Increase the Latency: Emulation Approaches for Real Network Conditions

Lukas Haneberg, Eric Hauser*, Sebastian Gallenmüller*
*Chair of Network Architectures and Services, Department of Informatics
Technical University of Munich, Germany
Email: ga92nen@mytum.de, hauser@net.in.tum.de, gallenmu@net.in.tum.de

***Abstract*—The increasing complexity of network setups in Information Technology fields and companies directly increases the need for constant testing and validation of the network. It is often important for the main network, or the production network, to be running constantly without interruptions, and to be running at full capacity without the additional overhead of testing software. In order to create a testbed that closely resembles the complexity of a production network, network emulation must be utilized. Another way of bridging the problem of missing network accuracy is network simulation, but the findings in this paper suggest that by using a emulation network setup, including actual traffic as opposed to simulated traffic, the production network can be emulated more accurately. In order for a testbed to accurately emulate the real latency behavior of network flow through a real Wide Area Network (WAN), multiple technological approaches can be put to use. This paper compares each of these different approaches and evaluates them based on emulation ability, costs and configuration efforts.**

***Index Terms*—network emulation, software link emulators, hardware wan emulators, fiber optic delay lines**

1. Introduction

Testing, validating, and implementing new technologies in computer networks is a core aspect of achieving a solid network infrastructure. The optimal way of doing so is to carry out the testing in the actual production environment, since this is the actual network setup through which all of the live traffic flows, and thus is already setup in the optimal way. But since testing in the production environment can also result in major downsides, such as unwanted crashes and down-times of nodes in the network, this does not present the ideal way of testing or expanding a network [1].

The solution to this problem is to set up a testbed, which is an environment that mirrors or imitates the production network, so that new technologies, expansions in soft- and hardware and general testing of the network devices can be tested. Since a testbed network setup is usually hosted in a single location, this provides the setup with overly ideal conditions. The cable lengths are short, usually staying below 100 m, and there are no additional network nodes, which would be introduced if the network was geographically separated and had to utilize WAN connections. Connections over a WAN can impose delays, latency and other types of interferences simply due to additional network nodes in the network, or propagation delay induced by longer cable routes.

This poses the question of how a testbed can reproduce the characteristics of a live environment, especially over longer distances, in order to make the testing even more accurate. A common answer to this question is network simulation. Network devices are modeled in a virtual environment, which allows for simulation based testing [2]. This is a valid first step in order to test the functionalities and study the behaviors of the network. However, network simulation quickly reaches its limits, since it fails to reproduce the real conditions imposed on the production network.

We propose to use network emulation, which introduces additional factors in terms of actual network devices, while still keeping the valuable aspects of a simulation environment [2]. In order to test the hardware and software, and various types of behaviors within the network over a longer distance, and additional number of devices in the network while still remaining in the local testbed, emulation methods have to be introduced. Traffic over a WAN, for example, introduces latency, packet loss, delays etc. This can be achieved by a variety of methods and techniques, for example a simple Linux device running NetEm, an extension of the already available network manipulation functions of Linux, which can be used to add fixed amounts of delays with additional random latency variation to outgoing packets [3].

This paper focuses on emulation techniques in order to accurately simulate a production environment and analyzes their benefits and limitations in terms of introducing actual network behaviors such as the latency introduced by the cable lengths in WANs. The structure of the paper is as follows: Section 2 lists and compares free software link emulators. Section 3 lists and compares all-in-one hardware WAN emulators. In the following Section 4, a comparison of the functionalities of free software emulators and hardware emulators is made. Section 5 discusses the use of fiber optic delay lines in fiber networks. Finally, Section 6 summarizes and contrasts all of the previously mentioned approaches.

