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

2005－ＣＳＥＣ－30 (41)
2005／7／22

証明書検証サービスにおける認証パスキャッシュ方式の開発

橋本 洋子† 藤城 孝宏† 鍛 忠司† 羽根 慎吾† 手塚 悟†

†株式会社 日立製作所 〒212-8567 神奈川県川崎市幸区鹿島田 890
E-mail: †kumagai@sdl.hitachi.co.jp

あらまし 近年，電子商取引や，電子申請のための認証基盤として，多くの認証局が構築されてきている．日本政府でも，政府認証基盤（GPKI）を始めとする多数の認証サービスが開始されている．GPKI のように多数の認証局が連携する構成では，利用者による電子証明書の検証が複雑になるという問題がある．この問題解決の為，報告者らは，従来，利用者側で行ってきた証明書検証処理を，サーバ側で行うことにより，利用者の負担軽減と，検証処理の高速化を行うことを提案してきた．本稿では，この検証処理をさらに高速化する手法として，認証パスをキャッシュする方式を提案し，そのパスキャッシュを管理する方法の検討結果を報告する．

キーワード 公開鍵認証基盤，電子政府，PKI，X.509，証明書検証

Development of a certificate path cache method for Certificate Validation Service

Yoko HASHIMOTO† Takahiro FUJISHIRO† Tadashi KAJI† Shingo HANE† Satoru TEZUKA†

† Hitachi, Ltd. 890 Kashimada, Saiwai-ku, Kawasaki-shi, Kanagawa, 212-8567 Japan
E-mail: †kumagai@sdl.hitachi.co.jp

Abstract Recently, many certification authorities (CAs) have been built as the base of an electronic commerce and an electronic application. In the Japanese Government, many certification services including the Government Public Key Infrastructure (GPKI) have been launched. Because of such many CAs like GPKI, there is a problem that the certificate validation processing is complicated. Therefore, we proposed the Certificate Validation Server (CVS) which provide certificate validation service to users. Users can validate the certificate simply and fast by using CVS. We propose the certification path cache form as a technique which speeds up validation processing more. And we report a result of an examination of the way of managing certification pass cashes.

Keyword Public Key Infrastructure, Electronic Government, PKI, X.509, Certificate

1. はじめに

今日，情報セキュリティに対する関心が非常に高まっており，政府においても安全・安心なインフラの整備が加速されている．この情報セキュリティ対策の一つとして，公共分野，民間分野共に，公開鍵認証基盤（PKI）の利用が急速に広まっている．PKI は，ネットワーク上での相手方の認証や，通信内容の改ざんチェックを可能にするものである．

日本では，政府主導で PKI の導入を進めており，政府認証基盤（GPKI）[1]を始めとし，地方公共団体における組織認証基盤（LGPKI）[2]や，全国の住民に証明書を発行する公的個人認証サービス（JPKI）[3]が，既にサービスを開始している．これらの公的な認証基盤は，相互認証により連携し，電子申請・申告サービスに利用されている．

PKI を用いた認証においては，電子証明書の検証がとても重要となる．しかしながら，GPKI, LGPKI, JPKI の様に多数の認証局が連携する認証基盤では，証明書の検証処理において，利用者の負担が増大するという問題があり，IETF 等でオンラインで検証処理を実現するプロトコルやサービスが検討されている[4] [5]．我々の研究グループでも，この問題にいち早く着目し，利用者の証明書検証処理を代理で行う証明書検証サーバの研究を行ってきた[6] [7] [8] [9]．

本稿は，証明書検証サーバにおける証明書検証処理の高速化を行う手法について述べる．

以下，2 章では，多数の認証局が相互認証する構成における電子証明書検証処理を示し，検証処理が複雑になることを示す．3 章では，日本における公的な認証基盤の概要を示し，多数の認証局が相互接続していることを示すと共に，これを解決するために，公的認証基盤で採用されている証明書検証サーバの概要について示す．4 章では，証明書検証サーバにおける証明書検証処理を高速化する方法について示す．5 章では，高速化の手段である認証パスキャッシュ方法を実現するにあたり，認証パスの管理方法について検討した結果を示す．最後に，6 章において，本稿のまとめを述べる．

