システムソフトウェアと
オペレーティング・システム 77−40
マルチメディア通信と分散処理 87−40
（1998. 2. 27）

スケーラビリティを考慮した インターネット環境における個人認証システムの開発

濱口 伸　　岡山 聖彦　　山口 英
shin-h@is.aist-nara.ac.jp　*okayama@is.aist-nara.ac.jp*　*suguru@is.aist-nara.ac.jp*
奈良先端科学技術大学院大学情報科学研究科
尾家 祐二
oie@cse.kyutech.ac.jp
九州工業大学情報工学部

本研究は、公開鍵暗号方式に基づいた個人認証システム SPLICE/AS-II のコミュニティモデルを拡張して認証局を導入することにより、認証に必要な公開鍵をコミュニティ間で配送する仕組みを提案する。従来の単一コミュニティ内における鍵配送に加え、他コミュニティとの鍵配送が可能となったことにより、インターネットのような広域かつ大規模ネットワークに対してより柔軟な鍵配送機構を実現することができる。

本稿では、提案したコミュニティ間通信モデルに基づいた個人認証システムを設計し、実際のネットワーク環境で運用実験を行なうことによってシステムの有効性を評価した。

Design and Implementation of an Authentication System with the scalable key-exchange scheme for the Internet

Shin Hamaguchi　　Kiyohiko Okayama　　Suguru Yamaguchi
shin-h@is.aist-nara.ac.jp　*okayama@is.aist-nara.ac.jp*　*suguru@is.aist-nara.ac.jp*
Graduate School of Information Science, Nara Institute of Science and Technology.
Yuji Oie
oie@cse.kyutech.ac.jp
Dept.of Computer Science and Electronics, Kyushu Institute of Technology

We propose the authentication key distribution mechanism with expanding the community model of SPLICE/AS-II which is authentication system based on public-key algorithm. Using certificates issued by the certification authority, we provide the inter-community key exchange scheme.

In this paper, we describe the new authentication system based on both SPLICE/AS-II and the certification authority, and confirm its validity of this system through the experiment.

1 はじめに

近年のインターネットの普及はめざましく、多くの人々が日常的にインターネットを利用するようになってきた。インターネットに接続している組織が増えるにつれ、その利用形態も当初の研究利用中心から、特定ユーザ向けのサービスや電子商取引などの商用利用中心に移行してきている。

しかし、インターネットで利用されている通信プロトコル (TCP/IP) は、身分詐称を防ぐユーザ認証サービス、伝送路での盗聴・改ざんを防止するサービスなど、安全性に関して考慮されてい。しかし、今日のインターネットの利用形態の変化により、これらの技術に対する要求が高まっている。

このため、我々の研究グループではネットワーク上でユーザ認証を行なうシステムである SPLICE/AS-II の開発を行なっている。SPLICE/AS-II は公開鍵暗号方式に基づいており、ユーザ間で認証を行なう機能と、認証に必要となる公開鍵を配送する仕組みを提供している。SPLICE/AS-II では、あらかじめ

決められたユーザの集団と、その集団に対してサービスを提供するサーバプロセスの集団をコミュニティと呼ぶ。コミュニティはサーバ、クライアント、鍵配布サーバ (Key Distribution Server、以下 KDS と略す) の 3 種類の要素から構成される。同一コミュニティに所属するクライアントとサーバが KDS を通じて相手の公開鍵を取得することにより、認証・暗号化通信を行なう。

しかし、SPLICE/AS-II は単一コミュニティにおける認証システムとして設計されており、複数コミュニティが存在したときに、それらの間で鍵配送を行なうことは考慮されていない。しかし、SPLICE/AS-II をインターネット環境で運用することを考えた時、鍵配送のルーティングテーブルは管理者の手作業で維持しているため単一コミュニティでは運用できる範囲は限定され、スケーラビリティに問題がある。さらに、メンバとして鍵交換を行なう際にはコミュニティに所属しなければならないが、インターネット環境で運用する場合、単一のコミュニティという一定のポリシーを持った管理機構に全てのメンバを所属させることは不可能である。また、クライアント・サーバ間の暗号化通信機能をアプリケーション層で実現しているため、暗号化通信機能を提供するためにアプリケーションのソースコードを変更する必要があった。

