１０－開－０６

データベース構築促進及び技術開発に関する報告書

複数認証局間における認証データベースの有効利用に関する調査研究

平成11年３月

財団法人　データベース振興センター

委託先　株式会社イフ・アドバタイジング

この事業は、競輪の補助金を受けて実施したものです。

序

データベースは、わが国の情報化の進展上、重要な役割を果たすものと期待されている。今後、データベースの普及により、わが国において健全な高度情報化社会の形成が期待される。

さらに海外に対して提供可能なデータベースの整備は、国際的な情報化への貢献および自由な情報流通の確保の観点からも必要である。しかしながら、現在わが国で流通しているデータベースの中でわが国独自のものは1／3にすぎないのが現状であり、わが国データベースサービスひいてはバランスある情報産業の健全な発展を図るためには、わが国独自のデータベースの構築およびデータベース関連技術の研究開発を協力に促進し、データベースの拡充を図る必要がある。

このような要請に応えるため、(財) データベース振興センターでは日本自転車振興会から機械工業振興資金の交付を受けて、データベースの構築および技術開発について民間企業、団体等に対して委託事業を実施している。委託事業の内容は、社会的、経済的、国際的に重要で、また地域および産業の発展の促進に寄与すると考えられているデータベースの構築とデータベース作成の効率化、流通の促進、利用の円滑化・容易化などに関係したソフトウェア技術・ハードウェア技術である。

本事業の推進に当たって、当財団に学識経験者の方々で構成されるデータベース構築・技術開発促進委員会（委員長　東海大学教授　上條史彦氏）を設置している。

この「複数認証局間における認証データベースの有効利用に関する調査研究」は、平成10年度のデータベースの構築促進および技術開発促進事業として実施した課題の一つで、当財団が「株式会社イフ・アドバタイジング」に対して委託実施したものである。この成果が、データベースに興味をお持ちの方々や諸分野の皆様方のお役に立てば幸いである。

なお、平成10年度データベースの構築促進および技術開発促進事業で実施した課題は次表のとおりである。

平成11年３月

財団法人　データベース振興センター

平成10年度　データベース構築・技術開発促進事業委託課題一覧

分　野		課　題　名	委　託　先
社　会	1	生命保険商品情報データベースプロトタイプ構築	(株) 平和情報センター
	2	ビジネス法定文書のサンプルフォームデータベースの構築	ジャスネットコミュニケーションズ (株)
	3	グループウェアによる ISO 9001品質文書管理データベースツールの構築	日本電子開発 (株)
地域活性化	4	地域対応型の農業技術情報データベース構築	アシストマイクロ (株)
技　術	5	登録ジオコードに基づく世界データベースシステムの開発	アジア航測 (株)
	6	複数の認証局間における認証データベースの有効利用に関する調査研究	(株) イフ・アドバタイジング
地域振興	7	十勝毎日新聞記事データベース構築	(株) ケーシーズ
	8	付加価値データベースシステムの開発に関する調査研究	(株) インタークラフト
	9	新規産業創出のための業種を融合した企業情報のデータベースの構築	(財) 相模原市産業振興財団
	10	地域ホームページに係わるリンク集整備および自動巡回システムと全文検索エンジンを利用した効果的検索機能の開発	ミネルバ (株)
	11	古墳・遺跡の指標となる土器データ検索パイロットシステムの作成	凸版印刷 (株)
	12	出雲古代遺跡デジタルデータベース構築	(株) 出雲王国
	13	インターネットを利用した四国地域の研究者データベースの構築	(財) 四国産業・技術振興センター
	14	長崎県観光写真素材データベースの構築	長崎メディアミックス協同組合
	15	沖縄伝統空手・古武道国際人材リソースデータベース	(株) アイエムアイコーポレーション

目　　次

1．　課題性の研究

1．目的

近年のインターネットブームは、一般消費者等による電子メールやWebの利用をも急速に増加させつつある。こうした中、盗聴やなりすましによるインターネットの不正使用が表面化しており、重大な問題となっている。

このため、盗聴されない秘匿通信や本人確認を行うディジタル署名等セキュリティの重要性が高まっている。暗号通信や署名通信について、メールではS/MIME、同時性通信ではSSLが標準プロトコルとなりつつある。これらのプロトコルではITU-T X.509公開鍵証明書を用い、公開鍵証明書は認証局で発行される。今後の、ＥＣを中心とした情報化社会の健全な発展、あるいはデータベースの円滑な推進にとって、公開鍵証明書を利用した通信は、極めて重要な役割を担うことになる。インターネットの利用が今後ますます広がり、ユーザが急増することを考えた場合、認証局の分散化や階層化といった対応が必要不可欠となる。

こうした視点から、オープンネットワーク環境下での分散・階層化された認証局間でのデータのセキュア通信に関する調査・研究を基に、オープンネットワーク上でのデータベース間通信の構築に寄与することを目的とする。

2．実施内容

本調査研究では、公開鍵証明書をターゲットとして以下の項目を主たる内容としている。

（1）データベースの形態

　　データベース間での更新を行う上で最適な配列の考察。

（2）データベース間のセキュリティの確保

　　データベース更新時の盗聴・改ざん・なりすまし防止の考察。

（3）データベースへのアクセスコントロール

　　データ更新サーバの認証方法の考察。

3. 実施体制

2．　複数認証局間における認証データベースの有効利用

1．　はじめに

近年のインターネットの発展により、WWW サーバの数も増大し、日本でのインターネットの利用者数を1000万人と伸ばすことになった［11］。WWW サーバの数の増大に伴い、情報量も増え、GOO［1］や Altavista［2］などの検索エンジンに代表されるデータベースの重要性が高くなってきている。上記検索エンジンなどのデータベース技術はインターネット普及にも一役かっているばかりでなく、単体の WWW サーバでの検索機能など、1企業の利用までに至っている。今後、インターネットでのデータベースの利用は増えていくのは確実である。

インターユーザの増加と共に、盗聴・改ざん・なりすましなど、最近のインターネットにおける不正アクセスが増大している。インターネットでは通信プロトコルとして TCP/IP を採用しており、共通伝送経路の採用、IP アドレスによるコンピュータの認識等の問題が出てきている。

現に銀行などの金融関係のネットワークはインターネットとの専用線を利用しこれらの問題に対応している。また、最近ではインターネットでのセキュアなオンライン決済を行うためのプロトコルである SET(Secure Payment Protocol)［5］が提唱されているが、決済時のみのデータを暗号化して通信するだけで、決済機関間の重要な顧客情報のやりとりは Gateway の裏側の専用線でやり取りされているのが現状のようである。

しかし、安価で便利なインターネットで重要なデータをやりとりしたいという要望は多い。秘匿にするまでもないデータであっても、ウィルスや改ざん防止の要望はある。特にデータベースで扱うデータは間違いがあってはならず、データベース間の複製（Replication）には注意を払うべきである。

Internet のようなオープンネットワークでセキュアに通信を行うには、どうしても“暗号”が必要となり、改ざんやなりすましを防ぐにはどうしても“認証”が必要となる。

現状のデータベースを見てみるとファイアウォールの中の安全なネットワーク上での使用を前提としているところが多々見受けられ、今後のデータベースの普及を考える上で不可避な問題を提唱し報告する。

今回複数の認証局間の形態をテーマに以下のような問題を検討する。

（1）データベースの形態

連携すべきデータベースが複数ある場合、これらデータベース間で検索・更

新を行う上で最適な配列の考察。

（2）データベース間のセキュリティの確保
オープンネットワークを介したデータベース更新時の盗聴・改ざん・なりすまし防　止の考察.

（3）データベースへのアクセスコントロール
データ更新サーバの認証方法の考察。

2.　基礎となる暗号技術

送信者・受信者以外の第三者への情報の漏洩を防止する守秘機能や、データの改ざん防止および電子的な認証を可能にする認証機能は、暗号技術やこれを利用した認証技術により可能となる。これらはネットワークのセキュリティを技術的に保証する有力な情報処理技術であり、情報通信基盤の基盤となりつつある基礎技術である。

この暗号・認証技術は、インターネットのようなオープンなネットワークの登場によって近年、最も注目を集めてきている技術であり、ネットワークに接続されているコンピュータ間の通信のセキュリティを確保するために不可欠な技術である。

2.1　暗号技術

データの暗号化には、送りたい情報（平文）を暗号化する鍵（暗号鍵）と、暗号化した情報を元の平文に戻す鍵（復号鍵）が必要となる。これらの2つの鍵を総称して暗号鍵と呼ぶ。現在、このデータの暗号化には、大別すると共通鍵暗号方式と公開鍵暗号方式の2つが存在する。

K1,K2：暗号鍵

平文	K1 →	暗号文	K2 →	平文
0		0		0
1		4		1
2		7		2
3		5		3
4		2		4
5		6		5
6		3		6
7		1		7

図1　暗号/復号の例

共通鍵暗号方式は、メッセージをあるブロック単位毎に高速にスクランブルする方

法である。暗号化鍵と復号鍵が同一の方式で暗復号を行うため、通信相手毎に異なる鍵を用いなければならず、n対n通信ではシステム全体でn（n-1)/2 （≒ 1/2 × n2)個の暗号化鍵が必要となる。

このような問題を解決するために公開鍵暗号方式は考案された。ある鍵とそれに対応する鍵がペアになっており、一方の鍵を使用して作られた暗号文は、ペアであるもう一方の鍵を使用しなければ復号化できない方式である。2種類の鍵のうち暗号化鍵を不特定の相手に公開するすることが可能であり、公開された暗号化鍵を「公開鍵」という。n対n通信全体では2n個の鍵の管理で済むメリットがある。

また、一般的に共通鍵暗号方式は公開鍵暗号より暗号化のスピードが速いので（約1000倍)、メッセージの暗号化には高速な共通鍵暗号方式、暗号鍵の鍵配送としてこれを公開鍵暗号方式で暗号化し、相手に送信する方法が通常採られる。

2．2　共通鍵暗号方式

共通鍵暗号方式は、暗号化鍵と復号鍵が同一の方式で暗復号を行うものである。データをあるブロック単位（8～128ビット）毎に、ビット毎のXOR（排他的論理輪)、ビット交換、シフトなどでスクランブル化する方法で、復号化するときはこれの逆を行うことで元のメッセージが得られる。

$$M \xrightarrow{K} C,\quad C \xrightarrow{K} M$$

M：平文、C：暗号文、K：暗号鍵

代表的な共通鍵暗号としてFIPS化されたDES（Data Encryption System）があり、これは1977年にIBMの考案により米政府の公式暗号方式に指定されており、現在まで20年近く事実上の国際暗号標準として知られている。最近では米RSA社のRC2、RC5、三菱電機のMISTY、スイスのAscom社のIDEA、RFC化されたCASTなどが出てきている。DESの暗号化鍵は64ビットであったが、最近出てきている方式は128ビットへと拡大している（ブロックサイズは64ビットのまま)。また、DESの処理を3回施すトリプルDES（TDESまたは3DES）は、DESをインプリメントしたシステムにほとんど変更を加えることなくビット数を増やせるため、金融界を始めとして広く利用されている。

NIST（米国技術標準局）からAES（Advanced Encryption Standard（[24]が公募されている。AESの仕様としてはブロックサイズが128 bitに拡張されており、今までの暗号と互換性を保てなくなる。

新生として最近カオス関数を取り入れたカオス暗号が登場した[26]。カオス関数とは例えば下記の式のように前回（t-1)、の値を決定していく方法で、2個の初期値（$\chi 0$と $\chi 1$）で無限のランダムな値をとる[29]。

$$\chi t+1 = f\ \lambda\ (\chi t) = \lambda\ \chi t\ (1-\chi t)$$

実装のアルゴリズムが非公開なままであり、強度評価が十分でない。カオス特有のアトラクタの考察など安全性に対する不明な部分が多く、またブロックサイズが無限であるが故の安全性の未確認などを理由に躊躇する傾向がある。

2.3 公開鍵暗号方式

公開鍵暗号方式は、上記共通鍵暗号方式で暗号化した鍵を配送するために、1976年に Diffie と Hellman によって考案されたのを祖とする。これが Diffie-Hellman 公開鍵方式（DH）と呼ばれるものである。その翌年に1977 年に Rivest、Shamir、Adelman によって RSA 公開鍵暗号方式が発表された。この方式は DH のアイデアをよりよく実現させた方法であり、電子署名が出来るメリットもあるので、RSA の方が業界標準化となった。

