A Novel Architecture and Mechanism for High-Performance Real-Time Intrusion Detection and Response System

Jintae Oh† , Byoungkoo Kim†, Seungyong Yoon†, Jong-Soo Jang†
Yong-Hee Jeon††0, and Jaecheol Ryou†††

†Electronics and Telecommunications Research Institute, Daejeon, Rep. Of Korea
††Catholic University of Daegu, Gyeongsan, Rep. Of Korea
††† Chungnam National University, Daejeon, Rep. of Korea

Summary
Many Network Intrusion Detection System(NIDS)s have been developed to detect and respond against several kinds of intrusion activities in widespread networks. Due to the explosive growth of network bandwidth, software approach in developing a high-speed NIDS is becoming impractical due to the performance constraint. Accordingly, it seems unavoidable to investigate the hardware-based solutions. Another critical problem of NIDS is a problem of false positive alerts. In order to solve these two problems, we propose a high-performance real-time intrusion detection and response system that has FPGA-based reconfiguring hardware architecture and SPI(Stateful Packet Inspection)-based intrusion detection module in the FPGA. In this paper, we present the novel architecture and mechanisms for design and implementing the system. Some experimental results are also provided.

Key words:
Intrusion detection, header lookup, SPI(Stateful Packet Inspection), pattern matching

1. Introduction

The fast extension of computer networks has increased the problem of unauthorized access and tampering with data [1]. In order to encounter with the increased threats, many NIDSs have been developed to serve as the last line of defense in the overall protection scheme of information system. These NIDSs have two major approaches; misuse intrusion detection and anomaly intrusion detection [2,3]. However, most of existing NIDSs, such as Snort [4], NFR [5], and NetSTAT [6], only employs the misuse detection approach for reducing the degradation of performance to the minimum level. Also, most of NIDSs based on misuse detection approach has concentrated on catching and analyzing only the audit source collected on Fast Ethernet links. With the advancement of network technology, Gigabit Ethernet has become the actual standard for large-scale network deployment. However, the existing NIDSs have problems for the performance, thus becoming a bottleneck due to the overhead in collecting and analyzing data in a specific component. Therefore, the effort to enhance the performance of NIDS on high-speed links has been the focus of much research in the intrusion detection community. Several NIDSs, such as Real-Secure [7], ManHunt [8], and CISCO IDS [9], that is run on high-speed links were developed. Nevertheless, there is an emerging need for security analysis techniques that can keep up with the increased network throughput [10].

Another limiting factor with the NIDS is the high false positive alert rate. In order to reduce these false positive alerts, a lot of methods and techniques are proposed. Stateful Packet Inspection (SPI) is one of these solutions. SPI was originally developed for Firewall [11,12], but it became a very important factor in the NIDS. Stateless NIDSs generate tremendous false positive alerts while ‘stick’ or ‘snot’ attempts to attack [13,14]. Most existing NIDSs have SPI module which supports statefulness, but they don’t satisfy high-performance in gigabit Internet environment. It is challenging problems that we manage a lot of session state information with limited hardware resource and satisfy performance of high-speed Internet. In other words, the rapid evolution of recent network technologies to gigabit network environments require existing SPI module to have more improved functions and performance. SPI basically requires a session table which stores source and destination IP addresses and port numbers. It is necessary to perform real-time packet inspection by checking, for each input packet, whether or not a corresponding entry is present in the session table. Real-time packet processing at wire speed should not cause any packet delay or loss even when the number of managed sessions is increased to more than one million [15].

Focusing on the problems of both performance and false positive alerts mentioned above, this paper presents a Gigabit IDS to detect and respond against attacks on the

Manuscript received March 5, 2008
Manuscript revised March 20, 2008
– 0: The Corresponding Author

high-speed network. It is possible through FPGA(Field Programmable Gate Array)-based intrusion detection technique. To guarantee both performance and functionality with respect to statefulness, we designed and implemented SPI-based intrusion detection module in a FPGA to help alleviating a bottleneck in network intrusion detection systems. The performance of SPI-based intrusion detection system mainly depends on the performance of processing session table [16] and pattern matching [17]. In this paper, we only focus on session state management scheme.

