moCA 実験報告

櫻井 三子 [†] 服部 裕之 [‡]

[†]日本電気株式会社 ネットワーキング技術研究所

[‡]明治大学 情報科学センター

1997 年 7 月 12 日から 10 月 31 日の期間に行なった、WIDE moCA(members only CA) の運用実験および SSL のクライアント認証機能を用いた Web のアクセスコントロール実験の結果について報告する。

1. 実験の背景

インターネット上で通信相手が名乗った時にそれが本当に正しいかどうかを確認する技術 (認証技術) として、公開鍵暗号の公開鍵とユーザ識別子の対応を証明する CA(Certification Authority) が注目を浴びている。最近では、VeriSign 社をはじめとして CA を運用する専門会社が立ち上がったり、CA を自分で運用するための製品パッケージが出始めている。しかし、ユーザ識別子を具体的に何にするか、ユーザ識別子と公開鍵の対応を証明するのに何を確認すべきか、といった点については CA を運用する各組織で検討しなければならないことであり、運用に関するケーススタディが求められているといえる。

CA および CA が発行する証明書の存在を前提とするアプリケーションとしては、S/MIME などの暗号メールや、SSL(Secure Sockets Layer) などの Socket レベルでのセッションの暗号化や認証があげられる。前者は、数年前から実験や製品化が進められているが、フォーマット等の標準化がなかなか進まない点や既存の多くのメールツールへの対応が必要になる点などから、現状では比較的閉じた環境での利用に留まっている。後者は World Wide Web の普及にともなって急速に注目を浴びており、利用者の多い Netscape 社や Microsoft 社などの Web ブラウザに SSL のクライアント機能が標準で提供されるようになってきている。

WIDE では、メンバ間が情報共有するための手段の 1 つとして Web サーバを運用しているが、メンバ以外に見せたくない情報の保護方法はパスワード認証のみであり、さらなるセキュリティ強化について検討すべき時期にきている。昨年度 ICAT(認証実用化実験協議会) とタイアップした活動の中で CA 運用パッケージ ICAP を開発したこともあり、CA と SSL の組合せを使った Web のアクセスコントロール実験をするにはちょうどよい時期である。さらに、Web ブラウザはメールツールに比較すれば種類がまだ限られ、特定のブラウザを使った実験をする場合でも比較的多くの人の参加が期待できる。

CA ベースのシステムの需要が高まる中で、WIDE 全体で実験を行なうのに好都合な条件がちょうど揃ったことになる。そこで、ICAP を使って WIDE メンバのための CA を運用し、SSL 対応ブラウザ用の証明書を発行する実験と、SSL 対応ブラウザ・SSL 対応 Web サーバを使って、メンバのみで共有するページを証明書を持っているメンバのみが見えるようにアクセスコントロールする実験を行なうことにした。

2. 実験の目的

- CA 運用パッケージ ICAP を実際に使用

moCA WG, WIDE Project
m-sakura@ccs.mt.nec.co.jp
hhat@isc.meiji.ac.jp
Graphs are written by taiji-k@is.aist-nara.ac.jp

して実運用に耐えられるか評価すること
- 様々な組織に所属するメンバで構成されるコミュニティ内において、証明書を発行する上でのよりより手続きについて検討すること
- 実験をきっかけとして、証明書の今後の応用について検討すること

3. 実験内容

今回 moCA では、Web ページへのアクセスコントロールを、クライアント側 (Web ブラウザ側) から提示される証明書情報を利用して行なった。以下にその仕組みと、実現方法について紹介する。

3.1 アクセスコントロールのレベル

Web サーバへのアクセスコントロールを実現するには、

- クライアントコンピュータ単位でのアクセスコントロール
- ユーザ単位でのアクセスコントロール

の 2 つのレベルが考えられる。

クライアントコンピュータ単位でのアクセスコントロールとは、Web サーバへのアクセスを、クライアントの IP アドレスや DNS ドメイン名を元に制限する方法である。この場合、アクセスが許可されているクライアントコンピュータの使用者が誰であるかは問わない。

一方、ユーザ単位でのアクセスコントロールとは、Web サーバへのアクセスを、個々のユーザが持っている識別情報をもとに制限する方法である。この識別情報には、

- パスワード
- 証明書

などがある。

パスワードによるものは、ログイン名とパスワードのペアでユーザ認証を行なう方法であり、Web サーバ側では、事前にアクセス可能な者のログイン名とパスワードの組を登録する必要がある。

一方、証明書によるものは、ユーザの提示した暗号通信用の公開鍵とその所有者名との対応が、CA によって保証されていることを拠り所としている方法であり、Web サーバ側では、事前にアクセス可能な者の証明書情報を登録する必要がある。

3.1.1 実験の仕組み

証明書を利用したアクセスコントロールは、たとえば Web サーバおよび Web ブラウザ間の通信を SSL を用いることにより可能となる。

SSL は、実際のデータの暗号化に先立って、公開鍵暗号を用いてセッション鍵をサーバ・クライアント間で共有する。サーバおよびクライアントそれぞれの公開鍵情報は、X.509 証明書フォーマットで交換する。X.509 証明書は、図 1 に示すように、単に公開鍵ばかりでなく、その所有者、発行者、有効期限などの情報も含み、これは、発行者 (CA) によって内容が保証されている。