2. Free Software Link Emulators

Software link emulators are software based tools, which require underlying hardware to run on. A PC is enough to install and run software emulators, most of them being integrated into the operating system environment. They are the most commonly used tools for network emulation, as they are usually free and open source [2]. Software emulators can be versatile tools and are useful

doi: 10.2313/NET-2023-06-1_08

for network emulation based testing, however their functionalities reach their limits quickly, as high line rates are not realistic and the underlying hardware is not dedicated to the emulation software. In the following sections, the most popular and promising software link emulators are discussed and compared.

2.1. NetEm

NetEm is a free extension to the already existing Linux Traffic Control package. It is used for its emulation functionalities for simulating the characteristics of a WAN, making it an important tool for testing. Command line parameters allow introducing latency to outgoing packets, packet loss, corruption, re-ordering and control bandwidth through rate control. This paper investigates the latency functions of Net Em. Installation is kept simple, when using a Linux kernel 2.6 or higher, it is already enabled. In older kernel versions the implementation is also simple and can be enabled in the Networking Options [3].

Listing 1: NetEm Delay with non purely random variation

```
# tc qdisc change dev eth0 root NetEm
delay 100ms 10ms 25%
```

The command line snippet Listing 1 is taken out of the official NetEm for Linux documentation [3]. The 100 ms parameter adds a fixed delay of 100 ms. The 10 ms adds or subtracts additional 10 ms in a purely random fashion, meaning the outcome of the delay value is 90 ms or 110 ms. Because WAN connections do not usually behave in a purely random fashion, the 25 % percent parameter is necessary to approximate real variation, meaning that the next random element of the delay is dependent on the previous outcome by 25 % [3].

2.2. Nist Net

Nist Net is a free Linux-based package used for network emulation. Much like its descendant NetEm, which was mentioned previously, it runs as an extension of the Linux kernel. It can be used to test and simulate a wide array of WAN properties, such as packet loss, bandwidth limitations, latency, and network congestions. The word "emulation" is defined by the two creators Mark Carson and Darrin Santay as the testing of a network in a simulated environment in addition to a real hardware network setup [4]. The real component in this sense is the actual machine running Nist Net, and the simulated component being the simulation factors such as introducing delays or bandwidth limitations in a logical sense inside of the Nist Net package. Therefore, Nist Net, much like NetEm, benefits from the simulation environment, which is easily changed and reproducible, but also benefits from the factors of a real network device setup [4].

The implementation of Nist Net is simple, as it is implemented and configured through a Linux operating system command line, and, for example, only requires a simple PC-router setup in order to use its emulation functionalities. By changing parameters in the Nist Net configuration, the user can define the desired network manipulation rules in a table of emulator entries, in which the user must also specify for which packets the rules apply, and which emulation factors should be applied [4]. The delay parameter sets the delay of incoming packets in milliseconds and has multiple parameters which can be applied to the delay behavior. For example the added delay can be static, following a random distribution, or by default follow a right tailed delay distribution which closely resembles the actual distribution of ping delays, tested in a three hour connection of machines in a Network, as observed by the authors of Nist Net [4] .

Listing 2: Configuration of Delay in Nist Net

```
# cnistnet -a 0.0.0.0 0.0.0.0 --delay 60
```

The delay parameter in Listing 2 adds 60 ms of simple delay to all traffic passing through a network node running Nist Net [5].

2.3. Dummynet

Dummynet is a network emulation tool developed in the late 1990s [2]. It was originally designed for running configurable experiments in network setups and has been developed for FreeBSD, a Unix-like operating system, for which it later became a default package. Across the years the support for other operating systems was expanded, now supporting Linux, MacOS, and Windows [6]. Dummynet works as a network emulator by utilizing pipes, which are used as a communication link with configurable bandwidth and other factors, such as delay [6]. These pipes are combined with different queuing methods which simulate those used by actual network devices, with FIFO queues being the default setting of Dummynet [6] [2]. The user can choose which traffic gets routed through which pipe, or alternatively, set up a pipe, which accepts any traffic. By defining the parameters of a pipe, for example the delay parameter, any traffic set to pass through the pipe has these parameters applied. The following command snippet shows how to define a simple delay of 60 ms and route all traffic through a pipe, the emulated link [6].

