HETEROGENEOUS NETWORKING TESTBEDS INTEGRATION AND WIRELESS NETWORK VIRTUALIZATION

BY RAJESH MAHINDRA

A thesis submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Master of Science

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Professor D. Raychaudhuri

and approved by

_____

_____

_____

New Brunswick, New Jersey

January, 2008

ABSTRACT OF THE THESIS

Heterogeneous Networking Testbeds Integration and Wireless Network Virtualization

by Rajesh Mahindra

Thesis Director: Professor D. Raychaudhuri

Networking research has grown immensely over the past few years. This has urged the need for a heterogeneous networking research infrastructure, to experiment with the interaction and integration of different types of networks. This requirement led to the Global Environment for Network Innovations (GENI) effort, supported by NSF, which aims at creating a global infrastructure for conducting networking experiments across diverse substrates such as wired, wireless, sensor and cellular networks. In this work, we discuss challenges involved in federating two diverse testbeds - PlanetLab and ORBIT and present a model for building a united infrastructure for the models. PlanetLab is a global research wired network that supports the development of new network services. ORBIT is a laboratory-based wireless network emulator for 802.11 testing. An integrated wired-wireless testbed will increase the scalability of experimentation. Proof-of-concept experiments are also presented reinforcing the usefulness of the model in terms of facilitating experiments over the integrated infrastructure.

Such an integrated infrastructure poses a requirement of support for wireless network virtualization - supporting multiple concurrent wireless experiments. Unlike wired networks, wireless networks present unique challenges making the task of wireless virtualization a difficult problem. The critical problem of simultaneous experimentation

in networks involving the wireless medium are identified and approaches towards it are discussed. We evaluate and compare two approaches towards wireless virtualization - SDMA (Space Division Channel Multiplexing) and VAP (Virtual AP Channel Multiplexing) suitable for supporting long running experiments. In this study conducted on ORBIT we quantify the difference in performance and interference when using wireless virtualization and suggest measures to mitigate the same. The feasibilty study will serve as the first step towards ORBIT virtualization.

Acknowledgements

I would like to express my deep and sincere gratitude to my advisor, Dr. Dipankar Raychaudhuri, Director, WINLAB. It was an unforgettable experience working on this project under his guidance. His understanding, encouraging and personal guidance have provided a good basis for the present thesis.

I am deeply grateful to my supervisor, Ivan Seskar, Associate Director, WINLAB, for his detailed and constructive comments, and for his important support throughout this work. Compiling this thesis would not have been possible without his presence.

I wish to express my warm and sincere thanks to Dr. George Hadjichristofi, Gautam Bhanage and Shruti Singhal. Their ideals and concepts have had a remarkable influence on my entire research career. Their extensive discussions around my work and interesting explorations in operations have been very helpful for this study.

My special thanks to Sachin Ganu, Pandurang Kamat and Kishore Ramachandran who directed me in my early days in Winlab. Their kind support and guidance have been of great value.

I enjoyed the company and support of many fellow WINLAB students including Sumathi Gopal, Haris Kremo , Sankar Subramanium, Hariharasudhan Viswanathan and Tejaswy Hari.

Last but not the least, my special gratitude to Madhur Mirji for her never ending encouragement and support.

Dedication

This work is dedicated to my parents and my cute little sis *Priyanka Mahindra* for their inspiration.

Table of Contents

List of Figures

List of Abbreviations

AP	Access Point
BSS	Basic Service Set
BSSID	Basic Service Set Identification
CSMA-CA	Carrier Sense Multiple Access - Collision Avoidance
CPU	Central Processing Unit
DCF	Distributed Coordination Function
DIFS	DCF Interframe Space
DSSS	Direct Sequence Spread Spectrum
ESS	Extended Service Set
ESSID	Extended Service Set Identification
FDMA	Frequency Division Multiple Access
FH	Frequency Hopping
GENI	Global Environment for Network Innovations
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
LAN	Local Area Network
MAC	Medium Access Control
NSF	National Science Foundation
OFDM	Orthogonal Frequency Division Multiplexing
OML	ORBIT Measurements Library
ORBIT	Open Access Research Testbed for Next Generation Wireless Networks
OS	Operating System
OTG	ORBIT Traffic Generator
OTR	ORBIT Traffic Receiver
PHY	Physical Layer
RTP	Real Time Protocol
SDMA	Space Division Multiple Access
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
UDP	User Datagram Protocol
VAP	Virtual Access Point
VINI	Virtual Network Infrastructure
VLC	Video Lan Client
VMAC	Virtual Medium Access Control
WLAN	Wireless Local Area Networks

Chapter 1

Introduction

1.1 Overview of the Problem

To support realistic and large scale experimentation on integrated testbeds, the testbed framework should have a flexible design that will enable a variety of network architectures, services and applications. In this dimension, we feel that there is lot to learn from the currently deployed wireless testbeds like ORBIT [1], Emulab [2] and Deter [3]. The ORBIT wireless testbed at WINLAB, Rutgers University was used as a platform for evaluating solutions for wireless virtualization and challenges faced in wired-wireless network integration with integration of ORBIT with PlanetLab [4]. The integration efforts include extension of the ORBIT control framework to PlanetLab nodes to allow ORBIT users to include one or more PlanetLab nodes in their experiments. In addition we have conducted example end-to-end wired + wireless experiments as a part of the proof-of-concept prototyping. The other part of our work focuses on possible solutions to virtualizing the ORBIT wireless grid to allow concurrent experiments to take place without interference. This would facilitate long term experiments running concurrently on ORBIT. Use of these long running ORBIT slices would resolve the issue of differences in the experiment models of PlanetLab and ORBIT. PlanetLab allows for experiment duration of the order of months while currently, the ORBIT is a multi-user wireless experiment testbed that allows sequential short term access to the radio grid resources. Virtualization schemes will allow for more efficient use of the testbed resources. A number of different forms of wireless virtualizations have been proposed like TDMA, FDMA, SDMA and a combination of them. In our study we empirically evaluate the space division multiplexing (SDMA) and the virtual access point (VAP) based approaches to wireless virtualization and conclude the suitableness of each of

these approaches. We show that though packet capture results in an improved performance for the SDMA approach, both approaches need a higher layer policy manager to ensure isolation between the individual experiments.

Our study does not aim to provide a comprehensive virtualization solution across all wireless devices and drivers but serves as a reference to show the trends in performance that may be observed with the use of the two aforementioned approaches. Our study would lay the foundation for some of the key design issues and deployment strategies for wireless virtualization on large-scale network testbeds like GENI [5].

1.2 Motivation

Networking research has expanded into new frontiers with the need for heterogeneous networking research infrastructure. The researchers have felt a need to experiment with integration of different types of networks and to evaluate the performance of various networking protocols in realistic networks. The NSF [6] led GENI project [5] aims at setting up a shared experiment facility for testing network designs and protocols that may be part of the next-generation internet architecture. The GENI infrastructure will include a large scale wired network programmable with virtualized network elements like servers, switches, routers etc. to allow simultaneous experiments. The project also plans to provide several wireless network deployments to support experiments including sensor networks, radio networks, 802.11 networks etc. The GENI working group has identified two short term goals as key technical issue for the design and deployment of the GENI testbed [7]:

1. The Control framework for the wired and wireless testbeds has to be integrated providing a single experiment framework and methodology for researchers.

