社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－MBL－32 (7)
2005－UBI－ 7 (7)
2005／3／17

# モバイル環境への適用を想定したPKI設計方針について

清本 晋作† 田中 俊昭†

† KDDI研究所 〒356-8502 埼玉県上福岡市大原2-1-15
E-mail: †{kiyomoto,tl-tanaka}@kddilabs.jp

**あらまし** 近年, 携帯電話端末に対してクライアント公開鍵証明書を発行し, モバイル環境における公開鍵認証基盤(PKI)を整備しようとする試みがなされている. 携帯電話を利用しているユーザの数は, 数千万人という膨大な数であり, またモバイル特有の制約条件から, PCと同様の構成をそのまま適用することはできない. 本稿では, PKIを設計する際に必要な, 公開鍵証明書の有効期限などの各種パラメータについて, 現在のユーザ数や環境を想定した場合の最適値を導出する. 特に, 系全体の通信量を決定する要因となる失効情報提供手法について比較を行ない, 最適な失効情報提供手法を検討する. また, 系全体の計算時間の定式化を図ることで, 計算時間が極小となる有効期限が必ず存在することを示すとともに, 現状の携帯電話環境において, PKIを導入した場合の最適な鍵長, 有効期限について導出を行なう.

**キーワード** 公開鍵認証基盤, PKI, 携帯端末, 証明書失効情報管理, 最適化

# Design Principle of PKI for Mobile Services

Shinsaku KIYOMOTO† and Toshiaki TANAKA†

† KDDI R & D Laboratories Inc., Ohara 2-1-15, Kamihukuoka-shi, Saitama, 356-8502, Japan
E-mail: †{kiyomoto,tl-tanaka}@kddilabs.jp

**Abstract** Recently, a public key infrastructure (PKI) for mobile terminals has been discussed to improve the security of mobile services. In the mobile PKI, public key certificates of mobile users are issued to use authentication of the user, and stored the certificates into their mobile terminals. Open problem is the scalability of the PKI, because the number of the mobile users is extremely larger than the number of users who uses a PKI for PC environments. This paper discusses a design principle of PKI for mobile services. Especially, we compare the method of providing certificates revocation lists on mobile service environments, and suggests the most efficient method. Furthermore, we show how to calculate the minimum computational cost of a general public key based authentication method of a whole system, and estimate feasible lengths of public/secret keys and validity.

**Key words** Public Key Infrastructure, PKI, Mobile Terminal, Certificate Revocation Management, Optimization

## 1. はじめに

携帯電話を利用したインターネットサービスの爆発的な普及に伴い, コンテンツの有料配信サービスが立ち上がり始めている. このような背景から, より安全で信頼性の高いモバイルコマースの実現と, オープンな標準技術によるモバイルコマースの普及促進を目的として, PKI技術のモバイルコマースへの適用が注目されている. 特に近年, 携帯電話端末に対してクライアント公開鍵証明書を発行し, モバイル環境における公開鍵認証基盤(PKI)を整備しようとする試みがなされている. 携帯電話を利用しているユーザの数は, 数千万人という膨大な数であり, またモバイル特有の制約条件から, PCと同様の構成をそのまま適用することはできない. 例えば, 解約率の問題がある. 携帯電話サービスでは, 解約を行なうユーザが比較的多いため, PC環境に適用しているPKIなどと比べて, 公開鍵証明書の失効が頻繁に発生する. また, 携帯電話端末とPCとの性能差は, おおよそ10倍程度あり, この処理性能差を考慮してシステム全体を設計する必要がある. 本稿では, PKIを設計する際に必要な, 公開鍵証明書の有効期限などの各種パラメータについて, 現在のユーザ数や環境を想定した場合の最適値を導出する手法について検討する. そして, 具体的な数値を用いて最適なパラメータを例示する. 特に, 通信量の観点からは, PC環境のおいて議論していた既存研究を基に, モバイル環境を想定したパラメータ設定の場合における最適な失効情報取得手段について論じる. また, 計算量の観点からは, PKIをサーバ-クライアント認証のみに使用する場合の公開鍵/秘密鍵の最適な有効期限及

び鍵長を導出する方法を提案し, 系全体の計算時間が極小値となる有効期限が必ず存在することを示すと共に, モバイル環境を想定したパラメータを用いて最適値を導出する.

以下, 本稿の構成を示す. 第 2 章では, 本稿の議論の前提となるモバイル環境における PKI について述べる. 第 3 章では, 通信量の観点から最適な PKI の設計方針について検討した結果を述べる. 第 4 章では, 計算量の観点から最適な有効期限及び鍵長を導出する方法を提案すると共に, 具体的なパラメータにおける最適な有効期限を例示する. 最後に第 5 章にて本稿のまとめを行なう.

