FPGA による HMAC-SHA-512 のハードウェア化

Nguyen Truong Son† 黒川 恭一† 岩井 啓輔†
防衛大学校情報工学科†

1 はじめに

現在のコンピュータネットワークでは，送受信される情報の正当性を認証することが非常に重要である．IPSEC 中の認証方針として代表的なものに，元のメッセージ及びその正当性を確保できるという特徴を持っている MAC (Message Authentication Code)が提案されている．中でも暗号ハッシュ関数と秘密鍵とを組み合わせる HMAC というアルゴリズムは，近年 VPN（Virtual Private Network）の IPSEC アプリケーションのための高速性能暗号アクセラレーター等に利用され始めている[1]．

暗号ハッシュ関数としては，MD5・SHA-1・SHA-512 等が知られ，現在広く使われている．最近では，HMAC-MD5 や HMAC-SHA-1 等はハードウェア実装もされている．またソフトウェア実装では，HMAC-SHA-512 等も実現されている．これに対して本稿では，SHA-512 に基礎を置く HMAC の新たな実装として，FPGA を用いてハードウェア化し，その性能評価を回路規模及び処理速度の面から検討する．さらに，SHA-512 の FPGA 実装が文献[2]で示されており，その実装結果と今回実装した結果との比較・検討も行う．

2 HMAC とハッシュ関数の概要

2.1 HMAC の概要

HMAC(Keyed-Hashing for Message Authentication)は，1997 年に H. Krawczyk (IBM), M. Bellare (UCSD), R. Canetti (IBM)によって提案された．ハッシュアルゴリズムをカプセル化して，鍵による保護を可能とするための方法である．現在では FIPS PUB 198 としても制定されている[3]．

HMAC とは，SHA-512 などの反復暗号ハッシュ関数を秘密鍵と組み合わせて使用し，メッセージ認証を行うものである．HMAC の暗号としての強度は，使用しているハッシュ関数のプロパティに依存する．HMAC を定義するためには，暗号ハッシュ関数 H と，秘密鍵 K とが必要となる．ここで，暗号ハッシュ関数は，基本的な圧縮関数をデータブロック単位で繰り返すことによってデータがハッシュされるものである．また，MAC の有名な実施法に MAC の出力を省略し，そのビットの一部のみを出力する方法[4]がある．HMAC のアプリケーションでは，あるパラメータ t を用いて HMAC の計算結果の左から t ビットまでを出力することによって HMAC の出力を一部省略することができる．HMAC を計算するために，バイト値 0x36 を B 回繰り返した文字列(ipad)と，バイト値 0x5C を B 回繰り返した文字列(opad)とが必要である．データ”Text”に対する HMAC の計算は、次のような式に従う．

$$HMAC(K,Text) = H((K_0 \oplus opad)||H((K_0 \oplus ipad)||Text)) \quad (1)$$

この式に従って HMAC は，以下のように処理を進めていく．

1. B バイトの文字列を作るように K の終わりまでゼロを追加し，B バイトの K_0 とする．
2. 1.で計算された B バイトの文字列と ipad との XOR をとる．
3. 2.の結果に，データ”Text”のストリームを追加する．
4. 3.で生成されたストリームに H を適用する．
5. 1.で計算された B バイトの文字列と opad との XOR をとる．
6. 5.の結果に，4.の計算結果を追加する．
7. 6.で生成されたストリームに H を適用し，その結果を出力する．

FPGA Implementation of HMAC-SHA-512
†Nguyen Truong Son, KUROKAWA Takakazu and IWAI Keisuke
Department of Computer Science, National Defense Academy

2.2 SHA-512 の概要

電子署名に使用される公開鍵暗号方式は非常に低速なので，署名対象ドキュメントを全部暗号化していたのでは遅い上にサイズも大きくて実用的ではない．そこで，ハッシュ関数を用いてハッシュ値を生成し，これを暗号化することでドキュメントの署名としているわけである．

NIST は，1992 年に暗号ハッシュ関数として，160 ビット出力のハッシュ関数 SHA を，また 1994 年にはその改定版の SHA-1 を FIPS-180 で提案している．さらに NIST は，SHA-1 とほぼ同様の設定原理に基づいて，256 ビットのハッシュ関数 SHA-256，384 ビットのハッシュ関数 SHA-384，512 ビットのハッシュ関数 SHA-512 の三種類をそれぞれ提案している[5]．これらのハッシュ関数を導入した最大の理由は,衝突攻撃のセキュリティレベルが各々128，192，256 ビットであり，最近採用された三種類のブロック暗号 AES-128，AES-192，AES-256 と対応させるためである．これらのハッシュ関数の特徴を表 1 に示す．

表 1. ハッシュ関数 SHA の特徴

種類	SHA-1	SHA-256	SHA-384	SHA-512
Message size	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Block size	512	512	1024	1024
Word size	32	32	64	64
Digest Round	80	64	80	80
Constant	4	64	80	80
Digest size	160	256	384	512

3 FPGA によるハードウェア化

3.1 FPGA チップと PCI ボードの概要

今回利用した FPGA は，Xilinx 社製の Virtex-II XC2V6000 である[6]．Virtex-II CLB の基本ビルディングブロックは，LC (Logic Cell)と呼ばれ，1 個の LC には，4 個のファンクション・ジェネレータ，キャリーロジック，記憶エレメント,ロジックゲートが内蔵されている．XC2V6000 と[2]の実装で使われた Virtex XCV1000 の特徴の比較を表 2 に示す．