K1　　K2
M → C、C → M　　M：平文、C：暗号文、K1：暗号鍵, K2：復号鍵

DH や RSA では、割り算のあまり余剰定理

$$A = B \pmod{P}$$

を用いる。

DH や RSA に遅れること約十年、楕円曲線関数を利用した楕円曲線暗号が発表された。これは楕円曲線特有の交点の和を利用した方法で、国内外の会社から楕円曲線暗号[27]を開発している（NTT、松下電器、Certicom、Next、Siemens、Thompson等）。それぞれの会社がそれぞれのパラメータを使用しており、互換性はない。それは、一言に楕円曲線暗号といっても様々だからであり、楕円曲線での DSA や ElGamal 等他の署名アルゴリズムを利用するのが通常である。

楕円曲線上での任意の2つの点 A（xa, ya）と B（xb, yb）が求まれば、AとBを結ぶ直線と楕円曲線との３番目の交点 C（xc, yc）とすれば、C の x軸対象点 C'（-xc, yc）を

$$C' = A + B$$

というように定義する。楕円曲線では、
楕円曲線の接線を任意の点とすると

$$2A = A + A$$

とし、

3 A = A + 2 A
4 A = ...

というようにテーブルを作成し、これを暗号に使用する。これは、RSAでいう mod (商のあまり) と同様の作業に相当するが、鍵長が短く計算量が少ないので、特に計算スピードの遅い IC カードでは魅力である。

A (x1, y1)、B (x2, y2)、C (x3, y3) とすると、

$$x3 = \left(\frac{y2 - y1}{x2 - x1}\right)^2 - x1 - x2$$

$$y3 = \frac{(y2 - y1)(x1 - x3)}{x2 - x} - y1$$

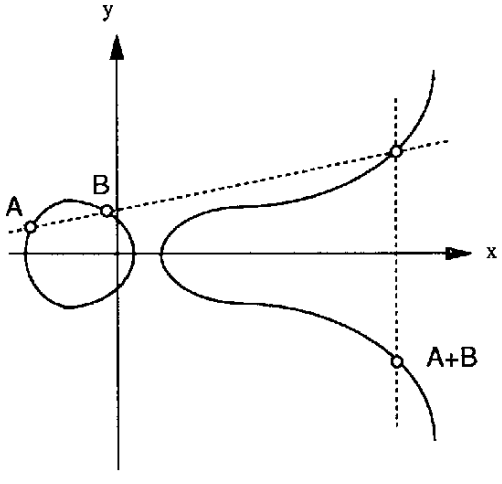


図 2　楕円曲線の加法

このように公開鍵暗号とは、一方の鍵を公開しても他方の鍵を他人に容易に推定されないようにすることであり、数学的難しさを強度の拠り所としている。これに伴い、暗号化・復号には複雑な計算が要求されるようになり、必然的に暗復号化速度が遅くなる宿命を抱えている。

現在では、DH、RSA 公開鍵暗号方式を代表として、実用に耐えうる暗号化技術が入手可能となっており、この技術を活用することによって高度なセキュリティの確保が可能になってきている。

2．4　鍵管理方式

暗号技術とは異なり、使用者固有の暗号鍵を配送する技術が鍵管理である。代表的な方法は、国産技術である KPS（Key Predistribution System）［28］であり、公開されている情報（利用者のID等）から通信の当事者が双方で同じ鍵を生成する方式である。KPS 方式では、マトリックスを用い、鍵を集中的に管理するセンターが、ユーザ固有の情報（公開鍵暗号方式でいう公開鍵に相当）から、それぞれユーザ固有の鍵（秘密鍵に相当）を生成する。n×n のマトリックスを使って、通信相手で異なる暗号鍵を作成することができる。基本的なアルゴリズムについて以下に説明する。

センターマトリックスを G （n×n行列）、A, B をそれぞれのID（nベクトル）、Xa, Xb をそれぞれの秘密鍵情報とすると、

$$Xa = G \cdot AT \qquad （注釈：\ AT\ は\ A\ の転置行列）$$
$$Xb = G \cdot BT$$

となる。G が対称行列であるとすると

$$Xa \cdot b = Xb \cdot a$$

となり、A、B間で鍵を共有することができる。

KPS 方式では、公開鍵暗号方式のように数学的な難しさより、情報量の多さを強度の拠り所にしているのが特徴的である。また全体の鍵の数は n 個で済む。また KPS の特許は日米欧でアドバンス社が保持している［28］。

アメリカで専ら利用されている鍵管理システムとして Kerbros がある。

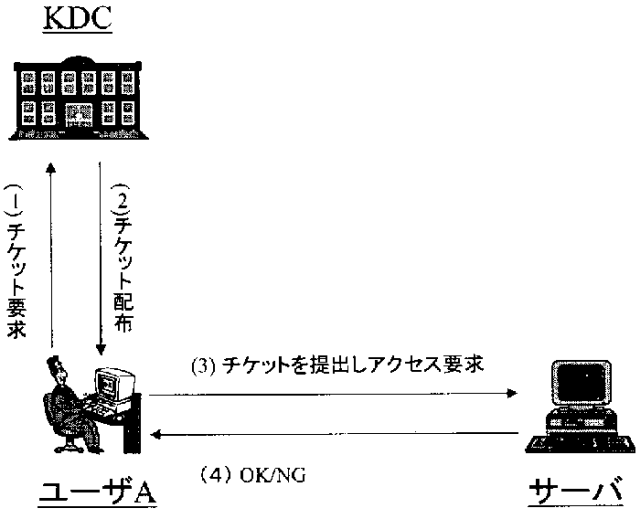


図 2　Kerberos の概略

Kerberos とは MIT と IBM で開発された鍵管理システムであり、共通鍵を基本とし

ている。Kerberous では鍵配布センター（KDC）に各ユーザの共通鍵を登録し、ログイン要求時に時間限定のチケットを KDC から受け取る。このチケットはアクセスのためのパスワードが暗号化されており、要求者以外は読めないようになっている。

Kerberos は RSA による公開鍵方式が確立する前に発表されたものであるが、MIT をはじめ現在でも多くのところで使用されている。また、FreeBSD などの OS にもフリーで実装されている。

3．　公開鍵証明書と認証局

現在、インターネットでの暗号・署名通信では、SSL や S/MIME などのセキュアなプロトコルが提唱・標準化されつつあり、これらのプロトコルは ITU-T X.509 を利用した公開鍵証明書を利用している。X.509 は公開鍵暗号を利用し、“認証局”と呼ばれる通信者以外の第三者へお互いの公開鍵を登録し、通信者 はX.509 のフォーマットに則り認証局の秘密鍵で署名された公開鍵証明書を暗号、または署名検証のための手がかりとする。通信同士でお互いに認証局を信用することを拠り所とし、通信両者が認証局を信用するとの共通認識の基で、認証局が発行する公開鍵証明書は正しいと判断される。

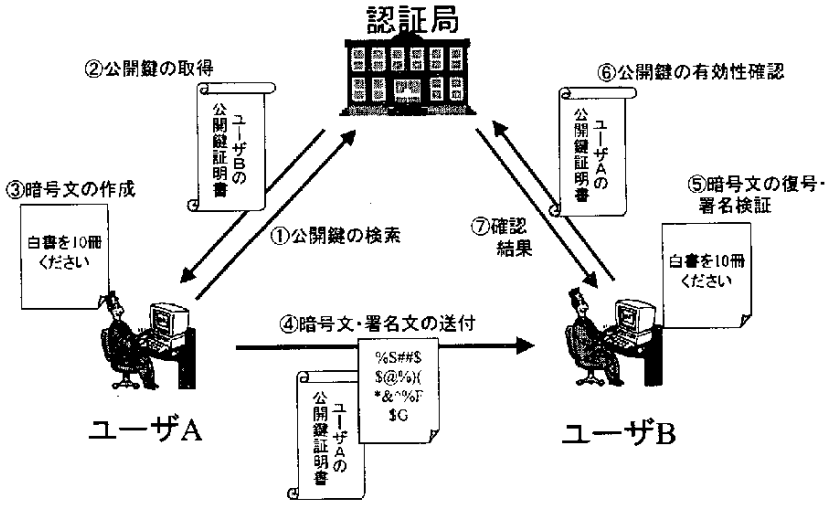


図 3　認証局

X.509 は図 4 のように ASN.1（Abstruct Syntax Notation One）のBER（Basic Encoding Rule – CCITT X.209）で符号化というフォーマットで符号化されている。これは MIME（Multipurpose Internet Mail Exchange encoding）と似たようなものと考えてよく、Internet で公開鍵証明書を交換するためのフォーマットとしてデファクトスタンダードとなっている。

```
Certificate ::= SEQUENCE {
    tbsCertificate  TBSCertificate,          #署名対象データ
    signatureAlgorithm  AlgorithmIdentifier,  #署名アルゴリズム
    signature BIT STRING                      #署名データ
  }
```

図4　ITU-T X.509 フォーマット

```
MIICVTCCAf+gAwIBAgICBz0wDQYJKoZIhvcNAQEEBQAwQDELMAkGA1UEBhMCSlAx
ETAPBgNVBAoTCFBJTE9ULUNBMR4wHAYDVQQLExVDZXJ0aWZpY2F0ZSBBdXRob3Jp
dHkwHhcNOTgxMDAxMDMxMjA1WhcNOTkwMzMwMDMxMjA1WjCBgTELMAkGA1UEBhMC
SlAxGTAXBgNVBAoTEElDQVQgU2VjcmV0YXJpYXQxGTAXBgNVBAsTEElDQVQgU2Vj
cmV0YXJpYXQxGDAWBgNVBAMTD0tJVEFOTyBIaXJveXVraTEiMCAGCSqGSIb3DQEJ
ARYTa2l0YW5vQGppcGRlYy5vci5qcDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkC
gYEAvBdPKTe2EYfewMqj/Xc9CyEqJQdCZbZZnnv67v6xD2A0pcj8ra6YKRAJDL4d
//epl5uagazx9sxW1qtARax8cnYwAU5CWxrUF7ywN+cSgTn5lPf5+GzWzD5yzHSd
Se68HJaEqVgy9gQHeUWcQW0UgduGnuGBHQboOyfxfiO8pPECAwEAAaNdMFswDwYD
VR0TAQEABAUwAwEBADAcBgsqgwiBnF8LAwEdAgEBAAQKMAigAoYAoQKGADAUBgNV
HR8BAQAECjAIMAagBKAChgAwFAYJYIZIAYb4QgEBAQEABAQDAgCgMA0GCSqGSIb3
DQEBBAUAA0EAJuhOv5evH5SLb4ym5l5DeF0JOWHj4UiA0THkW80aAxHpcdaiL2yW
lUj7XfoB4rJIdNQeba/wqe+5Dx5hQSNVpF==
```

図5　X.509 データ(サンプル)

図5をdecode すると図6のような情報が取り出せる。図4に記載されているもの以外に

- ●署名対象者の公開鍵情報
- ●これら全部のハッシュ値（認証局の署名情報）

が盛り込まれている。

公開鍵証明書にはそれぞれ発行者（issuer)、署名対象者（subject）のC（国名)、O（所属)、OU（更に細かい所属)、DN（本名）が明記されており、証明書検証しやすいような手がかりを記載している。

Version No = 2
Serial No = 73D
Validity from 981001031205Z
to 990330031205Z
issuer:
C=JP
O=PILOT-CA
OU=Certificate Authority
subject:
C=JP
O=ICAT Secretariat
OU=ICAT Secretariat
CN=KITANO Hiroyuki
emailAddress=kitano@jipdec.or.jp
signature:
md5WithRSAEncryption
publickey:
alg = rsaEncryption
basicConstraints:
not critical
not CA
authorityInfoAccess:
not critical
authorityInfo:

certStatus:

cRLDistributionPoints:
not critical
DistributionPointName:
fullName:

netscape-cert-type:
not critical
Type: SSLclient S/MIME

拡張フィールド

図 6 X.509v3証明書情報（サンプル）

```
30 (597) {
  30 (511) {
    A0 (3) 020102
    02 (2) 073d
    30 (13) {
      06 (9) 2a864886f70d010104
      05 (0) [ ]
    }
    30 (64) {
      31 (11) {
        30 (9) {
          06 (3) 550406
          13 (2) [JP]
        }
      }
      31 (17) {
        30 (15) {
          06 (3) 55040a
          13 (8) [PILOT-CA]
        }
      }
      31 (30) {
        30 (28) {
          06 (3) 55040b
          13 (21) [Certificate Authority]
        }
      }
    }
    30 (30) {
      17 (13) [981001031205Z]
      17 (13) [990330031205Z]
    }
    30 (129) {
      31 (11) {
        30 (9) {
          06 (3) 550406
          13 (2) [JP]
        }
      }
      31 (25) {
        30 (23) {
          06 (3) 55040a
          13 (16) [ICAT Secretariat]
        }
      }
```