これらの問題点を解決するため、本研究では複数存在するコミュニティ間での公開鍵配送を実現し、異なるコミュニティに属するユーザ間で認証および暗号化通信を可能とするシステムを提案する。コミュニティは多数存在するため、コミュニティ管理者は事前に公開鍵交換を行ないたいコミュニティをすべて把握することは不可能である。そこで、コミュニティの身分を保証する第 3 者による認証機関として認証局 (Certification Authority、以下 CA)[2] を導入し、CA を介して未知のコミュニティとの鍵交換を行なう仕組みを提案する。一方、現実世界を考えたとき、不特定多数のコミュニティと無条件に鍵交換を行なうことは避けたいという要求がある。そこで、本システムでは他コミュニティに対する鍵交換ポリシーを定義することにより、他コミュニティとの鍵交換を制御する。

さらに、我々の研究グループが開発している暗号化通信機構 Secure-TCP[3] との統合を行なう。Secure-TCP は TCP 層での暗号化通信機能を提供するもので、その上位層であるアプリケーション層のプログラムは暗号化機能を特別に意識することなく用いることができる。Secure-TCP を導入することにより、既存のアプリケーションに対するソースコードの変更を最小限に抑えることができる。また、本システムが Secure-TCP に暗号化通信で利用するセッション鍵を与えることで、Secure-TCP におけるセッション鍵の生成および共有という問題を解決することができる。

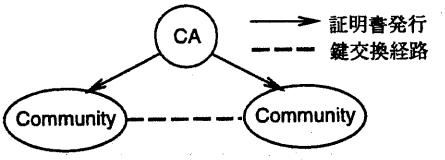

図 1: CA の発行する証明書を用いた接続の例

本稿では、新たに提案したシステムの概要について述べた後、システムの動作実験とその評価について述べる。

2 コミュニティ間通信モデル

本研究では、コミュニティ間での鍵配送を実現するために、SPLICE/AS-II のモデルを拡張し、スケーラビリティをより重視したインターネット環境での認証モデルを提案する。

コミュニティ間で公開鍵配送を行うためには、他コミュニティの鍵配送機構を信用しなければならない。あるコミュニティがインターネット上に独立して存在する全てのコミュニティを把握することは不可能であるため、本システムでは新たに CA を導入し、CA がコミュニティに対して発行する証明書を用いて他コミュニティに対する認証を行ない、他コミュニティが信用できるかどうかを判断する。CA はコミュニティの身分を審査し、信頼できるものであればコミュニティの身分を CA の名前で保証する。他コミュニティと鍵交換を行う際は、相手コミュニティの証明書を検査し、有効であれば公開鍵配送を行う (図 1参照)。

また、コミュニティ内の鍵配送機構の構造を他コミュニティから隠蔽するため、本システムでは外部に存在を公開する特別な KDS を用意する。本システムでは、この KDS を**代表鍵配布サーバ** (以下**代表 KDS**) と呼ぶ。

図 2に本研究が提案するコミュニティ間通信モデルを示す。代表 KDS は、コミュニティ内にリージョンから独立してひとつ存在する。その役割は、他コミュニティと通信を行う際の鍵交換ポリシーに基づいた鍵配送と他コミュニティの認証である。

代表 KSD 間で鍵交換を行う際にも、KDS 間での鍵配送と同様に相手代表 KDS の公開鍵が必要であるが、それは CA から発行された証明書に記載されている。代表 KDS は鍵交換を行いたいコミュニティの代表 KDS の公開鍵を事前に入手する必要はないため、未知のコミュニティとの間で鍵交換を行なうことが可能となる。図 3に示すように、CA は階層構造を持っている。上位の CA は下位の CA に対して証明書を発行する。階層構造の頂点には全ての CA の証明の基になっている **Root CA** が存在する。こ