## 2. モバイル PKI

### 2.1 モバイル環境における PKI フレームワーク

PKI 技術は, 様々な観点から検討がなされているが, その運用面においても, 検討すべき点が多々ある. さらに, モバイル環境では, 有線系のサービスと比較して, 考慮すべき制約も多い. しかし, 携帯電話端末や, その通信環境の飛躍的な性能向上により, PKI 技術の利用可能性が徐々に見えつつある. Mobile Electronic Transaction(MeT) [1] では, PKI によるクライアント認証が可能な安全なモバイルコマースの共通フレームワーク構築を目標の 1 つとして活動している. また, モバイル IT フォーラム [2]MC 部会においても, 携帯電話端末へのクライアント証明書発行を前提としたモバイルコマース方式の検討が進められている [9].

従来, WAP (現 OMA) [3] などではクライアントの処理負荷を考慮して, 暗号化、署名などの一切の公開鍵暗号アルゴリズムの処理をネットワーク側にある GW(ゲートウェイ) サーバで処理するモデルが検討されていた [4]. しかし, 端末の性能向上と, エンドーエンドでのセキュリティ確保の必要性から, 現在では携帯電話端末上で, 公開鍵暗号アルゴリズムの処理を行う方式が主流となりつつある. 公開鍵暗号を利用する上で最も重要な処理に, 公開鍵証明書の検証処理がある. 公開鍵証明書の検証処理は, その失効確認と同時に, 自身の保有する信頼の基点 (ルート証明書) に提示された証明書がチェーンされているかどうか (証明書の連鎖が繋がっているかどうか) を確認するものである. 最新の携帯電話端末では, 既にこの検証処理が実装されている.

### 2.2 モバイル PKI を利用したサーバ/クライアント認証

モバイル環境において PKI が整備されれば, 携帯電話端末-サービス提供者間でのサービスの授受において, 公開鍵を用いた相互認証を行なうことが可能となる. 公開鍵を用いた相互認証手順のモデルは以下の通りである.

(1) 携帯電話端末とサービス提供者は公開鍵証明書を交換する.

(2) 携帯電話端末とサービス提供者は, それぞれ相手の公開鍵証明書を検証する. その際, 必要があれば認証局 (CA) に失効情報の問い合わせを行なう.

(3) 携帯電話端末とサービス提供者は, 自身の秘密鍵で演算した結果を相手に送付する.

(4) 携帯電話端末とサービス提供者は, 送られてきた演算結果を, 検証済み公開鍵を用いて確認する. 確認結果が正しければ相手を正当なエンティティと認証する.

各々は, 公開鍵の検証処理に加え, 自身の秘密鍵による演算 1 回, 相手の公開鍵による演算 1 回を行なう. 実際には, 上記の手順が相互認証プロトコルとして複数のフローに分割して実装され, いくつかの付加的な情報の送受信や演算が追加される. 本稿では, 上記の単純化したモデルを基に計算量の見積もりを行なう. また, 公開鍵証明書はある有効期限を持ち, 有効期限が切れる前に, 新しい公開鍵証明書に更新される. 更新処理は, オンラインで行なうことを前提とし, 現在の公開鍵証明書を用いて相互認証及びセキュアチャネルの構築を行ない, そのセキュアチャネルを使用して新たな公開鍵証明書の発行を行なうものとする.

### 2.3 公開鍵証明書の検証処理

公開鍵検証処理は、大きく信頼パスの検証と、失効確認の 2 つの処理に分けられる. 信頼パスの検証は, 指針の保有するルート証明書を信頼の基点として、証明書の署名を順次検証していく処理である. 失効確認処理は公開鍵証明書が有効であるか (失効していないか) を検証する処理である. 公開鍵証明書 (PKC) は, その有効期限内であっても, その証明書が失効していないかどうかの確認 (失効確認) を行う必要がある. 失効確認の技術としては, 大きく分けて, 失効リスト (CRL: Certificate Revocation List) を随時リポジトリに配布し, 有効性を確認する検証者がリポジトリから CRL を取得する手法 [5] [6] と, 認証時にその都度, PKC が失効していないかどうかを問い合わせるオンライン証明書状態確認手順 (OCSP: Online Certificate Status Protocol) [8] がある. これら 2 つの方式については, 証明書の母数, 失効の頻度などによって, それぞれ, 利点, 欠点が存在するため, 想定する利用形態によって使い分ける必要がある. また, CRL 方式では, 次回のアップデートする時間が NextUpdate に記載されるため, NextUpdate 時に, 各公開鍵証明書の検証者からの負荷集中が予想される. これを回避するため, いくつかの方式が提案されている [7]. 本稿では, モバイル環境を想定し, CRL 方式と OCSP 方式の通信量について比較を行なう.