図7 ASN.1形式（その1）

```
  31 (25) {
    30 (23) {
      06 (3) 55040b
      13 (16) [ICAT Secretariat]
    }
   }
   31 (24) {
    30 (22) {
      06 (3) 550403
      13 (15) [KITANO Hiroyuki]
    }
   }
   31 (34) {
    30 (32) {
      06 (9) 2a864886f70d010901
      16 (19) [kitano@jipdec.or.jp]
    }
   }
  }
  30 (159) {
   30 (13) {
    06 (9) 2a864886f70d010101
    05 (0) [ ]
   }
   03 (141) 0030818902818100bc174f2937b61187dec0caa3fd773d0b212a25074265b6
599e7bfaeefeb10f6034a5c8fcadae982910090cbe1dfff7a9979b9a81acf1f6cc56d6ab4045ac7c
727630014e425b1ad417bcb037e7128139f994f7f9f86cd6cc3e72cc749d49eebc1c9684a95832f6
040779459c416d1481db869ee1811d06e83b27f17e23bca4f10203010001
  }
  A3 (93) 305b300f0603551d13010100040530030101003 01c060b2a8308819c5f0b03011d
02010100040a3008a0028600a102860030140603551d1f010100040a30083006a004a00286003014
06096086480186f842010101010004040302 00a0
 }
 30 (13) {
  06 (9) 2a864886f70d010104
  05 (0) [ ]
 }
03 (65)
0026e84ebf97af1f948b6f8ca6e65e43785d093961e3e14880d131e45bcd1a0311e97
1d6a22f6c969548fb5dfa01e2b24874d41e6daff0a9efb90f1e61412355a4
```

図 8　ASN.1 形式(その 2)

図6の情報が公開情報となり、認証局が認証した情報となる。

具体的な信用の構造としては、通信両者が認証局の公開鍵証明書を信用し、認証局自身の公開鍵で証明検証できるユーザの公開鍵証明書は正しいと判断され、その公開鍵証明書に書かれてある項目は正しいと判断される。つまり、認証局がセキュアであることが第1に重要となる。しかし、認証局を単体で行う上では、他の認証局とデータのやり取りを行う必要がないので、

・認証局サーバを単に不正アクセスから耐えられるものとする

・認証局をオフラインで運営する

などの処置を行えばよい。

RFC1422 [3] では IPRA (Internet PCA Registration Authority) [5] をインターネット唯一の Root CA(トップの認証局)と規程され、IPRA の下に PCA(Policy Registration Authority)、PCA の下に CA(Certificate Authority -- 認証局)が置かれることとなっている。認証局(CA) から更に sub-CA として認証局を作ることも出来る。

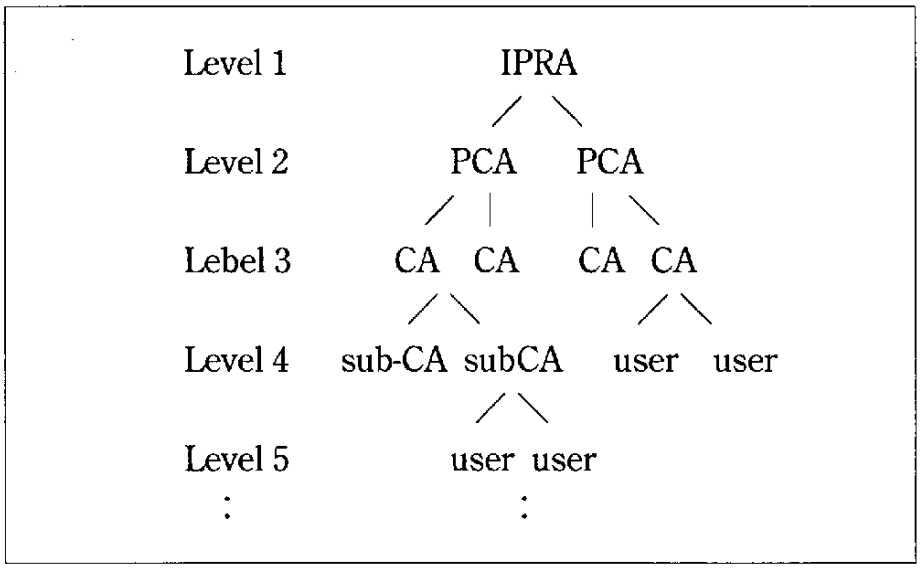


図9　認証局の階層構造

X.509 は ITU-T X.500 シリーズであり、directory 構造を取れるように公開鍵証明書の中に発行局名が入っていおり、メカニズム上は階層構造を辿れることができることになっている。

RFC1422 では、これら認証局間の信用のヒエラルキーは存在するものの、階層的な公開鍵証明書のやりとりやオンラインでの認証は規定されておらず、運用的には、それぞれオフラインで上位認証局の証明書を取得してローカルに保存し、必要に応じて使用することになる。フリーの SSL サーバである SSL-Apache[6] では公開鍵証明書を用いたアクセスコントロールが可能であるが、同様の処置を行い、ローカルで証明書の検証を行っている。

また、 Netscape や Internet Exploror に代表される SSL を採用したWeb Browser で

は事実上 RFC1422 を無視する形で乱立している。 level 3 の CA の証明書を全部ブラウザ内に保存しており、必要に応じ保存された認証局の証明書を使って WWW サーバの認証を行っている。

更に、1999年1月10日に IPRA の自己署名による公開鍵は有効期限が切れたが、ISOC（Internet Society）[7] では IPRA のサポートを今後行わないとしている [8]。

このように IPRA を頂点とした認証の階層構造は崩壊しつつあるが、今後、企業・団体でイントラネット/エクストラネットとして認証局を立ち上げるのが増えてくると思われ、地域的・業務的な理由で認証局の分散化が図られてくるであろう。

有効性確認においては ITU-T X.509v2 CRL（Certificate Revocation List）を使用する。

```
MIHiMIGBMA0GCSqGSIb3DQEBAgUAMDwxCzAJBgNVBAYTAkpQMQ0wCwYDVQQGEwRJ
Q0FUMR4wHAYDVQQLExVDZXJ0aWZpY2F0ZSBBdXRob3JpdHkXDTk4MDkzMDA2MDQw
OFoXDTk4MTAzMDA2MDQwOFowFDASAgEaFw05ODAxMjgwMzM0MDBaMA0GCSqGSIb3
DQEBAgUAA00AFpB2DE0m7gP/nLAv9DfY6zthM+QEFd4Gle0gat1YtAa26NXZMBeL
lam7ISA1yL066njD5KDxtdbEgiqHk5C3AQvd5mQQ0WB5rvX5Fd==
```

図10　CRL の例

CRLとは破棄リストであり、いわゆるブラックリストである。無効になった証明書のシリアル番号と破棄された時刻を記載し、認証局の秘密鍵で署名したものである。CRLも証明書と同様にASN.1のBERエンコードされている。

```
signature:
        md2WithRSAEncryption
  issuer:
        C=JP
        C=ICAT
        OU=Certificate Authority
  thisUpdate:    980930060408Z
  nextUpdate:    981030060408Z
  revokedCertificates:
        serial  revokedtime
          1A  980128033400Z
```

図11　CRL情報

通常の証明書の有効性確認では、この CRL をそれぞれ検証したい人が検証したい証明書からシリアル番号を取りだし、CRL にそのシリアル番号が入ってないかを確認することが必要である。もし CRL にリストされているようであれば、その証明書は何らかの理由で破棄されており、その証明書は使えなくなる。CRLv2 からは破棄理由を個別に明記できるようになっている。

CRL はそれぞれの認証局で個別に発行するものであり、検証を行うにはそれぞれの CRL が必要となる。また、CRL 自体も有効期限があり、例えば一ヶ月に 1 回しか発行されないとなると、最新一ヶ月の破棄情報は入手不可能となる。

4. 認証局間データの連携

X.509 は ITU-T の X.500 シリーズのひとつであり、元々ディレクトリを構成する上での公開鍵のデータ交換を行うために設けられたフォーマットである。暗号メールの標準であった PEM（Privacy Enhanced Mail）[3] では、X.509 v1 [4] を用い、認証局を用いることを定めている。

ただし、前章でも述べたように IPRA を頂点とした認証局機構の規程はあるものの認証局間のデータのやりとりなどのプロトコルは存在してない。唯一 ICAP（ICAT CA Package）[23] が提案しているのみであるが、まだ標準化まで行ってない。

IETF（Internet Engineering Task Force）[9] では RFC1421シリーズに代わる X.509 を利用した公開鍵証明書機構を検討する WG である PKIX（Public-key Infrastructure (X.509)）[10] では、SSL や S/MIME 等のセキュアプロトコルのインフラとして ITU-T X.509 v3 を用いた新しいプロトコルを検討している。しかし、ここではまだ認証局間の連携については検討されてない。

複数の認証局がある環境で、各認証局が発行した証明書の検索・有効性確認を行うための認証局連携を考えてみる。

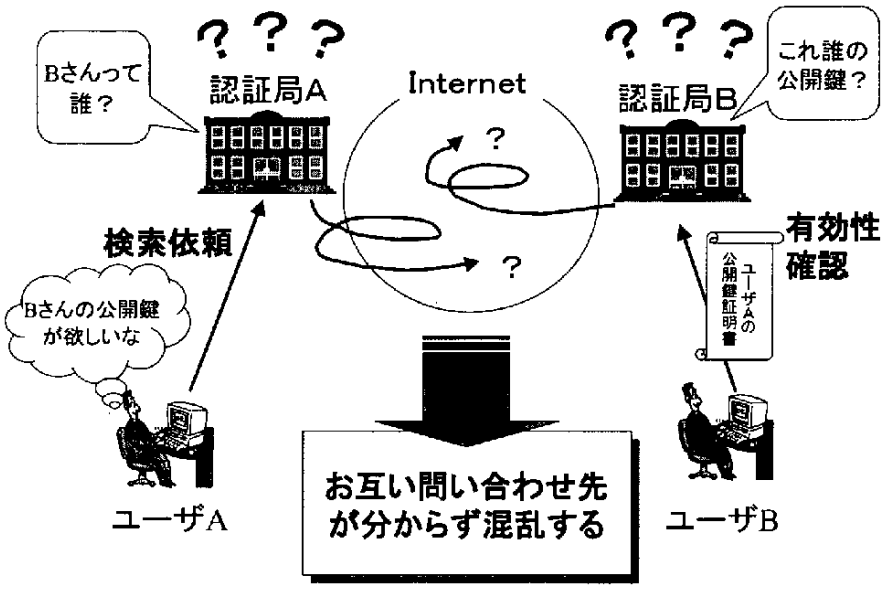


図12　複数認証局での問題点

（1）各リンク型
各認証局がリクエスト毎に他の認証局へ問い合わせを行う。

（2）バケツリレー型
ネットワークトポロジーなど、認証機構とは別に配送経路を決め、更新と同時に他の認証局へ配送する。

（3）階層化型
ひとつの root CA を設定し、きっちりとした階層構造を取り、きっちりとレベル毎に共通の意味を持った認証局を設定し、必要に応じ通信する。

（4）集中管理型
認証局とは別に収集・検証・配布を行う存在するすべての認証局から逐次データをもらい、キャッシングする。

（1）の状態は現状の VeriSign や CyberTrust など商用の認証局が乱立している状態に相当する。Netscape や Internet Explorer（IE） などのアプリケーションでは、初期状態でこれら認証局の公開鍵証明書が50以上入っている（図6）。ただし、認証局の公開鍵証明書の新規・更新を行うサービスは行われておらず、アプリケーションのバージョンアップと同時に行われているようである。

証明書検証時においてはアプリケーションに梱包されている証明書を使って検証を行う。すなわち、それぞれのユーザのそれぞれのアプリケーションにローカルに保存されている認証局の公開鍵証明書を使って、ユーザ、またはサーバの公開鍵証明書の有効性確認を行っている。

証明書検索時においては、他の LDAP（Lightweight Directory Access Protocol）サーバへ別途登録されたものを対象として検索を行うことができる。しかし、この LDAP サーバも乱立している状態であり、ユーザが一意的に検索を行うのはできない状況である。

