Framework for Network Management to Support Simulation of Varying Network Conditions

Petr Praus*
Department of Computer Science and Engineering
Faculty of Electrical Engineering
Czech Technical University
Charles square 13, 121 35, Prague, Czech Republic
prauspet@fel.cvut.cz

Abstract

Modern peer-to-peer systems are increasingly complex yet their testing still requires controlled and repeatable environment. The complexity of these systems and their protocols severely lowers feasibility of simulation in this regard. The solution that permits the use of the actual applications and associated protocols in the controlled and repeatable environment is emulation. Emulation allows real hosts to communicate through a transparent virtual network modelling the real network. However, the virtual network is often created by the emulation layer placed in the operating system's kernel posing a certain challenges regarding practical usability – the features needed for creating the environment are there but a significant expertise is required to properly use them. In addition, the components in the kernel are often primarily concerned with practical traffic shaping and Quality of Service in general rather than emulation. This work presents a framework significantly simplifying the creation and configuration of the network emulation environment while pointing out and helping avoid some common pitfalls an unsuspecting user might run into.

Categories and Subject Descriptors

C.2.1 [**Computer-communication Networks**]: Network architecture and Design—*Distributed networks*;
C.2.5 [**Computer-communication Networks**]: Local and Wide-Area Networks—*Internet*

Keywords

networks, peer-to-peer, emulation, linux

*Bachelor degree study programme in field Software Engineering. Supervisor: Tomáš Černý, Department of Computer Science and Engineering, Faculty of Electrical Engineering, CTU in Prague.

Praus, P. Framework for Network Management to Support Simulation of Varying Network Conditions. Information Sciences and Technologies Bulletin of the ACM Slovakia, Special Section on the ACM Student Project of the Year 2011 Competition, Vol. 3, No. 4 (2011) 47-48

1. Introduction

Motivation for this work stemmed from the need for efficient network emulation for Cooperative Web Cache project [1, 2]. We needed a way to test our implementation on a larger network of at least tens but preferably hundreds of network nodes. Our resources were very limited and we needed the network emulation to work in the scope of a very few real machines where one physical machine would represent several Cooperative Web Cache nodes. Fortunately, requirements of a single CWC node were relatively slim and we soon found out it is possible to run about forty CWC nodes on a dual-processor machine with 4 gigabytes of memory without hitting swap or incurring CPU load that would skew our measurements. This approach looked feasible and we set about finding the best way to emulate a real network environment on a Unix-like system in the scope of one machine. Both Linux and FreeBSD have their own IP bandwidth management and network emulation solutions with features often comparable to those of commercial offerings. We eventually chose the Linux solution because we are by far more adept at GNU/Linux administration than *BSD systems.

The work aims to convey experience in network emulation of complex P2P applications using Linux Traffic Control while serving as an intelligible guide to the technology. The inseparable goal of the project is object-oriented pure-Python (without any C extensions or wrappers) framework called *ShaPy Framework* for accessing traffic control capabilities of the Linux kernel using the Netlink interface. The framework is used in *Shapy Emulation* – an extension that makes construction of virtual networks for emulation a breeze. Python is becoming the language of choice for Linux system administration and offers seamless integration with BSD sockets essential for a Netlink communication making Python a natural choice for Linux traffic control configuration.

2. ShaPy

ShaPy is a framework for programmatic control of Traffic Control capabilities within the Linux kernel. ShaPy consists of two parts: framework and emulation. The Framework comprises of a set of classes representing various elements of Linux Traffic Control (Qdisc, Class, Filter, Interface) that can be tied together in a tree structure with interface as a root. Context in which Traffic Control works can be seen on figure 1. The second part builds on Framework and enables you to simply state what upload and download speed should be available for a transmission

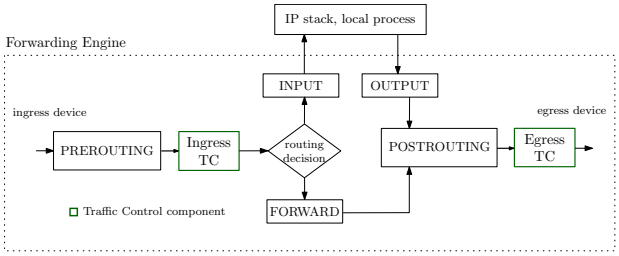

Figure 1: Linux forwarding engine

of packets with the given IP address. ShaPy Emulation also supports traffic aggregation from multiple interfaces using Intermediate Function Block – a virtual device available in Linux. Both parts have extensive unit tests.

