認証局の信頼モデルと証明書パス検証
～S/MIMEを利用したPKIの電子署名検証と証明書パス構築・検証～

PKI認証局の構築と証明書パス検証の実験結果を報告する。今回の実験は、従属階層型モデルとブリッジ型相互認証モデルの認証局を利用した場合の証明書パス構築・検証がどのように動作するかを中心に行い、各種機能の実装や、利用上の問題点を考察した。また、Windows XPで新しくなったCryptoAPIを用いた証明書パス検証の動作検証も行った。

技術研究部　岩崎貴行

1．はじめに

1.1　背景

PKI[*1]が登場してしばらくは、SSLを中心とした暗号化や認証技術に注目が置かれてきた。しかし、これらの技術もさることながらPKIの本質は「電子署名」にあると筆者は考えている。電子商取引等において、電子署名を用いた「改ざんの防止」「本人性の確認」「否認防止」は非常に重要な要素となる。また、文書やデータに対する署名だけでなく、電子証明書の検証にも電子署名が利用され[*2]、署名の検証および証明書の検証はPKIの本質的な部分である。このように、電子署名や証明書の検証は、PKIにおいて非常に重要な要素となる。しかし、異なるPKIドメイン間での検証や証明書失効情報の確認、オンラインでの検証など、様々な難題を抱えているのも事実である。また、アプリケーションによる実装の違いも多く存在し、相互運用性の問題も指摘されている。

一方、GPKIでは相互認証を用いたブリッジ型の認証局信頼モデル[*3]が採用され、近年では主要な商用認証局製品では、ほとんどの製品で相互認証がサポートされるようになってきた。今後、GPKIの普及やPKIの基盤が整備されるにつれ、相互認証の技術は重要になってくるであろう。しかし、それを利用するアプリケーションは、そのPKIベンダー独自のものを除いてほとんど対応できていないのが現状である。

このような状況の中、Windows XPでは以前とは異なるCryptoAPIが採用されており、部分的ではあるが相互認証環境における証明書のパス検証が行えるようになったようである。しかし、この部分についてはまだ有用な情報は少なく、発展途上という感を受ける。

1.2　目的

前述の背景で述べたように、電子署名とその検証は非常に重要なPKIの要素であるにもかかわらず、問題が多い部分でもある。そこで今回はこの「電子署名」に焦点をあてるとともに、Windows XPでの証明書パス検証の動作を確認し、問題点等を抽出する。

また、証明書の失効情報の確認についても、パス検証の一部として検証していくことにする。加えて、署名文書が改ざんされた場合など、実際にはあまり目にすることが無い状況でアプリケーションがどのように動作するかも確認していく。

以下に、本調査・実験での主な注目ポイントを挙げる。

●電子署名の検証（S/MIMEを利用）
- 正常な状態での署名検証
- 本文が改ざんされた場合の動作
- Fromアドレスが詐称された場合の動作

●さまざまな信頼モデルにおける認証局の構築
- 認証局自体の構築
- 各信頼モデルにおける、証明書パス構築
- 証明書失効情報を含めた、証明書パス検証

＊1）Public Key Infrastructureの頭文字語。GPKIのGはGovernment。

＊2）証明書が正しい認証局から発行されたものであることを保証するために「電子署名」を証明書に組み込むのであるから、いわば当然ではある。

＊3）階層型、メッシュ型、ブリッジ型などの形態がある。IPA/ISECの「PKI 関連技術解説」にわかりやすい解説がある。

なお、失効情報の確認については、CRLと呼ばれる失効情報リストをHTTPサーバーもしくはLDAPサーバー上に配置し、各アプリケーションが参照する方法を採用する。このような確認方法は、動的検証時のパフォーマンスの問題などが指摘されており、OCSPやSCVPなどの規格が存在もしくは検討されている。しかし、実装されているアプリケーションはほぼ皆無であるため、従来どおりのCRLによる確認（HTTP、LDAPによるオンライン確認 およびファイルをダウンロードしてのオフライン確認）を採用することとした。

2．実験環境の概要

2.1　構成および使用製品・アプリケーション

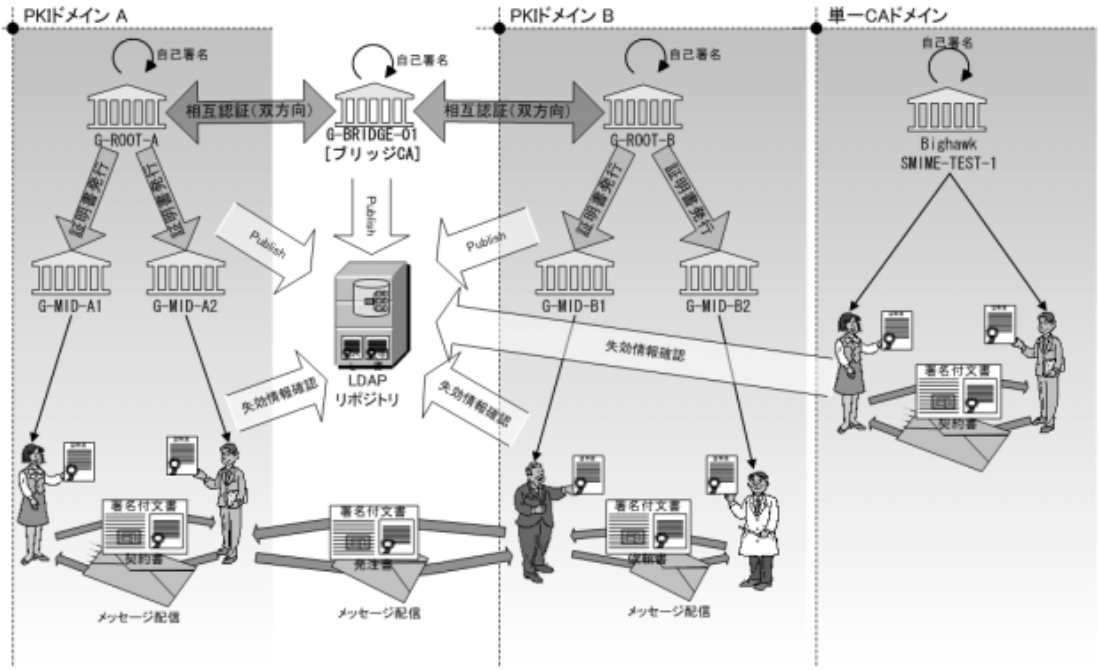
図1に示すようなPKI環境を構築した。複数のユーザー間でS/MIMEメールをやりとりすることによって、署名検証、証明書パス構築・検証を行う。認証局にはRSA Security社のKeon Certificate Authority 6.0.2（以下KeonCA）を利用する。ブリッジ認証局用に1台、その他のすべての認証局用に1台サーバーを用意[*4]して構築。また、LDAPサーバーは別途Solaris9を利用して構成した。