## 3. 系全体の通信量の最適化

### 3.1 失効情報取得方式の通信量比較

ユーザにクライアント証明書を発行し, 公開鍵を使用して認証を行なう方式を考えた場合, 証明書の発行・失効, 認証プロトコル, 失効情報取得の各処理の合計通信量が系全体の通信量を決定する. このうち, 方式によって通信量に大きく差異が生じると考えられるのが, 失効情報取得処理に関する部分である. そこで本節では, 失効情報取得処理を通信量の観点から比較する.

菊池 [10] らは, 公開鍵証明書破棄率と, ユーザ数を与えたときの CRL のサイズを見積もり, OCSP が CRL に対して優位な条件を明らかにしている. また, 田中らは, 多くの CA が存在する場合において, 通常の CRL 方式と, Delta CRL 方式について, 通信量の比較を行なっている [11]. 田中らは, 通信量の比較から最適に発行間隔を設定すれば, Delta CRL 方式のほうが

優れていることを示した．ここでは，以上の議論を携帯電話端末環境に適用し，個々の携帯電話端末（のユーザ）に対してクライアント証明書を発行し，サービス提供者がクライアント証明書を利用してユーザ認証を行なう，という前提の下に，最適な失効情報提供手段について考察を行う．携帯電話端末環境における PKI は，以下の特徴を有する．

- クライアント公開鍵証明書を発行する C A の数は，きわめて小さい．
- クライアント公開鍵証明書の発行数は膨大である．
- クライアント公開鍵証明書が検証される条件は，ユーザが認証を要求するサービスを利用する場合であり，その利用頻度は，インターネットにおけるサービス利用頻度と同等か，やや大きい．

以上の条件を考慮して最適な方式を選択する必要がある．

### 3.2 CRL 方式と OCSP 方式の比較

本節では，代表的な失効情報取得方式である CRL 方式と OCSP 方式について，モバイル環境に適用した場合の，通信量の比較を行なう．まず，必要なパラメータとして，

初期ユーザ数:$u_0$
単位時間当たりのユーザ増加率: $\delta u$
単位時間当たりのユーザ解約率: $\zeta u$
ユーザ数: $U = u_0 + (\delta - \zeta)u_0$
ある不測の事態で，証明書が失効するユーザの率: $\alpha$
サービス提供者数: $m$
ユーザが単位時間当たりに利用するサービスの数: $q$

を定義し，ユーザ認証の場合の最適な失効状態確認手法について考察する．今議論を単純にするために，$\delta u \doteqdot \zeta u$ であるとする．すなわち，ユーザ数はほぼ一定であり，$U \approx u_0$ と近似できるものとする．あるサービス提供者が単位時間当たりに認証する回数の期待値は，$Uq/m$ である．従って，ある時間間隔 $T$ の間に，発生する検証処理は，各サービス提供者につき，$UqT/m$ 回となる．一方，ある単位時間に失効する証明書の数 s は，$s = \zeta u \cdot u_0 + \alpha u_0$ で与えられる．このうち，有効期限が切れた証明書は，失効リストに載せる必要がない．ユーザ数一定の条件下では，文献 [11] の議論から，CRL のデータサイズも一定であると考えることができる．従って，証明書の有効期限を $T_v$ とした場合，$s'$ を失効リストに記載する証明書の数として，以下の式が成立する．

$$(\zeta u + \alpha)u_0 = \frac{s'}{T_v}$$

よって，$s' = (\zeta u + \alpha)u_0 \cdot T_v$ である．一般に携帯電話サービスにおいては，$\zeta u \gg \alpha$ を仮定できるので，$s' \approx (\zeta u \cdot u_0) \cdot T_v$ と近似できる．従って，CRL のデータサイズは，シリアル番号 1 つあたりのデータサイズを $a$，署名やその他固定長の大きさを $b$ として，$d_{CRL} = a \cdot \zeta u \cdot u_0 \cdot T_v$ となる．以上のことから，CA とサービス提供者間で交換される失効情報に関するデータサイズは，CRL の有効期限 $T_C$ としたとき，システム全体で単位時間当たり

$$D_{CRL} = m \cdot d_{CRL}/T_C = \frac{ma \cdot \zeta u \cdot u_0 \cdot T_v + b}{T_C}$$