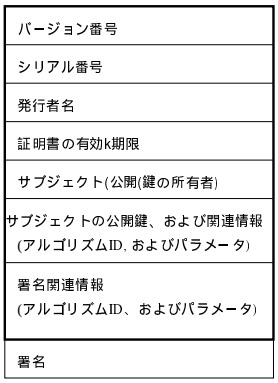


図 1: 証明書に含まれるデータ

つまり、SSL は単にサーバ・クライアント間の暗号通信機能を提供することによって盗聴・改竄の脅威からデータを守るばかりでなく、認証機能も提供することによってサーバ・クライアント双方へのなりすまし攻撃も防ぐことができる。

3.2 実験のステップ

以下の 2 つに分けた。

ステップ 1 ICAP を利用して CA を運用し、X.509 形式の証明書を発行する

ステップ 2 SSL クライアント認証機能を用いて、証明書つきの SSL 対応ブラウザから、アクセスコントロールをかけた Web ページにアクセスする

ステップ 1、ステップ 2 の実験構成図を図 2,3 に示す。

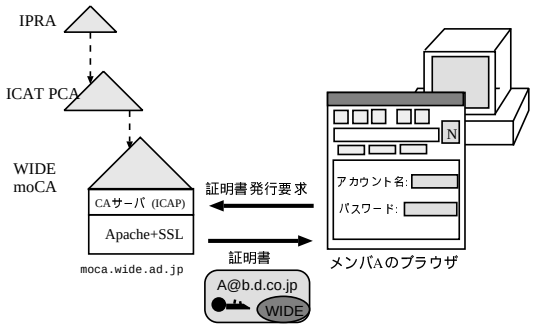


図 2: ステップ 1 の構成

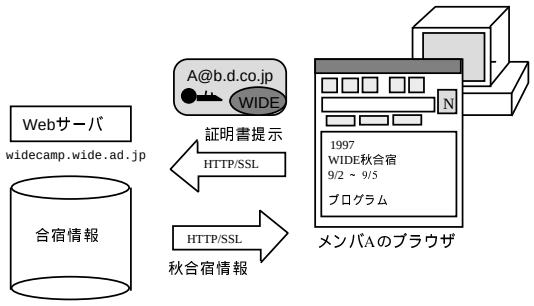


図 3: ステップ 2 の構成

3.3 実験対象

- 証明書の発行対象となるユーザ
  1997 年度 9 月の WIDE 合宿参加者
- アクセスコントロールの対象となるページ
  1997 年度 9 月の WIDE 合宿情報

3.4 実験期間

ステップ 1 1997 年 7 月 12 日 ~ 9 月 5 日
ステップ 2 1997 年 8 月 5 日 ~ 10 月 31 日

3.5 実験環境

- CA サーバ
  - OS ··· SunOS 4.1.4
  - CA プログラム ··· ICAP1.0
  - Web サーバプログラム ··· Apache1.2 +SSLpatch (SSL 対応のパッチ)
  - SSL プログラム ··· SSLeay-0.8.0
- Web サーバ
  - OS ··· BSD/OS 2.1
  - Web サーバプログラム ··· Apache1.2 +SSLpatch
  - SSL プログラム ··· SSLeay-0.6.6
- Web ブラウザ
  - OS は特に限定しない
  - Netscape Navigator または Netscape Communicator

3.5.1 ICAP について

ICAP1.0 は以下の特徴を持つ CA 運用パッケージである。

- CA の運用に必要な以下の機能をオンラインで提供
  - 発行
  - 廃棄
  - 検索
  - 有効性確認
- 発行する証明書の形式は X.509 version1
- Web インターフェイスによる簡単操作
- ユーザ登録により、ユーザが自分で証明書を発行することが可能
- オフラインによる発行・廃棄も可能

3.5.2 SSLeay について

SSLeay は以下の特徴を持つ SSL のパッケージである。

- オーストラリアの Eric Young 氏らが開発したフリーソフトウェア
- SSL version2 に対応
- SSL version3 に対応 (SSLeay 0.8.0 以降)

telnet, ftp, Apache などいくつかのアプリケーションでは SSLeay 対応のためのパッチが存在する。実験で使用した Apache にもこのパッチを適用した。

4. ステップ 1 に関する考察

ステップ 1 の処理の流れを図 4に示す。

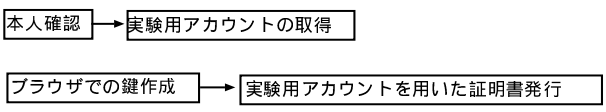


図 4: ステップ 1 の流れ

4.1 証明の対象となるユーザ識別子

公開鍵と対で証明されるユーザ識別子(サブジェクト)は、CA を運用する組織が確認できる情報であるとともに、CA ベースのアプリケーションで使うユーザ識別子となることが必要である。

WIDE メンバの識別子は、wide の ML 登録時の電子メールアドレスである。また、今回の Web ページのアクセスコントロールの実験では、個々の WIDE メンバのユーザ識別子を確認するというよりは、WIDE メンバであることを確認すればよいため、アプリケーション側からのユーザ識別子に対する要求は特にないと考えられる。そこで、証明の対象となるユーザ識別子を表 1のようにした。

表 1: ユーザ識別子

属性	値
Country	JP(固定値)
Organization	WIDE Project(固定値)
Organizational Unit	(自由に記入)
Common Name	wide ML に登録されている電子メールアドレス