また、本実験に利用した主な製品・アプリケーションは表１のとおりである。

3．実験内容と結果

3.1　事前準備

実験環境の設計を行う。構成については前章を参照。認証局および認証局が発行するユーザー用証明書のプロファイルについては、本稿末尾にある「証明書プロファイル指



・G-ROOT-AおよびG-ROOT-Bをルート認証局とする従属階層型のPKIドメイン※「PKIドメインA」、「PKIドメインB」を構成する。また、単一の独立した認証局「Bighawk SMIME-TEST-1」を作成する。
・G-ROOT-AおよびG-ROOT-Bは、下位の認証局G-MID-XXを持つ。
・各ユーザーは、G-MID-XXもしくはBighawk SMIME-TEST-1に証明書の申請を行い、取得する。
・各ユーザーは、G-ROOT-Xの証明書をルートストア（信頼されたルート証明機関）に、G-MID-XXの証明書を中間証明機関ストアに格納する。これにより、G-ROOT-Xを信頼ポイントとする証明書チェーンが構築される。
・単一認証局を利用する場合は、Bighawk SMIME-TEST-1のCA証明書をルートストアに格納し、同認証局から証明書の発行を受けたユーザー同士で署名文書（S/MIME）のやり取りを行う。
・従属階層型モデルの検証を行う際は、PKIドメインA内で署名文書のやり取りを行う。
・ブリッジ型モデルの場合は、PKIドメインAとBのユーザー間で署名文書のやり取りを行う。
・各認証局は、外部LDAPリポジトリ（図中）もしくはCA内部リポジトリ（図中には示していない。HTTP or LDAP）に証明書、CRLを格納する。

※「PKIドメイン」という言葉に明確な定義はないが、ここでは、あるポリシーを共通にもつ認証局とエンドエンティティの集まりとする。一般に、１つまたは複数の信頼ポイントとなりうる認証局をもち、そのドメインのユーザーはそのいずれかの認証局を信頼する。本実験では、1つのPKIドメインには1つの信頼ポイントを置く形をとっている。

図１　実験環境概念図

＊4）KeonCAでは、同一ホスト上で稼動している認証局同士の相互認証（CrossCertificate作成）が行えないため、物理的なサーバーを分離した。このようなケース以外では、同一サーバー上に複数の認証局を作成できる。

表1　使用製品＆アプリケーション

認証局：
・RSA Keon Certificate Authority 6.0.2（評価・初期導入用のパイロットパック）
・Windows 2000 Server SP2

クライアント：
・Windows 2000 Professional クライアント
メールアプリケーション：
Outlook Express 6, Outlook 2000, Outlook 2002
Becky! Internet Mail 2.05.04 + S/MIME Plug-in Ver-0.9.0
WinBiff 2.42 + S/Goma 2.14
・Windows XP Professional クライアント
メールアプリケーション：
Outlook Express 6, Outlook 2002

リポジトリ（LDAP）サーバー*1：
・Solaris 9 + iPlanet Directory Server 5*2

＊1）KeonCA内部リポジトリ（LDAP、HTTP）もリポジトリとして利用する
＊2）DirectoryServerはSolaris 9に統合されている

針」のとおり方針を定め、これをもとに実験内容に応じて適宜変更して適用することとした。

その後、KeonCA、LDAPサーバーの初期構成を行い、実験を開始する。また、KeonCAの管理操作を行うための証明書（SSLのクライアント証明書）を作成し、管理を行うPCに格納しておく。

3.2　認証局の構築（単一認証局）

証明書プロファイル指針に従い、認証局を構築。この作業はKeonCAのWebインタフェース上から行う。単一の認証局であるため、自己署名証明書を作成する。

また、認証局構築後はKeonCA全体を再起動して新しく作成した認証局をKeonCAに認識させる必要がある。

3.3 エンドユーザーからの証明書の申請と発行

Keonの証明書申請用Webインタフェースを利用してクライアントのブラウザから証明書の申請を行う。

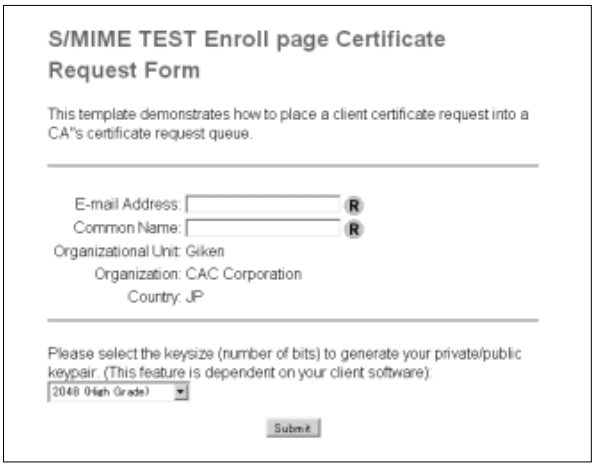


図2　Webインタフェースを利用した証明書の申請ページの例

その後、管理者（もしくは、申請の審査を行う担当者）が内容を確認し、受理する場合は申請に対し認証局の秘密鍵で署名を行う。これにより、申請したユーザーの証明書が発行される。