となる．一方，OCSP の使用した場合のデータサイズは，1 回の検証処理に要するデータサイズと認証回数の積になる．1 回の検証処理に要するデータサイズは，おおよそ署名のデータサイズで近似できるので，システム全体で交換されるデータサイズは，

$$D_{OCSP} = bu_0q$$

となる．一般に $b = 10a$ 程度であり，また $mau_0T_v \gg b$ であるので，$D_{CRL} = D_{OCSP}$ となる場合は，

$$m\zeta u\frac{T_v}{T_C} = 10q$$

を満たす場合である．例えば，単位時間を 1 日とすると，一般に携帯電話の解約率は月当たり 1% ～ 1.2% なので，$\zeta u = 0.004$ 程度となる．仮にサービス提供者を 1000 程度，$T_v/T_C = 50$ とすれば，ユーザが 1 日 20 回以上サービスを利用する場合には，CRL 方式のほうがシステム全体の通信量の観点から優れていると判断できる．

一方，クライアントである携帯電話端末が，サービス提供者のサーバを認証するサーバ認証の場合は，$\zeta u$ に相当する公開鍵証明書の失効確率とサービス提供者数の積は，全ユーザの単位時間当たりのサービス利用回数に比べて極めて小さい．従って，長い時間間隔で CRL を発行する方式が有効であると考えられる．

## 4. 系全体の計算量（時間）の最適化

### 4.1 公開鍵の鍵長と計算量の関係について

一般に公開鍵暗号方式は，鍵長が長くなるに従って，その安全性は向上する．ここで言う “安全性” とは，現在考案されているもっとも効率の良いアルゴリズムを用いて，公開鍵に対応する秘密鍵を手に入れるための計算量である．計算量は，ある計算機の性能を仮定すれば，処理時間に変換することが可能である．例えば，現在，共通鍵暗号 DES を全数探索により破る計算量（$2^{56}$）の処理時間は最速で 20 時間程度であり，この場合，$2^{57}$ の計算時間は 40 時間程度となる．上記の計算量と処理時間の関係を用いて，Lenstra らは，文献 [12] において，共通鍵暗号 DES を破る計算時間と公開鍵暗号方式を破る計算時間を，それぞれ以下のように定式化している．$EMY_{DES}(d)$ を DES を破るために必要な処理時間としたとき，ある鍵長 $d$ の共通鍵暗号方式を破るために必要な処理時間は，$EMY(d) = 2^{(d-56)} \cdot EMY_{DES}(d) \cdot \nu$ と記述される．$\nu$ は，攻撃の効率を評価するパラメータであり，ここでは議論を簡単にするために $\nu = 1$ とする．すなわち，鍵長 d が 1bit 増加するごとに，必要な処理時間は 2 倍になるとする．このとき，RSA などの古典的暗号方式の鍵長 $k$ を，$l$ ビットの数体篩法 (NFS) の計算時間 $L(2^l)$ に対して，

$$L(2^k)/EMY(d) \geqq L(2^{512})/EMY_{NFS}(L(2^{512}))$$

を満たすように取れば，鍵長 $d$ の共通鍵暗号と同等の安全性を持つと判断できる．$EMY_{DES}(d) = 5 * 10^5$ であるとき，

表 1　鍵長と有効期限の関係

| Symmetric | Key Len. | Validity |
|---|---|---|
| 56 | 417 | 10 hour |
| 57 | 440 | 20 hour |
| 58 | 463 | 40 hour |
| 59 | 488 | 80 hour |
| 60 | 539 | 160 hour |
| 61 | 566 | 320 hour |
| 62 | 594 | 640 hour |
| 63 | 652 | 1280 hour |
| 64 | 682 | 2560 hour |
| 65 | 713 | 5120 hour |
| 66 | 777 | 10240 hour |

$EMY_{NFS}(L(2^{512}))$ は $10^4 * 26 * P$ で与えられる．$P$ は，共通鍵暗号と公開鍵暗号の攻撃におけるコスト差を考慮して設定される値で，Lenstra らは，$P = 100$ と仮定している．なお数体篩法の計算量は，文献[12] によれば，鍵長 $k$ に対して $O(exp(((64/9)^{1/3} + o(1)) \cdot (logk)^{1/3} \cdot (loglogk)^{2/3}))$ である．

以下に，鍵長に対する安全性を担保しうる時間の対応関係を表 1 に示す．安全性を担保しうる時間は，DES を破る計算時間を基に，56bits 共通鍵が約 10 時間で破られるとの仮定で計算した．

