平成 13 年度

学士学位論文

SAS Proxy ネットワークの導入とその評価

The introduction of SAS Proxy network and its evaluation

1010365　　伊藤　哲

指導教員　　清水 明宏

2001 年 2 月 8 日

高知工科大学 情報システム工学科

要 旨

SAS Proxy ネットワークの導入とその評価

伊藤　哲

安全で高速な http 通信を実現する新しいシステムを提案し、従来技術との比較を行う．高速な通信を確立するために SAS 認証方式を用い, 暗号化復号処理をクライアントではなくローカルエリア内に新たに設置する Proxy サーバ(代理サーバ)に代行させる．Proxy サーバを Java アプリケーションにて構築することにより，安価なシステム導入を可能にする．

本論文では，既存技術の問題点を解決した SAS Proxy を提案し，安全な http 通信を行うプロトコルとして最も普及している https との比較評価実験を行い，その有用性を検討する．

キーワード　　SAS，**認証**，Proxy **サーバ**，Java **アプリケーション**，https

Abstract

The introduction of SAS Proxy network and its evaluation

Satoshi Ito

I propose to a new system that realizes security and speedy communication of http, and compare with existing method. Proxy server in local area network enable to communicate speedily with an authentication system of SAS, and perform an encryption transaction in place of client. I construct Proxy server from Java application to introduce the system inexpensively.

In this paper, I propose to SAS Proxy system solving the existing matter, I evaluate and compare with https that is a most popular protocol.

key words SAS, authentication, Proxy server, Java application, https

目次

図目次

表目次

第 1 章

はじめに

インターネットは, その誕生以後爆発的に普及し, 様々なサービスが提供されている. 代表的なものとして, インターネットを用いたオンラインショッピングや電子商取引がある. これらのサービスにおいては, ユーザの要求が確実に送信先へ配送される必要がある. また, 取引を行う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など情報は, 悪意の第三者に悪用される可能性があるため, 特に安全に配送される必要がある. 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, ユーザ認証に SSL 認証方式を用いた https や, 送受信データをパケットレベルで暗号化し仮想専用回線を用いてデータ送信を行う VPN がある.

これら現行の方式には, ユーザのクライアントマシンごとに https に対応したブラウザが必要・暗号化復号処理を行う専用のハードウェアが高価であるなど, 問題点が多く指摘されている.

本研究はそれら問題点を解決した上で, より高速で安全な通信を実現するため, ユーザの認証に SAS 認証方式を用い, ユーザ認証情報の生成とリクエストデータ・レスポンスデータの暗号化復号処理をクライアントに代わり行う Proxy サーバ（代理サーバ）を設けた新しいシステム・SAS Proxy を提案する. さらに現在最も普及している, 安全な http 通信を可能にする方式 https 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 2 章で研究背景として前述の https と VPN の特徴とそれらが抱える問題点を述べ，第 3 章ではその問題点を解決する新たなシステムとして SAS Proxy を提案し，その機能とアルゴリズムについて述べる．第 4 章では提案した方式と https との比較評価実験を行い，主にクライアント視点からその有効性を評価する．さらに，第 5 章で今後の課題を述べ，第 6 章でむすびとする．

第 2 章

研究背景

本章では，従来からあるユーザの認証に SSL 認証方式を用いた https と仮想専用回線を用いてデータを送受信する VPN の特徴を挙げ，それらの抱える問題点を明らかにする．

2.1 https の特徴と問題点

https は，http 通信を行う際に安全な通信を確立するためユーザの認証を行う方式であり，ユーザの認証には SSL（Secure Socket Layer）認証方式を用いる．通信手順は，ユーザの ID・パスワードを DES や RC4 などの共通鍵暗号方式を用いて暗号化し，生成された共通鍵を RSA や DSS などの公開鍵暗号方式を用いて暗号化，ユーザ認証情報と公開鍵を，要求するコンテンツを持つ web サーバに送信する．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はデータをメッセージダイジェスト関数（MD5 や SHA-1 など）に通して得られる MAC（Message Authentication Codes）と呼ばれる数値を比較することで確認する．

ユーザ認証情報の生成・デジタル署名書の受信などの処理は www ブラウザが行う

が，これはサービスを受ける全てのユーザの環境に対応していなければならない．各ユーザの異なるマシン・OS 毎に https に対応した www ブラウザを用意する必要があり，複数のマシンを有するユーザは同一のサービスを享受する場合でも一元的な管理を行うことができない．これが第一の問題点である．さらに，SSL 認証処理は公開鍵暗号方式を用いておりここで浮動小数点演算など複雑な計算を行うため，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．これが第二の問題点である．

以下の図 2.1 に，https のシステムを示す

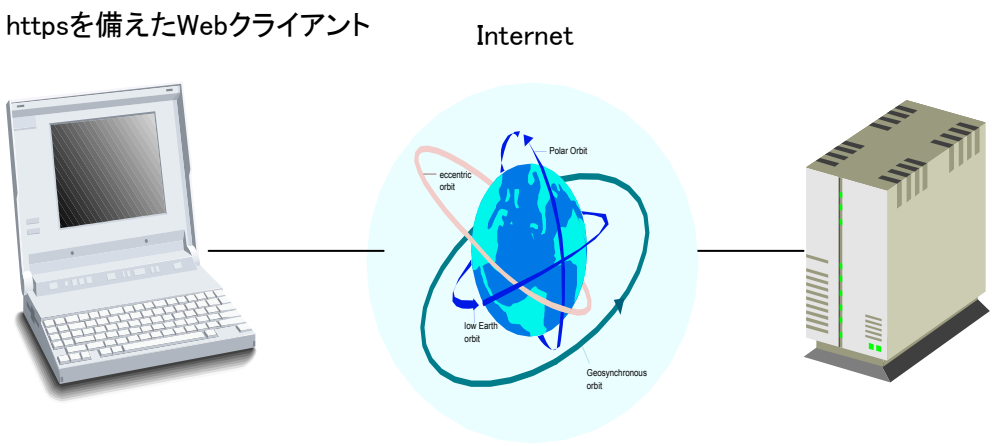

図 2.1　https システム構成

2.2 VPN の特徴と問題点

VPN（Virtual Private Network）は複数の LAN を暗号化して接続し，仮想的に専用回線などで直接接続しているようにする技術である．通信手順は，サーバ・クライアント間で送受信されるデータに IP パケットレベルで新たなヘッダを付与し暗号化（カプセル化）を行い，暗号化を施した上で仮想的な専用回線（トンネル）を用いてデータを送信する．ネットワークを増やすごとに仮想施設間回線を 1 対 1 で張り，リクエストデータ・レスポンスデータの暗号化復号処理をゲートと呼ばれる専用のハードウェアで行う．施設内の全てのマシンは個々に暗号化復号処理を行うことなく，安全な通信を行うことができる．さらに本来ならインターネットを経由できないプライベートアドレスの通信，TCP/IP 以外の通信も可能となる．

