Vol.2014-DPS-158 No.45
Vol.2014-CSEC-64 No.45
2014/3/7

Android端末を用いたモバイル匿名属性認証システムの実装

三嶋 徹[1]　中西 透[1]　渡邊 寛[1]　舩曳 信生[1]

概要：ID を用いたユーザ認証システムでは、サービス提供者が ID とひも付けすることにより，ユーザの利用履歴を蓄積できてしまう問題がある．一方，サービス提供者はユーザの属性を用いて認証を行いたい場合も多い．この問題の解決策として、個人を特定できる情報を用いずにユーザの属性のみを用いて認証を行う匿名属性認証システムが提案されている．本研究では Android 端末に搭載されている NFC および Bluetooth を利用し，端末間でかざす動作により簡便に匿名属性認証を行うシステムを実装した．実機での評価実験の結果，認証プロトコルの実行に要した時間は約 0.98 秒であり、端末間の接続確立も含めた全処理時間は約 1.85 秒となった．

An Implementation of Moblie Anonymous Attribute Authentication System on Android Devices

Abstract: In current ID-based authentication systems, service providers can collect user's history via user's ID. On the other hand, service providers often want to authenticate user's attributes. To solve this problem, an anonymous attribute authentication system was proposed, where the user's attribtes are autheticated without revealing any information that can speciy the user. In this paper, we implemented a mobile anonymous attribute system using NFC and Bluetooth, where, by bringing the device close to another one, we can perform a direct anonymous attribute authentication. From the experiments on real devices, the execution time of anonymous attribute authentication is 0.98 seconds, and the total processing time including the connection establishment is 1.85 seconds.

1. はじめに

現在，ユーザの持つ属性を確認および取得するシステムやサービスが増加している．例として，施設への入室管理時に所属を確認したり，たばこの販売における年齢認証がある．現在利用されているパスワードベースの認証では，ユーザ登録やシステム利用の際にユーザは自身の属性を検証者に対して公開し，検証者がユーザ ID に対して属性を紐付ける．ユーザ認証時に，検証者はユーザ ID に紐付けられた属性を利用して属性認証を行う．しかしながら，このような認証システムでは，検証者がユーザの利用履歴や個人情報を収集でき，さらにユーザ ID と紐づけることによりユーザのプロファイリングが可能になることから，重大なプライバシ問題が発生しうる．

これらのプライバシ問題の解決策として，匿名属性認証システム [1] が提案されている．匿名属性認証システムでは，ユーザ ID のような個人を特定できる情報を用いずに，ユーザは検証者が要求する属性を所持していることを証明する．検証者は，認証の際にユーザから受け取った署名が指定された属性を持つユーザが作成したものであることは検証可能であるが，どのユーザが作成した署名であるかは特定できない．このため，匿名属性認証システムを用いることにより，プライバシを保護しつつ正確な属性の確認および取得が行える．

先行研究として，[1] のシステムをプロキシに実装し Web サービスと連携させた，匿名属性認証システム [3] が提案されている．さらに，[3] のシステムのユーザ側の署名生成機能及び Web システムとの連携機能の Android への実装 [4] も行っている．しかしながら，従来のシステムは Web システムとの連携を目的として実装されていたため，認証を行うための環境として IP ネットワークの構築や検証者用にサーバコンピュータを必要とし，モバイル環境下での利用が困難であった．

そこで本研究では NFC を用いて，端末同士をかざすことにより匿名属性認証を行うシステムを実装する．昨今の多くの Android 端末は NFC や Bluetooth といった近距離

[1] 岡山大学
Okayama University, Okayama, Okayama 700–8530, Japan

無線通信機能を備えており，簡単に端末間で直接通信を実装することができる．通信手段として NFC を選択した理由として，端末同士をかざす動作だけで通信が可能であるため，ユーザの操作が簡単であることが挙げられる．また，端末間の直接通信を利用するため，Wi-Fi のようなネットワークインフラが整っていない環境においても認証を行うことができる．