### 4.2　公開鍵証明書の有効期限と暗号処理時間の関係について

前節では，Lenstra らの結果を基に鍵長に対する安全性を担保しうる時間を求めた．安全性を担保しうる時間は，公開鍵/秘密鍵の使用用途をリアルタイムな認証に限定すれば，公開鍵証明書の有効期限に置き換えることができる．なぜなら，リアルタイムな認証のみに用いるのであれば，有効期限内において公開鍵/秘密鍵が安全であれば十分だからである．一般に，RSA 暗号方式における公開鍵は，演算を高速化するために短い値が用いられる．これは，署名においては公開鍵による検証処理が署名生成処理より多く行われること，秘密鍵を知らない場合，適用できる高速演算手法が限定されることによる．従って，公開鍵による処理時間は極めて短く，かつ鍵長に対してほぼ一定と考えることができる．一方，秘密鍵は，鍵長とほぼ同等の長さの値が用いられ，復号処理では，鍵長 $k$ に対して $O(k^3)$ の計算量が必要となる．携帯電話機 (MT) と PC 上で測定した，鍵長に対する RSA 演算処理の処理時間を図 1 に示す．携帯電話端末は Brew $^{TM}$ 搭載機，PC は Pentium4 3GHz のマシンを使用した．

安全性の根拠となる数体篩法の計算量は，前節で述べたとおり，鍵長 $k$ に対して，$O(exp(((64/9)^{1/3} + o(1)) \cdot (logk)^{1/3} \cdot (loglogk)^{2/3}))$ であるが，$400 < k < 800$ の領域では $(loglogk)^{2/3}$ はほぼ一定と考えることができるので，公開鍵証明書の有効期限 $T_v$ と秘密鍵処理の処理時間 $T_s$ に関して，以下のような関係式が成立する．

$$T_s = c_2 \cdot e^{c_1 \cdot (logT_v)^3} + c_3$$

図 2 に鍵長から算出した有効期限に対して，実測した秘密鍵処理，公開鍵処理の処理時間をプロットしたグラフを示す．

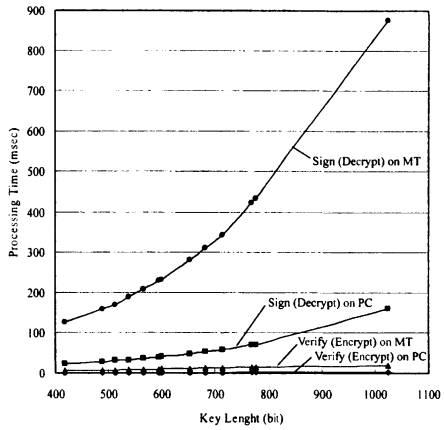


図 1　鍵長に対する RSA 処理時間

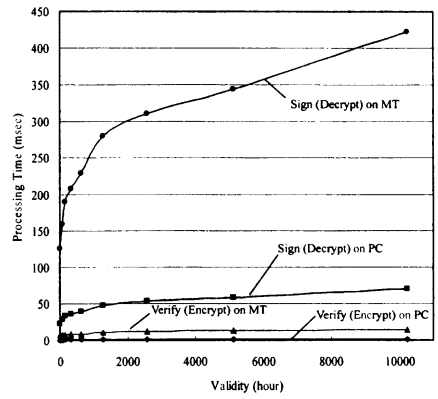


図 2　有効期間に対する RSA 暗号処理時間

グラフに対して，上記の関係式をフィッティングすることにより $c_1, c_2, c_3$ の各値を求めると，MT においては $c_1 = 0.00025$, $c_2 = 42.8$, $c_3 = 71.7$, PC においては $c_1 = 0.00027$, $c_2 = 5.96$, $c_3 = 19.0$ という値が得られた．これらの値が，公開鍵証明書の有効期限と暗号処理時間の関係を定義するパラメータである．この値は，攻撃手法の進歩や，端末の処理性能の増加に伴って変化するため，設計時点でのデータを入力して導出しなければならない．なお，公開鍵処理に関しては，MT において 10msec，PC において 1msec で，有効期限に依存せず一定であると仮定する．

### 4.3　鍵生成処理時間と有効期限の関係について

鍵生成のコストは，鍵長の増大に伴って増加する．RSA 公開鍵暗号において，鍵生成処理は素数生成処理と 2 つの素数の乗算及び最大公約数生成などの演算から構成される．このうち，素数生成処理がもっとも処理時間を要する．$n = k/2$ ビットの素数生成処理は，多くの方式が提案されているが Miller-Rabin 法[13] による素数判定を用いた方式が最も効率的であるとされており，多くの実装において用いられている．Miller-Rabin 法

