An Efficient Multi-hash Pattern Matching Scheme for Intrusion Detection in FPGA-based Reconfiguring Hardware

Byoungkoo Kim, Seungyong Yoon, and Jintae Oh
Security Gateway System Team, Electronics and Telecommunications Research Institute
161 Gajeong-dong, Yuseong-gu, Daejeon, 305-700
Republic of KOREA

Abstract: Many Network-based Intrusion Detection Systems (NIDSs) are developed till now to respond these network attacks. As network technology presses forward, Gigabit Ethernet has become the actual standard for large network installations. Therefore, software solutions in developing high-speed NIDSs are increasingly impractical. It thus appears well motivated to investigate the hardware-based solutions. Although several solutions have been proposed recently, finding an efficient solution is considered as a difficult problem due to the limitations in resources such as a small memory size, as well as the growing link speed. Therefore, we propose the FPGA-based intrusion detection technique to detect and respond variant attacks on high-speed links. It was designed to fully exploit hardware parallelism to achieve real-time packet inspection, to require a small memory for storing signature. The technique is a part of our system, called ATPS (Adaptive Threat Prevention System) recently developed. Most of all, the proposed system has a novel content filtering technique called Table-driven Bottom-up Tree (TBT) for exact string matching. But, as the number of signatures to be compared is growing rapidly, the improved mechanism is required. In this paper, we present the multi-hash based TBT technique with memory-efficiency. Simulation based performance evaluations showed that the proposed technique used on-chip SRAM less than 20% of the one-hash based TBT technique.

Key-Words: Intrusion Detection, Pattern Matching, Memory-efficiency

1 Introduction
The fast extension of inexpensive computer networks also has increased the problem of unauthorized access and tampering with data. As a response to increased threats, many NIDSs have been developed to serve as a last line of defense in the overall protection scheme of a computer system. Also, most NIDS has concentrated on catching and analyzing only the audit source collected on Fast Ethernet links. However, with the advancement of network technology, Gigabit Ethernet has become the actual standard for large network installations. Therefore, the effort of performing NIDS on high-speed links has been the focus of much debate in the intrusion detection community, and several NIDSs that are run on high-speed links actually have been developed [3]. But, these NIDSs are still not practical because of technical difficulties in keeping pace with the increasing network speed, and real-world performance also will likely be less. Therefore, there is an emerging need for security analysis techniques that can keep up with the increased network throughput. Most of all, content filtering problems have been extensively studied, and many of the proposed solutions are based on the general purpose pattern matching algorithms such as Boyer-Moore (BM)[10], Aho-Corasick (AC)[11], and Wu-Manber (WM)[12]. Although these algorithms greatly improve pattern matching speed compared to naive string matching techniques, providing real-time content filtering in high-speed networks is still challenging. In the previous work, we proposed our Gigabit IDS, called ATPS, to detect and respond attacks on the high-speed network [1]. The proposed system has a novel content filtering technique called TBT [2]. This paper presents the improved mechanism, which used on-chip SRAM less than 20% of the previous TBT technique.

The remainder of the paper is structured as follows. The next section presents related works about early studies of NIDS. Then, section 3 presents the architecture of our system, and describes the multi-hash based TBT technique with memory-efficiency. Section 4 shows the simulation based performance evaluations about the improved TBT technique, which is compared with the previous technique. Section 5 briefly introduces the prototype that we have developed. Finally, we conclude and suggest directions for further research in section 6.

2 Related Work
With the widespread use of the Internet, IDSs have become focused on network attacks. Therefore, most IDSs employed network-based IDS [14]. NIDS uses the network as the source of security-relevant information.
Essential to almost every NIDS is the ability to search through packets and identify content that matches known attacks. As network technology presses forward, space and time efficient string matching techniques have been important for identifying these packets at line rate. Therefore, software solutions in developing high-speed NIDSs are increasingly impractical. It thus appears well motivated to investigate the hardware-based solutions.

With the advance of hardware technologies, there have been several attempts to catch up the line speed using Field Programmable Gate Arrays (FPGAs). Cho et al. [6] proposed to use parallelized patterns (rule units) to find matches in a four-byte input stream on every clock cycle. In addition, Sidhu et al. [5] and Moscola et al. [13] mapped regular expressions into a FPGA using Nondeterministic Finite Automation (NFA) to minimize the space required for pattern matching. However, hardware resource consumption linearly increases as the number of patterns and the number of characters to be stored increase. Another approach to achieve the high-speed content filtering is to use a fast parallel pattern matching circuit, called a Content Addressable Memory (CAM). Although the CAM-based solutions are relatively easy to implement, and can exploit fine grain pipelining in general, high hardware cost and power consumption, as well as high area cost are the main concerns.