Listing 3: Adding simple delay and configuring traffic to a pipe in Dummynet

```
# ipfw pipe 1 config bw 2Mbit/s delay 60ms
# ipfw add pipe 1 ip from any to any
```

The parameter "bw 2 Mbit/s" in Listing 3 defines the bandwidth and the following "delay 60 ms" sets a delay of 60 ms to pipe 1. The second line routes all the traffic through pipe 1 [6].

Dummynet can be set up as a network router, which accepts the incoming taffic, applies the emulation factors and then forwards it to the network. It can also be established in a distributed fashion, where every device in the network must have Dummynet enabled [2].

2.4. Comparison of NetEm, Nist Net and Dummynet

In some network setups it might be sufficient to test the network behavior under the effects of simple constant latency, but this is sometimes not enough for more complex network setups and testing them under real network conditions. With Dummynet it is only possible to

doi: 10.2313/NET-2023-06-1_08

add a constant amount of delay to a given pipe. It does not offer more complex distributions for the latency that are possible in Nist Net and NetEm. When performing simple experiments in a emulated network, Dummynet is definitely sufficient for adding constant latency, but it cannot accurately simulate the latency behaviors of a real network setup with WAN characteristics like it is possible in Nist Net and NetEm [7].

In NetEm and Nist Net, the user can define the delay distribution to be constant, or alternatively specify a more complex distribution so that the latency follows a real latency behavior more accurately. In this regard, Nist Net and NetEm are similiar, allowing for the latency to follow all kinds of distributions with optional correlation [2].

Emulation accuracy is also an important factor to consider, especially for emulating latency behaviors in a network setup. The more accuracy the emulation is able to achieve, the better one can test and approximate the network setup towards the real environment. In the context of the previously mentioned free network emulation tools NetEm, Nist Net and Dummynet, this accuracy is achieved by the used time resolution of the kernel clock [7]. Dummynet is able to utilize the system clock at the maximum of 10 kHz, whereas Nist Net and NetEm under Linux are only able to achieve up to 1 kHz [2]. In more recent Linux kernel versions, NetEm can utilize High Resolution Timers, providing more accurate emulation effects [7].

The point of emulation plays a role in deciding for the optimal setup of a emulation network. Dummynet is able to emulate inbound and outbound traffic, and can be set up as a network router with emulation functionality, or in a distributed way, in which every device in the network will run a copy of Dummynet [2]. NistNet on the other hand is only able to emulate inbound traffic, while NetEm can only emulate outbound traffic [7]. NetEm is effectively setup in a distributed fashion, in which every device has to run NetEm for the emulation network to function properly, whereas Nist Net must be setup as a router between network nodes [2].

It is important to note that Nist Net is currently not in active development. Drawing a comparison for the three tools, Nist Net and NetEm have similar traffic impairment functions, with the possibility of latency distributions. Dummynet on the other hand only offers basic delay and impairment functionalities. Keeping in mind the active development of NetEm and Dummynet combined with their usability, the decision of a free software emulator definitely falls between either Dummynet or NetEm. Including the accurate emulation functionalities and other features useful for network emulation which Dummynet does not have, the choice should fall on NetEm followed closely by Dummynet as the best software emulators.

3. All-in-one Hardware WAN Emulators

While a software emulator setup on a PC could technically also be specified as a hardware emulator, the all-in-one hardware solutions discussed in this paper differ from the commonly used software emulators in regard to their underlying execution platform. While the software emulators running on a PC setup are limited to the resources and computing power provided by the underlying PC, which varies depending on other current system tasks, the all-in-one hardware solutions run on hardware dedicated to the network emulation tasks. Additionally the operating system of the all-in-one solutions are typically optimized towards network emulation as well [8] [9] [2]. The execution of tasks in the all-in-one solutions may also be offered in a Field Programmable Gate Array (FPGA) based environment, which results in even higher execution and emulation speeds of network effects, since the execution of some tasks is directly carried out on the FPGA hardware, instead of relying on higher level software [2] [10].