2. Virtualization of Wireless Networks to support multiple experiments to run concurrently on the same radio devices.

The GENI working committee had realized that parallel work on a proof of concept prototyping of Wireless Virtualization and Wired-Wireless Testbed Integration is important. This would provide guidance for the design of GENI by providing key design

issues and practical challenges faced in the above goals. The design of the PlanetLab Testbed would be a base for the design and virtualization of the wired part of the GENI testbed. However several extensions to would required to the present PlanetLab infrastructure to support full range of experiments. Some of the key extensions would be :

- Device heterogeneity: To include wireless network components like wireless routers, ad-hoc radios, sensors to the wired networks. Short Term Experiments: To add support for running short term experiments as opposed to long term experiments. End-User
- Requirements: Experienced programmers needing little experiment support vs protocol analysts needing high level tools to help them evaluate their protocols and applications. PlanetLab Test-bed's integration with the large scale wireless testbed ORBIT would be an ideal project to provide insight into the integration of heterogeneous networks and necessary challenges in extending the control and management protocols for an efficient experimental system.

The second part of the Project evaluates several approaches to wireless virtualization that have been proposed on ORBIT, a large scale 802.11 wireless facility. Hence the combined goal of this project is to demonstrate working solutions on ORBIT to define the hardware and software platforms for the GENI network.

1.3 Objectives

This thesis covers the integration framework of PlanetLab and ORBIT that has been developed and demonstrates integrated experiments that use Frequency Division Multiplexing (FDMA) and Virtual MAC (VMAC) as forms of wireless virtualization. The proof-of-concept prototyping presented in this thesis would be extremely valuable in building a practical understanding of integration issues and providing guidance to the design of GENI. Results from the prototyping are expected to feed into ongoing system engineering work aimed at specifying key technology components that will constitute GENI. Contributions of our work are:

- Extension of ORBIT framework functionality to the PlanetLab nodes providing an integrated framework to facilitate joint wired-wireless experiments.
- Address the problem of supporting multiple concurrent experiments over these substrates and provide proof of concept experiments conducted using the framework.
- Evaluate and compare two approaches towards virtualizing the wireless network -Space Division Multiplexing (SDMA) and Virtual AP (VAP) schemes suitable for running long-running concurrent experiments.
- Quantify the deviation in performance and discuss effect of interference between the experiments when using wireless virtualization.
- Determine the strengths, drawbacks and suitability of each of these approaches.
- Discuss and implement the policy manager for ensuring successful experiment isolation

1.4 Thesis Organization

Rest of the thesis is organized as follows. Chapter 2 presents a detailed picture on the efforts for PlanetLab and ORBIT testbed integration. In Chapter 3, we give an overview of Wireless Virtualization and present different schemes for wireless virtualization. Chapter 4 explains two virtualization schemes in detail namely Virtual AP and Space Division Multiplexing(SDMA). Chapter 5 contains the results section. In Chapter 6, we summarize the main contributions and give possible directions for future work.

Chapter 2

Integration of PlanetLab and ORBIT Testbeds

The integration of PlanetLab and ORBIT would help in better understanding how virtualized slices can be extended to accommodate heterogeneous wireless technologies and serve as a guideline for the GENI design. The design and experiment model of PlanetLab and Orbit are suited to different experimental and research needs. In this section, we walk through the fundamental differences in the design of these two testbeds and later discuss the design of the integration model.

2.1 ORBIT Testbed

ORBIT Testbed is a Laboratory based Wireless network emulator involving 400 802.11 nodes wireless nodes. The nodes are arranged in a 20 by 20 square grid with 1m separation between each pair of nodes. It is designed to achieve reproducibility of experimentation and support for evaluation of application and protocols in real world settings. The nodes in the grid can be dynamically interconnected into the specified topology with reproducible indoor wireless channel models. The current reservation schemes allow only one user to carry on controlled experiments on the ORBIT grid. Hence several virtualization schemes have to be investigated and analyzed to allow several users to share the wireless resources simultaneously. In addition to this a central resource manager has to be designed to provide automated resource sharing among the different users.

Each experimentation node in ORBIT is powered by a 1 GHz VIA C3 processor, 512 MB RAM, a 20 GB local hard disk, two wireless mini PCI 802.11 a/b/g interfaces and two Ethernet ports. The Ethernet ports facilitate control signaling, remote access and maintenance without affecting the wireless traffic that is part of the experiment.

The two wired networks are the Data and Control Networks. The nodes are interconnected using 1 Gbps Switched Ethernet networks. The Control Network carries all the commands to the nodes before and during the experiment. The traffic in the Control Network also includes the measurements from each experiment node during the execution of the experiment. This facilitates collection and processing of real time measurements in an experiment. This framework is called OML (ORBIT Measurement Library) [8] which is responsible for collection, processing and storage of measurements in ORBIT. The Data Network on the other end is mostly a part of the experiment that emulates wired connectivity between the nodes.

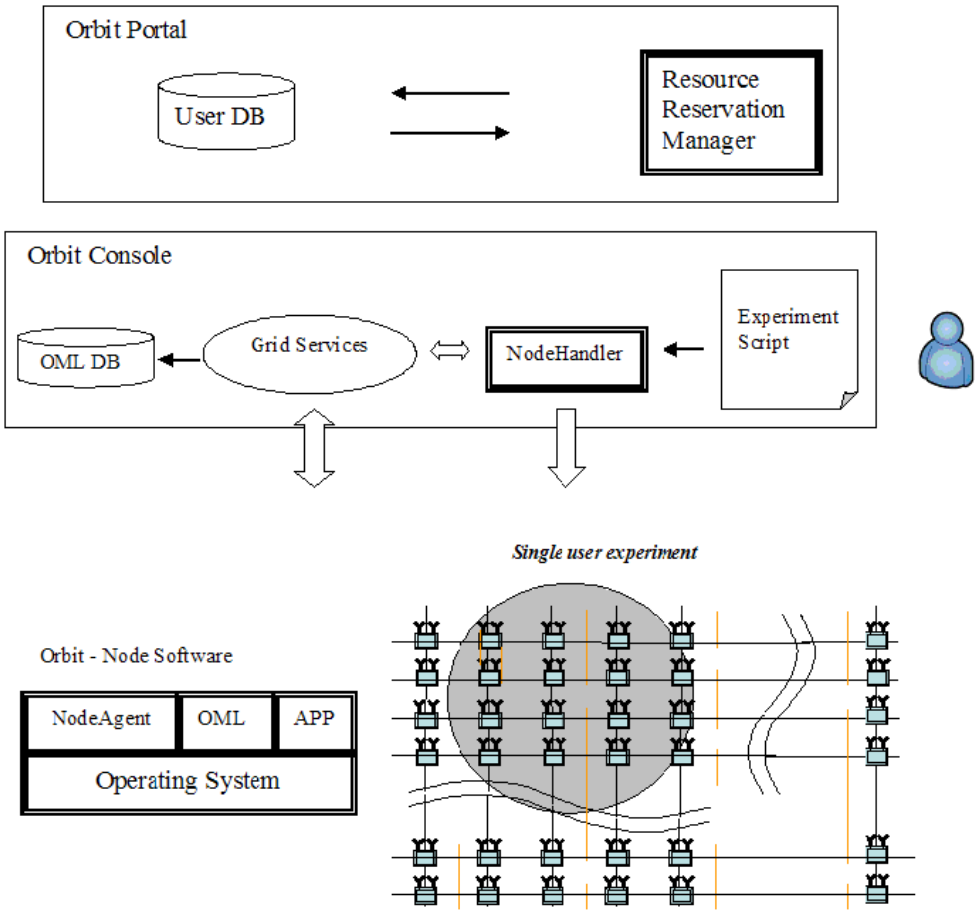


Figure 2.1: Current ORBIT testbed architecture

The ORBIT architecture is represented by Figure 2.1 depicting the major components of the testbed. The ORBIT was designed for conducting sequential short term experiments on the radio grid. Hence this requires the user to reserve the entire radio grid for sufficient time. Thus it becomes essential to accommodate as many users as possible and the design of the software architecture of ORBIT is certainly motivated by this criterion. A typical experiment compromises of the following steps:

- Selecting the nodes and the role (AP, client, sender, receiver, relay etc) that they play in the experiment.
- Configuration of the wireless and/or wired interfaces of the nodes (infrastructure/ad-hoc mode, physical rate, transmission power etc.)
- Deploying standard or custom applications on the nodes.
- Collection of results in a statistical or graphical form.

