提案されている．谷内の方式は，各業務ファイルを AES のような共通鍵暗号で暗号化し，（AES の）秘密鍵をアクセスストラクチャを構成するユーザごとの公開鍵によって暗号化する方式となっている．アクセスストラクチャに含まれるユーザが業務ファイルにアクセスする際には，自身の公開鍵に対応する復号鍵を用いて AES の秘密鍵を復元し，復元された秘密鍵を用いて業務ファイルを復号する．谷内の方式における安全性は，秘密鍵を暗号化する公開鍵暗号の暗号学的強度と，業務ファイルを暗号化する共通鍵暗号の暗号学的強度に依存する．一方，提案方式の安全性は，設定ファイルに格納する変数を生成する際に用いるハッシュ関数の暗号学的強度と，業務ファイルを暗号化する共通鍵暗号の暗号学的強度に依存する．

### 2.4 アクセスストラクチャの変更やユーザ固有鍵の変更に伴う業務フロー

提案方式では，業務ファイルのアクセスストラクチャが変更された場合やユーザ固有鍵が変更された場合，設定ファイルに格納する変数を再生成する必要がある．以下ではアクセスストラクチャを変更する場合とユーザ固有鍵が変更された場合における業務フローを記述する．ただし，業務ファイルへのアクセス権を設定する信頼できる第三者は，以下の機能を備える Web サーバを運用しているものとする．

- アクセスストラクチャに基づいて，2 章で記述した方式でスマートデバイスの設定ファイルに格納する変数を生成する機能
- 設定ファイルをスマートデバイスに配置する機能
- ユーザ（スマートデバイス）からユーザ固有鍵の変更通知を受け取る機能やアクセス権の新規付与に関する要求を受け付ける機能，アクセスストラクチャからの除外要求を受け付ける機能，変更された固有鍵を受け取る機能
- 各業務ファイルに紐づけて，アクセスストラクチャを構成する各ユーザの固有鍵を保管する機能（キーストア）

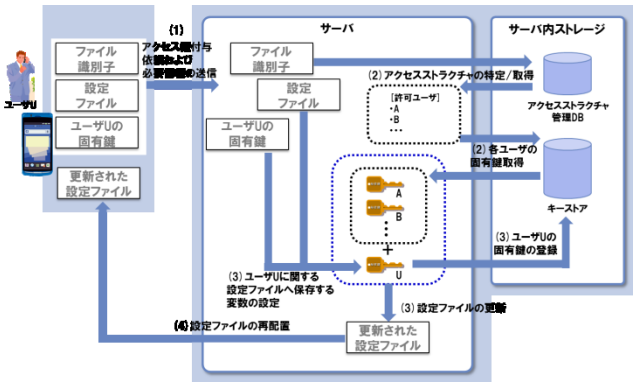


図 2：新たにユーザを加える際の業務フロー

#### 2.4.1 アクセスストラクチャを変更する場合の業務フロー

以下ではある業務ファイルのアクセスストラクチャに対し新たにユーザ U を加える際の業務フローを示す．

(1) ユーザ U は第三者が運用する Web サーバへ接続し，アクセス権の新規付与に関する要求とともに当該業務ファイルを一意に識別するためのファイル識別子，スマートデバイスに格納されている設定ファイル，および自身のユーザ固有鍵をサーバへ送る．

(2) ユーザ U から要求を受けたサーバは，ファイル識別子をキーとして該当する業務ファイルのアクセスストラクチャを特定し，アクセスストラクチャを構成する各ユーザの固有鍵を用いて設定ファイルに格納されている変数からシード 2 の値を計算する．

(3) サーバは(2)で計算したシード 2 を変更することなく 2 章で示した方式によって設定ファイルへ格納する変数を新たに生成し，設定ファイルを更新する．ユーザ U の固有鍵がキーストアに登録されていない場合には，ファイル識別子に紐づけてキーストアへあらたに登録する．

(4) サーバは更新した設定ファイルをスマートデバイスへ再配置し，アクセス権の付与が完了したことをユーザ U へ通知する．

上記は，ユーザ U による要求を前提とした業務フローとなっているが，別のアクターがアクセスストラクチャの更新を要求する場合においてもほぼ同様のフローにより実行される．