The remainder of this paper consists of as follows. In Section 2, we present a novel architecture of the proposed NIDS. Section 3 presents SPI-based detection scheme. Then, Section 4 describes the efficient detection techniques implemented in FPGA-based reconfiguring hardware. Section 5 introduces the prototype we have developed, and some experiment and simulation results are given. Finally, we conclude and suggest directions for further research in Section 6 [10,15].

2. Architecture

2.1 System Architecture and Components

As shown in Fig.1, the proposed system is aimed at real-time network-based intrusion detection based on misuse detection approach [10]. It mainly consists of two parts; Application Task for policy management, alert management and system management, and Security Engine Board for wire-speed packet forwarding, packet preprocessing, high-performance intrusion detection and response. Most of all, in order to detect network intrusions more efficiently on high-speed links, Security Engine Board is composed of several sub-FPGA Logics. Here, communication between Security Engine Board and Main CPU is run through PCI interface. Communication through PCI interface is divided into two channels. One is a channel for policy enforcement. The other is a channel for alert information transmission. Through the interoperability of these components, our system analyzes data packets as they travel across the network for signs of external or internal attack. That is, the major functionality of our system is to perform the real-time traffic analysis and intrusion detection on high-speed links. Therefore, we focus on effective detection strategies applied to FPGA Logics.

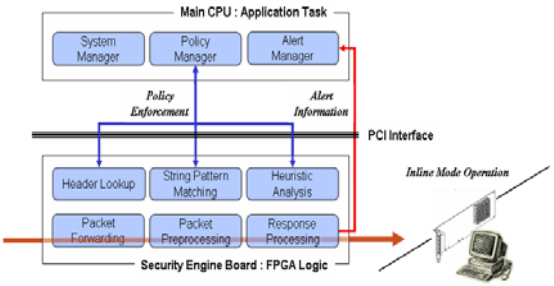

Fig. 1 System Architecture and Components

2.2 Security Engine Board

Security Engine Board is composed of three FPGA chips and one FPGA chip for PCI interfacing, as shown in Fig. 2.

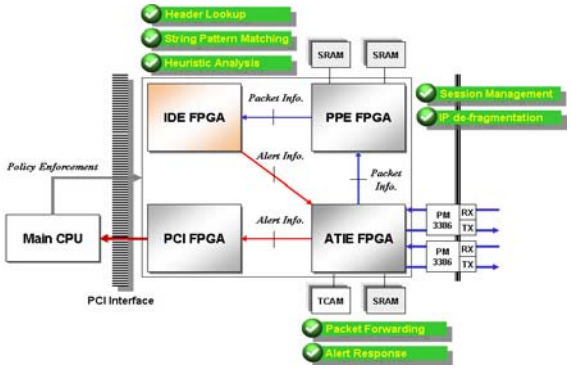

Fig. 2 The FPGA Chips Arrangement for Security Engine Board

The main characteristics of each component are as follows [10]:

- ATIE(Anomaly Traffic Inspection Engine) FPGA

It uses the XILINX XC2VP70 FPGA chip and is connected to the PM3386 for incoming packet forwarding. Also, it uses external TCAM and SRAM for incoming packet scheduling and management. The main function of ATIE is the wire-speed packet forwarding and response coordinating such as alert message generation and packet filtering. Basically, incoming packets from PM3386 is sent to PPE FPGA chip, and if it is determined as attack according to the analysis result from other FPGA chips, alert information is sent to the main CPU through FPGA chip for PCI interfacing.

- PPE(Pre-Processing Engine) FPGA

It uses the XILINX XC2VP50 FPGA chip and two external SRAMs for session management, IP de-fragmentation and TCP reassembly. The main function of PPE FPGA chip is to process the prior steps for intrusion detection. The preprocessing function supports the

SPI(Stateful Packet Inspection) based intrusion detection and IDS evasion attack detection.