このようなルールを持つユーザ識別子を含めた証明書は、ステップ 2 の実験中は問題にならなかったが、このままでは Netscape Communicator の S/MIME には利用できないことがわかった。S/MIME などの他のアプリケーションでも利用できるような証明書にするには、ユーザ識別子について再検討する必要がある。

4.2 本人確認の方法

証明書を発行する際に最も重要となるのは、証明書を発行してよいかどうかの確認方法である。具体的には、証明書の発行要求者と証明の対象となるユーザ識別子の対応が正しいことを確認することである。このような手続きを行なう機能のことを、一般に RA(Registration Authority) と呼ぶことがある。RA を「誰が」「いつ」「どのような方法で」行なうかを決める必要がある。

WIDE ではメンバの電子メールアドレスを wide の ML に登録する際に WIDE プロジェクト運営機関であるボードメンバが承認する形となっている。そして、メンバには必ず担当ボードが一人つく。実験ではこのしくみを利用して、メンバの担当ボードがメンバ本人と電子メールアドレスを確認し、安全な方法で CA の運用者 (CA オペレータ) に伝える、というおおまかな流れを決めた。つまり、RA は担当ボードが行なうことにした。

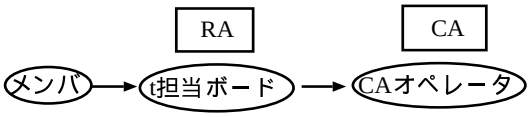


図 5: 本人確認の流れ

本人確認は、メンバ、担当ボード、CA オペレータが顔を合わせる機会に行なうのが最適である。しかし、WIDE では定例研究会以外で顔を合わせる機会がないため、「いつ」「どのように」本人確認を行なうかについては、以下の 2 通りの方法を試した。

(1) 7 月の定例研究会で顔を合わせたとき
メンバが所定の用紙 (付録 A) に氏名、電子メールアドレスなどを記入し、担当ボードがメンバの身元と用紙の記入事項を確認して、CA オペレータの目の前で用紙に署名をする。

(2) 9 月合宿前日までの随時
メンバが担当ボードにメールアドレスなどの個人情報を何らかの方法で渡し、担当ボードがメンバを何らかの方法で確認し、担当ボードから CA オペレータには PGP を利用してオンラインで情報を渡す。[1]

実際に実験を行なったところ、(1) は多かったが、担当ボード本人が立ち会うことについては現実にはうまくいかず、担当ボード以外のボードが代理で署名することになった。(2) はメンバがボードに個人情報を渡す

[1] (2) を実施する前には、あらかじめ CA オペレータがボードの身元確認を行ない、各ボードの PGP の公開鍵が正しいことを確認した。

方法を決めなかったことから、逆にメンバを戸惑わせることになり、予想以上に数が少ない結果となった。そこで、合宿当日には臨時に以下の方法で本人確認をおこなうことにした。

(3) 9月合宿当日顔を合わせたとき
メンバが所定の用紙(付録A)に氏名、電子メールアドレスなどを記入し、CAオペレータが身元と用紙の記入事項を確認する。

方法別の集計を表2に示す。担当ボードがメンバ本人を確認する具体的な方法については特に決めなかったが、(1),(3)では写真つきで所属のわかる証明書類をできるだけ提示してもらった。具体的な身元確認方法について表3にまとめる。

表2: WIDEメンバ数(411)に占める実験アカウント取得の割合

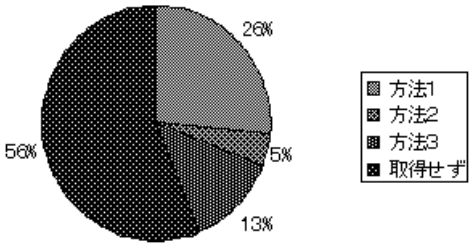


表3: オフラインでの身元確認方法

学生証、社員証、身分証明書	104(人)
(各種)免許証 + 名刺	33
パスポート + 名刺	2
クレジットカード + 名刺 + 写真	2
健康保険証 + 名刺 + 写真	0
ボードが保証	10
その他	
名刺のみ	1
免許証のみ	1
クレジットカード + 名刺	1
CAオペレータが保証	1
記入洩れ	6

4.3 証明書の発行方法

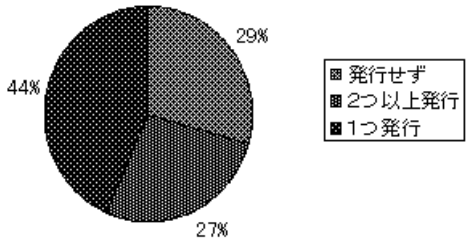
一般の手続きでは、本人確認後にすぐに証明書を発行するのが典型的であるが、以下の実験環境を考慮して、本人確認と同時に実験用アカウントを割り当て、証明書発行作業は後日そのアカウントを使って各自が行なう方式とした。

- Netscapeブラウザ用の証明書を発行するにあたってはブラウザ側で鍵を作成する必要があるが、本人確認するその場でブラウザを持っているとは限らない
- ICAPにはアカウント登録機能があり、登録されたユーザがパスワードを使って自分で証明書を発行することが可能である

具体的には実験用アカウント名は各自の電子メールアドレスとし、パスワードについては節4.2の(1)では用紙に印刷して配布、(2)ではメンバ自身が申告して決めるようにした。