証明書発行の設定で、申請受理の際にユーザーにメールを送信する設定にすることにより、受け取ったメールに記述されているURLにアクセスすることで自分の証明書をダウンロード可能である。なお、申請を行ったPCとは異なるPCやブラウザでこのURLにアクセスしても、申請に使用した鍵ペアの秘密鍵を所有していないためにダウンロードは不可能である（図3）。

図3　異なるPC、ブラウザで証明書を取得しようとした際のエラーメッセージ（上はInternet Explorer、下はNetscapeの例）

3.4　単一認証局モデルにおける署名付きメールの送受信

取得した証明書を用い、電子署名付きメールを送信する。送信先は、同じ認証局を信頼しているユーザーとする。

利用したアプリケーションは、表1「使用製品・アプリケーション」に記載したとおり。さまざまなメールアプリケーションを用いて、メールの送受信・署名検証を行った。このテストに関してはすべてのメールアプリケーションで特に問題なく動作した。

3.4.1　単一認証局モデルー実験結果

単一認証局のもとでの通信では、基本的にパス構築の問題は起こらない。

次ページ図4は、Outlook 2000で受け取ったメールの署名を確認した例と、証明書のパスがどのようになっているかを確認した例である。署名が正常な場合の動作については、今回実験に使用したいずれのメールアプリケーションでも機能的な差はなかった。

また、Outlook Expressでは、メールの内容やFromアドレスの改ざんを検出すると、次ページ図5のような警告が表示される（一瞬どきりとしてしまうデザインではある）。

しかし、一部のアプリケーションでは、Fromアドレスと証明書のメールアドレスが異なる場合に、何もメッセージが表示されなかった。この場合でも、Fromアドレスと

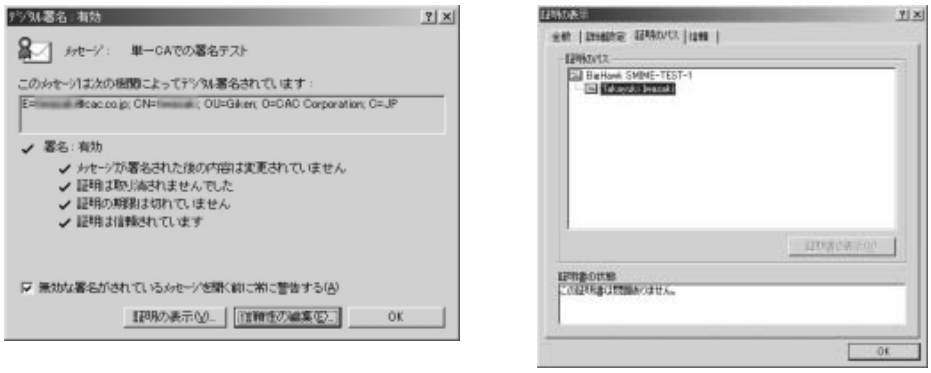


図 4　署名検証結果と証明書のパス（Outlook 2000の例）

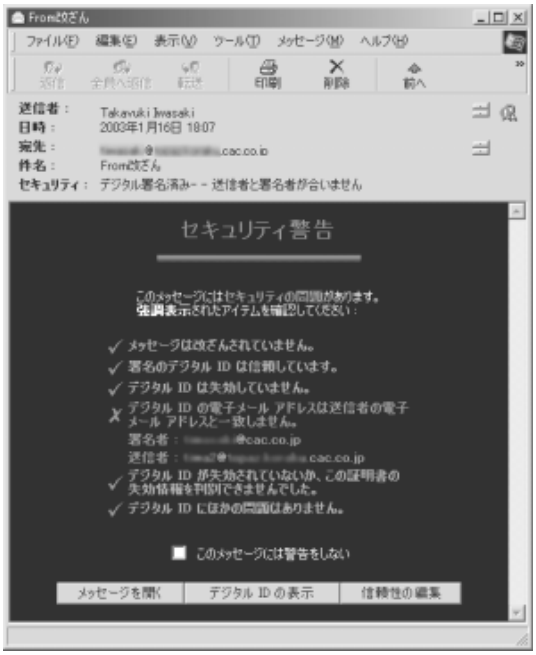


図 5　アドレスが詐称されていた場合の警告メッセージ（Outlook Expressの例）

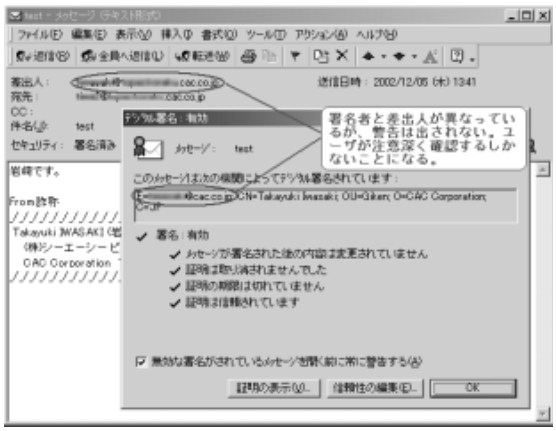


図 6　Outlook2000では、Fromアドレスが詐称されていても警告などは表示されない

証明書のメールアドレスを注意深く確認すれば発見できるが、一般のユーザーがここまで確認することはまず考えられない。これでは、発信者の詐称がたやすくできてしまうことになりかねないので注意が必要である（図 6 ）。