Vol.2013-DPS-154 No.58
Vol.2013-CSEC-60 No.58
2013/3/15

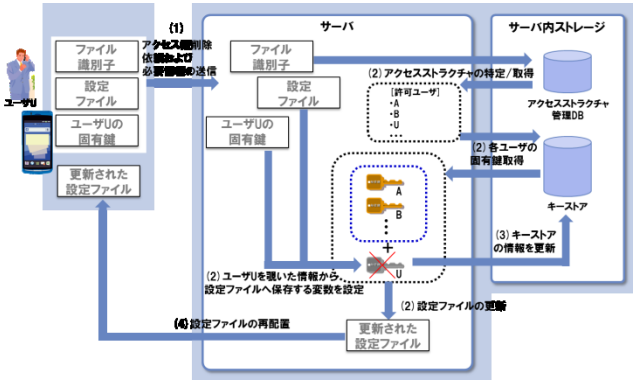


図 3：ユーザを除く際の業務フロー

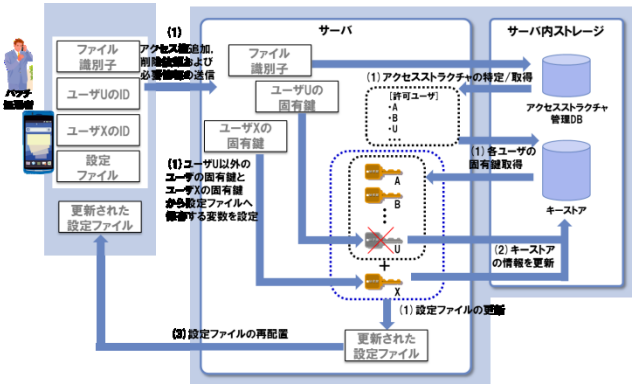


図 4：一部のユーザを除外し，同時に新規ユーザを追加する際の業務フロー

次にある業務ファイルのアクセスストラクチャからユーザ U を除く際の業務フローを示す．

(1) ユーザ U は第三者が運用する Web サーバへ接続し，アクセスストラクチャからの除外に関する要求とともに自身を識別するための ID，当該業務ファイルを一意に識別するためのファイル識別子とスマートデバイスに格納されている設定ファイルをサーバへ送る．

(2) 要求を受けたサーバは，ファイル識別子をキーとして該当する業務ファイルのアクセスストラクチャを特定する．アクセスストラクチャからユーザ U を除いた各ユーザの固有鍵に基づき，2 章で記述した方式により設定ファイルに格納する変数を新たに生成し，設定ファイルを更新する．このとき，シード 2 の値については旧値と異なる値が新たに選びなおされているものとする．

(3) サーバはアクセスストラクチャ（ユーザ U を除いたストラクチャ）に関する情報を更新する．（キーストア内の情報を更新する．）

(4) サーバは更新した設定ファイルをスマートデバイスへ再配置し，アクセスストラクチャからの除外が完了したことをユーザ U へ通知する．

上記は，ユーザ U による要求を前提とした業務フローとなっているが，別のアクターがアクセスストラクチャからの除外を要求する場合においてもほぼ同様のフローに沿って実行される．

次に，サーバにおけるバッチ処理である業務ファイルのアクセスストラクチャから一部のユーザ（ユーザ U）を除外し，同時に新規ユーザ（ユーザ X）を追加する際の業務フローを示す．

(1) サーバは，ファイル識別子をキーとして該当する業務ファイルのアクセスストラクチャを特定する．ただしファイル識別子は，サーバのバッチ処理を起動する別のアクターから受け取るものとする．アクセスストラクチャからユーザ U を除いた各ユーザの固有鍵とユーザ X の固有鍵に基づき，2 章で記述した方式により設定ファイルに格納する変数を新たに生成し，設定ファイルを更新する．設定ファイルは，サーバのバッチ処理を起動する別のアクターから受け取るものとする．シード 2 の値については旧値と異なる値が新たに選びなおされているものとする．

(2) サーバはアクセスストラクチャ（ユーザ U を除きユーザ X を追加したストラクチャ）に関する情報を更新する．（キーストア内の情報を更新する．）

(3) サーバは更新した設定ファイルをスマートデバイスへ再配置する．再配置のタイミングはサービスの仕様により様々であるが，アクセスストラクチャの変更と同期して行われることが望ましい．

### 2.4.2 ユーザ固有鍵を変更する際の業務フロー

以下では，ある業務ファイルのアクセスストラクチャに含まれるユーザ U がユーザ固有鍵を変更する際の業務フローを示す．

(1) ユーザ U は第三者が運用する Web サーバへ接続し，ユーザ固有鍵の変更に関する要求とともに当該業務ファイルを一意に識別するためのファイル識別子，スマートデバイスに格納されている設定ファイル，および新たなユーザ固有鍵をサーバへ送る．