実験の結果、実際に証明書を発行したユーザ数を表4に示す。これを見ると実験用アカウントを取得したものの、証明書を発行しなかったユーザの割合が多いことがわかる。

表4: 実験用アカウント総数(180)に占める証明書発行者数の割合



証明書を発行するときには、ブラウザで鍵を作成してから証明書を発行するが、この処理の間には秘密鍵保護のためのパスワード要求と、実験用アカウントに対するパスワード要求の2種類があった。この2種類を使い分けることについての説明が不十分だったことから、ユーザに混乱を与えることになった。また、順番的にも、実験用アカウントに対す

るパスワードを確認してから、鍵を作成するようにするべきであり、ICAP 側で対応すべき問題といえる。

4.4 証明書の発行個数

実験に際して、メンバの利用するブラウザは一つとは限らないことが十分予想された。対策としては、Netscape ブラウザの環境設定ファイルをコピーする方法があるが、今回の実験では一人あたり 5 個まで証明書を発行できるように ICAP の設定を行なった。

実際に証明書を発行したユーザの 4 割程度が証明書を 2 個以上発行している。アンケート結果(付録Bと照らし合わせると、複数のブラウザへの対応のためだけでなく、以下のような鍵管理上の問題が発生していることがわかった。

- ブラウザや OS をバージョンアップした時に秘密鍵と証明書が消えたから
- ブラウザの秘密鍵保護機能のパスフレーズを忘れたから

SSL では、秘密鍵を盗まれるなどの深刻な理由でない限り、証明書を発行しなおしてもその後のセッションから新しい証明書を差し支えなく使うことができる。結果的には、1 人あたり複数の証明書を発行できるようにしたことで、証明書の再発行の手続き上のコストを下げることになった。

しかし、S/MIME などの暗号メールでは、1 人で複数の証明書を持っていると通信相手がそのうちのどの証明書を使えばよいかがわからなくなるなどの影響が考えられ、証明書の発行しなおしをできるだけ避けられるような対策を考える必要がある。

4.5 証明書フォーマット

一般によく利用されている証明書のフォーマットは X.509 version3 である。ICAP 1.0 では、X.509 version1 の証明書発行機能のみをサポートしているため、フォーマットについては CA の証明書、メンバの証明書どちらについても X.509 version1 で発行した。

今回の実験では、Netscape ブラウザや Web サーバ双方で X.509 version1 の証明書を扱うことができ支障はなかったが、Microsoft 社の Web ブラウザである Internet Explorer や、Netscape Communicator の S/MIME で利用するには X.509 version3 とすることが必須である。

X.509 version3 では X.509 version1 の形式(図 1)に加えて、拡張フィールドが定義されている。X.509 version3 の証明書に関しては、アプリケーションごとに採用する拡張フィールドが異なる傾向があり、アプリケーションごとに発行する証明書フォーマットを変えるか、いくつかのアプリケーションで共通に使えるようにするかを検討しなければならない。

4.6 証明書の有効期限

一般には、エンドユーザの証明書の有効期限は以下の要素が関係して決まる。

- 証明する公開鍵の長さ、アルゴリズム
- アプリケーション、サービスの性質
- CA の証明書の有効期限

特に CA を長期運用する場合には十分に検討して有効期限を決めないと、予想外に証明書の再発行が必要になったり、予想外に CRL(証明書の取消リスト) のサイズが大きくなるなど運用上深刻な問題が発生する可能性がある。

今回は実験終了日に実験を終了するために、メンバの証明書の有効期限を 1997 年 10 月 31 日の固定値にした。今後は、継続的なサービスを前提とした実験を行なう必要がある。

4.7 CA の位置づけ

CA は階層構造を取ることが可能である。証明書の発行を受けるエンドユーザの数が非常に多い場合や、ポリシが異なる場合に分散して管理することが可能で、大規模なシステムに適しているといわれている。

WIDE メンバ数は 100 のオーダであり規模は比較的小さく、ポリシを分ける必要性が特にないことから、CA を 1 つだけ運用することにした。CA は自己署名で単独とするか、既存の階層 CA の証明を受けて階層構造に位置づけるかの選択肢があるが、以下の理由から ISOC が運営する IPRA (Internet Policy Registration Authority) の下の ICAT PCA(Policy Certification Authority) に位置づ

けた。

- 共通の上位 CA を持つ場合でも、他の CA が発行した証明書では合宿情報にアクセスできないことを確認するため
- WIDE moCA のなりすましを防ぐため ICAP ではオンラインで証明書発行を受け付けるため、オフラインで管理するのと比較した場合に CA の秘密鍵が盗まれる危険性が高くなる。万が一盗まれても簡単には CA のなりすましができないようにするため第三者による証明を受けた方がよい。

実験では、証明書の署名確認時に CA の証明書が必要なことから、Web サーバ、Web ブラウザに IPRA, ICAT PCA, WIDE moCA の 3 つの証明書を最初に登録することになった。

また、実験中の 9/15 に IPRA の証明書の有効期限が切れて無効となったため、証明書の更新が行なわれるまでの 10 日間ステップ 2 の実験が中断した (図 6)。実験を再開するには、Web サーバ、Web ブラウザ各々が IPRA の証明書を再登録する必要があった。このような作業は CA を長期運用すれば起こり得ることであるが、異なる組織が運用する CA で構成される階層構造では特に、オペレーションがとまらないようにするために連絡を密にとる必要がある。