As appeared in the related works, the content filtering at gigabit line speeds is still a challenging problem due to the limitations in resources such as memory size and the growing line speed. In addition, the number of signatures to be compared is growing rapidly and thus giving the signature explosion problem. With current processor and hardware technologies, neither increasing the speed of network processor nor providing more network processors to keep up with the multi-giga bits links is practical and cost-efficient. In addition, the number of signatures to be compared with the payload of each packet has been doubled during the last two years. For example, the number of signatures in SNORT [7] has increased from 1,500 in 2003 to 2,800 in 2005. Therefore, software-only-solutions in developing high-speed NIDS are increasingly impractical. It thus appears well motivated to investigate the hardware-based solutions. In particular, several content filtering techniques have been proposed to used FPGAs for taking advantage of the hardware parallelism and the configurable nature of the device [4], [5], [6]. Another concern in FPGA-based solutions is that an on-chip SRAM is generally favored for constructing signature database primarily due to the faster access time compared to an off-chip SRAM.

However, the size of an on-chip SRAM is generally hundreds of kilobytes, whereas the space required for storing only all the characters in the latest SNORT signatures is almost 512 Kilobytes. Therefore, finding efficient techniques, which can minimize the memory consumption, is becoming a key success factor in the development of high-speed NIDSs.

This paper presents a novel FPGA-based content filtering technique, called TBT, which is a multiple hash implementation of a bottom-up tree. The hashing scheme helps in reducing the data access time, while the bottom-up tree helps in minimizing the memory consumption in TBT. The TBT was evaluated through extensive simulations, and implemented in a XILINX FPGA, XC2VP70 [8].

3 System Description

In this section, we briefly introduce the architecture of our system and components of the architecture called ATPS. Then, we present the multi-hash based TBT technique with memory-efficiency.

3.1 System architecture

Our system is aimed at real-time network-based intrusion detection based on misuse detection approach [15]. As shown in the figure 1, the proposed system 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. Here, Security Engine Board is composed of several sub FPGA Logics, and we focus on effective detection strategies applied FPGA Logics for performing the real-time traffic analysis on high-speed links.

3.2 Multi-hash based TBT mechanism

In the previous work, we present our novel content filtering mechanism, so called TBT, which is one hash implementation of a bottom-up tree [2]. To detect certain patterns from packets, the attack patterns (signatures)
have to be effectively stored in the memory to minimize memory consumption and to facilitate data retrievals. Therefore, we proposed the design concepts of the TBT mechanism in terms of the data structure and the algorithm. The solution that we propose takes advantage of a bottom-up tree to maximize the memory efficiency, and uses multiple hash tables to improve the run-time performance of the content filtering algorithm. All hash tables used in TBT share the same structure, and a slot in the hash tables consists of four fields: substring, previous node pointer (Prev_Ptr), flags to indicate the head node and the tail node, and a field to store signature specific information such as signature description. For example, figure 2 and figure 3 illustrate how a bottom-up tree is constructed when k is 5 or 7. As shown in the figure 2, the first five-byte substring of “/bin/echo” pattern is equal to the first five-byte substring of “/bin/kill” pattern. Therefore, “/bin/” substring is stored in the same memory space. Through this tokenizing, other patterns are also constructed. Here, a way to improve memory utilization is to use a bottom-up tree, where each child node points to the address of its parents.

Identifies the signature. First, a signature, “/bin/echo”, is divided into five-byte substrings, “/bin/” and “echo”, and then the substrings are hashed to find the tokens, 30 and 10 respectively. Finally, “/bin/” and “echo” are inserted at slots 30 of the hash table in Table Block 1 (T1) and slots 10 of the hash table in Table Block 4 (T4). Then, the link between the adjacent tokens is represented by Prev_Ptr. Similarly, other signatures are stored in each Table Blocks.

Fig. 4 Multi-hash based Signature Table Configuration

Here, the value of slot 180 induces the hash collision problem. For example, “motd” collides with the previously inserted key of “kill”. In TBT, collisions can be resolved simply by searching all the hash tables simultaneously and using the first table with unoccupied slots for the corresponding token. Therefore, the new key is inserted in other same copying Table Block. Although the sequential search on multiple tables is an expensive operation when implemented with software, it can be easily converted into a parallel search with hardware. However, the more it does if the hash collision like figure 3 frequently happen, the waste of the hardware resource becomes severe. Therefore, we propose the multi-hash based TBT mechanism. If hash collision happens frequently by 1st hash function, signature table configuration by multi-hash based TBT can be performed by 2nd hash function as shown in the figure 4. That is, although the key by 1st hash function collides with the previous inserted key, the key by 2nd hash function can be used. Here, the 1st hash function and the 2nd hash function are the hash function which is suitable to the hardware implementation.