施設をつなぐネットワークは常に 1 対 1 で張られるため，新たな施設との接続を 1 つ増やすたびに暗号化復号処理を行う専用ハードウェアが 2 つ必要になり仮想専用回線を設けるための帯域の確保も必要となる．イントラネットなどの限られたユーザ内での利用には向いているが，www のような世界中と通信を行うサービスには不向きである．これが第一の問題点である．さらに，クライアントからのリクエストデータ・レスポンスデータの暗号化復号処理を行う専用ハードウェアは，非常に高価でありコストがかかってしまう．これが第二の問題点である．

以下の図 2.2 に，VPN のシステムを示す

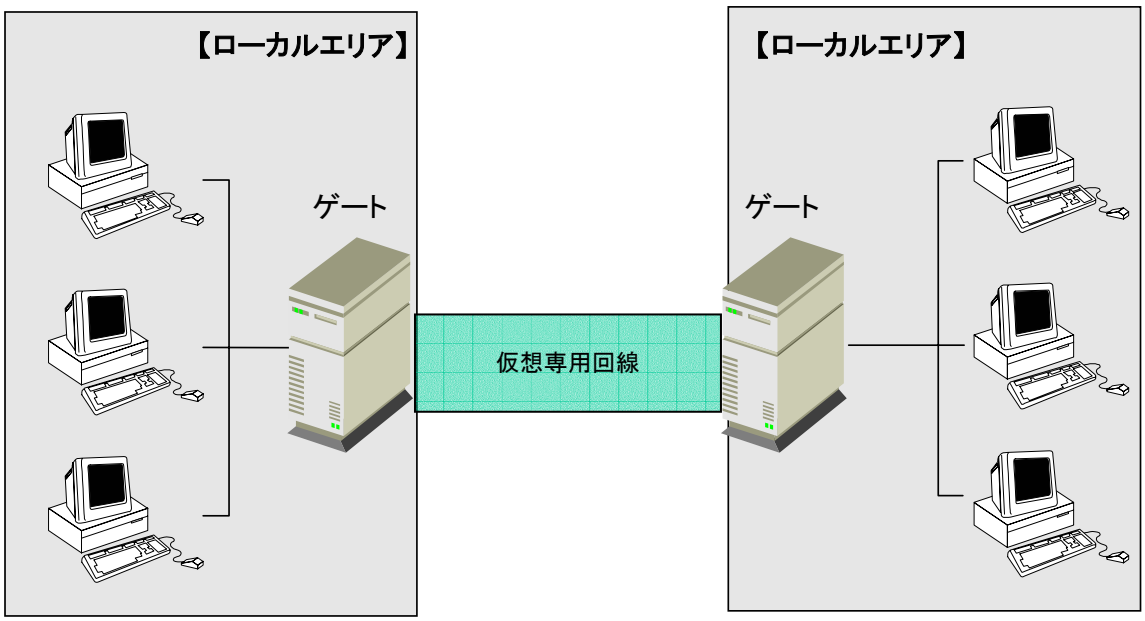

図 2.2　VPN システム構成

第 3 章

SAS Proxy の提案

本章では，前章で述べた https や VPN の利点は生かしつつ，列挙した問題点を解決した新たなシステム・SAS Proxy を提案し，その概要を示す．

3.1 システムの概要

SAS Proxy は https や VPN の持っていた問題点を以下の構成により解決する．

（ｉ）ユーザの認証に SAS 認証方式を用いる

前述の https にて使われていた SSL 認証方式に代わり，ユーザの認証に SAS 認証方式を用いる．これは，SSL 認証方式がユーザ認証情報を暗号化配送する際に公開鍵暗号方式を用い，送信先の公開鍵の信頼性をデジタル署名を受け取ることにより確認するという処理手順を踏んでいるのに対し，SAS 認証方式はワンタイムパスワードを用いて認証を行っている．このため，SAS 認証方式は SSL 認証方式に比べ処理手順・演算の点で簡素であり，処理の高速化が期待できる．

（ⅱ）暗号化復号処理を代理サーバにて行う

web クライアントが存在するローカルエリアネットワーク上に，web クライアン

トに代わりリクエストデータの暗号化処理を行う Proxy サーバ（代理サーバ）を設け，これを SAS Proxy サーバとする．リクエストのあったコンテンツを配信する web サーバに，ユーザの認証処理・リクエストデータの復号を行う認証サーバを設ける．web クライアントはユーザ認証情報の生成・リクエストデータの暗号化・レスポンスデータの復号処理を行う必要がなくなり，通常の http 通信と同様の処理量でブラウジングを行うことを可能とする．

（iii）代理サーバを Java アプリケーションにて作成

SAS Proxy サーバを Java アプリケーションにて作成することにより，特別なサーバ専用マシンを用意することなく，さらに VPN のような特別なハードウェアに依存することなく web クライアントに代わり暗号化復号処理を行う Proxy サーバを構築する

3.2　システムの構成

3.2.1　システムの構成

以下に，各構成要素の概要を述べる

（ｉ）web クライアント

ユーザの入力操作を直接受け付ける端末

（ii）SAS Proxy サーバ

web クライアントと同一のローカルエリアネットワーク上に配置され，ユーザ認証・暗号化の処理を代理で行う．web クライアントからユーザ情報を受け取り認証情報を生成･リクエストデータを受け取り暗号化しリクエスト先のサー

バの送信・web サーバからのレスポンスデータを復号し web クライアントに返す.

（iii）認証サーバ

SAS Proxy サーバを介して web クライアントから送られてくるリクエストデータを復号し，リクエストされたデータを暗号化して web クライアント側に返す．ユーザ認証情報の管理も行う．誤ったパスワードで暗号化されたリクエストデータは正常に復号されないため, 正当なユーザかどうかの判定を行うことが可能である.

