End-to-End Performance Evaluation of Virtual Networks Using a Prototype Implementation

Asanga Udugama[1], Liang Zhao[1], Yasir Zaki[1], Carmelita Goerg[1], and Andreas Timm-Giel[2]

[1] Communications Networks, TZI, University of Bremen
Otto-Hahn-Allee, 28359 Bremen, Germany
{adu,zhaol,yzaki,cg}@comnets.uni-bremen.de
[2] Institute of Communication Networks, Hamburg University of Technology, Schwarzenbergstr. 95E, 21073 Hamburg, Germany
timm-giel@tuhh.de

Abstract. Network virtualization is a concept where physical resources are used to create virtual resources that are combined to form virtual networks. As one of the key enablers of the future Internet, network virtualization solves a number of issues associated with today's networks. Concepts of network virtualization that are not restricted to virtualization technology, termed as overall concepts that include roles of parties and deployment aspects are currently being defined in different research activities. An area that lack attention is the evaluation of performance in prototypes that consider these overall concepts of network virtualization. The work presented here discusses and presents the performance in a network virtualization prototype that considers these overall concepts.

Keywords: Network Virtualization, Infrastructure Provider, Virtual Network Operator, Virtual Network Provider, Virtual Resources, FP7 4WARD Project.

1 Introduction

Network virtualization (Fig. 1) is the concept of using physical resources to create virtual resources that are formed into virtual networks. These networks can be brought up on-the-fly to serve different requirements in very short time frames. Transient networks for such purposes as conferences or emergency management can ideally utilize VNets to setup short lived networks that serve a required purpose. VNets have a number of advantageous properties such as isolation, dynamic provisioning and security.

As one of the most important enablers of the future Internet, network virtualization has received a higher level of research attention all over the world, e.g. VINI [1], GENI [2] and PLANETLAB [3] in US; AKARI [4] and AsiaFI [5] in Asia; 4WARD [6] in Europe.

The 4WARD project was formed to undertake research on the architecture of a future Internet adopting a "clean slate" research approach. This approach temporarily ignores the practical constraints of evolving from the existing TCP/IP-based network architecture in the interest of identifying a design that is appropriate to the present and expected future usage and is not forced to adapt to architectural decisions made some thirty years ago with quite different objectives and constraints. The main concepts of 4WARD include a "Generic Path" allowing inherent support of mobility, multipath connections and network coding, new addressing paradigms, referred to as "Network of Information" and network virtualization enabling coexistence of several legacy and new networking concepts on the same infrastructure. A prototype has been developed to demonstrate and evaluate the concepts developed for network virtualization in this research. This work presents the framework of this prototype and its performance.

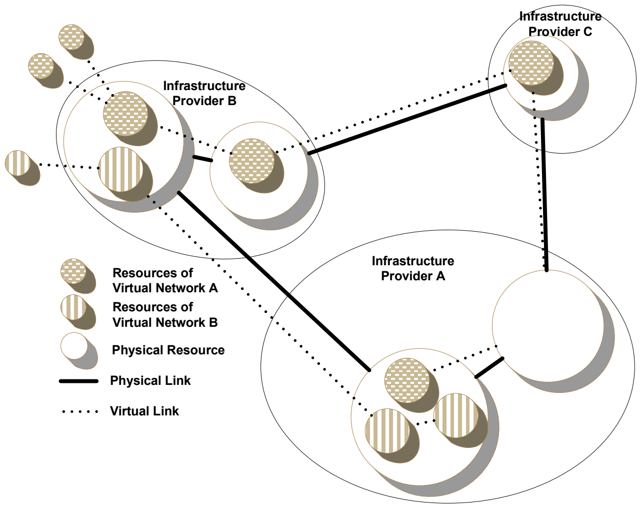
Infrastructure Provider C
Infrastructure Provider B
Infrastructure Provider A
Resources of Virtual Network A
Resources of Virtual Network B
Physical Resource
Physical Link
Virtual Link

