Internet Traffic and the Behavior of Processing Workloads

Ruken Zilan*[1,3] Javier Verdu*‡[1,3]
Jorge Garcia Vidal‡[2,3]
Mario Nemirovsky*[1,3,†] Mateo Valero*‡[1,3]

* *BSC, Nexus II 29, 08034 Barcelona, Spain*
‡ *UPC, Campus Nord, D6 08034 Barcelona, Spain*
† *ICREA Research Professor*

ABSTRACT

Nowadays, the evolution of network services provided at the edge of Internet increases the requirements of network applications. Such applications result in complexities thus, the processors need to execute more complex workloads that can deal not only with the packet header, but also with the packet payload (e.g. Deep Packet Inspection). Unlike common routing applications that show similar processing among packets, next-generation of network applications present variations in the processing procedure among packets. Thus, different traffic behaviors can produce different process patterns and present different memory and processing requirements.

The aim of this work is to present an ongoing work towards correlating Internet traffic features with variations of processing workloads on the next-generation of edge routers.

KEYWORDS: Internet traffic behavior, processing workload, edge routers

1 Introduction

The development of network services is strongly related to the network technology trend. Today's and future's expected technological developments bring large amount of memory and complex execution requirements. Considering the complex applications with the high amount of traffic and bandwidth limitations, more capable routers are necessary. In traditional networks, network nodes forward packets from a source to a destination by executing lightweight packet processing, or even negligible workloads. On the other hand, with the new generation applications, since the links provide more

---

[1]E-mail: {name.surname}@bsc.es
[2]E-mail:jorge@ac.upc.edu
[3]This work has been supported by the Ministry of Science and Technology of Spain under contract TIN-2007-60625, the HiPEAC European Network of Excellence and a Collaboration Agreement between Cisco Systems and BSC. The authors wish to thank Rodolfo Milito from Cisco for his support.

complex services, packet processing demands execution of more computational intensive applications. Complex network applications deal with both packet header and payload (i.e. packet contents) to provide upper layer network services, such as enhanced security, system utilization policies, video on demand management, and ubiquitous computing on the network.

Mentioning the importance of the evolution of services at the edge routers, let's look at the relation between the traffic and the processing workload under different conditions. In a simple scenario, such as IP Forwarding, most of the packets execute the same instructions and access to the same memory references. However, at the most complex scenario, for instance advanced Intrusion Detection Systems [VGNV05], packets execute different amount of instructions and access to different data structures. Thus, the processing requirements are quite sensitive to the behavior of traffic as we move to complex scenarios. Nonetheless, they are also sensitive to a wider variety of traffic features, while simple scenarios are only sensitive to few traffic features, such as IP address distribution and packet size.

The goal of this work is to introduce an ongoing work towards correlating Internet traffic characteristics with the variations of processing workloads on the edge routers. This work is focused on the variations of processing and memory requirements up to the Transport network processing, although future work addresses upper network layers.

2 Background

Network applications mainly present three stages; receiving the packet (R); processing the packet (P); and transmitting the packet (T). All applications show very similar R and T stages, but the P stage can be quite different depending on the application itself. In this study, we focus on applications that the P stage presents stateful processing or Deep Packet Inspection (DPI). In fact, many of the new network applications merge stateful processing and DPI.

Statefulness is the capability of keeping information about previous packet processing. Advanced network applications keep state information of processed packets. The state can be related to packets of a given flow or packets across different flows. These applications called stateful, increase dependencies among packets and reduce the amount of available parallelism. Beside, current execution models of network applications don't take advantage of the massive multithreaded capabilities.

Deep Packet Inspection (DPI) is the operation that examines the packet contents of both header and payload of all OSI Model layers (e.g. pattern matching, high layer header decoding). This is similar to a mailman opening the envelope and actually reading the content of the letter. Pattern matching has some heavy processing requirements, but it does not necessarily require keeping state. Lower layer applications deal with well known located fields in the packet header. For instance, a router forwards packet by searching an entry in the forwarding table according to the IP destination address. On the contrary, DPI demands much more intensive processing to deal with payload fields. DPI presents complex field recognition process, due to non-well known formats of payload. That is, it brings a large variety of protocols that lead to different packet payload formats.

Common routing applications that deal with Layer 3 of the OSI Model (IP processing) decode the IP packet header and access a global table to perform some actions (IP Lookup). The distribution of IP addresses can determine the behavior of accesses to such global data structure and thus, the data cache performance. Instead, network applications of the next-generation edge routers, mostly present DPI and/or stateful processing that makes the application sensitive to other network features.

3 Selection of Processing Workload Features

Next-generation of network applications especially deal with high network layers of the OSI Model (i.e. from Transport to Application layer). We set this work focusing on applications that process up to Transport layer (e.g. TCP, UDP).