以下の図 3.2.1 に，本システムのシステム構成図を示す

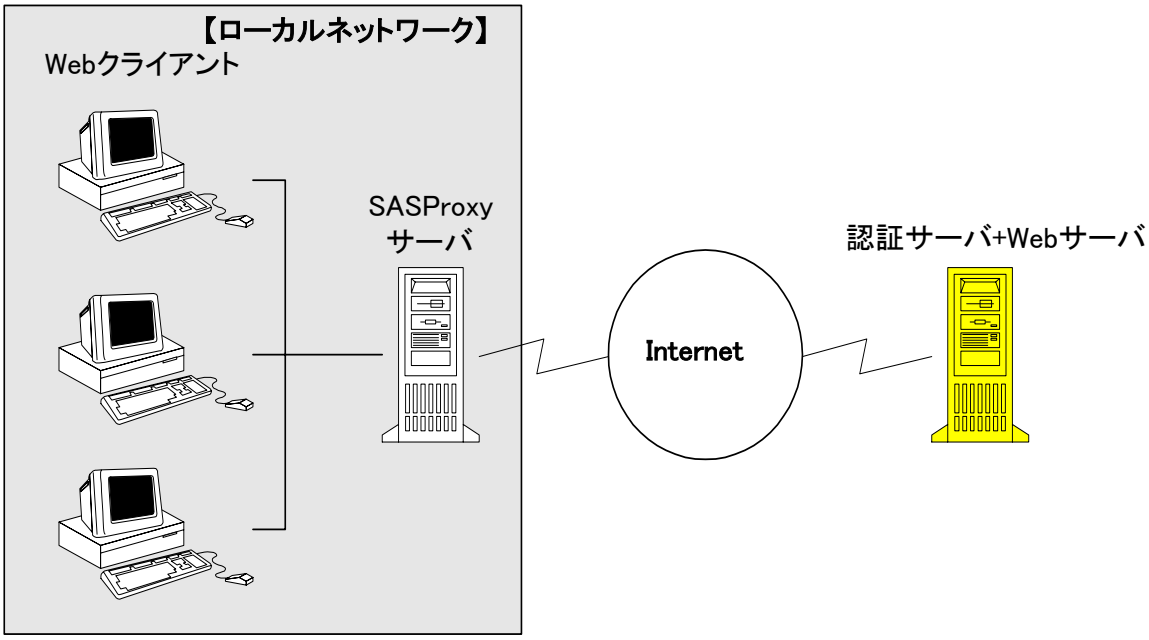

図 3.2.1　SAS Proxy システム構成図

3.2.2 動作環境

以下の表に各構成要素の動作環境を示す

（ｉ）web クライアント

H/W 要件	下記ブラウザの動作する環境	
S/W 要件	ＯＳ	Microsoft Windows 98 Second Edition を推奨.
	WWW Browser	Internet Explorer 5.5 Service Pack 1 以降，または Netscape Navigator6 以降

表 2.2.2.1　Web クライアントの動作環境

（ⅱ）SAS Proxy サーバ

H/W 要件	機種	Sun Ultra10
	CPU	400Mhz 以上
	Memory	512MB 以上
	Hard Disk 容量	5GB 以上
S/W 要件	OS	Solaris2.6 以降
	Java Runtime	Java2 Runtime 1.3.0 以降

表 2.2.2.2　SAS Proxy サーバの動作環境

（iii）認証サーバ

H/W 要件	機種	Sun　Enterprise 450
	CPU	400MHz 以上
	Memory	512MB 以上
	Hard Disk 容量	10GB 以上
S/W 要件	OS	Solaris2.6 以降
	Web Server	Apache 1.3.12 以降
	Servlet Engine	Tomcat 3.2.1 以降
	Database	Oracle8 R8.0.5
	Java Runtime	Java2 Runtime 1.3.0 以降

表 2.2.2.3　認証サーバの動作環境

（iv）ネットワーク環境

TCP/IP 群をサポートするものとする．（HTTP は必須）

3.3　機能概要

本システムは SAS 認証方式を http 通信上に実装することにより，認証機能を備えた暗号化通信を実現する．その概要を以下に述べる．

3.3.1　ユーザ認証と暗号化の流れ

ユーザ認証と暗号化の流れを以下に述べる

（1）web クライアント（ブラウザ）からのリクエストデータを SAS Proxy

サーバが受け取る．

（２）SAS Proxy サーバは認証サーバと http を用いた通信を行い，ユーザの認証回数，セッション ID などの取得，及びそれらの情報とユーザのパスワードなどを用いて暗号化鍵の計算を行う．

（３）SAS Proxy サーバは（２）で計算した暗号化鍵を用いて Web クライアントから受け取ったリクエストデータを暗号化し，認証サーバに送信する．

（４）認証サーバは登録されているユーザ情報から復号のための鍵を計算し，受信したリクエストデータを復号する．ここで同時に正しく復号できたかどうかの検証を行う．パスワードが不正である場合は正しく復号できないためこの時点でエラー（認証失敗）となる．

（５）正常にリクエストデータを復号できた場合，認証サーバは復号したリクエストデータを用いて www サーバとの通信を行い，www サーバからのレスポンスデータを復号の際に用いた鍵で暗号化して，レスポンスデータとして SAS Proxy サーバに送信する．

（６）SAS Proxy サーバは，受け取ったレスポンスデータを（２）で計算した鍵を用いて復号する．この際，レスポンスデータが正しく復号されたかどうかの検証を行う．正しく復号されなかった場合は，ネットワーク上でデータの改竄が行われた可能性が考えられるため，処理をやり直す．

（７）正常にレスポンスデータを復号できた場合，SAS Proxy サーバは，復号されたレスポンスデータを web クライアントに対して送信する．正常に復号されなかった場合はクライアントに対してデータを返さず，エラーを返す．