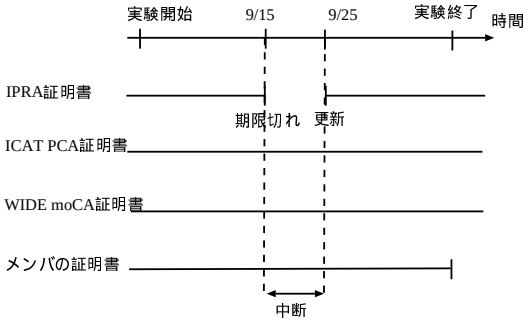


図 6: IPRA の有効期限ぎれによる実験中断

4.8 証明書の廃棄と CRL

証明書を廃棄する際にも証明書を新規発行する際と同様に、本人確認が必要となる。また、証明書を廃棄したら、そのリストを CRL として定期的にまとめて発行する必要がある。CRL(Certificate Revocation List) の発行間隔は、CA のポリシやアプリケーションの性質によって決まる。

実験では、証明書の廃棄手続きについては特に決めなかった。証明書の廃棄は何件か行なったが、CA オペレータの操作ミスがほとんどであり、秘密鍵を盗まれるというような深刻な理由による証明書の廃棄要求はなかった。

CRL の発行については、1 週間単位で発行し Web サーバに配布する実験を検討したが一部の実験にとどまり、継続的な実験はできなかった。

4.9 証明書の検索

通信相手の証明書をあらかじめ入手しておく必要がある場合、証明書の検索が必要である。CA パッケージの専用の検索機能を利用する他、既存のデータベースを活用するなどの方法がある。

SSL では、通信相手の証明書をあらかじめ入手しておく必要がないため、ICAP の証明書検索機能、既存のデータベースとも利用しなかった。

4.10 Web ブラウザ

ブラウザについては既存のツールをそのまま利用した。

実験中、BSD/OS 版、NEWS-OS 6.1.2 版の Netscape Navigator 3.01 では鍵作成時にストールする場合のあることが報告された。ソースコードがないこともあり明確な原因は究明していないが、Navigator の文字コードセットの切替え、ICAP の表示切替えによって現象が変わることから、日本語コードが影響しているようである。

このように、ブラウザのバージョンが同じでも、OS によっては動作が不安定であることがわかった。

4.11 ファイアウォールとの関係

ステップ 1 の証明書発行の際には、実験用アカウントとパスワードを入力するため、パスワードを保護するために SSL の暗号機能のみを利用した。

WIDE に限らず、ほとんどの組織ではファイアウォールがあっても HTTP を通すような

対策が取られているが、SSL のセッションについてはポリシの観点から通さない設定になっている場合もある。

ステップ 1 では、SSL の標準ポート (443 番) のみを使ったため、SSL プロキシ機能が既に設定されているところでは設定の変更なく実験に参加可能であった。ポリシとして SSL のセッションを通さない組織では、ステップ 1 の時点から実験に参加できないということになった。実験参加数が半分に満たなかった背景にはファイアウォールとの関連があると思われる。

4.12 ICAP の評価

実験の仕様に合わせて、ICAP1.0 にはおもに以下の変更を行なった。

- ユーザの証明書の氏名欄の値を CA オペレータが指定できるようにする
  電子メールアドレス以外を記入できないようにするためと、記入時のミスを防ぐためである
- ユーザ認証に OTP 方式をオプショナルに導入
  実際にこの方式を使用したユーザはいなかった
- ユーザインターフェイス上の諸変更

ICAP は全てのソースを公開しているわけではないが、上記の変更を行なうのは公開されているソースだけで対応が可能であった。このようにカスタマイズして使える点は ICAP の利点といえる。

結果として、100 人規模の CA を運用することができたが、定常運用を考えると以下のような改善が必要である。

- CA の秘密鍵のセキュリティ管理
  タンパフリーハードウェアとの組合せなど
- 設定機能の充実化
  証明書の有効期限を設定ファイルで簡単に変えられるようにするなど
- ログ機能の強化
  証明書発行要求に失敗したときのログを取るようにするなど
- ユーザインターフェイスの改善
  ブラウザの鍵作成画面が、ICAP のユーザ認証が終わった後に出るようにするなど
- X.509 version3 への対応

5. ステップ 2 に関する考察

ステップ 2 の処理の流れを図 7に示す。

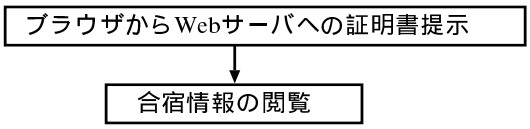


図 7: ステップ 2 の流れ

5.1 Web サーバ

5.1.1 アクセスコントロールの実現

moCA では、証明書を用いたアクセスコントロールを実現するために、Apache+SSL を Web サーバとして利用した。Apache+SSL を用いたアクセスコントロールの方法を以下に述べる。

i) アクセスコントロールを行ないたいドキュメントを置いたディレクトリに、.htaccess ファイル (図 8) を用意する。

ii) パスワードファイルにアクセスを許可する者の DN(Distinguished Name) を列挙する。書式は、

ユーザ DN:xxj31ZMTZzkVA

である。暗号化されたパスワード部分には、常に xxj31ZMTZzkVA を指定する。(図 9)

iii) 完了。
このディレクトリの Web ページへのアクセスに対しては、Web ブラウザから証明書を提示し、さらに、証明書のサブジェクトが、.htpasswd に記載されている DN と一致した場合にアクセスが可能となる。