- IDE(Intrusion Detection Engine) FPGA

It uses the XILINX XC2VP70 FPGA chip and three mechanisms for high-performance intrusion detection; Flexible Header Combination Lookup Algorithm for packet header pattern matching, Linked Word-based Store-less Running Search Algorithm for string pattern matching about packet payload, and Traffic Volume-based Heuristic Analysis Algorithm for DoS(Denial of Service) and Port-scan attack detection. Through these mechanisms, it executes the high-performance intrusion detection without packet loss.

The description of high-performance intrusion detection mechanisms are given in Section 4.

3. SPI(Stateful Packet Inspection) Scheme

The SPI-based intrusion detection module is illustrated in Fig. 3 [15]. Legitimate TCP sessions are established through 3-way handshakes and terminated through 4-way handshakes. State manager has session table and tracks these session states. If input packet doesn't exist in session entries, this packet will drop or forward to Intrusion Detection Engine(IDE) with additional state information according to security policies.

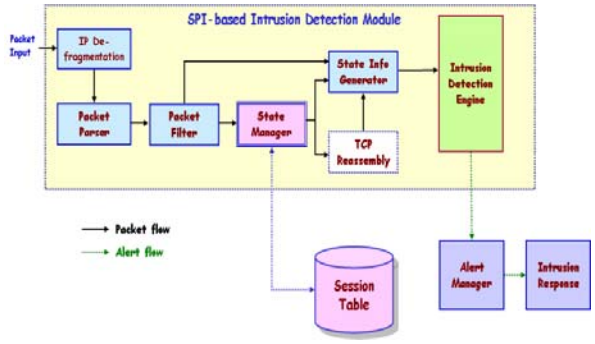

Fig. 3 SPI-based Intrusion Detection Module

A basic architecture of session state manager for stateful packet inspection is depicted in Fig. 4. Session state manager includes a hash key generator, a session table, a session detection module, a session management module, and a state information generation module. The session table stores session entries that are indexed and managed by the hash key generator. The input is 4-tuple information-<source IP address, a destination IP address, a source port, a destination port> and it is used as information to hash a newly received packet, to the hash key generator. Hash key generator has a dual hash structure with two different hash functions Hash1(x) and Hash2(x). The hash functions Hash1(x) and Hash2(x) are well-known functions that are used to hash packets. One hash function Hash1(x) is used to generate indices that point to hash sets permitting hash collisions in order to achieve faster session table search. The other hash function Hash2(x) is used to generate hash addresses that are used to identify session entries in a hash set pointed by the hash function Hash1(x). Session table may be designed and implemented using two or more SRAM devices, if necessary. For efficient session table management, the session table has an N-way set associative session table structure in which each hash set in the session table can include N session entries. The session table shown in Fig. 4 is a 32-way set associative session table that is constructed using two 72-Megabit SRAMs with each session entry having a length of 36 bits [15].

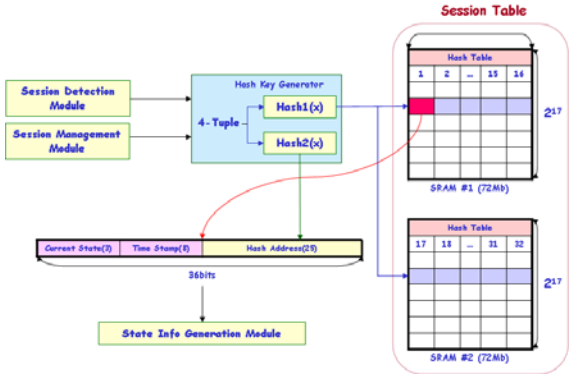

Fig. 4 Basic Architecture of Session Manager

Current state, time stamp, and hash address parts are stored in each session entry of the session table. The current state part includes current connection state information of a corresponding session, the time stamp part is used to determine which session entry is to be deleted when the session table is full, and the hash address part is used to identify each session entry in the same hash set. According to current state, State Info Generation Module generates the state information of the packet. Then, inspection of the packet is performed based on the generated state information.