3.5　従属階層型信頼モデル下での検証

3.4で用いた認証局とは別にルート認証局G-ROOT-Aを作成。その後、G-ROOT-Aが署名する下位の認証局G-MID-A1、G-MID-A2を作成して2階層の認証局モデルであるPKIドメインAを構築。ユーザーA1およびA2はそれぞれ、G-MID-A1、G-MID-A2に証明書の申請を行い、発行を受ける。また、G-ROOT-Aを信頼ポイントとして、G-ROOT-Aの証明書をルートストア*5に格納する。

その後、ユーザー同士で署名付きメールの送受信を行い、正しく署名検証が行えることを確認する。なお、メールには証明書チェーン*6を添付して送付する設定とする。

この構成では、メールアプリケーションが多階層の証明書の検証をサポートするかどうかが鍵となる。

3.5.1　階層型信頼モデル－実験結果

従属階層型信頼モデルにおいては、Becky!のS/MIME Plug-inが多階層証明書の検証をサポートしていなかったほかは特に問題なく動作し、署名検証を行うことができた。今回の実験で利用したPKIドメインでは、ルート認証局と

＊5）Microsoft Management Console（mmc）やIEでは「信頼されたルート証明機関」、S/Gomaでは「ルート証明書」などと表現される。これらの、信頼の頂点にあたる認証機関の証明書を格納する場所を指して「ルートストア」と表現する。ここに格納される証明書は証明書の検証パスの頂点になり、信頼ポイントとなる。

＊6）自分の信頼ポイントである認証局の証明書から、自分の証明書までたどることができるような複数の証明書をまとめたもの。ユーザーA1の例では、G-ROOT-A→G-MID-A1→ユーザーA の3枚の証明書が含まれる。

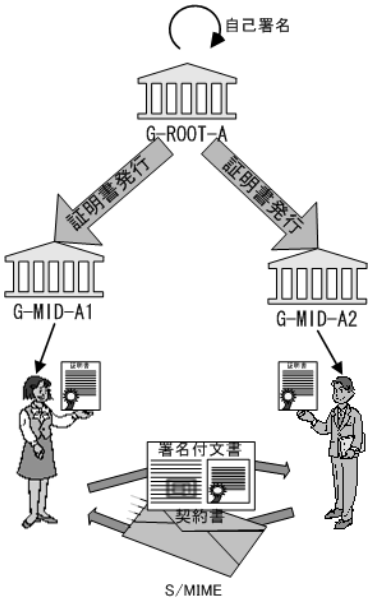


図7　PKIドメインAの認証局構成

ユーザーの証明書の間に中間認証局が1つ存在する。この中間認証局の証明書は、メール送信時に「署名付きメッセージで証明を送信する*7」設定になっていれば、電子署名とともに、署名に使用した証明書、ルート証明書、中間認証局の証明書を含んだ証明書チェーンが添付されて送られるので、特に意識することなく利用可能である。

3.6 ブリッジ型信頼モデル下での検証

前出のPKIドメインAと同様の手順で、新たにPKIドメインBを作成し、ユーザーは証明書の申請と取得を行う。また、ブリッジ認証局として、自己署名証明書をもつ認証局を作成。両PKIドメインのルート認証局（G-ROOT-A、G-ROOT-B）と双方向の相互認証を行う。その後、PKIドメインAとPKIドメインBのユーザー間でS/MIME署名付きメールの送受信を行う。

ブリッジモデルでは、メールに証明書チェーンを添付したとしても、それだけでは証明書のパスを構築できない。これは、証明書チェーン内には、署名者の信頼ポイントからのチェーンしか含まれず、受信者は署名者の信頼ポイントを信頼しているとは限らないからである。このため、ブリッジ型の場合は証明書のパス構築にリポジトリを利用するなどの方法で、他のPKIドメインの証明書も正しく検証できるようにすることが重要である（次ページ図9）。

当初は、LDAPをパス構築のための証明書リポジトリとして利用することを想定していたが、証明書のパス構築にLDAPリポジトリを利用することができるアプリケーションが存在せず、今回はこの実験は断念せざるを得なかった。

Windows XPではWindows2000やそれ以前のWindowsとは異なったCryptoAPIを採用しており、ブリッジ型の証明書パス検証を行うことができる。そのためには相互認証証明書を中間証明機関の証明書ストアに格納する必要がある。この作業を事前に済ませておくことにより、証明書パス構築と検証は問題なく行えた。

以下に、Windows XPの場合の利用手順を紹介する。

まず、相互認証証明書を何らかの方法で入手し、中間証明機関のストアにインポートする。

今回の実験においては、CrossCertificatePair形式のファイルをインポートする方法を発見できなかったため、証明書ペアを別々に分けた証明書をインポートすることにした。具体的には、