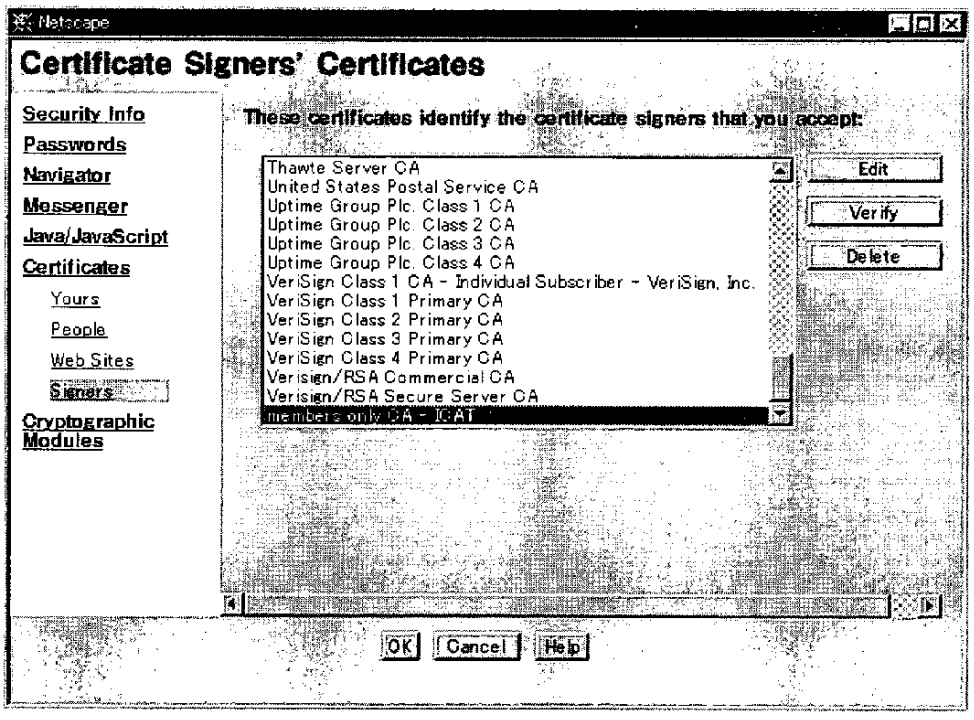


図13　Netscape にバンドルされている認証局証明書

そこで、認証局機構の連携が必要とされる下記２つの項目について、上記４つのスについてそれぞれ検討を行う。

・.証明書検索（取得）
・有効性確認（検証）

個別ケースについて検討する前に、共通のこととして以下のことを想定する。

１）証明書検索（取得）
　証明書の検索時には

・本名（DN）がわかっている
・ メールアドレスがわかっている
・所属とメールアドレスがわかっている

場合について検討を行う。

また、プライバシー保護や SPAM 防止のために“*.jp” や“kit*” などのバルク的な検索を排除するかどうかはここでは検討しないが、検索結果表示の上限を設けるなどの手段が考えられるであろう。

２）有効性確認（検証）
　証明書の有効性確認時には、検証を行いやすいように公開鍵証明書に

・証明書自体を検証するURL
・認証局の公開鍵証明書を取得するURL

をX.509 v3の拡張フィールドに書きこむことを考える。証明書の有効性確認には手元に検証したい公開鍵証明書があることが前提となるので、証明書自体の検証に必要な情報をあらかじめX.509の拡張フィールドに記載しておくことは有効である。

現実世界に即して言えば、運転免許証は氏名・住所などが記載されてあるが、それらはX.509でのsubjectに相当し、東京都公安委員会の判子がissuerに相当することになる。免許証に公安局の問い合わせ先が書いてあれば、その免許証が本当にその公安局で発行されたものであるかどうか検証が行える。

実際の世界ではあまりないことであるかもしれないが、都道府県レベルの公安局が本当に国レベルで認定、または承認されているものであるか検証するには、さらにその上の国際間では、などといった認証の階層構造に対応できるものとなる。

4.1 各リンク型

下図のように無秩序に乱立した認証局を想定する。証明書検索時においては、各認証局個別の設定になるであろうが、真っ先に

（1）自分の所で発行したものか探す。
（2）キャッシュの中を探す。
（3）検索順番を任意に決め、検索をしていく。
（4）検索結果をキャッシングする。

の順番で実行することを思いつく。（1）と（2）はそのままでいいと思われるが、（3）では

・検索順序
・新規認証局情報の取得

の問題がある。

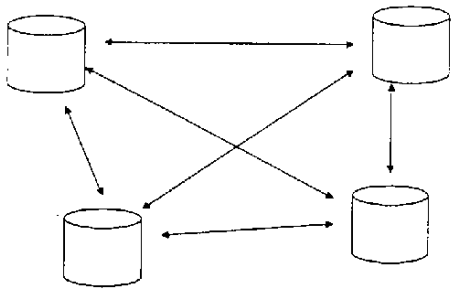


図14　各リンク型

（3）の検索時においては、リストアップされた認証局へ次々と通信して、マッチするものを探し出さなければならない。

検索条件にすべてマッチするものを取り出すのであれば、最初に問い合わせのあった認証局から他のすべての認証局へマルチタスク的に通信した方がいいし、また、バッチ的にひとつづつ検索し、ひとつだけを探し当てるのであれば、今後の効率を考えて、

（5）ヒット率を考慮に入れ検索順序を入れ替える。

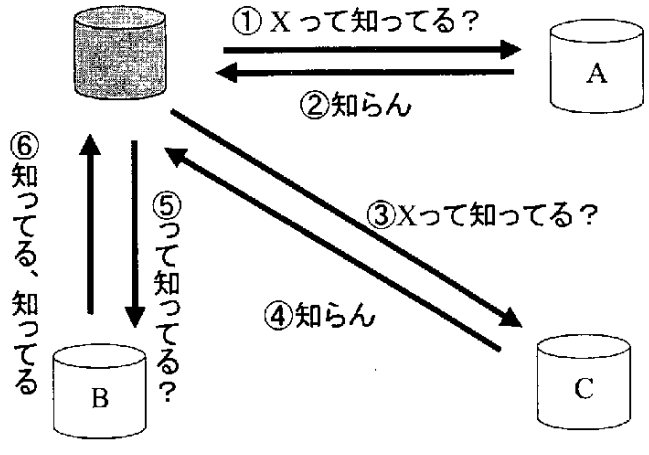


図15　検索のスキーム

を入れたほうがいいだろう。ただ同じメールアドレスで、利用目的などの違いにより複数の認証局へ登録されている場合も考えられるので、検索されたものをすべて表示するのが無難である。

署名検証時には、問い合わせのあった公開鍵証明書から

・証明書自体を検証するURL
・認証局の公開鍵証明書を取得するURL

を取りだし、それぞれの認証局へ問い合わせ、または認証局の公開鍵証明書の取得を行う。
ここで問題となるのが、

・すべての認証局情報をどうやって取得するのか？

ということである。すなわち、各自勝手に認証局を設立しているわけであるから、お互いの情報を相互にどうやって交換するか？という問題に直面する。
ひとつの方法は今までのNetscapeやIEのように

・アプリケーションで管理する

ということがある。しかし、これからWebBrowserや暗号化メール以外にも様々なアプリケーションでこのX.509をベースとした公開鍵証明書を利用することが容易に予想されるので、この方法は今後賢い方法とは思えない。

4.2 バケツリレー型

4.1の

・すべての認証局情報をどうやって取得するのか？

という解決法として、新規に認証局を立ち上げる場合、証明書の更新・検索を行えるように既存の認証局へ接続することが考えられる。

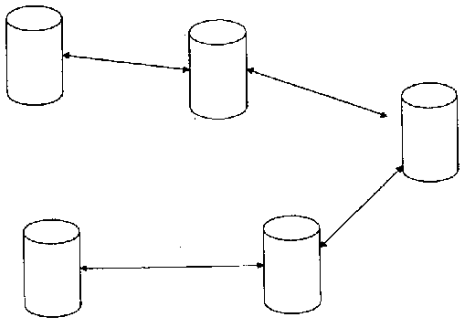


図16　バケツリレー型

昔のメール配送システムであった、UUCP（Unix to Unix CoPy）に倣ったものであり、PGP（Prity Good Privacy）の鍵サーバが行っている方法である。
この方法だと、接続経路が確立していれば

・すべての認証局情報をどうやって取得するのか？

という問題は解決する。

しかし、一次元的な配列では末端同士の情報のやり取りには時間がかかり、更新がすみやかに行われない可能性がある。どこかにノードを設け、２次元的経路を考慮すれば、ある程度緩和されるであろう。２次元的配列を考えるには、

・ネットワークトポロジー
・ネットワーク回線の太さ

などを考えなければならない。

署名検証時には、問い合わせのあった公開鍵証明書から

・証明書自体を検証するURL
・認証局の公開鍵証明書を取得するURL

を取りだし、それぞれの認証局へ問い合わせ、または認証局の公開鍵証明書の取得を行えばよいだろう。

4.3　階層化型

下図のような階層化構造を取らせ、きっちりとレベル毎に共通の意味を持った認証局を設定する。

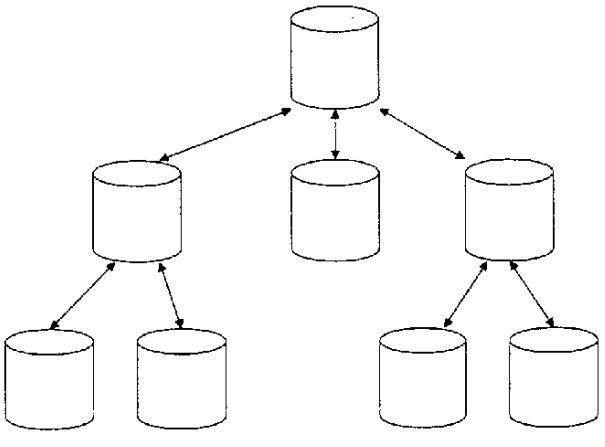


図17　階層化

階層レベルとして以下の２通りを想定する。

	例１）	例２）
Level 1：	・全社	・全世界
Level 2：	・事業部	・アジア
Level 3：	・部	・日本
Level 4：	・課	・東京
Level 5：	：	：

両方ともよくあるタイプの階層化構造である。一見スマートに見える方法であるが、よく考えてみると重要な問題がある。検索したい通信相手の所属がわかっていればいいのであるが、例えば

・本名しかわからない場合

では最初に問い合わせた認証局が、最初にどこをどう探していいかわからない。また、

・階層の構造を把握する手段

が別にないと難しいだろう。

ひとつの方法として、

・認証局の階層化構造とメールアドレスのマッチングを行う

という方法が考えられる［12］。すなわち、

kit@domain.company.co.jp（or kit@country.area.organic.org）

のメールアドレスであれば、必ず

ca.domain.company.co.jp（or ca.country.area.organic.org）

の認証局に証明書が登録されている

という一意的に認証局ホストのマシン名を決めてしまう手段も考えられる。これはかなり有効的な方法であるが、メールアドレスのドメイン毎に必ず認証局サーバを立ち上げなければならない、組織構造が変わればそれに合わせて認証局構造も変えなければならないというマイナスの面もある。

署名検証時には、問い合わせのあった公開鍵証明書から

・証明書自体を検証する URL
・認証局の公開鍵証明書を取得する URL

を取りだし、それぞれの認証局へ問い合わせ、または認証局の公開鍵証明書の取得を行う。取り出した認証局証明書が自己署名でなかったら、更に同様に親の公開鍵証明書を取りだし、自己署名に行きつくまで繰り返して結果を返す。

ここでひとつだけ注意しなければならないことは、認証局が階層構造を取る場合には、更に上の認証局の公開鍵証明書を取得し、自己署名に至るまで証明書の検証まで行わなければならない。何故なら、直接発行された認証局が有効でも、階層構造の上の方で、その認証局の証明書が破棄されている可能性もあるからである。