The user would be given access to all the nodes in the grid, deploy his own custom built image on the nodes and execute his experiment using the Orbit control framework namely NodeHandler. The user is responsible for feeding an experiment script to the NodeHandler. The experiment script is a simple ruby script that defines the nodes, their topology, the role that they play and applications to be deployed on them. The NodeHandler parses the experiment script and sends appropriate commands to the NodeAgents running on the Orbit Nodes. The NodeAgent runs on every Orbit Node and receives commands from NodeHandler. It then executes the requests on the node. It may be a command to the Operating System or to a user level application. For example the wireless interfaces are configured using the NodeHandler-Agent mechanism. The NodeAgent is capable of generating IOCLT commands to the wireless driver to set the PHY rate of the card, setup the essid, transmission power etc. The NodeHandler makes use of the Grid-Services that are independent sub-systems developed for specific tasks.

- The CMC service provides the service for powering the Orbit Nodes and provides feedback in terms of the status of the nodes.

- The PXE service loads a initial PXE image on the Orbit Nodes before the actual image is flashed.
- Frisbee service helps imaging the nodes with the user specified custom image in a short time prior to the start of the experiment.
- OML(Orbit Measurement Library) to set up the databases for collection of experiment results.
- Noise Generators for controlled interference and noise injection to get the desired SNR for the experiment.
- Real Time Visualization of the experiment results.

2.2 PlanetLab Testbed

The PlanetLab Testbed consists of globally distributed set of nodes connected to each other via the internet. PlanetLab users run either short term or long term experiments. These models are supported by distributed virtualization of the PlanetLab nodes [9] - the concept of slicing the global resources. A slice is a set of allocated resources distributed across PlanetLab nodes. Multiple slices run concurrently on the PlanetLab nodes independent of each other. Each PlanetLab node runs Linux V-servers that provide namespace and performance isolation among the various slices on a node - defined as a sliver. A sliver is the set of allocated resources on a single PlanetLab node. Network virtualization is provided through VNET [10].

The first step for every PlanetLab user is to create a slice and add the desired nodes to it. Experiments on PlanetLab are best effort and there is no guarantee on the network resources allocated to the slices. PlanetLab consortium (PLC) manages the creation, authentication and release of the slices. Upon the creation of the slice, the PlanetLab nodes are populated with the minimal Fedora Core Linux installation. The PlanetLab users are given restricted root permissions to the underlying software and hardware devices.

Despite providing a globally distributed substrate for conducting experiments in

a realistic network, the PlanetLab does not offer any support for choreographing an experiment and controlling the nodes using automated scripts. Popular tools like ssh, pssh, scp and pscp [11] are used to gain access to the nodes, to control them and to populate the nodes with the required applications and software. In addition, the results and measurements of the experiment have to be manually collected. Despite all these, there are third party softwares available for controlling and monitoring experiments in PlanetLab.

2.3 Integration Model: ORBIT driven Integrated Experimentation

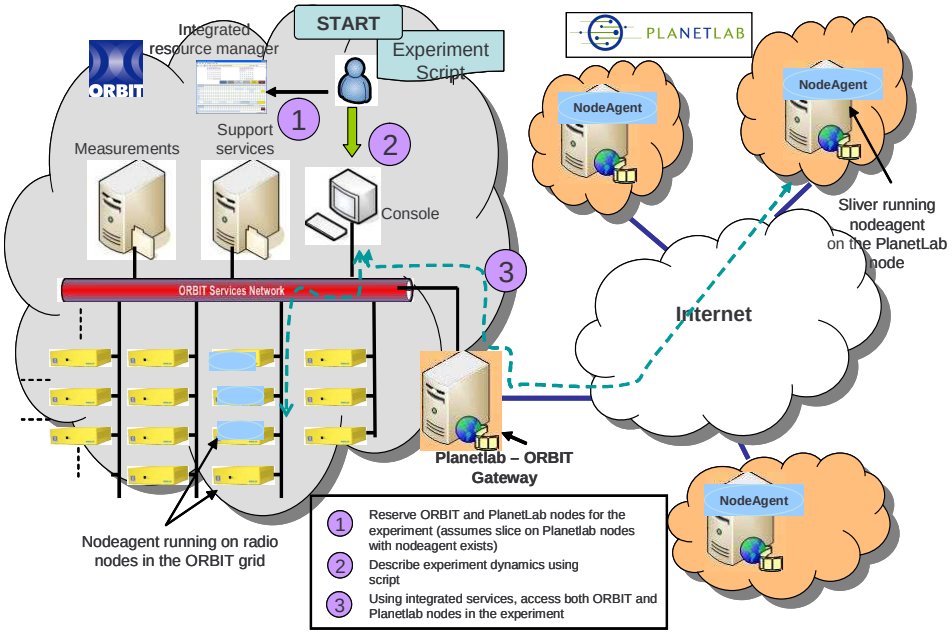


Figure 2.2: Proposed ODIE model to integrate PlanetLab into ORBIT testbed.

This model is intended to serve the ORBIT wireless testbed users who want to augment their experiment by the addition of wired PlanetLab nodes without drastic changes in their code or experiment script. In this model, ORBIT users have the provision of adding a long running "ORBIT Slice" in PlanetLab nodes in their experiment. Figure 2.2 shows the conceptual diagram for the ODIE model.

This model has been implemented by extending the ORBIT nodeAgent functionality to run on PlanetLab node "ORBIT Slivers". In addition to this, the ORBIT nodeHandler was modified to support a unified experiment definition, download and execution. This nodeHandler running in the ORBIT framework communicates with the modified nodeAgents running on the PlanetLab slivers. In this section, we describe the main directions/features that were added to support the unified experiments with PlanetLab nodes in addition to ORBIT nodes.

2.3.1 Topology Selection

In the PlanetLab testbed, users do not have the privilege of defining custom topologies for experimentation. Based on this, PlanetLab nodes for an ORBIT experiment are chosen in one of the following ways:

- *Manual Addition:* experimenters choose their PlanetLab nodes individually. The list of PlanetLab nodes in the "ORBIT Slice" is provided to every user. User is also given the option to add more PlanetLab nodes to the slice. This approach suits users that don't have any bandwidth, delay and other network parameter constraints in their experiments.

- *Metric based Addition:* experimenters prescribe the link specific characteristics desired for their experiments. The users can include those PlanetLab nodes that meet certain criterions, like link bandwidth and delay, using popular third party tools available for PlanetLab users. HP Labs has deployed S3(Scalable Sensing Service) [12] on all PlanetLab nodes. S3 currently provides the following network metrics between all pairs of Planet-Lab nodes:

    1. End-to-end latency
    2. Bottleneck bandwidth capacity
    3. End-to-end available bandwidth
    4. Lossrate

2.3.2 Extended Addressing Scheme

An extended addressing scheme to include PlanetLab nodes allows to address PlanetLab nodes as part of the ORBIT framework and have the local DNS map requests for PlanetLab nodes to their respective public domain names. As an example, in our current implementation, we address the PlanetLab nodes as [21,1..20] while ORBIT nodes continue to be addressed as [1..20,1..20] based on their row and column numbers. For e.g., *node21-3.orbit-lab.org* maps to *planetlab01.cs.washington.edu*.

2.3.3 Extended Communication Layer

In order to communicate with nodes in the local network (ORBIT nodes) as well as remote PlanetLab nodes, our model extends the current communication protocol for the nodeHandler-nodeAgent Framework to allow access to geographically diverse nodes of PlanetLab.