5.1.2 Apache+SSL の問題点

moCA の目的の一つには、証明書を活用した Web ページのアクセスコントロールの実用性を確認することがあったのだが、Apache+SSL には次の問題があった。

（例）.htaccess ファイル

```
AuthUserFile /usr/local/etc/httpd/conf/.htpasswd
AuthName ByPassword
AuthType Basic
<Limit POST GET PUT>
require valid-user
</Limit>
```

これは、クライアントから Web サーバに提示された証明書の所有者名（サブジェクト）を
/usr/local/etc/httpd/conf/.htpasswd ファイル
と比較し、アクセスの許可不許可を決定するという意味である。

図 8: .htaccess の設定例

（例）/usr/local/etc/httpd/conf/.htpasswd ファイル

```
/C=JP/O=Meiji University/OU=Information Science Center/CN=hhat:xxj31ZMTZzkVA
```

これは、クライアントから提示された証明書の所有者名（サブジェクト）が、

```
/C=JP/O=Meiji University/OU=Information Science Center/CN=hhat
```

であるならばアクセスを許可するという意味である。

図 9: .htpasswd の例

i) パスワードファイルを記述する際の柔軟性の問題
Apache+SSL は、ある Web ページへのアクセスを特定のユーザにのみ許可したい場合、パスワードファイル（.htpasswd 等）に全てのユーザの DN を列挙する必要があった。つまり、例えば 200 名のメンバーに対するアクセス許可を行ないたい場合は、200 名分の DN 情報を記載する必要があった。
この問題は、パスワードファイルの DN に正規表現を利用できるよう、プログラムの改良を行なうことによって解決した。

ii) パスワードファイルにサブジェクトしか記述できない問題
Apache+SSL のパスワードファイルは、ユーザの DN、つまり証明書のサブジェクトしか記述できず、その証明書がどこの CA で発行されたものかを記述することはできない。
通常、A 社の CA が発行した証明書は、そのサブジェクトに、証明書の所有者が A 社の者であることを示す DN を選択する。しかし、CA が発行した証明書のサブジェクトの内容については、基本的にその CA のポリシーで縛るしかない。つまり、理論上は A 社の CA が、B 社の者のサブジェクトと見間違えるような証明

書を発行することは可能である。
つまり Apache+SSL は、B 社の社員へのアクセスを許可するつもりで、パスワードファイルに、B 社の者の DN を記載したところ、発行者が A 社でサブジェクトが B 社となっている証明書のユーザに対しても、アクセスを許可してしまう恐れがある。
この問題は、Web サーバ側で、信頼する CA が発行した証明書しか受け付けないようにすることによって、一応の回避はできる。しかし理想的には、証明書のサブジェクトおよび発行者の組で、パスワードファイルは記述できるようにした方が望ましい。

iii) アクセスコントロールのポリシーが、Web ページ毎に設定できない
Apache+SSL では、アクセスコントロールのポリシーは「個人証明書は必要ない」「個人証明書を持っていないとだめ」という 2 つのタイプしか設定できない。
さらにこのポリシーは、Web ページ毎、ディレクトリ毎に変更することはできず、サーバ単位となっている。
つまり、一つのサーバ上で、複数のセキュリティポリシーを持つ Web ページを提供できない為、サーバの運用形態によっては問題となる。
現在のところ、この問題は未解決である。

iv) クライアント側から提示される証明書の有効性チェックが不完全
秘密鍵の漏洩など、セキュリティ的な事故が発生した時は、証明書をすみやかに無効にする必要がある。通常、その証明書の発行局は CRL を発行する。
しかし、Apache+SSL は、CRL のチェックを行なわず、クライアントから提示された証明書の有効性については、署名や有効期限くらいしかチェックしていない。
この問題は、CRL のチェックを行なうように、プログラムの改良を行なうことによって解決した。ただし、今回は試作のみで運用までには至っていない。

5.2 Web ブラウザ

メンバによっては WIDE moCA が発行した証明書以外に複数の証明書を持っている。Netscape Navigator には、SSL のセッション中 Web サーバに提示する証明書を複数の中から自動的に選択する機能が設けられていたが、その機能を選択するとブラウザが強制終了するという不具合があった。WIDE moCA が発行した証明書が原因かどうかは不明である。

5.3 ファイアウォールとの関係

ステップ 2 では、SSL のポートとして標準以外のポートも用意した。
ステップ 1 の場合と異なり、SSL プロキシの設定をおこなっているところでも、標準ポート以外ではアクセスできない場合が多かった。例えば、以下のような場合には標準ポート以外でもアクセスできることがわかった。

- SOCKS をプロキシとして利用している場合
- HTTP プロキシとして squid を利用している場合
  内部から外部への直接通信が可能な環境でメインの squid の デフォルト値、
  `http_access deny CONNECT !SSL_ports`
  の指定をはずす。

6. ユーザの観点からの考察

ステップ 1 の実験が終了する 1997 年 9 月 5 日 に合宿参加者に対して無記名によるアンケートを行なった。結果を付録 Bに示す。

6.1 実験手続きについて

まず、実験のガイドページの説明がわかりにくかった、と答えているメンバが多かった。実験ガイドのページには、テキストだけでなく、手順ごとにブラウザの操作イメージなどを入れるべきであった。今回の実験の反省点の 1 つである。特に、節 4.3で述べたようにブラウザの秘密鍵を保護するためのパスワードと ICAP に登録した実験用アカウントのパスワードの使い分けが最も混乱を招いた