2. 証明書の検証

2.1. 単一認証局における証明書検証

PKI を用いた認証では，電子証明書の検証が重要となる．電子商取引や，電子申請等のやり取りにおいて，相手から電子証明書を受け取った場合には，以下の2つの項目をチェックすることにより，証明書が信頼できるものであることを確認する．

(1) 電子証明書が信頼できる認証局から発行されていること．

(2) 証明書が失効されていないこと．

例えば，図1のような単一の認証局からなる構成において，EE2 が，EE1 の証明書を検証する場合には，まず，予め安全な方法で取得している，CA（自身が信頼する CA）の自己署名証明書を用いて，EE1 証明書の署名検証を行う．これにより，EE1 証明書が，確かに CA から発行されたことを確認できる．さらに，CA のリポジトリから，証明書失効リスト（CRL）を取得し，EE1 証明書のシリアル No.が，CRL に記載されていないことを確認し，EE1 証明書が失効されていないことを確認する．ここで，CA の自己署名証明書を用いて，CRL の署名検証を実施し，確かに CA から発行された CRL であることを確認しておく．

これら全ての項目を確認できた場合のみ，EE1 の証明書は信頼できるものとなる．

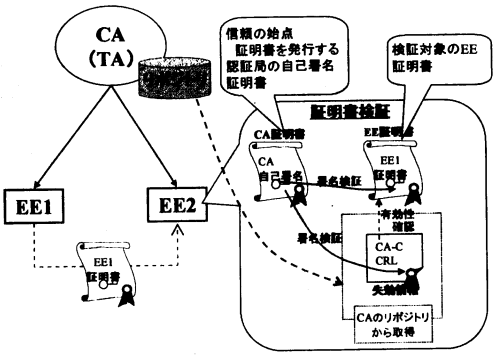

図1　単一認証局における証明書検証

2.2. 相互認証環境での証明書検証

現状では，国内外において，多数の認証局が存在している．また，異なる認証局から証明書を発行されている利用者の間でやり取りができるように，認証局間で相互接続し，連携している場合がある．

認証局間の相互接続は，相互認証によって行われており，各認証局間で相互認証証明書を互いに発行して，信頼のパスをつないでいる．

このように，多数の認証局が相互認証を行っている構成において証明書の検証を行うには，次のような2つの処理を実行する．[10] [11]

(1) 認証パス構築

自身の信頼する CA（トラストアンカ（TA））の自己署名証明書から，検証対象証明書まで，相互認証証明書等により信頼関係がつながっているかどうかを検索し，一連の証明書からなる認証パスを構築する．