- G-BRIDGE-01 から発行された G-ROOT-Aの証明書
- G-ROOT-Aから発行された G-BRIDGE-01の証明書

（以上2つがG-ROOT-AとG-BRIDGE-01の証明書ペア）

- G-BRIDGE-01 から発行された G-ROOT-Bの証明書
- G-ROOT-Bから発行された G-BRIDGE-01の証明書

（以上2つがG-ROOT-BとG-BRIDGE-01の証明書ペア）

図8　従属階層型信頼モデルでの署名検証（左）と、証明のパス（右）

*7）この設定により、証明書チェーンが添付されたメールが送信される。

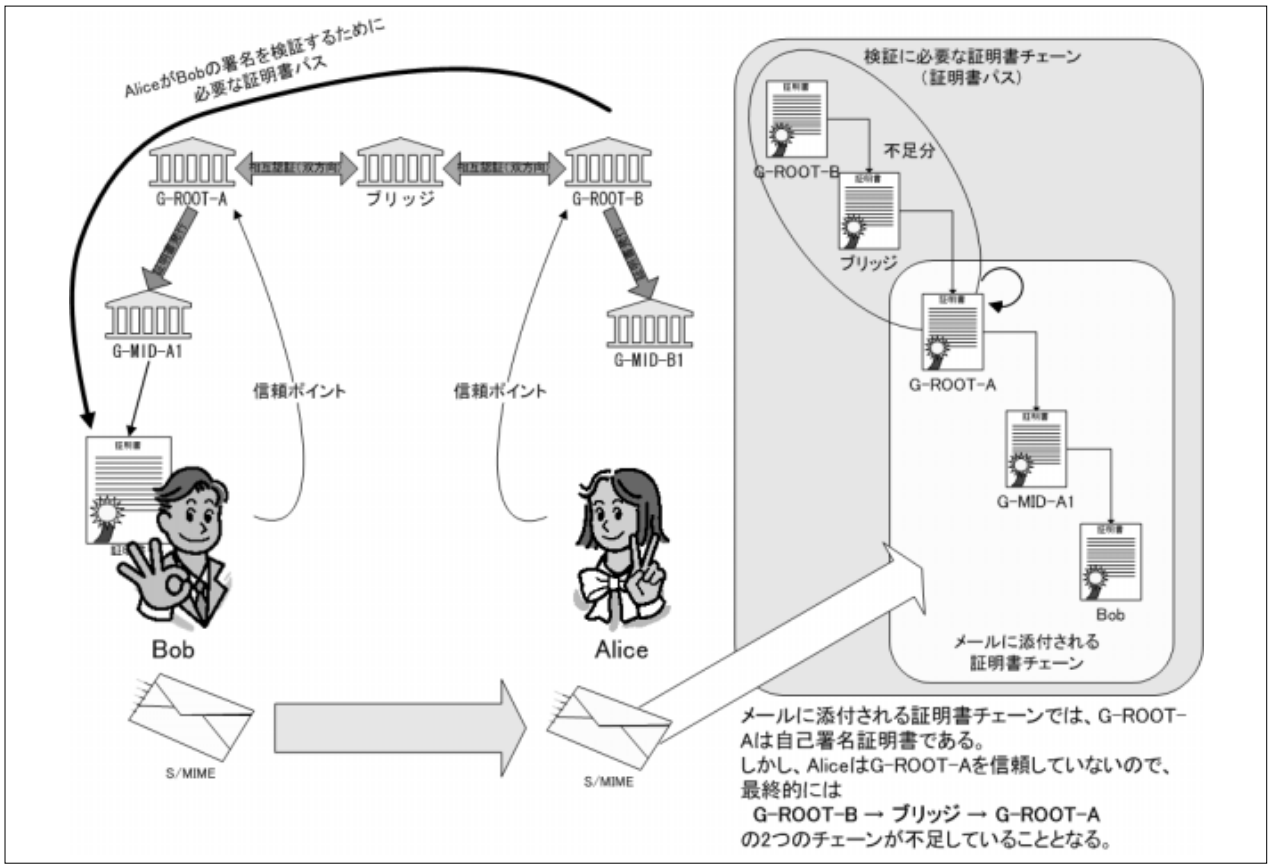


図9　ブリッジモデルで必要な証明書パス

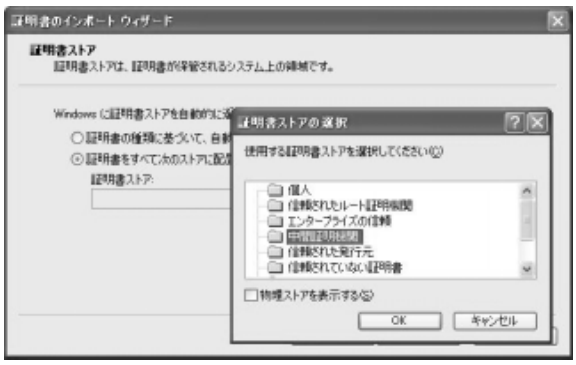


図10　証明書を中間証明機関のストアにインポートする

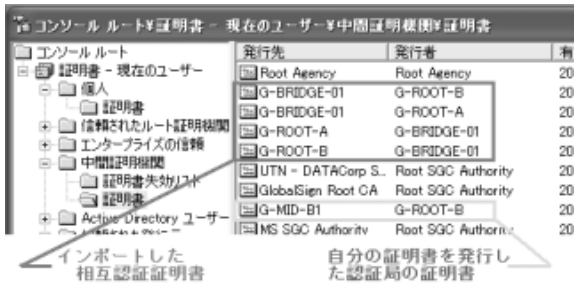


図11　相互認証証明書をインポートした後の、証明書ストアの中身

の4つの証明書を中間証明機関のストアにインポートする。

また、自分の信頼する認証局のルート証明書（今回の構成では G-ROOT-B）をルートストアにインポートし（これはほかの信頼モデルの場合と同様）、自分の証明書を発行した認証局の証明書を（今回は多階層のモデルであるので）中間証明機関のストアにそれぞれインポートする。