After above signature table configuration, linked word based store-less running search algorithm is performed as string pattern matching mechanism. This algorithm uses the hardware architecture of multi-hash based TBT scheme as shown in the figure 5. When a packet arrives, the first k-byte substring is extracted from the beginning of the payload. For the successive substrings, the k-byte window shifts toward the end of the payload one byte at a time. For each substring, the multiple hash function generates an index, and the corresponding slots in all the hash tables are accessed simultaneously. In figure 5, the simultaneous accesses for the multiple hash tables are
represented with dashed-lines. If the key values in the accessed slots match the substring (of the packet) and the addresses of the accessed slots match the values in the previous slot address registers (PSA registers), then the registers are updated with the matched slot addresses. Otherwise, the PSA registers are cleared. Here, the PSA register is cross-referred by each Table Blocks, and PSA register comparison is performed only when the matched slot is not the first substring, called the head substring, in a signature. An attack is detected when the last substring (the tail substring) in a signature is matched.

Through these operations, our system performs the string pattern matching operation without lowering of performance and packet loss.

4 Performance Evaluation

The approach that we presented here is a part of an Intrusion Detection System, called ATPS recently developed. A prototype of multi-hash based TBT was implemented in a XILINX Virtex-II Pro platform FPGA. The FPGA device, XC2VP70, has 74,448 logic cells and 5.9Mbits of an on-chip SRAM, which is a configurable block select memory. There are various ways to configure the block memory in the XC2VP70 FPGA to implement TBT. As shown in the previous work, 512 entries per hash table h with the substring length k of 1~7 bytes showed the best performance in terms of the memory consumption and utilization. However, Implementing 4~7 bytes wide hash tables requires two memory blocks in parallel, since the single block width of XC2VP70 with 512 entries is 4.5 bytes.

The current implementation of our prototype is capable of performing only single-content filtering. Therefore, 1660 signatures except signatures with header only and multiple contents are loaded in the prototype system. Here, Table 1 summarizes the amount of the block memory used by the one-hash based TBT implementation. Next, Table 2 shows it by the multi-hash based TBT implementation. As shown in Table 1 and 2, multi-hash based TBT used 68 block memories, which is less than 20% of the previous one-hash based TBT.

5 Implementation

We have developed Gigabit IDS based on our architecture, called ATPS. Our system is implemented in programming languages that is best suited for the task it has to perform. Basically, application tasks of our system are implemented in C programming language. FPGA Logic of our system is implemented in verilog HDL (Hardware Description Language) that is best suited for high-speed packet processing. That is, our system has developed in the side of improvement in performance for packet processing. As shown in the figure 6 (a), our system was implemented in a XILINX Virtex-II Pro platform FPGAs. In the above figures, it is marked with

![Fig. 5 The hardware architecture of Multi-hash based TBT](image-url)
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.

Fig. 6 The Prototype of ATPS

6 Conclusion

Providing seamless protection for secure network service is becoming difficult primary because of the increasing link speed and the number of attack patterns, signatures to be maintained. In this paper, we designed the architecture of our system, called ATPS that performs the real-time traffic analysis and intrusion detection on high-speed links, and proposed the novel detection mechanism in FPGA-based reconfiguring hardware that supports more efficient intrusion detection. Also, we have developed the prototype of our system. Most of all, our system focus on reducing a lowering of performance caused by high-speed traffic analysis to the minimum. Therefore, we present an improved content filtering techniques, called multi-hash based TBT. The hashing scheme helps to reduce the data access time, while the multi-hash based TBT technique helps to minimize the memory consumption.

Simulation-based performance evaluation showed that the proposed technique is memory efficient, thereby outperforming the previous one-hash based technique. A prototype of the proposed approach implemented on a FPGA, used the minimum amount of memory for storing 1661 signatures, whereas it can support a 2Gbps link regardless of the number and length of the signatures. Although the prototype can only perform single pattern comparison as of now, we are currently working toward the complete implementation of the TBT architecture to support multi-pattern signatures. Our future work includes optimizing the circuit timing to achieve higher throughput, and extending current TBT to harmonize with the high-speed signature generation approaches that employ packet content filtering.

References:


