平成 14 年度

学士学位論文

SAS 認証を用いた Web 通信方式

An Encription Communication Method with SAS

1030266　　小西 竜也

指導教員　　清水 明宏

2003 年 2 月 12 日

高知工科大学 情報システム工学科

要 旨

SAS 認証を用いた Web 通信方式

小西 竜也

安全かつ高速な暗号化通信を実現する新しい方式を提案し，従来技術との比較を行う．

Web 上での通信において，保全性と機密性を確保する暗号化ベースのプロトコルとして，現在最も広く普及している技術は SSL(Secure Socket Layer) である．SSL は公開鍵暗号方式を用いての共通鍵共有を行っているため，コンピュータや携帯電話等の各端末にかける処理負荷が大きい事や，また，認証部分についても幾つかの問題点が存在する．

本稿では，既存技術の問題点を解決するための，新しい方式を提案する．提案方式には SAS（Simple and Secure）認証方式を用い，SSL の暗号化通信方式との比較・検証を行う．

キーワード　SSL，公開鍵暗号方式，鍵共有，SAS

Abstract

An Encription Communication Method with SAS

Tatsuya Konishi

I propose a new method that realizes secure and speedy encription communications. Moreover, I compare with a existing method.

The SSL(Secure Socket Layer) is most used in Web communication systems. However, the SSL has costs because the SSL uses a public-key cryptography for a key agreement. Therefore, such technique is useless for many machines: mobile phones and servers which has many user.

In this thesis, I propose a new method which solves such problems. The new method applies SAS(Simple and Secure) password authentication method, In addition I evaluate and compare with SSL that is encipherment communication form.

key words SSL, Public-key cryptography, key agreement, SAS

目次

図目次

表目次

第 1 章

はじめに

近年，インターネットの爆発的な広がりにより様々なサービスが提供されるようになった．その代表的なものとして，電子商取引やオンラインショッピング等が挙げられる．これらのサービスにおいて，サービスを利用する個人または企業は自らを証明し支払いを行う必要があるが，特定の組織や個人を示す情報や金額等の情報は悪意ある第三者に悪用される可能性があるため，特に安全に送信する必要がある．従来よりこのような要求に対応する様々な技術が開発されているが，代表的なものとして，ユーザ認証及び暗号化通信に SSL プロトコルを用いた https が存在する．この現行の方式は広く普及しているが，幾つかの問題点も指摘されている．

本研究はそれらの問題点を解決し，より高速で安全な通信を実現するため，ユーザの認証及び鍵共有に SAS 認証方式 [1] を用いたプロトコルを提案する．さらに，現在最も普及している SSL の暗号化通信方式との比較評価実験を行い，その有用性を評価した．

本論文では，第二章で研究背景として SSL の特徴とそれらが抱える問題点を述べ，第三章ではその問題点を解決する新たなシステムとして SAS-2 認証方式 [2] を用いたプロトコルを提案し，その機能とアルゴリズムについて述べる．第四章では実験環境等の詳細を述べ，提案した方式と SSL 暗号方式との比較評価実験を行いその有効性を評価する．さらに，第五章で今後の課題を述べ，第六章でむすびとする．

第 2 章

研究背景

本章では，従来からあるプロトコルである SSL の特徴について述べる．また，SSL の抱える問題点を明らかにする．

2.1 SSL について

インターネットの爆発的な成長とユーザの増加に伴い，オンラインでの決済などが広く行われるようになった．しかし，現在インターネットで使われている通信方法は，情報の安全性があまり考慮されなかった時代に設計されているため，オンライン決済で使われるような守秘義務を必要とする個人情報の送信には適していない．そこで，現在の通信方法のまま，より安全に情報をやり取りできるように，SSL(Secure Sockets Layer) を用いてユーザの認証や暗号化通信を行う https 通信が考案された．

Web 上での暗号化通信には，通信を行う両者間で鍵を保有しなければならない．この鍵は暗号化及び復号を行うための鍵で，共通鍵暗号方式では，この鍵は同一のものである．通信者の一方が，この鍵を共有するために通信相手に同一のものを送信する必要があるが，送信途中で悪意のある第三者に盗み取られると，通信を行う両者間でやりとりされるデータを全て読み取られてしまうという問題が起こる．他の暗号化方式としては，公開鍵暗号方式と呼ばれる暗号方式がある．この方式は共通鍵暗号方式とは異なり，同一の鍵を共有する必要が無いというメリットがある．暗号化通信を行う時は相手の公開鍵で暗号化し，そのデータを相手に送信する．送信された側は，自分の公開鍵に対応した秘密鍵を保持しており，これを用いてデータの復号を行う．しかし，公開鍵暗号方式は計算量が多いため，処理時間及び

マシン負荷の増加といった問題点が存在する．そこで，SSL ではこの両者を組み合わせたハイブリッド方式が用いられ，共通鍵の共有に公開鍵暗号方式を利用している．また，その中で，証明書やデジタル署名を使った通信相手の認証及び検証が行われている．

SSL は，サーバ・クライアント暗号化通信部分に DES[3] や IDEA[4] 等の共通鍵暗号方式を用い，その共通鍵を共有する手段として RSA[5] や Diffie-Hellman[6] 等の公開鍵暗号方式を用いる．また, 送信先の公開鍵の信頼性を確認するために, Web コンテンツの信頼性を保証する第三者機関からデジタル署名書を受け取る．そして，伝送データの完全性の確認には，メッセージダイジェスト関数 (MD5[7] や SHA-1[8] 等) を用いている．しかし, この SSL プロトコルも幾つかの問題点が指摘されている．

以下に，SSL の通信略図を示す．

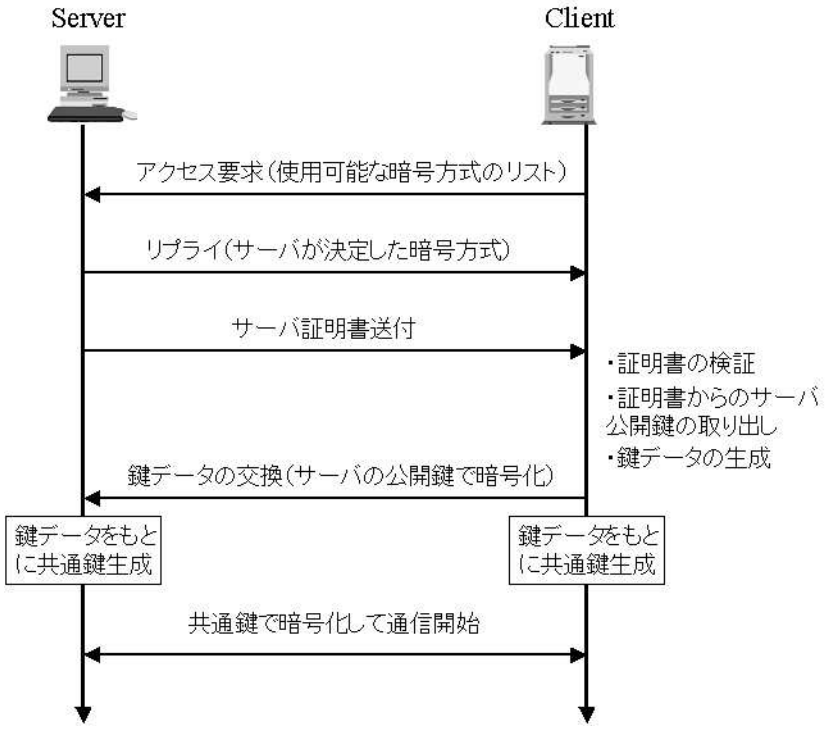

図 2.1　SSL プロトコル略図

2.2 SSL の問題点

現在の暗号化通信で最も使用されている SSL にも問題点が存在する．それらを以下に示す．

1. 処理速度・負荷の問題

   第 2.1 節でも述べたが，SSL ではデータの暗号化及び復号を行う共通鍵を共有するための手段として，公開鍵認証方式を使用している．代表的なアルゴリズムは RSA であるが，これは共通鍵アルゴリズムに比べて非常に処理速度が遅いという欠点がある．RSA は通常，代表的な共通鍵アルゴリズムである DES の実装と比較して 100 から 1000 倍の時間がかかってしまう．そのため，SSL では共通鍵を安全に共有するためだけに用いている．しかし, 共通鍵共有部分のみでの使用であっても，SSL を用いて Web 通信を行う https において，演算処理能力の低いスペックのマシンでは通常のブラウジング品質が得られなくなってしまうという問題点や，クライアントの増加やそれに伴うサービスの多様化により，サーバへの負荷の増大という問題点が指摘されている．

2. 認証の安全性

   SSL を用いた https 通信では，証明書の偽装等による以下のような問題点が存在する．

   - 別の Web サイトのように見える Web サイトを作成し，正規の Web サイトとして SSL セッションを確立する． （Web サイトの偽装）
   - 別のユーザに属すと見なされるデジタル証明書を使用して，署名された電子メールを送信する． （電子メール署名の偽装）
   - 信頼される証明機関から発行されたように見える証明書を使用して，悪意のあるプログラム (Malware) にデジタル署名を行う．
   - 証明書を基にした認証システムを偽装し，高い特権を持つユーザーとして権利を取得する．

