# Design Challenges in Security Processing

Yunfei Wu and Stephan Wong
Computer Engineering Laboratory,
Electrical Engineering Department,
Delft University of Technology,
{Y.Wu,J.S.S.M.Wong}@EWI.TUDelft.NL

*Abstract*— **In this paper, we present an investigation into the challenges in designing a network processor that is additionally capable of security processing. First, we provide an overview of the cryptographic algorithms utilized in security processing and highlight several current network protocols that employ these algorithms. Second, we discuss several issues in designing a solution for security processing through the utilization of general-purpose processors, application-specific circuits, and/or reconfigurable hardware. Third, several existing commercially available security processors are surveyed followed by an examination of their system architecture and functional objectives. Finally, we highlight the challenges that designers are facing in designing a novel network processor intended to perform both security processing and other network processing tasks. The challenges lies in finding solutions for the following: programmability, reconfigurability, dynamic reconfiguration, new instruction sets, wireless security processing, and secure multimedia data processing.**

*Keywords*— **Network Processors; Security Processing; Protocol Processing; Reconfigurability; Programmability**

## I. Introduction

The widespread utilization of the Internet coupled with the explosive growth in e-commerce has created a huge demand for information security. Next to the growing need for secure online transactions, networks are constantly facing a wide variety of security issues, e.g., viruses and the possible threat of a massive attack from terrorist hacker groups. As a consequence, the importance of network security must never be overlooked or taken lightly. In order to secure information in networks, a multitude of cryptographic algorithms and protocols have been introduced. Recently, the security processing market has bloomed and its market is expected to growth from about $30 million in 2002 to over $280 million by 2007 [12].

Current security processing solutions entail the utilization of application-specific circuits (ASICs) or a hybrid between ASICs and general-purpose processors (GPPs). Both solutions are capable in providing adequate processing performance, however, their shortcomings quickly become evident when other requirements must be met, e.g., low cost and high flexibility. Low cost solutions must be sought after in order to speed up the uptake of security processing solutions in current and future networks. Flexibility is needed to cope with the ever-changing manner that security processing is performed. This is due to the simple fact that the forms of attacks on networks change over time. Consequently, security processing solutions must be able to support new cryptographic algorithms and security protocols that most certainly will be introduced in the future. This must be achieved preferably without completely replacing the security processing solution(s) that were already installed in the past. In recent years, reconfigurable hardware, e.g., field-programmable gate arrays (FPGAs), is making headway in reducing their cost and increasing their processing performance. This combined with their inherent flexibility is making the utilization of reconfigurable hardware an increasingly more viable approach. Finally, the recent advent of flexible and high-performance so-called network processors[1] opens up possibilities in incorporating security processing into such processors and, therefore, more tightly integrating security processing with other types of network processing.

The challenges in designing a single solution for security processing lie in simultaneously meeting the mentioned requirements of high performance, low cost, and high flexibility. Concurrently, other requirements must be met when considering a specific network type. For example, mobile devices in a wireless network require low-power solutions. In this paper, we discuss these challenges in more detail. This paper is organized as follows. Section II provides a general overview on security processing. Section III presents possible implementation choices in designing a stand-alone security processor and discusses three system architecture possibilities in which it can be placed. Section IV presents a comparison between currently available security processors used in different system architectures. Section V addresses the challenges in designing a novel network processor architecture that also incorporates security processing. Finally, Section VI presents the conclusions of this paper.

[1]We have to note that the precise definition of what is a network processor remains an open issue as well as their constituent parts.

## II. Security Processing

In this section, we discuss the four main issues that must be addressed in designing security systems, introduce the cryptographic algorithms, and present the network protocols that incorporate these algorithms. The four main security issues are confidentiality, authentication, integrity, and non-repudiation. Confidentiality means that only authorized users can access the information by prohibiting access by unauthorized users. Authentication means that the user accessing the information is truly the person who he says he is and not an imposter. Integrity has to ensure that the received information is identical to the transmitted information without being modified by others during transmission. Non-repudiation ensures that senders and receivers have undeniably transmitted or received information, respectively. The four mentioned issues are interdependent and must therefore be addressed simultaneously in the design of security systems.

