Emulating NDN-based Multimedia Delivery

Daniel Posch, Benjamin Rainer, Sebastian Theuermann,
Andreas Leibetseder, Hermann Hellwagner
Institute of Information Technology
Alpen-Adria-Universität Klagenfurt
{firstname.lastname}@itec.aau.at

ABSTRACT

Today, the global share and increase of Internet traffic is largely caused by multimedia delivery, mainly encompassing video, audio and image sharing on social, news, and entertainment platforms. This fact is well known to the Internet research community, which tries to counteract by increasing the content delivery efficiency. So-called Information-Centric Networks (ICN) are of considerable interest, advertised as enablers for intelligent networks, where effective delivery is to be provided as an inherent network feature. Most research proposals in this area are evaluated in simulated environments, using simulation frameworks such as *OMNeT++* or *ns-3*. However, simulations always have shortcomings and cannot substitute measurements in physical networks. In this demonstration, we show how to readily set up an ICN-based testbed using low-budget single-board computers to conduct comprehensive emulations. We choose the scenario of pull-based adaptive video delivery as a showcase and evaluate the performance of different client-based adaptation mechanisms at the application level and different content forwarding strategies at the network level. All of the presented tools and visualization features are provided as open source contributions to the community.

CCS Concepts

•**Networks** → **Network simulations; Network experimentation;** *Network design principles;*

Keywords

Named Data Networking, Information-Centric Networking, Network Emulation, Adaptive Multimedia Delivery, Testbed.

1. INTRODUCTION

The concept of Information-Centric Networking (ICN) [1, 2] was proposed to deal with the growing demands on the Internet, especially with a focus on efficient content delivery. The fundamental idea of ICN is to enable data-centric communication, where named content objects are treated as first-class citizens. One concrete approach to realize an information-centric communication infrastructure is Named Data Networking (NDN) [3]. Here, the communication follows a strictly consumer driven pattern. Consumers emit so called Interest messages, carrying the name of the content a consumer is interested in. The network is responsible for forwarding these messages to a possible content source, which fulfills the request by replying with the corresponding Data packet. Possible content sources are: *i*) the content origin; or *ii*) so-called network inherent caches (intermediate nodes holding a replica of the requested object). Data integrity and authenticity of replicas is assured by content-based security mechanisms, usually relying on digital signature schemes [3].

Currently data-centric communication is a hot topic in the network research community leading to many published papers proposing new approaches for ICN-related challenges including: forwarding strategies, routing algorithms, caching policies, mobility schemes. The majority of the proposed methods is evaluated by their authors either via theoretical analysis or by conducting simulations using network simulation frameworks such as *OMNeT++* and *ns-3*. While both evaluation methods are valid, an additional evaluation on physical hardware is desirable, potentially exposing weak components, performance bottlenecks, and further challenges that are not observable in a synthetic environment.

The objective of this work is to provide researchers with a third option that readily allows to evaluate new approaches on a physical testbed. Besides the previously mentioned advantages, a physical testbed also enables the observation of parameters that cannot be observed during simulations, e.g., the power consumption of the devices. We provide a framework that allows researchers to build an NDN-based Future Internet testbed using low-budget single-board computers as the basis. While focusing on low costs, the selected hardware is powerful enough to conduct significant evaluations in nearly all research areas of interest regarding ICN. The costs for a testbed of sufficient size with about 20 nodes is approximately 3400 USD, linearly scaling with about 160 USD per additional node.

In the demonstration, we will present an NDN-based multimedia delivery scenario using adaptive bit-rate streaming as foreseen by MPEG-DASH [4] to substantiate our claims, visualizing the streaming session using a Web interface. Additionally, we provide a graphical user interface where demonstration visitors may create their own network topology, configure a streaming scenario and observe in near-

MMSys'16 May 10-13, 2016, Klagenfurt, Austria

ACM ISBN 978-1-4503-4297-1/16/05.

DOI: http://dx.doi.org/10.1145/2910017.2910626

real time the network traffic and received video quality by the consumers. The visitors may select among different network forwarding strategies and client-based adaptation logics to learn which configuration reveals the best performance in an NDN-based network.

The remainder of this paper is structured as follows. In Section 2 we discuss the architecture of the emulation network and present the open source tools used to perform and visualize a network emulation. In Section 3 we depict our demo setup, giving details on the chosen showcase and interaction possibilities for the demonstration visitors. Section 4 summarizes the paper and the planned demonstration.

2. EMULATION NETWORK

Figure 1 illustrates the architecture of the testbed. Resting upon an IP-based network, a virtual NDN-based overlay is created using the Networking Forwarding Daemon (NFD) [6]. The hardware requirements for the testbed are basically a number of single-board computers, (at least) two network switches and an ordinary Personal Computer (PC) acting as gateway server. The testbed consists of two dedicated networks which connect the single-board computers in a star-like topology. The so called *Management Network* (MN), indicated by the red lines, is used for configuration and monitoring. It is physically separated from the second network, which is denoted as the *Emulation Network* (EN). The EN is realized as a virtual network overlay that is created using the common networking tools such as *iptables* and *traffic control (tc)*. The clear separation between the MN and EN ensures that in no case management/control traffic may interfere with, and influence, running network emulations. The gateway is only connected to the MN and provides external communication and control functionally among the testbed and its user. Furthermore, the gateway shall provide the single-board computers with Internet access, allowing to easily receive new software packages or code updates. The gateway also maintains an Apache HTTP server, providing a Web interface that allows to observe and control network emulations in near-real time.

2.1 Hardware Considerations

As single-board computer we have chosen the Banana Pi Router (BPI-R1) [7]. Table 1 lists the most important hardware specifications of the Pi. We have chosen this model for two reasons: *i*) it is one of the few single-board computers that provides more than one Ethernet interface; *ii*) it maintains a SATA 2.0 interface, which allows to add a Solid State Drive (SSD). Typical single-board computers only maintain Micro SD cards as storage media, which have significant lower read/write rates than SSDs. Equipping the Pi with an SSD allows to perform network evaluations where nodes have to cache content larger than the main memory of the Pis (1 GB). We consider a testbed size of at least 20 nodes as necessary to perform significant network emulations. We have listed all required components for a testbed with 20 nodes in Table 2, excluding the hardware for the gateway. As for the gateway, we suggest to use an old/unused PC for cost reasons. Alternatively, another single-board computer may be used as gateway, however, with more constrained resources as compared to a classical PC.

2.2 Assembling the Testbed

Once all hardware components are available, the Pis need to be assembled, and the two networks illustrated in Figure 1 are created connecting the Pis with the network switches. The testbed should be placed in an air-conditioned environment, and the Pis should not be stacked on top of each other to avoid thermal issues. We suggest to place them upstanding with a small space between the individual cases as shown in Figure 2. We provide images [5] for the Pis on the basis of Bananian Linux. We strongly suggest to reuse these images as we customized the kernel to have all required Linux networking tools/features available required to enforce the virtual overlay network. There are two images available, one for the Micro SD card, and another one for the SSD. Please note that even when using the SSD, one still requires the Mirco SD card as boot device (the BPi-R1 is only able to boot from a storage media in the SD card slot). Furthermore, we also provide a 64-bit Ubuntu-based image [5] for the gateway having pre-installed/configured the Apache HTTP Server and all scripts required to perform network emulations and visualizing them in a Web interface. For a more detailed documentation we refer to [5].

COMPONENT	DESCRIPTION
CPU	ARM-Cortex-A7 (2x1.0 GHz)
GPU	Mali-400MP2
Memory	1 GB DDR3-SDRAM
Storage	1x Mirco SD, 1x SATA 2.0
Network	1x RJ45, 4-Port-Switch, 802.11b/g/n
Power Source	5 volt / 2 ampere via Micro USB

Table 1: Specification of a Banana Pi Router (BPI-R1).

COMPONENT	COMMENT	USD/UNIT
20x BPI-R1		80 USD
20x Case for BPI-R1	optional	15 USD
20x SSD	$\geq$ 120 GB	50 USD
20x microSD	$\geq$ 8 GB	4 USD
20x USB power cable	spec. for 2 Amp.	3 USD
4x USB power hub	$\geq$ 6 ports	20 USD
2x Gigabit switch	$\geq$ 24 ports	100 USD
40x Ethernet cable	CAT6, 5 ft	2 USD
	Total:	**3400 USD**

Table 2: Required components for a testbed with 20 nodes and their *approximate* cost per unit exluding the gateway.

2.3 Performing Network Emulations

As previously mentioned, we use common Linux networking tools to realize the virtual network overlay. Note that, if one decided to not reuse the available images, one still finds the required scripts at [5] for separate download. As a first step, a user may specify the overlay topology within the Web interface (cf. Figure 3), using one of the available scripts to generate a random topology, or by simply writing it down as a text file. The topology file basically specifies the number of nodes, their links, the available link capacities and link delays for the virtual overlay network. Basically this topology is then enforced on the EN by blocking physical links which are not specified in the overlay topology using the networking tool *iptables*. Using *traffic control (tc)* we build a hierarchy of token bucket filters and use multiple

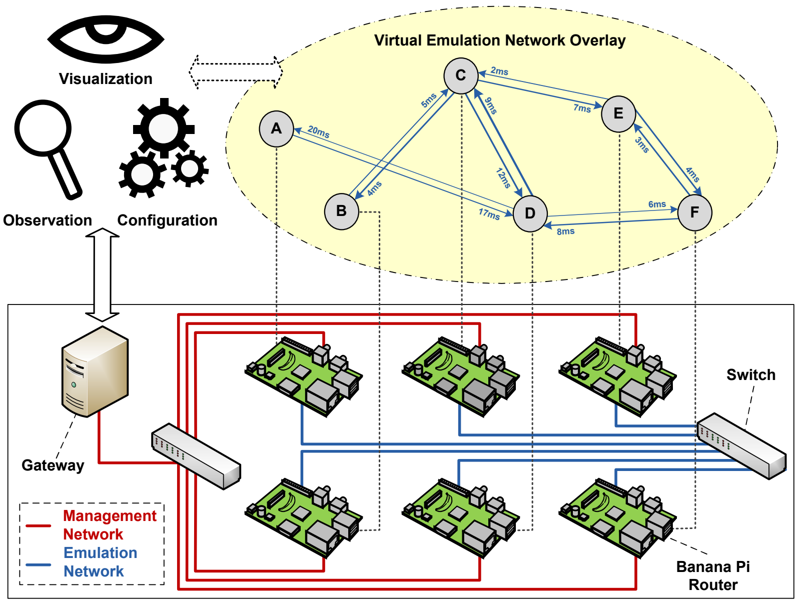

Figure 1: An overview of the testbed architecture. The testbed uses two dedicated networks. The *Management Network* (red lines) is used to setup and control the testbed nodes, while the *Emulation Network* (blue lines) is the virtual overlay for the network emulations. The gateway is used to configure, observe and visualize the emulation task.

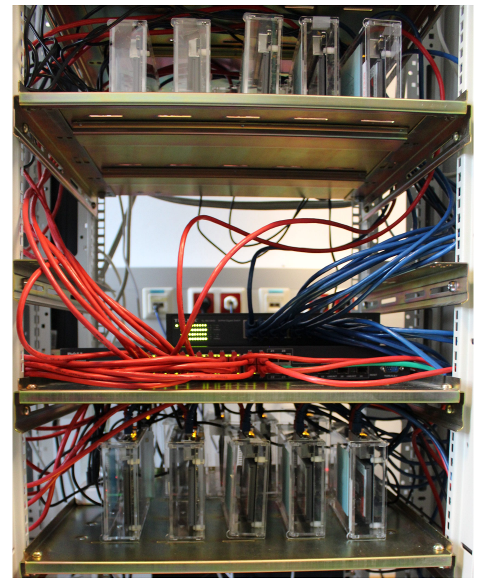

Figure 2: The figure shows the NDN-based testbed at ITEC [5]. It has a size of 20 nodes and consists of the components listed in Table 2. The single-board computers and switches are placed in rack within an air-conditioned environment.

queuing disciplines to enforce the specified bandwidth and delay parameters for each link in the overlay network.