3. 導入においての個人の負担

   https を実際に導入をしようとした場合，VeriSign 社等の証明書をインストールする必

要がある．これには数万円の費用がかかり，個人での導入・運用を考えた場合は大きな負担となる．

2.3 モバイル端末でのインターネット利用について

近年，携帯電話や PDA 端末（Personal Digital Assistants）をはじめとするモバイル端末の急速な普及とそれらのインターネットへの接続性の強化により，種々のインターネットモバイル向けサービスが提供され始めている．特に，情報メモリー装置等の小型化・高機能化・低廉化に伴い、NTT DoCoMo の i モードや，AU の EZweb，J-Phone の J-sky 等，携帯電話からのインターネット接続サービスが爆発的に普及してきており，モバイル端末からインターネットに接続して情報を受発信できる情報通信環境が目覚ましい進展をみせている．さらに，音声やデータに加えて，動画像等の通信が可能となるいわゆる次世代携帯電話のサービスも開始される等，一層の高機能化が進んできている．今後も，携帯電話加入者のインターネット接続サービス加入者数が増大する事は確実であり，近い将来には携帯電話加入者の大半がサービスに加入するであろうと言われている．

また，モバイル端末によるインターネット接続の著しい普及・進展は，インターネットへの接続端末がパソコンの独占ではなくなる先駆けであり，今後は高機能化するモバイル端末とともに，テレビ等を筆頭とする「情報家電」が，パソコンよりもはるかに簡易な操作性を伴って，パソコンにとってかわる時代が到来するものと考えられている．そのため，急増するインターネットサービス利用者による，ダウンロードサイトやオンラインショッピングの利用における個人情報等の取り扱い，また，情報家電においては各端末の認証等のための，安全かつ効率の良い通信方式が必要となってくる．

これらのニーズに対応するべく，携帯電話では実際に暗号通信に SSL が使用されている．しかし，いくら高性能化した携帯電話とはいえ，公開鍵暗号方式を用いる SSL の利用は端末にかかる負荷が大きい．ブラウザのバージョンによる，SSL 対応の有無という問題点もある．他にも，証明書の有効期限が切れてしまうという点や，それによる認証局（CA）から

の証明書取得の必要性，及び証明書のインストールに伴う Web サーバに関する知識の必要性といった様々な問題点が存在する．そのため，現行の技術よりも軽量且つ効率の良い，安全な通信方式が必要となってくる．

本研究での提案方式は，安全性は勿論，処理量や速度といった点からも特に携帯電話や PDA 等のモバイル端末，上述したインターネット対応製品への適用において特に有効であると考える．

第 3 章

SAS を用いたプロトコルの提案

本章では，前章で述べた問題点を解決した新たな認証方式を提案し, その概要を示す.

3.1　提案プロトコル

3.1.1　共通鍵の共有を SAS-2 認証方式で行う

SSL では，共通鍵の共有に RSA 等の公開鍵暗号方式を用いる．この方式は，第二章でも述べたように非常に複雑な計算を行うため，コンピュータにかかる負荷の増加や計算時間がかかってしまう等の問題点が存在する．この SSL における公開鍵暗号方式での共通鍵共有を，SAS-2 認証方式で用いたデータを元に互いに鍵生成をする事により，共通鍵のデータを送信せずに共有可能となる.

また，SAS-2 を用いた認証段階において，悪意ある第三者による何らかのデータ操作があった場合，その時点で通信を終了し鍵の生成は行わない．そのため，共通鍵を生成するのは安全に認証が行われた場合に限るため，鍵共有データの盗聴等による通信傍受という問題も解決している.

3.1.2　ユーザ認証に SAS-2 認証方式を用いる

SSL 認証方式は，認証を行う際，クライアントがサーバから送付されてくる証明書を用いて認証を行っている．相互認証を行うためには，クライアントも同様に証明書を送付するが，現在の Web 上の殆どのコンテンツは，サーバがクライアントに対して証明書を送付す

るだけの一方向性の認証である．また，第二章の SSL の問題点の部分でも述べたが，SSL は証明書の偽装等による，悪意ある第三者のなり済ましの可能性といった問題も存在する．しかし，SAS-2 認証方式を用いた本方式ならばワンタイムパスワードを用いているため安全に相互認証が可能であり，また，高額の証明書を購入する必要も無い．

以下に，提案方式の通信略図を示す．

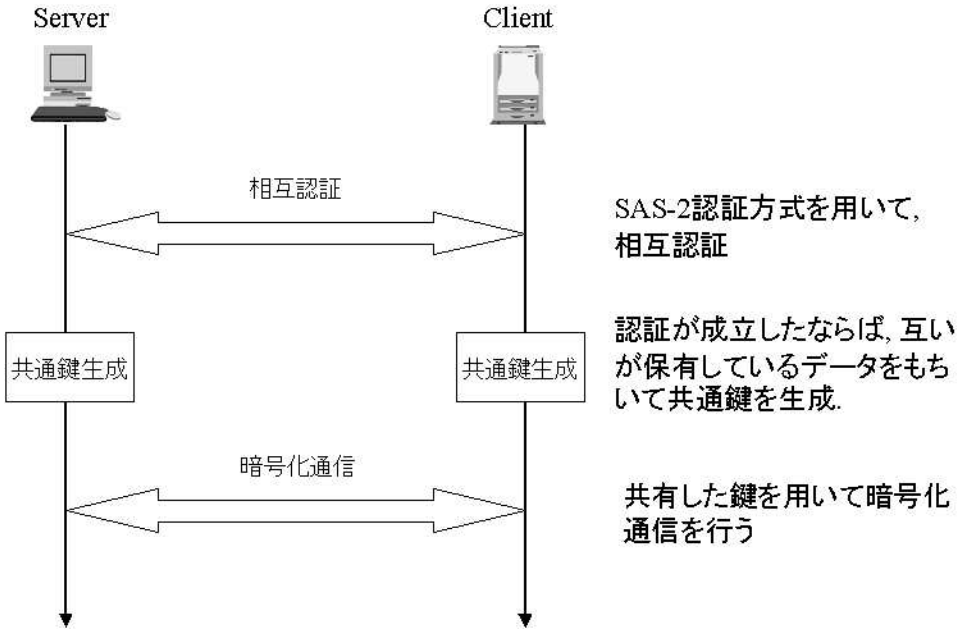

図 3.1　提案方式プロトコル略図

3.1.3 SAS 認証方式及び SAS-2 認証方式について

SAS 認証方式には，SAS-2 認証方式だけではなくその前身である初期の SAS 認証方式も存在するが，本方式では SAS-2 認証方式を採用した．その理由としては以下の要因が挙げられる．

1. 通常 SAS 認証方式では行っていないサーバ・クライアント間の相互認証が可能となる．
2. SAS 認証方式に比べ一方向性関数の使用回数が減少したため, 処理量の削減が可能となる．
3. アルゴリズムのバリエーションが数パターンあるため, 安全性をより向上させる事が出来る．

以下に，本システムで用いる SAS-2 認証の処理シーケンス及びシーケンス図を示す．

- ID：ユーザ ID
- S：ユーザのパスワード
- X,F,H：一方向性関数 (ハッシュ)

  例：H(x) は，x に 1 度ハッシュを適用したという事を意味する．
- i：認証のセッション回数を示す 0 以上の整数．
- N_i：i 回目の認証で用いられる乱数．
- $\oplus$：排他的論理和
- ＋：加算

登録フェーズ

1. ユーザは ID 及び S を入力．同時に，乱数 N_1 を生成し，登録する．その後，入力データ及び乱数を用いて $\mathrm{A} = \mathrm{X}(\mathrm{ID},\mathrm{S} \oplus N_1)$ を計算する．
2. ユーザは，セキュアなルートを用いて ID 及び A をサーバに送信．
3. サーバは，認証フェーズで使用するためのデータである ID 及び A を登録．

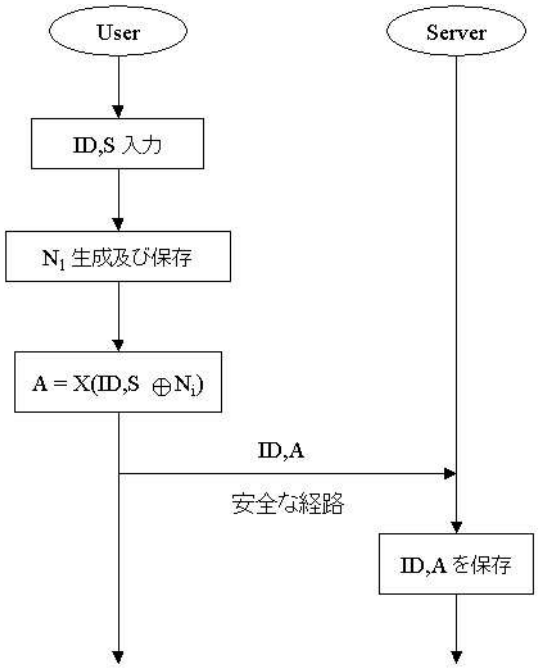

図 3.2　SAS-2 登録フェーズシーケンス図

認証フェーズ

1. ユーザは ID 及び S を入力．入力データ及び登録している N_i を用いて A = X(ID,S $\oplus$ N_i) を計算する．次にユーザは乱数 N_{i+1} を生成し，登録．その後，入力データ及び乱数 N_{i+1} を用いて C = X(ID,S $\oplus$ N_{i+1})，F(C) = F(ID,C) を計算．ここでの C とは，次回の認証情報を意味する．さらに，α = C $\oplus$ (F(C) + A) 及び β = F(C) $\oplus$ A の計算を行う．
2. ユーザは ID 及び α，β を，インターネット等の一般的なネットワークを用いてサーバへ送信する．
3. サーバは，登録データである A を用いて β $\oplus$ A を行い，F(C) を取り出す．そして，その F(C) を用いて α $\oplus$ (F(C) + A) から C を得る．次に，サーバは F(ID,C) を計算して，取り出した F(C) との比較を行う．もし一致しなければ認証失敗となり，その時点で終了となる．一致した場合は認証成立となり，次のステップへと移行する．
4. サーバは次の認証セッションのために，C を A の替わりに保存し γ = H(ID,F(C)) を計算する．
5. サーバは，γ をインターネット等の一般的なネットワークを用いてユーザへ送信．
6. ユーザは H(ID,F(C)) を計算し，送信されたデータである γ との比較を行う．もし一致したならば，サーバ・ユーザ間の相互認証が成立した事を意味する．

