UNICAMP - State University of Campinas
School of Electrical and Computer Engineering (FEEC)
Department of Computer Engineering and Industrial Automation

Date: Jul 26, 2015 **Working Plan:** #4
WP Period: Jul 26, 2015 - Aug 22, 2014
Student: Anderson dos Santos Paschoalon
Advisor: Christian Esteve Rothenberg

July 28, 2015

Report

1 What I have did

As cited in the NetFPGA *github wiki* [1], it is an low cost platform, that provide a useful tool for networks researchers, that permits the deployment of new features in hardware, at high speed rates comparable with the art state of networking hardware. This versions contains 4 Ethernet interfaces 1 Gigabit/second, a user programmable FPGA, and four banks of SRAM and DRAM memories.

My previous activities consisted in the study of the NetFPGA 1G platform and successful reproduction of two previous works: [2] and [3]. Although, during the reproduction of some other experiments, there was some issues, still unsolved.

The experiments successfully executed were:

- [2]: This work aims to give an introduction the the NetFPGA platform, and presents the process of installation of its infrastructure. Also provides an implementation of a simple hardware-based firewall, with some regression tests to verify its functionality, presenting then a detailed explanation of the implementation process, showing its Verilog code.
- [3]: This paper presents a NetFPGA based multiprocessor architecture called NetThreads. This work aim to provide an platform were developers can deploy over the NetFPGA using multi-threaded processor. A C compiler were provided too. The paper presents details of the architecture, and some results involving throughput tests of some deployed applications.

In the second case, the original applications weren't provided by the authors. Instead, another one, more simple, were, so it was possible to reproduce the same experiments, which achieved better results, in terms of performance

Although, in these both cited experiments cases the regression tests and experiments were succeed, in some others tries, there were some (still) unsolved issues. Two different embedded packet generators were tried:

- The PacketGenerator [4], first packet generator deployed for the NetFPGA platform. This one uses pre-generated *pcap* files to create a traffic trough the NetFPGA interfaces.

- The Caliper [5], [6]. This packet generator is expected to be able to create traffic without *pcap* files. The host driver transmits enough information to Caliper build the packets, and than, dynamically generates them. Caliper is developed in C, and runs over the NetThreads multiprocessor.

In both cases, there were problems during the compilation time. Also, the performed experiment uses two NetFPGA 1G boards, so the reproduction of the same experiment is unpractical.

In the first case, the suspected issue is a different version of the NetFPGA base used was the 3.0.1, but the referenced at wiki was the 2.1.1. In the case of the Caliper, a problem occurs during a compilation of an modified version of the NetFPGA driver. In the original experiment, the Linux kernel version used was 2.6.28. The version used in the reproduction was 2.6.33.8. A next step should be perform these modifications, and tried to run these experiments with both traffic generators.

So, the news activities related with the previous ones, must be focused in the correct reproduction of those experiments and in a deep understanding of both traffic generators, its differences and capabilities and drawbacks. In this way, will be possible to get the art state of traffic generators of the NetFGPA 1G

2 What is there to do/research

There are still some issues to solve from the last activities, and with them new view of possibilities. So, here are going to be things that still must be done, and with them new things to do. These things will be listed below.

- **Study the Network tester OSNT** [7], making a summary of its capability and drawbacks. This tool, implemented to run over the NetFPGA 10G provide features as Traffic Generation and Traffic Monitoring. Also, study its related work, and see if it is possible to run it over the NetFPGA 1G. Evaluate the art state of NetFPGA 10G

- Try to **solve the issues with the *PacketGenerator*** [4], and execute its experiments. This generator uses pcap files, so it should be an important point of reference.

- **Solve the problems with the packet generator Caliper, and evaluate its capabilities and drawbacks**. This packet generator do not use *pcap* files, and create the traffic dynamically. Then, evaluate its process, and see the capability of this generator in creating different network traffic shapes. In the Caliper's article, the proposed experiment uses two NetFPGA platforms.

It should be checked if there is any issue of executing a load test using just one NetFPGA platform, as presented in the Figure 1.

- A **knowledge of how the communication between the host and the NetFPGA is needed** to make possible any deployment. In principle, it is not required the have deep understanding, if a interface is provided, this should be enough. So will be necessary to study the driver code.

- One important gap of study is how to **emulate network traffic without use of *pcap* files**. For low rates of data transfer the use of *pcap* files is suitable, but at high rates, it inevitable consumes larges amounts of space just to record some seconds of data. So, to avoid this there some possibilities of creating the packets dynamically, as:

  - Store just the headers;
  - Use an dictionary of packets;
  - Store just the information to create the packets (as packet type, addresses, payload size).

  Also, the dynamical creation of the packets, without *pcap* files may be either **static** (with a description file of the traffic) or **statistical** (the traffic characteristics are created in time of execution, according to some specified parameters). Therefore there is many possibilities of implement a dynamical traffic creator. So it is need to make a deeper study in this context, understanding what the NetFPGA tools are able to provide, what tools for ordinary hosts have, and how they do it it.

- A knowledge of how **manipulate *pcap* files** is need too, in order to enable the capture and analysis of traffic data, or even create packets dynamically.

- Another possibility of work, still related with traffic generation the creation of **traffic distribution models**. It is expected that different kind of inter-communication between hosts should have different behaviors in therms of bandwidth, QoS and types of packets expected. For example, suppose a video streaming, a VoIP communication, and a host making *http* requests. Each traffic for case has very particular traffic characteristics, in terms of required data rates, types of packets transferred and payload sizes. One possible work, is analyze these different types of traffics and a combination of them in real situations, record necessary information, and than be able to reproduce it (or at least create a traffic with similar characteristics) in a controlled test-bed using traffic models (for example, with a packet generator in the NetFPGA platform). So, an work that must be done, is get a picture of the art state in this type of works, and what still must be done. It expected found be found shape's models as presented in Figure 2 (Gaussian) and Figure 3 (Chi-square) for packet-size and packet-types distribution, and distribution models like in the Figure 4 for transference rates (throughput).

- With this sort of tools mentioned before, an possibility of application is use them to **implement a test-bed for NFV environments**. Going back to the Figure 1, the server may be running an NFV environment. Test it with different traffic scenarios and capture important data information about the environment as performance, throughput and resilience is necessary.

Figure 1: NetFPGA Diagram of a scenario of time consuming in a load. The first loop (*nf2c0–nf2c1*) captures the amount of time expended to forward a packet and receive it, as well the wire time-consuming. The second loop (textitnf2c2–nf2c3) captures this time plus the amount of time expend in the load (in this example, a server). So the second time minus the first represents the time expend in the load environment.

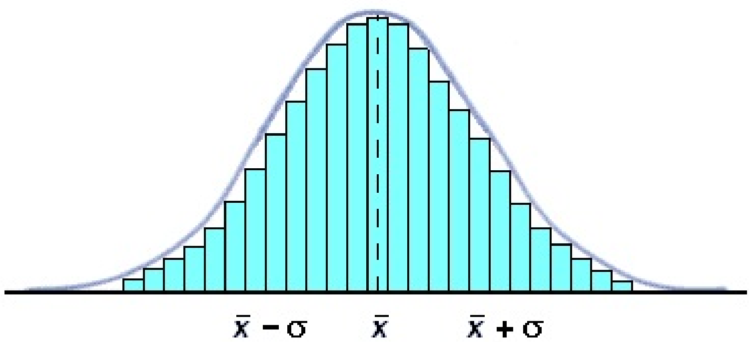

Figure 2: Example of a discrete Gaussian Slope

3 Additional references and links

A collection of external resources as papers and internet links were made. The apparently more relevant studies were disposed in the schedule.

