Computer Communications 62 (2015) 47–58

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Algorithms to speedup pattern matching for network intrusion detection systems ☆

Kai Zheng [a], Zhiping Cai [b,*,1], Xin Zhang [c], Zhijun Wang [d], Baohua Yang [a]

[a] IBM China Research Lab, Beijing, PR China
[b] School of Computer, National University of Defense Technology, ChangSha 410073, PR China
[c] Carnegie Mellon University, Pittsburgh, PA, USA
[d] Dep. of Computing, Hong Kong Polytechnic University, Hong Kong, PR China

ARTICLE INFO

Article history:
Received 8 April 2013
Received in revised form 13 February 2015
Accepted 15 February 2015
Available online 21 February 2015

Keywords:
Negative pattern
Exclusive matching
Pattern matching
Intrusion detection

ABSTRACT

High-speed network intrusion detection systems (NIDSes) commonly employ TCAMs for fast pattern matching, and parallel TCAM-based pattern matching algorithms have proven promising to achieve even higher line rate. However, two challenges impede parallel TCAM-based pattern matching engines from being truly scalable, namely: (1) how to implement fine-grained parallelism to optimize load balancing and maximize throughput, and (2) how to reconcile between the performance gain and increased power consumption both due to parallelism. In this paper, we propose two techniques to answer the above challenges yielding an ultra-scalable NIDS. We first introduce the concept of negative pattern matching, by which we can splice flows into segments for fine-grained load balancing and optimized parallel speedup while ensuring correctness. negative pattern matching (NPM) also dramatically reduces the number of Ternary Content Addressable Memory (TCAM) lookups thus reducing the power consumption. Then we propose the idea of exclusive pattern matching, which divides the rule sets into subsets; each subset is queried selectively and independently given a certain input without affecting correctness. In concert, these two techniques improve both the pattern matching throughput and scalability in any scenario. Our experimental results show that up to 90% TCAM lookups can be saved, at the cost of merely 10% additional 2-byte index table lookups in the SRAM.

1. Introduction

As a real-field tested mechanism, the network intrusion detection system (NIDS) has remained a prominent resort in industrial security practices. Though not bullet proof to every exploit, NIDS dramatically increases the bar for adversaries and diminishes the attack surface. As a result, NIDS empirically prevents a vast array of malicious intrusions that could potentially incur losses of hundreds of millions of dollars every year [1].

Despite its importance and widespread deployment, throughput and scalability have plagued NIDS due to the expensive *Pattern Matching* (PM) operations involved. PM requires linearly inspecting every packet payload for potential matches against any rule (pattern) in a pattern set. Hence, PM is inherently resource-intensive regarding both computation and communication (i.e., I/O), when the traffic volume and the size of the pattern set are non-trivial. Though the *Ternary Content Addressable Memory* (TCAM) based PM algorithms tend to deliver superior throughput compared to pure software-based counterparts, the reported performance achieved to date is still a far cry from the rocketing network line speed. In a nutshell, such a performance limitation is attributed to the relatively low working frequency of TCAM chips and the high power consumption of TCAMs.

We observe that content pre-filtering and parallelism are two promising means to dramatically increase PM throughput for TCAM-based algorithms. First, content pre-filtering refers to the pre-processing of flows or pattern sets to reduce the overhead of the actual real-time PM operations. Second, exploiting parallelism has been a conventional tenet for achieving greater system efficiency/performance. Unfortunately, traditional pre-filtering schemes require sequential processing of the input flows and are difficult to be parallelized. In addition, parallelism must be deployed while retaining the scalability and correctness of the PM operations;

☆ The preliminary version of this work has been published in the proceeding of IEEE INFOCOM 2010.

* Corresponding author. Tel.: +86 731 84573674; fax: +86 731 84573675.
E-mail address: zpcai@nudt.edu.cn (Z. Cai).

[1] This author is supported by the NSFC (Nos. 61379145, 61202429, 61363071, and 61272510).

http://dx.doi.org/10.1016/j.comcom.2015.02.004
0140-3664/

this is highly challenging as parallel processing tends to increase power consumption and packets in a flow may mandate sequential scanning to detect malicious patterns that span across multiple packets.

To substantially speed up PM operations via parallelism while preserving scalability, we propose to partition both individual flows and the pattern set. First, an incoming flow is divided into consecutive segments, which can be inspected in parallel. Second, the entire pattern set is partitioned into multiple subsets, which can be looked up individually and selectively (as opposed to checking the entire pattern set every time), to provide extra dimensions for parallelism and to reduce the power consumption of hardware accesses. Implementing parallelism via such a two-level partitioning, however, creates two fundamental challenges for retaining PM correctness. First, partitioning the flows is non-trivial, since a sophisticated attacker can split a malicious pattern across multiple packets within a flow. Consequently, to capture such cross-packet dependencies (thus avoiding false negatives), flows are usually re-assembled and the packets in each flow are inspected sequentially [2]. Second, the partition of the pattern set should not cause false negatives. Moreover, it is challenging to minimize the power consumed by hardware accesses and to effectively balance the workload among the pattern subsets.