3.1　提案プロトコル

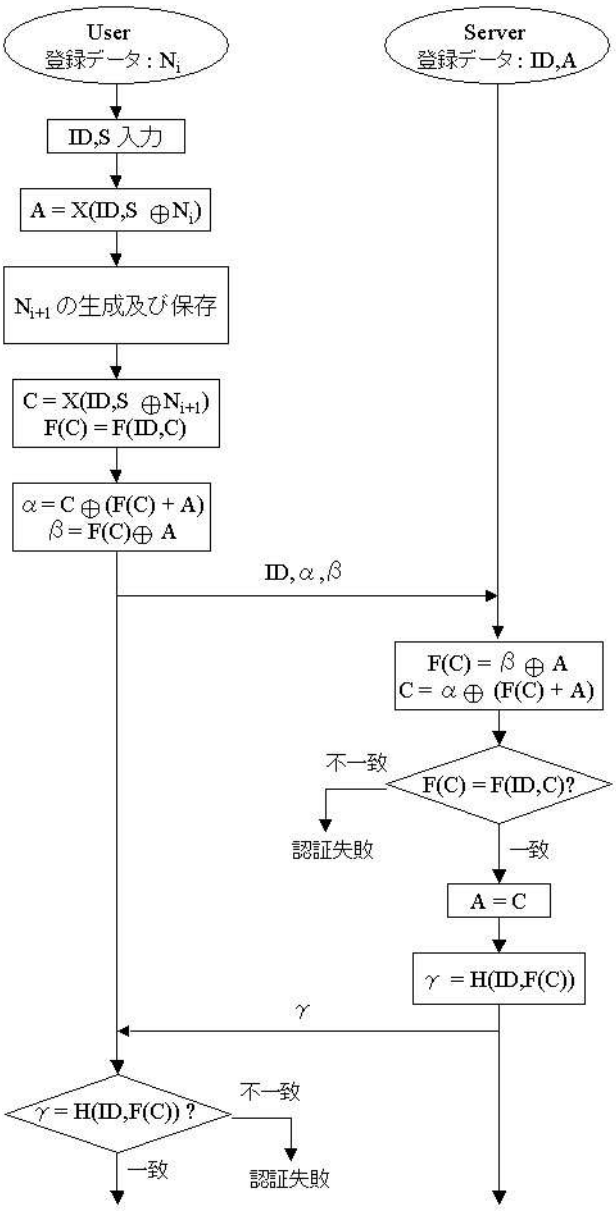

図 3.3　SAS-2 認証フェーズのシーケンス図

3.2　機能概要

提案方式の認証及び暗号化の流れを以下に述べる．

1. ユーザからの要求をサーバが受け取る．
2. SAS-2 を用いてサーバ・ユーザ間で認証を行う．相互認証が成立しなかった場合はエラーを表示して終了する．この場合は，鍵の生成は行わない．
3. 相互認証が成立した場合，サーバ及びユーザは，SAS-2 認証で使用したデータを用いて乱数を生成する．
4. サーバ及びユーザは，生成した乱数を用いて鍵の基となるデータを生成．鍵のアルゴリズムには DES を用いている．
5. 生成したデータを基に，各自で共通鍵を生成．
6. 相互認証及び共通鍵の共有が成功すると，サーバはリクエストデータをサーバ側の共通鍵を用いて暗号化．ユーザに送信する．
7. 暗号化されたデータを受け取ったユーザは，クライアント側の共通鍵を用いて受け取ったデータを復号する．

以下に，提案方式のシーケンス図を示す．

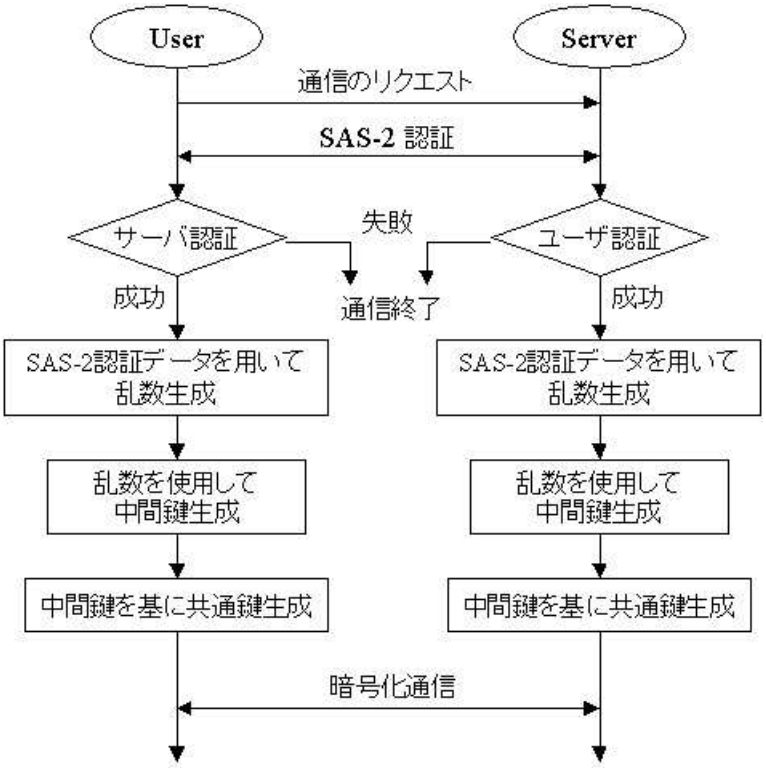

図 3.4　提案方式シーケンス図

第 4 章

比較評価実験

本章では，前章で提案した新たなプロトコルと現在最も普及している SSL を比較し，その有用性を評価するために行う実験について説明する．また，実験によって得られた結果についての評価も行う．

4.1 比較方式について

4.1.1 プロトコルの実装方法

提案方式である，SAS-2 認証方式を用いたプロトコルを Java を用いて構築．さらに，既存技術である SSL の暗号化アルゴリズムも Java を用いて実装し，同一環境での比較・検証を行う．その上で，提案方式の有用性を検討する．

4.1.2 暗号方式について

SSL で用いられる鍵共有のアルゴリズムは，主に RSA と Diffie-Hellman の 2 種類である．

RSA(Rivest, Shamir, Adleman) アルゴリズムは，大きな数を因数分解する事の難しさを利用してセキュリティを確保する．このアルゴリズムは，暗号化にもデジタル署名にも利用可能である．

以下に，RSA を用いた場合のシーケンス図を示す．

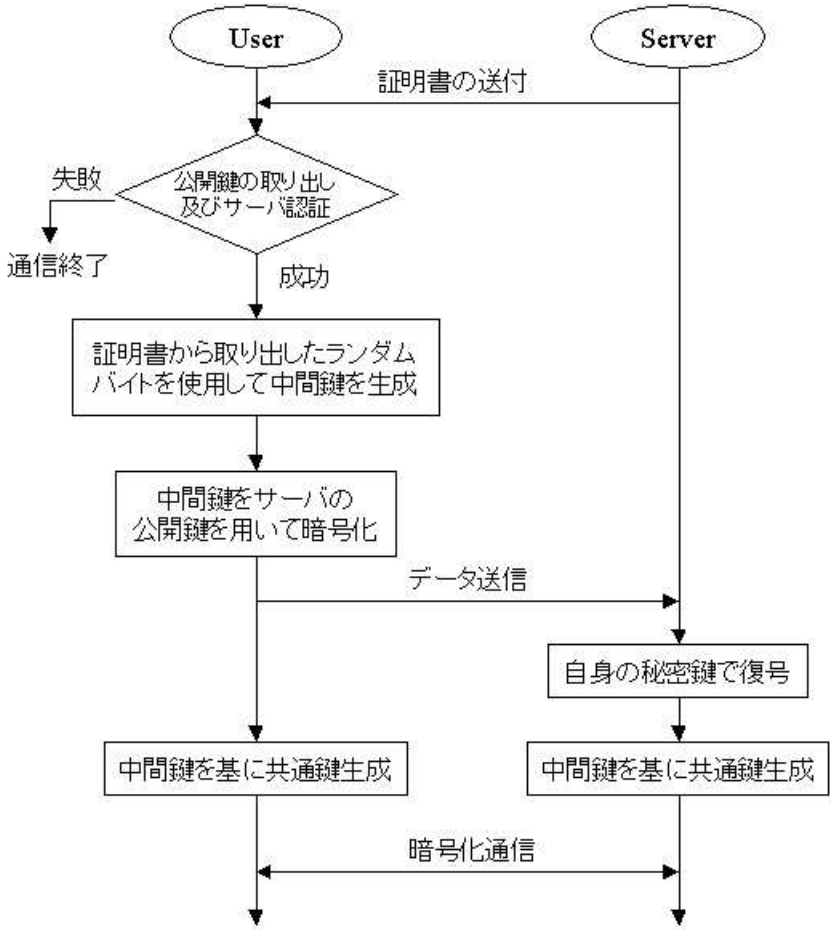

図 4.1　RSA シーケンス図

Diffie-Hellman アルゴリズムは，大きな有限体に対する離散対数の計算が困難であるという事実に基づいた方法である．RSA とは異なり，このアルゴリズムは暗号化の実行には使用出来ず，鍵共有のみに適用可能である．