実装の方針として，匿名属性認証システムの署名検証機能を Android に実装し，通信方法を NFC に変更する．しかし，NFC は対話的な通信の実装が困難なため，接続する端末は NFC で決定し，データの送受信は Bluetooth で行う NFC-Bluetooth ハンドオーバを利用する．また，認証アルゴリズムや送受信するメッセージの変更に対する柔軟性を考慮し，匿名属性認証でやり取りするメッセージを JSON フォーマットに変換して送受信する設計とした．実装にあたって，ユーザインターフェースや NFC，Bluetooth といった機能は Java 言語を用いて実装した．匿名属性認証に必要な署名生成・署名検証機能は C 言語で記述されているため，JNI を用いてネイティブコードを呼び出している．

実装システムの評価として，Android4.3 を搭載したタブレット PC を 2 台用いて，それぞれユーザ側・検証者側の機能を動作させて認証に必要な時間を計測した．その結果，NFC-Bluetooth ハンドオーバに要した時間が約 0.87 秒，匿名属性認証プロトコルの実行時間が約 0.98 秒となった．全体の処理時間は約 1.85 秒であり実用的とは言い難いため，高速化の検討が今後必要である．

本論文の章構成について述べる．まず，2 章で先行研究として匿名属性認証システムおよびその Android 実装ついて述べる．次に，3 章で本システムの実装について述べる．そして 4 章で実験と評価を示す．最後に，5 章で本論文をまとめる．

2. 先行研究と利用する技術

2.1 先行研究

2.1.1 匿名属性認証システム

匿名属性認証とは，個人を特定できる情報を用いずに，検証者が要求する属性の所持をユーザが検証者に対して証明する認証方式である．匿名属性認証における属性とは，ユーザが共通して持つ性質や特徴の情報であり，国籍，性別，所属といったものが挙げられる．[1] で提案されている匿名属性認証システムでは，ユーザの特定が不可能な範囲での属性値を提示することにより，ユーザは検証者が指定する属性の所持を検証者に匿名で証明することができる．この認証システムでは以下の 2 つの関係の証明を行える．

- AND relation
  指定された複数の属性すべてをユーザが所持すること
- OR relation
  指定された複数の属性のうち 1 つをユーザが所持すること

例えば，AND relation では，ユーザが学生かつ特定の学部に所属しており，在学中であることを学部棟に入るために証明できる．OR relation では，コピー機を使用する際に，スタッフか教員のどちらかであることを証明できる．AND・OR relation のどちらも，証明コストは属性数に依存せず一定である．

匿名属性認証システムの概要を図 1 に示す．証明書発行

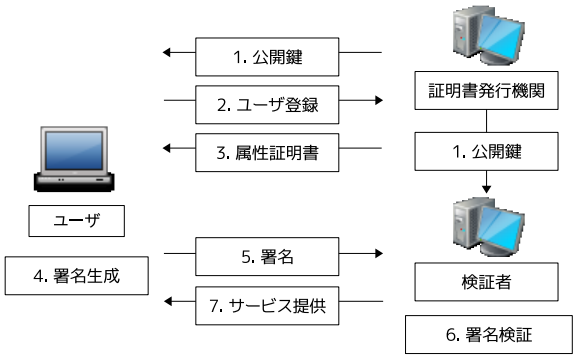

図 1 匿名属性認証システムの概要

機関は署名作成および検証用の公開鍵を配布する．ユーザは証明書発行機関に対してユーザ登録を行い，属性証明書を受け取る．ユーザ認証では，まず，ユーザと検証者の間で事前に共有した乱数メッセージ，属性証明書，証明書発行機関の公開鍵，署名に使用する属性カテゴリおよび属性値を用いてユーザが署名を作成し，検証者に送信する．その際，検証に必要な属性カテゴリも検証者に送る．属性カテゴリとは，ユーザの属性値ではなく属性値が所属するグループである．例えば，属性値が男性あるいは女性である場合，属性カテゴリは性別となる．次に，検証者は乱数メッセージ，証明書発行機関の公開鍵，属性カテゴリ，属性値を基に署名の検証を行う．このシステムでは，ユーザと検証者との間で複数回メッセージをやり取りするため，対話的通信が可能な環境が必要である．