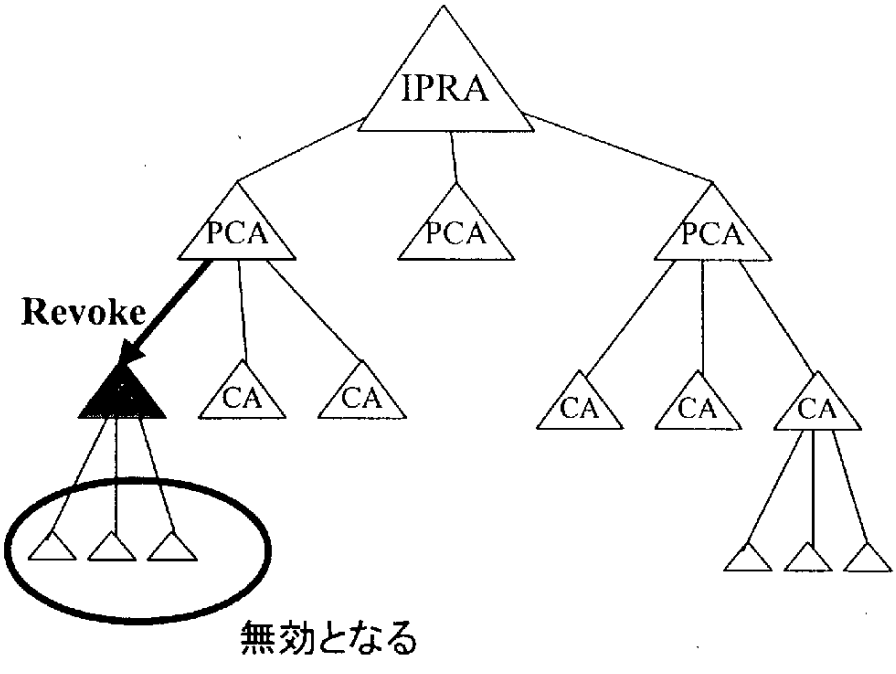


図18　階層構造での証明書無効

4.4　集中管理型

主に検索時の負荷を下げるために、すべての証明書情報を予め一箇所に集めておく方法である。

これは検索時にネットワークを介し、あらゆる方向へ検索要求を出すのをやめて、検索ロボットのように一定期間毎に公開鍵情報を収集するものである。検索と証明書の有効性確認を代理的に行い、検索・検証時にはその代理サーバに保存されたデータを元に処理を行う。

検索時にはその代理サーバへ問い合わせ、そのサーバに保存されているデータを検索し出力してやれば良い。1サーバで行うため、データベースを効率的に処理することができる。

署名検証時には、問い合わせのあった公開鍵証明書から

・証明書自体を検証するURL
・認証局の公開鍵証明書を取得するURL

を取りだし、それぞれの認証局へ問い合わせ、または認証局の公開鍵証明書の取得を行う。

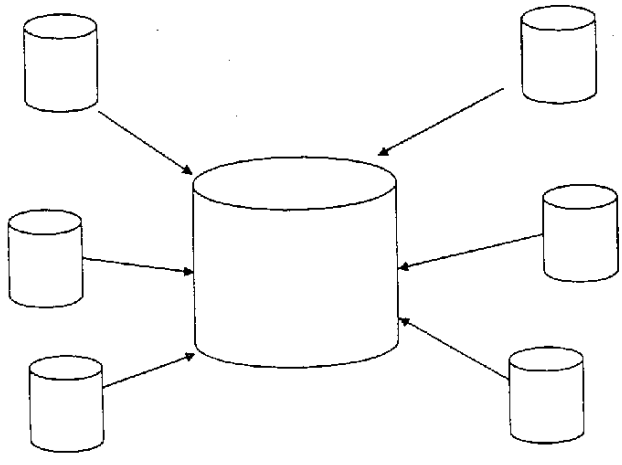


図19　集中管理型

4.5　最適な組み合わせ

どれも長所・短所を持ち合わせ、有効的な方法はない。ここでそれぞれの長所を生かしつつ、コンビネーションでうまく行ける方法を検討する。

検索を考えると 2.4 の集中管理型が一番いいように思える。しかし、有効性確認を考えるとリアルタイムで検証できる方がいいであろう。

	(1)	(2)	(3)	(4)
検　索	×	○	△	○
検　証	○	×	○	△

表1　認証局構造のまとめ

組み合わせで考えると（3）と（4）の組み合わせがよさそうである。

しかし、あるひとつのルートCA を頂点とした階層的認証局構造を作るのは難しい。例えば、2つの会社がそれぞれ階層構造を持っているとして、統合するためにさらに上にルートCA をつくることを考えてみると非常に大変である。階層構造は、ある限られた空間（会社、組織）内で有効であり、統合的な階層的認証局構造を取るには非常に大変な労力が必要とされるので、階層構造だけを選ぶのは得策ではない。

また、会社内や組織内のある閉じた世界での階層構造も、その作り方が様々になると思われるので、一意的に検索ができるとは限らないだろう。例えば、会員組織などではメールアドレスと発行局サーバのアドレスが必ずしも一致するとは限らない。

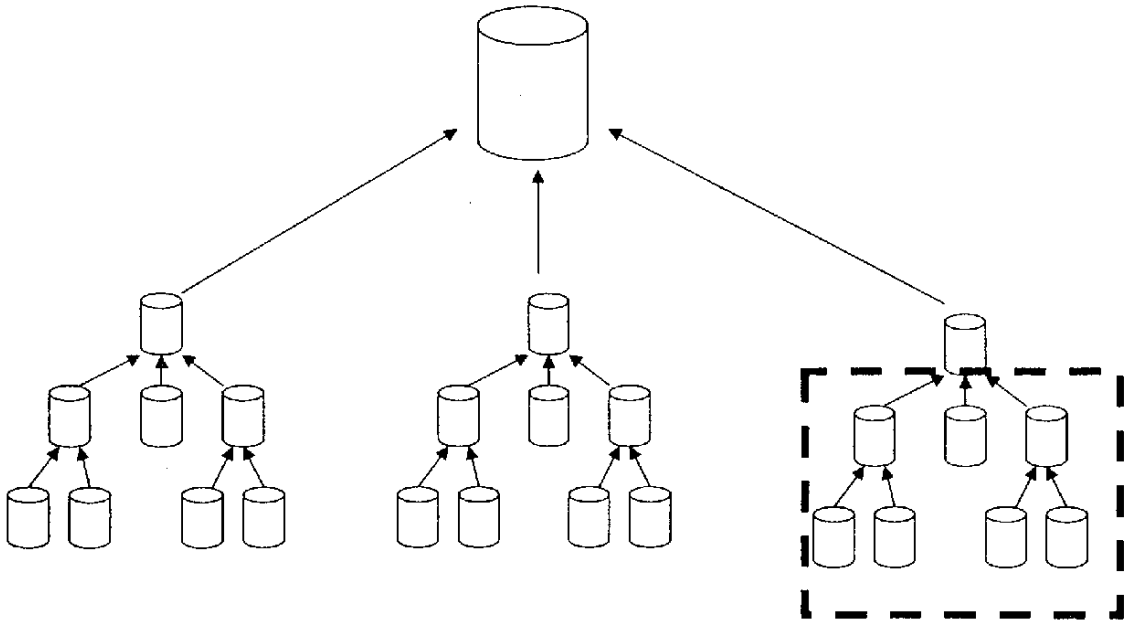


図20　（階層構造＋バケツリレー）＋集中管理

結論として、

配送：（階層構造＋バケツリレー）＋集中管理
検証：各リンク型

がよさそうであるということになる。

このモデルでは、例えば会社にファイアウォールがあり、企業内のトップCAしかInternet側から見えない場合でも、ファイアウォールに隠された認証局から発行された証明書でも検索は可能となる。

5．　データベース間のセキュア通信

インターネット、というより会社・部署内の LAN 化に伴い、イントラネットの普及が目覚しくなってきている。Oracle や Lotus Notes を始め、業務に適したデータベースの製品が多々出ている。メールソフトでもデータベースを実装したものまで販売されている。情報が氾濫している現在、データベースは個人にも必要なものとなりつつある。

現在データベースが使用されている環境は、ファイアーウォールの中であり、安全な領域である。よってデータベースでは、個人認証をメインとしたセキュリティに重きを置いているようである。すなわち、個々の個人を個別認証し、アクセスコントロールを行っており、誰々が管理できるなどの処理である。ほとんどがパスワードによる認証であると思われるが、Lotus Notes では X.509 のユーザ証明書を用いたアクセス制限が可能なようである。

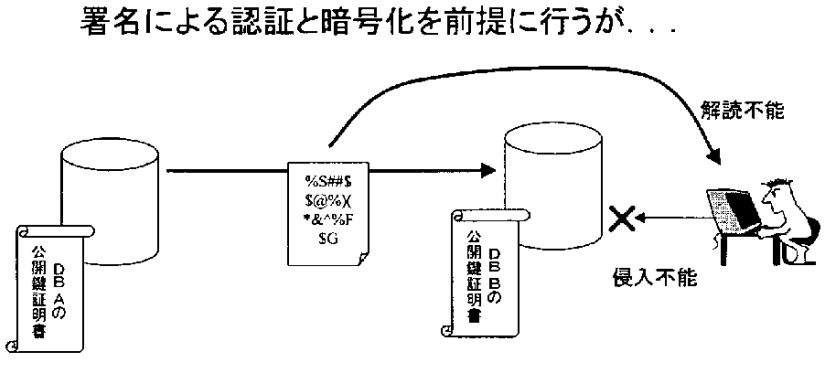


図21　認証局間のデータ交換の問題点

一時期言葉としてもてはやされたエクストラネットがあるが、これはお互いの認証局が相互認証を行い、ユーザ認証の下、アクセスコントロールを行っているものであり、メカニズム的にはファイアウォール内でのユーザ認証とほぼ同じである。

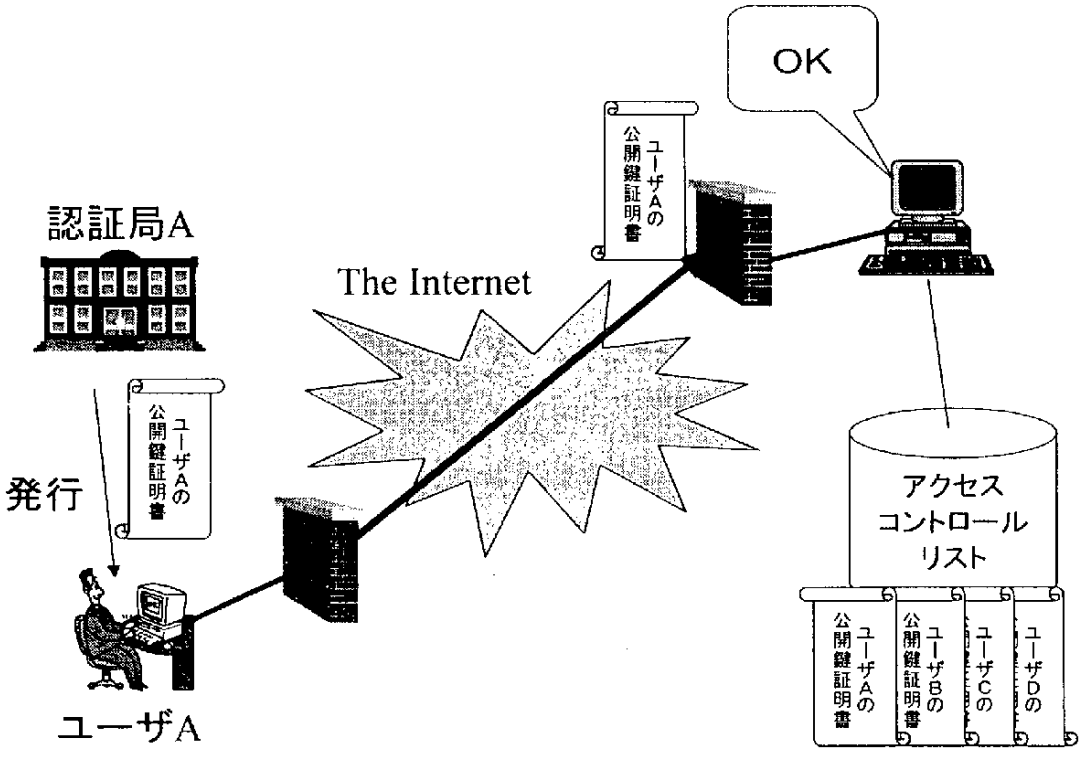


図22　エクストラネット

ただし、エクストラネットでは、ある組織（会社）の者が別の組織（会社）のサーバにNetscapeなどのWeb Browserを用いSSLプロトコルでアクセスし、その時にユーザの公開鍵証明書でのアクセスコントロールをしているものである。

ここで、エクストラネットを参考にし、インターネットを介したサーバ - サーバ間でのセキュア通信データベース間のセキュア通信について考えてみる。インターネットを介しているので

・盗聴
・改ざん
・3なりすまし

について検証を行わねばならない。しかし、公開鍵証明書の性格上、公開が目的の物であるので、暗号化は必要ないように思われるが、ここでは汎用性を考えて暗号するものとする。

IETF（Internet Engineering Task Force） ではインターネットに関わるプロトコルを検討し、標準化している団体である。IETF では TCP/IP に関するものを始め、アプリケーションからセキュリティに関するものまでディスカッションを行っている。既にあるセキュリティプロトコルを調査し、検討を行う。

以下、IETF の Security Area［13］ で検討されている代表的な WG 名である。

- PKIX（Public-Key Infrastructure （X.509））
- TLS（Transport Layer Security）
- S／MIME（S／MIME Mail Security）
- IPSec（IP Security Protocol）
- CAT（Common Authentication Technology）