Fig. 1. Network Virtualization: Virtualized physical resources are connected together to form virtual networks

When considering prototyping and performance evaluation on network virtualization, especially on XEN-based [13] solutions, a number of activities can be found. A number of exemplary research works that have been recently published are taken and grouped them into the specific topic areas they address (I/O performance, resource scheduling, and virtual routing). [7] compares the streaming performance of open source hypervisors like XEN and OpenVZ in a Linux environment. [8] also gives a quantitative analysis on popular hypervisors, e.g. XEN, VMware and KVM, to compare their performance in terms of application test, disk test and I/O test, where the issue of scalability (meaning the ability of hosting multiple virtual machines in one physical machine) is also discussed since isolation among virtual networks is important to guarantee the performance of each. The authors of [9] and [10] discuss the performance degradation of network interfaces due to virtualization and the large receive offload (LRO) concept is employed as an effective method to improve the I/O performance. And similarly, in order to enhance the I/O performance, [11] proposes a command-line tool in the XEN environment to do the bandwidth reservation on XEN virtual machines.

When considering the above mentioned research, it is clear that they focus on specific aspects of network virtualization, leaving out the overall operations of virtual networks. That is the key difference in the work presented in this contribution. This prototype is able to present the creation and the operation of complete virtual networks. It includes the operation, management and visualization of virtual network related activities by the different parties involved in the whole network virtualization architecture. It is able to create wired as well as wireless QoS guaranteed virtual networks on the fly, initiated by the Infrastructure Providers (InPs) based on the requirements of the Virtual Network Operators (VNet Operators) who will finally make services available for their clients.

The sections that follows focuses on provisioning of VNets utilizing end-to-end infrastructure, reflecting on the concepts of VNets, revealing the bottlenecks when realizing multiple VNets and recommendations from the experiences. Specifically, the immediately following section explains the concepts and processes involved in network virtualization briefly. Then it moves on to explaining the architecture of the software framework that was developed. This is followed by an explanation of the scenarios considered together with the test-beds on which these scenarios are tested. The subsequent section explains and provides an analysis of the results taken from the above scenarios. The last section is a concluding summary that includes a look at the future direction of this work.

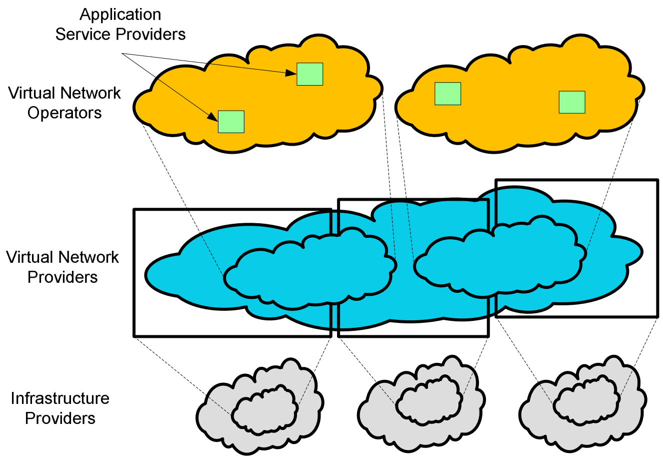
Application Service Providers
Virtual Network Operators
Virtual Network Providers
Infrastructure Providers

Fig. 2. Future Internet: Roles of the different parties associated with network virtualization

2 Network Virtualization Architecture of the Future Internet

Today, the Internet is operated by multiple Infrastructure Providers (InP). Similarly, the future Internet can be assumed to have a similar status-quo where large organizations that own and operate infrastructure enabling communications between different locations and providing connectivity for end users [12]. These InPs create virtual resources from their physical infrastructure for other parties to configure and use. Unless this aspect of inter-domain QoS guarantees is considered, intra-domain

QoS guarantees may seem meaningless due to bottlenecks of different InPs. Therefore in the future Internet, an intermediary called a VNet Provider will negotiate with different InPs and obtain virtual network slices that are in turn combined and/or sub-sliced to be made available for VNet Operators based on their requirements.