During an ORBIT experiment set-up and execution, commands from the nodeHandler are sent to the nodeAgents running on the ORBIT nodes using reliable multicast. For PlanetLab nodes on the Internet, these commands will need to be tunneled using reliable unicast since multicast support on the routers in a path cannot be assumed. The nodeHandler has been modified to communicate with the nodeAgents on the PlanetLab nodes over unicast-TCP. This modification eliminates the need to provide reliability in the application layer. The nodeHandler performs this function of communicating with the PlanetLab nodes in each experiment that requires wired networking resources.

Sequence of nodeHandler-nodeAgent communication during an experiment is as follows: The major upgrades to the nodeHandler-nodeAgent framework are reflected in the state diagram of Figure 2.3. When an experiment is started, the NodeHandler starts the NodeAgents on the specified PlanetLab nodes using the popular tool 'pssh' and waits for them to report back. After a timeout it records all the PlanetLab nodes that have successfully reported back. The nodes that fail to report during the timeout period are replaced by other PlanetLab nodes in the ORBIT Slice. This procedure is repeated till the desired number of PlanetLab nodes have reported back. (A failure could result

from node failure, node maintenance, slice clean-up, link failure etc.). The next step for the NodeHandler is to send commands to the NodeAgents to start the necessary applications on the PlanetLab nodes. The NodeAgents then report success or error messages back to the NodeHandler indicating the status of the nodes. This feature removes the need to manually ssh each of the PlanetLab nodes in the experiment to start the applications. After setting up the PlanetLab nodes, the NodeHandler configures and sets up the ORBIT nodes. The user simply provides a unified experiment script including both PlanetLab and ORBIT nodes and the application definition that the nodeHandler parses to execute the experiment automatically.

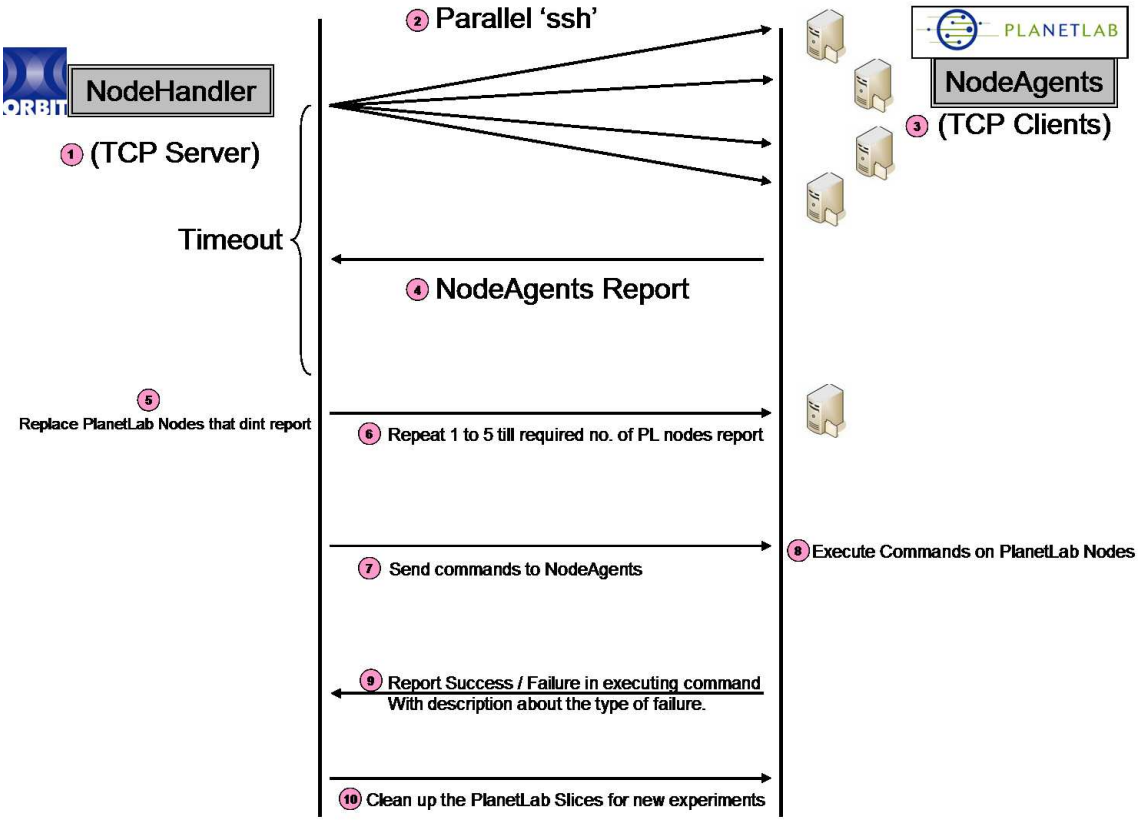


Figure 2.3: Modified nodeHandler-nodeAgent Framework for PlanetLab nodes.

2.3.4 Experiment Scripting

This section explains the unified experiment script that a user needs to supply to the ORBIT nodeHandler to conduct integrated ORBIT-PlanetLab experiments. The ODIE

based experiment script is parsed and executed by the NodeHandler to choreograph the experiments. The NodeHandler runs on the console of the ORBIT grid.

A single script for the ODIE models may be described in two sections:

- Node configuration section
- Experiment timing and execution section.

The node configuration section is responsible for setting up all the nodes being used as a part of the integrated experiment while the timing and execution section of the ODIE script describes the execution sequence of the experiment. The corresponding topology created by the experiment script is shown in Figure 2.4.

Figure 2.5 shows the section of the script that configures the nodes for the experiment. The first part of the code configures the wireless interfaces of two ORBIT nodes; one as an access point and the other as a client. The configuration part also defines the PlanetLab nodes in Washington and Japan to include in the experiment. The nodes are manually chosen by the user. However the PlanetLab nodes are addressed as extension to the ORBITs Node addressing scheme. As mentioned in the previous sub-section, all ORBIT nodes are addressed as x,y where x is the row number [1..20] and y is the column number [1..20] and the PlanetLab nodes in the ORBIT Slice are addressed as [21,1..20].

Figure 2.6 describes the timing and execution section of a typical ODIE script. The *WhenAllInstalled()* module is responsible for checking if all the ORBIT nodes have been configured as per the specification in Figure 2.5. Typically when this is verified, individual applications like running a traffic analyzer, traffic generator or any custom defined

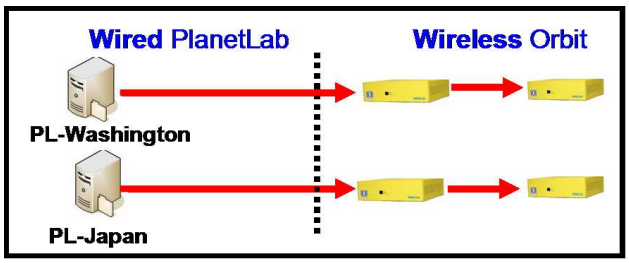


Figure 2.4: Topology setup by the Experiment Script.

```
#-------------ACCESS POINT-----------#
defNodes('AccessPoint',
[11,20]) {|node| node.prototype("test:proto:mvlcrelay",
    {'duration' => prop.duration})
    #802.11 Master Mode
    node.net.w0.mode = "master"
    node.net.w0.type='a'
    node.net.w0.channel="48"
    node.net.w0.essid = "link1"
    node.net.w0.ip="192.168.7.1"
} #---------------CLIENT--------------#
defNodes('Client', [19,2])
{|node| node.prototype("test:proto:mvlcdest",
    {'duration' => prop.duration})
    node.net.w0.mode = "managed"
    #802.11 Managed Mode
    node.net.w0.type='a'
    node.net.w0.channel="48"
    node.net.w0.essid = "link1"
    node.net.w0.ip="192.168.7.7"
}
#----------- PlanetLAB nodes----------#
defPNodes('[21,3],[21,5]')
# MAPS to planetlab01.cs.washington.edu and planet0.jaist.ac.jp #
```

Figure 2.5: Node configuration section of a sample script (ODIE model).