(2) ユーザ U から要求を受けたサーバは，ファイル識別子をキーとして該当する業務ファイルのアクセスストラクチャを特定し，アクセスストラクチャを構成する各ユーザの固有鍵と用いて設定ファイルに格納されている変数からシード 2 の値を計算する．

(3) サーバは(2)で計算したシード 2 を変更すること

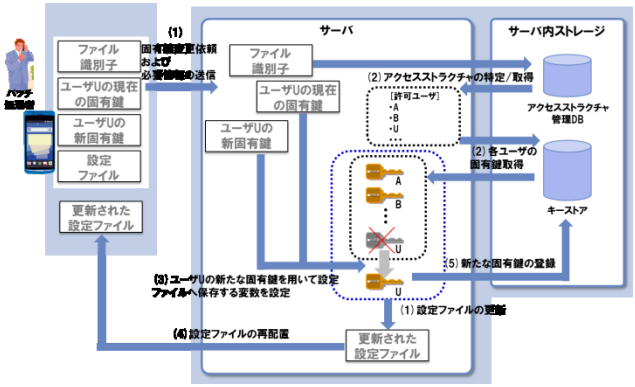


**図 5：ユーザ固有鍵を変更する際の業務フロー**

なく 2 章で示した方式によって設定ファイルへ格納する変数を新たに生成し，設定ファイルを更新する．このとき，ユーザ U の固有鍵については(1)で受け取ったあらたな固有鍵を用いるものとする．

(4) サーバは更新した設定ファイルをスマートデバイスへ再配置し，ユーザ固有鍵の変更が完了したことをユーザ U へ通知する．

(5) ユーザ U のあらたな固有鍵をファイル識別子に紐づけてキーストアへ登録する．

## 3. まとめ

本論文では，マルチユーザセッティングにおいて，あるユーザがアクセスすることを許可されているすべての暗号化された業務ファイルを，ひとつのユーザ固有鍵から復号することを可能にする方式を提案した．提案方式は，ユーザにパスワードなどのユーザ固有の情報を入力させ，スマートデバイス内に保存されている設定ファイルと共に業務ファイルを復号するための共通鍵を生成する仕組みとなっている．また，業務ファイルのアクセスストラクチャが変更する際の業務フローや，ユーザの固有鍵を更新する際の業務フローを示した．

提案方式では業務ファイルのアクセスストラクチャを変更する度に，信頼できる第三者が設定ファイルを更新し，更新した設定ファイルをスマートデバイスに再配置する必要がある．アクセスストラクチャを変更するプロセスをスマートデバイスのユーザが起動する場合には，アクセスストラクチャの変更と設定ファイルの更新・再配置が同期して実施されるが，サーバのバッチ処理によって設定ファイルの更新を行う場合には，スマートデバイスへの再配置を非同期で行うこととなり，業務ファイルへのアクセス権が失効しているにも関わらずアクセスが可能となる状況が生じる．このような状況を解消するためには，クライアント側（スマートデバイス）から定期的にサーバへ接続し，設定ファイルの最新化を自動で行う管理アプリを実装する，あるいは信頼できる第三者を想定せずにスマートデバイス内で設定ファイルの更新を行うことのできる仕組みを考案するなどのアプローチが考えられる．

また，提案方式では異なるアクセスストラクチャごとに設定ファイルへ格納する変数が異なるため，そのようなアクセスストラクチャが増加するに従って設定ファイルのサイズが大きくなる．ただし，通常アクセスストラクチャは業務ごとに設計されること，１スマートデバイスを共有するユーザの数は限られることが想定されるため，設定ファイルのサイズが実用上許容できなくなるほど増大することはないものと考えられる．

## 参考文献

1) Specification for the ADVANCED ENCRYPTION STANDARD (AES)
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
2) Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms; Taher Elgamal; CRYPTO 84, pp. 10-18, Springer-Verlag.
3) タオソフトウェア株式会社: Android Security 安全なアプリケーションを作成するために，インプレスジャパン (2011)
4) 谷内 崇浩: 暗号を用いたファイルのアクセス制御についての研究，情報処理学会全国大会講演論文集, vol. 70, No. 3, pp. 3.435-3.436