Netbench: Framework for Evaluation of Packet Processing Algorithms

Viktor Puš
CESNET, a. l. e.
Zikova 4, Prague, Czech Republic
pus@cesnet.cz

Jiří Tobola, Vlastimil Košař, Jan Kaštil, Jan Kořenek
Faculty of Information Technology
Brno University of Technology
Božetěchova 2, Brno, Czech Republic
{itobola, ikosar, ikastil, korenek}@fit.vutbr.cz

ABSTRACT

Many algorithms and hardware architectures are proposed to increase processing speed of time-critical operations in the field of longest prefix matching, packet classification and regular expression matching. Despite this fact, there is still no free and easily extensible platform for evaluation, comparison and experiments with existing approaches. We propose the Netbench Framework which aims to serve as an independent platform for researchers seeking the easiest way to implement their algorithms, as well as the comparison of their algorithms with reference implementations of other approaches. The framework is provided as an open source and can be easily extended to support new algorithms or new comparison methodology. Netbench is publicly available at `http://www.fit.vutbr.cz/netbench`.

Categories and Subject Descriptors

D.2.6 [**Software Engineering**]: Programming Environments—*Programmer workbench*

General Terms

Design, Performance, Complexity

Keywords

IP Lookup, Longest Prefix Match, Packet Classification, Regular Expression Matching, Library, Python

1. INTRODUCTION

As the speed of network lines increases, there is a need for new algorithms and architectures for acceleration of time-critical operations used in network devices. In current networks, the IP lookup, packet classification and regular expression matching are highly important and ubiquitous operations. All three listed operations are often required to work at wire-speed and therefore may become a bottleneck, or may incur high costs of high-performance devices. While technologies are rapidly innovated, algorithms may often only benefit from technology improvements and remain the same. We argue that research of algorithms is highly needed to keep pace with the industry requirements.

New algorithms from the discussed fields are often published in renowned proceedings and journals. In this situation, it may be surprising that there exists no common platform for researchers to evaluate and compare their algorithms. Moreover, access to real data sets is often limited due to security and confidentiality issues. The lack of standard data sets renders it hard to compare the properties of algorithms, especially their memory requirements. Implementations of published algorithms are often not available, so running own tests includes also reimplementation of previous approaches presented by other authors. Such reimplementation may be imperfect and differ from the original implementation, so it can not be fully trusted.

These facts contribute to lower quality of published research results. The Netbench Framework aims to address the stated issues by providing common platform for implementation and evaluation of packet processing algorithms, specifically IP lookup, packet classification and regular expression (RE) matching algorithms.

At the time of writing, there is no platform for early experiments with packet processing algorithms. The most related work to Netbench is the Click modular router [4]. It is a software implementation of a router with modular and configurable architecture. The Click router consists of a number of elements connected in a packet processing pipeline. Each element is a C++ class and implements single router function such as packet classification, queuing, and others.

In the viewpoint of previous works, Netbench aims at experimenting with inner function of these elements – leaving aside issues related to performance and complexity of the whole pipeline. This renders it very simple to prototype and investigate new packet processing algorithms.

2. NETBENCH FRAMEWORK

We present goals of the framework, together with explanation of how these goals are fulfilled. The Netbench Framework is designed with several objectives in mind:

- To serve as a uniform independent platform for researchers, without being tightly associated to one research group or algorithm.
- To enable rapid prototyping of algorithms and their software models.
- To simplify tasks such as parsing data sets and rule sets from files.
- To provide comprehensive data sets for IP lookup, packet classification and RE matching.

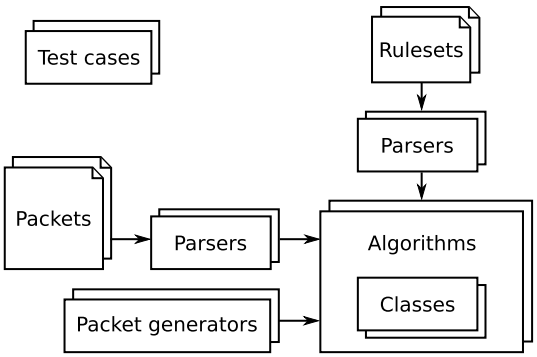

Figure 1: The Netbench Framework structure.

- To serve for education purposes.