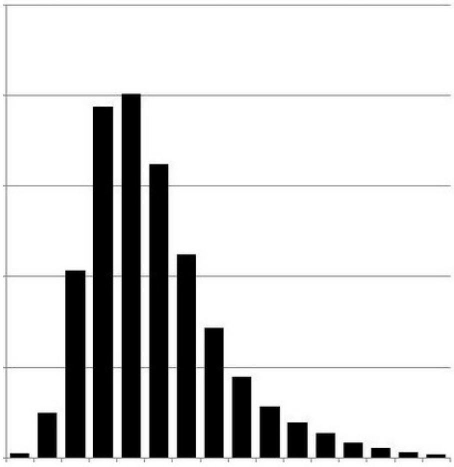

Figure 3: Example of a discrete Chi Square Slope

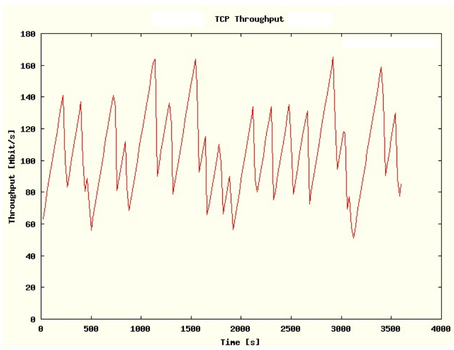

Figure 4: Example of an TCP throughput expected slope. At the begin (not presented) it should have an exponential grow, until the fist packet loss happen. Than, the transfer rate decreases to the threshold value, than start to increase linearly.

3.1 Papers:

- A Method for Summarizing and Visualizing City Traffic Data;
- Bivariate Gamma Distribution - A plausible solution for joint distribution of packet arrival and their sizes;
- Detecting DoS Attacks Using Packet Size Distribution;
- Detection of Applications Within Encrypted Tunnels Using Packet Size Distributions;
- iCloud Traffic Control and Monitoring;
- Internet Packet Size Distributions - Some Observations;
- Internet Traffic Characterization Using Packet-Pair Probing;
- On the Multi-Scale Behavior of Packet Size Distribution in Internet Backbone Network;

- Packet Delay Distribution of the IEEE 802.11 Distributed Coordination Function;
- Packet Size Distribution Tendencies in Computer Network Flows;
- Real-time traffic shape monitoring system for video-audio streaming systems;
- Session Level Flow Classification by Packet Size Distribution and Session Grouping;
- Slot Assignment Scheme for Integrated Voice and Data Traffic in Reservation-type Packet Radio Networks;
- The Packet Size Distribution Patterns Of The Typical Internet Applications;
- User Data Traffic Analysis for 3G Cellular Networks;
- Using Packet Size Distribution To Identify P2P-TV Traffic;
- Using packet size distributions to identify real-time networked applications;
- Wireless Media Access Depending on Packet Size Distribution over Error-prone Channels;
- A Packet Distribution Traffic Model for Computer Networks.

3.2 Additional resources:

- C code to read *pcap* files: `https://code.google.com/p/pcapsctpspliter/issues/detail?id=6`
- Traffic shaping: `https://en.wikipedia.org/wiki/Traffic_shaping`;
- Traffic analysis: `https://en.wikipedia.org/wiki/Traffic_analysis`, `http://iris-network-traffic-analyzer.softonic.com/`, `http://www.networkmanagementsoftware.com/5-free-netflow-analyzer-tools-for-windows`;
- Packet generator: `https://en.wikipedia.org/wiki/Packet_generator`;
- Traffic analysis: `http://www.cse.wustl.edu/~jain/cse567-06/ftp/net_monitoring.pdf`;
- A Survey of Network Traffic Monitoring and Analysis Tools: `http://www.cs.wustl.edu/~jain/cse567-06/ftp/net_traffic_monitors3/index.html`;
- Network traffic monitoring: `http://www.topology.org/comms/netmon.html`;