以下の図 3.3.1.1 に，ユーザ認証と暗号化の流れを示す

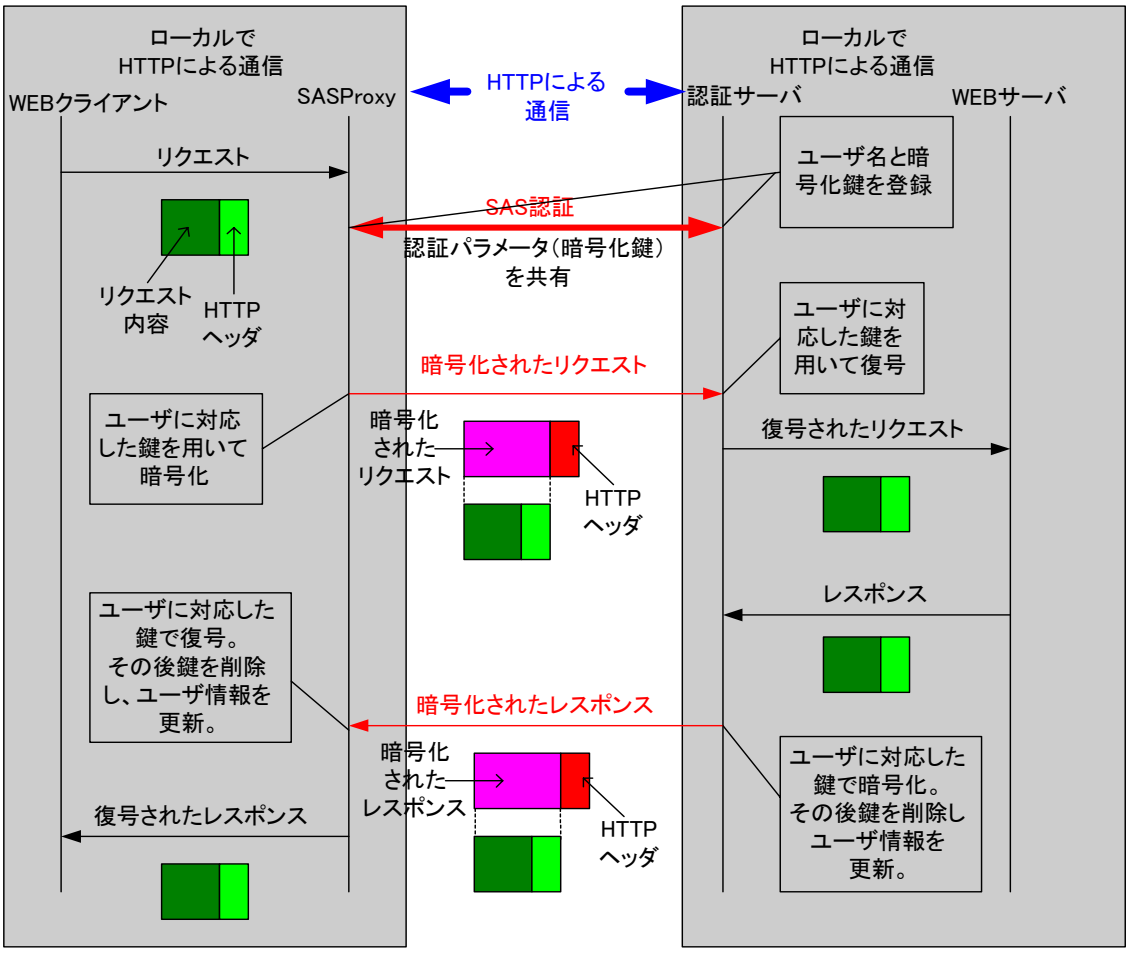

図 3.3.1.1　ユーザ認証と暗号化の流れ

以下に，本システムで用いる SAS 認証の処理シーケンスを示す

（1）ユーザ（被認証者）は，自身のユーザ名（U）を入力し，認証サーバに送信する．

（2）サーバ（認証者）は，期限付きのセッション番号（SID）を生成し，当該ユーザの認証履歴から現在のパスワードでの認証成立回数（n）を取得し，ま

た，前回の認証時に生成された認証パラメータ（A_n^2）を取得し，SID と n をクライアントに送信する．

（３）被認証者は，SID と n の排他的論理和と U から認証パラメータ X_n を，SID と n+1 の排他的論理和と U から認証パラメータ X_{n+1} を導出する．

（４）先の過程で導出した X_n と U から X_n^2 を，X_{n+1} と U から X_{n+1}^2 をそれぞれ導出する．

（５）先の過程で導出した X_n と X_n^2 との排他的論理和から Y_n^2 を，X_n^2 と X_{n+1}^2 との排他的論理和から Z_{n+1} をそれぞれ導出する．

（６）認証者側で取得済みの A_n とクライアントから送信された各パラメータ(SID, Y_n）を元に整合性がとれれば，認証成立とする．

（７）認証が成立した場合，次回認証パラメータ（A_{n+1}）を導出し，保存する．また，認証成立回数は，+1 したものを保存する．

セッション ID はステートレスなプロトコルである http 上でセッション管理を行うために使用されるだけでなく，毎回異なるセッション ID を使用することで，リトライ攻撃を防止し，安全性を高めることが可能となる．従って，（７）の時点で認証の成否に関わらず期限切れとし, セッション情報を削除すること理想である.

以下の図 3.3.1.2 に，本システムで用いる SAS 認証のシーケンス図を示す

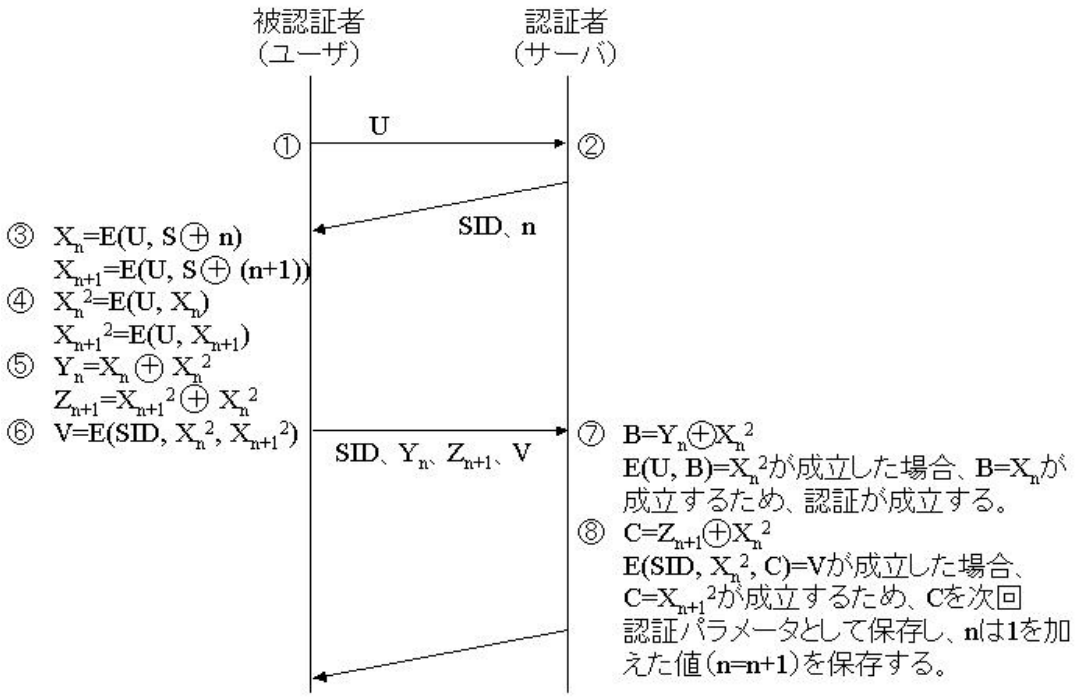

図 3.3.1.2　SAS 認証の処理シーケンス

第 4 章

https との比較評価実験

本章では, 前章で提案した新たなシステム・SAS Proxy と現在最も普及している https を同様の環境で比較し，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.

4.1 実験概要

4.1.1 実験目的

今回提案した SAS Proxy と現在最も普及している https を比較し，SAS Proxy の有用性を評価検討する.

評価基準は，主にクライアントの視点からとし，評価項目を以下のようにする.

（ⅰ） 認証サーバの CPU 使用率

（ⅱ） 認証サーバの応答時間

（ⅲ） クライアントの CPU 使用率

（ⅳ） クライアントの応答時間

（ⅱ）の応答時間とは，クライアント側からのリクエストを受けユーザ認証・リク

エストデータの復号を行い，要求されたデータを暗号化し，レスポンスデータとして送信するまでの時間を指し，（iv）の応答時間とは，サーバ側にリクエストを送り，レスポンスデータが配信され，web コンテンツが表示終了するまでの時間を指す.