In light of the above observations, in this paper we develop an efficient parallel PM module based on TCAM coprocessors, which achieves high PM throughput and significantly reduces the number of TCAM accesses (thus minimizing the power consumption). We first introduce the novel concept of *Negative Patterns* (NPs) to address the challenge of partitioning flows while preserving correctness. Simply put, an NP is a string, any part of which (i.e., any substring of which) will not match any "normal" pattern in the given pattern set. Hence, by partitioning a flow only at locations within NPs appearing in that flow, it is ensured that no malicious patterns across multiple packets in a flow will be missed. We also show that with flow partitioning thanks to NPs and performing parallel inspection at flow-segment level, the number of TCAM accesses is considerably reduced and better load balancing is achieved compared to performing parallel inspection at the flow level.

Second, we propose the idea of *Exclusive Subsets* to efficiently partition the pattern set as follows. We identify all pairs of two patterns that will not be matched simultaneously within one TCAM lookup input, and separate the two exclusive patterns in such a pair into *exclusive subsets*. Intuitively, each exclusive subset can be looked up independently, since one TCAM lookup input can find matches in at most one of the exclusive subsets, thus saving power consumption. Finally we will show that the techniques of negative patterns and exclusive subsets are inherently complementary, enabling our integrated scheme to outperform previous approaches in any scenario (with either clean or dirty traffic).

1.1. Contributions

We believe this paper can advance the state-of-art NIDS in the following aspects.

- First, we achieve high throughput PM through parallel TCAMs with fine-grained load balancing at the granularity of flow segments.
- Second, both the NPs and exclusive subsets techniques reduce the number of TCAM accesses by 30–90% compared to the prior schemes.
- Third, we show that the two techniques in concert can yield desirable performance in any scenario (with either clean or dirty traffic).
- Finally, we perform extensive theoretical analysis and experimental evaluation to consolidate the effectiveness of the proposed techniques.

2. Related works

Pattern-matching schemes have been studied for a long time. The classic pattern matching schemes include those proposed by Aho and Corasick [3], Boyer and Moore [4], and their extensions/variations [5–11]. These schemes either require a large memory for software implementation or can hardly work with large pattern sets. Some pattern matching schemes [12,13] use heuristics to filter the strings that cannot be matched by any suspicious patterns. For example, these schemes use prefix/suffix of the patterns to do preprocess so as to improve the string matching speed. Regular expression matching and string matching [5–10,14–16] have also been widely studied to speed up the pattern matching performance. Very recently, a multi-core platform was developed for parallel pattern matching [17]. The Bloom filter is a kind of space efficient algorithm for pattern matching. It has also been proposed for intrusion detection in [18,19]. In [18], prefix Bloom filters are proposed to detect multi-packet intrusion signature patterns. In [20], a Bloom filter implemented with on-chip memory block is designed for fast pattern matching. However, due to the variable length of intrusion patterns, a large number of Bloom filters with different length may need to be constructed, and thus making them un-scalable.

To improve the matching speed, some hardware based solutions [19,21–27] using FPGA technology have been proposed for design high speed intrusion detection systems. Some of these approaches are claimed to be able to support 10Gbps wire speed by exploiting parallelism available in hardware implementations and using the state-of-the-art FPGA technology. But the intrusion patterns in the hardware need to be reconfigured when the patterns are updated, and hence these schemes cannot be used to support fast dynamic updates and also with huge implementation overheads in terms of both cost and time.

TCAM is a fast network search engine, and widely used for IP lookup [28,29], packet classification [31,32], and pattern matching [11,22,27,33,34]. In [22], a gigabit rate intrusion detection system using a single-TCAM coprocessor is proposed. The system can handle up to OC-48 (i.e. 2.5Gbps) line rates by using the currently fastest TCAM chips. Due to the need to inspect a data stream by shifting one byte at a time in the TCAM, the system requires high TCAM lookup throughput and also results in very huge power consumption. TCAM based regular expression matching approaches were developed in [27,29,30] to achieve high speed pattern matching. A covered state encoding scheme was designed for the Aho-Corrasick multi-pattern matching by implemented in TCAM [11]. Recently, we proposed a TCAM architecture for integrated policy filtering, content filtering and longest prefix matching [34].

Different from the tradition filtering schemes, the proposed algorithm uses flow/content partition which is proven to have the same effect. The "negative patterns" are leveraged to inspect the packet streams and target at finding "negative sub-strings" which should not be included in any patterns, and thus indicates the valid partition at the positions. Since the NP matching is false negative tolerant, parallelism can be much easier to explore than the previous approaches.

3. Design concepts

In this paper, we do not intend to take part in the debate on "*which category of PM schemes are the best*", which has been an open topic in the literatures for years. Each category has its own