匿名属性認証システムの Web サービスへの応用として，[1] の匿名属性認証システムをアンケートサービスに応用した Web システム [3] が提案されている．また，匿名属性認証システムにおけるクライアント機能の Android への実装 [4] も提案されている．

2.1.2 匿名属性認証システムにおけるクライアント機能の Android 実装

[4] の実装では，匿名属性認証システムの署名生成アルゴリズムを認証用アプリケーションとして実装し，Web ブラウザと連携させている．

Android における匿名属性認証システムの概要を図 2 に示す．

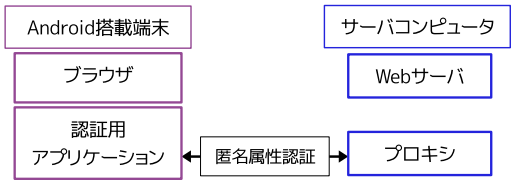

図 2　Web サービスに対する Android 用匿名属性認証システムの概要

この実装では，クライアント側に認証用アプリケーション，サーバ側にプロキシを配置することにより，匿名属性認証システムを Web システムと連携させている．クライアント側の端末で特定の Web サイトにアクセスすると，認証用アプリケーションがサーバ側のプロキシと匿名属性認証を行い，結果ページを Web ブラウザに表示させる．

認証用アプリケーションの通信機能および Web ブラウザとの連携機能の実装は Java 言語，匿名属性認証アルゴリズムの実装は C 言語を用いている．また，匿名属性認証アルゴリズムは C 言語で書かれた多倍長演算ライブラリ GMP(The GNU Multiple Precision Arithmetic Library)[9]，および GMP に基づいた楕円曲線暗号ライブラリ ELiPS(Efficient Library for Pairing - based System)[5] に依存している．

Android OS 上で C 言語で書かれたプログラムを実行するためには，クロスコンパイラを用いて端末に搭載された CPU が実行できるようコンパイルする必要がある．この実装では，ARM アーキテクチャの CPU を対象にネイティブコードを作成している．ネイティブコードは Android NDK[7] および droid-wrapper[8] を使用して作成し，JNI を用いて Java 側から呼び出している．

2.2　利用する技術

本研究では，NFC-Bluetooth ハンドオーバを用いて実装するため，各技術の概要を以下に示す．

- NFC
  NFC(Near Field Communication) とは 13.56MHz 帯の電波を利用する短距離通信規格である．NFC の通信距離は 10cm 程度であり，かざす動作で 2 つの Android 端末間でのピアツーピア通信が可能になる点が特長である．「かざす」とは，端末に搭載された NFC チップ同士を物理的に近づけることをいう．一方，NFC の通信速度は 100〜400kbps と低速であり，送受信できる実用的なデータ容量としては数キロバイト程度である．また，1 つの接続につき一度しかデータを送受信できないため，匿名属性認証に必要な対話的な通信環境の構築が困難という問題点がある．
- Bluetooth
  Bluetooth とは 2.45GHz 帯の電波を利用する短距離通信規格である．Bluetooth は通信距離が 10m 程度，通信速度が 1Mbps(High Speed プロファイル利用時最大 24Mbps) と NFC に比べて長距離かつ高速な通信が可能である．また，ソケット接続を用いた対話的な通信も簡単に実装可能である．
- NFC-Bluetooth ハンドオーバ
  NFC-Bluetooth ハンドオーバとは，NFC によって通信相手を決定しデータの送受信は Bluetooth を用いる仕組みである．Bluetooth 接続を確立するためにはペアリング作業が必要であるが，ペアリングに必要な情報を NFC を用いて送信することで接続を自動化できる．

3.　Android 端末間における匿名属性認証システムの実装

3.1　実装方針

[4] の実装では IP ネットワークを用いて通信を行っていたが，本実装では端末同士で直接無線通信を行う．Android 端末間において直接無線通信を行う手段として NFC，Bluetooth，WiFi Direct が利用できるが，端末同士をかざすことによって接続が確立できる点がユーザ・検証者の両者にとって利便性が高いことから，本実装では NFC を利用する．しかしながら，NFC は 1 つの接続につき一度しかデータを送受信できないため，対話的な通信の実装が困難であり，NFC のみで匿名属性認証プロトコルを実行することが難しい．そこで，NFC-Bluetooth ハンドオーバを用いて Bluetooth のソケット接続を確立できるようにする．Bluetooth のソケット接続は Java Socket API をサポートしており，従来のシステムにおける通信部分の実装を容易に移植することができる．

