Federal University of Pernambuco
Graduation in Computer Science

Informatics Center
2008.2

ON THE OPTIMIZATION OF DEEP PACKET INSPECTION

Undergraduate Thesis Proposal

Author: Thiago de Barros Lacerda (tbl2@cin.ufpe.br)
Advisor: Djamel Sadok (jamel@cin.ufpe.br)
Co-advisor: Stênio Fernandes (stenio@gprt.ufpe.br)

Recife, August 26, 2008

Table of Contents

1. Present Context

Internet Service Providers (ISP) and network administrators always had deep interest about knowing what type of traffic is going through their backbone. Due to that interest, lots of studies have been done in this area aiming at inspecting and classifying packets of a given flow successfully.

Online Traffic Classification is the most useful and desirable type of classification that is sought after among the research community. Additionally, because of the huge and rapid growth of the speed on the internet links, that task is becoming more and more difficult to accomplish, without packets loss and the desirable performance. Therefore, the scientific community has been working on how to combine accuracy in classification and performance, in a way to acquire the best of both (correctly classify flows and handle with high-speed links).

With that aiming, packet classifying, tools for such purpose are created and therefore rely on the operating system capture model. That capture method that is supported by Linux System (we will work only with Linux based systems) consists of, 1) copying the captured packet from the Network Interface Card (NIC) to the kernel memory space; 2) copying it again to the user memory space, for the applications, that the packet was destined for. So, to a Traffic Classification Application, the copies mentioned previously, represent an overhead that can lead to a loss of performance.

Previous studies [8] have shown that the traditional packet capture method using libpcap[1] suffers from several rates of packet loss. In fact, such loss rate is an issue that must be considered within the objective of classification precision of all flows. It additionally gives a limited vision of what is passing through the network link.

Among the methods used to make such classification, Deep Packet Inspection (DPI) is one of the most used, due to its precision and maintainability. DPI relies on a very expensive processing task, which is the packet's payloads inspection and attempting to match it with a large amount of protocol's signatures (most of all of those signatures consisting in regular expressions). Previous studies have verified that the matching process with regular expressions is responsible for about 90% of the CPU time in DPI [6], because of that, there have been lot of research in how this processing can be improved.

In order to successfully classify flows with high performance, many approaches have been applied in DPI. Those Approaches range from multicore architectures usage to algorithms to group Deterministic Finite Automatons (DFA), generated from the regular expressions [6], [12], [7] and [10].

2. Related Work

A great deal of work has been done among the scientific community in order to achieve higher performance and accuracy in traffic analysis and classification. Schneider [14] performs a deep analysis on how packet capture works on both BSD and Linux based systems. Additionally he points out some problems that exist in the present model of capture, such as packet copying.

A study of different approaches on packet capture was performed in [8]. In such work Deri evaluates the traditional approach using libpcap, among the Mmap libpcap [11], a kernel module implementation and also packet capture using device polling. He then proposes a ring buffer implemented at kernel level, that is shared by both user and kernel levels, which avoids copies to user space, achieving high gains of performance of packet capture on Linux based systems.

Schneider et al. on [13] studied how Linux based systems and FreeBSD behave by receiving a workload of 1 Gigabit/s. In fact they have observed a high rate of packets loss, with FreeBSD having a better performance than Linux. Additionally they point out the time spent on copies made from kernel space to user space and emphasize the use of device polling for better performance.

In the DPI context, a large amount of studies have been done to improve such task. In [3], Bernaille et al. propose to analyze the first 5 packets, more specifically only the packet's size, of

[1] http://www.tcpdump.org

a given TCP flow. They have used machine learning techniques to train their classifier. They also have achieved good results regarding the accuracy of his classifier.

Sen et al. in [15], made a study on peer-to-peer (P2P) application classification, using signature matching, performing analysis on five widely known P2P protocols. They also have shown that the packet examination is only needed on the first packets of a given flow (less than 10 packets), which led to less than 5% of false positive rate.