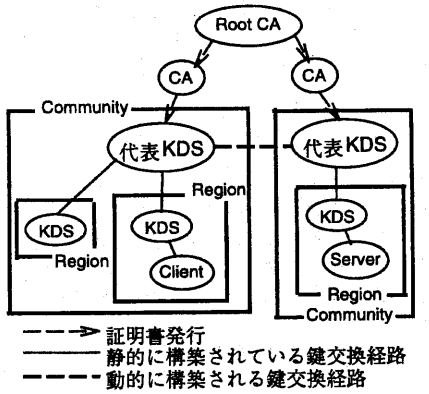

図 2: コミュニティ間通信モデル

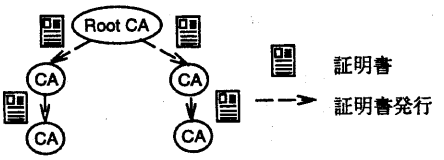

図 3: CA の階層構造

のような CA の階層構造をここでは「認証の鎖」といい、証明書から上位 CA をたどることを「認証の鎖をたどる」と呼ぶことにする。

本システムでは、鍵交換を行ないたい 2 つのコミュニティを証明する Root CA が同一のものである場合、相手コミュニティの存在は正しいものであると認証する。

3 システムの設計

本章では、前章で提案したコミュニティ間通信モデルに基づき、SPLICE/AS-II と CA を統合した新たな個人認証システムの設計について述べる。

3.1 鍵配布サーバ (KDS) および代表鍵配布サーバ (代表 KDS) の設計

KDS、代表 KDS はリージョン内のメンバの ID、公開鍵、秘密鍵、パスワード、鍵交換ポリシーなどのセキュリティ的に重要なデータを管理している。従って KDS および代表 KDS はネットワークセキュリティ上安全な計算機で運用され、悪意のない人物によって管理されていることを前提とする。

3.1.1 代表 KDS と KDS の接続形態

代表 KDS は、他コミュニティの認証を行うため認証の鎖をたどらなければならない。他の KDS に比べて処理に時間がかかることが予想されるため、代表 KDS は他コミュニティとの鍵交換にのみを行ない、コミュニティ内の鍵交換は行なわない。

3.1.2 鍵交換ポリシーの管理

代表 KDS では、他コミュニティとの間で鍵交換を行う際に考慮される鍵交換ポリシーを定義することができる。ここで定義されるポリシーとは以下のようなものである。

- 鍵交換を行うコミュニティの定義
  本システムでは、他コミュニティの認証を CA の発行する証明書の鎖をたどることにより実現している。しかし、他コミュニティを認証することと、そのコミュニティと鍵交換を行うことは別の問題である。このため、代表 KDS が鍵交換を行なうコミュニティを定義することにより、認証後の鍵交換を制御する。
- 信用する CA の定義
  Root CA を頂点とする CA の階層構造が存在しているとき、コミュニティによっては信用したくない CA が存在する可能性がある。そこで本システムでは、信用する CA を制限する機能を提供する。
- コミュニティ間での鍵交換の際の相手代表 KDS 公開鍵をキャッシュする
  CA の認証の鎖をたどることは時間がかかるため、キャッシュ機能を提供する。キャッシュを行なえば、認証の鎖をたどらない分高速に動作する。キャッシュ機能を利用するかしないかをポリシーとして定義できる。

3.1.3 代表 KDS と CA によるコミュニティ認証と公開鍵配送経路の構築

本システムでは、他コミュニティの認証を CA が発行する証明書を用いて行う。CA は Root CA を頂点とした階層構造を持つ。代表 KDS が他コミュニティを認証するには以下の手順をたどる。

ステップ 1
まず、代表 KDS は認証したいコミュニティの代表 KDS の証明書を発行した CA から、相手代表 KDS の証明書とその CA 自身の証明書を入手し、代表 KDS の証明書の有効性を検査する。さらに CA 自身の証明書から上位 CA の有無を調べる。その CA が Root CA である場合は、CA の証明書が自己証明書になっていることを確認し、ステップ 3 に進む。そうでない場合はステップ 2 へ進む。この様子を図 4 に示す。