Fig. 2 shows the relationships of the different roles of the parties in the architecture of the Future Internet. Once the VNet is made available by the VNet Provider in the requested topology, the VNet Operator can setup and offer the network for its customers (Application Service Providers and end users). Setting up by the VNet Operator includes activities such as installing the different protocols in the VNet and configuring it to suit the requirements of its customers. The VNet Operator has full control over the VNet and the InP may not necessarily know what is in the VNet and how it is operated.

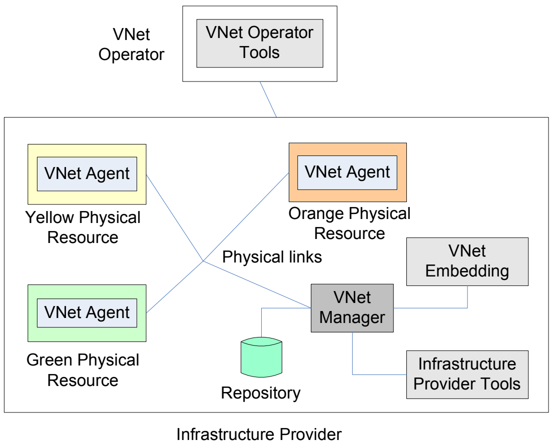

Fig. 3. VNet Management: Component connections of a VNet Management environment

3 VNet Management Prototype

The VNet Management (VNM) prototype is built to evaluate and demonstrate the VNet concepts and processes described in the previous section. This is built to operate on IP based networks and utilizes XEN [13] as the primary virtualization technology. The typical architecture of such a VNM environment consists of agents that reside in different physical resources and control the resource to create, remove or modify virtual resources. These agents are controlled by managers located centrally or distributed, which accept commands from an InP to manage the physical resources. The managers hold repositories that are continuously updated based on the current status of the VNM environment. The interactions of agents and managers are similar to the operations of SNMP, but differ from SNMP due to its independence from specific networking technologies (such as IP).

Fig. 3 shows an example of a VNet management environment consisting of 3 physical resources (colored boxes). Each physical resource has a VNet Agent running in it to manage the virtual resources instantiated within the physical resource. The

VNet Manager controls all the VNet Agents and the repository. Using a set of tools, the InP initiates management requests based on the requests of the VNet Operators.

3.1 VNet Manager

VNet Manager (Fig. 4, left) coordinates all the activities of the whole VNet environment. It is a multi-threaded daemon that can reside in the hardware of the InP and communicate with the other components of the environment. It has the following functional activities:

- Handling of requests and information communicated with VNet Agents and InP management tools
- Handling of synchronized updates to the VNet Repository which serves as the storage for the configurations of the VNets

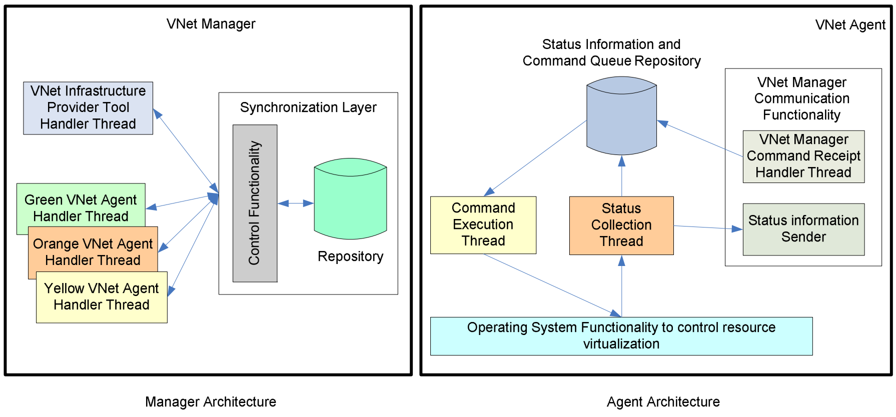
VNet Manager
VNet Infrastructure Provider Tool Handler Thread
Synchronization Layer
Control Functionality
Repository
Green VNet Agent Handler Thread
Orange VNet Agent Handler Thread
Yellow VNet Agent Handler Thread
Manager Architecture
VNet Agent
Status Information and Command Queue Repository
VNet Manager Communication Functionality
VNet Manager Command Receipt Handler Thread
Command Execution Thread
Status Collection Thread
Status information Sender
Operating System Functionality to control resource virtualization
Agent Architecture