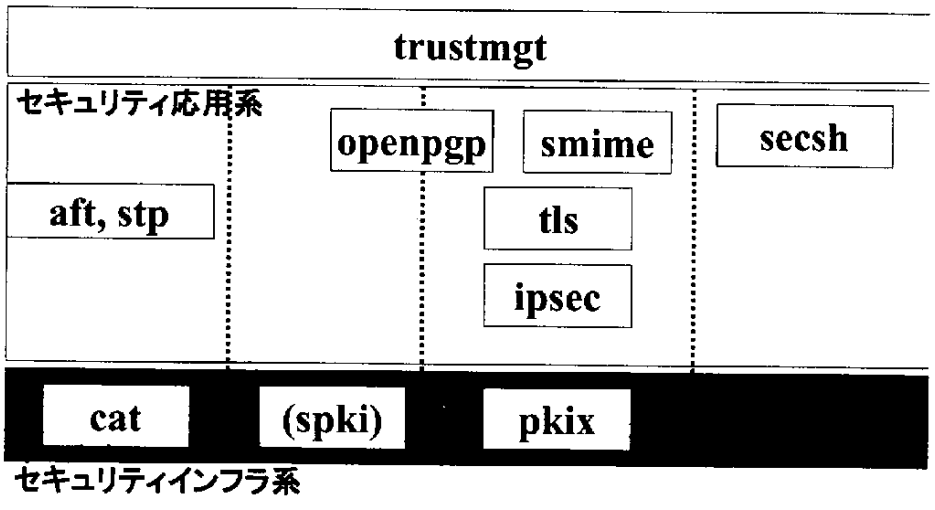


図23 IETF セキュリティエリア

セキュリティ関係ではないが、有用なプロトコルとして

- LDAP（Lightweight Directory Access Protocol）
- NFSV4（Network File System Version 4）

がある。

LDAP はベースが X.509 と同じく X.500 の directory であり、 DN, CN などのネーミングは同じである。また、LDAP には SSL（Secure Socket Layer）上で使える LDAPS がある。

NFSV4［16］では CAT［20］で議論されている Kerberous や GSS-API の認証を使ったファイル共有システムのディスカッションを行っている。

5.1 PKIX

PKIX［15］は X.509 をベースとし、認証認証機構の基礎となるプロトコルを作成している。いわゆる認証局自体の通信プロトコルなどを標準化している WG で、4 年に渡るディスカッションの結果、ようやく一連の I-D（Internet-draft）が RFC（Request for Comments）になるようである。RFC になれば、インターネットでの標準プロトコルになるので、皆これをサポートしなければならない。しかし、現時点では認証局間の連携についてはディスカッションされておらず、フェーズ 2 からの課題であると思われる。

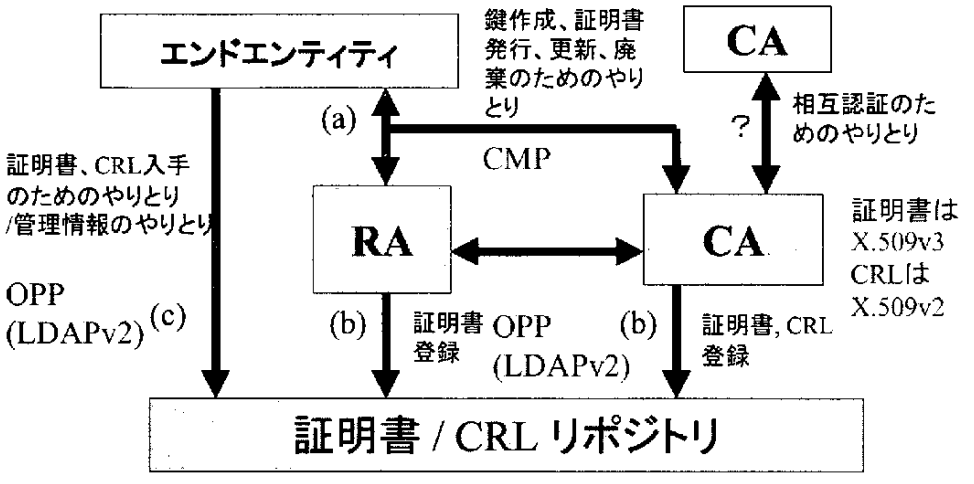


図24 PKIX

5.2 TLS

SSL（Secure Socket Layer）で親しまれてきたプロトコルであるが、HTMLの標準化団体であるW3C（World-wide web Consortium）から標準化作業を受け継いだ時に、Microsoftが提唱してきたPCT（Private Communications Technology）をも統合し、Web以外でも使えるよう名称を変更した。現在プロトコルSSL3.0がプロトコルTLS1.0として審議中である。しかし、一般にはいまだにSSLの名称で呼ばれている。

TLSではトランスポート層での暗号/認証が可能であるので、上位アプリケーション層では特に暗号/認証を気にすることはなく、プログラムの改造は必要としない。つまりTLSでのコネクションの上で

・SQL
・telnet

など既存のアプリケーションが動くことになるが、現状のアプリケーションを見るとNetscapeやIEのようにTLS層を含んだアプリケーションを作らざるを得ない。今後TLSが浸透していくと、個々のアプリケーションとは独立に、例えばMicrosoft Windowsでいえばwinsock.dllなどのTCP/IP関連ライブラリでサポートされるようになるだろう。事実Microsoftではwinsock2.dllの開発・一部実装を行っているようである。今後このような実装が増えてくれば、今までのTCP/IP層のように透過的に、また総合的にTLS機能を使うことができるだろう。

現時点では、Eric A. Youngらが作成したSSLeay［17］がTLS（SSL）のライブラリを提供しており、対応プラットフォームはUNIXとWindowsである。このライブラリを使ったアプリケーションに代表されるのが、フリーのhttpdソフトであるApache[18]をSSL対応にしたApache-SSL［19］である。

SSLでのhttpであるhttpsでは、サーバ認証はもちろん、接続してきたクライアントの認証も可能である。通常のhttpdでも

・usernameとpasswordによる認証
・IPアドレス

によるアクセスコントロールが可能であるが、

・盗聴
・なりすまし

には対応できない。

SSLのクライアント認証機能ではX.509のDN（Distinguished Name）によりアクセスコントロールが行え、

・ある認証局で発行されたものしかアクセス許可しない
・メールアドレスが“*.jp”のものしかアクセス許可しない

などが行える。

5.3 S/MIME

S／MIME［21］［22］はPEM（Privacy Enhanced Mail）に代わり提唱されている暗号化メールのプロトコルであり、PGPと並び事実上世界標準となっている。S／MIMEではX.509 v3の証明書を用いており、認証局機構としてPKIXのプロトコルを利用する予定である。

S／MIMEは名前のとおり、MIME（Multipurpose Internet Mail Extension encoding）を使い、マルチパートで本文（署名の場合）と暗号／署名文を送る方法である。メールにも利点がある。つまり、メールが届きさえすれば、ファイアウォールの中にあっても大丈夫ということである。

```
Message-ID: <35B9F307.7BAF38C2@icat.or.jp>
Date: Sun, 26 Jul 1998 00:00:23 +0900
From: KITANO Hiroyuki <kit@icat.or.jp>
Organization: Info. Sec. Office, JIPDEC, JAPAN
X-Mailer: Mozilla 4.5b1 [en]  (Win95; I)
X-Accept-Language: ja
MIME-Version: 1.0
To: kit@icat.or.jp
Subject: test
Content-Type: multipart/signed; protocol="application/x-pkcs7-signature";
micalg=sha1; boundary="———ms3AF04F66640B1136A6812B92"
X-UIDL: f36b4c51a13e3313c56b8e7ce8123463

This is a cryptographically signed message in MIME format.

———ms3AF04F66640B1136A6812B92
Content-Type: text/plain; charset=iso-2022-jp
Content-Transfer-Encoding: 7bit

test
———ms3AF04F66640B1136A6812B92
Content-Type: application/x-pkcs7-signature; name="smime.p7s"
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="smime.p7s"
Content-Description: S/MIME Cryptographic Signature
```

```
MIIEIwYJKoZIhvcNAQcCoIIEFDCCBBACAQExCzAJBgUrDgMCGgUAMAsGCSqGSIb3DQEHAaCC
Ap0wggKZMIICQ6ADAgECAgEKMA0GCSqGSIb3DQEBBAUAMGUxCzAJBgNVBAYTAkpQMSkwJwYD
VQQKEyBKYXBhbiBOZXR3b3JrIEluZm9ybWF0aW9uIENlbnRlcjErMCkGA1UECxMiQ2VydGlm
aWNhdGlvbiBBdXRob3JpdHkgVGFzayBGb3JjZTAeFw05ODA3MTcwMjEzMDBaFw05OTAxMTMw
MjEzMDBaMHMxCzAJBgNVBAYTAkpQMQ8wDQYDVQQKEwZKSVBERUMxGjAYBgNVBAsTEUluZm8u
IFNlYy4gT2ZmaWNlMRgwFgYDVQQDEw9LSVRBTk8gSGlyb3l1a2kxHTAbBgkqhkiG9w0BCQEW
DmtpdEBpY2F0Lm9yLmpwMFwwDQYJKoZIhvcNAQEBBQADSwAwSAJBAKWWrGRb19Ez9uAZ8xS7
+mHn89d+nDLrBzW+9GABZGaKnSEYlEMq94BV+XvNQRZtawmTzEehm/d2Id5XeJFLsU0CAwEA
AaOBzzCBzDAPBgNVHRMBAQAEBTADAQEAMGoGCyqDCIGcXwsDAR0CAQEABFgwVqAqhihodHRw
Oi8vY2ExLm5pYy5hZC5qcC9jZ2ktYmluL2NhbG9va3VwcmVxoSiGJmh0dHA6Ly9jYTEubmlj
LmFkLmpwL2NnaS1iaW4vdmVyaWZ5cmVxMDcGA1UdHwEBAAQtMCswKaAnoCWGI2h0dHA6Ly9j
YTEubmljLmFkLmpwL2NnaS1iaW4vY3JscmVxMBQGCWCGSAGG+EIBAQEBAAQEAwIAoDANBgkq
hkiG9w0BAQQFAANBAH4t3nME5ByLBHUVak/372PYpP+5btkiE7WwKXXiDbhxRsJRMx9swaHH
p022HjJJahNHJNptc2XSaIi2aGeLOd4xggFOMIIBSgIBATBqMGUxCzAJBgNVBAYTAkpQMSkw
JwYDVQQKEyBKYXBhbiBOZXR3b3JrIEluZm9ybWF0aW9uIENlbnRlcjErMCkGA1UECxMiQ2Vy
dGlmaWNhdGlvbiBBdXRob3JpdHkgVGFzayBGb3JjZQIBCjAJBgUrDgMCGgUAoH0wGAYJKoZI
hvcNAQkDMQsGCSqGSIb3DQEHATAcBgkqhkiG9w0BCQUxDxcNOTgwNzI1MTUwMDIzWjAeBgkq
hkiG9w0BCQ8xETAPMA0GCCqGSIb3DQMCAgEoMCMGCSqGSIb3DQEJBDEWBBSUKZcYkoQCzCfV
VhSOVaOqMjc5nzANBgkqhkiG9w0BAQEFAARAXJgsI7xgyxo76zGw4Xc1ad8B1YsnRh//KUCn
bYobaPjgd+AXA1xu/PtcaXvkYCsj5rqB5PZdip7iBOA8Z5SweA==
------ms3AF04F66640B1136A6812B92-
```

図25　S/MIME サンプル

5.4　IPsec

IPsec も 1998年に約6年をかけて標準化されたプロトコルである。トランスポーと層で暗号/認証を行うTLSとは違い、ネットワーク層での暗号/認証を行う。

ネットワーク層で暗号/認証を行う利点は、

・上位レイヤが暗号/認証を気にしなくてよい
・Gateway/Firewall/ルータ 等ネットワークの入り口であるマシン1台に

設定すればよい

ということである。

それぞれの Gateway／Firewall／ルータに IPsec を実装し、お互いのGateway／Firewall／ルータに公開鍵と IP アドレスを登録すれば、あとは IPsec が勝手に暗号／認証を行って、VPN（Virtual Private Network）のようにセキュアに通信が行える。