Android 端末間における匿名属性認証システムの概要を図 3 に示す．システム利用の流れとして，まず端末同士をかざすことによって NFC の接続を確立し，ユーザから Bluetooth 接続を行うための情報を送信する．この情報を送信するためには，ユーザが画面をタップする操作が必要である．次に，検証者がユーザから受け取った情報をもとに Bluetooth 接続要求を送信する．ここまでの手順が NFC-Bluetooth ハンドオーバである．NFC-Bluetooth ハンドオーバによって Bluetooth 接続が確立されるため，Bluetooth 接続を用いて匿名属性認証を行う．本研究では認証アルゴリズムとして AND relation を用いた匿名属性認証を利用する．署名生成機能は [4] で実装済みであるため，本実装では [4] と同様の手法を用いて署名検証機能を実装する．通信部分については，ユーザ・検証者双方に NFC-Bluetooth ハンドオーバを用いて新たに実装する．

本実装におけるアプリケーションの構成を図 4 に示す．このアプリケーション構成はユーザと検証者で共通しているため，単一のアプリケーションでユーザおよび検証者の双

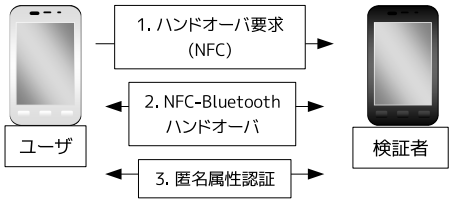

図 3 Android 端末間における匿名属性認証システムの概要

方の機能を利用することができる．本実装では，NFC でハンドオーバ要求を送信した端末がユーザになるよう設定している．ユーザインターフェース・NFC・Bluetooth・Java Socket API は Android OS に標準で提供されている API を利用する．NFC-Bluetooth ハンドオーバの実装には，EasyNFC ライブラリ [10] を用いる．匿名属性認証 API で入出力するデータを Java Socket API に渡す際には，JSON と Java オブジェクトの相互変換を行うため google-gson ライブラリ [11] を利用する．

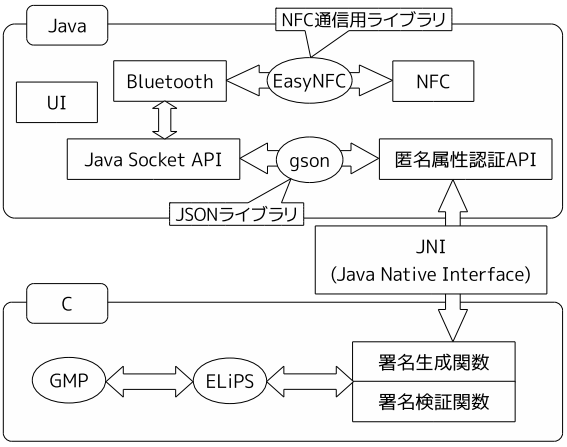

図 4 アプリケーションの構成

3.2 NFC-Bluetooth ハンドオーバ

NFC-Bluetooth ハンドオーバの実装で用いる EasyNFC ライブラリは，Android の NFC に関連する API を拡張したものであり，NFC-Bluetooth ハンドオーバを行うメソッドも実装されている．図 5 および以下に NFC-Bluetooth ハンドオーバの流れを示す．

(1) 端末同士をかざす
(2) ユーザ端末が MAC アドレス・UUID・使用する Bluetooth チャネルを NFC タグに埋め込んで検証者端末に送信する．
(3) 検証者端末は受け取った MAC アドレス・UUID・Bluetooth チャネルの情報に基づいて Bluetooth 接続を確立する．

