1Z-6

HTTPS 通信における暗号処理のオフロードのパフォーマンス解析

古澤喜明† 小川梨恵‡ 金子洋平‡ 齋藤孝道†
明治大学† 明治大学大学院‡

1. はじめに

近年，WAN/LAN 間の通信や遠隔地からのリモートアクセスなど，インターネット上でのデータ通信が広く使われるようになった．他方，通信の盗聴，改竄や成り済ましなどの脅威も増えている．それらの脅威への対策として，セキュリティプロトコルである SSL/TLS(Secure Sockets Layer/ Transport Layer Security)が利用されている．しかし，これらは，暗号化/復号といった高負荷な演算処理を伴うので，システム全体のスループットの低下を招く．その解決策の一つとして，暗号処理に最適化されたモジュールにオフロードする方法（以下，オフロードという）がある．

研究[1]では，UltraSPARC T2 上に Web サーバを構築し，DTrace を用いて SSL/TLS を用いた通信のパフォーマンスを測定することで，暗号処理が通信に与える影響について調査した．

本論文では，HTTPS 通信におけるオフロードの効果を評価するため，さらに SSL/TLS を用いた通信を解析し，通信全体における暗号処理の割合を計測した．さらに，研究[1]との比較により，オフロードが HTTPS 通信全体にどの程度パフォーマンス向上に寄与するかを考察した．

2. 計測環境

2.1 UltraSPARC Tx

UltraSPARC Tx プロセッサはサン・マイクロシステムズのマルチスレッド・マルチコアのプロセッサである．

本論文では，UltraSPARC T2 プロセッサを使用し，メインメモリは 16GB，OS は Open Solaris 10 とした．UltraSPARC T2 は 1 つのプロセッサに 8 つのコアが搭載されており，各コアは 8 つのスレッドを同時に扱うことができる．

そのため，最大 64 個のスレッドを同時に実行可能である．また，各コアには浮動小数演算ユニット（FPU：Floating point / Graphics Unit）と暗号処理ユニット（SPU：Stream Processing Unit）を搭載している．各コア，FPU 及び SPU はメインメモリを共有しており，並列に動作できる．

SPU は主に MAU（Modular Arithmetic Unit）と暗号/ハッシュ・ユニットから構成される．MAU は FPU を利用し，公開鍵暗号方式（RSA），及び楕円曲線暗号を処理する．暗号/ハッシュ・ユニットは共通鍵暗号方式やハッシュ関数に対応しており，DES，3DES，AES，RC4，SHA-1，SHA-256 と MD5 を利用できる．

2.2 DTrace

DTrace は Solaris10 から Solaris OS に導入された，システム情報をトレースする機能である．OS もしくはアプリケーションに手を加えることなく，稼働中のシステム上で動作する OS，アプリケーションや，リソースに関する情報をリアルタイムで採取することができる．D スクリプトを用いて，情報収集を行うための独自のスクリプトを記述することもできる．本論文ではパフォーマンス解析を行うために，Apache モジュール内で動作する関数ごとに実行時間を計測するスクリプトを新たに記述した．

2.3 OpenSSL

OpenSSL は，SSL/TLS だけでなく，証明書の発行といった PKI（Public Key Infrastructure）関連の処理や公開鍵暗号方式，共通鍵暗号方式などを容易なインタフェースで利用可能とした API ライブラリを含むツールキットである．

OpenSSL が提供するライブラリは libssl と libcrypto があり，本論文の実験環境では，libssl.so.0.9.8 と libcrypto.so.0.9.8 である．前者は SSL/TLS 通信を，後者は暗号技術を提供するライブラリである．共通鍵暗号方式として DES，3DES や AES など，公開鍵暗号方式として RSA や DH（Diffie-Hellman）など，ハッシュ関数として SHA-1 や MD5 など，主要なアルゴリズムが利用可能である．

また，PKCS#11 を用いることで，暗号モジュールへアクセスし，暗号処理をオフロードすることができる．PKCS#11 は，ハードウエアモジュールに対して暗号処理を行うための API が規定されている．RSA PKCS#11 v2.11 標準を実装したものが libpkcs11.so.1 である．libcrypto とは別である．

3. 評価

3.1 評価方法

本論文では，研究[1]と同様に，UltraSPARC

†Yoshiaki FURUSAWA, ‡Rie OGAWA, Yohei KANEKO
†Takamichi SAITO
Meiji University (†), Graduate School of Meiji University (‡)
1 -1-1 Higashimita, Tama-ku, Kawasaki-shi, Kanagawa, 214-83571, Japan (†) (‡)

T2 上に Apache を用いて構築した SSL-Web サーバに対して HTTPS リクエストの負荷をかける．さらに，HTTPS 通信に伴う暗号処理を暗号モジュールにオフロードしてパフォーマンスの評価を行い，研究[1]の評価結果と比較し，考察する．HTTPS リクエストを生成する負荷生成器には，Avalanche2007 モデル B（以下，Avalanche と呼ぶ）を用いる．Avalanche により複数の利用者の閲覧をエミュレートし，各利用者が HTTPS リクエストをそれぞれ送信するような負荷を生成する．生成する負荷は，今回の実験環境のネットワークで最大限の負荷である毎秒 200 リクエストとした．認証モードは，サーバ認証モードとした．各リクエストで使用する暗号方式のセットである暗号スイートには，研究[1]と同様の DES-CBC-SHA を用いた．さらに，SSL/TLS Handshake 処理に与える影響が大きい EDH 鍵交換方式[3]を採用する EDH-RSA-DES-CBC-SHA との比較をする．表 1 に各暗号スイートを示す．また，リクエストに返信する Web ページのサイズは，平均的な Web ページのサイズである 50Kbytes[2]とした．