Once the topology is enforced on the EN, the NFD is started on all Pis and routes (forwarding entries) are installed in the Forwarding Information Base [3]. The routes have been pre-calculated based on the provided topology files and the user may choose from installing all possible or only the shortest route(s) per prefix. As default, all possible routes are installed, since NDN communication foresees multi-path content delivery. In the next step, automatic logging is enabled on the Pis, which accumulates data about the Pis' most important attributes (CPU load, RAM usage, network traffic, etc.) during the emulation is in progress. Finally, the scripts trigger the start of the specified emulation applications, e.g., video streaming. As a starting point, we provide a simple consumer/producer application pair to enable a quick and easy start in performing first emulations on the testbed. Once the applications have finished, the recorded logfiles from the Pis are automatically gathered and stored on the gateway for later processing/analysis. During the evaluation, the Pis actively push some of the logfiles to the gateway for near-real time visualization (discussed in Section 3, cf. Figure 3).

3. DEMONSTRATION PROPOSAL

In this section we first outline the chosen scenario for the demonstration. Then we describe the demonstration itself and indicate the interaction opportunities for its visitors.

3.1 Scenario: Adaptive Multimedia Delivery

We have chosen the scenario of pull-based adaptive video delivery to present the capabilities of the proposed testbed. We generate network topologies using the Erdős Rényi model and assign random bandwidth resources and link delays. Furthermore, we have implemented an MPEG-DASH-like adaptive streaming client and server using NDN-based data communication to emulate a typical video streaming scenario in an NDN-based environment. Network nodes are randomly assigned as clients, servers and/or routers. Client nodes stream MPEG-DASH-compliant SVC-encoded [8] content from the servers. The SVC-encoded content provides three different quality layers using SNR scalability. The streaming clients implement different types of client-based adaptation mechanisms as foreseen by MPEG-DASH, including rate- and buffer-based strategies. Furthermore, we implemented/reuse several Interest forwarding strategies for the NFD to investigate the performance interplay between client-based adaptation mechanisms and network-based forwarding strategies (BroadCast [6], BestRoute [6], Stochastic Adaptive Forwarding [5], etc.).

3.2 Demonstration

The basic sequence of the demonstration is illustrated in Figure 4. First, a brief introduction to the testbed architecture is given to the visitors. We are going to use the poster board to illustrate the overview depicted by Figure 1. Then, we present our NDN-based Banana Pi Router testbed at ITEC [5] shown in Figure 2 consisting of 20 network nodes. Unfortunately, due to the large number of components and space required for the testbed hardware (cf. Table 2) we cannot take the entire testbed with us. However, we are going to bring one fully equipped Banana Pi Router, so visitors can get their hands on some of the components. The testbed itself will be accessed via a Virtual Private Network (VPN) using the provided Internet connection at the venue. Neither the configuration, execution, or the visualization of an emulation is delay sensitive or requires a lot of bandwidth. So, we expect no complication during the live demonstration

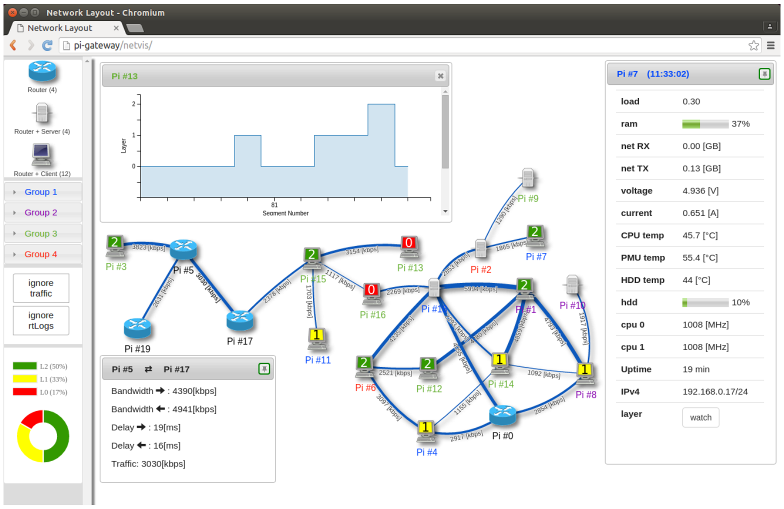

Figure 3: The figure shows the Web interface. Here users can configure the emulation scenario, as well as observe the testbed in near-real time.