点である。ガイドや ICAP を改善して意味を理解してもらい、2 つのパスワードを使い分けられるようにするのか、実験用アカウントを使って各自で証明書を発行するという方法自体を見直した方がよいのか、検討する必要がある。

6.2 Web のアクセスコントロールについて

今後証明書を使って Web のアクセスコントロールを行なうことについては、肯定的な意見が多かった。今回の試みが、証明書を使ったアプリケーションの 1 つとして将来ユーザに受け入れられる可能性が示されている。

アクセスコントロールのかかったページへのアクセス数について表 5 ~ 8に示す。アクセスは合宿終了までの期間に集中していることがわかる。

今回の実験では、証明書を持っているユーザは誰でもアクセスが可能、という設定が中心であったが、これだけではパスワード認証による方法との違いが実感できない。テストページ 3 ではメンバの中でさらに moCA WG のメンバのみがアクセスできるページを設けたがアクセス数は少なかった。

表 5: 合宿情報ページへのアクセス数 (概算)

a) 実験期間中のアクセス総数	2500
b) 合宿期間中までのアクセス総数	不明
c) a) のうちアクセスに成功した総数	2000
d) a) のうちアクセスに失敗した総数	500

表 6: テストページ 1(明治大) へのアクセス数

a) 実験期間中のアクセス総数	183
b) 合宿期間中までのアクセス総数	159
c) a) のうちアクセスに成功した総数	135
d) a) のうちアクセスに失敗した総数	48

表 7: テストページ 2 へのアクセス数

a) 実験期間中のアクセス総数	43
b) 合宿期間中までのアクセス総数	32
c) a) のうちアクセスに成功した総数	43
d) a) のうちアクセスに失敗した総数	0

注)SSL ポート 8000 番

表 8: テストページ 3 へのアクセス数

a) 実験期間中のアクセス総数	56
b) 合宿期間中までのアクセス総数	48
c) a) のうちアクセスに成功した総数	12
d) a) のうちアクセスに失敗した総数	44

注)SSL ポート 8000 番、 moCA WG メンバのみアクセス可

7. 実験の進め方に関する課題

7.1 他のシステムとの連携

当初の予定では、合宿情報だけでなく、合宿 PC によって新しく開発された Web インターフェイスの合宿申込システムに対しても、証明書を使ってアクセスできるようにしようとしていた。しかし、合宿申込システムの Web サーバを SSL 対応にして動作確認するところまで至らなかった。今後は、他のシステムで同じ証明書を使えるように適用範囲を広げたい。

また、証明書発行に手間のかかる分、証明書発行時に確認した個人情報を合宿申込や報告書に載せる名簿作成に利用できるようにするなど、 WIDE の他の活動とのリンクについても検討したい。

7.2 アクセスコントロールの対象となる情報

今回の実験では、合宿情報全てにアクセスコントロールを行なったが、メンバ以外には見せたくない情報であっても、それが読めないとメンバ自身が本当に困るのかどうか確信できない情報もある。

合宿情報、あるいはそれ以外の情報を含めて、それがないとメンバが困る情報がどの程度あるのかを調べる必要がある。環境の問題等で実験に参加できないメンバの対応を考え

つつ、実験への参加意識をより高められるような情報に対してアクセスコントロールをかけて、定常運用の可能性を探りたい。

謝辞

実験にご協力下さった方々に深く感謝いたします。

A　実験アカウント申請用紙 (例)

WIDE moCA 登録申請用紙

1997 年　月　日

* 印のついた項目のみ記入し、担当ボードと一緒に CA オペレータのところまで来て下さい。

*　　氏名:

*　　所属:

*　　e-mail:
(wide@wide に登録されている e-mail アドレスを記入して下さい。)

パスワード:　　7-=N@XLq

身元確認方法 (確認時 CA オペレータが記入):

- □ 学生証、社員証、身分証明書
- □ (各種) 免許証 + 名刺
- □ パスポート + 名刺
- □ クレジットカード + 名刺 + 写真
- □ 健康保険証 + 名刺 + 写真
- □ ボードが保証
- □ その他 _____

担当ボード署名: _____

---------- 切 り 取 り 線 ----------

1997 年　月　日

WIDE moCA 登録申請 (控)

証明書発行時に利用するアカウント名 (e-mail アドレス) とパスワードは以下になります。大切に保管して下さい。

*　　e-mail:
(wide@wide に登録されている e-mail アドレスを記入して下さい。)

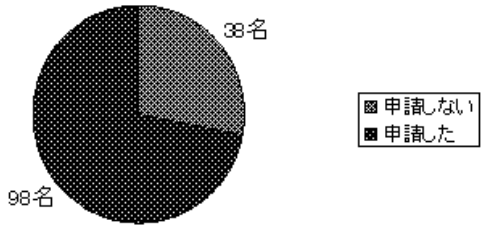
パスワード:　　7-=N@XLq

※登録作業終了の通知後、以下の URL へアクセスして下さい。
<URL:http://moca.wide.ad.jp/index.html>
※証明書は、1 人あたり 5 個まで発行できます。