ステップ 2

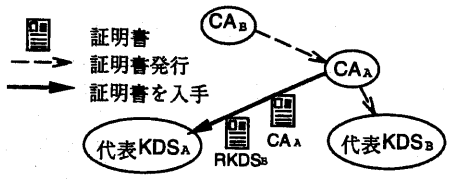

図 4: CA から代表 KDS の証明書と CA 自身の証明書を入手

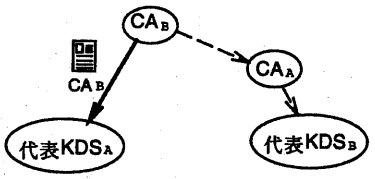

図 5: 上位 CA から、上位 CA 自身の証明書を入手

代表 KDS はステップ 1 で判明した CA から、CA 自身の証明書を入手する。さらに、その CA の上位にある CA に対し、入手した証明書が取り消されていないか問い合わせる。取り消されている場合は、そこで認証の鎖が切れているため、認証に失敗したと判断する。有効である場合は、上位 CA 自身の証明書を入手し、証明書の有効性を確認すると共に、上位 CA の有無を調べる。その CA が Root CA である場合は、CA の証明書が自己証明書になっていることを確認し、ステップ 3 に進む。そうでない場合はステップ 2 を Root CA が見つかるまで繰り返す。この様子を図 5に示す。

ステップ 3

代表 KDS は、自分自身の証明書を発行した CA から自分自身の証明書を入手する。以下、ステップ 1、ステップ 2 と同様に、自分自身の認証の鎖をたどり認証の鎖のチェックと Root CA の確認を行なう。

ステップ 4

代表 KDS 自身の証明書をたどって得られた Root CA と、認証したいコミュニティの代表 KDS の証明書をたどって得られた Root CA が一致するかどうか調べる。一致した場合、2 つのコミュニティは同一の Root CA によって存在が保証されているため、相手コミュニティが正しく認証できたものとする。

このように、認証の鎖をたどる作業はコストがかかる。しかし、現在の CA には上位 CA で取り消しが行われた証明書の情報を即座に CA の階層構造全体に伝える有効な方法が存在しない。つまり、ある CA から相手の代表 KDS の証明書を入手しその有効性が確かめられても、それより上位の CA によって、CA の証明書が取り消され、認証構造が成立しない状況になっていることも考えられる。そこで本システムでは、認証の鎖を Root CA までたどることによって証明書の有効性を検査する。

3.2 Secure-TCP との統合

クライアント、サーバ間の認証が成立すると、必要に応じて両者の間で暗号化通信を行う。本システムでは暗号化通信は Secure-TCP を利用し、以下のようなソケットオプションを定義することによって、暗号化通信で用いられるセッション鍵の受渡しを行なう。

- 暗号化を行なう鍵の受渡し

```
setsockopt(int sockfd, int IPPROTO_TCP,
           int STCP_ENC_KEY,
           char key, int sizeof(key))
```

- 復号化を行なう鍵の受渡し

```
setsockopt(int sockfd, int IPPROTO_TCP,
           int STCP_DEC_KEY,
           char key, int sizeof(key))
```

ここで、IPPROTO_TCP、STCP_DEC_KEY、STCP_ENC_KEY は Secure-TCP で定義されている変数、key は受け渡す鍵である。

4 実験と評価

4.1 実験環境の構成

本システムの動作実験を行なうために、2 つのコミュニティと 3 つの CA から構成された実験環境を構築した。CA には SPARCstation-10(SunOS4.1.3) を 2 台、Sun4/40(SunOS4.1.3_U1) を 1 台、コミュニティを構成するマシンには主に BSD/OS2.1 の Pentium マシンを使用した。実験環境を図 6に示す。

本システムでは、CA として認証実用化実験協議会 (Initiatives for Computer Authentication Technology、以下 ICAT) の広域認証技術研究タスクフォースが開発した ICAT Certification Authority Package を用いた。また、本システムは公開鍵暗号として松下電気産業株式会社が開発した楕円暗号プログラム MY-ELLTY[4] を用いて実装した。

4.2 評価

今回開発したシステムのライブラリを telnet クライアント、telnetd に組み込み、動作実験を行なった。その結果システムが設計通りに動作していることを確認した。今回開発したシステムでは、以下の 2 点について評価を行なう。