Fig. 4. VNet Component Architecture: The modular architecture of a Manager and an Agent

All the InP management tools use the same interface in this prototype to send and receive data to the VNet Manager. Each tool instance started will result in a separate process being created in the VNet Manager to handle the requests and return information. The communication between the user tools and the VNet Manager is based on TCP sockets in this prototype, to maintain reliable communications.

The connections of VNet Agents also have a similar concept where each VNet Agent instance will result in a process being created to handle the sending and receiving of information to the VNet Agent. The interfaces to the agents have the same format for every agent and uses TCP sockets for communications.

The control functionality which is protected (made thread safe) for synchronized access provides the following functionality to the different processes created:

- Handle command actions received from the InP tool processes
- Command VNet Agents to perform activities
- Handle information received by the VNet Agents through the processes
- Send information to the InP tools
- Update the repository

3.2 VNet Agent

VNet Agents manage the actual virtualization of physical resources. A VNet Agent is executed on each physical resource. Each VNet Agent is specific in its functionality to the resource under its control. Fig. 4 (right) shows the generic architecture of a VNet Agent.

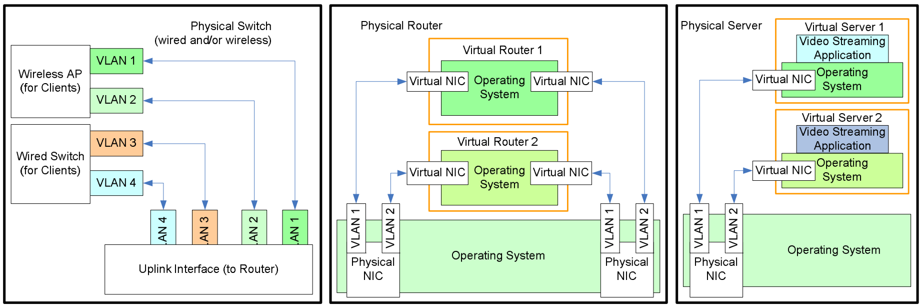

Fig. 5. Physical Device Configurations: Virtual resources and their connections in physical resources to form virtual networks

Every VNet Agent has an interface with the VNet Manager to get commands to create, remove, modify, bring up or shut down virtual resources associated with a given Agent. There are a number of processes in an instance of an Agent which perform the following tasks:

- Queue incoming virtual resource management instructions
- Execute these instructions in a FIFO manner
- Retrieve current status information
- Notify current status information

The operating system functionality component in the Agent architecture is the component that holds functionality that is unique to each of the resources. That means, an Agent for a Wireless Access Point (WAP) of a particular hardware manufacturer differs from the Agent for a WAP of another hardware manufacturer in its operating system functionality component. There are a number of different VNet Agent types supported by the VNet environment.

The VNet Agent for Servers manages the virtual servers that can be created, removed, brought up, etc. on a physical server with a single network attachment. Fig. 5 (right) shows a VNet Agent for a physical server that has created two virtual images of servers.

The virtualization at the link level is achieved through the use of Virtual Local Area Networks (VLANs) where each VNet is carried with a separate VLAN ID.