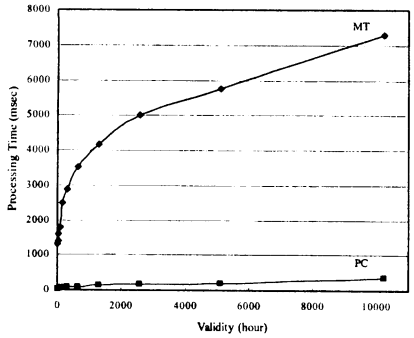


図 3　有効期間に対する RSA 鍵生成処理時間

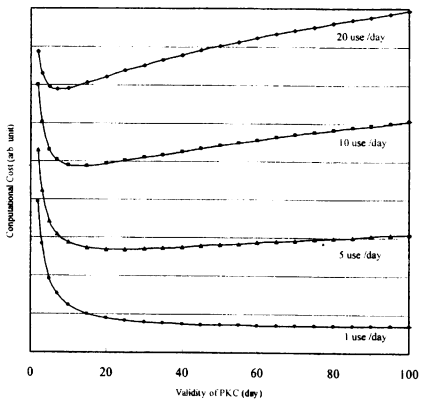


図 4　クライアント認証における PKC 有効期限と系全体の計算量

の演算時間において, ほとんどの時間を有するのはべき乗剰余演算であり, べき乗剰余演算の計算量は一般にビット長の 3 乗に比例する. 以上のことから, 鍵生成処理時間 $T_i$ と有効期限 $T_v$ の関係も, $T_i = c_2 \cdot e^{c_1 \cdot (logT_v)^3} + c_3$ のように記述できる. 実験結果を図 3 に示す. 実験結果から, 各値を求めると, MT において $c1 = 0.00026$, $c2 = 549.7$, $c3 = 82.0$, PC において $c1 = 0.00027$, $c2 = 9.38$, $c3 = 12.06$ となった. なお, 公開鍵発行処理時間は, 以上の鍵生成計算時間に加えて, DB 登録などのその他の処理に要する一定時間を加える.

### 4.4　クライアント証明書の最適な有効期限についての検討

本節では, 前節での計算結果を利用して, 公開鍵証明書の最適な有効期限の導出を試みる. まず, クライアント認証を公開鍵を用いて行なう場合を想定して議論を行なう. 公開鍵証明書の発行処理に要する処理時間を $C_{issue}$ とする. 実験環境で計測したところ $C_{issue}$ は, CA 公開鍵として 1024bits の公開鍵を使用した場合で平均 360.7msec である. なおこの値には, ユーザに関するプロファイル情報などを入力する時間は含まれていない. また PC 上での公開鍵証明書の検証処理は, 署名検証処理と失効情報取得処理の和で与えられるが, 署名検証処理の時間を $C_{verify}$ とする. 実験結果から, $C_{verify}$ は, 1msec 程度と見積もられる.

単位時間当たりに発行する証明書の枚数は, 新規加入者と再発行者の和となるため, $\delta u \cdot u_0 + (1 - \zeta u)\frac{u_0}{T_v}$ となる. また, 認証処理においては, クライアント端末上で署名処理 1 回, サーバ上で署名検証処理 1 回を行なう. 署名検証処理時間は, 鍵長に依存せずほぼ一定と仮定しているので $C_{verify}$ と一致すると仮定できる. なお発行処理は, 失効直前の公開鍵証明書を用いて認証を行い, その後新しい公開鍵証明書を発行すると仮定する. 公開鍵発行処理は, 鍵生成処理時間に加えて, 2000msec 程度処理時間を要するものとする. これらの処理に加えて, 失効情報生成及び確認処理時間を加えたものが, 系全体の処理時間となる. なお, データの通信時間は, 現状の通信帯域を考慮した場合, 鍵長の変化に対してほとんど差がないと仮定できるため, 省略する. 失効情報取得方式として, CRL 方式を使用する場合, CRL の発行処理時間を $C^{CRL}_{issue}$, 検証処理時間を $C^{CRL}_{verify}$ とすれば, $(C^{CRL}_{issue} + mC^{CRL}_{verify})/T_C$ となる. なお失効確認処理の計算時間は, 鍵長に非依存であるため, ＯＣＳＰを用いたとしても最適な鍵長を導出する本節の議論に影響を与えない. また, ユーザの公開鍵証明書の正当性確認処理の処理時間を $C^{CA}_{verify}$ とする.