Cryptographic algorithms are utilized to encrypt an original plaintext message into a ciphertext at the sender side and to decrypt the ciphertext back to the original message at the receiver side. The encryption and decryption processes generally depend on a secret key being shared between the sender and the receiver. There are three types of cryptographic algorithms: symmetric-key algorithms, public-key algorithms, and hashing functions, which are explained in the following:

1. *Symmetric-key algorithms:* In symmetric-key algorithms, both the sender and the receiver utilize the same key for both encryption and decryption. In a two-party communication, both parties must know/obtain the same key before transmission and measures must be taken to keep the key a secret. The key distribution becomes increasingly more difficult when the network grows since each pair of users must exchange keys. The total number of key exchanges required in an $n$-person network is $n(n-1)/2$. Historically, distributing the keys has always the weakest link in most cryptosystems [3]. If an intruder could steal the key, the system becomes worthless. Therefore, though symmetric-key algorithms provide strong security, they suffer from key distribution and key management [5]. The widely adopted symmetric-key algorithms by the industry include Data Encryption Standard (DES), Triple DES (3DES), RC4, RC5, and Advanced Encryption Standard (AES).

2. *Public-key algorithms:* Public-key algorithms are based on each party having their own private key, which is shared with no-one, and a public key that is known to all other communicating parties. When sending a message to a particular receiver, the receiver's public key is used to encrypt the message. After receiving the message, it is decrypted using the receiver's own private key. Furthermore, encryption/decryption operations when using private keys are generally slower than those involving the public key. Research [9] has shown that private key operations can be more than $38$ times slower than public key operations with a $2048$-bit key. In addition, the time to perform a public/private key operation increases with a power of the key length $K$. Compared to symmetric-key algorithms, public-key algorithms eliminate the need to secretly distribute a key, therefore, increasing the security over symmetric-key algorithms. On the other hand, the processing of symmetric-key algorithms can be more efficiently performed than the processing of public-key algorithms. Examples of public-key algorithms include RSA, Digital Signature Algorithms (DSA), and Elliptic Curve Cryptography (ECC).

3. *Hashing functions:* Unlike the two types of cryptographic algorithms mentioned above, hashing functions do not involve the use of keys. They take a variable-length input string (called a pre-image) as input and convert it to a fixed-length (generally smaller) output (called a hash value). The most widely used hash function is the message digest version 5 (MD5). An important advantage of MD5 is that it is much more efficient to compute than either RSA or DES. Another well-known hash function is Secure Hash Algorithms 1 (SHA-1).

In order to provide Internet security, network protocols were introduced that incorporate the above-mentioned cryptographic algorithms. The most popular and widely used protocols include IP security (IPsec) [24], Secure Sockets Layer protocol (SSL) [15], and IEEE 802.11i (for wireless LAN), which are discussed in the following:

1. *IPsec:* The IPsec protocol is a set of protocols that support authentication and privacy services at the network layer (the third layer of the OSI model), and can be used in conjunction with both IPv4 and IPv6. The IPsec protocol comprises two principal parts. The first part describes two approaches that can be utilized to carry security identifiers, integrity control data, and other information. The first approach entails the insertion of the authentication header (AH) [22]. The second approach, called encapsulation security payload (ESP) [23], modifies a datagram by inserting a header and trailer and encrypting the data being sent. The second part of IPSec entails Internet security association and key management protocol which deals with establishing keys.

2. *SSL:* The SSL protocol developed by Netscape Communications Corp. has been widely used on the World Wide Web (WWW) to authenticate and encrypt data communication between clients and servers. The SSL protocol

defines two types of authentication. The first type is SSL server authentication that allows a user to confirm identity of a server. This confirmation is important when the user is sending sensitive information over the network to a specific server. Prior to transmitting information, the client first validates the server's certificate. The second type is SSL client authentication which allows a server to confirm the identity of an user using the same techniques as those used in server authentication. Furthermore, all information transmitted between client and server is encrypted to provide confidentiality. The SSL protocol includes two subprotocols: the SSL record protocol and the SSL handshake protocol. The SSL record protocol defines the format utilized to transmit data. The SSL handshake protocol involves using the SSL record protocol to exchange a series of messages between an SSL-enabled server and an SSL-enabled client when they first establish an SSL connection. The latest version of the SSL protocol is called Transport Layer Security (TLS), which is described in RFC 2246.