The VNet Agent for Routers controls physical routers to manage the virtual environment. A physical router consists of multiple network interfaces to perform the routing. These interfaces are bridged to the virtual router instances to perform the routing. VNets are realized using VLANs. Fig. 5 (middle) shows a router that has two virtual routers instantiated. VNet Agents for Switches/Wireless Access Points use

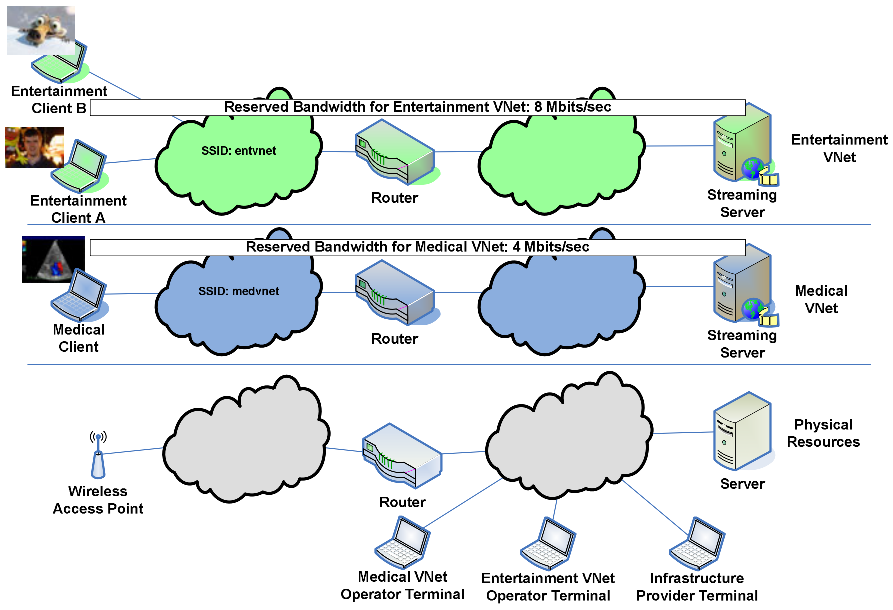

Fig. 6. Wireless Scenario: Test-bed setup for the scenario with 2 simultaneous VNets

VLAN and the capability of multiple virtual SSIDs to distribute traffic to different virtual network. Fig. 5 (left) shows a physical Wireless Access Point handling network traffic related to four VNets.

4 Scenarios and Test-Beds

The evaluation of the VNM prototype is done using 2 test-beds that focus on 2 different evaluation scenarios. The hardware used in these test-beds is installed with the VNM prototype.

4.1 Wireless Scenario

The first scenario, referred to as the Wireless Scenario (Fig. 6) considers an InP who offers hardware for VNet Operators to create virtual networks. Two such operators use this service and request the InP to create 2 specific virtual networks with differing QoS characteristics. In this case the role of the VNet Provider is left out due to the use of only one InP. The first of these networks is created for an operator who serves a medical organization's requirements by providing a secure network with a guaranteed bandwidth to carry video and vital signs data of remote surgical operations.

The second virtual network operator maintains an entertainment network that hosts an Application Service Provider providing movie streaming services to its customers. The customers of both of these VNet Operators connect over wireless networks (WLAN).

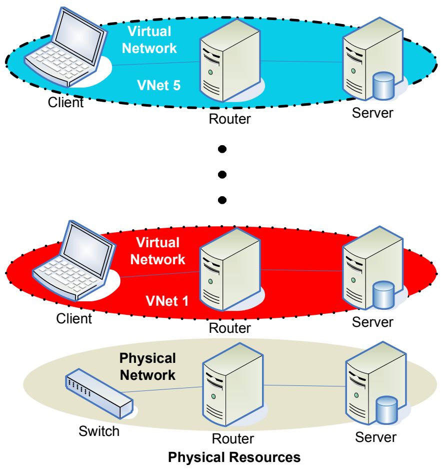

Fig. 7. Wired Scenario: Test-bed setup for the scenario with multiple simultaneous VNets