系全体の計算量 $C$ は, 以下のように与えられる. このとき $T_v$, $T_C$ は日単位で与え, 計算時間は msec 単位で与えている. なお, 厳密には鍵長の変更に伴ってデータサイズが変化し, 通信処理時間が変化するがここでは一定として扱っている.

$$C = (\delta u \cdot u_0 + (1 - \zeta u)\frac{u_0}{T_v})(C_{issue} + C_{sign} + C_{verify} + C^{CA}_{verify}) + (C_{sign} + C_{verify} + C^{CA}_{verify})u_0 q + \frac{1}{T_C}(C^{CRL}_{issue} + m \cdot C^{CRL}_{verify})$$

ここで, 失効情報取得処理の処理時間など有効期限に非依存な項については省略する. また, $u_0 \gg m$, $\delta u = \zeta u$ であることを考えると, 以下の式を得る.

$$C \approx \left( \frac{C_{issue} + C_{sign} + C_{verify} + C^{CA}_{verify}}{T_v} + qC_{sign} \right) u_0 + A$$

A は定数である. $C_{sign}$ は, 4.2 節より $T_s = c_2 \cdot e^{c_1 \cdot (logT_v)^3} + c_3$ であるため, 上の式は, おおよそ $e^{(logT_v)^3}/T_v$ に比例する. 従って, $C$ は $T_v > 0$ の領域において必ず極小点を持つ. $q = 20$, $\delta u = \zeta u = 0.004$, $u_0 = 3.0 * 10^7$, $T_C = T_v/50$, $C_{issue} = 360.7$, $C_{verify} = 1$, $C^{CA}_{verify} = 10$, $m = 1000$ の各パラメータと前節までの実験結果を代入して $T_v$ に対してプロットすると, 図 4 のようになる.

図 4 から, 設定したパラメータにおいては, 利用者が 1 日 20 回サービスを利用する場合には 7 日程度, 1 日 1 回程度しかサービスを利用しない場合には, 80 日程度が最適な有効期限となる. また, 有効期限から 4.2, 4.3 節で議論した関数により, 最適な鍵長を導出できる. 設定したパラメータにおいては, ユーザが 1 日 1 回サービスを利用する場合, 655bits 程度である.

ここで, ＣＡに対する負荷集中について考察する. 80 日程度

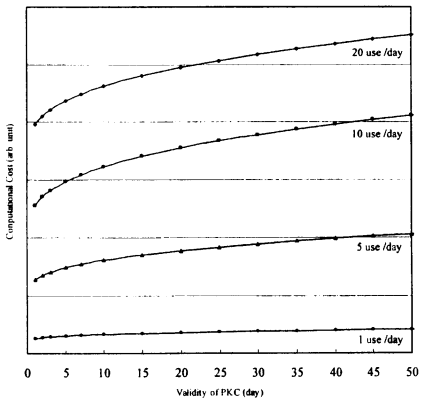


図 5　サーバ認証における PKC 有効期限と系全体の計算量

の有効期限で 3000 万程度のユーザが公開鍵証明書の再発行を受けるとすると，1 日当りの発行処理の件数は，37.5 万件である．仮に時間に対して発行処理件数が均等に分布すると仮定すると，1 秒当り 4-5 件の発行要求が CA に対して行なわれることになる．従って，CA サーバとして 10 台程度のマシンを用意し負荷分散を行なう必要があると考えられる．CA に対する負荷は，有効期限が長くなれば小さくなる．負荷集中の問題は多重化などによりＣＡの処理能力を向上させることで対処しうるが，現実的な CA の許容負荷も考慮して有効期限を設定することが望ましい．例えば，有効期限を 2 倍の 160 日にすることで，1 秒あたりの発行要求は 2-3 件まで減少させることができる．なお，以上の議論は，クライアント間での相互認証にも同様に適用することができる．

### 4.5　サーバ認証に対する検討

前節では，クライアント認証における最適な鍵長導出について議論を行なった．本節では，サーバ認証における鍵長に関して議論を行なう．サーバ認証についても，同様の方法を使用することにより，最適化が可能である．サーバ認証における基本モデルは，サービス提供者が自身の秘密鍵により署名を行い，その情報を携帯電話端末が検証するというモデルである．4.2 節で示したように，署名検証処理は，携帯電話端末上においても非常に軽量である．サービス提供者の処理能力を考慮すると，現状の有効期限及び鍵長における計算量と，最適化した場合の署名処理の計算量は大きく変化しない．また，クライアント証明書のように失効頻度が比較的高いことは想定されない．ただし，サービス利用におけるアクセス数は膨大であるため，最適化には一定の効果がある．サービス提供者の公開鍵証明書に関する系全体の計算量は以下の式で表される．

$$C \approx m\left(\frac{C_{issue}+C_{sign}+C_{verify}+C_{verify}^{CA}}{T_v}\right)+qu_0C_{sign}+B$$

B は定数である．$m = 1000$ とし，その他のサービス提供環境として 4.4 節と同様のパラメータを設定して $C_{issue}, C_{sign}, C_{verify}$ などにそれぞれ式を代入すると，図 5 のようになる．