(2) 認証パス検証

認証パス構築で得られた一連の証明書群について，各証明書の署名検証や，失効確認等を行い，信頼のパスが確かなものであることを確認する．

図2に，認証パス構築と検証の概要を図示する．

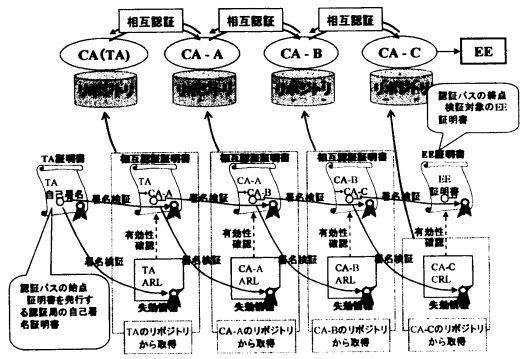

図2　認証パスの構築と検証

3. 公的認証基盤の概要

3.1. 認証局の構成

国内の公的認証基盤は，図3のように構成される．

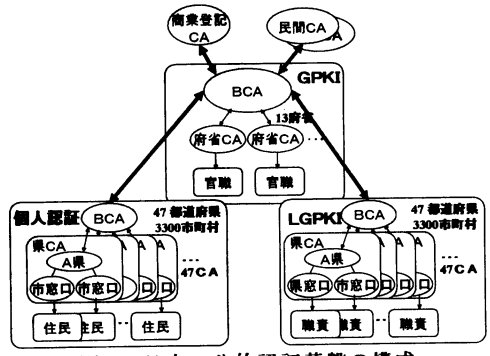

図3　日本の公的認証基盤の構成

GPKI では，各府省は各々ルート認証局を構築し，自府省の処分権者である官職（官職 End Entity（官職 EE））に対し，証明書を発行している．同じく，LGPKI では，地方公共団体の都道府県，市区町村の職責者に証明書を発行している．公的個人認証サービスでは，都道府県単位に認証局が設置され，市区町村を窓口として，住民に証明書を発行している．これらに加え，法務省の運営する商業登記認証局，民間の運営する認

証局がある．これらの認証局は，それぞれ GPKI ブリッジ認証局との間で相互接続を行っている．したがって，国内において，既に 100 を越す認証局が相互に接続され，運用されている．

3.2. 証明書検証サーバ

公的認証基盤は，100 を越す認証局が相互認証しており，とても複雑な構成となっている．このため，官職 EE や住民 EE が証明書検証を行うためには，とても複雑な認証パス構築・検証を行うことになる．例えば，住民 EE が，GPKI の官職 EE の証明書を検証する場合の例を図 4 に示す．住民 EE 側では，公的個人認証サービスのリポジトリ，GPKI の統合リポジトリから，相互認証証明書や CRL／ARL を取得し，複雑な認証パス構築を行う必要がある．このため，証明書検証における利用者側への負担が大きいという課題がある．

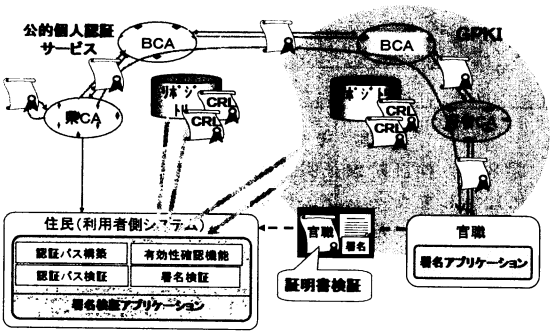

図 4　クライアントによる認証パス構築・検証

我々の研究グループでは，この問題にいち早く着目し，証明書の検証を利用者の代理で行い，利用者に対して検証結果を応答する機能を持つ証明書検証サーバを提案してきた[6] [7] [8] [9]．住民 EE が，官職 EE の証明書を検証する際に，証明書検証サーバを利用する場合の例を，図 5 に示す．

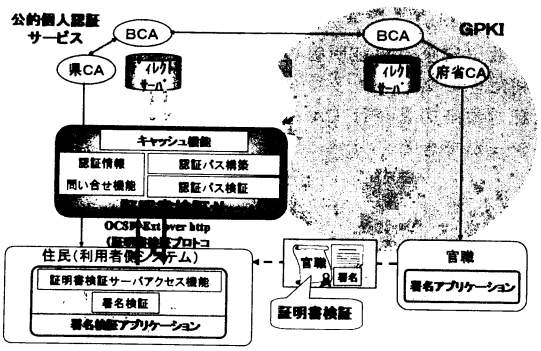

図 5　証明書検証サーバによる認証パス構築・検証

証明書検証サーバのように，利用者の負担を軽減する取り組みとして，例えば，IETF の PKIX WG において要求仕様がまとめられており，SCVP（Simple Certificate Validation Protocol）[5]，DVCS（Data Validation and Certification Server Protocols）[12]等の検討が行われている．

4. 証明書検証処理の高速化

我々の研究グループでは，以下のような手法により，証明書検証サーバにおける検証処理を高速化することを提案している．

4.1. 証明書，CRL／ARL のキャッシュ

証明書検証サーバにおいて，証明書ならびに CRL／ARL の情報をファイル単位にキャッシュする．キャッシュにある証明書や CRL／ARL を利用すれば，ネットワークを通じてファイルを取得する時間を短縮できるため，証明書検証を高速化することができる．（図 6）

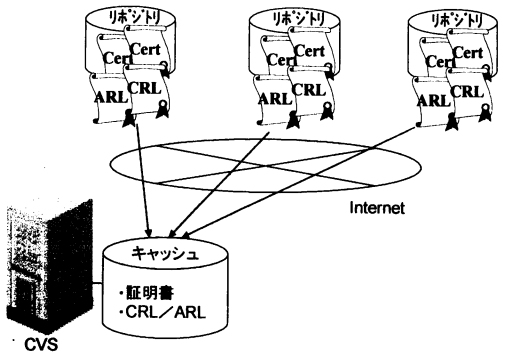

図 6　証明書，CRL／ARL のキャッシュ

4.2. 認証パス情報のキャッシュ

一般に，処理速度を高速化するためには，一度処理した処理内容の再利用をすることが行われている．証明書検証処理における再利用の範囲として，証明書や CRL／ARL のファイルキャッシュに加えて，再利用する情報の範囲を拡大することで，証明書検証処理の高速化を行うことが考えられる．

証明書検証処理は，認証パス構築処理と検証処理に分けることができる．このうち，認証パス構築処理は，相互認証を行う認証局の数に応じて，処理時間がかかるようになる．したがって，認証パス情報をキャッシュすることで，認証パス構築処理を高速化する手法を提案する．（図 7）

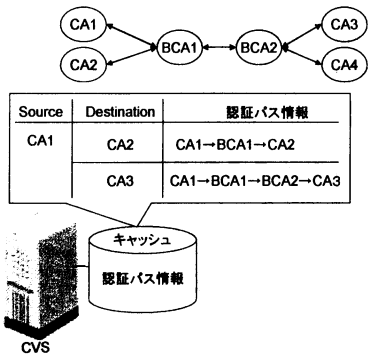

図 7　認証パス情報のキャッシュ概要

5. 認証パスキャッシュの管理

我々の研究グループでは，認証パスのキャッシュ機能を実現するにあたり，認証パスキャッシュの管理方法について，以下の検討を行った．

5.1. 認証パスキャッシュの範囲の検討

キャッシュに登録する認証パスの範囲としては，認証パス全体をそのままキャッシュする方法に加えて，共通的に使用できる部分だけを切り出しキャッシュする方法も考えられる．例えば，信頼点から，検証対象証明書を発行する認証局までの部分パス(EE 証明書を抜かした部分）は，その認証局が発行する EE 証明書で，共通で利用できる．

以下では，次の（a）（b）どちらの範囲で認証パスをキャッシュするのが効率的なのかを検討する．

（a）信頼点となる証明書から検証対象証明書までの範囲（認証パス全体）

（b）信頼点となる証明書から検証対象証明書の発行者の証明書までの範囲(検証対象証明書発行 CA まで）

（1）キャッシュのヒット率

ユーザから証明書検証要求があった場合に，その検証要求に対応する認証パスがキャッシュに登録されていれば(キャッシュにヒットすれば)，証明書検証が高速化できる．このため，キャッシュのヒット率が高い方が，再利用性が良いといえる．