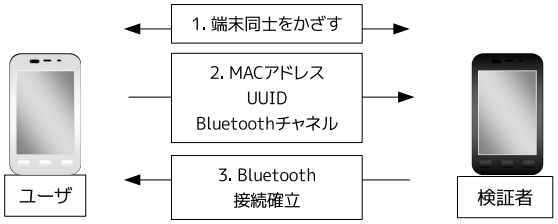

図 5 NFC-Bluetooth ハンドオーバの流れ

手順 2 で送信している UUID とは，NFC 接続を確立するたびにランダムに生成される 128 ビットの一意な識別子である．Android で UUID を自動生成する場合，128 ビットのうち 6 ビットがバージョン番号等の固定値に使用され，残りの 122 ビットが擬似乱数により生成される．

Bluetooth の API に接続先の MAC アドレス・UUID・Bluetooth チャネルを渡すことにより接続が確立される．同じ UUID を持つもの同士でのみ Bluetooth 接続が確立可能であり，同じ UUID を共有できる端末は NFC で通信している 2 端末のみであるため，意図しない端末への接続は回避できる．

3.3 ユーザ・検証者間の認証プロトコル

本節では匿名属性認証のプロトコルおよび各フェーズで送受信されるメッセージの内容について述べる．図 6 に匿名属性認証プロトコルの流れを示す．

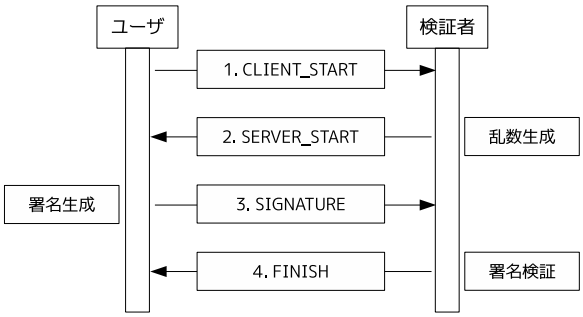

図 6 匿名属性認証プロトコルの流れ

匿名属性認証プロトコルは 4 フェーズで構成されており，相互にメッセージを対話的に送受信する．以下に，各フェーズで送信するメッセージの内容を示す．

(1) CLIENT_START
　フェーズ識別子:CLIENT_START
　使用する認証アルゴリズム番号
(2) SERVER_START
　フェーズ識別子:SERVER_START
　乱数
(3) SIGNATURE

フェーズ識別子:SIGNATURE
署名
認証に使用する属性カテゴリ
属性カテゴリに対する属性値

(4) FINISH
フェーズ識別子:FINISH
署名の検証結果

各フェーズで送信するフェーズ識別子は，メッセージがどのフェーズで生成されたかを示す．メッセージ内のフェーズ識別子がユーザと検証者がそれぞれ保持するフェーズ情報との不整合を起こした場合にエラーとして認証を異常終了させる．

認証アルゴリズムが複数存在する環境では，ユーザと検証者間で使用する認証アルゴリズムが何であるかという情報を共有する必要がある．そこで，認証アルゴリズムの番号をあらかじめ定義しておき，CLIET_START フェーズでユーザから検証者に認証開始要求として認証アルゴリズムの番号を送信する．

CLIENT_START のメッセージを受け取った検証者は，Java の SecureRandom クラスを用いて 64 バイトの乱数を生成する．この乱数は認証毎に生成され，次の認証では同じ乱数を使用しない．これは，第三者の盗聴による再送攻撃を防止するためである．毎回同じ乱数を使用した場合，乱数は署名の生成および検証に使用されるため，攻撃者がユーザの署名を盗聴し再利用することにより不正に認証成功する危険性がある．生成した乱数は SERVER_START フェーズでユーザに送信する．

SIGNATURE フェーズでは，ユーザが属性証明書，属性カテゴリ，属性値，検証者から受け取った乱数を用いて署名を生成し検証者に送信する．

最後に，FINISH フェーズで検証者が署名を検証し，結果を検証成功または検証失敗の真偽値でユーザに送信する．

3.4 送受信するメッセージの JSON 表現と Java オブジェクトとの相互変換

匿名属性認証の各フェーズで処理するメッセージは，以下に示す SigMsg クラスを用いて表現する．セッターおよびゲッターは省略している．

```
public class SigMsg {
    public static final int PHASE_INIT = 0;
    public static final int PHASE_CLIENT_START = 1;
    public static final int PHASE_SERVER_START = 2;
    public static final int PHASE_SIGNATURE = 3;
    public static final int PHASE_FINISH = 4;

    private int phase;
    private int gsMethodNumber;
    private byte[] sv_random_key;
    private byte[] cl_signature;
    private int[] cl_attributes;
    private int[] cl_categories;
    private boolean isValid;
}
```