4.1.2　実験構成

以下の図 4.1.2 に，比較評価実験のシステム構成図を示す

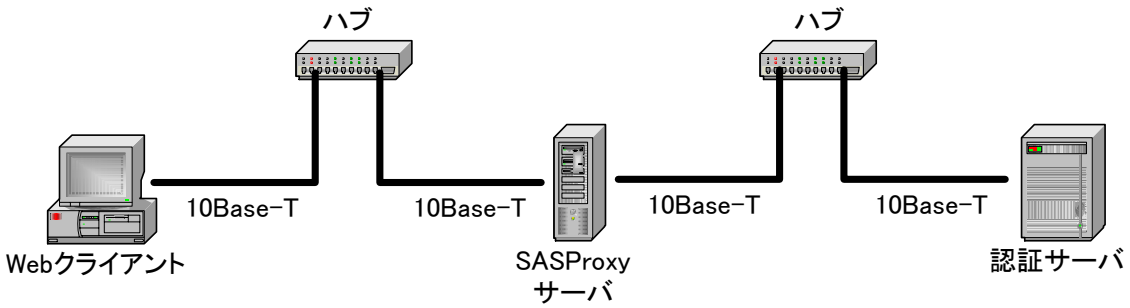

図 4.1.2　比較評価実験システム構成図

以下に，各構成要素の概要を述べる

（ｉ）web クライアント

ユーザの入力操作を直接受け付ける端末. 実験では比較的高スペックのマシンと低スペックのマシンの二種類を用意する.

（ii）SAS Proxy サーバ

web クライアントと同一のローカルエリアネットワーク上に配置され，ユーザ認証情報生成・暗号化の処理を代理で行う.

（iii）認証サーバ

SAS Proxy サーバを介して web クライアントから送られてくるリクエストデータを復号し，リクエストされたデータを暗号化して web クライアント側に返す．ユーザ認証情報の管理も行う．なお，認証サーバと web サーバを同じマシンで起動するため，以後これら二つをまとめて認証サーバとする．

（iv）10Base－T ケーブル

10Base－T，カテゴリー5 のケーブル．

（ｖ）Hub

10/100Base－T のスイッチングハブ．

本システムは, ネットワークの状況により性能値が大きく変動する. したがって, 他のトラフィックの影響をほとんど考慮する必要のない, 以上の環境における性能を計測値として設定する. また, TCP/IP 以外のプロトコルは流さないようにする.（NetBEUI，Apple Talk など）

4.1.3 **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

（ｉ）web クライアント

高スペックマシン

H/W 要件	機種	PC/AT 互換機
	CPU	Celeron 400Mhz
	Memory	128MB
	Hard Disk 容量	10GB
S/W 要件	OS	Microsoft Windows 98 SE
	WWW Browser	Internet Explorer 5.5 SP1

表 4.1.3.1　高スペックマシンの実験時の動作環境

低スペックマシン

H/W 要件	機種	PC/AT 互換機
	CPU	Pentium 100Mhz
	Memory	32MB
	Hard Disk 容量	1GB
S/W 要件	OS	Microsoft Windows 98 SE
	WWW Browser	Internet Explorer 5.5 SP1

表 4.1.3.2　低スペックマシンの実験時の動作環境

（ⅱ）SAS Proxy サーバ

H/W 要件	機種	PC/AT 互換機
	CPU	PentiumⅢ 650Mhz
	Memory	256MB
	Hard Disk 容量	10GB
S/W 要件	OS	Kondara Linux
	Java Runtime	Java2 Runtime 1.3.0 以降

表 4.1.3.3　SAS Proxy サーバの実験時の動作環境

（ⅲ）認証サーバ

H/W 要件	機種	Sun　Enterprise 400
	CPU	Ultra Sparc 400Mhz
	Memory	512MB
	Hard Disk 容量	20GB
S/W 要件	OS	Solaris2.6
	Web Server	Apache 1.3.12
	Servlet Engine	Tomcat 3.2.1
	Database	Postgre SQL 7.0.2
	Java Runtime	Java2 Runtime 1.3.0

表 4.1.3.4　認証サーバの実験時の動作環境

（iv）ネットワーク環境

TCP/IP 群をサポートするものとする.（http と https は必須）

4.2 実験結果

高スペック低スペックのそれぞれのマシンから web サーバに対し 10Kb のテキストベースの web コンテンツを要求し, その時のクライアントマシンの CPU 使用率, 応答速度, 認証サーバの CPU 使用率, 応答速度を計測した. 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）高スペックマシンにおける SAS Proxy と https との比較

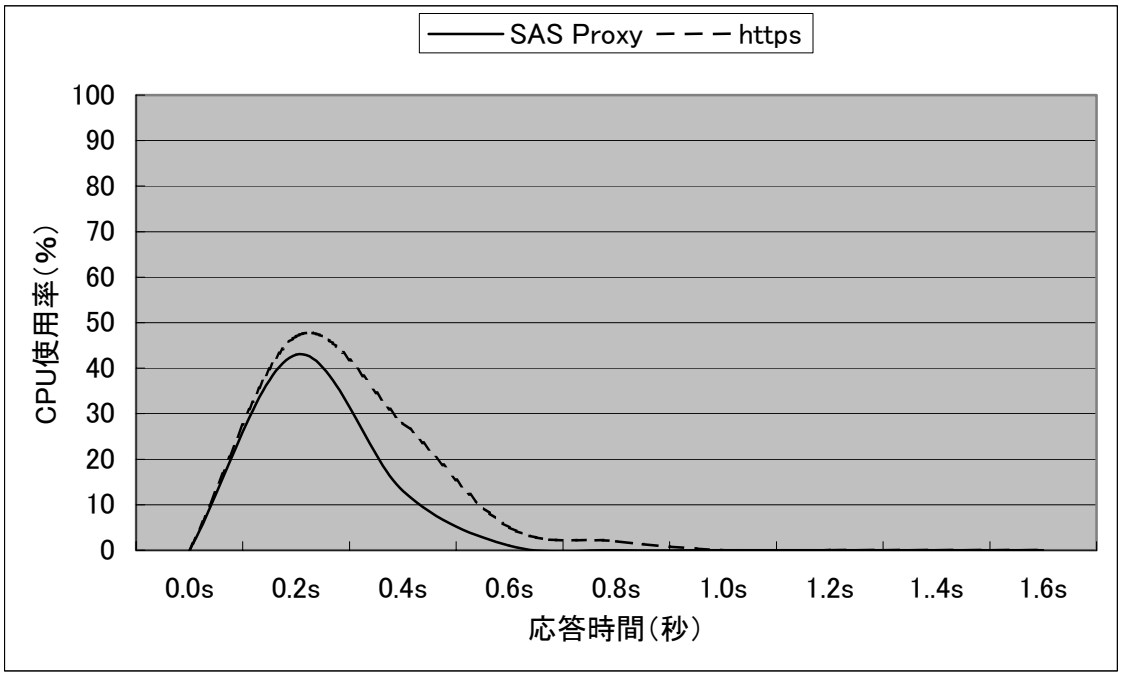

図 4.2.1.1　高スペックマシンにおける SAS Proxy と https との比較

高スペックマシンの比較では，CPU 使用率において SAS Proxy は応答時間が短く低負荷であった．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，SAS Proxy が https に比べ 25%小さいことが分かり，一つの目安として 25%動作が軽快であるといえる．