表 2　XC2V6000 と XCV1000 との比較

FPGA デバイス	XCV1000	XC2V6000
System Gates	1M	6M
CLB Slices	12,228	33,792
Multiplier Blocks	-	144
SelectRAM Blocks	32	144
User I/O Pads	404	1,104

今回の実装では，XC2V6000 を用いた KAC-02A という PCI 大規模 FPGA ボード[7]を利用した．KAC-02A の特徴としては，内蔵メモリを含めて 12.3M のシステムゲートを有するため，大規模な論理を構築でき，ユーザの構築した論理回路の動作周波数を 6MHz~200MHz の範囲で任意に設定可能である．KAC-02A ボードを図 1 に示す．

図 1　KAC-02A

3.2　ハードウェア構成

今回の実装では，コンパクトな回路規模を目指し，特に CLB スライス数の低減を目的としてハードウェア設計を実現する．

2.1 節に述べたように HMAC 計算ではハッシュ関数の処理を 2 回繰り返す．そのため，今回の実装では，HMAC を 2 つの処理に分けて実現した．一つは H_1(K⊕ipad||Text)であり，もう一つは H_2(K⊕opad||H_1)で，この計算結果を HMAC の出力とする．

SHA-512 という反復ハッシュ関数の処理の部分に対しては，今回はより小さなエリアを占めるということでループアーキテクチャを採用した．秘密鍵 K，データ Text，K⊕ipad あるいは K⊕opad の結果，ハッシュ関数の定数，メッセージスケジュール，前回処理のハッシュ値等を格納するためにレジスタを用いた．HMAC-SHA-512 のブロック図を図 2 に示す．

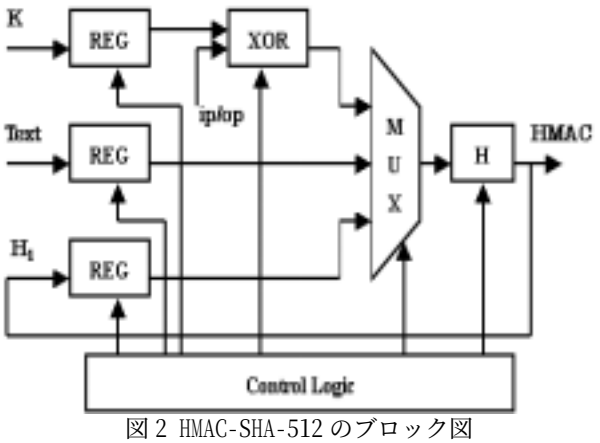

図 2 HMAC-SHA-512 のブロック図

4　実装結果

今回の実装では，システムデザイン用のツールに ISE 4.1.03i を利用した．回路設計には Verilog HDL を用い，またシミュレーションに ModelSim XE 5.5b を利用した．今回の実装結果を表 3 に示す．同じくループアーキテクチャを用いて SHA-512 を実装した[2]の結果も合わせて示す．今回実装した回路はコンパクトな回路規模を目指しているので，[2]と比較して、スライス数を 23%減らすことができた．

表 3　実装結果と[2]の結果

	[2]の SHA-512	本実装の HMAC-SHA-512
FPGA	XCV1000	XC2V6000
PCI ボード	SLAAC-1V (64bit/66MHz)	KAC-02A (32bit/33MHz)
Slices	2,826	2,177
Gates	-	440,694
Throughput (Mbps)	616	488

5　おわりに

本稿では，新たな SHA-512 に基礎を置く HMAC の設計を行い，さらに KAC-02A 大規模 FPGA ボードに実装した．[2]の実装結果と CLB スライス数,スループットの面から，比較・検討を行った．その結果，小さなエリアを占めるループアーキテクチャとして，スライス数を 2177，また，スループットでは 488 Mbps という値が得られた．今回ハードウェア設計に対して，より高速化が可能なアーキテクチャや入出力の並列処理化を行うことにより，どの程度の性能向上が得られるのかを検討することが今後の課題となる．

参考文献

[1] Security Architecture for the Internet Protocol: http://www.cis.ohio-state.edu/cgi-in/rfc/rfc2401.html

[2] Tim Grembowski, Roar Lien, Kris Gaj, Nghi Nguyen, Peter Bellows, Jaroslav Flidr, Tom Lehman, Brian Schott, “Comparative Alnalysis of the Hardware Implementations of Hash Functions SHA-1 and SHA-512”, Proc. Information Security Conference, Sao Paulo, Brazil, September 30-October 2, 2002.

[3] The Keyed-Hash Message Authentication Code (HMAC), 2002: http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf

[4] Meyer S. and Matyas S.M., “Cryptography”, New York Wiley, 1982.

[5] Secure Hash Standard, 2002: http://crypto.nknu.edu.tw/crypto/fips180-2.pdf

[6] Virtex-II 1.5V Field-Programmable Gate Arrays: http://direct.xilinx.com/bvdocs/publications/ds031-1.pdf

[7] KAC-02A 大規模FPGAボード: http://www.mee.co.jp/pro/sales/fpga/fpga.html

[8] Handbook of Applied Cryptography, by A. Menezes, P. van Oorschot, and S. Vanstone, CRC Press, 1996.