ここで，下記のように前提条件をおいた場合のキャッシュヒット率は，下記（式 1）のように表すことができる．

【前提条件】

・トラストアンカは一箇所とする

・検証対象証明書の枚数：M（枚）

・証明書検証サーバが受け付ける検証要求の回数：N（回）

・検証 n 回目のキャッシュヒット率：P_n

・全ての EE 証明書に関して平均的に検証要求がなされるものとする

$$P_n = P_{n-1}^2 + (1 - P_{n-1})(P_{n-1} + (1/M)) \quad \cdots (式1)$$

この漸化式を解くと，下記（式 2）のようになる．

$$P_n = 1 - ((M-1)/M)^{n-1} \quad \cdots (式2)$$

この（式 2）に基づき，検証対象証明書枚数「M」を変化させた場合のキャッシュヒット率 P_n を求めると，下図のグラフのようになる．

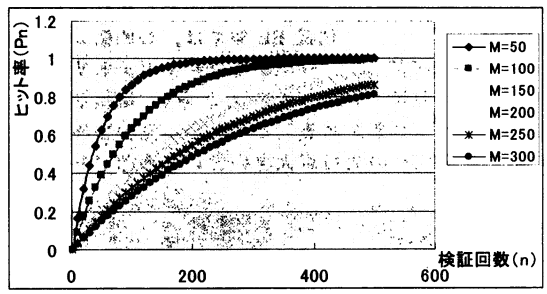

図 8　キャッシュヒット率

図 8 は，検証対象証明書枚数「M」が増えると，それに比例してキャッシュヒット率の増え方も減少することを示している．例えば，「M=50」の場合，検証要求を 81 回受け付けた時点でキャッシュヒット率が 8 割を越すのに対して，「M=250」の場合，検証要求を 323 回受け付けた時点でキャッシュヒット率が 8 割を越す．

公的認証基盤等の一般的認証基盤においては，「EE 証明書の数>>EE 証明書発行 CA の数」である．例えば，JPKI の住民 EE の数は 2005 年 2 月現在で約 76000 枚（『自治日報』(2005 年 3 月 11 日）より）であるのに対し，住民 EE 証明書を発行する都道府県 CA の数は 47 である．また，住民 EE 証明書の数は，さらに増大することが見込まれる．

これらより，キャッシュする範囲としては，なるべく母数（検証対象証明書の数）を少なくしたほうが，再利用性の高いキャッシュができる．すなわち，「(b) 検証対象証明書発行 CA まで」の方が，効率的なキャッシュができる．

（2）管理のしやすさ（管理するキャッシュの量）

証明書検証サーバにおいて，管理するパスキャッシュの量が少ない方が，リソースの有効活用ができ，更新，削除，検索等の管理がしやすいというメリットがある．

キャッシュする認証パス情報の量は，存在する EE 証明書または EE 証明書発行 CA の数により増減する．GPKI の省 CA をトラストアンカとして，JPKI の住民 EE を検証する場合を例にとった場合において，各パタ

ーンの最大キャッシュ件数は，以下のようになる．

（a）認証パス全体をキャッシュする場合

JPKI 住民 EE 証明書が約 7600 枚であるので，

[最大キャッシュ件数]＝約 76000（件）

となる．

（b）検証対象証明書発行 CA までをキャッシュする場合

都道府県 CA が 47 であるので，

[最大キャッシュ量]＝47（件）

となる．

（a）の場合，キャッシュするパスの件数が住民 EE 証明書の数にしたがって増えるため，将来的にますます膨大な数になる可能性が高い．これに比べて，（b）の場合は，最大でも検証対象証明書発行 CA の数までしか増えず，CA の数は変動が少ないため，数が増大する可能性は少ない．

すなわち，（a）よりも（b）の方が，管理するキャッシュの量は圧倒的に少ない数となるため，管理性に優れる．

（3）その他考慮すべき点

キャッシュした認証パスを検索する際に，検索のキーとなる情報は，下記のようになる．

「（a）認証パス全体」の場合には，検証要求で指定されたトラストアンカ証明書と，検証対象証明書の情報をキーにして，対応するパスがキャッシュされているかどうかを検索できる．

また，「（b）検証対象証明書発行 CA まで」の場合には，検証要求で指定されたトラストアンカ証明書と，検証対象証明書の発行者情報をキーにすれば，対応するパスがキャッシュされているかどうかを検索できる．この発行者情報は，検証対象証明書から容易に取得することができる．