4. High-Performance Intrusion Detection Mechanisms

The detection mechanism of the proposed system is mainly run on the IDE FPGA chip, as illustrated in Fig. 2. In order to achieve effective high-performance intrusion detection, the system has three detection mechanisms; the header lookup mechanism for flexible header combination

lookup, the string pattern matching mechanism about packet payload, and the heuristic analysis mechanism for DoS and Port-scan attack detection [10].

4.1 Header Lookup Mechanism

This mechanism is performed by flexible header combination lookup algorithm. It compares pre-defined header related rule-sets with header information of incoming packets. If the incoming packet matches with the existing header patterns, the final 256 bit result is sent to String Pattern Matching Logic and traffic volume-based Heuristic Analysis Logic, as shown in Fig. 5. The number of different header combination about pre-defined header related rule-sets is thus limited to 256.

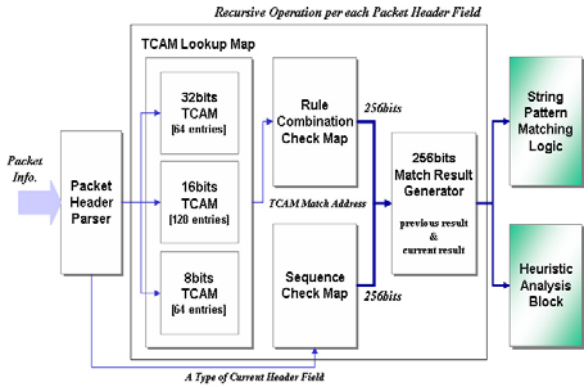

Fig. 5 Header Lookup Mechanism

This algorithm uses three memory maps; internal TCAM(Ternary Content Addressable Memory) lookup map, rule combination check map and sequence check map. - The internal TCAM lookup map is composed of three TCAM; 8bits lookup map for 8bits header fields such as ICMP type and TCP flags, 16bits lookup map for 16bits header fields such as service port value, and 32bits lookup map for 32bits header fields such as IP address field. The match address of these lookup maps is used by rule combination check map.

- The rule combination check map is composed of combination map of 256*256. That is, the 256bit result of rule combination check map, which is indicated by match address from TCAM lookup map, presents the rule-set information of current matched field. For example, if the match result of ICMP type field is "{255{2'b0}, 2'b1}", the first rule-set(header combination) is to be matched.

- The sequence check map is composed of sequence map of 32*256, and includes don't care information of current matching field. Here, '32' means the number of header field types. For example, if don't care information of ICMP type field is "{255{2'b0}, 2'b1}", the first rule-set is always to be matched irrespective of TCAM matching. The 256bit result of this map is combined with result of rule combination check map. Basically, this mechanism is performed recursively about all packet header fields of incoming packet. The final 256bit result is referred by logics for String Pattern Matching Logic and traffic volume-based Heuristic Analysis Block.

4.2 String Pattern Matching Mechanism

String pattern matching mechanism about packet payload is performed by store-less running search algorithm based on linked word. It compares pre-defined packet payload related rule-sets with packet payload information of incoming packets. If the incoming packet matches with existing payload patterns, an alert message is generated in combination with the header lookup result. For this operation, this algorithm uses the pattern reconstruction technique. As shown in Fig. 6, reconstruction substring length of each pattern has boundary length of 5 or 7 because of the limit of block memory in FPGA chip. The first 5byte substring of "/bin/echo" pattern is equal to the first 5byte substring of "/bin/kill" pattern and "/bin/chmod" pattern. Therefore, "/bin/" substring of these patterns is stored in the same memory space. Similarly to this, other patterns are also reconstructed. Through such a pattern reconstruction, the system can have approximately 2,000~3,000 rule-sets in the limited memory storage on the FPGA chip.