3. *IEEE 802.11i:* The advent of wireless network also raised questions in how to incorporate security in such networks. An initial attempt entails a data link-level security protocol called Wired Equivalent Privacy (WEP), which was designed to match the security level of a wireless network to that of a wired network. The WEP protocol utilizes the RC4 algorithm with a variable length key for confidentiality to protect information. However, several security problems related to WEP have been reported and are described in [20], [21], [1]. As a result, the IEEE Task Force I (802.11i) is considering an improved and more robust method to incorporate security in wireless networks. The 802.11i utilizes a much stronger AES encryption algorithm. However, implementing the AES-based solution will require new hardware and protocol changes. A less elaborate and short-term solution to improve security in wireless is the WiFi Protected Access (WPA). The WPA group is defining the Temporal Key Integrity Protocol (TKIP) for improved security without requiring hardware changes.

## III. Implementation Issues in Security Processing

Traditionally, security processing has been performed in software running on general-purpose processors (GPPs). This has been an advantage since the GPP is programmable and has the flexibility to adapt to new algorithms. However, security processing is computationally intensive and requires increasingly more CPU power as the bandwidths of networks continue to increase to well beyond 10 Gbps and reaching 40 Gbps in the near future. For example, when IPSec processing is added to a GPP processor able to perform other network processing tasks to reach a bandwidth of 2.4 Gbps, the maximum bandwidth will dramatically drop to less than 1Gbps [4]. In order to overcome the performance bottleneck in security processing, researchers have traditionally investigated hardware acceleration by using application-specific integrated circuit (ASIC) implementations for various encryption algorithms (several examples are discussed in Section IV). However, the main drawback of ASICs is that they lack the flexibility to be easily updated to support new features. Moreover, ASICs have a relatively long development time and high design costs.

In order to achieve both high performance and high flexibility, researchers are starting to implement security algorithms in reconfigurable hardware (e.g., field-programmable gate array (FPGA)) [16][18][25][11]. High performance is achieved by customizing datapaths and operators and by parallel processing. It was shown that reconfigurable hardware can considerably speed up security processing. High flexibility is achieved by changing functionalities of the reconfigurable hardware. In addition to the high performance and high flexibility, the utilization of reconfigurable hardware in cryptographic processor design can reduce the power consumption [11], which is important in wireless communications.

In the previous paragraphs, we have discussed several approaches in utilizing GPPs, ASICs, or FPGAs to implement a processor solely for security processing. However, how the processor is incorporated into the final security system architecture is another aspect that must not be overseen. Moreover, the overall system contains a multitude of components to perform many other network processing tasks and we are witnessing a convergence trend that these network processing tasks are being handled by a single network processor. Therefore, in order to facilitate the following discussion, we have separated security processing (to be incorporated into the security processor) from other network processing tasks (to be incorporated into the network processor). This results in that three approaches can be identified: look-aside approach, inline approach, and embedded approach. First, in the look-aside approach, the security processor is working as a separate processor using a shared bus, e.g., PCI, to communicate with a network processor. This approach creates a serious performance bottleneck in data transfers in two ways: first, every packet for security processing traverses the memory four times and, second, other peripherals must share the same PCI bus with the network processor and security processor. Second, the inline (also called flow-through) approach solves the performance problems of the look-aside approach in moving the security processor closer to

the network processor. The security processor is placed 'before or after' or 'before and after' the network processor. The main disadvantage is that the security processor has to perform many of the functions that the network processor is targeted for, e.g., packet reassembly. Third, in the embedded approach, security processing is integrated with other network processing tasks into a single processor. Compared to the other two approaches, memory and bus traffic overhead is greatly reduced and redundant operations are avoided. Even though some initial attempts are known, many challenges need to be overcome in designing a network processor that additionally incorporates security processing. These challenges are discussed in Section V.

## IV. Commercial Processors for Security Processing

In this section, we compare the several state-of-the-art commercial security processors with respect to the its embedding system architecture and processing of cryptographic algorithms and security protocols. These processors include Layer N LNN2010 [17], Corrent CR7020 [8], Broadcom BCM5801 [6], Motorola MDC180 [19], Interphase 55NS [14], Hifn HIPP8065/8165 [10], AMD Au1550 [2], Intel IXP2850 [13], and Cavium NITROX II [7]. This comparison is not meant to imply that these vendors or products are in any way better than others. All information is based on documents released by vendors.

### A. System Architecture Comparison

Table I presents the system architecture features and some implementation features of each security processor. From Table I, we can see that current security processor implementations employ either ASICs or ASICs in conjunctions with a RISC-type general-purpose processor. Furthermore, we notice that the ASIC implementations generally employ the look-aside approach and communicate with other parts of the system through the PCI bus. In the hybrid ASIC/RISC implementation, the ASIC is mainly used to accelerate security processing and the RISC procesor is mainly used to process other network processing tasks implemented in software.

### B. Functionality Comparison

Table II shows which functionalities/security processing (discussed in Section II) each security processor performs:

- Symmetric-key algorithms: All security processors support hardware acceleration of DES and 3DES. Some processors (like Interphase 55NS, AMD Au1550, Hifn HIPP8065/8165, and Cavium NITROX II) also support the other two symmetric-key algorithms, AES and ARC4.
- Public-key algorithms: Most implementations provide hardware acceleration of RSA. Exceptions are the Intel IXP2850 and Corrent CR7020 that accelerate part of operations in hardware and implement the remaining operations in software. Motorola MDC180 has a programmable key length from $80$ bits to $2048$ bits and supports the ECC public-key algorithms. Layer N LNN2010, Hifn HIPP8065/8165 and Cavium NITROX II support a public key length of more than $2048$ bits.
- Hashing functionalities: All implementations support the SHA-1 and MD5 algorithms in hardware. The only exception is the Intel IXP2850. In this case the MD5 algorithm was implemented in another part of the system and was therefore left out.
- IPsec processing: Most of these security processors can accelerate IPsec processing. The AMD Au1550 implements the whole IPsec processing in hardware and the Intel IXP2850 provides both hardware acceleration and software flexibility in IPsec processing.
- SSL processing: Most of these security processor can accelerate both SSL and TLS processing and the Intel IXP2850 can provide both hardware acceleration and software flexibility.

## V. Challenges

In previous sections, we discussed the cryptographic algorithms and network protocols that need to be supported by a security processor. Furthermore, we presented several implementation approaches, discussed how security processing can be incorporated in an overall system, and showed several commercially available security processing solutions. Currently, we are witnessing an emerging trend in the network processing market to incorporate all processing tasks (including those for security) into a single network processor. In the following, we highlight the challenges and issues particular to security processing that must be addressed in order to achieve this goal:

- **Programmability and reconfigurability:** As discussed before, flexibility is an important requirement due to the fact that algorithms and protocols tend to change over time when faced with different types of attacks. Two possible methods in achieving flexibility are programmability and reconfigurability. Programmability entails the possibility to change over time the software running on the general-purpose processor (GPP) core within the network processor. This usually involves those operations which do not require high-speed processing. Reconfigurability entails the possibility to change over time specialized hardware circuits without the need for re-designing the network processor or for the subsequent chip roll-out. The operations that requires high-speed processing are implemented

| Processor | Architecture | performance | system interface | power | Other |
|---|---|---|---|---|---|
| Layer N LNN2010 | ASIC | up to 10000 SSL/TSL connection, 2.0Gbps ARC4, 1.0Gbps 3DES | GMII Ethernet connection, DDR SDRAM 640MB, Flash memory 16MB, SRAM 2MB | 1.2 core, 2.5v digital I/O, 3.3 DPLL | Integrating TCP/IP processing |
| Corrent CR7020 | ASIC | dual OC-12 or single oc-24 SONET line rates (1.4Gbps) | 100Mhz 64-bit DDR-SDRAM, 64-bit 66/100/133 MHZ PCI-X, and POS-Phy 3 | no information available | look-aside |
| Broadcom BCM5801 | ASIC | 200Mpbs IPSec(3DES, SHA-1) | PCI 2.2 | 3.3v | look-aside |
| Motorola MDC180 | ASIC | 1024-bit DH: 10 connection/s, 155-bit ECC: 30connection/s, 3DES SHA-1: 15Mbps | 60X bus, clueless interface to MPC8xx systems or MPC 826X local bus | 1.5w | programmable key length and secure wireless communications |
| Interphase 55NS | ASIC | 512K simultaneous sessions | 64-bit 66M PCI 2.2 | 5v:109A, 3.3V:1.6A | full duplex oc-3 rate support, compress algarithms: LZS, MPPC |
| Hifn HIPP8065/8165 | ASIC | 4500 SSL session /s, 500 Mbps IPsec, upto OC-3 to OC-12 data speeds | 32/64 bit 66Mhz PCI, SDRAM interface upto 8-512 MB | 1.5v core, 3.0 I/O | compression algorithms: LZS, MPPC |
| AMD Au1550 | RISC + ASIC | no information available | 16/32bit DDR or SDRAM, PCI2.2, 2 10/100 Ethernet, USB 1.1, SPI | 500mw for the 400Mhz | 32-bit architecture, inline mode |
| Intel IXP2850 | RISC + ASIC | Up to 10Gpbs IPsec | PCI2.2, SDRAM, SRAM, SPI-4, CSIX | 27.5w - 32w for 1.4Ghz | embedded in network processor |
| Cavium NITROX II | RISC + ASIC | 1 to 10Gps IPSec, 1 to 20bps SSL | SPI3/SPI4, PCI/PCI-X, DDR SDRAM | 8w - 15w | inline mode |