以下に，Diffie-Hellman を用いた場合のシーケンス図を示す．

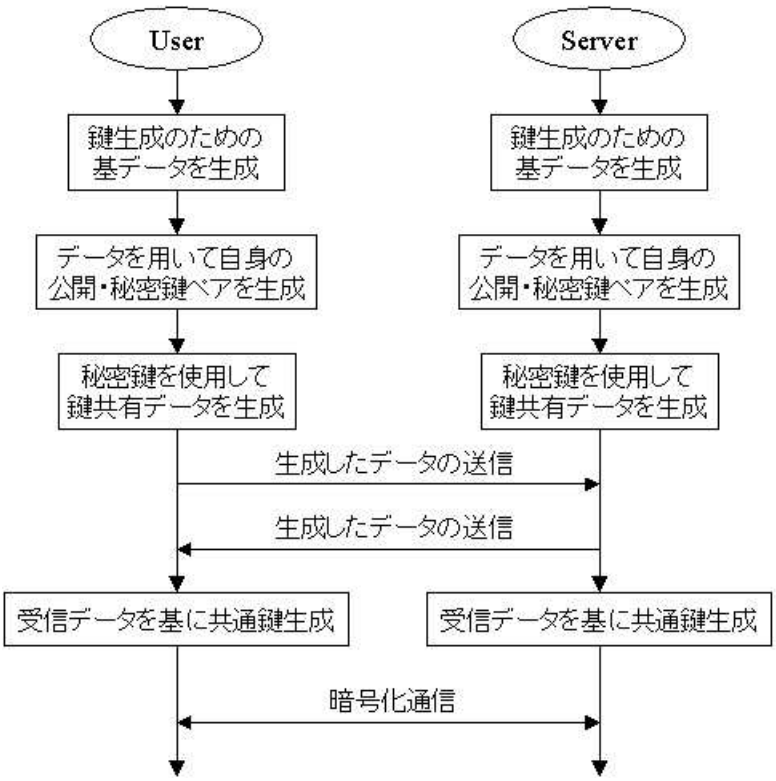

図 4.2　Diffie-Hellman シーケンス図

本研究では既存技術の比較対照として，最も一般的に利用されている RSA アルゴリズムを使用する．また，鍵共有部分でのみ SSL に使用されている Diffie-Hellman アルゴリズムについても比較を行う．これは，厳密に言えば認証には用いられていないため比較の対象としては異なる．しかし，鍵共有という部分のみでの比較対照として，Diffie-Hellman アルゴリズムも採用した．RSA アルゴリズムにおいては，使用されている鍵の bit 数は一様ではなく幾つかのパターンが存在する．最も一般的に使用される bit 数の範囲は 512bit から 1024bit である．そのため，本実験においてはこの範囲での最長と最短の値を選択し，512bit と 1024bit の 2 パターンでの実験を行う．

実験に用いるシミュレータに関しては本来の流れとは若干異なり，簡略化している．RSA を用いた場合はクライアントが最初からサーバの証明書を保持しており，Diffie-Hellman も同様に，アルゴリズムで使用する初期データをサーバ・クライアント双方が共有しているものとした．

共通鍵暗号方式のアルゴリズムには，現在最も一般的である DES(Data Encryption Standard) アルゴリズムを用いている．現在の SSL プロトコルでは，より強力な DESede(Triple DES) アルゴリズムを使用する場合もあるが，本提案方式でも DESede やその他の共通鍵暗号方式は適用可能である．また，乱数の生成に用いるアルゴリズムは，SHA-1 を使用している．これについても，同様に他のアルゴリズムの適用が可能である．

4.1.3 比較項目

先に述べた，SAS-2 認証方式，及び RSA，Diffie-Hellman を用いて共通鍵を共有する部分までの，以下の点についてを評価項目とする．

1. サーバでの処理時間及び CPU 使用率
2. クライアントでの処理時間及び CPU 使用率

4.2　実験環境

実装した各方式の開発環境及び比較実験を行う際の各環境を示す．

4.2.1　実験構成

- 本実験では，サーバ・クライアント間で通信を行うシミュレータを用いるため，サーバを想定した高スペックのマシン，一般的なクライアントを想定した比較的低スペックのマシンをクロスケーブルを用いて接続した．また，より詳細なデータを得るために，クライアントに高スペックなマシンを用いた場合についても実験を行った．以下の図 4.1 に，比較評価実験のシステム構成図を示す．

図 4.3　比較評価実験システム構成図

- ケーブル

  10Base-T，カテゴリー 5 のクロスケーブルを使用．

  本実験は，ネットワークの状況により計測値が大きく変動する事が予想される．従って，他のトラフィックの影響を考慮する必要の無い，クロスケーブルを用いた通信での実験を行う．

4.2.2　動作環境

以下の表に，実験時の各構成要素の動作環境を示す．

H/W 要件	名称
機種	PC/AT 互換機
CPU	Athron XP 1.6Ghz
Memory	128MB
Hard Disk 容量	40GB

表 4.1 サーバ ハードウェア環境

H/W 要件	高スペッククライアント	低スペッククライアント
機種	PC/AT 互換機	PC/AT 互換機
CPU	Pentium 750Mhz	Pentium 345Mhz
Memory	785MB	128MB
Hard Disk 容量	50GB	40GB

表 4.2 クライアント ハードウェア環境

S/W 要件	名称	バージョン
OS	Windows	2000
Java	Java 2 SDK	Standard Edition(1.4.1-01)
Provider	Java Crypto and Security Implementation(JCSI)	Provider 2.3

表 4.3 ソフトウェア環境

4.3 実験

これまでに述べたアルゴリズムや環境のもと，シミュレータを用いての実験を行う．

作成した各アルゴリズムのシミュレータを実行し，それぞれについて DES の共通鍵を共有するまでの処理速度，CPU 使用率を計測した．各グラフはそのプロセスを 10 回行った上での平均値を取ったものである．

以下の図にその結果を示す．

1. サーバにおける RSA(鍵長 512bit) と SAS-2 との比較

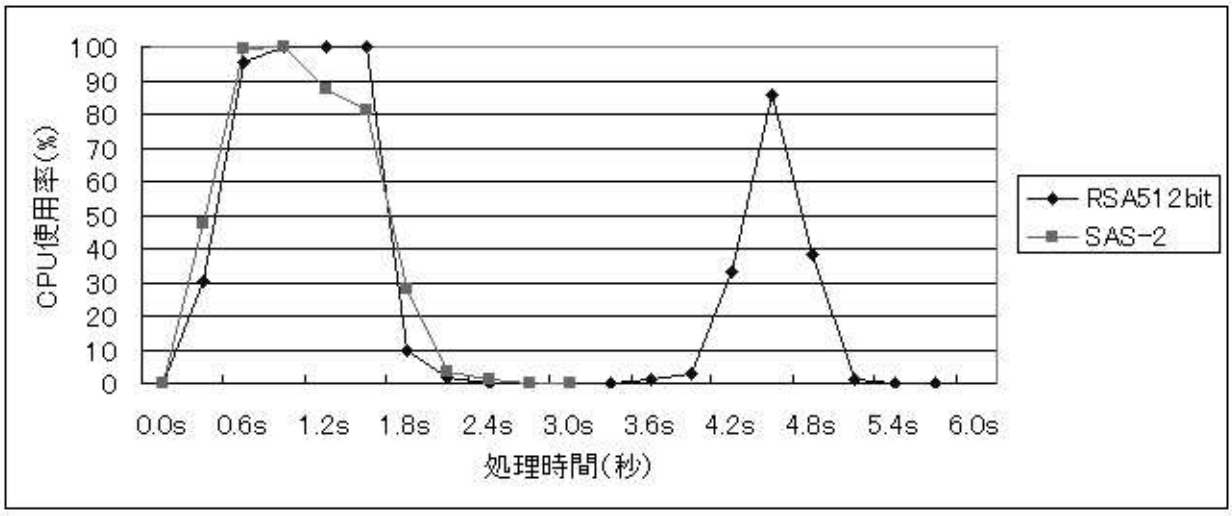

図 4.4 サーバにおける RSA(鍵長 512bit) と SAS-2 との比較

鍵長 512bit の RSA 及び SAS-2 の比較では，提案方式の方が処理速度が早く低負荷であった．RSA において CPU 使用率の上昇が二度存在する理由としては，クライアントの処理速度が低速であるための遅延であると考えられる．