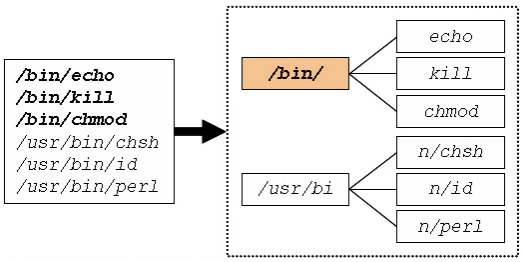

Fig. 6 Payload Pattern Reconstruction

After pattern reconstruction as described before, linked word-based store-less running search algorithm is performed as string pattern matching mechanism. This algorithm uses the spectrum dispersion technique as shown in Fig. 7. The technique is a method to calculate unique hash value about reconstructed substrings. For example, "/etc/" substring of 5byte length has the 9bit hash value by a given hash formula. Hash values calculated about each substring are used as a storing memory address for each substring. The pattern

reconstruction and hash memory allocation is performed by main CPU when the system booting is run in advance. After system booting, the IDE FPGA logic performs the hash value calculation about the incoming packet to the unit of byte. If the payload string in incoming packet matches with the substring in memory pointed by the calculated hash value, then it is checked out whether the related reconstructed substrings match or not. If all connected substrings of one pattern matches with incoming packet, alert message is generated in combination with the header lookup result. Through these operations, our system performs the string pattern matching operation without degradation of performance and packet loss.

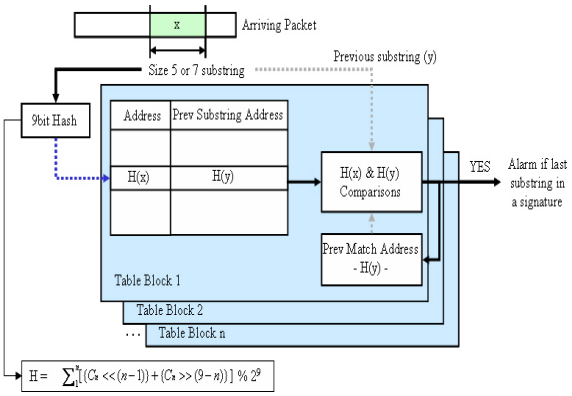

Fig. 7 Linked Word-based Store-less Running Search Algorithm

Based on the simulation result about our string pattern matching technique [18], it is revealed that the proposed technique is memory efficient outperforming the previous techniques. The substring length of 5 or 7 with a small hash table showed to consume a minimum amount of memory in storing SNORT rule-sets.

4.3 Heuristic Analysis Mechanism

This mechanism is performed by traffic volume and time threshold-based analysis algorithms. Similar to the pattern matching mechanism, this algorithm also compares pre-defined rule-sets with packet information of incoming packets. However, this mechanism generates alert message by traffic volume within time threshold. If the incoming packet matches with existing rule-set, then count value of the rule-set is increased, and count threshold and time threshold are checked out. If the count threshold is exceeded within time threshold, alert message is generated. Through this mechanism, the system is capable of detecting the DoS and Port-scan attacks.

5. Implementation and Test Results

5.1 Implementation

The system is implemented in programming languages that is best suited for the task it has to perform. Basically, application tasks of the system are implemented in C programming language. FPGA Logic of the system is implemented in Verilog HDL(Hardware Description Language) that is best suited for high-speed packet processing. Most of all, the system focus on FPGA logic for real-time intrusion detection on high-speed links. Also, we employed inline mode capable of effective response by using four Gigabit Ethernet links. That is, the system has developed in the consideration of improvement in performance for packet processing [10].