（2）低スペックマシンにおける SAS Proxy と https との比較

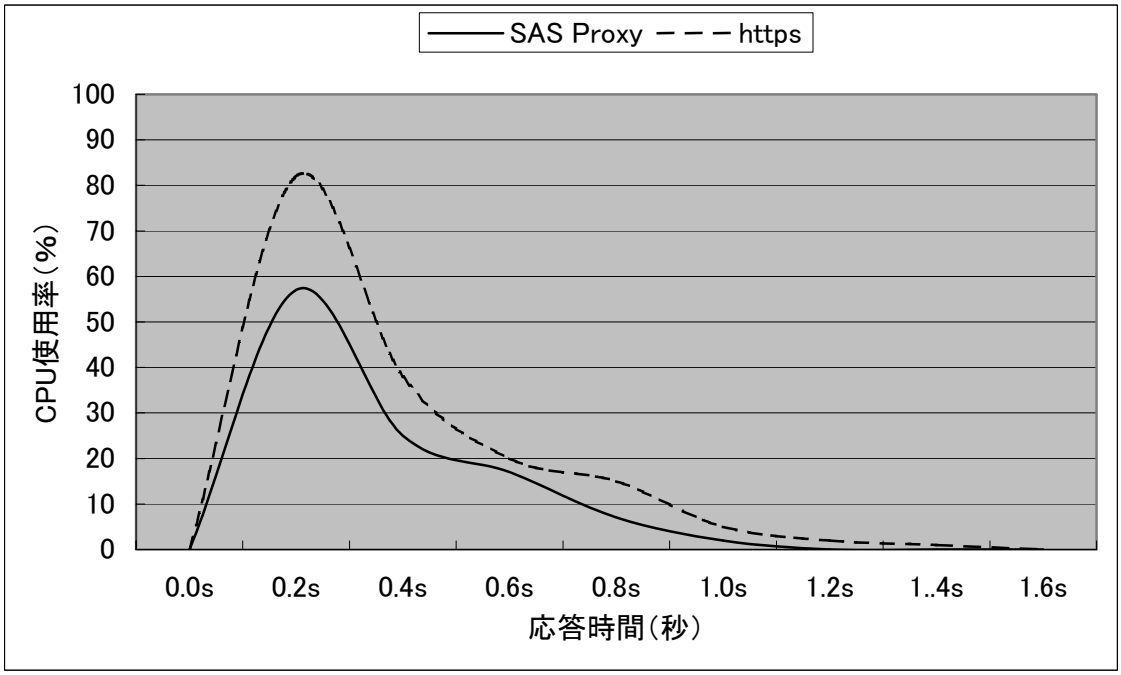

図 4.2.1.2　低スペックマシンにおける SAS Proxy と https との比較

低スペックマシンの比較では, CPU 使用率において SAS Proxy が https に比べ, 大幅に低負荷であることが分かる．https のユーザ認証情報生成処理が低スペックマシンにとっては，大きな負荷となっていることが分かる．さらに，応答時間においても SAS Proxy が https に比べ 0.4 秒速かった．（1）と同様，面積比を計算すると SAS Proxy が https に比べ 47.1%小さいことが分かった．

（3）高スペックマシンからのリクエストによる認証サーバでの比較

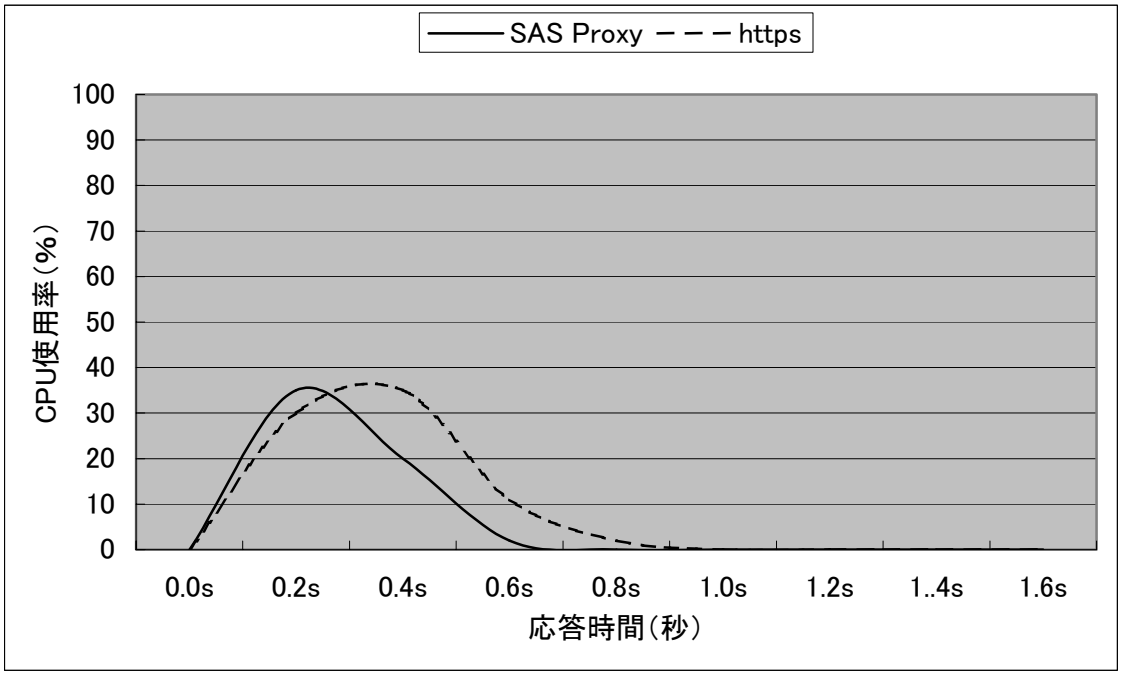

図 4.2.1.3　高スペックマシンからのリクエストによる認証サーバでの比較

高スペックマシンからの認証サーバでの比較では，それぞれの CPU 使用率のピークにずれが見られ，https の方が 0.2 秒遅い．これはクライアントからのリクエストデータが送信されるのが，https が SAS Proxy に比べ遅いためである事が分かる．CPU 使用率のピーク値にはほぼ違いが見られない．面積比では，SAS Proxy の方が https に比べ 37.7%小さかった．これらのことより，認証サーバではピーク時における CPU 使用率にそれほどの差は見られなかったが，リクエストデータの遅延によりクライアント側のユーザ認証情報生成・暗号化処理において，SAS Proxy が https に比べ処理が高速であることが分かった．