メンバの命名規則として，ユーザ側で生成するものは接頭辞 cl_，検証者側で生成するものは接頭辞 sv_を付与している．ユーザと検証者の両者で値が変更されるものは接頭辞をつけていない．この SigMsg クラスでは，フェーズ番号を定数として定義し phase メンバを用いてメッセージがどのフェーズで生成されたかを保持する．gsMethodNumber は使用する認証アルゴリズムの番号を示している．sv_random_key は SERVER_START フェーズで生成される乱数，cl_signature，cl_attributes，cl_categories はそれぞれ署名，属性値，属性カテゴリを保持する．isValid には FINISH フェーズで署名検証結果が真偽値で格納されるが，検証者側が対応していない認証アルゴリズム番号を受け取った場合やユーザが署名生成に失敗した場合など，アルゴリズム内部でのエラー通知としても利用する．

各フェーズで SigMsg から生成したオブジェクトの内容を書き換え，送信する際に JSON 形式に変換する．また，JSON 形式のメッセージを受信した場合，元のクラス定義に従ってオブジェクトに変換しプログラム内で利用する．この Java オブジェクトと JSON の相互変換には google-gson ライブラリを利用する．google-gson ライブラリは Java のプリミティブ型およびプリミティブ型を要素とする配列を持つクラスから生成したオブジェクトを，JSON 形式に変換することができる．

実装システムでは，メッセージ受信時には Bluetooth ソケットを用いて受け取った JSON データを Java オブジェクトに変換して匿名属性認証 API に渡す．メッセージ送信時には Java オブジェクトから JSON データに変換し Bluetooth ソケットに渡す．Java Socket API と匿名属性認証 API の間に google-gson ライブラリを配置することにより，ネットワークと認証アルゴリズムの実装を切り分けることができプログラムの変更に対して柔軟に対応できる．

4. 実験と評価

4.1 概要

実装したシステムの動作確認および評価のために，Android OS を搭載したタブレット PC を 2 台用いて，認証操作を 10 回試行しその平均時間を算出した．図 7 中に示すように，実験において，NFC-Bluetooth ハンドオーバに要した時間，Bluetooth 接続の確立に要した時間，署名生成時間，署名検証時間，匿名属性認証プロトコルの実行に要した時間を測定した．匿名属性認証プロトコルの実行に要した時間には，通信や署名生成，署名検証，JSON メッセージの変換時間も含まれている．

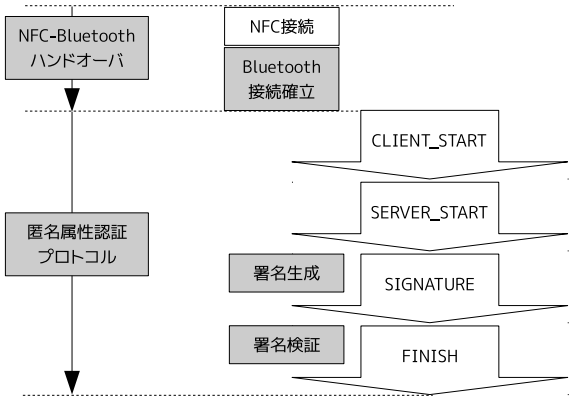

図 7　実験の概要

4.2　実験環境

表 1 に示すように，ユーザおよび検証者の実験環境には同一モデルのタブレット PC を用いた．

表 1　ユーザおよび検証者の実験環境

デバイス名	Nexus7 (2012)
OS	Android 4.3
CPU	NVIDIA Tegra3 1.3GHz
RAM	1GB
Bluetooth バージョン	3.0+EDR
多倍長演算ライブラリ	GMP 5.0.0