これらより，（a）（b）どちらの場合でもパス検索情報を容易に取得できる．

表 1　認証パスキャッシュ範囲の比較検討表

	(a)検証対象証明書まで	(b)検証対象証明書の発行者まで
(1)キャッシュのヒット率	× 検証対象証明書の数に比例して悪くなる	○ CAの数は比較的少なく，増える可能性も低いので安定している
(2)管理のしやすさ（管理するキャッシュの量）	× 検証対象証明書の数に比例して増大する	○ CAの数は比較的少なく，増える可能性も低いので増大しない
(3)その他	検索のキーは，検証要求から容易に入手できる	検索のキーは，検証要求から容易に入手できる
総合評価	△	◎

上記（1）～（3）の結果をまとめると，表 1 のようになる．この表が示すように，「(2) 信頼点となる証明書から検証対象の発行者の証明書までの範囲」で認証パスをキャッシュする方が，再利用性に優れていると考えられる．

5.2. 認証パスキャッシュの管理方法の検討

認証パスは，証明書の集まりである．認証パスをキャッシュする方法としては，次の二通りの方法が考えられる．

（イ）証明書のデータそのものの集まりとしてキャッシュに登録し，管理する．

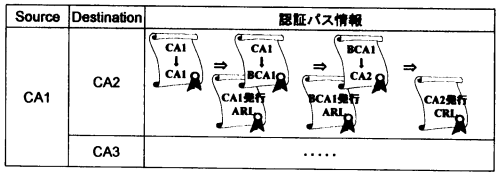

図 9　証明書そのものでパスを管理する方法

（ロ）各証明書を特定できる ID 情報の集まりを，パス情報としてキャッシュに登録し管理する．証明書データそのものは，別に管理する．

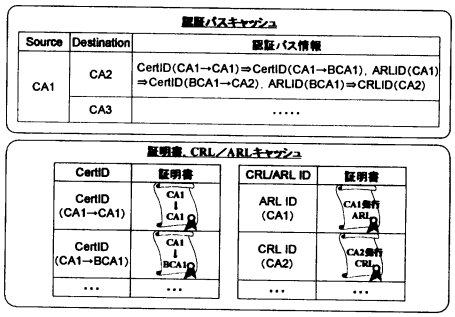

図 10　パス情報と証明書を別で管理する方法

これら二つのパスキャッシュ管理方法について，下記 3 つの観点で検討を行った結果を，以下に記す．

（1）効率的な情報のキャッシュ

（イ）の場合，各パスに，パスを構成する証明書全てを保持するため，同じ証明書や CRL／ARL を複数個キャッシュに登録することになる．例えば，GPKI の府省 CA をトラストアンカとして，JPKI の都道府県 CA までのパス 47 個がキャッシュされているときには，一つのパスにつき，4 つの証明書と，4 つの CRL／ARL を保持するので，最大で 188（＝4×47）の証明書と，188 の CRL／ARL を保持することになる．

これに対して，（ロ）の場合は，証明書と CRL／ARL は，パス情報とは別に管理するため，同じ情報を重複してキャッシュする事はない．例えば，上記と同様に，GPKI をトラストアンカとして JPKI までのパスをキャッシュする場合，50（＝47＋3）の証明書と，50 の CRL／ARL を保持すればよい．

CRLは，利用形態によって，容量が膨大になる可能性があるため，特に効率的にキャッシュに保持する必要がある．これらのことより，「（ロ）パス情報と証明書を別に管理する」方が，効率的なキャッシュ管理と言える．

（2）運用のしやすさ

認証パスをキャッシュするにあたり，キャッシュの更新や削除等の運用をすることを考慮する必要がある．

（イ）の場合，証明書だけの更新／削除，認証パスだけの更新／削除ができないという課題がある．

これに対して，（ロ）の場合，証明書だけの更新／削除，認証パスだけの更新／削除ができるため，柔軟な運用ができる．

以上の結果を，表2にまとめる．表2が示す様に，「(ロ)パス情報と証明書を別に管理する」方法で認証パスをキャッシュする方が，効率的で柔軟な運用ができると考えられる．

表2　認証パス管理方法の比較検討表

	(イ)証明書そのもので管理	(ロ)パス情報と証明書を別に管理
(1)効率性	×	○
(2)運用性	×	○
総合評価	×	○

また，証明書やCRL／ARLを特定するIDとしては，下記のような情報を用いることが考えられる．

- 発行者情報（issuer）
- 主体者情報（subject）
- シリアルNo.
- 発行者鍵識別子（authorityKeyID）
- 主体者鍵識別子（subjectKeyID）

