# マイナンバーカード利用者証明用電子証明書を用いた電子署名アプリケーションの開発

金子曜大†
明治大学総合数理学部　先端メディアサイエンス学科　菊池研究室†

## 1　はじめに

電子署名法[2]が施行され，電子的に署名した文書が法的にも認められている．しかしながら，従来の電子署名はファイルに秘密鍵を格納する必要があり，秘密鍵がコピーされる恐れがあった．そこでマイナンバーカードに着目する．マイナンバーカードには，日本国民の各個人に与えられた個人番号が記されているだけでなく，公開鍵電子証明書や 2048bit の RSA 秘密鍵が内蔵されている．マイナンバーカード内の秘密鍵を用いることで，カード内の秘密鍵は漏洩する恐れがなく，この問題を解決することができる．また，マイナンバーカードの利用者証明用電子証明書を用いると，安全性が高いだけではなく，自治体から正式に認証された本人認証のある署名をすることができる．

本研究ではマイナンバーカードを用いた電子署名アプリケーション Captain Signer を開発した．このアプリケーションの開発と，その評価について述べる．

## 2　Captain Signer の開発

### 2.1　OpenSC

本実装ではマイナンバーカードと通信する API として OpenSC を用いた．OpenSC は PKCS#15(RSA セキュリティにより公開されている公開鍵暗号の標準文書)と互換性のある IC カード，およびその他の暗号トークンの使用が可能．本実装では OpenSC を Java から外部コマンドとして呼び出している．pkcs15-crypt コマンドと pkcs15-tool コマンドを用いる．　OpenSC の pkcs15-crypt はスマートカードに保存されている秘密鍵を用いて，電子署名を計算したり，データを復号するなどの暗号化操作を実行する．pkcs15-tool はスマートカードに格納されている鍵や証明書などを読むことができる．公開鍵情報を pem ファイルに書き込むコマンドの例を図 1 に示す．

```
pkcs15-tool –read-public-key 1 > pubkey.pem
```

図 1　OpenSC コマンド例

### 2.2　実験環境

実験環境を表 1 に示す．

表 1　実験環境

| | |
|---|---|
| OS | OS X Yosemite 10.10.1 |
| メモリ | 8GB |
| 言語 | Java1.8.0 |

### 2.3　システム構成図

Captain Signer のシステム構成図を図 2 に示す．

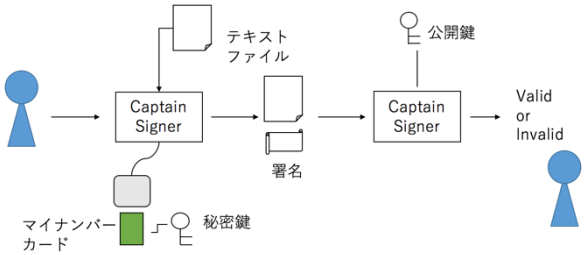


図 2　システム構成図

### 2.4　マイナンバーカードを用いた電子署名

公開鍵証明書から証明書の値を取り出し，テキストファイルの本文と証明書の値を結合．算出した値をSHA-256にかけて，ハッシュ値を取得．OpenSCのpkcs15-crypt -sコマンドを使い，マイナンバーカードの秘密鍵を利用して署名値を作成する．署名値はBase64 形式である．

マイナンバーカードには署名用電子証明書と利用者証明用電子証明書の２つが内蔵されているが，利用者証明用電子署名書を利用する．この時にマイナンバーカードに格納された秘密鍵が用いられる．

署名したファイルには，図４の様にテキストファイルの本文の末尾に公開鍵証明書情報と署名が追記される．BEGIN CERTIFICATEからEND CERTIFICATEまでが公開鍵証明書情報，BEGIN SIGNATUREからEND SIGNATUREまでが署名情報を表している．

署名の検証する際には、署名されたテキストファイルからopenssl rsautl -verifyコマンドを使い、署名値を復号。また、テキストファイルの本文と公開鍵証明書の値を結合して、SHA-256 にかけてハッシュ値を得る。復号した署名値とハッシュ値を比較して一致した場合は正しい署名値となる。

署名と，検証の実行例を各々，図３と図４に示す．

†Yodai Kaneko, Department of Frontier Media Science, School of Interdisciplinary Mathematical Science, Meiji University, Kikuchi Laboratory.