In [9], Neelam et al. propose a different way for traffic analysis. He developed an IPS (Intrusion Prevention System) prototype using Single Instruction Multiple Data (SIMD) with Graphical Processing Unit (GPU) architectures. Their IPS uses two types of state machines to perform the string matching operation, namely DFA (Deterministic Finite Automaton) and XFA (Extended Finite Automaton). He performed analysis using signatures from Snort [16] and Cisco Systems [5] and gained 6 to 9 times better performance, in comparison with a Pentium 4 system.

Additionally, as mentioned before, there are the works presented on [6], [7], [10] and [12], which are going to be explained in detail.

Fang Yu et al. in [6] have proposed regular expressions rewriting techniques, in order to reduce memory usage of such expressions (the set of used regular expressions were extracted from Bro [4], Snort and Linux L7-filter [2]). Additionally, grouping techniques of multiple patterns were proposed, which have lead to performance enhancements for the string matching operations.

In [7], Kumar et al. introduced a new representation for regular expressions, called Delayed Input DFA (D^2FA). This modification in the original DFA formalism substantially reduces space requirements as compared to a DFA. The D^2FA is based on a technique used in the Aho-Corasick string matching algorithm [1]. That approach can generally reduce the number of edges by more than 95% for the more complex DFAs that arise in network applications, dramatically reducing the space needed to represent the DFA. They also created some heuristics for the construction of an efficient D^2FA, to build an optimized D^2FA from a DFA is a NP-Hard problem.

Villa et al. [10] proposed a solution based on AC-opt, a variant of the Aho-Corasick algorithm that uses deterministic finite automata (DFA) to process separate chunks of input text. They also took into account the latest multi-core processors architectures to optimize their solution, namely IBM Cell Broadband Engine Architecture [17].

The authors in [12] present the importance of string matching for traffic identification and analysis, also citing some approaches on string matching algorithms development. They cite the Automaton-based (using DFAs and NFAs), Heuristic-based (that can make shifts while searching for a string in a payload, in order to avoid unnecessary comparisons) and the Filtering-based approaches (that makes a pre-filtering of the payload, to exclude patterns that definitely do not match). They discussed the pros and cons of those techniques.

As we can see on the related works, traffic classification still an unresolved problem, with a variety of approaches trying to minimize its problems. Therefore, our present work aims at proposing a different approach to that traffic analysis issue, presented above. In a way that tries to combine performance, accuracy in classification and memory space consumption reduction. This project will attack the time spent in regular expression searching, packet copies and consequently performance issues, but trying to lose the minimum in classification accuracy.

3. Project Description

The present project, proposes a DPI classifier, capable to handle with high speed broadband network links with good classification accuracy

3.1. Integrated Kernel Module

A kernel module (for the Linux kernel version 2.6) integrated is proposed with the following roles:

1) Copy only the necessary packets from the kernel memory space to user memory space

a. By necessary we mean, the packets of a flow that was not already classified
b. Copy only if the threshold of de maximum number of packets to be analyzed, of the given flow, was not reached (that threshold is going to be better explained at session 3.2)

2) Decrease the packet loss rate on capture

With the points described above, unnecessary copying of packets from the kernel memory space to user memory space (in the worst case we can prevent a copy of 1.5KB) can be avoided, which can lead to a significant gain of performance. Also, by decreasing the packet loss rate, the classification accuracy can be increased due to more packets being analyzed.

3.2. Packet Counting and Payload Truncating techniques

A combination of two techniques it is also proposed, which can lead to the same objective of increasing performance without accuracy loss. The first one consists of only analyzing the first packets of a given flow (getting based on previous studies [3] and [15]). Because if a given flow was not classified by analyzing the first packets, it has a high probability to remain unclassified. The second technique consists of only analyzing a fraction of the packet's payload and evaluating its impact in classification accuracy.

With both approaches described above we can have a gain of performance, on the time spent on copying the packet from the kernel memory space to the user memory space, which can help with the problem of handling network broadband links with high speeds.

