1Z－8

公開鍵暗号アクセラレーターカードの実装と評価†

宗藤誠治、高野光司、大庭信之

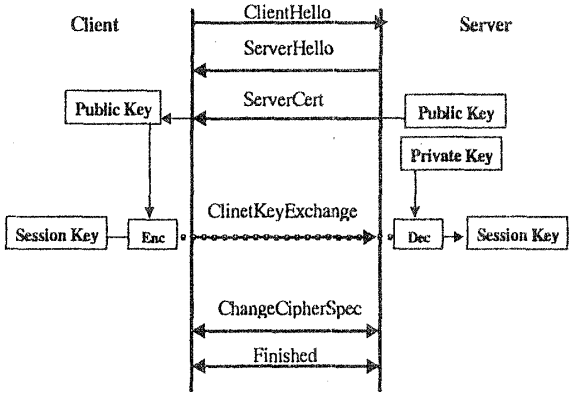
日本IBM、東京基礎研究所

1. はじめに

近年、SSL[1]をはじめとした公開鍵暗号を用いたインターネット上でのセキュリティプロトコルが普及している。また公開鍵インフラストラクチャー（PKI）の整備も進展している。そうしたなか、CPUによる公開鍵暗号の演算負荷が重い事が原因による、インターネットサーバーの処理能力の低下が問題として指摘されている[2]。本論文では、この問題の解決方法の一つとして、これらのサーバーに対して公開鍵暗号用の専用 H/W の実装およびその性能評価について報告をする。

2. SSL

図1に一般的な SSL のハンドシェイクプロトコルを示す。RSA 等の公開鍵暗号を用い以降通信の暗号化で用いるセッション鍵をクライアントとサーバー間で共有する事が出来る。



SSL Handshake Protocol

図1、SSLハンドシェイク

Client 側で生成された Session 鍵は Server の公開鍵で暗号化され Server に送られる。これを Server では Server の秘密鍵を用いて復号化し Session 鍵を Client/Server 間で共有するのだが、Server 側でのこの復号処理がもっとも重い計算であることになる[3]。よって専用 H/W によりこの部分を CPU からオフロードすることにより、Server の性能を 4～5 倍向上させる事が可能である。

表1に 1024-bit の RSA Sign 処理(=ModExp)の処理時間をまとめる。S/W の場合その実装方法により速度が左右されるが、いずれにせよ非常に多くの CPU 資源を消費することがわかる。

表1、ModExp 演算速度の比較

	1024-bit ModExp	備考(S/W の O/S は Linux)
S/W	230msec	BSAFE3(PentiumII,400MHz)
	60msec	OpenSSL(PentiumII,400MHz)
H/W	32msec	IBM,MEAC1024@33MHz

3. H/W の構成

専用 H/W の場合、その構成方法が重要である。今回使用するべき剰余演算 LSI は、場合によっては Server 側での SSL の処理全体を見た場合、転送するデータサイズによって左右されるが、S/W のみによって実装された SSL WebServer の場合、CPU 負荷の 70-80%がこの RSA Sign の演算に費やされる。現在の S/W による実装(CRT[4]を使用した場合、しない場合に比べ 4 倍程度高速になる）より劣る場合もある。しかしながら形状も消費電力も少ないために 1 枚の PCI カード上に 32 個搭載することが可能であり、その並列性を最大限活用すればカード単体の処理性能としては非常に高速なものになる（毎秒 800 回の 1024-bit べき乗剰余演算を実行）。

また、全体のシステムとしてもこのようなエンジンの並列実装は WebServer のようなマルチスレッド/マルチプロセスなアプリケーションとの相性も良いと予想される。問題は PCI カード上のエンジンと SSL を構成する S/W との I/F 部分でのオーバーヘッドであるが、デバイスドライバーはユーザープロセスとカード上のエンジンとの基本的な資源管理を行うだけでよく非常にシンプルに実装できる。エンジンの演算時間は既知のため、ドライバーは演算待ち処理では、決まった時間スリープする

† Development and Evaluation of Accelerator Card for the Public Key Cryptography
Seiji Munetoh, Kohji Takano, Nobuyuki Ohba
IBM Japan, Tokyo Research Laboratory

だけでよい（現在の実装ではこの時間を 40msec としている）。また RSA 演算での PCI バストラフィックは、演算時間に比してデータのやり取りが非常に小さいため、問題にならない。図 2 に全体の簡単な構成を S/W の実装の観点から示す。

