Summary Review Documentation for

"Wire-speed Statistical Classification of Network Traffic on Commodity Hardware"

Authors: P. Ró, D. Rossi, F. Gringoli, L. Nava, L. Salgarelli, J. Aracil

Reviewer #1

Summary: This paper proposes a software solution for the classification of traffic on commodity hardware. The authors are able to demonstrate tremendous improvements compared to a number of current systems. They further evaluate the bottlenecks in their solution and study different configurations.

Strengths:

- interesting software system that can perform statistical classification at 20 Gbps speeds on commodity hardware
- very good knowledge of the state of the art and comparison
- Interesting performance evaluation that studies different configurations
- public release of tool

Weaknesses: Very very narrowly focused - not sure how accessible it would be to the general IMC audience (but I would not reject for this reason - the work is of good quality). English could use some work.

Comments to authors: This is a very nice engineering paper with the goal to use commodity hardware and advances in software support for packet capture and classification at line speeds. The authors do demonstrate extensive knowledge of the literature and techniques that can be used to remove any associated bottleneck. As said by the authors, they are not only focusing on how fast they can process packets, but on how much they can actually classify if they are to use the first 5 packets.

The paper is nicely written clearly justifying the design choices made and comparing with all relevant work. Really nice!

I am not entirely sure how accessible this paper would be by the average IMC attendee, but it certainly advances the state of the art in traffic classification at line speed, which is a fundamental topic for our community.

Reviewer #2

Summary: This paper reports on a configuration of software and off the shelf hardware capable of doing flow classification at 10Gbps.

Strengths: The paper is economical and to the point. It demonstrates a useful capability that will be of interest to researchers seeking to do flow classification and packet processing generally. The tools are freely distributed. The paper describes various designs and the strengths/drawbacks.

Weaknesses: The application is a bit narrow.

Comments to authors: This paper does a nice job of describing a useful capability: high speed flow classification on commodity hardware. I don't find the application particularly exciting, but the general insights into system organization are valuable.

The paper basically says that by restructuring the NIC driver, keeping data copying to a minimum, pinning threads to processors, and using nonlocking data structures one can run a simple flow classifier at 10Gbps line rate on a commodity PC. This seems to be a significant increase over the state of the art based on the discussion in the paper.

The paper evaluates different software configurations and shows the benefit of the various improvements that they have made. Overall this is a solid paper.

Reviewer #3

Summary: This paper implements a software-based traffic classification engine on commodity hardware using *previously published* network driver and statistical classification techniques. The only contribution is in the 'engineering' aspect, i.e., proper tuning of the software/hardware, and stress-testing the system.

I find this paper out of scope of IMC. There is no novel research ideas related to "network measurements".

Strengths: Actual implementation and stress-testing of existing traffic classification engine mapped onto commodity hardware with improved network driver.

Weaknesses: Low in novelty. It implements existing statistical traffic classification techniques [19] using previously proposed/published network driver (packetShader [11]) and publicly available traces.

Comments to authors: The paper basically puts together existing components: improved network driver, Naive Bayes classification technique, and then evaluates different architectural configurations (different number of interfaces, queues, processes) in the experiments, using synthetic and real traffic. I find it a bit out of scope for IMC, and may be more suitable for venues such as ANCS.

The paper never formally defines the performance metric, "classification rate" used in the evaluation. Is the number of packets *correctly* classified, or just the total number of packets processed by the classifier engine?

Both accuracy and speed are important for traffic classification. The paper only implements one statistical classification technique and shows how much 'speed' can be achieved, without reporting any accuracy number, which makes it less meaningful.

Reviewer #4

Summary: This paper reports the performance of a software-based traffic classification engine that can process 14.2Mpps, the maximum possible packet rate over a 10Gbps link on commodity hardware. The system is evaluated by both synthetic traffic, and replayed real traffic traces collected from a 10Gbps link. The reported performance is much better than the state-of-the-art, and is enabled by (1) the use of a customized network driver (2) the use of a light weight statistical classification technique and (3) a careful parameter tuning of the system.

Strengths: This is a good systems engineering work that identifies bottlenecks in a modern off-the-shelf multi-core PC to be used for packet classification. The evaluation methodologies and the reported performance numbers of the components are useful for those who build measurement systems on commodity PCs.

The proposed system can handle 10Gbps full wire-speed on commodity hardware, which marks a key milestone in the current networking technologies.

The authors made the source code of the traffic generator publicly available (but apparently the source code of the classification system is not available).

Weaknesses: The main contribution of the paper is on system tuning, combining the existing techniques with recent commodity hardware, and the paper does not have a specific key insight to enhance the performance of packet classification systems. As a result, the paper does not provide much academic values.

Comments to authors: This paper sheds light on the performance of packet classification on commodity hardware, and is an interesting read. The paper shows that, with modern multi-core PCs, it becomes important how to split tasks into available cores and how to avoid memory access contentions among the cores. I can see a significant amount of efforts made by the authors to push the system performance to reach 10Gbps.

You should briefly describe the classification method used in the paper in Sec 3. The current description says only that a naive bayesian algorithm on the size of the first 4 packet is used, which is not enough for the readers to understand the mechanism.