（４）低スペックマシンからのリクエストによる認証サーバでの比較

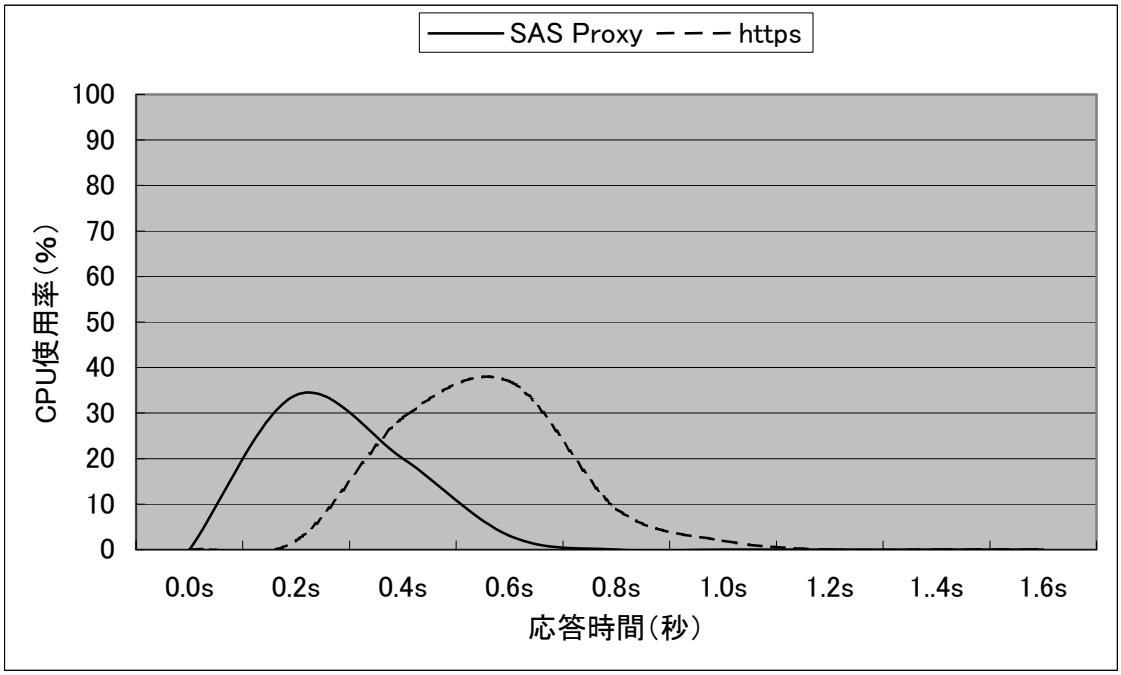

図 4.2.1.4　低スペックマシンからのリクエストによる認証サーバでの比較

低スペックマシンからの認証サーバでの比較では,（３）でも確認できた以上にhttpsクライアント側からのリクエストデータがSAS Proxy側に比べ遅れた．そのため,（３）同様CPU使用率のピーク値は変わらないものの，レスポンスデータを送信終了するまでにずれが生じた．このずれが，クライアント側でレスポンスデータを復号しwebコンテンツを表示し終わるまでの時間に影響し,最終的には大幅な応答時間の差となって低スペッククライアント側に現れたものということが分かった．面積比による比較ではSAS Proxyが27.9%小さかった．

（5）SAS Proxy と https とのクライアント総合比較

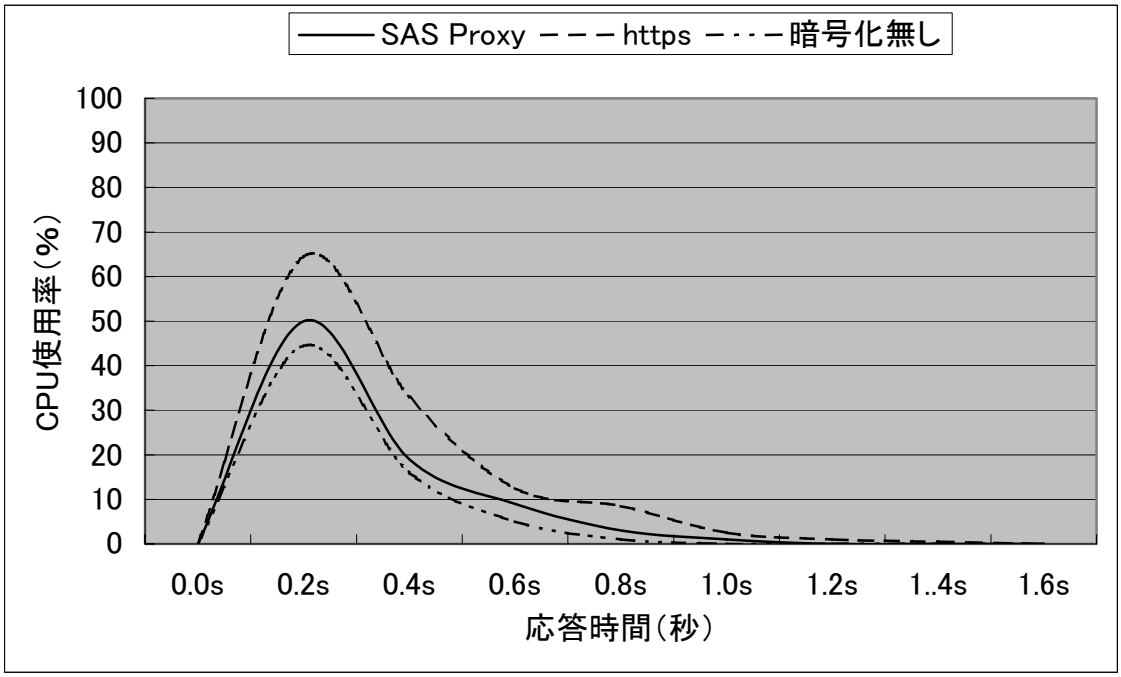

図 4.2.1.5　SAS Proxy と https とのクライアント総合比較

SAS Proxy と https のクライアント側での総合平均で比較を行い，また暗号化を行わない場合についても明記した．応答時間における違いは多少であるが，やはり CPU 使用率における違いは顕著で，面積比で SAS Proxy が 39.8%低負荷であった．また，暗号化を施さない場合と面積比で比べるとそれぞれ，SAS Proxy が 19.2%，https が 56.5%負荷が大きかった．総合的に,クライアント側では SAS Proxy は暗号を施さない場合と処理負荷の点であまり変わることなくブラウジングが行えるということが明らかとなった．