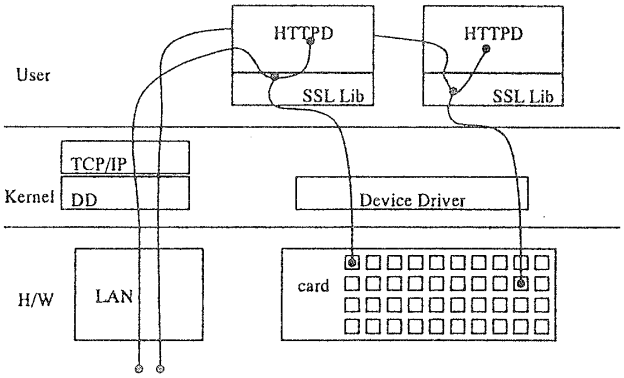


図 2、SSL-WebServer の構成図

4. 評価

現在 SSL 対応の WebServer の公式なベンチマークは存在しない為、SpecWeb96[5]で用いるワークロードを例にしてクライアント側のプログラムを作成した。Server 側では O/S に Linux を用い WebServer には Apache をまた SSL ライブラリーとして OpenSSL を使用した。表 2 に測定環境をまとめる。

表 2、測定条件

Server	CPU	PentiumII 400MHz
	Memory	320MB
	O/S	Linux 2.2.5
	WebServer	Apache 1.3
	SSL lib.	OpenSSL 0.9.3b
Client	CPU	PentiumII 400MHz
Network		100BaseT x 1

図 3 に S/W のみによる測定結果を示す。ここでは SSL ハンドシェイクの性能向上の 1 手法[2]である Session Reuse の効果も測定した。Session Reuse を行う事により、SSL ハンドシェイクでの RSA 演算を省略でき、この設定が有効になるような接続ケースでは Server の性能向上が期待できることが結果からもわかる。

次に図 4 で今回製作したカードを用いた場合の性能向上を示す。これから接続できるコネクション数は 5 倍以上、接続遅延も大幅に向上している事がわかる。本カードを用いることにより SSL ハンドシェイクの CPU 負荷は大幅に低減しており、この場合、Session Reuse の必要はもはやない。以上のことから、SSL においては、Server の CPU 能力に比して十分な能力の公開鍵アクセラレータを併用することにより公開鍵暗号演算を CPU からオフロードし大幅な性能向上が可能であることがわかる。

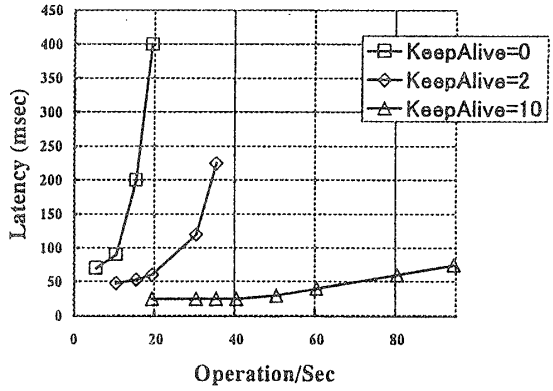


図 3、S/W のみの実装での SSL 性能の測定結果

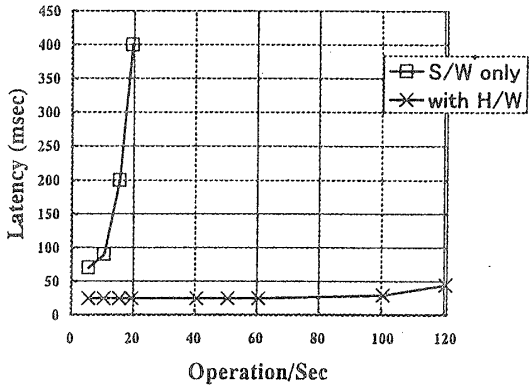


図 4、アクセラレーターを使用した場合の測定結果

5. 今後の課題

今回、公開鍵暗号アクセラレーターの有効性が確認できたが、これは非常に限定的な状況での性能評価である。今後はもっと大規模なシステムや現実のワークロード、CGI、データベースアクセス等によるバックエンド処理、クライアント認証等のより複雑なアクセスコントロールなども含めた形での性能評価を行う必要がある。

参考文献

[1] Secure Socket Layer, http://www.netscape.com/products/security/ssl/index.html

[2] G. Apostolopoulos, V. Peris, and D. Saha, "Transport Layer Security: How Much Does it Really Cost?", in the proceedings of INFOCOMM'99, New York, March 1999

[3] Shawn D. Abbott, "On the Performance of SSL", RSA Conference, Jan. 1997

[4] J.-J Quisquater and C. Couvreur, "Fast Decipherment Algolithm for RSA Public-Key Cryptosystem," Electronic Letters, v. 18, 1982, pp. 155-168

[5] SpecWEB96, http://www.spec.org/osg/web96