図11が、インポート後の中間認証機関ストアの中身。なお、G-MID-B1は自分の証明書を発行した認証局の証明書である（相互認証とは無関係）。

これで、PKIドメインA内のユーザーの証明書を検証する準備ができたことになる。

3.6.1　ブリッジ型信頼モデルー実験結果

PKIドメインAのユーザーからの署名付きS/MIMEメッセージを受信し、署名を検証すると図12、13のようになる。

図12は、メールの署名を検証した結果。Outlookでは、メールのウィンドウの のマークをクリックして確認することができる。

図13は、証明のパスを表示した画面である。ここでの注目点は、署名者の信頼ポイントではなく、自分の信頼ポイントが頂点にあることである。署名者の信頼ポイント（G-ROOT-A）は、あたかもG-ROOT-Bからの中間認証局のように扱われる。この形態が、ブリッジモデルでの証明書パスなのである。

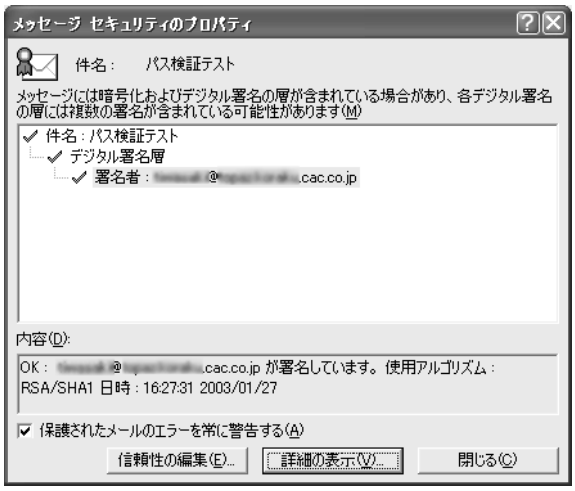


図12　署名の検証結果

図14　中間証明機関に相互認証証明書をインポートしないと、パス構築に失敗する

図13　証明のパス。自分の信頼ポイントが起点となっている

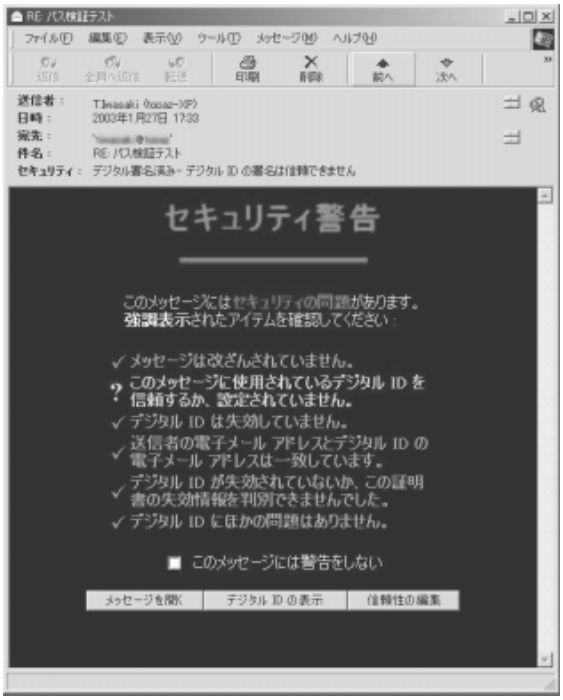


図15　Windows2000ではXPと同様の手順で行っても、証明書の検証ができない

なお、中間証明機関に相互認証証明書をインポートしないと、証明書のパス構築ができず、図14のようになる。

なお、Windows XP以外ではCryptoAPIの仕様により、相互認証を用いた環境での証明書パス検証に対応していない。そのため、XPの場合と同様の手順で相互認証証明書を格納したとしても、検証に失敗することになる。

4．考察

4.1 認証局の構築

KeonCAのインストールおよび初期設定は、ウィザードに従って質問形式で進んでいくため、さほど難しい作業ではない。また、認証局の作成も KeonCA の場合はWebインタフェース上から行うことができ、どのような構成で作成するかがはっきりと決まっていれば、いたって単純な作業となる。ただし、多数の、非常に専門的な用語を用いて構成する必要があり、必要とされる知識レベルは高いものになるだろう。

また、認証局はその性質上削除することができない*8)た

*8）認証局は証明書の発行のほかに、CRLの発行という重要な役目を果たす。このため、証明書を発行した後も、認証局の証明書は有効である必要がある。また、発行した証明書が失効・有効期限切れになったとしても、有効な期間内にその証明書によって行われた署名の検証のためにも必要である。ひとつも証明書をしていない状態であれば削除しても問題は無いように思われるが、KeonCAの場合は認証局の削除は不可能となっている。

め、入力ミス等で間違った設定の認証局を作成してしまうと、リカバリが非常に面倒になる場合がある。

そして、最も重要なのは設計時点で構成をはっきりさせておくことであるということはいうまでもない。

4.2 証明書の申請と取得

製品にもよるが、ブラウザから証明書を申請・ダウンロードするような仕組みを採用している製品も多い。また、製品によっては証明書の申請等を行える専用クライアントソフトが用意されているので、それを利用したほうが便利な場合もある*9。

ただしブラウザを利用した仕組みでは、JavaScriptやVBScriptを利用していることが多いため、ブラウザのバージョンやセキュリティパッチなどによりうまく動作しなくなることがある。このため、ブラウザをバージョンアップした際の動作テストや、最新のブラウザに対応するためのパッチが速やかに提供されるかどうかは非常に重要だ。このことは、設計時点では見落としやすいので注意が必要である。また、Internet Explorerでは動作するが、Netscape Navigatorでは動作しないといったような、ブラウザの種類にも依存する*10ことも多い。