applications can be executed based on the timing specified in the script. Typically the application script in Figure 2.6 starts the applications on the access point and waits for 5 seconds before starting the applications on the client nodes. The remaining part of the script involves that part of the experiment execution that deals with PlanetLab nodes. The module *WhenPlReady()* waits for the nodeAgents on the PlanetLab nodes to report. There is a time-out for every PlanetLab node after which the NodeHandler ignores the nodes that fail to report and chooses other available nodes in the "ORBIT slice" on PlanetLab. Once the desired number of PlanetLab nodes have reported, the applications on the respective nodes are initiated. The feature *defPInstall()* is responsible for installing applications, software and libraries, required for the execution of the experiment on the PlanetLab nodes. This module uses the *YUM* tool [13] to install the specified software from the PlanetLab repository. *defPApplication()* triggers the

```
#--Start applications on ORBIT nodes--#
whenAllInstalled() {|node|
    nodes('AccessPoint').startApplications
    wait 5
    nodes('Client').startApplications
    wait 195  # Experiment Duration
    allNodes.stopApplications
    Experiment.done
}
WhenPLReady(){
#--Install applications on PlanetLab node--#
   defPInstall([21,3],[21,5],'APACHE'){}
#--Start applications on PlanetLAB nodes--#
   defPApplication([21,3],'VIDEO1'){}
   defPApplication([21,5],'VIDEO2'){}
   wait 195
   defPApplication([21,3],[21,5],'STOP'){}
PLexpdone() }
```

Figure 2.6: Experiment execution section of a sample script (ODIE model).

execution of applications and scripts on the PlanetLab nodes specified . Both, *defPInstall()* and *defPApplication()* report success or failure in executing the application on the PlanetLab nodes to the ORBIT nodeHandler. The *Plexpdone()* module ensures the slice is cleaned up at the end of the experiment.

Since the NodeHandler/NodeAgent framework is Ruby-based (a highly portable, scripting language) and since both PlanetLab and ORBIT run different flavors of the same OS (Linux) , the porting of nodeAgent software on PlanetLab was done without much modification.

As stated in [14], wireless and mobile networks represent an essential part of network research. In addition, there is a need for identification of the architecture for next generation networks and protocols. The integration of large scale testbeds like PlanetLab and ORBIT would certainly be the first step towards these developments. However, a related aspect of this integration is the ability to carry out multiple concurrent experiments within the integrated platform to support large-scale experimentation. While PlanetLab serves as a base model for wired virtualization, wireless virtualization solutions need to be investigated for ORBIT. Currently, the entire ORBIT testbed resources

have to be reserved and allocated to one user. With the use of a virtualized ORBIT testbed, a PlanetLab slice can be extended to include individual ORBIT nodes. In the following section, we discuss challenges to wireless virtualization followed by schemes to overcome this barrier. We also show proof-of-type prototype integrated experiments to show the usefulness of our integrated infrastructure.

Chapter 3

Wireless Network Virtualization

3.1 Introduction To Wireless Virtualization

Virtualization is defined as making multiple logical resources out of a physical resource such that each logical resource appears as a single independent entity. The goal of virtualization is to enable multiple experiments that might run concurrently or sequentially to share a common network infrastructure with minimal interference. The concept of Virtualization of networks has been extensively used in wired network testbeds like PlanetLab to provide support for multiple concurrent and long running experiments on limited physical resources of the testbed. However, as mentioned in the GENI draft [5], the virtualization of a wireless network is recognized to be a difficult problem.

On the other hand, Slicing is defined as the process of allocating coherent subset(a slice) of the physical resources to a given experiment. In contrast to virtualization, slicing does not involve sharing of the same physical resource across multiple experiments.

3.2 Challenges involved in Wireless Virtualization

The nature of the wireless medium imposes some unique and interesting challenges in the virtualization of wireless networks that are not observed in its wired counterpart. Two fundamental problems imposed by the wireless environment are listed:

- **Isolation** The Wireless Channel is a broadcast medium. There is radio interference among the various nodes sharing the same wireless spectrum. The virtualization of a wired node relies on sharing CPU, BW etc to time slice the different experiments. These resources can be upgraded to meet the requirements of the

users but the wireless testbeds share the wireless spectrum. Hence a strict isolation of wireless resources will need more wide range of solutions to partition the resources than its wired counter part.

- **Experiment Repeatability** An important aspect with performing indoor controlled experiments is to ensure the repeatability of results. If resource sharing is not done properly and their effects are not known before hand, results obtained may not be consistent.

3.3 Requirements from virtualization schemes

- **Efficient:** The virtualization scheme assigned for experiments should be efficient in terms of its implementation and performance. The implementation overhead should be minimum so that experiments are promised satisfactory performance in terms of throughput, delay, jitter and other important metrics.

- **Minimal Interference:** The virtualization schemes should be designed for maximum isolation between experiments. In most virtualization approaches, there is bound to be interference between experiments. Improper resource sharing may result in unpredictable performance across multiple experiment runs.

Most importantly, adding virtualization capabilities to a wireless network testbed like ORBIT would require design and development of additional components. Some of the major components include:

- **Slice Manager(SM)** Similar to its role in wired testbeds, the slice manager would track the experiment's states and provide user authentication.

- **Resource Manager(RM)** This module is responsible for admission control and checks for the availability of resources in the grid. It will approve users based on their request for nodes and channels. A resource manager would need to map a given experiment scenario to a resource constrained setup. Selection of a particular mapping scheme depends on the following aspects:

1. Resource constraints of the testbed: number of nodes, available orthogonal channels, ability of the experiment control mechanism to handle parallel experiments.
2. Performance degradation: Virtualization of any resource almost always results in some form of compromised performance. While mapping virtualization to scientific experiments it is necessary to know the performance deviation in experiments if any, that may be seen due to sharing of resources.

- **Virtualization Support(VS)** This component would run on the testbed nodes to support multiple experiments on the nodes. It will provide efficient partitioning of the computer resources on the radio nodes. In addition to these responsibilities, the VS is also responsible for monitoring the experiment to ensure that the experiment execution confines to the resources assigned to it by the RM. For e.g., if an experiment is allotted a frequency, the VS on the transmitting nodes of the experiment should report and take some action if the nodes transmit at different frequencies that might effect other exeperiments.

3.4 Outline of Wireless Virtualization Approaches

3.4.1 Frequency Division Multiple Access (FDMA):

In this technique, different experiments are partitioned in the frequency domain. 802.11a wireless cards provide support for 12 orthogonal frequencies. Similarly 802.11b provides 3 orthogonal channels. This limits the virtualization to a maximum of 15 concurrent experiments to run on a node assuming three experiments use 802.11b and twelve make use of 802.11a with no specific channel requirements. Another penalty is the channel switching time taken by the cards to switch between frequencies. As an example, Artheros Wireless cards used in ORBIT have a switching time of the order of 5ms. The use of multiple cards as shown in the figure below can avoid switching time but this would require virtualization at the user level. Each virtual node would correspond to a distinct wireless card, each of them configured to a different frequency channel. This approach would introduce the cost of context switching and requirement for higher

CPU speed and a larger memory for the testbed nodes.

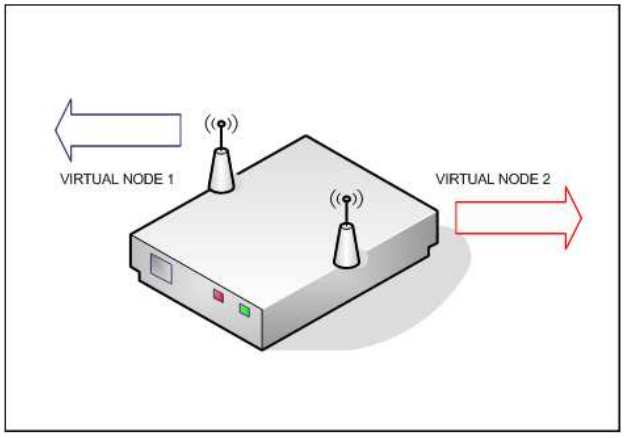


Figure 3.1: FDMA based wireless virtualization.