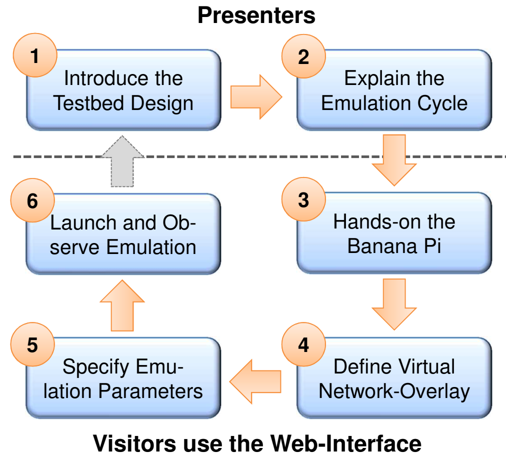

Figure 4: The figure shows the planned demonstration cycle for the demonstration.

of network emulations on the testbed.

The emulation of the multimedia delivery scenario can be configured by one or multiple volunteers among the visitors using the Web interface depicted in Figure 3. The volunteer(s) may vary the virtual network overlay by adapting the network topology, the link capacities and/or delays. This can be achieved by simple drag-and-drop functionality and the usage of sliders. Furthermore, the visitors may choose among two different client-based adaption mechanisms (a rate- and a buffer-based) for the implemented video streaming client. Also the Interest forwarding strategy used by the NFD can be chosen by the visitors. Once all emulation parameters are specified, the emulation is started by the visitors and they may observe the emulation in near-real time using the Web interface. Visitors may observe the achieved video streaming quality by the clients, link-resource utilization or inspect further relevant attributes of each individual single-board computer, such as CPU load, RAM usage, network traffic. As can be seen from Figure 3, the Web interface also provides the opportunity to investigate a history of the received video quality. Additionally to the live demonstration, we are going to present some previously obtained results from network emulations depicting the CPU load and power consumption of the single-board computers performing data communication using different Interest forwarding strategies on the poster board. From these results visitors may conclude which forwarding strategies should be used in various application scenarios, e.g., a mobile scenario with limited battery resources.

4. CONCLUSION

In this paper we proposed a demonstration of our NDN-based testbed using low-budget single-board computers. We presented the design and the architecture of the testbed and provide all necessary tools/images/scripts as open-source contributions [5], so interested researchers may use it as a framework for their own testbed. The demonstration includes the presentation of a live network emulation using the testbed realizing an NDN-based adaptive multimedia delivery scenario.

5. ACKNOWLEDGMENT

This work was supported in part by the Austrian Science Fund (FWF) under the CHIST-ERA project CONCERT (A Context-Adaptive Content Ecosystem Under Uncertainty), project nr. *I1402* at Alpen-Adria-Universität Klagenfurt.

6. REFERENCES

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, “A Survey of Information-Centric Networking,” *IEEE Communications Magazine*, vol. 50, no. 7, pp. 26–36, 2012.

[2] G. Xylomenos, C. N. Ververidis, V. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, G. C. Polyzos *et al.*, “A Survey of Information-centric Networking Research,” *IEEE Communications Surveys and Tutorials*, vol. 16, no. 2, pp. 1024–1049, 2014.

[3] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, “Named Data Networking,” *ACM SIGCOMM Com. Comm. Review*, vol. 44, no. 3, pp. 66–73, 2014.

[4] I. Sodagar, “The MPEG-DASH Standard for Multimedia Streaming Over the Internet,” *IEEE MultiMedia*, vol. 18, no. 4, pp. 62–67, 2011.

[5] Institute of Information Technolgy (ITEC), “ICN @ ITEC,” last accessed: Jan. 2016. [Online]. Available: http://icn.itec.aau.at/

[6] A. Afanasyev, J. Shi, B. Zhang, L. Zhang *et al.*, “NFD Developer’s Guide,” Technical Report NDN-0021, 2014. [Online]. Available: http://named-data.net

[7] Banana Pi Team, “The Banana Pi: Single-board Computers,” last accessed: Jan. 2016. [Online]. Available: http://www.bananapi.com/

[8] C. Kreuzberger, D. Posch, and H. Hellwagner, “A Scalable Video Coding Dataset and Toolchain for Dynamic Adaptive Streaming over HTTP,” in *Proc. of the 6th ACM Conference on Multimedia Systems*, 2015.