6. まとめ

証明書検証サーバにおける高速化方法として，証明書やCRL／ARLのキャッシュに加え，認証パス情報のキャッシュを行うことを提案した．また，認証パスキャッシュ機能を実現するにあたり，認証パスの管理方法について，認証パスキャッシュの範囲と，管理方法の二つの観点での検討を行った．

検討の結果，認証パスキャッシュの範囲は，「信頼点とする認証局から検証対象証明書まで」とするよりも，「信頼点とする認証局から検証対象証明書の発行認証局まで」とする方が，キャッシュの再利用性が良いという結論に至った．また，認証パスの管理方法としては，「証明書そのものとしてパスを管理する」方法よりも，「パス情報と証明書を別で管理する」方法の方が，効率的で運用性が良いという結論に至った．

今後，PKIの更なる拡大にしたがって，証明書の検証処理がますます複雑になると想定できる．また，PKIを用いたアプリケーションの利用が普及するにつれて，証明書検証サーバに対する処理性能の向上がさらに要求されると予想される．

これらの要求に対応するために，認証パスキャッシュ機能を適切に実装し，性能向上だけでなく，効率的で運用性の良いものにすることは重要であると考える．

文　献

[1] 総務省，”「政府認証基盤（GPKI）について」”，http://www.gpki.go.jp/documents/gpki.html，2001

[2] 総合行政ネットワーク運営協議会，”「地方公共団体における組織認証基盤（LGPKI）」”，http://www.lgpki.jp/，2004

[3] 公的個人認証サービス都道府県協議会，”「公的個人認証サービス（JPKI）について」” http://www.jpki.go.jp/jpkiguide/index.html，2004

[4] A. Malpani、S. Galperin、M. Myers、R. Ankney and C. Adams: RFC 2560: "X.509 Internet public key infrastructure online certificate status protocol - ocsp" (1999)

[5] A. Malpani, R. Housley and T. Freeman: IETF Internet drafts: "Simple Certificate Validation Protocol (SCVP)", "draft-ietf-pkix-scvp-17.txt", 2005(work in progress)

[6] 藤城孝宏，五島裕庸，手塚悟，オフィスシステム研究会発表 OFS2001-81，” 政府認証基盤（GPKI）における証明書検証方式の提案”，(2002)

[7] 藤城孝宏，鍛忠司，手塚悟，コンピュータセキュリティシンポジウム 2002，” 複数PKIドメインにおける証明書検証の高速化方式の研究”，(2002)

[8] 藤城孝宏，鍛忠司，羽根慎吾，熊谷洋子，手塚悟，電子情報通信学会論文誌 D-I Vol. J87-D-I No.8，“証明書検証サービスの開発” (2004)

[9] 羽根慎吾，藤城孝宏，橋本洋子，手塚悟，情報処理学会研究報告 2005-CSEC-28，” X.509 証明書の高速認証パス検証アルゴリズム”（2005）

[10]ITU-T Recommendation X.509，"Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks "(2000)

[11]R. Housley, W. Ford, W Polk and D. Solo: RFC 3280: "Internet X.509 public key infrastructure certificate and Certificate Revocation List (CRL) profile" (2002)

[12]C.Adams, P.Sylvester, M.Solotarev, R.Zuccherato, "Internet X.509 Public Key Infrastructure Data Validation and Certification Server Protocols", RFC3029, 2001

[13]政府認証基盤相互運用性仕様書 http://www.gpki.go.jp/session/CompatibilitySpecifications.pdf (2001)

[14]カーライル・アダムズ，スティーブ・ロイド，” PKI公開鍵インフラストラクチャの概念，標準，展開”，（株）ピアソン・エデュケーション，2000

[15]小松文子，” PKIハンドブック”，（株）ソフト・リサーチ・センター，2000