3.4.2 Time Division Multiple Access (TDMA):

This approach partitions the experiments along the time dimension. The virtual nodes are allotted non-overlapping time slots. This is very similar to the processor scheduling widely used for process virtualization. The more the degree of virtualization the more will be the delay faced by the individual experiments. Similar to FDMA, this approach faces the overhead of context switching across the virtual nodes.

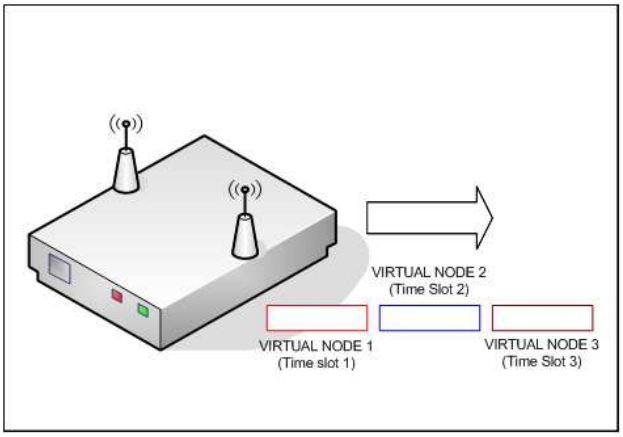


Figure 3.2: TDMA based wireless virtualization.

3.4.3 FDMA-TDMA Combination:

Each experiment is allotted a unique frequency and a unique time slot. 802.11a nodes can operate on 12 orthogonal frequencies. As an example, 4 logical groups can be formed each with 3 distinct frequencies partitions. The figure below depicts a scenario where a node is virtualized into 6 slices using two orthogonal frequency partitions in 3 time slots. Each virtual node is identified by the unique id FPx,TSx.

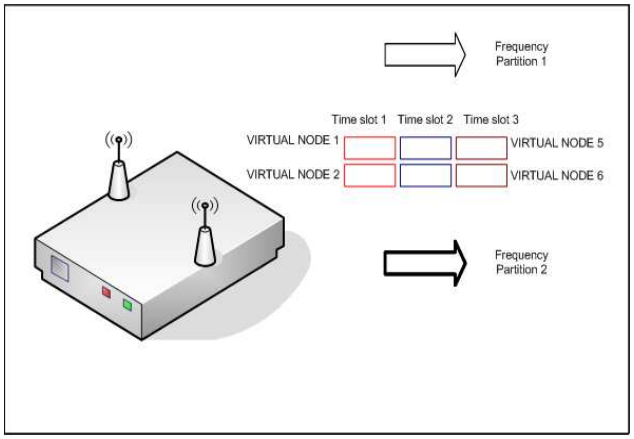


Figure 3.3: Combined FDMA-TDMA based wireless virtualization.

3.4.4 Frequency Hopping (FH):

The FDMA and TDMA techniques can be combined in another way to design a technique called frequency hopping. Each experiment uses a unique set of frequencies at different time slots. This approach also faces scalability issues due to scarcity in the orthogonal frequencies. The main design challenges are the overheads of context switching and channel switching time. Use of multiple wireless interfaces can eliminate the channel switching time completely. All the nodes of an experiment will have to operate on the same frequency for a given time slot for the synchronization of the experiment and no two nodes belonging to different experiments shall be on the same frequency in the same time slot to prevent interference. These points add complexity to the design of the resource manager for the testbed.

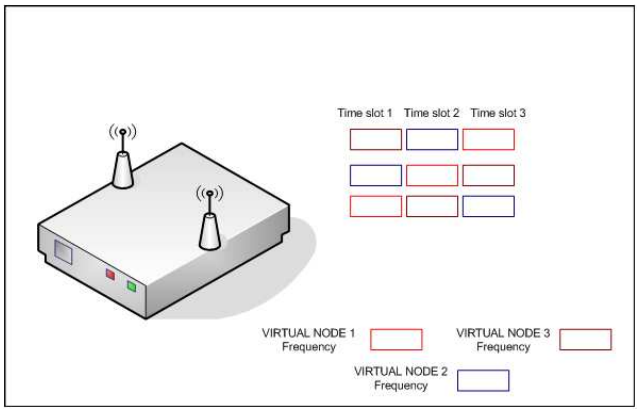


Figure 3.4: Wireless Virtualization using Frequency Hopping.

3.5 Slicing techniques for wireless networks

3.5.1 Space Division Multiple Access (SDMA):

The SDMA approach slices the testbed resources to provide sufficient spatial separation between the transmitting nodes to avoid interference across individual experiments. This means that SDMA provides virtualization across multiple nodes eliminating the need for experimenters to share experiment nodes. This partitions the experiments across the testbed grid. Use of different 802.11 protocols namely 802.11a/b/g also partition experiments and facilitate the run of simultaneous experiments in their respective partitions.

In this technique, scalability is limited by the number of "non-interfering partitions" and the number of "nodes per partition". It has a high requirement for space.

3.5.2 Combined SDMA, FDMA and/or TDMA:

The resources of each SDMA slice can be further virtualized by partitioning in frequency or time domain or a combination of both. This combination would support more general topologies with orthogonal frequencies or time slots assigned to the different experiments.

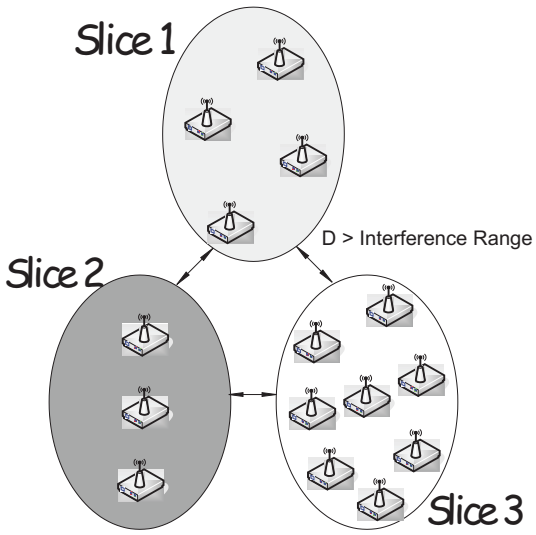


Figure 3.5: SDMA based wireless network "slicing".

3.5.3 FDMA based slicing:

In this slicing approach, the different experiments are assigned different slices of the testbed resources and assigned orthogonal frequencies. This ensures that the nodes of the experiment do not interfere with each other. Moreover, this approach does not have the limitation of space constraints. The number of orthogonal frequencies limits the scalability of this approach.

3.6 Virtual Access Point: VMAC based 802.11 virtualization

A Virtual Access Point [15] is a logical entity that exists within a physical access point. Multiple VAPs can be created in a physical AP, each providing the functionality of an AP. This Virtual MAC capability could be exploited to provide slicing based virtualization in the wireless testbeds for fixed AP based-star topology. Each experiment is mapped to a separate Virtual AP. The use of VAPs in wireless virtualization is motivated from its real world applications where multiple providers share common physical infrastructure. The VAPs operate on the same channel and save the cost of extra physical nodes. It also reduces the interference that may have aroused in a similar scenario when using different physical APs on the same channel. Figure 3.7 depicts the scenario

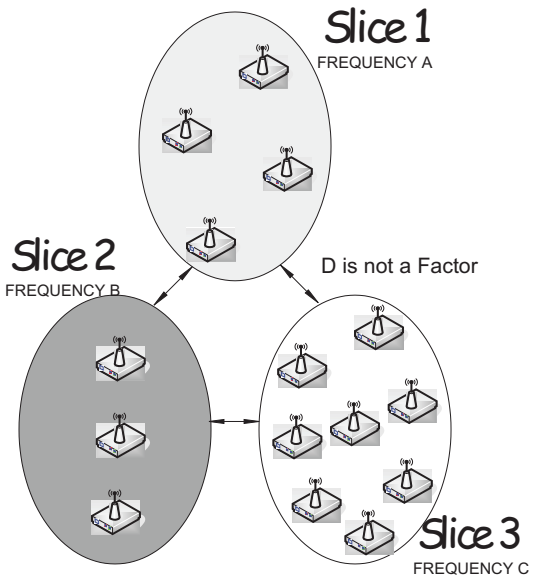


Figure 3.6: FDMA based wireless network "slicing".

where a VAP is used to support four concurrent experiments.