CPU 使用率を高さ，応答時間を底辺とし双方の面積比を計算すると，提案方式が鍵長 512bit の RSA を使用した場合に比べて約 25 %小さい事が分かり，一つの目安として 25 %動作が軽快であると言える．

2. サーバにおける RSA(鍵長 1024bit) と SAS-2 との比較

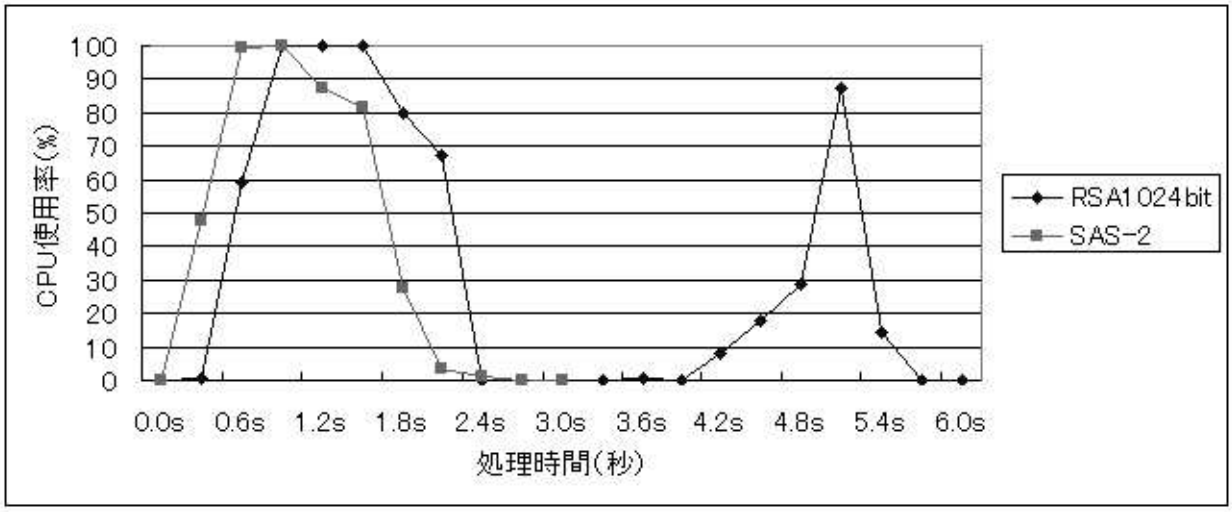

図 4.5 サーバにおける RSA(鍵長 1024bit) と SAS-2 との比較

鍵長 1024bit の RSA 及び SAS-2 の比較では，提案方式の方が処理速度が早く低負荷であった．1024bit の RSA においても，CPU 使用率の二度の上昇が確認できた．理由としては，512bit の RSA の場合と同様にクライアントの処理速度が低速である事が原因であると考えられる．

図 4.2 同様，双方の面積比を計算すると，提案方式が鍵長 1024bit の RSA を使用した場合に比べて約 32 %小さい事が分かった．

3. サーバにおける Diffie-Hellman と SAS-2 との比較

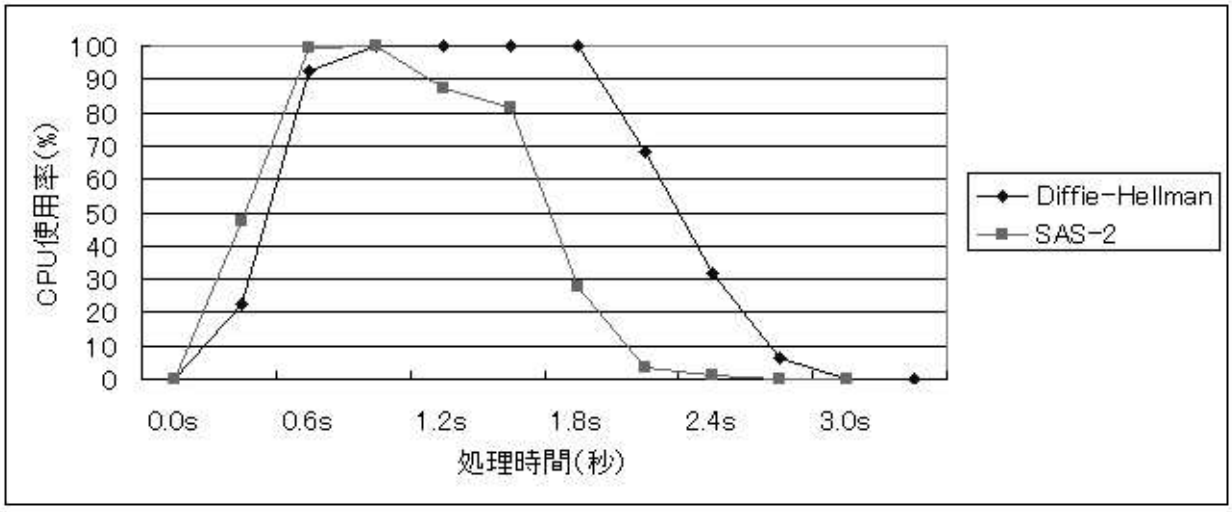

図 4.6　サーバにおける Diffie-Hellman と SAS-2 との比較

Diffie-Hellman 及び SAS-2 の比較では，提案方式の方が処理速度が早く低負荷であった．グラフの形状が SAS-2 と同形である点が，RSA とは異なっている．また，Diffie-Hellman では鍵共有のみであり認証は行っていない．その点を考慮すると，相互認証も行っている提案方式は，大幅に高速且つ低負荷であると言える．

双方の面積比を計算すると，提案方式が Diffie-Hellman に比べて約 28 %小さい事が分かった．

4. 低スペッククライアントにおける RSA(鍵長 512bit) と SAS-2 との比較

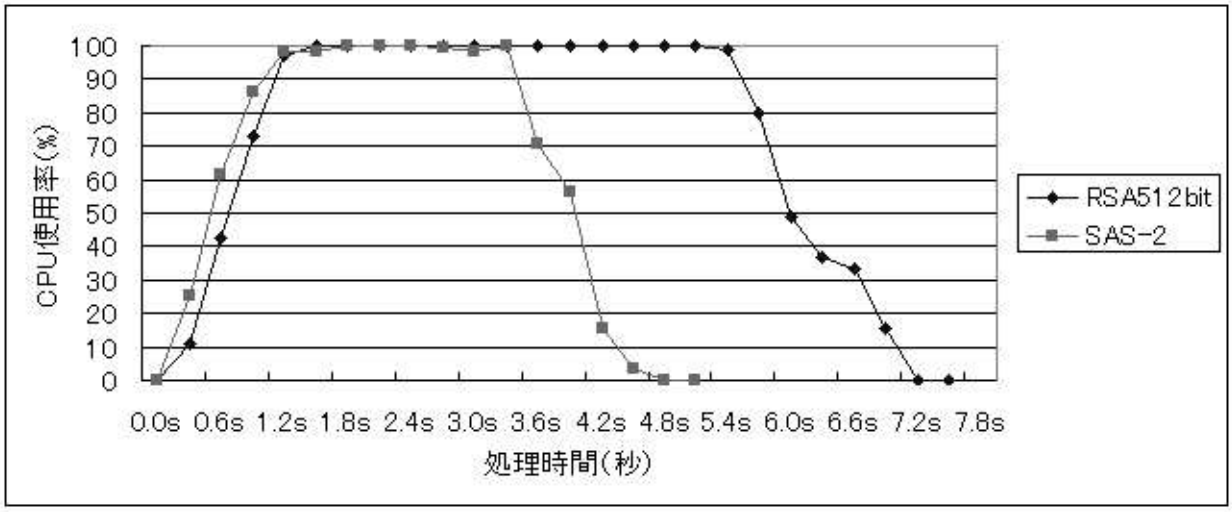

図 4.7　低スペッククライアントにおける RSA(鍵長 512bit) と SAS-2 との比較

鍵長 512bit の RSA 及び SAS-2 の比較では，提案方式の方が処理速度が早く低負荷であった．

グラフの形状についていうと，サーバ側での CPU 使用率での上昇率の大きな違いが見て取れる．低スペックのクライアントマシンでは，サーバに比べ CPU にかなりの負荷がかかり，それと同時に処理にも多くの時間を必要とする事が分かった．