The specific Hardware Emulators discussed in this paper, however, will be those typically used in a network setup in a corporate environment, which has additional functionalities compared to the available free software emulators. These Hardware Emulators are costly, and, therefore, only a few of them will be used in such environments, in most companies only one of them will be used in order to emulate the whole network [2]. The installation effort of commercial all-in-one hardware emulators compared to the software emulators is much simpler, they need to be physically installed in the testbed setup and can be configured through user friendly GUIs. Additionally, the companies offering these all-in-one solutions often include additional services in case of faulty equipment, or troubles while installing the devices. The software emulators are targeted towards the "do it yourself" networking specialists, and require a certain skill set in networking, such as the Linux traffic control package, and routing the network flow. Additional configuration also needs to be expected when a software emulator is to be used. Where the previously discussed free software emulators and hardware emulators drastically differ, are the speeds at which they support emulation. The software emulators support the speeds of its underlying computers, which will usually remain under 1 Gbit/s, while the Hardware Emulators such as the Netropy 10G or Netropy 100G are built for speeds of up to 10 Gbit/s and 100 Gbit/s respectively, which makes them very suitable for a real network emulation setup [2] [8].

3.1. Apposite Technology Netropy 100G

Netropy 100G made by Apposite Technology allows speeds of up to 100 Gbit/s [8]. The Installation of the Emulator is straight forward. It is installed directly into the testbed, similarly to the installation of a switch or server. Its 100 Gbit/s emulation engine supports 15 concurrently running WAN connections, allowing for complex network setups and concurrent testing [8]. It offers network degradation features similar to the previously discussed free software link emulators, including a variety of bandwidth, latency, latency variation, and packet loss functionalities [8]. The device is able to impose anywhere from 0 to 10 000 ms of delay to each of its simulated WAN links, with the user being able to impose additional distributions similar to NetEm and NistNet, including normal, constant, uniform and many other distributions [8]. The Netropy device lineup also offers features such as a live traffic monitor, recording of loss and delay behaviors and most importantly configuration of the paths and emulation characteristics of the device [8].

3.2. Spirent Attero-100G

The Attero-100G is manufactured by the company Spirent and provides line delay of up to 256 ms for speeds of up to 100 Gbit/s. A delay rate of 256 ms on 100 Gbit/s emulation speeds, translate roughly to 50 000 km of a typical WAN, making it useful for emulating real WAN conditions [9]. It offers similar traffic impairment functions to the Netropy 100G, such as packet corruption, duplication, reordering, bandwidth manipulation, latency and jitter, and does so with a timing accuracy of 5 ns, providing a very swift emulation onto the packets [9]. Through the restful API, the user can reach the web based GUI, and manipulate impairment functions by configuring profiles [9]. The product comes with two profiles as a standard, providing emulation to one incoming and one outgoing stream, but this can be extended to 16 profiles with 8 incoming and 8 outgoing emulated packet flows concurrently [9]. The Attero-100G emulator offers a range of delay functions such as gaussian-, uniform- and not limited to gamma distributions, which can be explained as continuous probability distributions of the latency. The independent traffic flows allow for a different delay set up for each profile, allowing for a complex testing network setup [9].

3.3. Comparison of Hardware Emulators

The Attero-100G and Netropy 100G are both very powerful hardware WAN emulation solutions, and are the golden standard for accurate network emulation. The dedicated hardware, including the emulation engines, allows for emulation at high line rates of up to 100 Gbit/s. The configuration of both devices is as simple as it gets, and they offer an extensive GUI for configuration and monitoring. Because the functionalities of these two devices are so similar, the only decision to be made is how many concurrent connections the emulator should handle. The Netropy 100G is able to support 15 concurrent WAN connections, while the Attero-100G can support 8 connections. The available traffic impairment functions are almost identical, and the user is able to specify custom delay functions via the GUIs. The Attero-100G and Netropy 100G are both powerful solutions and the use of either of these will result in highly realistic emulation environment, perfect for reproducible testing [8] [9].