モバイル環境ではサービス提供者数に比べてユーザ数が膨大な数になるため，有効期限 1 日以下が最適となる．しかし，サービス提供者の公開鍵は他の用途でも用いられる可能性もあり，認証用途に限定されたクライアントの公開鍵の場合と同一に議論することはできない．他の用途でも使用する場合には，十分に長い鍵長の公開鍵・秘密鍵を使用する必要がある．

## 5. ま と め

本稿では，モバイル環境—特に現在の携帯電話サービス環境を想定し，PKI を導入する場合の最適なパラメータ設定について議論を行なった．

通信量の観点からは，ＰＣ環境のおいて議論していた既存研究を基に，モバイル環境を想定したパラメータ設定の場合における最適な失効情報取得手段について論じた．また，計算量の観点からは，PKI をサーバ—クライアント認証のみに使用する場合の公開鍵/秘密鍵の最適な有効期限及び鍵長を導出する方法を提案し，系全体の計算時間が極小値となる有効期限が必ず存在することを示すと共に，モバイル環境を想定したパラメータを用いて最適値を導出した．本稿の基本方針として，電子署名とは違い長期保存を考慮しなくて良い認証処理においては公開鍵長を短くできる，との前提で最適化を検討している．ただし，認証処理と同時に短い有効期限を持つ公開鍵を使用して鍵共有を実行した場合には，その共有鍵の Forward Secrecy は必ずしも保証されるわけではないことを言及しておく．現実問題として，Forward Secrecy を破るような攻撃が行なわれる（攻撃者が存在する）かどうかは議論の分かれるところである．また，Forward Secrecy にも攻撃者の条件によっていくつかのレベルに別れるが [14] [15]，現実のサービス提供においてどの程度までの安全性を考慮しなければならないかについても，十分な議論が必要である．一般的な意味での Forward Secrecy を保証する 1 つの解決策としては，サーバ側の公開鍵を十分長くしておき，サーバ認証において鍵共有を実施するという方法がある．

また，本稿で示した手法は，PKI のみならず様々なセキュリティシステムに応用可能な手法であり，今後様々なセキュリティシステムへの適用を検討していきたいと考えている．さらに，モバイル環境向けの CA を実装することによって負荷試験などのスケーラビリティ評価を行なうと共に，楕円曲線暗号などの他の暗号方式についても評価を行なっていく予定である．

**謝辞**

本研究は，独立行政法人情報通信研究機構 (NICT) の委託研究「モバイルセキュリティ基盤技術の研究開発」の一環として行なわれた．

**文　献**

[1] Mobile electronic Transaction (MeT), http://www.mobiletransaction.org/.
[2] Mobile IT Forum, http://www.mitf.org/.
[3] WAP Forum, http://www.wapforum.org/.
[4] WAP Forum.: Wireless Transpotation Layer Security, WAP Specification, 2001.
[5] Draft revised ITU-T Recommendation X.509. ISO/IEC 9594-8: Public-Key and Attribute Certificate Frameworks (2000).
[6] D. A. Cooper.: A Model of Certificate Revocation, In Proc.

of ACSAC '99, December (1999).
[7] D. A. Cooper.: A More Efficient Use of Delta-CRLs, In Proc. of 2000 IEEE Symposium on Security and Privacy, May (2000).
[8] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams.: Online Certificate Status Protocol - OCSP. Technical Report RFC2560, IETF, June (1999).
[9] 田中, 関野, 菊地, 梅澤.: モバイルコマースにおける PKI の現状と課題 —mITF モバイルコマース部会認証WGの活動状況—, 情報処理学会研究報告, CSEC, No.22-028, 2003.
[10] H. Kikuchi, K. Abe and S. Nakanishi.: Dynamics Analysis on Public-Key Certificate and Limitations of Online Verification Protocols, Proc. of the 1999 Symposium on Cryptography and Information Security (SCIS'99), Vol.2, pp.615-620, 1999.
[11] 田中, 飯野.: PKI での証明書失効に必要な通信量の確率論的評価, 情報処理学会研究報告, CSEC, No.22-024, 2003.
[12] A. K. Lenstra, and E. R. Verheul.: Selecting Cryptographic Key Sizes, Journal of Cryptology, Vol.14, No.4, pp. 255-293, 2001.
[13] ANSI X 9.80.: American National Standard for Financial Services - Prime Number Generation, Primality Testing and Primality Certificates, American National Standard Institute, 2001.
[14] M. Bellare, P. Rogaway.: Provably secure session key distribution: the three party case, Proc. of 27th Annual Symposium on the Theory of Computing, ACM, pp. 57-66, 1995.
[15] M. Bellare, D. Pointcheval, P. Rogaway.: Authenticated Key Exchange Secure Against Dictionary Attacks, Proc. of Eurocrypt'00, LNCS, Vol. 1807, pp. 139-155, 2000.