TABLE I
ARCHITECTURAL COMPARISON OF SECURITY PROCESSORS.

| Processor | Symmetric-key | public-key | hashing | IPsec | SSL/TLS |
|---|---|---|---|---|---|
| Layer N LNN2010 | DES, 3DES, AES, ARC4 | 1k/2k/4k-bit RSA | MD5, SHA-1 | n | y |
| Corrent CR7020 | DES, 3DES, AES, ARC4 | randomizer for accelerating public-key encrption | SHA-1, MD5 | y, on-chip modulo engine | y, completely offload of SSL record processing |
| Broadcom BCM5801 | DES, 3DES | no | SHA-1, MD5 | y | n |
| Motorola MDC180 | DES, 3DES, ARC4 | RSA (upto 2048 bits), DH, ECC | MD5 with 128-bit, SHA-1 with 160-bit | y | y |
| Interphase 55NS | DES, 3DES, AES, ARC4 | RSA (upto 2048 bit), DH | SHA-1, MD5 | y | n |
| Hifn HIPP8065/8165 | DES, 3DES, AES, ARC4 | RSA (upto 3072 bit), DH | SHA-1, MD5 | y | y |
| AMD Au1550 | DES, 3DES, AES, ARC4 | no information available | SHA-1, MD5 | y, entire IPsec processing in hardware | y |
| Intel IXP2850 | DES, 3DES, 128-192- 256-bit AES | software | SHA-1 | hardware acceleration and software flexibility | hardware acceleration and software flexibility |
| Cavium NITROX II | DES, 3DES, 128-192- 256-bit AES, ARC4 | DH, RSA (upto 4096bit) | SHA-1, MD5 | y | y |

TABLE II
FUNCTIONALITY COMPARISON OF SECURITY PROCESSORS.

on these specialized hardware circuits.

- **Dynamic reconfiguration:** Many FPGA-based implementations [16][18][25][11] have shown that they can speed up security processing. However, the reconfigurable fabric maintains a fixed size in its lifetime and can only support a limited set of operations. Therefore, in order to further exploit the performance benefits offered by the reconfigurable fabric, methods must be sought after to allow dynamic reconfiguration, i.e., allow the functionality of the reconfigurable fabric to change during run-time of the network processor. This allows the network processor to adapt its behavior during runtime depending on the type of packets coming in which in determines the protocol/algorithm that need to be processed. In addition, we have to note that the reconfiguration must be performed as quickly as possible without allowing it to interfere with the normal operation of the network processor, i.e., decrease the performance of the network processor.
- **Instruction set design:** A common component found is most current and most likely also in future network processors is the GPP core that is mostly intended to perform those network processing operations that do not require high-speed processing. With the continued increase in performance of general-purpose processors it is a waste not to take advantage of this development by additionally supporting other high-speed network processing operations on the GPP core. Having said this, new instruction sets must be defined to provide this support. This is a complicated task, because special attention must be paid to backwards compatibility. That is, wrongly incorporated instructions must still be supported in future instruction sets which can possibly complicate the design of future generations

of network processors.

• **Wireless security processing:** It is expected that wireless networks will gain importance over wired networks in the future as they remove the need for cables and thereby increase people's mobility. Consequently, there is a growing need to incorporate security processing into such networks to protect them. However, mobile devices have a limited power supply which will be drained much faster when increasingly more functionality is incorporated into such devices. The challenge lies in deriving power-efficient security algorithms/protocols and simultaneously finding power-efficient processing solutions.

• **Ability to handle thousands of simultaneous connections:** A network processor needs to handle not only one or two connections at the same time, but many thousands of which sometimes a small and sometimes a high number of connections need to be secure. In addition, it is expected that specialized servers for e-commerce or virtual private networks (VPNs) will be dealing with increasingly more connections and users. Therefore, a scalable security processing solution must be found to deal with both situations.