- Packet size distribution comparison between Internet links in 1998 and 2008: `https://www.caida.org/research/traffic-analysis/pkt_size_distribution/graphs.xml`;
- History of network traffic models: `https://en.wikipedia.org/wiki/History_of_network_traffic_models`;
- Survey of Network Traffic Models: `http://www.cs.wustl.edu/~jain/cse567-06/ftp/traffic_models3/`;

4 Conclusion

As conclusion, is possible to abstract the main goals of the next research cycle in this main tasks:

- Solve the issues with the packet generators, and get a picture of its art state in the NetFPGA 1G;
- Get a picture of the art state of packet generators of NetFPGA 10G;
- Study the models for traffic distribution;
- Understand the tools for deployment over the NetFPGA to make it possible work with traffic distribution in this platform.

References

[1] "Netfpga - guide." `https://github.com/NetFPGA/netfpga/wiki/Guide`, 2013. [Online; accessed 26-July-2015].

[2] P. Goulart, I. Cunha, M. A. M. Vieira, C. Marconder, and R. Menotti, "Netfpga: Processamento de pacotes em hardware," *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC*, 2015.

[3] M. Labrecque, J. G. Steffan, G. Salmon, M. Ghobadi, and Y. Ganjali, "Netthreads: Programming netfpga with threaded software," *NetFPGA Developers Workshop*, August 2009.

[4] G. Covington, G. Gibb, J. Lockwood, and N. McKeown, "A packet generator on the netfpga platform," in *Field Programmable Custom Computing Machines, 2009. FCCM '09. 17th IEEE Symposium on*, pp. 235–238, April 2009.

[5] G. Salmon, M. Ghobadi, Y. Ganjali, M. Labrecque, and J. G. Steffan, "Netfpga-based precise traffic generation," *NetFPGA Developers Workshop*, August 2009.

[6] M. Ghobadi, G. Salmon, Y. Ganjali, M. Labrecque, and J. Steffan, "Caliper: Precise and responsive traffic generator," in *High-Performance Interconnects (HOTI), 2012 IEEE 20th Annual Symposium on*, pp. 25–32, Aug 2012.

[7] G. Antichi, M. Shahbaz, Y. Geng, N. Zilberman, A. Covington, M. Bruyere, N. McKeown, N. Feamster, B. Felderman, M. Blott, A. Moore, and P. Owezarski, "Osnt: open source network tester," *Network, IEEE*, vol. 28, pp. 6–12, September 2014.

Week Work Plan

	Tasks	Status	Week 07/26	08/02	08/09	08/16	08/23	08/30
Papers to read								
1	OSNT - Open Source Network Tester	To do						
2	Caliper - Precise and Responsive Traffic Generator	To do						
3	A Packet Distribution Traffic Model for Computer	To do						
4	A Packet Generator on the NetFPGA Platform	To do						
5	The Packet Size Distribution Patterns Of The Typical Internet Applications	To do						
6	Performing Time-Sensitive Network Experiments	To do						
7	Bivariate Gamma Distribution - A plausible solution for joint distribution of packet arrival and their sizes	To do						
8	Using Packet Size Distribution To Identify P2P-TV Traffic	To do						
9	Using packet size distributions to identify real-time networked applications	To do						
10	Packet Size Distribution Tendencies in Computer Network Flows	To do						
11	User Data Traffic Analysis for 3G Cellular Networks	To do						
12	On the Multi-Scale Behavior of Packet Size Distribution in Internet Backbone Network	To do						
Practical Activities								
13	Test the viability of OSNT over NetFPGA 1G	To do						
14	Solve the issues with Caliper packet generator	To do						
15	Solve the issues with the PacketGenerator	To do						
16	Study the NetFPGA driver code to undestand how the communication between the host and the NetFPGA is made	To do						
17	Learn how manipulate pcap files with C or other language	To do						
18	Make a report of trafic-distribution models	To do						

Anderson dos Santos Paschoalon
July 28, 2015