4. Comparison of Hardware WAN Emulators and Free Software Link Emulators

Having either an all-in-one Hardware WAN emulator or a high-end computer, set up with one of the previously mentioned free software emulators, up and running, the results of latency emulation does not differ significantly between the two. In terms of the ability to emulate a real WAN environment latency behavior, both solutions are able to attain similar results. Taking NetEm and the Netropy 100G for example, the functionalities are very similiar. The delay can be imposed in a simple fashion, but also in a more complex variation distribution. The speeds at which both, the hardware emulators, and software emulators are able to emulate depend on the hardware model, and the software emulators depend on the underlying computer, on which they run on [2]. This certainly also has to be noted, since software emulators realistically cannot reach high emulation speeds of up to 100 Gbit s, and reaching even remotely high speeds of around 10 Gbit s becomes costly, since the underlying PC must be able to provide high computing power for the emulation software. The Netropy 100G and the Attero-100G are specific models that have ports, which support up to 100 Gbit s right from the manufacturer with dedicated hardware and execution platforms [8] [9].

Performance can differ greatly between a software and all-in-one hardware emulator, depending on the hardware used for software emulation. A PC running a software emulator shares its CPU resources with all of the operating system related processes in addition to the emulation software. The CPU resources needed by other processes can vary greatly and in return lead to a lower performance of the software emulator in terms of emulation speeds. This becomes even more apparent when more connections and impairment functions are handled by the software, increasing the CPU load by significant amounts [2]. The hardware emulators on the other hand are specifically built and optimized for handling multiple connections with multiple impairment profiles at the exact speeds which are specified by the different models available [8] [9].

The Commercial all-in-one Hardware WAN Emulators are definitely more suitable for more complex network structures, since the setup, physical installation and configuration is straight forward. The software emulators are more complex to install, since they must be configured manually and the traffic flow through a PC running a software link emulator must be specified as well. The Attero-100G and Netropy 100G offer remotely accessible GUIs with ways to configure network degradation functions and real time performance statistics. This makes the commercial emulators useful for immediately evaluating the network behavior after imposing new delays, packet behaviors, etc. While these factors in terms of user friendliness and network evaluation are a clear improvement to the free software emulators, the high price of the commercial emulators weighs in on a decision between the two.

5. Fiber Optic Delay Lines

Fiber optic delay lines can be considered as another type of Network Emulation Hardware with regard to adding latency. They are another way of introducing fixed delay for network traffic, specifically in a optical network setup using glass fiber cables. A Fiber optic delay line functions by receiving input traffic through a fiber cable connection, then routing the traffic through the length of the spooled fiber cable inside of the device, and routing it back to an output channel [11]. The inherent characteristic of a fiber cable in being relatively resistant to interference factors, and having a low amount of propagation loss, makes this option suitable for adding a low amount of delay in a optical network [11]. Since the amount of delay added is proportionate to the length of the cable inside of the Fiber Optic Delay Line device, the introduced insertion loss will increase, meaning that the strength of light in the input is slowly lost through the length of the cables, but in

doi: 10.2313/NET-2023-06-1_08

most cases of hardware available, it is a negligible amount [12] [11]. A downside of using a fiber optic delay line is the increasing size and scale of the device, since the more delay is required, the more wound up fiber cable needs to be present inside of the device [12]. Another downside of this type of hardware are the amount of devices and ports needed to emulate a more complex network, since each fiber optic delay line needs its own switch port [12].