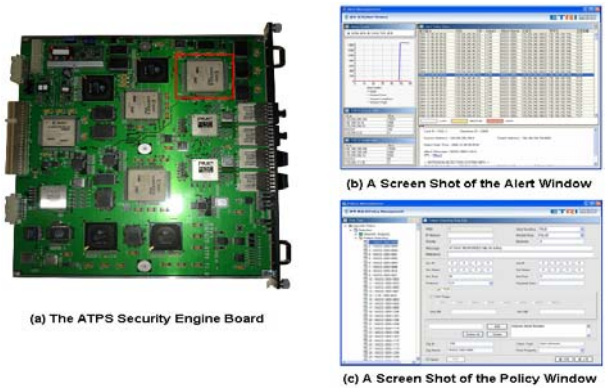

Fig. 8 Examples of Implementation

As shown in Fig. 8 (a), we named the proposed system as ATPS(Adaptive Threat Prevention System) [10]. The system was implemented in a XILINX Virtex-II Pro flatform FPGAs. Most of all, The IDE FPGA device, XC2VP70, has 74,448 logic cells and 5.9Mbits of on-chip SRAM, which is a configurable block select memory. In the Fig. 8, it is marked with the red square. Also, the screen shots were captured during experiments to validate the performance of the prototype. The screen shot (b) shows that web-related attacks were detected. The next screen shot (c) shows that rule-sets for intrusion detection and response were applied.

For SPI, the SPI-based intrusion detection module was also implemented on the prototype. Session State Manager of the system is implemented on a Xilinx Vertex-II Pro XC2VP70 FPGA(7M Gate) and Cypress CY7C1470V33

SRAM(72Mbit) using Verilog HDL(Hardware Description Language) that is best suited for high-speed packet processing.

5.2 Experiment and Simulation Results

The Snort rule-sets are applied for the performance evaluation of the proposed system in this paper. We used IXIA Traffic Generator for background traffic generation, IDS Informer Attack Tool and Nessus Vulnerability Scanner for attack traffic generation. At first, we observed the rate of alert generation when background traffic generated by IXIA increases gradually. That is, we measured the decrease in effectiveness of the detection when the traffic rate increases. As shown in Fig. 9 (a), increasing traffic rate does not have an effect on detection rate. Next, our experiment was run with a constant traffic rate of 100Mbps and an increasing number of signatures. The experiment starts with only the 200 rule-sets that are needed to achieve maximum detection for the given attacks. As shown in Fig. 9 (b), increasing number of signatures also does not have an effect on detection rate of our system. The previous two experimental procedures using Snort sensors as we adopted are also performed by Kruegel et al. [19]. Compared with Snort sensor, our system showed a consistent performance in traffic level and was independent with increasing number of signatures applied [10].

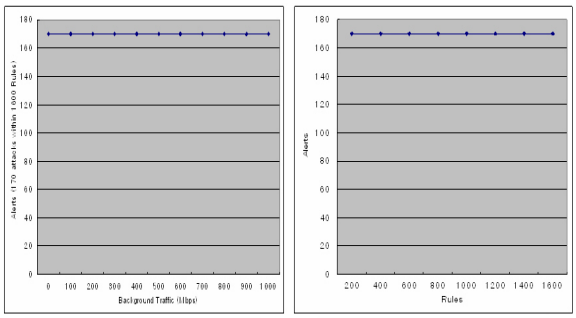

Fig. 9 Results of Performance Tests

Our session state management scheme in SPI-based intrusion detection system is affected by two major factors, hash collision rate and push-out rate. There is every probability of hash collision occurrence because hash function for faster session table search is used. The wrong state information is generated if the hash collision is occurred. Therefore, the SPI-based intrusion detection module generates the false positive alert. The hash collision rate is determined by the Hash2(x). Theoretically, the probability of hash collision is $1/2^{20}$ if the session table is full. As the number of session entries increase gradually, the session table is filled with new session. Also, there is every probability of push-out occurrence because the size of hash set has limitation (32-way set). When the session table is full, the probability that each session is brought into a push-out state is very important in a session table management scheme because wrong session state information is generated if any existing session, which has not yet been terminated, is replaced with a new session. In this case, since the SPI-based intrusion detection module generates the false negative alert, the push-out rate can be said to be the factor which is important than the hash collision rate. In order to ensure that push-out rate is reasonable in our design, we made a simulation for distribution of the number of sessions allocated to each hash set in the session table when one million sessions are established. We used a separate set of traffic data collected from various network environments for this simulation.