双方の面積比を計算すると，提案方式が約 39 %も小さい事が分かり，提案方式の有用性が伺える．

5. 低スペッククライアントにおける RSA(鍵長 1024bit) と SAS-2 との比較

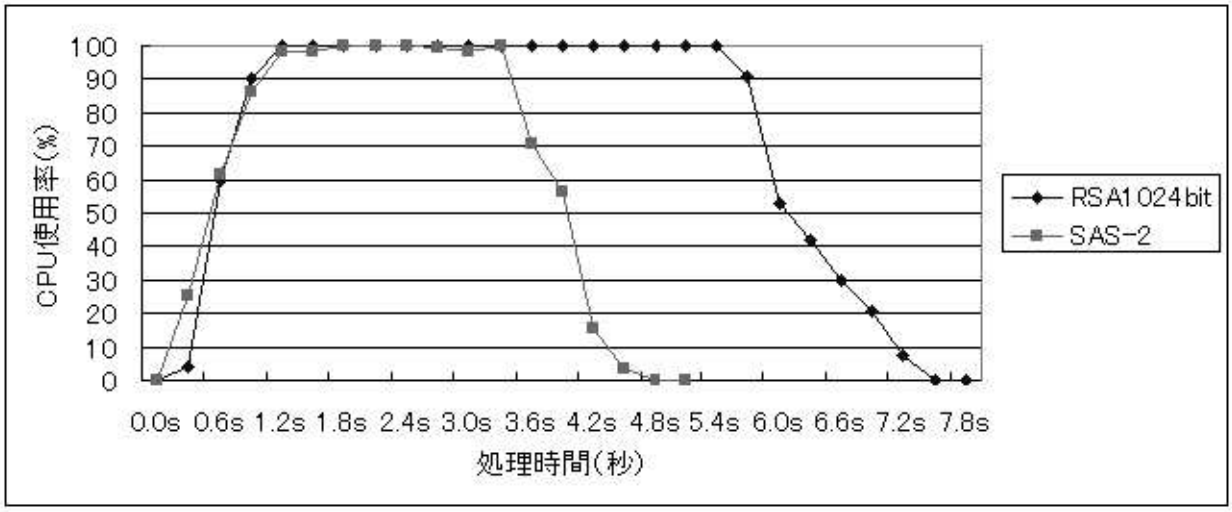

図 4.8 低スペッククライアントにおける RSA(鍵長 1024bit) と SAS-2 との比較

鍵長 1024bit の RSA 及び SAS-2 の比較では，提案方式の方が処理速度が早く低負荷であった．また鍵長が 512bit の場合に比べ，僅かではあるが CPU 及び処理に増加の傾向が見られた．どちらの鍵長も，低スペックマシンではかなりの負荷になる事が分かった．

これまで同様双方の面積比を計算すると，提案方式が約 41 %も小さい事が分かり，提案方式の有用性が伺える．

6. 低スペッククライアントにおける Diffie-Hellman と SAS-2 との比較

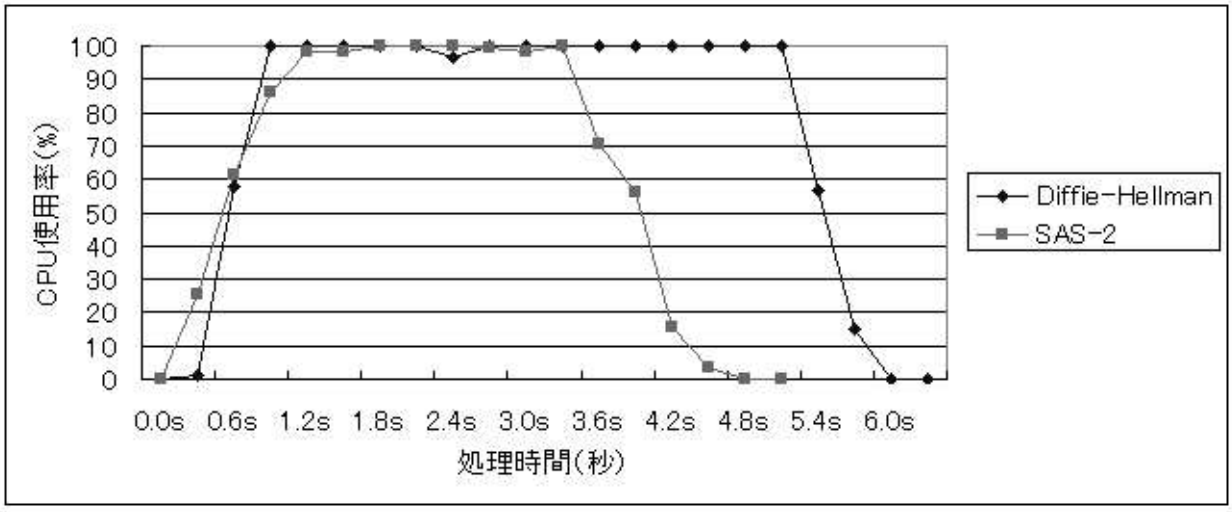

図 4.9 低スペッククライアントにおける Diffie-Hellman と SAS-2 との比較

Diffie-Hellman 及び SAS-2 の比較では，提案方式の方が処理速度が早く低負荷であった．RSA を用いた場合に比べると，Diffie-Hellman では，提案方式とのサーバ・クライアント双方における差の比率に大きな変化は見られなかった．また，サーバ同様 Diffie-Hellman では鍵共有のみを行い，認証は行っていない．その点を考慮するとクライアント側では特に，相互認証も行っている提案方式の有用性が分かる．

双方の面積比を計算すると，提案方式が Diffie-Hellman に比べて約 32 %小さい事が分かった．

7. サーバにおける総合比較

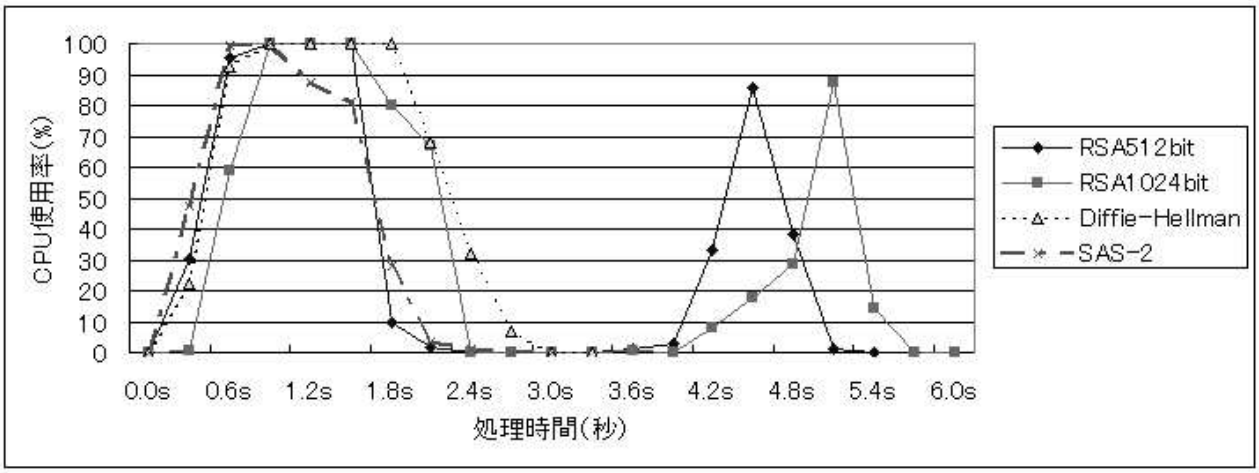

図 4.10　サーバにおける総合比較

サーバ側での全てのデータを同一のグラフに表示した．

マシンスペックが非常に高いため，全体的に低負荷且つ高速である．その中でも特に，提案方式のグラフが負荷・速度の両方で際立った違いを見せている．

RSA については，鍵長の違いによりマシンにかかる負荷や速度が若干異なる事が確認できる．また，クライアントとのスペック差による若干の遅延が見られるが，クライアントの高スペック化によって全体の処理時間は短縮可能である事が予想される．

8. 低スペッククライアントにおける総合比較

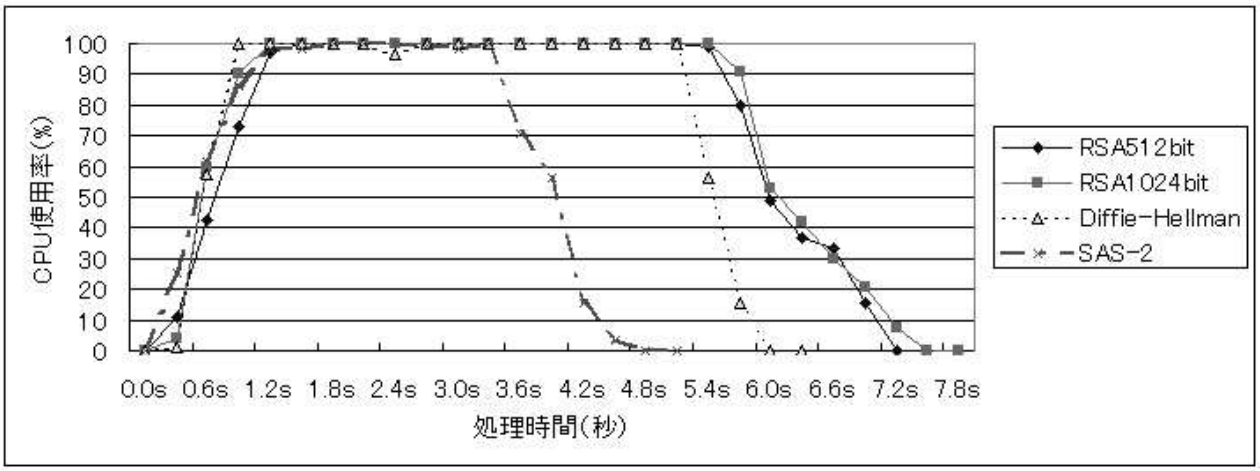

図 4.11　低スペッククライアントにおける総合比較

クライアント側での全てのデータを同一のグラフに表示した．

マシンスペックが低いため，サーバ側に比べると処理時間・負荷共に全体的に大きく増加している事が分かる．しかし，その中でもサーバ側同様，他の方式と提案方式との違いは顕著である．