Listing 1 shows a very simple example of ShaPy Emulation usage. This small piece of code is basically all you need to locally emulate homogenous network with two machines (represented with IP addresses 10.0.0.1 and 10.0.0.2). Providing you have the library installed, you can run this code with Python interpreter.

Basic usecase of ShaPy Emulation is that you have a process which you know communicates through a certain IP address. You might think that is a serious limitation but Linux supports a relatively convenient way to force a process to communicate over IP network with specified IP address using `LD_PRELOAD` without altering the program itself in any way. Details about this procedure can be found in bachelor's thesis of my colleague, Slavka Jaromerska[4].

```
from shapy import Shaper
shaper_conf = {
    ('10.0.0.1',) : {'upload': 256,
                     'download': 1024,
                     'delay': 5},
    ('10.0.0.2',) : {'upload': 128,
                     'download': 512,
                     'delay': 30},
}
sh = Shaper()
sh.set_shaping(shaper_conf)
```

Listing 1: Simple ShaPy Emulation example

2.1 Installation

ShaPy is an opensource project released under MIT licence. Its repository, including entire development tree from the initial proof of concept is available on GitHub[1]. ShaPy is also available on PyPI.

2.2 NetEM

NetEm [3] is Linux kernel module (more precisely, queueing discipline) very frequently used in ShaPy. It provides features for network emulation of wide area networks. It was originally conceived for the purpose of testing Linux 2.6 TCP Vegas implementation and currently supports variable delay, loss, duplication and re-ordering of packets. Availability starting from Linux 2.6.7 (June 2004) guarantees good support in any modern GNU/Linux distribution. The simplest case is adding a fixed delay to every packet that reaches a netem qdisc. Of course this does not reflect how real networks behave but for the sake of repeatable measurements and simplicity ShaPy uses just this for now. More advanced techniques are available for the advanced simulation – e.g. variance, correlation and Gaussian distribution of delay.

[1] https://github.com/praus/shapy

3. Future work

At present, ShaPy is only capable of emulating homogenous virtual network in terms of link latency. This means each node has fixed delay in all directions and thus it is not possible to create clusters of nodes that are close to each other but distant to other nodes outside the group. The required functionality is there but was not implemented so far. Most functionality in ShaPy is built around generic, object-oriented Netlink interface which could be leveraged for more than just Linux Traffic Control. Unit test coverage could use some improvement, especially for Netlink interface code and some more complex interactions between components. Also, the current implementation of the classes modelling the traffic control elements does not support all features of the underlying elements in the kernel.

4. Conclusion

We needed a way to test our research project – a large peer-to-peer network for cooperative sharing of static web resources. Since our budget was nearly nonexistent, we needed a way to test more nodes of the peer-to-peer network on one physical machine. Traditional virtualization methods would incur too much overhead, especially since the only thing we needed in such an early stage was the network separation of nodes. We understand our need is probably shared by many other researchers in the field and therefore I decided to generalize our solution in my bachelor's thesis. The final product is ShaPy, a Python framework for local network emulation released under MIT licence[2].

Acknowledgements. I would like to voice my great appreciation to Ing. Tomáš Černý M.S.C.S. for guiding me through such a complex task and my team members Slavka Jaromerska and Lubos Matl for their insightful remarks and overally very enjoyable cooperation.

References

[1] T. Cerny, S. Jaromerska, P. Praus, L. Matl, and J. Donahoo. Cooperative web cache. to be published on SOFSEM, 2012.

[2] T. Cerny, P. Praus, S. Jaromerska, L. Matl, and J. Donahoo. Cooperative web cache. In *18th International Conference on Systems, Signals and Image Processing*, 2011.

[3] S. Hemminger et al. Network emulation with netem. In *Linux Conf Au*, 2005.

[4] S. Jaromerska. Environment for peer-to-peer application simulation with application on cooperative web cache. Czech Technical University in Prague, Faculty of Electrical Engineering, Bachelor thesis, 2011.

[2] MIT licence means anyone can use the program for whatever purpose, even commercial one