Distribution of the number of sessions allocated to each hash set in the session table follows a normal distribution as expressed by Probability density function. This is standardized and then the push-out probability of each session in the 32-way set associative session table is calculated to obtain $P\{X>32\} = P\{Z>8.3\}$. This indicates Z-score of 8.3 which is nearly 0%. (Z-score of 6 corresponding probability is 0.0003%) According to the result of simulation, it is proved that our design for session state management is very reasonable with respect to hash collision rate and push-out rate [15].

6. Conclusion and Further Research

Due to the increasing link speed, the number of attack patterns, and signatures to be maintained, it is a challenging issue to provide a seamless protection for secure network service. In this paper, we presented the architecture of our system that performs the real-time traffic analysis and intrusion detection on high-speed links, and proposed the novel detection mechanisms in FPGA-based reconfiguring hardware that supports more efficient intrusion detection. We have implemented the prototype of our system for the analysis of the traffic carried by a Gigabit link. Most of all, our system focus on reducing a degradation of performance caused by high-speed traffic analysis to the minimum level. Therefore, it is run by the FPGA logic proposed for improvement in performance. Also, it has the advantage that is capable of supporting the effective response by using inline mode monitoring technique on four Gigabit links.

In this paper, we also proposed session state management scheme which can perform stateful packet inspection in real time by performing session table processing that allows more efficient generation of state

information. And we designed and implemented SPI-based intrusion detection module in a FPGA to help alleviating a bottleneck in network intrusion detection systems.

However, the current prototype needs some improvement and a thorough evaluation to be deployed in a real-world environment. In order to resolve the problem derived from the verification of implemented system, it is necessary to upgrade system performance and availability, and to perform faults-tolerance test with prototype. Also, we need to keep up much effort for improvement in performance of detection mechanism on high-speed links. We hope to implement and expand our designed system and give more effort to demonstrate effectiveness of our system.

References

[1] Byoung-Koo Kim, Ik-Kyun Kim, Ki-Young Kim and Jong-Soo Jang, “Design and Implementation of High-Performance Intrusion Detection System”, In Antonio Lagana et al., editors, Proceedings of the 2004 International Conference on Computational Science and its Applications (ICCSA 2004), volume 3046 of LNCS, pp. 594~602, Assisi, Italy, 14~17 May, 2004. Springer-Verlag.

[2] H. Debar, M. Dacier and A. Wespi, "Research Report Towards a Taxonomy of Intrusion Detection Systems", Technical Report RZ 3030, IBM Research Division, Zurich Research Laboratory, Jun., 1998.

[3] S. Kumar and E. Spafford, "A pattern matching model for misuse intrusion detection", In Proceedings of the 17th National Computer Security Conference, pp. 11-21, Oct., 1994

[4] M. Roesch. "Snort-Lightweight Intrusion Detection for Networks". In Proceedings of the USENIX LISA ’99 Conference, November, 1999.

[5] Marcus Ranum, "Burglar Alarms for Detecting Intrusions", NFR Inc., 1999.

[6] Thomas Ptacek and Timothy Newsham, "Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection", Secure Networks Inc., 1998.

[7] ISS. RealSecure Gigabit Network Sensor. http://www.iss.net/products_services/enterprise_protection/rsnetwork/gigabitsensor.php, September, 2002.

[8] Symantec. ManHunt. http://enterprisesecurity.symantec.com/products/products.cfm?Product ID=156, 2002.

[9] CISCO. CISCO Intrusion Detection System. Technical Information, November, 2001.

[10] Byoungkoo Kim, Seungyong Yoon, and Jintae Oh, “ATPS-Adaptive Threat prevention System for High-Performance Intrusion Detection and Response”, APNOMS 2007, LNCS 4773, pp.344-353, 2007.

[11] http://www.checkpoint.com, Firewall-1 Product

[12] Lance Spitzner, Understanding the FW-1 State Table, http://www.spitzner.net/fwtable.html

[13] Brian Caswell, Jay Beale, James C. Foster, Jeremy Faircloth, Snort 2.0 Intrusion Detection(Syngress Publishing, February 2003)

[14] http://www.snort.org, Snort Preprocessor Stream4

[15] Seungyong Yoon, Byoungkoo Kim, Jintae Oh and Jongsoo Jang, “High Performance Session State Management Scheme for Stateful Packet Inspection”, APNOMS 2007, LNCS 4773, pp. 591-594, 2007.