9. 高スペッククライアントにおける RSA(鍵長 512bit) と SAS-2 との比較

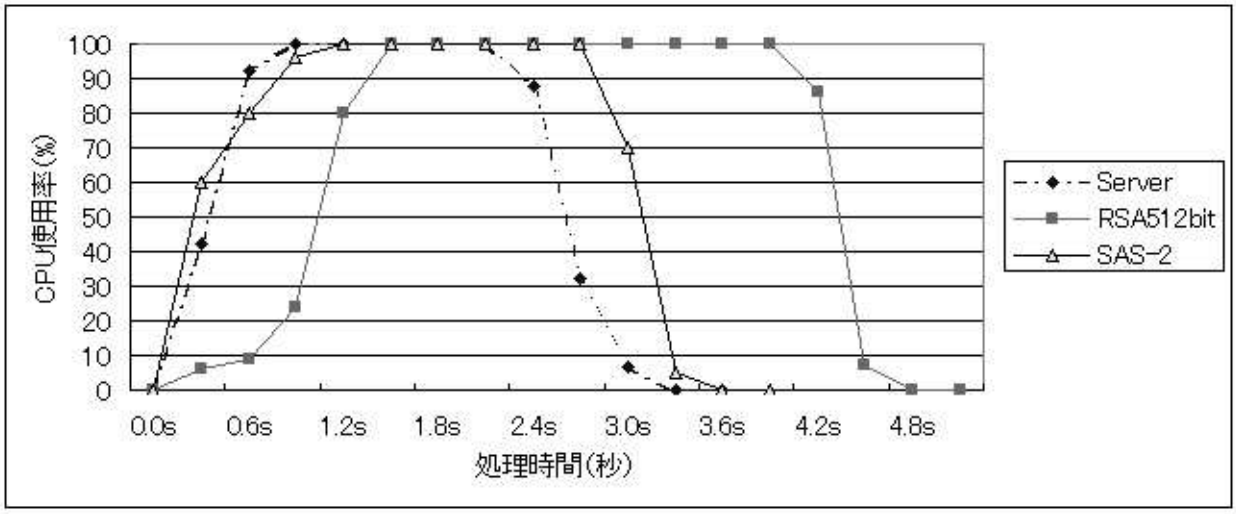

図 4.12　高スペッククライアントにおける RSA(鍵長 512bit) と SAS-2 との比較

この図は，クライアントに高スペックのマシンを用いた場合の，RSA 及び提案方式の比較グラフである．鍵長は 512bit を用いている．提案方式の方が処理速度が早く低負荷であったが，低スペックマシンと比較すると RSA も全体的に処理が大幅に向上している．

双方の面積比を計算すると，提案方式が約 20 %小さい事が分かった．また，提案方式・RSA の他に，サーバ側の RSA を用いた場合のグラフも記載している．クライアントに低スペックマシンを用いた場合とは異なり，スペックが高くなると遅延が起きていない事が分かる．また，高スペックマシン使用時の，サーバ側での提案方式使用時の負荷・処理速度も同様に計測したが，これについては低スペックマシン使用時の数値とほとんど変化が無かったため表示していない．この事から，サーバ側において，提案方式はクライアントマシンのスペックによる影響をほとんど受けないという事が分かった．

10. 高スペッククライアントにおける RSA(鍵長 1024bit) と SAS-2 との比較

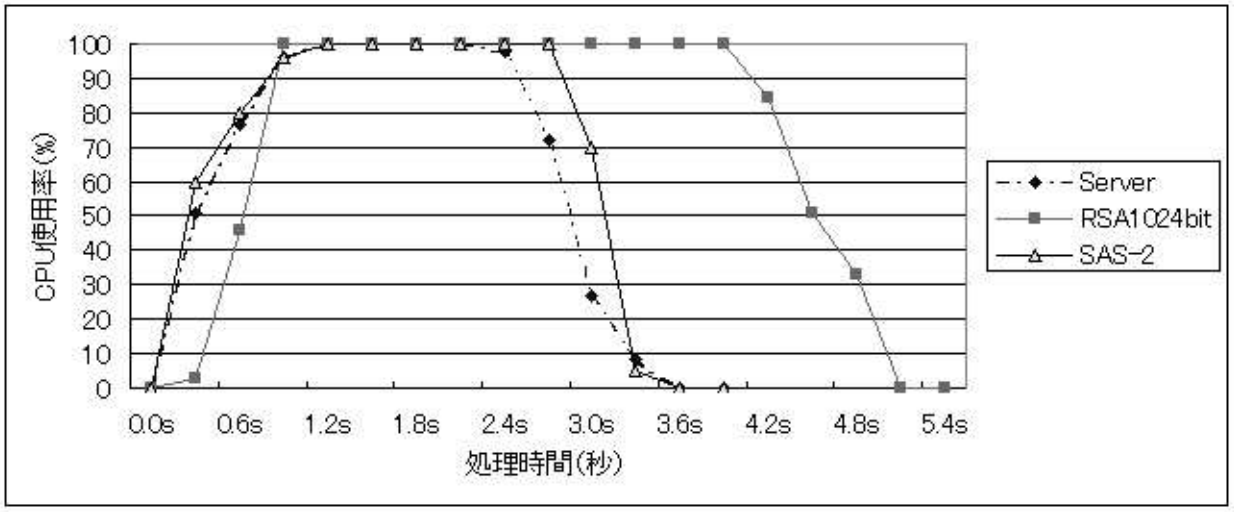

図 4.13 高スペッククライアントにおける RSA(鍵長 1024bit) と SAS-2 との比較

この図は，クライアントに高スペックのマシンを用いた場合の RSA 及び提案方式の比較グラフである．鍵長は 1024bit を用いている．提案方式の方が処理速度が早く低負荷であったが，低スペックマシンと比較すると 512bit を用いた場合と同様に処理が大幅に向上している．

双方の面積比を計算すると提案方式が約 30 %も小さく，鍵長によって大きな差が出たと言える．また，図 4.10 同様，提案方式・RSA の他にサーバ側の RSA を用いた場合のグラフも記載している．この図もやはり遅延が起きておらず，低スペックマシンとの通信時に比べ大幅に処理時間が減少している事が分かる．

11. 高スペッククライアントにおける Diffie-Hellman と SAS-2 との比較

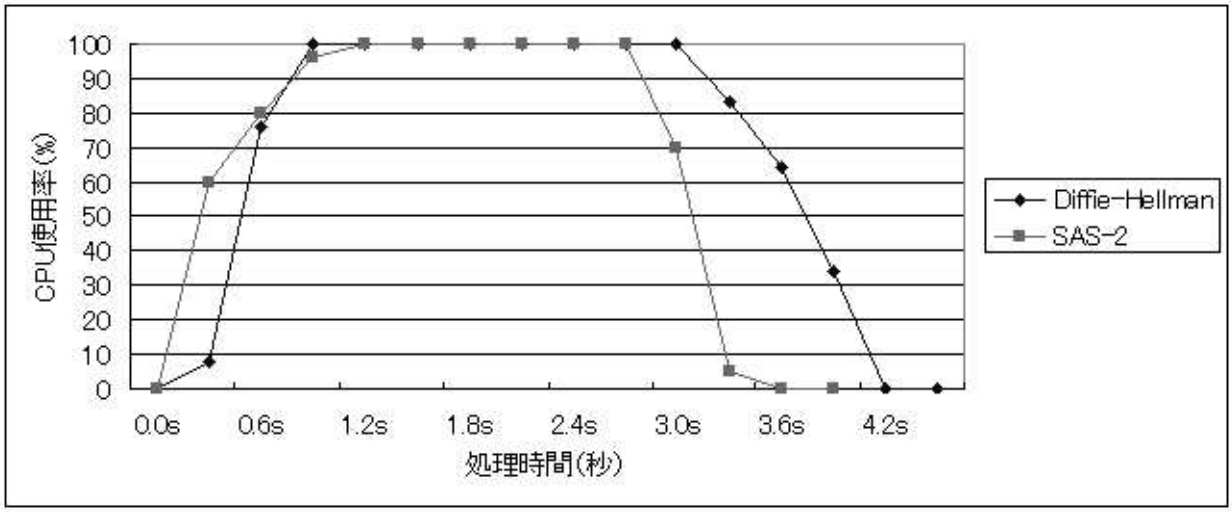

図 4.14　高スペッククライアントにおける Diffie-Hellman と SAS-2 との比較

この図は，クライアントに高スペックのマシンを用いた場合の，Diffie-Hellman 及び提案方式の比較グラフである．実験の結果，提案方式の方が処理速度が早く低負荷である事が分かったが，全体的に見ると低スペックマシンに比べ処理時間・負荷が大幅に減少している事が分かる．また，サーバ側の処理速度・及び負荷の計測も行ったが，Diffie-Hellman については全くといって良い程低スペック時の数値と変化は無かったため表示していない．このサーバ側の計測により，Diffie-Hellman についてもクライアントのスペックがほとんど影響しないという事が分かった．