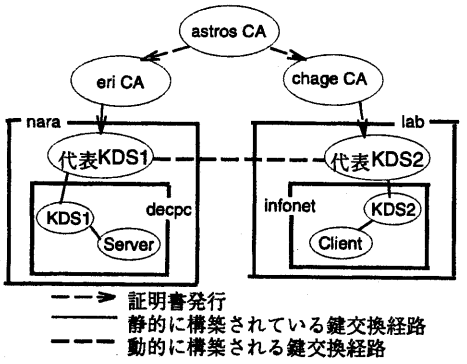

図 6: 実験環境

- クライアント、サーバがお互いに相手の公開鍵を入手するのにかかる時間
  クライアントとサーバが同一のコミュニティに属している場合と、異なるコミュニティに属している場合について、相手の公開鍵を入手するのに必要な所要時間の比較を行なう。
- 代表 KDS の処理性能
  コミュニティ間での鍵配送の性能は、代表 KDS の性能に大きく依存するものと思われる。代表 KDS の各プロトコルに対する処理時間を計測し、インターネット環境に適用する際の有効性を考察する。

4.2.1 クライアント、サーバがお互いに相手の公開鍵を入手するのにかかる時間

本システムでは、コミュニティ間で鍵交換を行なう際に認証の鎖をたどる必要があるが、キャッシュを用いることでその作業を軽減することができる。このため、キャッシュの有無別にクライアント、サーバ間でお互いの公開鍵を入手する際の所要時間をクライアント側で計測した。また、実験環境ではクライアント、サーバがお互いの公開鍵を入手するのに代表 KDS および KDS を計 4 段経由する。コミュニティ内の鍵交換に必要な時間とコミュニティ間の鍵交換に必要な時間を比較するため、コミュニティ内の鍵交換を行なう際に、KDS を 4 段経由する場合の取得所要時間を計測した。それぞれの場合の計測結果を表 1に示す。表 1によると、コミュニティ内で鍵交換を行なった場合と、認証の鎖を確認しないように設定されている場合とでかかった時間にはほとんど差がないことが分かる。しかし、認証の鎖をたどらない場合は、キャッシュの有効期間内に証明書が取り消されても、その結果は反映されない。鍵交換ポリシーの設定により公開鍵交換にかかる時間は大きく異なり、また安全性も異なる。コミュニティ間における鍵交換時間と安全性はトレードオフの関係にあるので、本システムではコミュニティ管理者がネットワーク環境に応じてキャッシュの有無と有効期間を設定可能である。

表 1: ポリシー別認証フェーズの所要時間

公開鍵転送フェーズ	所要時間 (s)
両方の認証の鎖を確認する場合	145.806
自分の認証の鎖を確認しない場合	81.400
相手の認証の鎖を確認しない場合	81.762
両方の認証の鎖を確認しない場合	15.886
コミュニティ内で鍵交換を行なう場合	15.136

表 2: 代表 KDS の動作別処理時間

対象動作	user time system time
a. 他コミュニティ公開鍵取得 (両方の認証の鎖を確認しない場合)	3.5422502 0.0464797
b. キャッシュしていた鍵の読み込み	0.0002458 0.0002999
c.CA 自身の証明書を入手する	0.0225472 0.0126583
d.CA に登録されているメンバの 証明書を入手する	0.0297794 0.0639457
e.CA に登録されているメンバの 証明書が有効であるか問い合わせる	0.0187803 0.0021930
f. 証明書の有効性を検査する	0.0091113 0.0092333

4.2.2 代表 KDS の処理性能

本システムは、他コミュニティと鍵交換を行なう際に必ず代表 KDS を経由するため、システム全体の性能は代表 KDS の性能に大きく依存する。そこで、代表 KDS の各プロトコルに対する処理時間を計測した。計測結果を表 2に示す。認証の鎖の段数を n 段 ($n > 0$) とすると、代表 KDS の処理時間 TIME は以下のように算出できる。

$$TIME = a - b + 2\{nc + d + (n-1)e + (n+1)f\} + f \quad (1)$$

現在 ICAT が行なっている CA 運用実験で構築されている Root CA からの段数は 3 段である。従って、ここでは n=3 のときの代表 KDS の処理性能を評価する。式 (1) に n=3 を代入すると以下のようになる。