The Medical VNet Operator requests a VNet with a 4 Mbit/s guarantee while the Entertainment VNet Operator requests a VNet with an 8 Mbit/s guarantee. The management terminals of the VNet Operators which are connected to the InP's network requests InP to create the required VNets in the configuration they require. In this case, both operators request for a network with a server, a router and a wireless access point. Each of the clients that connect request for 5 Mbit/s UDP streams from the respective server of the VNet, which hosts the service. In this scenario, the 2 VNets are created sequentially, where the Medical VNet is started before the Entertainment VNet, with a time difference of about 100 seconds. The 5 Mbit/s UDP streams, one in the Medical VNet (MedVNet) and 2 in the Entertainment VNet (EntVNet 1 and EntVNet 2) are started in around 150 second intervals.

The main purpose of this scenario is to show the overall operation of the Network Virtualization architecture with the different activities involved in establishing and using the VNets.

4.2 Wired Scenario

The second scenario called the Wired Scenario (Fig. 7) setup on the second test-bed is similar to the first scenario but is intended to evaluate the performance of multiple VNets with an emphasis on increasing the VNets until the performance degrades. This test-bed is equipped with commercially available off-the-shelf hardware. The equipment consists of Quad-core hardware and Giga-bit links connecting them. To leave out the effects of wireless networks, a completely wired setup is considered in this scenario. In this scenario, the VNets are created in 100 second intervals and each of the clients associated with a VNet is attempting to download 150 Mbit/s stream from its server.

5 Performance Results and Analysis

Fig. 8 shows an example of the throughput performance of the 3 VNet streams on each of the clients. It also shows how the physical hardware performs. Since the Medical VNet has a bandwidth limitation of 4 Mbit/sec over the Ethernet link, the 5 Mbit/s input stream is reduced to around 4 Mbit/s. In addition, it will not reach the exact 4 Mbit/sec level due to packet losses. Since a lost packet means a loss of a number of bytes, the stream will not be able to reach the 4 Mbit/s level. The packet losses graph (Fig. 9, left) shows how the packets for each VNet stream are lost. What could be noticed is that when the second stream of the Entertainment VNet (EntVNet 2) is started, both of the two other streams became very unstable. This is due to the insufficient bandwidth available for both streams. But, the performance of the MedVNet continues proceed unaffected due to the given bandwidth guarantee.

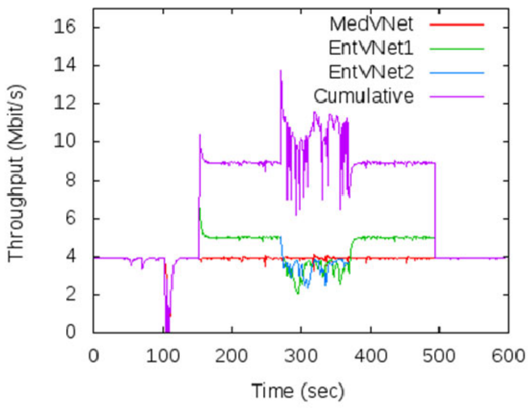

Fig. 8. Wireless Scenario: Throughput performance of VNets with UDP streams

The drastic throughput drop that is visible around 100 seconds in Fig. 8 is due to the creation of the second Entertainment VNet. This is specifically due to the creation of the second WLAN SSID which makes the WAP halt the data flows on other active SSIDs until the new SSID is configured.

This experiment was done in an environment where other WLAN APs not related to the experiment were in operation. Hence, all transmissions are influenced by the activities of the other APs. The constant fluctuations that are visible in graphs are a sign of that.