今回の実験に用いたKeonCAには専用のクライアントソフトや管理ソフト等は提供されていないが、証明書の申請から取得までをブラウザを利用して処理することができる。申請が受理されると、申請したユーザーに証明書取得を行うためのURLをメールで自動送信することも可能である。このメールは通常のメールで送信されるが、たとえ、このURLを他人に知られたとしても基本的には問題はない。これは、作成した秘密鍵は自分自身しか所有していないからで、申請したブラウザ以外からも取得できないようになっている。これは、証明書を申請したブラウザの証明書ストア（Internet Explorerの場合は、Windowsの証明書ストア）に申請時に作成した秘密鍵が保存されており、この鍵と、取得しようとする証明書が対のものであるかを検証するからである。

4.3 S/MIMEを用いた電子署名とその検証

単一もしくは従属階層型でPKIドメインを構成した場合の証明書の検証は、証明書のパスの構築における問題は見受けられなかった。Outlook、Outlook Express等では、CRLのオンライン検証も行うことができ、証明書にCRL配布ポイント（CRL Distribution Point）が記述してあれば自動で確認することもできる。

ただし、CRLのオンライン検証をONにした場合、状況によってはCRLの確認に非常に時間がかかる場合がある。一度、実験中にCRL配布ポイントの記述ミスで確認できない状況が発生したが、このときは1つのメールを開くのに十数秒を要した。なお、失効情報の確認については、OCSPその他、オンラインで失効情報を確認するプロトコルもあるが、実装しているアプリケーションは見受けられなかった。

また、一部のメールアプリケーションでは、署名に使用した証明書のメールアドレスとメールのFromアドレスが異なっていても警告が出なかった。メールヘッダは署名の対象ではないので動作としては正常であるとも言える。しかし、これでは簡単になりすましができてしまうことになりかねない。すなわち、正式な手順で入手した証明書を用いて署名したメールを、Fromアドレスを他人のアドレスにかえて送信した場合、受信者はたやすく（というよりは電子署名がついていることにより、むしろ確信をもって）偽られたFromアドレスを信じてしまう可能性があるということだ。このような機能は、アプリケーションに委ねるしかないのが現状で、改善に期待したい。

4.4 証明書パス構築・検証（S/MIMEを用いた署名付きメールの検証）

相互認証により構成されたブリッジモデルを採用し、異ドメイン間での通信を行ったケースでは、Windows XP以外はほぼ利用できない。これはCryptoAPIの仕様によるもので、同じOutlookやOutlook 2002であってもWindows XPとそれ以外のWindowsでは動作が異なる。

Windows XPでは、相互認証証明書ペア（CrossCertificatePair）*11を中間証明機関として証明書ストアに格納することにより、証明書パス構築を行うことができる。ただし、オンラインでの動的なパス構築・検証はサポートしていないようだ。

なお、Windows XP以外では相互認証におけるパス構築はサポートしておらず、OSのCryptoAPIを利用しない独自のアプリケーションでも、サポートしているものは見当たらなかった。

＊9） 実際には、自らのPC上にOpenSSL等の公開鍵暗号方式に対応したアプリケーションを用意し、手動で鍵の作成、証明書署名要求ファイルの作成を行って申請することも可能ではあるが、現実的ではないであろう。

＊10） Javascriptはブラウザによって実装が異なる場合が多く、ブラウザの種類を限定せざるを得なくなる場合も多い。

＊11） 相互認証の際に作成した証明書のペアで、自認証局が相手の認証局に署名した証明書と、相手の認証局が自認証局に署名した証明書の対である。

4.5　全体をとおして

商用の認証局製品については、現時点での基本的な機能（キーリカバリ・証明書拡張フィールドへの対応・相互認証・OCSP・SCEP・CMP、LDAPなど）は、ほぼ実装されてきている。今回利用した製品はRSA社のものだけだが、展示会での紹介やカタログスペック等を見る限り、主要な製品については、ほぼ横並びという状況である。ただし、製品独自の機能やインタフェースには、それぞれ個性がある。基本機能に差がない状況では、これらの機能は運用負荷に大きく関わってくる。認証局に限ったことではないが、製品の選択の際には利用用途によって綿密に検討する必要があるだろう。

一方、クライアント側のPKIアプリケーションは、やっと対応が始まってきた状態と見受けられる。雑誌等では比較的明るい話題が多く紹介されているが、やや視点を遠くにおいて、すでに世間に浸透したインフラ（PKIはインフラである）と比較してみると、明らかにいまだ過渡期であることがわかる。企業内の閉じた環境で、単一もしくはごく少数のベンダーの製品で運用する場合には大きな問題はなくなりつつあるが、ひとたびオープンな環境で運用する必要性が出てくると、さまざまな問題が発生する可能性がある。

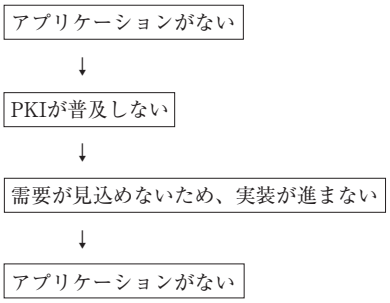
相互認証環境下のパス検証においては、ようやくWindows XPで対応が始まったという状況だ。OS自体にその機能を持たせるか、各アプリケーションが機能を持つべきかは賛否が分かれるところであり、利用用途にも関連する話題であるため一概には言えないだろう。しかしS/MIMEに限って言えば、Outlook、Outlook Expressが（それが最適な選択かどうかは別として）多くの企業ユーザーの標準メールクライアントとして利用されていることを考えると、Windows OS自体がその機能を果たすことには意味があり、Windows XPにおけるパス検証サポートの第一歩と言えるだろう。