In [VGNV05] the authors study the **number of instructions** required to process each packet of a given generic flow. However, throughout the whole packet processing there are several processing stages that present different behavior and necessities. Simple applications present well defined processing stages called software pipeline stages, which mainly improve cache locality (either data and/or instruction) in every stage as well as similar processing requirements. Complex applications, though, are more difficult to identify pipeline stages to improve cache and to clusterize processing requirements. Thus, alternative approaches must be followed to detect proper processing stages. One of the aims of this study is to get a deeper understanding of processing stage identification in complex network applications by analyzing the processing requirements throughout the whole packet processing.

On the other hand, to improve the memory requirements, we analyze the instruction and data memory accesses. The **instructions footprint** will let us study the variations of instructions executed throughout the processing of network traffic. Moreover, it will let us analyze how to identify sections of the processing workload that may present similarities among packets processing. We also study the **data footprint** and the **data working set** to address the effects of memory demand. The main goal is to identify which processing stages increase memory requirements under different traffic scenarios and which parts of the workload show similar memory demand regardless of the network traffic.

4 Selection of Traffic Features

This section describes some of the main network traffic parameters that we select to analyze their impacts on the processing workload. In this work we study the effects up to the Transport layer, but future works will include parameters of upper network layers.

The amount of active network connections within a period of time means the **traffic aggregation**. The more aggregation the traffic shows, the more interleaved active flows can be found within a chunk of packets and then, the more memory requirements present the processing workload, especially if it does stateful processing. However, this memory demand depends on the goal of data structures for stateful purposes of the application itself. Besides, the traffic aggregation impact can be isolated within a given few processing stages of the application or can be distributed throughout the whole packet processing.

Bursty traffic is the traffic in which the short-term bandwidth exceeds the average bandwidth. Bursty traffic may cause excessive queuing delay and packet loss, if it is not supportable by the network. However, it has different effects on the processing workload. Depending on the type of **traffic burstiness**, the processing requirements can be quite different than the average. Thus, when there is bursty traffic within a flow, the distance between two packets of the same flow can be significantly reduced for a period of time. That is, there are peaks where temporal cache locality can be increased for certain sets of data memory accesses. In fact, the **flow temporal distribution** describes the locality among packets of stateful data structures [VGNV05]. Our work focuses on how traffic burstiness and flow temporal distribution can affect data structures, especially those that are sensitive to the flows.

As applications that present DPI, target pattern matching, parameters such as **packet size** (for generic DPI) and **flow size** (for stateful DPI) directly affect the processing requirements. For example, mostly larger packet sizes need to execute more instructions to do a given pattern matching. Nevertheless, it is necessary to get a deeper understanding of what types of DPI are more sensitive to the behavior of these traffic parameters.

5 Related Work

There are other authors that address similar workload analyses. Ramaswamy et al. [RWW05] analyze the processing characteristics of individual packets and the variation between them in order to understand the network processing workloads. They present low network layer applications and thus, they cannot include any result about flow related impact, due to traffic features. Verdú et al. [VGNV05] analyze architectural bottlenecks running stateful network applications. The study details the variations of system performance (e.g. cache hit rate, branch prediction hit rate) and performance requirements (e.g. number of instructions per packet) throughout the processing of every packet within a given TCP flow. However, this work is done from the point of view of processor performance and requirements, instead of the processing workload.

There are many studies which show that Internet traffic continues to grow and provided services become more complex, over relatively short time scales. This is not only simply due to the traffic volume, but also due to heterogeneity which is caused by the mixed-traffic, protocols, applications, and users [ZBPS02]. One of the first papers on traffic and semantic characteristics of flows is [HVGV05]. Holanda et al. show that, using clustering techniques, there is no much variety in high speed link flows so that flows can be grouped in few clusters. Therefore, they implemented a template of flows and demonstrated that the methodology can be used to develop header trace comparison and trace classifications. The understanding of traffic behaviour, leads us to model the variations of processing workloads which is sensitive to the traffic.

6 Ongoing Work

Our next step spotlights analyzing the impact of upper network layers' behavior on multithreaded processing workloads. Once we have a deep understanding of the relation between workload behavior and traffic characteristics, we will extend our analysis to multithreaded processors, such as thread contention in critical sections and bottlenecks on shared resources.

References

[HVGV05] R. Holanda, J. Verdu, J. Garcia, and M. Valero. Performance analysis of a new packet trace compressor based on TCP flow clustering. In *Proc. of ISPASS'05*, pages 219–225, Austin, TX, March 2005.

[RWW05] Ramaswamy Ramaswamy, Ning Weng, and Tilman Wolf. Analysis of network processing workloads. In *Proc. of ISPASS'05*, pages 226–235, Austin, TX, March 2005.

[VGNV05] Javier Verdú, Jorge García, Mario Nemirovsky, and Mateo Valero. Architectural impact of stateful networking applications. In *Proc. of ANCS'05*, NY, USA, 2005. ACM.

[ZBPS02] Y. Zhang, L. Breslau, V. Paxson, and S. Shenker. The characteristics and origins of internet flow rates, 2002.