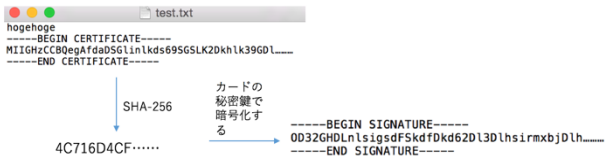


図３　署名実行例


図４　検証実行例

## 3　評価

### 3.1　処理速度の計測

Captain Signer の署名と検証にかかる処理速度をそれぞれ計測した．署名するテキストファイルの文字数 n を 0, 10000, 20000, …, 50000 文字と変更し、10 回ずつ計測した．その結果を図５，図６に示す．

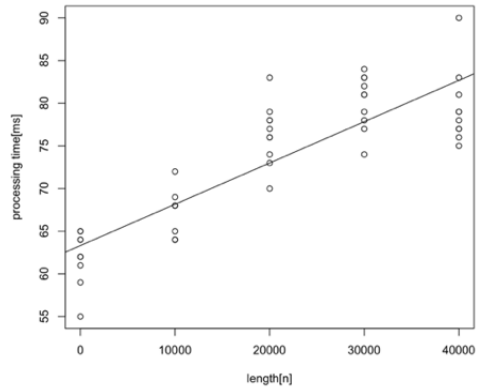


図５　署名の処理速度

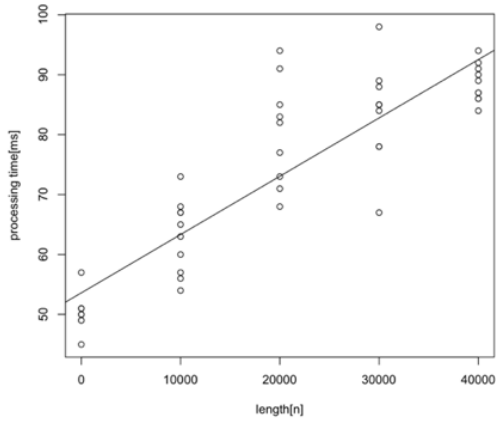


図６　検証の処理速度

図５よりハッシュにかかる時間が 10000 文字につき 5ms，署名にかかる時間が 62.5ms であることが算出された．

また，署名するテキストファイルのサイズと署名時間の関係を計算した．$1^{-1}$MB で 103.5ms，$10^{0}$MB で 472.5ms，$10^{1}$MB で 4162.5s，$10^{2}$MB で 41062.5ms となる．

従来の署名ツールとして代表的な Acrobat との性能評価を以下の表にまとめる．

表２　性能評価

| | 秘密鍵の保管場所 | 署名者情報 | タイムスタンプ | 署名できるファイル |
|---|---|---|---|---|
| Acrobat | PC 上のファイル | 有 | 有 | PDF |
| Captain Signer | マイナンバーカード | 無 | 無 | テキストファイル |

## 3　おわりに

本研究ではマイナンバーカードを用いた電子署名システムを試験実装した．従来のファイル上の秘密鍵を利用する場合と比べると，秘密鍵をコピーされる脅威をなくすことに成功した．署名者情報やタイムスタンプ等の情報の提示，テキストファイル以外に対して署名をすることを今後の課題とする．

## 参考文献

[1] 黒澤馨, 尾形わかは : 現代暗号の基礎数理, pp.106-107, 2004.

[2] 法務省電子署名法の概要と認定制度について (http://www.moj.go.jp/MINJI/minji32.html，2016 年 12 月参照)

[3] OpenSC(https://github.com/OpenSC/OpenSC，2016 年 9 月参照)