すなわちアプリケーション側から見れば、如何にも Local LAN で接続しているのと同様に SQL を投げたり作業が行える。

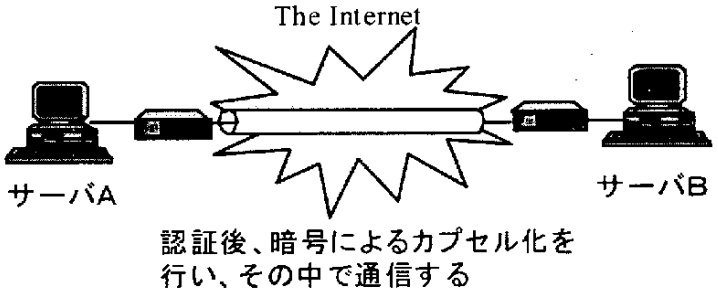


図26 IPsec 概念図

次世代の IP である IPv6 は IPsec をサポートするようであるので、今からIPsec のプロトコルを実装するのは問題ではない。

5.5 LDAP

LDAP（Lightweight Directory Access Protocol）とは、ITU-T X.500 の DAP (Directory Access Protocol) を Internet 上でも稼動するよう 90%の実装を 10%のスピードで行えるようにしたものである。DAP の Internet 側の gateway として設計され、本々 DAP にあった階層構造は実装されてない。

LDAP は X.509 とネーミングについて親和性があり、無視するわけにはいかない。今後、暗号化メールのためのX.509 公開鍵証明書の Repository（保管）に限らず、サーバへログインするときのユーザ認証としてLDAP を使った個人認証が行われるようになるようである。

5.6 最適なプロトコル

一般的なネットワーク構成とともに最適なプロトコルを考えてみる。

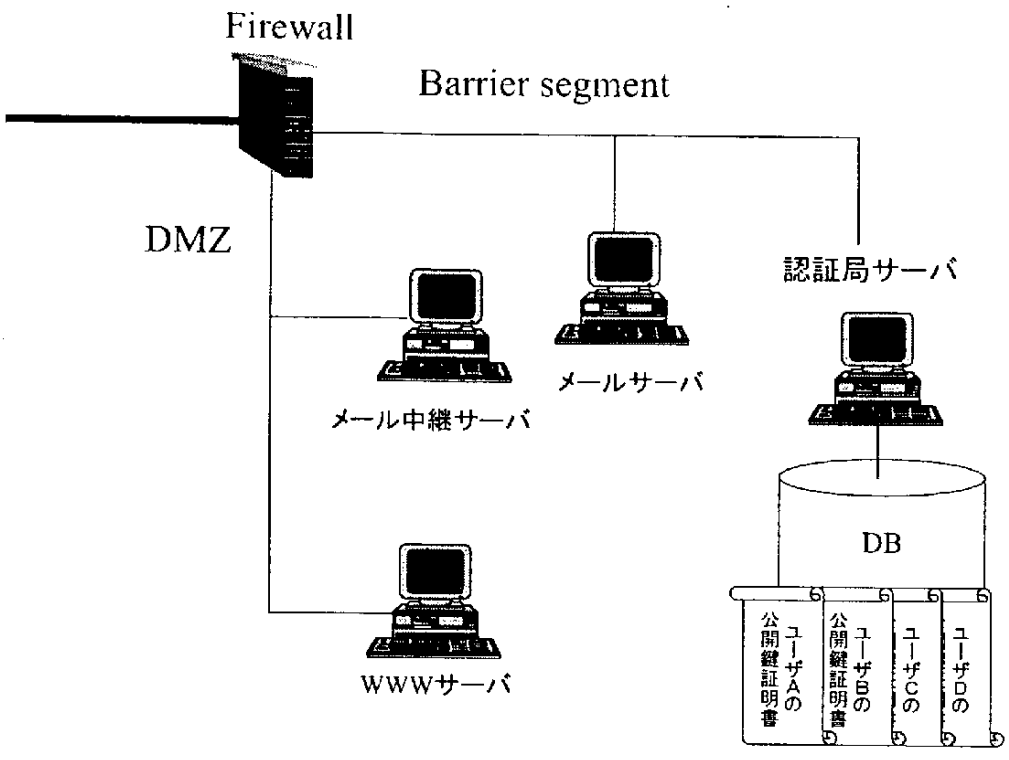


図27　一般的なネットワーク構成

通常は Firewallを設置し、DMZ（DeMilitarized Zome - 非武装地帯）に WWW サーバとメール中継サーバが置いてある。Internet の外と中とで直接通信するのは制限し、電子メールや Proxy などはそれぞれの中継サーバをおいて通信を行えるようにするのが通常である。

DB サーバはセキュアである必要があるので、直接通信を行うのは控えたい。よって下図のように DMZ に中継サーバを置き、外部と内部を Proxy のように中継するのが望ましい。

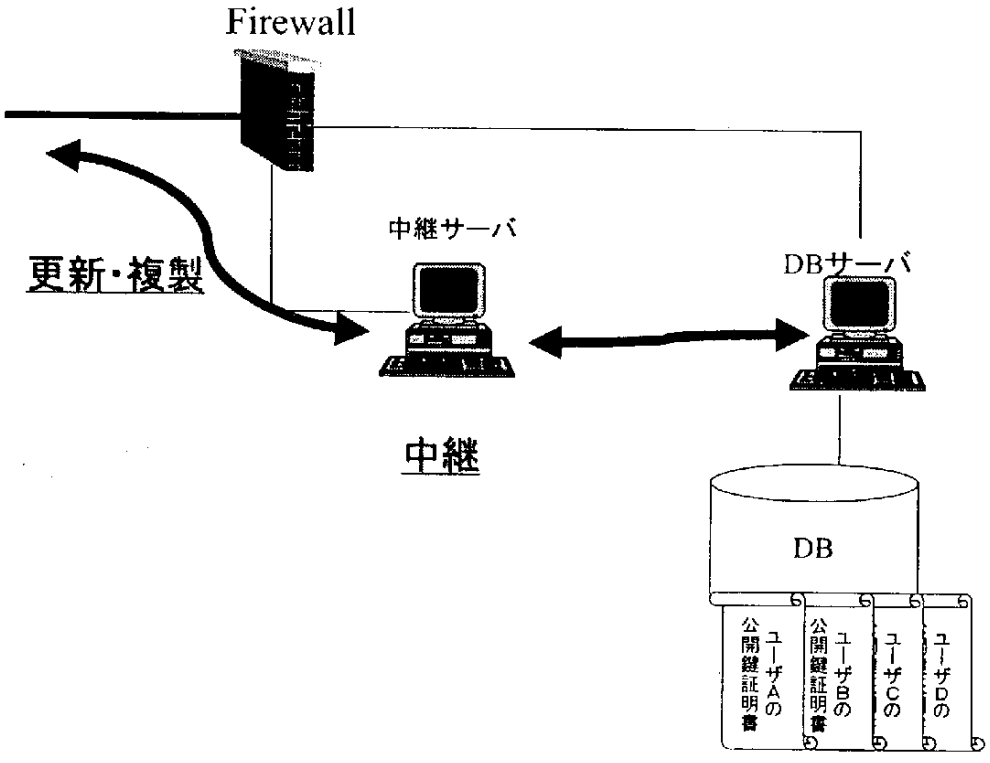


図28　中継サーバ

中継するのはメールでもいいし、SSL, IPsec などのプロトコルでもいい。更新・複製するデータ自体をセキュアに、さらに署名をしたいのであれば

・SSL を使ったトンネリング通信
・S/MIME を使ったバッチ的処理

のどちかかが望ましいであろう。

公開鍵証明書に限って言えば、元々公開するものであるので、下図のような構成も考えられる。図17と異なるのは中継があるかないかの違いだけであり、使用するプロトコルは同じである。

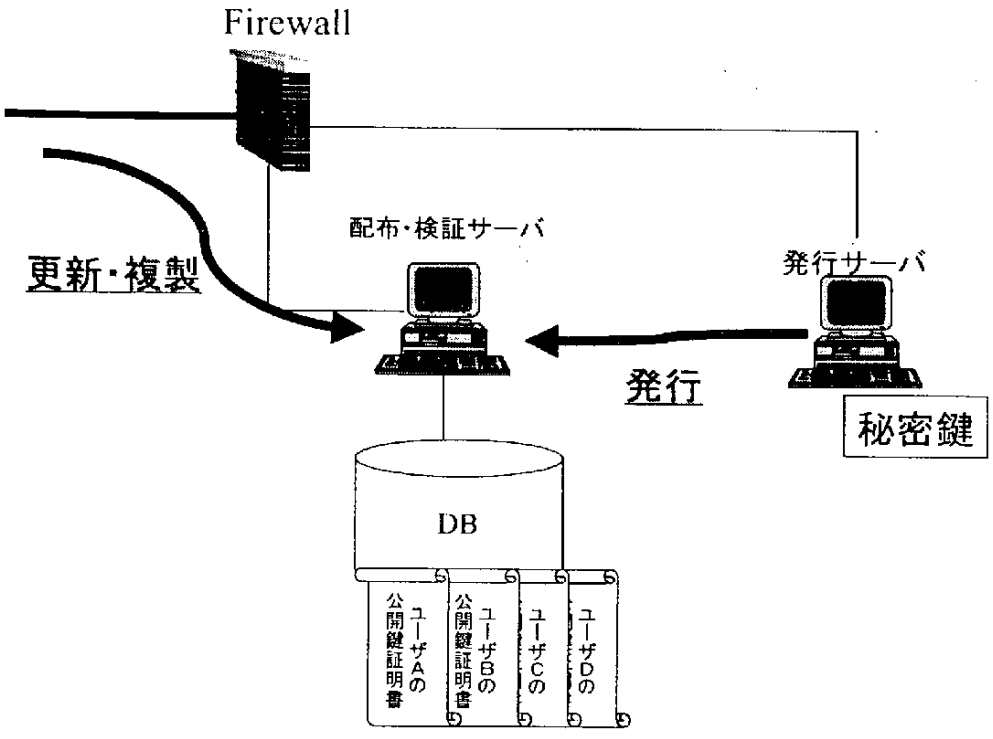


図29　DBをDMZに置く場合

実装をする上では、これらプロトコルと別途考えなければならないことがある。暗号を利用した製品には

・暗号アルゴリズムの疲弊化
・ワッセナー協定に基づく暗号製品の輸出規制

などの問題点がある。暗号アルゴリズムはコンピュータスピードの凄まじい上昇により、鍵長を長くしなければならなくなるかもしれないし、もしかしたらアルゴリズムを変更した方がいい場合も出てくる。
さらに輸出規制への対応として、今まで述べた通信/認証のプロトコルとは別に、

複数の暗号アルゴリズムをサポートできるようなカセット方式

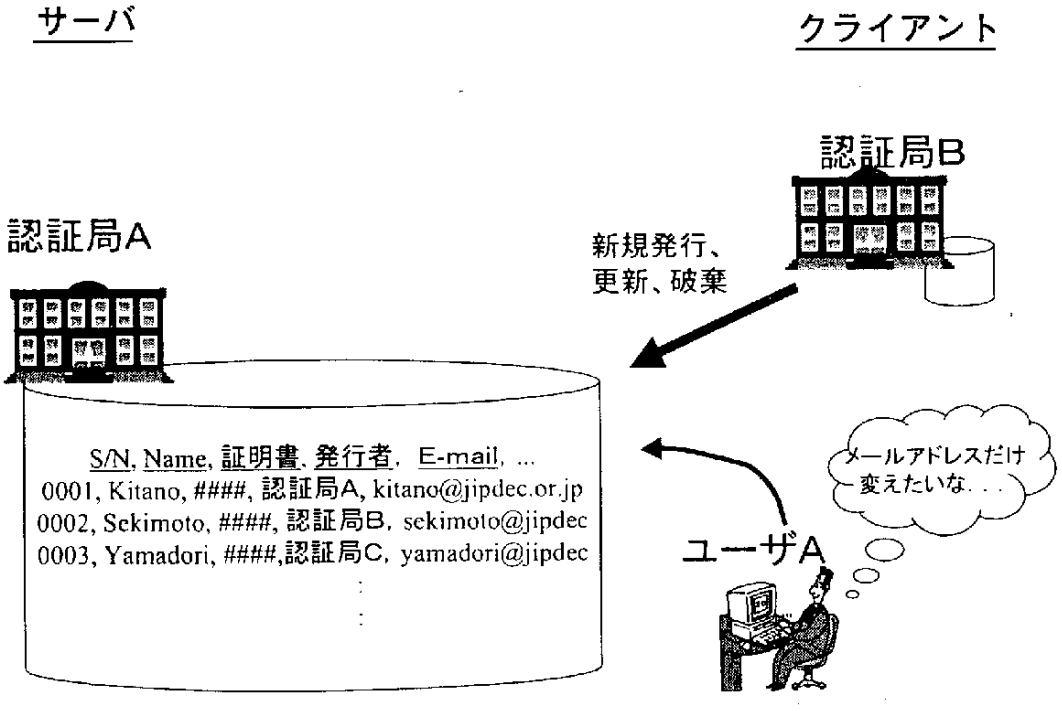
が望ましく、暗号ライブラリを別パッケージとして、任意に実装、取り替えられる方式を取るのが今後賢明な方法であろう。