双方の面積比を計算すると，提案方式が約 15 %小さかった．

12. 高スペッククライアントにおける総合比較

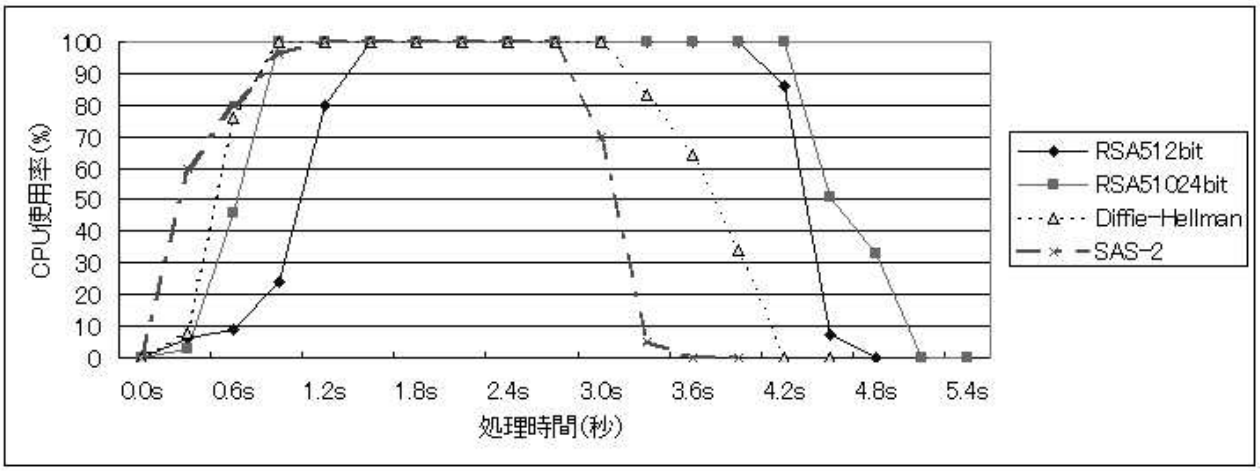

図 4.15　高スペッククライアントにおける総合比較

高スペッククライアントでの全てのデータを同一のグラフに表示した．

マシンスペックが大幅に向上したため，全てのグラフにおいて処理時間の短縮及び CPU 使用率の減少が確認できた．その中でも，提案方式は低スペックマシン使用時と同様に優れている事が分かった．また，クライアントマシンのスペックがサーバに及ぼす影響も確認する事が出来た．

RSA については，マシンスペックが低下するごとに提案方式との処理時間の差が広がっていく事が分かった．この事から，マシンスペックが低下すればするほど提案方式の有用性が増加する事が推測される．これは，特に携帯電話等のモバイル端末においての利用において期待出来ると考える．

13. 面積の総合比較

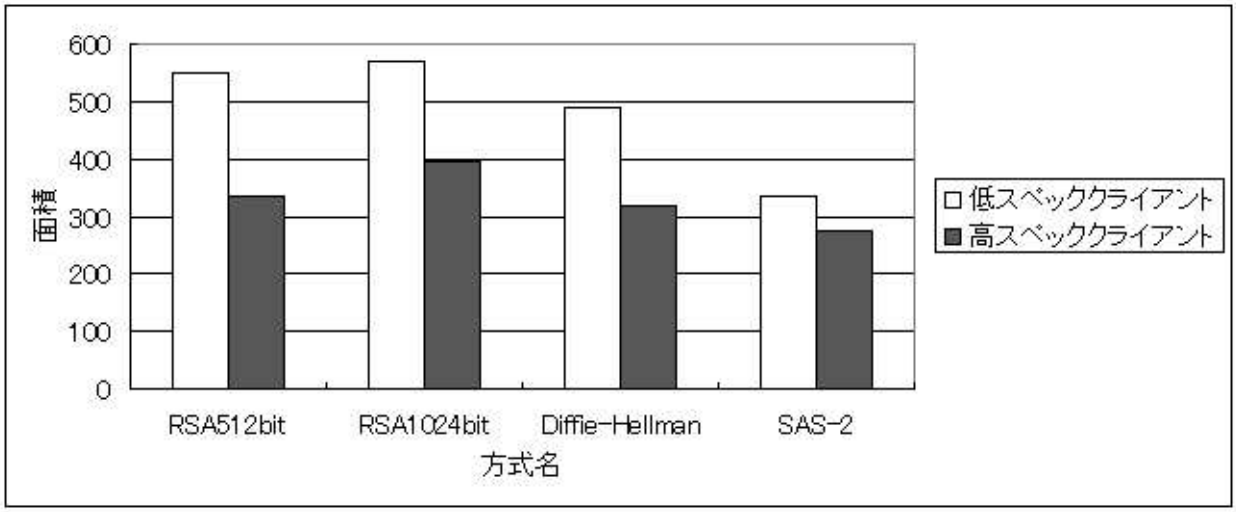

図 4.16　各方式における面積の総合比較

高スペッククライアント及び低スペッククライアント使用時の，各方式におけるグラフの面積を表示した．

CPU 使用率を高さ，応答時間を底辺とした場合の面積であるが，提案方式が高・低どちらのスペックのマシンにおいても小さく，優れている事が分かる．これまでのグラフ同様，特に低スペックマシンを使用した場合の差が大きく，CPU 負荷・処理時間の総合的な面で提案方式が有効である事がよく分かった．

第 5 章

今後の課題

本研究の実験により，SSL プロトコルと SAS-2 認証を用いた提案方式とを比較した場合，マシンに及ぼす負荷，処理速度ともに提案方式が優れている事が分かった．特に低スペックのマシンではその傾向が顕著に現れた．また，第 4.1.2 節でも述べたが，シミュレータを作成するに当たり SSL の各アルゴリズムは若干簡略化して構成している．それらの事を考慮すると，ブラウザ等で実際に比較を行った場合は，今回の実験結果以上に提案方式との差は広がるであろうと考える．他にも，本方式はクライアントのスペックがサーバ側へ殆ど影響を及ぼさないという事も分かり，これらの事から提案方式はスペックの低い，特に携帯電話等のモバイル端末においてその効果が期待出来ると推測する．これまで SSL では低スペックマシンにおける Web ブラウジングの質の劣化や，接続に時間を要するといった部分が問題点として指摘されてきたが，サーバは勿論，特にクライアント側の結果において，提案方式は SSL の抱える問題点を解決出来る技術であるという事が実証できた．また，SSL での認証部分に用いる証明書も必要とせず非常に高速な相互認証が行えるため，現在そのほとんどが一方向性の認証である SSL に比べるとより有効な認証を行っている事が分かる．

実験の結果から，提案方式は 1 クライアントからのアクセスについては，非常に高速な通信を行える技術である事が分かった．今後はより現実的な場面を想定し，同時に複数クライアントからのアクセスがあった場合における，サーバ・クライアント双方の評価実験を行っていく必要がある．また，今回はシミュレータという形式での実験を行ったが，実際にブラウザへ実装しての評価，さらには本提案方式が最も有用であると考える，携帯電話等のモバイル端末に適用しての実験を行って行きたいと考えている．その上で，さらに様々な視点での評価・検証を行っていく必要がある．

第 6 章

むすび

本論文では，従来技術である SSL プロトコルの問題点について考察し，SAS-2 認証方式を用いた新しいプロトコルを提案した．この方式は，SSL プロトコルに比べ高速に共通鍵共有が行える事が分かった．また，証明書を用いて行う SSL での認証に比べ，低スペックマシンにおいても安全且つ高速な相互認証を行う事が出来た．

今後は，前章に述べた問題について様々な見地から評価していく必要がある．

謝辞

高知工科大学情報システム工学科清水明宏教授には，生活指導，就職活動，卒業研究等，様々な面にわたって貴重な御教示・御助力を賜りここに深く感謝申し上げる．

また，本研究室院生である辻貴介氏，大垣文誉氏，上岡隆氏，河村智氏，ならびに本研究室学部生，池上奈津子氏，岸田光生氏，田岡慎也氏，永井慎太郎氏，奈良裕介氏，福冨英次氏，そして在学中に様々な御助言，御指導を頂いた諸先生方に心から感謝する．

参考文献

[1] T. Kamioka, A. Shimizu, “The Examination of the Security of SAS one-time password authentication,” IEICE Technical Report, OFS2001-48, Vol.101, No.435, pp.53-58, 2001.

[2] T. Tsuji, T. Kamioka, and A. Shimizu, “Simple and secure password authentication protocol, ver.2 (SAS-2),” IEICE Technical Report, OIS2002-30, vol.102, No.314, pp.7-11, September 2002.

[3] NBS, “Data Encryption Standard,” FIPS-PUB-45, 1977.

[4] Lai, X. , Massey, L. and Murphy, S. “Markov Cipher and Differential Cryptoanalysis,” Proc. of EuroCrypto’91, LNCS547, Springer-Verlag, pp.17-38, 1991.

[5] 岡本龍明, 山本博資, “現代暗号,” 産業図書, 1997.

[6] Diffie, W. and Hellman, M. “New Directions in Cryptography,” IEEE Trand. on Information Theory, IT-22, 6, pp.644-654, 1976.

[7] R. Rivest, “The MD5 message-digest algorithm,” Internet Request For Comments 1321, April 1992.

[8] National Institute of Standards and Technology, “Secure hash standard,” FIPS Publication 180-1, April 1995.