$$TIME = 4.235862$$

代表KDSに対する鍵要求の到着がポアソン分布に従い、ポアソンの到着率をλ、サービス率をμ、計算機上で稼働させる代表KDSの数をmとすると、待ち合わせ時間とサービス時間の合計Tは次のように導出できる。

$$\rho = \frac{\lambda}{m\mu} \tag{2}$$

$$p = \left[\sum_{n=0}^{n-1} \frac{(mp)^n}{n!} + \frac{(mp)^m}{m!(1-\rho)} \right]^{-1} \tag{3}$$

$$P = \frac{p(mp)^m}{m!(1-\rho)} \tag{4}$$

$$T = \frac{1}{\mu} + \frac{P}{(m\mu - \lambda)} \tag{5}$$

また、式(2)はトラフィック密度を表すので、これを変形すると1プロセス当たりに鍵配送要求が到達する間隔は以下の式で表される。

$$\frac{m}{\lambda} = \frac{1}{\rho\mu} \tag{6}$$

よって、負荷が変化した際に、Tが$\frac{m}{\lambda}$を下回れば代表KDSは動作すると考えられる。

表2の計測結果と$n=3$を式(1)に当てはめると、TIMEは約4.2であったので、1つの鍵配布サーバのサービス率μを1/4.2としたとき、鍵配布サーバの数が1、5、9、40それぞれの場合の応答時間を図7に示す。この結果、負荷が0.8のとき、代表KDS内のサーバプロセスを9つ駆動させれば代表KDSは動作することが分かる。これは、代表KDSに対し鍵配送要求が到着する間隔が約1.7秒以上であれば、代表KDSが動作することを意味する。

以上で得られた代表KDSに対する要求の頻度から考えると、例えばコミュニティのメンバが平均して約30分に1回以下の頻度で他コミュニティとの通信を行なうのであれば、1つのコミュニティは約1000人程度のメンバを維持することが可能である。

本システムは、メンバ間でコネクションを確立するたびに鍵配送が行なわれる。よって、例えばWebのように頻繁にコネクションの確立と切断を繰り返すサービスでは、鍵配送系にかかる負担が大きい。従って、本システムを大規模ネットワーク環境に適用する場合は、例えばtelnetやftpのように一度メンバ間でコネクションを確立すると、ある程度長期に渡ってコネクションが維持されるようなサービスに対して有効であると考えることができる。

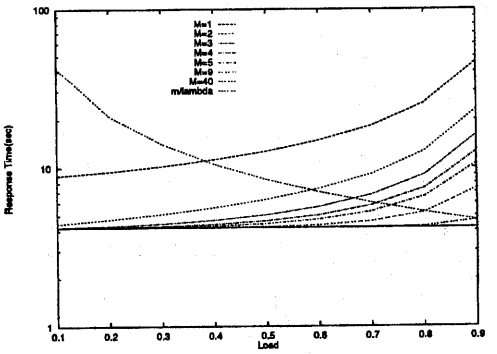

図7: 代表KDSの鍵配送要求処理時間

5 おわりに

本稿では、認証システムSPLICE/AS-IIと認証局であるCAを基に、よりスケーラビリティを重視して開発した個人認証システムを提案して、運用実験を通じて評価を行なった。今後、インターネット環境での運用機能を強化するため、さまざまなアプリケーションのシステム対応化を行なう予定である。

参考文献

[1] 白崎博生. インターネット環境における個人認証システムの開発. 修士学位論文. 大阪大学大学院基礎工学研究科 Mar.1996

[2] S.Kent. Privacy Enhancement for Internet Electronic Mail:Part II: Certificate-Based Key Management. RFC1422. Feb.1993.

[3] T. Tsutsumi, and S. Yamaguchi. "SecureTCP — providing security functions in TCP layer" INET'95, Honolulu, USA, no.T3, pp.905-913. June 1995.

[4] 宮地充子. 楕円曲線暗号の設計とIEEE標準化動向 ftp://ftp.icat.or.jp/pub/crypto-tf/wdeca-4.ps