5.1. Fiberplus D8 Series

The Fiberplus D8 is a fiber optic delay line, which is installed in a server rack. The device can hold up to 45 km of fiber, or additionally 90 km of fiber cable spooled up inside of the device [13]. It offers network degradation emulation factors, such as loss, time delay and fiber emulation such as reflectance [13]. The amount of delay added is only available as a constant value, since the delay is not added artificially, but realistically, since the amount of cable that the traffic passes through is actually present. Since the propagation delay of fiber is about the vacuum speed of light [11], the total amount of delay added within the device will be in the example of the Fiberplus D8 in total up to 440 µs with a minimum of 12 µs [13].

6. Conclusion

In conclusion, network emulation is a valuable tool for accurately approximating a test network environment in terms of real network behavior. Accurately simulating the latency behavior of a real network setup composed of multiple network nodes, including WAN structures, is an important step to achieve these real conditions. Multiple approaches were listed and analyzed in this paper. This includes free software link emulators, such as NetEm, Nist Net, and Dummynet, which need an underlying computer to run its emulation software on, but also hardware emulators, such as the all-in-one devices Netropy 100G and the Attero-100G. The use of either hardware all-in-one solutions or software emulators, when setup properly, can achieve similar latency emulation results. Although doing so with high emulation speeds of 10 Gbit s up to 100 Gbit s, all-in-one hardware solutions must be used, as software emulators are limited to their underlying hardware, and realistically cannot reach these high speeds. The configuration efforts and acquisition costs vastly differ between the software and hardware approaches. On the one hand the software emulators discussed in this paper are free to use, they impose a more significant configuration effort and require a certain knowledge of its underlying operating system. On the other hand, the hardware emulators offer an all-in-one package, keeping the installation and configuration simple, and offering user friendly GUIs, with a drawback of high acquisition costs. For a large company, a all-in-one hardware package might be preferable, as one device can be sufficient to emulate a large scale network with high line rates, while the software solutions offer a more budget friendly approach. For fiber optic networks, fiber optic delay lines can provide realistic network emulation, as these devices house the actual length of cable inside, routing the traffic through the actual cable length and not just emulating it within software.

References

[1] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, N. Mckeown, and G. Parulkar, "Can the production network be the testbed?" in *9th USENIX Symposium on Operating Systems Design and Implementation (OSDI 10)*, 2010.

[2] R. Beuran, *Introduction to network emulation*. CRC Press, 2012.

[3] "NetEm," https://wiki.linuxfoundation.org/networking/NetEm, 2021.

[4] M. Carson and D. Santay, "Nist net: a linux-based network emulation tool," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 111–126, 2003.

[5] "NIST Net WAN Emulation Software Installation and Configuration Note," https://www.cisco.com/web/software/280836713/47304/NIST-v4.pdf, 2005.

[6] M. Carbone and L. Rizzo, "Dummynet revisited," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 2, pp. 12–20, 2010.

[7] L. Nussbaum and O. Richard, "A comparative study of network link emulators," in *Communications and Networking Simulation Symposium (CNS'09)*, 2009.

[8] "Netropy Network Emulators," https://www.epsglobal.com/Media-Library/EPSGlobal/Products/files/apposite/N40G-40G.pdf?ext=.pdf, 2017.

[9] "Spirent Attero-100G," https://www.spirent.com/assets/u/spirent_attero-100g_datasheet, 2020.

[10] P. Varga, L. Kovács, T. Tóthfalusi, and P. Orosz, "C-gep: 100 gbit/s capable, fpga-based, reconfigurable networking equipment," in *2015 IEEE 16th International Conference on High Performance Switching and Routing (HPSR)*, 2015, pp. 1–6.

[11] R. Paschotta, "Optical Delay Lines," https://www.rp-photonics.com/optical_delay_lines.html.

[12] T. Zhang, K. Lu, and J. Jue, "Shared fiber delay line buffers in asynchronous optical packet switches," *IEEE journal on Selected Areas in Communications*, vol. 24, no. 4, pp. 118–127, 2006.

[13] "D8 Compact Rack Mount Fiber Optic Delay Line," https://www.gofiberplus.com/Fiber-Optic-Delay-Lines_c_29.html, 2019.