4.3　結果・考察

実験結果として，表 2 に各処理の実行時間と全体の処理時間，表 3 に匿名属性認証プロトコルにおける署名生成および署名検証の実行時間を示す．

表 2　各処理の実行時間と全体の処理時間

NFC-Bluetooth ハンドオーバ [s]	約 0.87
匿名属性認証プロトコル [s]	約 0.98
全体処理 [s]	約 1.85

表 3　署名生成・署名検証時間

署名生成 [s]	約 0.23
署名検証 [s]	約 0.52

表 2 に示すように，NFC-Bluetooth ハンドオーバに要した時間が約 0.87 秒である．NFC-Bluetotoh ハンドオーバの処理において，Bluetooth 接続の確立に要した時間は約 0.73 秒であり，Bluetooth 接続の確立が NFC-Bluetooth ハンドオーバの実行時間の大半を占めている．

また，匿名属性認証プロトコルの実行時間が約 0.98 秒である．匿名属性認証プロトコルの処理において，署名生成時間が約 0.23 秒，署名検証時間が約 0.52 秒となっている．

ユーザが画面を操作してから認証結果を受信するまでの全体の処理時間が約 1.85 秒となっており，レスポンス速度の観点から本実装のシステムは実用的とは言い難い．しかしながら，NFC-Bluetooth ハンドオーバおよび認証プロトコルの実行時間の両者を高速化することによって，より実用的な処理時間での利用が可能になるものと考えられる．

5.　むすび

本研究では，Android モバイル端末間において NFC-Bluetooth ハンドオーバを用いて直接通信により匿名属性認証を行うシステムを実装した．処理時間の測定結果より，NFC-Bluetooth ハンドオーバに要した時間が約 0.87 秒，匿名属性認証プロトコルの実行時間が約 0.98 秒であった．ユーザが操作してから認証が完了するまでに 1.85 秒程度の時間を要し，実用的な処理時間ではないため，高速化が必要である．

今後の課題として，まず高速化が考えられる．アプローチとしては，端末間の接続方法を WiFi-Direct や NFC 単体での実装へ切り替えることや，認証アルゴリズムをより高速なものに変更することが考えられる．また，本実装ではユーザと検証者のユーザインターフェースが同一であり実用性に乏しいため，より実用性を考慮したインターフェースの実装が必要である．さらに，他の認証方式の組み込みや MAC アドレスによる追跡を不可能にすることも課題として挙げられる．

参考文献

[1] A. Sudarsono, T. Nakanishi, and N. Funabiki, "Efficient Proofs of Attributes in Anonymous Credential Systems Using a Pairing-based Accumulator", コンピュータセキュリティシンポジウム (CSS), pp. 801-806, 2010-10.
[2] 大林弘樹, 中西透, 舩曳信生, "Web サービスにおけるプロキシを用いた匿名認証システムの実装", コンピュータセキュリティシンポジウム (CSS), pp. 163-168, 2008-10.
[3] 濱田雄治, 中西透, 舩曳信生, "Web サービスにおける匿名属性認証システムの実装", コンピュータセキュリティシンポジウム (CSS), pp. 60-65, 2011-10.
[4] 三嶋徹, 中西透, 舩曳信生, 渡邊寛, "匿名属性認証システムの Android 携帯端末への実装", コンピュータセキュリティシンポジウム (CSS), pp. 146-152, 2012-10.
[5] M. Akane, Y. Nogami, and Y. Morikawa, "Fast Ate pairing computation of embedding degree 12 using subfield-twisted eliptic curve", IEICE Trans. Fundamentals, vol. E92-A, no. 2, pp. 508-516, 2009.
[6] Java Native Interface, http://docs.oracle.com/javase/7/docs/technotes/guides/jni
[7] Android NDK, http://developer.android.com/tools/sdk/ndk/index.html
[8] droid-wrapper, https://github.com/tmurakam/droid-wrapper
[9] GMP, http://gmplib.org/
[10] EasyNFC, https://github.com/Mobisocial/EasyNFC
[11] google-gson, https://code.google.com/p/google-gson/