（6）SAS Proxy と https との認証サーバ総合比較

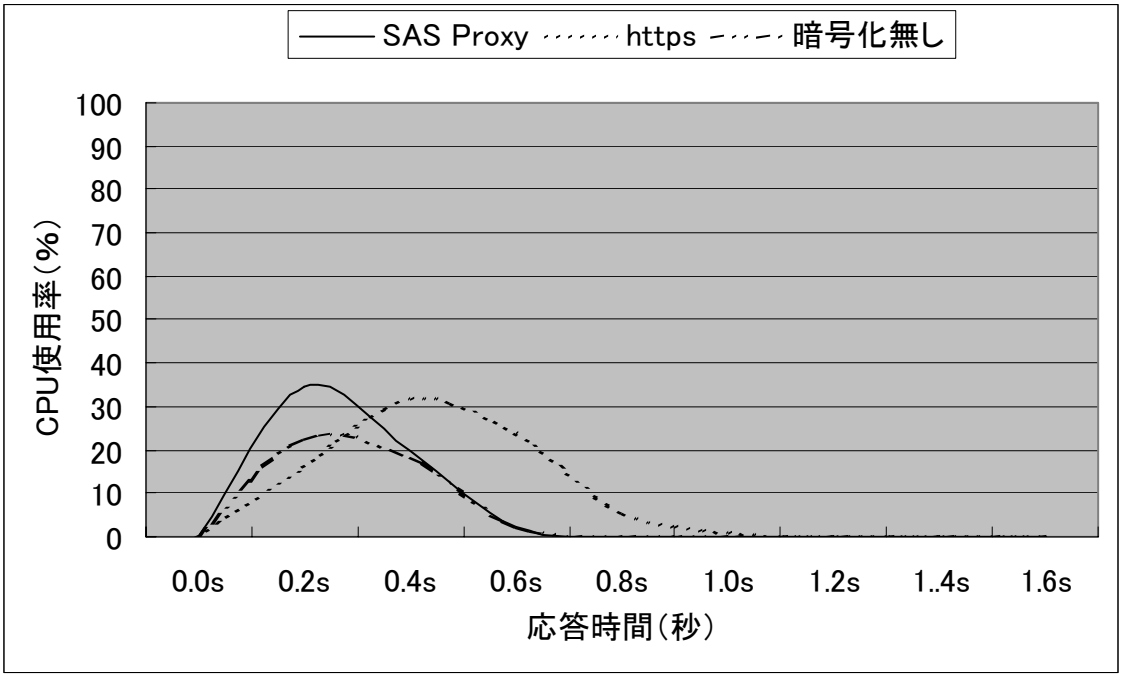

図 4.2.1.6　SAS Proxy と https との認証サーバ総合比較

SAS Proxy と https の認証サーバ側での総合平均で比較を行い，またユーザ認証を行わない場合についても明記した．グラフでは SAS Proxy の方が https に比べ CPU 使用率のピークが 0.2 秒の時点で高いが，これは SAS Proxy での CPU 使用率のピークが高スペック・低スペックともに 0.2 秒であり，https ではリクエストデータの遅延により，CPU 使用率のピークが同一時点ではなかったため，ピーク値だけに着目すると SAS Proxy が高負荷であった．

面積比でみてみると SAS Proxy の方が 28.4%低負荷であり，また暗号化を施さない場合と比べてみてもそれぞれ，SAS Proxy が 31.1%，https が 72.5%負荷が大きく,https では大きな負荷が認証サーバにかかっているのが分かる.それに比べると,SAS Proxy は負荷が低いといえる.また,レスポンスデータを配信終了するまで

の時間は，SAS Proxy と暗号を施さない場合で全く違いがなかったが，https はそれより 0.4 秒遅れている.総合的に,認証サーバにおいても SAS Proxy が https に比べ処理が高速であることが明らかとなった.

第 5 章

今後の課題

本研究の実験により，SAS Proxy と https を比べた場合，認証サーバ側では CPU 使用率のピーク値には大きな差は現れなかったが総合的にみてみると高速である．そしてクライアント側では CPU 使用率・応答時間ともに SAS Proxy の値が低く高速であることが分かり，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，暗号化を施さない場合との比較においては，認証サーバ側では処理負荷が多少かかるものの，クライアント側では CPU 使用率の点でほとんど違いが現れなかった．実験結果から SAS Proxy は 1 クライアントからのアクセスについては従来のブラウジング品質を損なうことなく，高速な通信を行える技術であることが実証できた．

今後は，同時に複数クライアントからアクセスがあった場合の SAS Proxy サーバと認証サーバの評価実験, https を Proxy サーバで動作させた場合の Proxy サーバ同士の比較実験を行い，さらに様々な視点での評価を行っていく必要がある．また卒業研究発表会にて, SAS 認証方式を用いたユーザ認証処理を www ブラウザで行えるようにしてはどうかとの意見を頂いたが, やはりこの場合でも個々のユーザの環境（マシンや OS）ごとに www ブラウザを用意する，またはその機能を付与するパッチ・プラグインなどを用意する必要があり，さらに付与する場合は www ブラウザのバージョンアップにも対応していかなければならないため, SAS 認証処理を www ブラウザに搭載させるとい

った方式は採らなかった．しかし，SAS Proxy は暗号を用いない場合との比較においても，遜色なく高速であるため机上での比較計算を行い，複数クライアントからアクセスがあった場合などの評価が必要である．

第 6 章

むすび

本論文では，従来技術である https や VPN の問題点について考察し，SAS Proxy を提案した．このシステムではユーザの認証に SAS 認証方式を用いることにより，SSL 認証方式を用いた https に比べ，処理手順を少なくすることができ，実験の結果でも SAS Proxy の方が高速にユーザ認証が行えることが分かった．また，ユーザ認証情報の生成とリクエストデータ・レスポンスデータの暗号化復号処理をクライアントに代わり行う Proxy サーバに代理させることにより，低スペックの web クライアントからでも従来のブラウジング品質を保ったまま，安全な通信が行えることができた．

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

本校に入学し２年目以来, 高知工科大学情報システム工学科清水明宏教授にはサークル活動，生活指導，就職活動，本来ならば資格のなかった自分に卒業研究をさせていただくなど学生生活全般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．

NTT アドバンステクノロジの真島大介氏には，本研究についての内容指導や評価実験についての御助言をいただき，また本研究室院生 田鍋潤一郎氏，岡田実氏，辻貴介氏，ならびに本研究室学部生 植田　洋加さん，越智裕架子さん，上岡隆君，河村智君，木村大樹君，窪内美紀さん，中務秀和君，福留一夫君，そして在学中に様々な御助言，御指導を頂いた諸先生方，いつでも快く力をかしてくれた友人たちに心から感謝する．

参考文献

[1]上岡 隆, 清水 明宏,“ ワンタイムパスワード認証方式 SAS の安全性に関する検討, ”
信学技報, OFS2001-48, Vol.101, No.435, pp.53-58, 2001