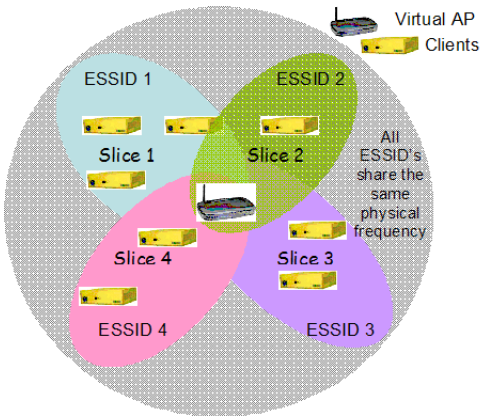


Figure 3.7: Mapping VAP defined ESSIDs to different experiments.

1. A new Virtual AP (VAP) is created for every new slice to partition the different experiments on ESSIDs. This is equivalent to creating new VLANs for every user experiment. All the slices share the same radio frequency and the access to the channel is controlled by the standard CSMA/CD Algorithm.

2. The performance of one slice would be effected by traffic flows in other slices. Also the control and management traffic of the 802.11 limits the number of slices

running on the VAP. Currently a maximum of 4 concurrent VLANs can run on a single physical access point (Artheros, Intel Wireless Cards). An AP with 'N' VAPs is referred to as N-VAP.

3.7 Most Suited Approaches

Selection of a virtualization scheme primarily depends on the resource being conserved. Wireless virtualization are targeted at either the conservation of nodes (hardware) or channels (frequency). Frequency multiplexing of the wireless channel (FDMA), allows for node conservation where the same node could be shared using a UML [16] like mechanism on multiple channels to emulate different experiments. Keeping in mind Moores Law, the present testbeds would not be held back on the number of wireless interfaces that they deploy. For instance, with access to 800 wireless interfaces on the ORBIT grid the focus was more on channel conservation rather than node conservation. In the next chapter, our results show that FDMA may not scale well, with provision for only three orthogonal channels in 802.11b mode of experimentation. Since we aim primarily at channel conservation, the list of virtualization schemes boil down to three choices :

- TDMA based Virtualization
- SDMA based Virtualization
- VAP based Virtualization

TDMA has been implemented and tested on the ORBIT grid in [17]. This approach runs multiple UML instances on the same node which use the same wireless device. They ensure through tight synchronization, that at any time all the nodes are running the same experiment slice across the network of nodes. Efficiency of implementation and overall performance seen with a TDMA scheme will greatly depend on:

- Experiment Synchronization: In TDMA, there is a stringent need for high degree of time synchronization between all the experiment nodes. Moreover, wireless experiments can potentially involve a high number of experimental nodes.

Though tools like the network timing protocol daemon (NTPD) [18] can provide distributed time synchronization, accuracies achieved may not be sufficient.

- Design Dilemma: The choice of time slot allotted to the different experiments is another design issue for the TDMA approach. A small value may not be possible due to practical limitations of wireless hardware like switching time and a large value would adversely affect the performance and results in delay sensitive experiments. Since, in this approach, several concurrent experiments share one or more physical nodes, there is also a need to provide isolation on every node between the experiments.

The TDMA approach requires design and deployment of a complicated infrastructure on current testbeds like ORBIT which does not seem plausible. To offset these disadvantages we will compare and evaluate the SDMA and VAP-based approaches towards wireless network virtualization.

Chapter 4

Experimental Set-up and Results

4.1 Experiment Configurations

In the initial part of our study, we are aiming to determine the amount of performance degradation ,if any, that will be seen with the use of VAPs instead of an AP followed by the performance comparison between SDMA and VAP based virtaulization schemes. To ensure this all measurements will adhere to the following:

- Relative Measurements: To ensure that the vagaries hardware and the experiment environment do not have a significant role in the generated results, results for all the scenarios are based on the experiments performed in succession. While using this experimentation approach we are making an assumption that though the use of different hardware may result in different relative results, the trends seen with the results will hold.

- Traffic Type: We make inferences on the results based on UDP traffic which represents a best effort service. However, we also use TCP traffic to show its effect on a virtualized network.

- Uplink Flows Only: Our study is mostly restricted to Uplink flows unless explicitly specified. Uplink flow implies traffic flowing from clients to the Access Points.

- Protocol: Our study is restricted to 802.11a [19] protocol unless mentioned otherwise. 802.11a supports multiple data rates upto a maximum of 54Mb/sec. Throughout our experiments, we use a bit-rate of 36Mb/sec.

4.2 Performance Metrics

Metrics are selected on the basis of their popularity in experiments. In this paper we will consider three metrics that have the most impact on the performance in wireless networks. They are as follows:

1. Throughput: It is necessary to see how the throughput difference will vary with other changing experiment parameters such as packet size and net offered load.

2. Delay: Relative latency between the different scenarios for virtualization is an important metric. We also compare latency results of virtualized and non-virtualized cases to see differences that may be seen in the experiments.

3. Jitter: Jitter in a link is defined as the variance of delays between packet inter-arrival times. Jitter proves to be an important factor for experiments that evaluate the performance of algorithms dealing with real time audio traffic.

In our experiments, we use UDP, TCP and RTP traffic for measurements and comparisons. The popular bandwidth measurement tool IPERF [20] was used to generate UDP and TCP traffic. Delay experiments with Iperf will not produce the right results because the traffic generator has a blocking operation i.e., it relies on blocking sockets. To obtain steady state delay values, ping [21] was used as a simple tool to measure the latency. Even though the granularity provided by ping is argued to be considerably coarse, it works fine for our application. For experiments with video applications, we use VLC media player [22] for streaming and receiving the video. The bit-rate and jitter measurements are recorded after analyzing the tcpdump [23] using Ethereal [24].

4.3 Proof-of-Concept Integration Experiment

In the first section in the results, we evaluate an experimental scenario with integrated tests of the PlanetLab and ORBIT testbed. This integrated experiment evaluates FDMA based slicing of the ORBIT testbed to support concurrent experiments. Towards the end, we discuss the scalability issues with FDMA based slicing approach for virtualization.

Aim: In this proof-of-concept experiment we show the use of our architecture in testing the performance of video delivery algorithms. The objective of this experiment is not to do a comparative study of the video delivery algorithms themselves, but rather demonstrate a way to evaluate these algorithms with our setup.

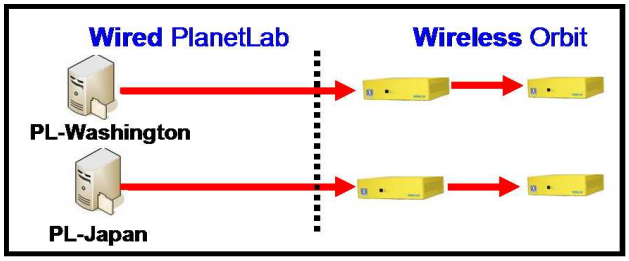


Figure 4.1: Integrated Experiment Layout with FDMA slices on ORBIT nodes.