Fig. 9 (right) shows how bandwidth limitations can influence the jitter of the streams. As can be seen, the jitter is higher for the MedVNet while the EntVNet streams have lower jitter values. This is due to the operation of the queues of traffic shaping. It was seen that when the bandwidth limit (e.g. 4 Mbit/s in the MedVNet) is close to the utilization of the bandwidth (e.g. 5 Mbit/s in MedVNet), a higher jitter is shown compared to the opposite.

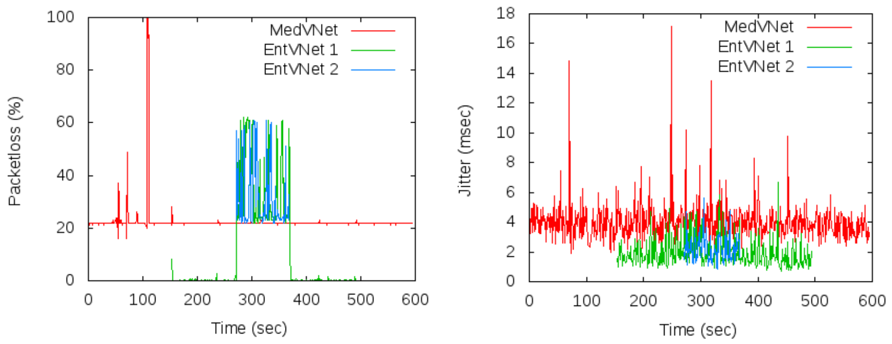

Fig. 9. Wireless Scenario: Packet loss and Jitter with UDP streams

The second set of experiments (based on the Wired Scenario) puts the emphasis on stress testing the test-bed, which is made out of commercially available hardware. The stress test mainly looks at how much of VNets can be created in this hardware and looks at what the performance statistics show.

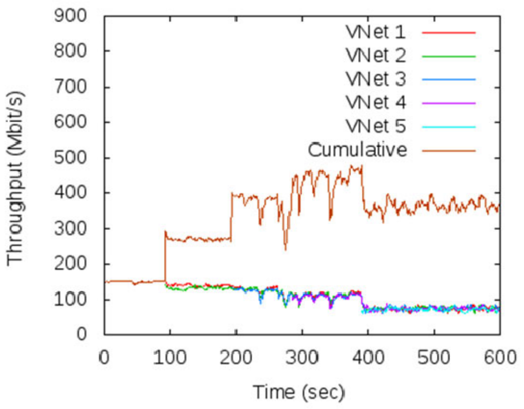

Fig. 10. Wired Scenario: Throughput performance of 150 Mbit/sec UDP streams

Throughput performance of this setup (Fig. 10) shows the effects of creating multiple VNets using the same infrastructure. Up to 2 VNets can be operated without any degradation of throughput but when the 3rd VNet is started, the throughput of all the active VNets starts fluctuating. This is in spite of using Giga-bit Ethernet links between them. When the 5th VNet is created, the throughput of all the active VNets simply start to decrease and experiences close to 50% packet losses (Fig. 11, left). The jitter values of these streams (Fig. 11, right) show the same tendency where fluctuations become more visible starting from the 3rd VNet.

A look at the resource utilization of all the hardware used to create the VNets show that CPU utilization is not an issue with the start of the 3rd VNet and the corresponding 150 Mbit/sec stream. Since other resources such as disk space and link capacities are