[16] Xin Li, Zheng-Zhou Ji, and Ming-Zeng Hu, Stateful Inspection Firewall Session Table Processing, Proc. Of the International Conference on Information Technology: Coding and Computing(ITCC’05), Volume 2, April 2005, Pages:615-620

[17] Sergei et al., SNORTRAN: An Optimizing Compiler for Snort Rules, Fidelis Security Systems, Inc., 2002.

[18] Sungwon Yi, Byoung-koo Kim, Jintae Oh, Jongsoo Jang, George Kesidis and Chita R. Das, "Memory-Efficient Content Filtering Hardware for High-Speed Intrusion Detection Systems ", Proceedings of the 2007 ACM Symposium on Applied Computing, pp. 264-269, Seoul, Korea, 11~15 March, 2007.

[19] Kruegel, C., Valeur, F., Vigna, G. and Kemmerer, R. "Stateful intrusion detection for high-speed networks", In Proceedings of the IEEE Symposium on Security and Privacy, pp. 266-274, 2002.

Jin-Tae Oh received the B.S and M.S degrees in Electronics Engineering from Kyungpook National University in 1990 and 1992, respectively. He worked at ETRI (Electronics and Telecommunications Research Institute) from 1992 to 1998. During 1998-1999, he stayed in MinMax Tech , USA, as a Research staff. He served as a Director in Engedi Networks, USA, during 1999-2001. He was both Co-founder and CTO Vice President in Winnow Tech. USA during 2001-2003. From 2003, he works with the Security Gateway Team, ETRI, Daejeon, Korea.

Byoung-Koo Kim received the B.S. and M.S. degrees in Information and Communication Engineering from Sungkyunkwan University in 1999 and 2001, respectively. Since 2001, he has stayed in Security Gateway System Team, Electronics and Telecommunications Research Institute(ETRI) of Korea to study Network Security related Topics.

Seung-Yong Yoon received the B.S. and M.S. degrees in Computer Engineering from Chungnam National University in 1999 and 2001, respectively. Since 2001, he has stayed in Security Gateway System Team, Electronics and Telecommunications

Research Institute(ETRI) of Korea to study Network Security related Topics.

Jong-Soo Jang received the B.S and M.S degrees in Electronics Engineering from Kyungpook National University in 1984 and 1986, respectively. He received his Ph. D degree in Computer Engineering from Chungbuk National University in 2000. Since 1989, he has been working with ETRI, Daejeon, Korea and now is the Director of Applied Security Group. Since January 2008, he has been a Vice-President of KIISC(Korea Institute of Information Security and Cryptology)

Yong-Hee Jeon received the B.S degree in Electrical Engineering from Korea University in 1978 and the M.S and Ph. D degrees in Computer Engineering from North Carolina State University at Raleigh, NC, USA, in 1989 and 1992, respectively. From 1978 to 1985, he worked at Samsung and KOPEC(Korea Power Engineering Co.). Before joining the faculty at CUD (Catholic University of Daegu) in 1994, he worked at ETRI(Electronics and Telecommunications Research Institute) from 1992 to 1994. Currently, he is a Professor at the School of Computer and Information Communications Engineering in the CUD, Gyeongsan, Korea. Since January 2008, he has been a Vice-President of KIISC(Korea Institute of Information Security and Cryptology)

Jaecheol Ryou is a Professor in the Division of Electrical and Computer Engineering at Chungnam National University in Korea. He is also the director of the Internet Intrusion Response Technology Research Center (IIRTRC), Chungnam National University, Korea. He received the B.S. degree in Industrial Engineering from Hanyang University in 1985, the M.S. degree in Computer Science from Iowa State University in 1988, and the Ph.D. degree in Electrical Engineering and Computer Science from Northwestern University in 1990. His research interests are Internet Security and Electronic Payment Systems including Wireless Internet Security.