Additionally, a considerable gain of performance can be achieved by truncating the packet's payload. Because of the packet's payload reduction, the operation of regular expression matching will have less work, due to less bytes to compare with the regular expression.

4. Work Plan

Getting based on the project objectives described on the previous session, the following work plan is proposed.

	Period															
Task	**September**				**October**				**November**				**December**			
Linux Kernel development study																
Study Linux Network internals																
Architecture Proposal																
Kernel module implementation																
Performance Evaluation and Tests																
Packet counting and payload sampling integration																
Evaluation of all techniques together																
Comparison with raw DPI application																
Document writing																

References

[1] Aho, A. V. and Corasick, M. J. 1975. "Efficient string matching: an aid to bibliographic search," *Commun. ACM* 18, 6 (Jun. 1975), 333-340

[2] Application Layer Packet Classifier for Linux. http://l7-filter.sourceforge.net/, visited on August 25 2008

[3] Bernaille, L., Teixeira, R., Akodkenou, I., Soule, A., and Salamatian, K. 2006. "Traffic classification on the fly," *SIGCOMM Comput. Commun. Rev.* 36, 2 (Apr. 2006), 23-26

[4] Bro Intrusion Detection System. http://www.bro-ids.org/, visited on August 25, 2008

[5] Cisco Intrusion Prevention System. http://www.cisco.com/en/US/products/sw/secursw/ps2113/index.html, visited on August 25, 2008
[6] Fang Yu, Zhifeng Chen, Yanlei Diao, T. V. Lakshman, Randy H. Katz, "Fast and memory-efficient regular expression matching for deep packet inspection," ancs, pp. 93-102, *Symposium On Architecture For Networking And Communications Systems*, 2006
[7] Kumar, S., Dharmapurikar, S., Yu, F., Crowley, P., and Turner, J. 2006. "Algorithms to accelerate multiple regular expressions matching for deep packet inspection". In Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols For Computer Communications (Pisa, Italy, September 11 - 15, 2006). SIGCOMM '06. ACM, New York, NY, 339-350
[8] L. Deri, "Improving Passive Packet Capture: Beyond Device Polling," Proceedings of SANE 2004, 2004.
[9] Neelam Goyal, Justin Ormont, Randy Smith, Karthikeyan Sankaralingam, and Cristian Estan, "Signature Matching in Network Processing using SIMD/GPU Architectures," Technical Report TR1628, Department of Computer Sciences, The University of Wisconsin-Madison, 2008
[10] Oreste Villa, Daniele Paolo Scarpazza, Fabrizio Petrini, "Accelerating Real-Time String Searching with Multicore Processors," Computer, vol. 41, no. 4, pp. 42-50, Apr., 2008.
[11] P. Wood, *libpcap-mmap*, Los Alamos National Labs, http://public.lanl.gov/cpw/.
[12] Po-Ching Lin, Ying-Dar Lin, Yuan-Cheng Lai, Tsern-Huei Lee, "Using String Matching for Deep Packet Inspection," *Computer*, vol. 41, no. 4, pp. 23-28, Apr., 2008
[13] Schneider, F., Wallerich, J., Feldmann, A., "Packet Capture in 10-Gigabit Ethernet Environments Using Contemporary Commodity Hardware," PAM 2007
[14] Schneider, Fabian, "Performance evaluation of packet capturing systems for high-speed networks," Master Thesis, 2005, Technical University of Munich
[15] Sen, S., Spatscheck, O., and Wang, D. 2004. "Accurate, scalable in-network identification of p2p traffic using application signatures," In *Proceedings of the 13th international Conference on World Wide Web* (New York, NY, USA, May 17 - 20, 2004). WWW '04. ACM, New York, NY, 512-521
[16] Snort - the de facto standard for intrusion detection/prevention. http://www.snort.org/, visited on August 25, 2008
[17] The Cell project at IBM Research. http://researchweb.watson.ibm.com/cell/, visited on August 25, 2008

Signatures

_____

Djamel Sadok
(Advisor)

_____

Thiago Lacerda
(Student)

Recife, August 26, 2008