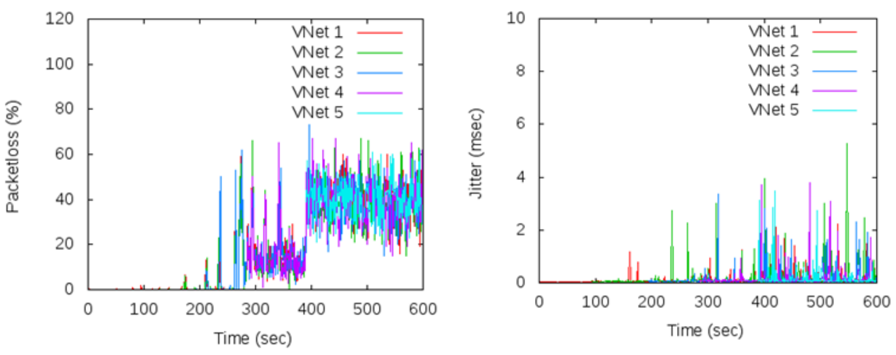

Fig. 11. Wired Scenario: Packet loss and Jitter with 150 Mbit/s UDP streams

sufficiently available, the degraded performance points to inefficiencies in the way the Hypervisor schedules the data in different queues related to communications.

6 Conclusions

Network virtualization is considered as a key enabler of the Internet of the future. A number of projects are developing concepts and processes related to network virtualization. A noticeable drawback of those works is that they attempt a piecemeal evaluation of network virtualization and thereby leaves out certain aspects when evaluating the others. The prototype that was built in this work attempts at focusing on all the aspects starting from the roles of the parties involved to the technology used. The experiences gained through this prototype, points to the following conclusions.

- Network virtualization appears to be feasible even in an overall setting where all aspects (from roles of the parties to virtualization technologies) are considered
- Use of commodity hardware in the scenarios point to the feasibility of using them for creating multiple virtual networks (does not necessarily require high end purpose-built hardware)
- Virtualization environments used (e.g. XEN in this prototype) needs to be enhanced further to improve on scheduling related to networking traffic
- Sufficient resource margins must be considered from a deployment perspective to ensure optimum performance

The work done with this prototype will be extended further to consider scenarios that increase the number of physical resources used.

Acknowledgments

The work presented here was done during the FP7 4WARD project funded by the European Commission. The authors wish to thank all the partners of the FP7 4WARD project, especially the partners of the network virtualization work package who have contributed in terms of discussions and ideas to this work. Further, the authors wish to thank Nikola Zahariev for his assistance during this work.

References

1. Bavier, A., Feamster, N., Huang, M., Peterson, L., Rexford, J.: VINI Veritas: Realistic and Controlled Network Experimentation. In: Proc. ACM SIGCOMM 2006 (September 2006)
2. GENI: Global Environment for Network Innovations, `http://www.geni.net`
3. Bavier, A., Bowman, M., Culler, D., Chun, B., Karlin, S., Muir, S., Peterson, L., Roscoe, T., Spalink, T., Wawrzoniak, M.: Operating System Support for Planetary-Scale Network Services (March 2004)
4. Architecture Conceptual Design for New Generation Network (akari-project.nict.go.jp)
5. Asia Future Internet (AsiaFI), `http://www.asiafi.net`
6. Niebert, N., et al.: The Way 4WARD to the Creation of a Future Internet. In: PIMRC 2008, Cannes, France (September 2008)
7. Sukaridhoto, S., Funabiki, N., Nakanishi, T., Pramadihanto, D.: A comparative study of open source softwares for virtualization with streaming server applications. In: ISCE 2009, Kyoto, Japan (May 2009)
8. Xu, X., Zhou, F., Wan, J., Jiang, Y.: Quantifying Performance Properties of Virtual Machine. In: ISISE 2008, Shanghai, China (December 2008)
9. Fumio, N., Hitoshi, O.: Optimizations of Large Receive Offload in Xen. In: NCA 2009, Cambridge, MA (August 2009)
10. Hitoshi, O., Fumio, N.: Performance Analysis of Large Receive Offload in a Xen Virtualized System. In: ICCET 2008, Singapore (February 2008)
11. Zhang, J., Li, X., Guan, H.: The Optimization of Xen Network Virtualization. In: CSSE 2008, Wuhan, China (December 2008)
12. 4WARD, `http://www.4ward-project.eu`
13. Williams, D.E.: Virtualization with Xen: Including XenEnterprise, XenServer, and XenExpress, Syngress (2007)