6． データベースへのアクセスコントロール

前章では、データベース間の通信について検証を行ったが、本章では通信手順が確立した後のアクセスコントロールの方法についてを行う。つまり、データベースアクセスの認証をどう行うかであるが、ここでクライアントとして、

・ユーザ
・他の認証局

の２つが考えられる。



認証後のデータ（or フィールド）更新、削除の権限設定は？

図30　データベース更新の問題点

アクセス制限を行う方法として現在考えられるのは、

（1）パスワード

（2）OTP（One-time Password）
（3）Kerberos
（4）公開鍵
（5）SOCKS
（6）公開鍵証明書
（7）LDAP

である。

6.1　パスワード

ユーザー名とパスワードの組で認証を行う。すなわち、
「正しいユーザがそのユーザしか知らない秘密のパスワードを知っている」ということを認証の拠り所にしている。周知の通り、平文のパスワードには

・盗聴
・なりすまし

の問題がある。
経路上を暗号でカプセル化し、そのカプセルの中で行うとすれば“盗聴”の心配はないが、“なりすまし”の問題は解決できない。
なお、“パスワード”では8文字までしか設定されないことが多く、現在ではパスワードより長い“パスフレーズ”が使用されるようになってきている。フレーズと“I am very hungly through these days”など慣用句で設定できるため、長くなっても覚えやすくなる。

6.2　OTP（One Time Password）

ワンタイムパスワードとは、元々盗聴されてもいいように一回こっきりのパスワードを使用できるようにするパスワード認証方法であった。
方法としては

（1）シーケンス型
（2）時間同期型
（3）チャレンジレスポンス型

の3通りがある

6.2.1 シーケンス型

シーケンス型は、毎回異なるパスワードを利用するために設計されたもので、OTPの作成自体は

パスワード＋シード（種）

で逐次ハッシュをかけて生成しているようである。以下のようにまとめてOTPを計算できる。

```
suzy(kit)% key -n 10 10 kitano
Reminder - Do not use this program while logged in via telnet or rlogin.
Enter secret password:
1: LADY MOOR OTTO FELL COKE DOVE
2: TURF DAD LEG BUY TEET TONE
3: OWLY GUST ROB DIRT NEAR LAB
4: HAW DELL VIE SANK SLOG FORT
5: RICH TOOT LOS TERN REND PUP
6: DARK PUP ORAL DINE LAID FORT
7: VINE TOLD BIT LAWS FORT DISH
8: FALL FEAT BAT IRE DO LESS
9: JOIN HOG LASS DEEM CRAG RIG
10: DUMB SON ADAM ALSO NECK BEAM
```

図31 S/key のOTP作成例

telnetやftpなど既存のシステムを若干変更するだけで使える場合が多く、UNIXユーザの間では広く使われていたが、最近ではWindowsやMac用のS/Key OTP作成プログラムもあるようである。

6.2.2 時間同期型

時間同期型は、パスフレーズの生成をサーバ、クライアントで

パスワード＋時刻

でその都度生成するものである。一般には“分”までの情報を使うので、1分以内であれば、同じパスワードということになる。

6.2.3　チャレンジレスポンス方式

チャレンジレスポンス方式は、サーバが時間毎に異なるシードを送り、クライアント側がそのシードと時刻情報を元に OTP を作成する。接続要求を受けたサーバも同様に OTP を作成し、ユーザ名と一致することで認証を行う。

製品では SecureDynamics 社の SecurID などがあるようである。

6.3　Kerberos

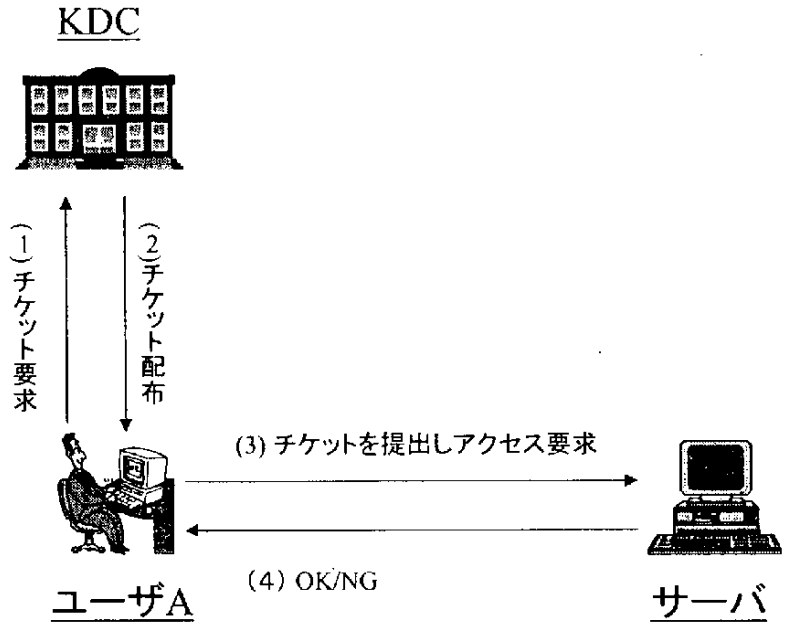


図32　Kerberos の概略

Kerberos とは MIT と IBM で開発された鍵管理システムであり、共通鍵を基本としている。Kerberous では鍵配布センター（KDC）に各ユーザの共通鍵を登録し、ログイン要求時に時間限定のチケットを KDC から受け取る。このチケットはアクセスのためのパスワードが暗号化されており、要求者以外は読めないようになっている。

Kerberos は RSA による公開鍵方式が確立する前に発表されたものであるが、MIT をはじめ現在でも多くのところで使用されている。また、FreeBSD などの OS にもフリーで実装されている。

6.4　公開鍵

SSH（Secure Shell）では公開鍵のみによるアクセスコントロールを用いて、暗号化によるセキュアな通信を行うことができる。

（1）予めそれぞれのサーバにクライアントの公開鍵を登録する
（2）暗号化されたクライアント秘密鍵を復号するパスフレーズを入力する
（3）サーバ側で公開鍵を使って暗号のやりとりができるか確かめる

SSH は rlogin と同様の UNIX コマンドであり、rcp にかわる scp などのコマンドも付属する。

SSH はフリーであり、ソースファイルも取得できるので利用したいプロトコルを SSH 対応にするのはたやすいようである。

```
suzy（kit）% less .ssh/authorized_keys
1024 37 135008134065571177176813010330741051586586189324854683499113436713056755
870444307855678308624850666263865104892957132602583124626079664146111656314101553482813393784573121944509535629900902890411791108840536728580929859020405855506552703458463587691251070006902108734043757304644032554256583391837794030788427 kit@luci.jipdec.or.jp
```

図33　SSHのアクセス制限に使うauthorized_keys

6.5　SOCKS

SOCKS は NEC が開発した Proxy サーバで、GSS-API に基づいた個人認証が行える。Firewall の中から外へのアクセスはもちろん、Firewall の外から中へアクセスできるので、VPN（Virtual Private Network）などにも利用できる。

SSH 同様ソースが公開されているので、利用したいプロトコルを OCKS 対応にするのはたやすいようである。例えば上記 SSH も SOCKS 対応にすることができる。

6.6　公開鍵証明書

発行元である認証局毎にアクセスコントロールすることは簡単である。すなわちアクセスコントロールリストに認証局の公開鍵証明書を置き、その証明書で検証できるユーザ証明書でなければアクセスを拒絶する、ということである。これは実際フリーの http ソフトである Apache を SSL 対応した Apache-SSL で可能である。

```
# Set SSLVerifyClient to:
# 0 if no certicate is required
# 1 if the client may present a valid certificate
# 2 if the client must present a valid certificate
# 3 if the client may present a valid certificate but it is not required to
#   have a valid CA
SSLVerifyClient 1
```

図34　Apache-SSL でクライアント認証の設定（httpd.conf）

上記のように設定し、アクセスを許可したい発行元の認証局の証明書を指定すると、下図のようにユーザ証明書を求められ、指定した認証局から発行されたユーザ証明書を持っている者のみがアクセスできるようになる。

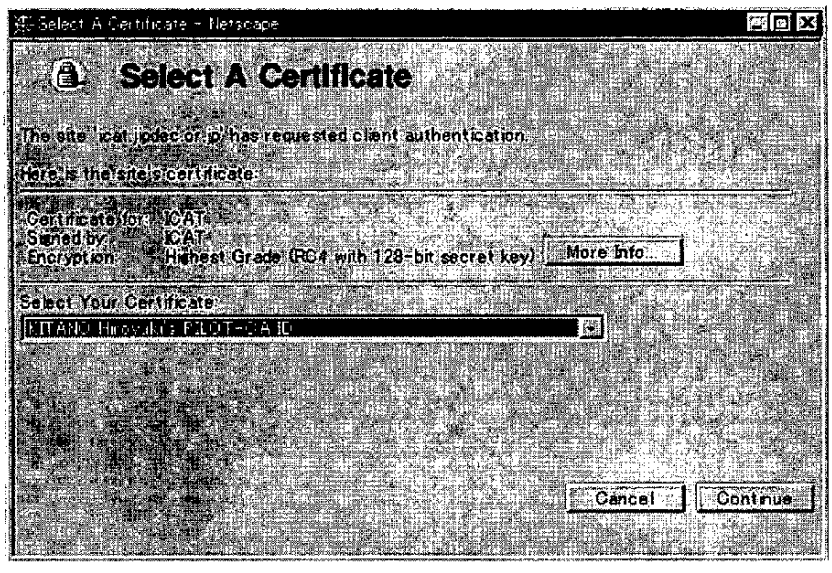


図35　SSLのクライアント認証

各ユーザのアクセスコントロールに関しては、DN情報により区別することができる。

6.7　最適なアクセスコントロール

オープンネットワークで使用するので、SSL や IPsec で暗号化のトンネリングを行えば平文パスワードでも十分使用できる。ただし、認証された仲間同士での盗聴・攻撃や間違いもあるので、平文パスワードは望ましくない。

一方、セキュア通信で使用するトンネリングで使用した公開鍵証明書をアクセスコントロールにも使うのが便宜的にも適しているようである。

7．　参考文献

[1] http://www.goo.ne.jp/
[2] http://altavista.digital.com/
[3] RFC1421～1424,
[4] http://bs.mit.edu:8081/
[5] SET
[6] SSL Apache
[7] http://www.isoc.org/
[8] 1998年12月 第43回 IETF にて IPRA 管理者である Jeffery I. Schiller 氏との会談
[9] http://www.ietf.org/
[10] pkix
[11] http://www.watch.impress.co.jp/internet/www/article/1999/0212/cia.htm
[12] http://www.ietf.org/internet-drafts/draft-ietf-pkix-webcap-00.txt
[13] http://www.ietf.org/html.charters/wg-dir.html#Security_Area
[14] LDAP インターネット ディレクトリ アプリケーション プログラミング、ティム・ハウズ＋マーク・スミス著、松島栄樹＋岡薫訳、プレンティスホール出版、1997年11月、ISBN4-89471-008-0
[15] http://www.ietf.org/html.charters/pkix-charter.html
[16] http://www.ietf.org/html.charters/nfsv4-charter.html
[17] http://www.ssleay.org/
[18] http://www.apache.org/
[19] http://www.apache-ssl.org/
[20] http://www.ietf.org/html.charters/cat-charter.html
[21] RFC2311
[22] RFC2312
[23] http://www.icat.or.jp/icap/
[24] http://www.icat.or.jp/workshop/19981119/matsui/
[25] “暗号のおはなし” 今井秀樹著、日本規格協会、1993年、ISBN4-542-90159-9
[26] http://www.iisi.co.jp/reserch/GCC-gaiyou.htm
[27] ICAT 報告書“暗号アルゴリズムの設計と実装” ICAT事務局編、1998年
[28] http://www.advance.co.jp/icdiv/faq/kps02.html
[29] “カオスの素顔” ニーナ・ホール編 宮崎忠訳、講談社、1994年、ISBN 4-06-257029-7

平成11年３月発行

発　行　　財団法人　データベース振興センター
東京都港区新橋二丁目１３番８号
新橋東和ビル５階
TEL ０３－３５０８－２４３０

委託先　　株式会社　イフ・アドバタイジング
東京都新宿区四谷４丁目２１番３７号
TEL ０３－３３５５－１５９４

印刷所　　株式会社　現代工学社
東京都新宿区四谷２丁目１３番
TEL ０３－３３５７－７１６１