We make the framework independent by making it publicly available under open-source license and by inviting researchers to submit their contributions.

We choose the Python 2.6 programming language to implement the algorithms, because of its popularity, sharp learning curve, and its feasibility for rapid prototyping of complex systems. Python is also considered to be the language with clear and easily readable syntax. If the performance of the Python interpreter is too slow for some particular computation, it is still possible to implement time-critical part of the algorithm in the C language. However, Netbench is intended to be used for rapid prototyping and experimental work, not for real deployment in the network.

To simplify common tasks such as loading data sets from files, Netbench provides set of classes in the form of library together with the documentation generated by the Sphinx [2] tool. There are also classes representing basic data structures, such as prefix, classification rule, regular expression, automaton, packet header etc. Overall structure of the Netbench Framework is in Figure 1. Comprehensive selection of algorithms for IP lookup, packet classification and RE matching is already available in the Netbench Framework.

2.1 Data sets and algorithms

We add several data sets to Netbench. For IP lookup, we focus on routing and firewall tables (both IPv4 and IPv6). For packet classification, rule sets generated by ClassBench [6] with different settings are used. Data sets for the pattern matching include several sets of rules originating from Snort [5], Bro [3] and other.

Netbench currently implements 10 longest prefix match algorithms, 4 packet classification methods and 7 regular expression matching algorithms.

3. USE CASES

Netbench is an excellent tool for comparing various aspects of algorithms. It helps researchers to easily evaluate memory requirements, input data set dependencies, suitable parameters settings, advanced rule sets properties, resource utilization for HDL design generators and more. For example, if researchers design new LPM algorithm they can easily compare memory usage of all current methods on the same data sets. Figure 2 gives such a comparison of recent LPM algorithms for current IPv6 routing table [1].

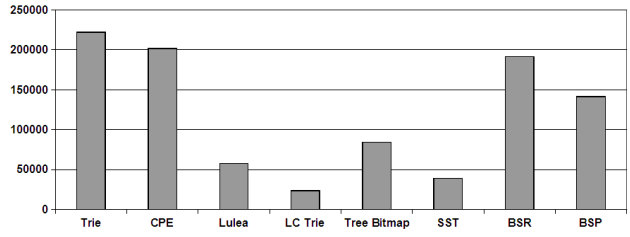

Figure 2: Comparison of memory usage for IPv6 routing table (bytes)

4. CONCLUSION

We present new framework for evaluation and rapid prototyping of packet processing algorithms. While previous works in this field focused more on the feasibility for deployment, Netbench targets the early stages of algorithm development when the ease of use and rapid prototyping are highly appreciated. Extensive set of algorithms was implemented into the framework in order to provide reference implementations and allow comparison of new approaches to existing algorithms. Moreover, the framework is designed to easily combine features from different algorithms.

Netbench also aims to define a baseline standard in data sets for evaluating new and existing algorithms, but does not restrict usage of other data sets. This should contribute to better quality of the published results in the field. We envision that every newly published algorithm for IP lookup, packet classification, and regular expression matching is included to Netbench. Therefore all researchers are invited to submit new algorithms, patches, data sets or suggestions to email address `netbench@fit.vutbr.cz`. After a review, these patches will be added to the framework.

Acknowledgment

This research has been partially supported by the Research Plan No. MSM, 0021630528 – Security-Oriented Research in Information Technology, the grant BUT FIT-S-11-1 and the CESNET Large Infrastructure project funded by the Ministry of Education, Youth, and Sports of the Czech Republic.

5. REFERENCES

[1] Bgp table data. `http://bgp.potaroo.net/`, 2011.

[2] Sphinx: Python Documentation Generator. `http://sphinx.pocoo.org/`, 2011.

[3] Bro IDS. Project WWW Page. `http://http://www.bro-ids.org/`, 2011.

[4] R. Morris, E. Kohler, J. Jannotti, and M. F. Kaashoek. The click modular router. In *Proceedings of the seventeenth ACM symposium on Operating systems principles*, SOSP '99, pages 217–231, New York, NY, USA, 1999. ACM.

[5] Snort. Project WWW Pages. `http://www.snort.org/`, 2011.

[6] D. E. Taylor and J. S. Turner. Classbench: a packet classification benchmark. *IEEE/ACM Trans. Netw.*, 15(3):499–511, 2007.