B　アンケート結果

回答数 136

B1　moCA のアカウント申請について

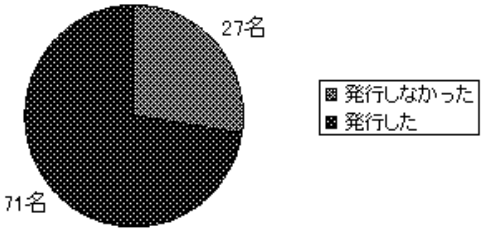


B2　moCA アカウントを申請しなかった理由

時間がない	20
方法がわからなかった	7
担当 board にお願いしたが連絡がとれなかった	3
(その他)	
必要性を感じなかった	2
(合宿当日)PC を持って来なかった	2
メンバではなく、発行不可のようだった	1
担当 board にお願いしそびれた	1
間に合わなかった	1

注) 合宿参加者は誰でも申請できた

B3　moCA の証明書発行について



B4　moCA の証明書を発行しなかった理由

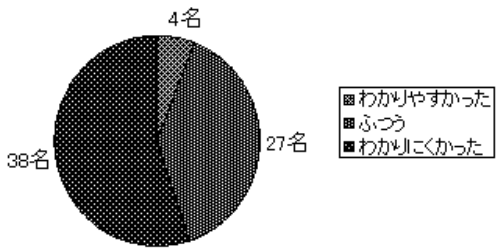
時間がない	13
ファイアウォールの環境で参加できなかった	2
Netscape のブラウザを使えない (使いたくない)	7
実際に試したがうまくいかなかった	4
その他	
(合宿当日)PC を持って来なかった	1

B5　moCA の証明書を発行した時の環境

OS	Netscape ブラウザ名	ブラウザのバージョン
WindowsNT 4.0	Navigator	3.0, 3.02
	Communicator	4,01, 4.02
Windows95	Navigator	3.0, 3.01
	Communicator	4.01, 4.02
SunOS 4.1.x	Navigator	3.0, 3.01
Solaris	Navigator	3.01, 3.02
BSD/OS 2.x	Navigator	3.01
BSD/OS 3.0	Navigator	3.0, 3.01
Linux	Navigator	3.01, 3.02
FreeBSD 2.x	Navigator	3.0, 3.01
FreeBSD 2.2.x	Communicator	4.01x
NetBSD-1.2G	Navigator	3.01
OSF/1	Navigator	3.01
IRIX	Navigator	3.01
NEWS-OS 6.1.2	Navigator	3.01
AIX	Communicator	4.01

注)Windows95 で Internet Explorer を使用したという報告があったが、今回の実験では対応させていなかったため、証明書発行はうまくいかなかったはずである。

B6　moCA の証明書を発行するときのガイドページについて



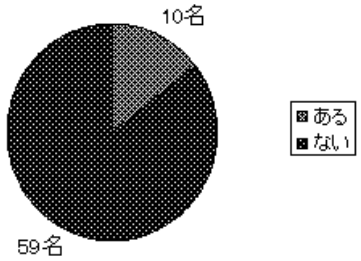
B7　証明書ベースの Web アクセスコントロールについて

a) 今後は、WIDE 内の他の Web 情報も全部個人証明書を使ってアクセスできるようにした方がよい	11
b)WIDE 内の sensitive な情報をよく吟味した上で、一部は個人証明書を使ってアクセスできるようにした方がよい	49
c)WIDE 内の情報の保護には、従来のパスワードの方式で十分	4
d) その他	
a) と b) 両方	1
b) がいいが a) で妥協	1
b) がいいが SSL 対応でないブラウザへの配慮も必要	1
(証明書ベースは) 仕組みがわからずハードルが高い	1
もっとユーザの立場を考えた仕組みを考えるべき	1

B8　moCA の証明書を 2 個以上を発行した理由

複数のホスト毎に必要になったから	23
ブラウザのバージョンアップなどで、設定ファイルがおかしくなったり、消えたりしたから	6
ブラウザの鍵を人に知られたと思われたから	0
(その他)	
最初は使い方がわからなかったから	1
Netscape パスワードを忘れたから	1
パスワードを忘れたから	1
いつのまにか	1
試してみたかったのと、明治大へのテストがうまくいかなかった状況を解決するため	1

B9　moCA 以外の証明書取得について



ある場合の具体例

CA 名	アプリケーション名
ISIT	
ICAT Pilot CA	Netscape Navigator 3.01
VeriSign 社	お試し ID
VeriSign 社	Netscape Navigator
VeriSign 社	Netscape Communicator
VeriSign 社	Netscape commerce server
WIDE CA	FJPEM
Xcert 社	サポート cert
Xcert 社	Netscape Navigator 3.01

B10　その他、気づいた点

仕組みや実験手続きの説明が不十分という指摘、実験のアプローチや進め方についての意見が合計 5 件、ブラウザが不安定といった指摘が 4 件あった。

C　利用したパッケージの入手先一覧

ICAP	<URL:http://www.icat.or.jp/>
SSLeay	<URL:http://www.psy.uq.oz.au/ ftp/Crypto/ssleay>
Apache	<URL:http://www.apache.org/>
Apache の SSL パッチ	<URL:ftp://ftp.ox.ac.uk/pub/crypto/SSL>
実験用の Apache へのパッチ	
パッチ	<URL:http://www.isc.meiji.ac.jp/ hhat/report/client_regex_patch.txt>
ドキュメント	<URL:http://www.isc.meiji.ac.jp/ hhat/report/client_auth.txt>