なお、現在一般には公開されていないが、今回利用したメールクライアントの１つWinBiff・S/Gomaの開発元であるOrangesoft社は、IPAの「平成13年度GPKI対応電子メールソフトのためのプラグイン開発」プロジェクトにて、ブリッジ認証局をまたいだ証明書パス構築・検証のためのプラグインを開発している。同社では、MozillaやEudora、Becky!などのプラグインも開発し、S/MIMEの普及の一助としたいと述べている*12。

5．最後に

本調査・実験を通し、認証局の各種信頼モデルの構築やS/MIMEを利用したメールアプリケーションの動作について検証することができた。全体的には、ユーザービリティの面でもまだまだであるし、アプリケーションの実装も各種規格に追いついているとはいえない。さらには、多くのユーザーを抱える企業等では認証局の運用もかなりの負荷になると思われる。

一部の企業では実験的にPKIを導入してきており、企業内などの閉じた環境では使われ始めていることは確かだ。しかし、一般的に見ると、PKIの現状は、



という悪循環に陥っている感がある。PKIを爆発的に普及させるには、それなりのインパクトを持つアプリケーションやサービスの登場が必須要件となるであろう。

PKIや電子署名の技術は成熟しつつあると判断できるが、PKIはそれを活用するアプリケーションが非常に重要であることは自明である。いまだ製品やアプリケーションによる実装の相違が多く見受けられる現状では、認証局とともにアプリケーションの相互運用性の確保が最重要課題である。

また、本実験でのメインテーマであった「相互認証・ブリッジ環境下での電子署名・証明書パス検証」については、PKIベンダー独自のアプリケーションを除いて、対応が始まったばかりである。GPKIは基本的に民間は参入できず、政治的な問題もあることから先行きが読みにくい状況ではあるが、PKIが一般にも普及するようになれば、これらの技術が重要になってくるのは確かだろう。

なお、本稿では「ユーザーの目に直接触れるようなPKI」についてしか述べていないことに気付かれた方もいるかもしれない。実はそのとおりで、VPNで利用されることの多いIPsecや、IEEE 802.1x EAP / TLS などもPKIを利用しており、特にIPsecは多く利用されてきている。このような意味から、「PKIは水面下で進化を続け、時が熟すのを待っている」という考え方もあるのではないだろうか。

＊12）http://www.ipa.go.jp/security/fy13/tech/gpki_smime/gpki_smime.html を参照。

＜PKI関連情報URL＞

1. IPA/ISEC（情報処理振興事業協会 セキュリティセンター）のページ内
   http://www.ipa.go.jp/security/pki/index.html（PKI 関連技術解説）
   http://www.ipa.go.jp/security/pki/pki.html（PKI関連技術情報）
2. @IT（アットマークIT）内の記事
   http://www.atmarkit.co.jp/fnetwork/rensai/index/index-serial.html#pki（連載 PKI基礎講座）
3. GPKI関連
   http://www.gpki.go.jp/（府認証基盤（GPKI））

＜参考文献＞

1. Andrew Nash、William Duane、Celia Joseph、Derek Brink著『PKI eセキュリティの実装と管理』翔泳社（2002）
2. Carlisle Adams、Steve Lloyd著『PKI 公開鍵インフラストラクチャの概念、標準、展開』ピアソンエデュケーション（2000）
3. 小松文子著『PKIハンドブック』ソフト・リサーチ・センター（2000）

＜参考資料＞証明書プロファイル指針

以下は、実験で作成する証明書の「証明書プロファイル（証明書エクステンションの設定）」の指針である。これは設計の一部となる。証明書エクステンションは他にも多数存在するが、実験の柔軟性を確保するため初期の段階では特に規定せず、進行に応じて適宜設定していくものとする。

認証局の証明書（単一認証局）	X.509バージョン	V3
	署名アルゴリズム	sha1RSA
	公開鍵	RSA 1024bit
	DN	CN = BigHawk SMIMETEST-1 OU = Giken O = CAC Corporation C = JP
	基本制限	Subject Type=CA Path Length Constraint=None
	鍵使用方法	Digital Signature Certificate Signing Off-line CRL Signing CRL Signing(86)
	CRL配布ポイント	当初は指定しない。実験の進行に伴い、指定することがある

認証局の証明書	X.509バージョン	V3
	署名アルゴリズム	sha1RSA
	公開鍵	RSA 1024bit

	DN	CN = [CAの名称]
		OU = (Bridge ¦ Giken - A ¦ Giken - B) # PKI　ドメインごとの OU
		O = CAC
		C = JP

	基本制限	Subject Type=CA ROOT-CAのみ Path Length Constraint=4
	鍵使用方法	Digital Signature Certificate Signing Off-line CRL Signing CRL Signing(86)
	CRL 配布ポイント	外部 LDAPもしくは Keon内部の LDAP, HTTPの配布リポジトリを指定 一部、実験のために指定しない認証局も作成する。

ユーザ用証明書	X.509バージョン	V3
	署名アルゴリズム	sha1RSA
	公開鍵	RSA 1024bit
	DN	E = [メールアドレス] CN = [USERNAME]=[利用するメールサーバ名]([発行する CA名]) OU = (Giken-A ¦ Giken -B) #　信頼ポイントとなる CAのOUが入る O = CAC C = JP ※ 単一認証局における証明書は、CNは任意、OU以降は CAと同一
	基本制限	なし
	鍵使用方法	特に規定しないが、実験内容により digitalSignature, dataEncipherment を指定する
	CRL 配布ポイント	外部 LDAPもしくは Keon内部の LDAP, HTTPの配布リポジトリを指定 一部、実験のために指定しない認証局も作成する。