• **Security processing on multimedia data:** In the near future, the transmission of multimedia will remain the dominant type of traffic over the Internet. Moreover, an increasingly larger part of such transmissions will become real-time as applications such as Voice-over-IP and video-conferencing are gaining popularity. When these transmissions must be secured, the security processing must be extended with mechanisms to deal with real-time transmission problems, e.g., delay and jitter.

Summarizing, future wired or wireless networks needs to provide a good-enough security with high performance and high flexibility. Therefore, designers are facing challenges in developing novel network processor architectures for security processing.

## VI. Conclusion

In this paper, we reviewed the main cryptographic algorithms for security processing and discussed the issues and challenges for designing a processor and system architecture for security processing. We have identified the trend in network processing to embed security processing in network processors and put them onto a single chip. Subsequently, we have surveyed existing commercial security processor solutions and compared them from a system architecture and functionality points of view. Finally, we highlighted the challenges that designers are facing in designing a novel network processor intended to perform both security processing and other network processing tasks. The challenges lies in finding solutions for the following: programmability, reconfigurability, dynamic reconfiguration, new instruction sets, wireless security processing, and secure multimedia data processing.

## References


[1] A. Stubblefield, J. Ioannidis, and A.D. Rubin. Using the Fluhrer, Mantin, and Shamir Attack to Break WEP. Proceedings of the ISOC Symposium on Network and Distributed Systems Security, 2002.

[2] AMD Alchemy au1550 Processor Product Brief. `http://www.amd.com/`.

[3] A.S. Tanenbaum. *Computer Networks*. Prentice Hall, 2003.

[4] B. Gain. Security IC Suppliers split over Encryption Methods. `http://www.my-esm.com/showArticle.jhtml?articleID=2915629`.

[5] B. Schneier. *Applied Cryptography: Protocols, Algorithms, and Source code in C*. John Wiley & Sons, Inc, 1996.

[6] BCM5801. `http://www.broadcom.com/`.

[7] Cavium Networks. `http://www.cavium.com/`.

[8] Corrent CR7020 Security Processor. `http://www.corrent.com/`.

[9] D.A. Menasce. Security Performance. volume 7, pages 84–87. IEEE Computer Society, May-June 2003.

[10] Hifn HIPP Security Session Processor 8056/8156. `http://www.hifn.com/`.

[11] I. Goodman and A.P. Chandrakasan. An Energy-Efficient Reconfigurable Public-Key Cryptography Processor. volume 36. IEEE Journal of Solid-State Circuits, November 2001.

[12] In-Stat/MDR. Security Processor Market in Perfect Position to Capitalize on Demand. `http://www.instat.com/press.asp?ID=855&sku=IN030852NT`.

[13] Intel IXP Network Processor. `http://www.intel.com/`.

[14] Interphase Corp. `http://www.iphase.com/portal/`.

[15] Introduction to SSL. `http://developer.netscape.com/docs/manuals/security/sslin/contents.htm`.

[16] K.H. Leung, K.W. Ma, W.K. Wong, and P.H.W. Leong. FGPA Implementation of a Microcoded Elliptic Curve Cryptographic Processor. In the Proceeding of the IEEE Symposium on Field-Programmable Custom Computing Machines, 2000.

[17] Layer N Network. `http://www.layern.com/`.

[18] M. Mcloone and J. McCanny. A Single-Chip IPSec Cryptographic Processor. In the IEEE Workshop on Signal Processing Systems, October 2002.

[19] MPC180 security processor. `http://e-www.motorola.com/`.

[20] N. Borisov, I. Goldberg, and D.A. Wagner. Intercepting Mobile Communications: The Insecurity of 802.11. The $7^{th}$ International Conference on Mobile Computing and Networking, ACM, 2001.

[21] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the Key Scheduling Algorithm of RC4. In $8^{th}$ Annual Workshop on Selected Areas in Cryptography, 2001.

[22] S. Kent and R. Atkinson. IP Authentication Header. In *RFC 2402*, 1998.

[23] S. Kent and R. Atkinson. IP Encapsulating Security Payload. In *RFC 2406*, 1998.

[24] S. Kent and R. Atkinson. Security Architecture for the Internet Protocol. In *RFC 2401*, 1998.

[25] S. Li, J. Torresen and O. Soraasen. Exploiting reconfigurable hardware for network security. In the Proceeding of the 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, April 2003.