表 1：各暗号スイートで使用する暗号方式

暗号スイート	鍵交換	認証	暗号化	ハッシュ関数
DES-CBC-SHA	RSA	RSA	DES	SHA-1
EDH-RSA-DES-CBC-SHA	DH	RSA	DES	SHA-1

3.2 評価結果

SSL-Web サーバがリクエストを処理する際のライブラリやモジュールごとの処理時間，それらの割合を DES-CBC-SHA を用いた場合は表 2，EDH-RSA-DES-CBC-SHA を用いた場合は表 3 に示す．計測においては，5 回計測しその平均を取った．なお，DES-CBC-SHA を用い，オフロードを行っていない場合の評価結果には研究[1]の結果を使用した．

暗号処理を行うライブラリやモジュールは，暗号化ライブラリ libcrypto.so.0.9.8，libpkcs11.so.1，カーネルレベルの暗号化機構にアクセスするための共有オブジェクト pkcs11_kernel.so.1 と，libssl.so.0.9.8 である．表 2 より， DES-CBC-SHA を用いて暗号処理のオフロードを行った場合，全体の処理時間に対する暗号処理の割合は 29.55%となる．暗号処理以外には C の標準ライブラリなどがある．オフロードを行っていない場合，暗号処理の割合は 77.20%である．暗号処理のオフロードを行った場合と比較すると，通信全体に対して暗号処理が占めている割合が 2.6 倍程度大きいことがわかる．

EDH-RSA-DES-CBC-SHA を用いた場合，表 3 より，オフロードを行っていない場合の暗号処理の割合は 48.76%である．それに対し，オフロードを行った場合は 17.11%である．図 1 に示す通り，DES-CBC-SHA を用いた場合と同様に通信全体に対して暗号処理が占めている割合が減少し，通信全体の処理時間が 41%程度になっていることがわかる．

表 2：オフロード実行の内訳（DES-CBC-SHA）

ライブラリ/モジュール	オフロード無[1]		オフロード有	
	時間(ms)	割合	時間(ms)	割合
libc.so.1	1862	13.01%	3169	59.19%
libapr-1.so.0.3.3	578	4.03%	348	6.51%
libpkcs11.so.1	-	-	549	10.25%
libcrypto.so.0.9.8	10923	76.34%	633	11.82%
pkcs11_kernel.so.1	-	-	367	6.86%
libssl.so.0.9.8	122	0.86%	33	0.62%
その他	822	5.76%	255	4.75%
全体	14307	100%	5355	100%

表 3：オフロード実行の内訳（EDH-RSA-DES-CBC-SHA）

ライブラリ/モジュール	オフロード無		オフロード有	
	時間(ms)	割合	時間(ms)	割合
libc.so.1	8261	38.25%	5809	65.93%
libapr-1.so.0.3.3	2717	12.58%	1255	14.25%
libpkcs11.so.1	-	-	516	5.86%
libcrypto.so.0.9.8	10507	48.64%	611	6.94%
pkcs11_kernel.so.1	-	-	347	3.93%
libssl.so.0.9.8	26	0.12%	33	0.38%
その他	89	0.41%	239	2.71%
全体	21600	100%	8810	100%

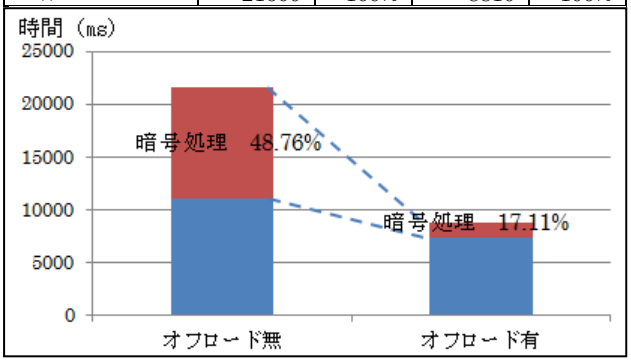

図 1：処理時間の割合の比較（EDH-RSA-DES-CBC-SHA）

4. まとめ

本論文では，SSL-Web サーバの暗号処理のオフロードが，HTTPS 通信全体において，どの程度パフォーマンス向上に寄与するかを考察した．実験から，暗号処理のオフロードを行うことで，通信全体に対して暗号処理が占める割合が減り，通信全体の処理時間が半分以下になるなど， 本論文の実験環境においては，大幅なパフォーマンス向上が期待できることがわかった．

参考文献

[1] 小川，天野，齋藤，SSL/TLS 処理のパフォーマンス解析について，第 75 回全国大会講演論文集
[2] L.Badia, Real World SSL Benchmarking, Rainbow Techologies Whitepaper, Sept.2001
[3] 齋藤孝道，マスタリング TCP/IP 情報セキュリティ編，オーム社，2013.