Sec 5.1: "forged with incremental IP addresses and TCP ports" Does it reduce collisions of hashes for the flow table?

The descriptions in Sec 5.2 are a bit redundant, as these results are expected from the system configurations explained in Sec 4.

Sec 5.2: Please describe how you obtained the CPU usage from the system, as well as the precision of the CPU usage measurements (e.g., confidence intervals).

Reviewer #5

Summary: The authors designed a system using a software based traffic classifier that runs on off-the-shelf Intel multi-commodity hardware. They demonstrate that their system is able classify packets faster than any existing system by dramatic amounts. They can sustain processing at maximum input line rate (10Gbps which means 14.2MPkt/sec) and simultaneously achieve traffic classification rates that beat the state of the art: 93x faster than [15], and 560x times faster than [26].

Strengths: They incorporated lots of state-of-the-art techniques into a simple architecture and demonstrated that it is feasible to do traffic classification roughly 2 orders of magnitude faster than state-of-the-art methods using off-the-shelf hardware. The performance is very impressive.

Weaknesses: They claim previous work [26] cannot close the gap between the effort in [26] and that shown here for 2 reasons. Neither of these 2 reasons is explained well enough to be fully convincing, and the amazing performance gains shown here hinge upon this comparison.

Comments to authors: They incorporate a number of ideas from recent research efforts into a simple architecture to build a complete system. They use PacketShader [11] that captures packets quickly, and a NaiveBayes classifier [19] that was shown to do traffic classification accurately using only the first 4 packets, zero copy technology (for exposing packets to user space), hashing to do quick flow lookup and handle collisions by chaining, etc.

In the description of the sniffing module, it isn't clear if any specific idea belongs to the authors, or if they just cleverly assembled the most recent developments for mapping packets to queues. For example, the zero copy trick is already known, so you just incorporated it, right?

Some of the findings seem obvious. Clearly having only 1 RSS queue is going to cause a bottleneck. No? Am I missing something subtle? Why would one ever even consider using 1 RSS queue on a system with multiple RSS queues and multicores?

You claim that the performance gap with [26] cannot be overcome for 2 reasons. First, you say there is a bottleneck in the path from the NIC to the GPU. Could you say more? How can we be sure this is "intrinsic" to the GPU-based approach (not just their hardware tried)? Second, although the authors in [26] used DPI, why couldn't they replace that block by the NaiveBayes classifier? Using a NaiveBayes classifier over a GPU might remove one order of magnitude of the difference between the two techniques, no?

Response from the Authors

We would like to thank the anonymous reviewers for their valuable comments which allowed us to improve our work in its camera-ready version. We believe that the camera-ready version manages to address almost all comments and suggestions in the final version of the paper.

Additionally, as TPC suggested, we released the code of our implementation to the community as open-software, which may be useful for researchers and designers in order to implement their own network monitoring tools (or modify the existing ones) to cope with current high speeds (10Gbps and beyond).

In more details, some reviewers point out classification accuracy to be a key issue. While we agree accuracy (i.e., packets correctly classified) to be a key issue, it has been already thoroughly analyzed, and is thus outside the scope of this paper. Nevertheless, we have chosen a representative classification engine (early classification) and machine learning technique (Nave-Bayes, that is now better described according to reviewers suggestion), whose combination has previously been shown to yield to interesting results as for the traffic accuracy is concerned. Hence, in this paper we focus on the feasibility and performance of its implementation at 10Gbps. Anyway, extension of this work to other machine learning techniques (such as C4.5) is part of our research agenda.

For reason of space, many of the details that make this work a non-straightforward compilation of previous techniques are omitted (though we managed to add some that were explicitly asked by reviewers). For example, incremental TCP and IP pattern may seem a best-case pattern in terms of hash collision. However, we did several tests with different patterns (random, 5-tuples of real traces from both backbone and edge routers) and the results were similar (note that 14.2 Mpps filling a 50M sized hash table produce a sizeable amount of collisions) but not included due to lack of space.

Focusing on traffic classification issue, we believe our proposed approach to bring significant advance with respect the state of the art. Indeed, current DPI over GPU approaches are forced to pass through main memory to transfer data between the NIC and the GPU creating a bottleneck (and wasting processing time). Although there are preliminary results which may avoid such limitation, they can be only used with Infiniband technology [21], while our solution apply to off-the-shelf hardware. Moreover, statistical methods only need packet headers and, therefore, they reduce the amount of data to transmit between the NIC and the GPU by definition. If statistical identification techniques were suitable to replace DPI technology in IDS context, [28] would be able to reduce the gap with our work.

Finally, we better summarized the lesson learned, so as to give general design guidelines whose extent goes beyond that of the traffic classification issue we examined in this paper. As such, while we identify our major contribution is in eliminating bottlenecks in multi-queue system, we also pinpoint where single RSS queue may be preferable to multi-queue in some cases. Namely, whenever a single CPU is enough to cope with line-rate processing, the lower CPU usage translate into a lower power consumption and thus carbon footprint, and additionally avoids packet reordering issues due to multi-queue[31].