Topology: Figure 4.1 shows the topology for this experiment. The wireless parameters we use for our experiments is shown in Figure 4.9. We consider a typical scenario for streaming video delivery across a network path that includes an edge wireless link. The experimentation includes two flows from PlanetLab nodes to two Access Points configured within ORBIT. The Access Points relay traffic to their respective clients over orthogonal channels. Isolating experiments on different and possibly orthogonal frequencies is one of the easiest approaches to wireless virtualization (FDMA based slicing). The video streamed from PlanetLab goes over the internet giving experimenters the characteristics of a realistic network. Physical and MAC layer parameters such as channel rate, transmission power, injected noise, packet sizes can be varied to test the effect of the wireless link on the overall performance of the link. The video is streamed and played using the Video LAN (vlc) player. The ORBIT Measurement

Parameter	*Value*
Channel Rate	36Mbps
Offered Load	Time Varying
Experiment Duration	2 Minutes
Averaging Duration	Per Second
Operation Mode	802.11a

Figure 4.2: Experimental Parameters Used With ORBIT Nodes

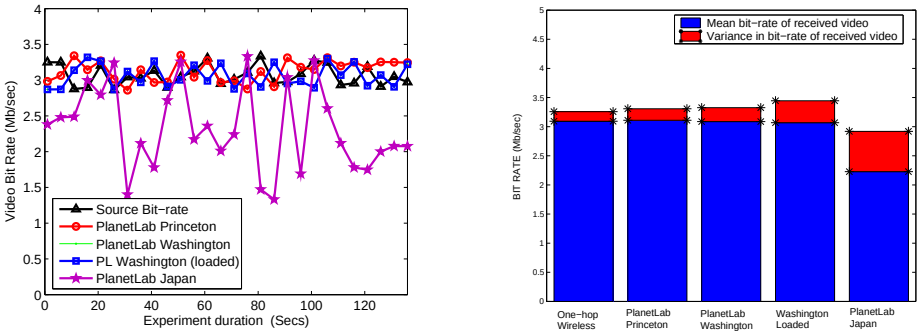
Library (OML) framework of ORBIT provides means for recording the bit rate and jitter in the video received at the ORBIT clients. We record measurements for different PlanetLab nodes in terms of their geographical distance from ORBIT.

$Experimentsetting$	$MaxJitter\ (ms)$	$MaxDelta\ (ms)$	$MeanJitter\ (ms)$
$Wireless\ One\ hop$	1.68	7.66	0.98
$Pl - Princeton$	1.69	7.69	0.98
$Pl - Washington$	1.88	8.76	1.05
$Pl - Washington\ (Loaded)$	35.76	415.8	3.62
$Pl - Japan$	52.18	779.02	7.38

Figure 4.3: Jitter results observed with different PlanetLab nodes serving the same video over the internet to wireless clients in the ORBIT grid.

Jitter measurement: Figure 4.3 shows the results for the jitter values from the FDMA experiments. A video was delivered from PlanetLab nodes in Princeton (NJ), Washington and Japan. Results show relatively comparable jitter values for the Princeton and Washington PlanetLab nodes. Japan on the other hand sees a higher jitter for video delivery possibly due to higher traffic and geographical distance. In another case we simulated a heavily loaded server by adding traffic to the Washington node. The results show the increased delay for such a busy node. The jitter obtained for one-hop wireless link is also shown as a case of an exclusive ORBIT experiment and that might be the baseline scenario for comparisons. All these readings are easily obtainable either through the OML framework using the integrated tcpdump tool.

Bit rate measurement: Figure 4.4(a) shows a plot of the observed video bit rate at the client as a function of time. The observed bit rate for the videos is lower for the PlanetLab node in Japan as compared to those in Princeton or Washington. Calibration tests performed over the one hop wired and wireless networks server as a baseline for comparison. Figure 4.4(a) also shows the performance of the individual flows in terms of the average bit rate at the wireless client for the same video. Surprisingly the increased load on the Washington node only shows increased jitter in the video delivered with a comparable mean bit rate. The PlanetLab node in Japan has a deteriorated bit rate at the client.



(a) Observed bit rate with the same test video being delivered from different PlanetLab nodes.

(b) Mean and variance statistics from the video bit rates observed at the client node.

Figure 4.4: Measurements of bit-rates for different PlanetLab nodes.

These results could help provide a deep understanding and evaluation of proposed video delivery algorithms. They could also help to better understand the limitation of media delivery over wired/wireless networks. In addition, research on spectrum allocation, bandwidth management and inter- Access Point communication protocols would require the presence of a similar framework.

A disadvantage of using frequency division multiple access (FDMA) based slicing for virtualization on the ORBIT grid is that it does not scale well with the number of available orthogonal frequencies. For e.g., 802.11b operates on 11 channels, but only 3 of them are orthogonal, namely 1,6 and 11. We investigate the effect of experiments on channel 2,3,4,5 and 6 will have on an experiment using channel 1. Since these channels are non-orthogonal, some interference is expected that will result in drop in throughput. We make use of two extreme scenarios for comparison. The worst case would be when the nodes operating on non-orthogonal channels are close in space and the transmission power is set to maximum possible value (20 dbm). On the other hand, the best possible case would be to have the node pairs farthest apart on the grid and setting the value of power to 1 dbm. The throughput values for the pair-wise communication are plotted against the interfering channel as shown in Figure 4.5.

As expected, the effect reduces as we move from channel 1 to channel 6. The variations in throughput are a result of the working of the CSMA protocol. Moreover as we increase spatial separation and decrease transmission power, the aggregate throughput

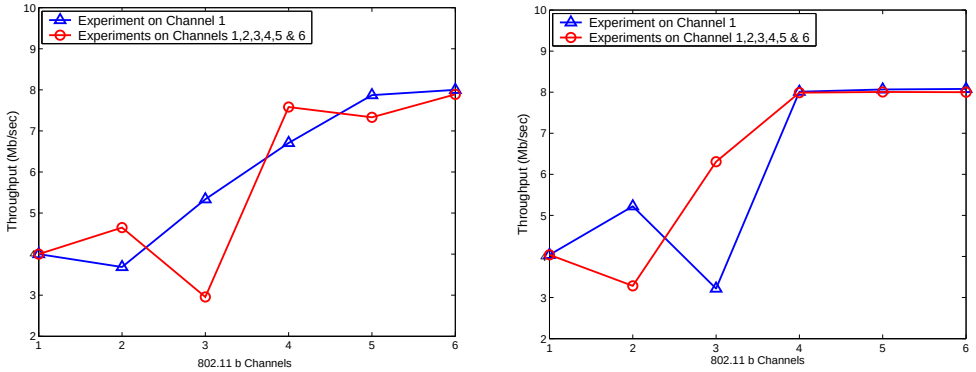


Figure 4.5: Investigation of FDMA based slicing with 802.11b. (Left) Experimental nodes chosen close to each other (Right) Experimental nodes chosen with maximum space separation and minimum transmission power.

increases. The results for channel 4 and 5 suggest that FDMA based slicing can be incorporated in indoor wireless testbeds but it needs to be combined with other approaches like SDMA and using means such as transmission power control to increase its scalability. In the following sections, we show results of our investigation on more scalable virtualization schemes.

4.4 Virtual Access Point Based Time Sharing

The VAP technique is based on logical partitioning of the channel with individually assigned ESSIDs that run on a physical access point while emulating the behavior of conventional physical APs to stations in the network. Using a VAP allows for two or more AP mechanisms to share the same channel thereby helping channel and energy conservation. We plan to exploit this mechanism to provide virtualization of fixed star topology wireless networks. The individual experiments would be assigned different ESSIDs on the same physical AP. As described in the previous chapter, the provision of channel multiplexing by VAPs makes it a suitable candidate for supporting long-term experiments.

4.4.1 VAP Implementation Overview

The concept of VAPs is incorporated in the 802.11 driver which operates just above the MAC layer and below the IP layer as shown in the Figure 4.6. The driver provides the

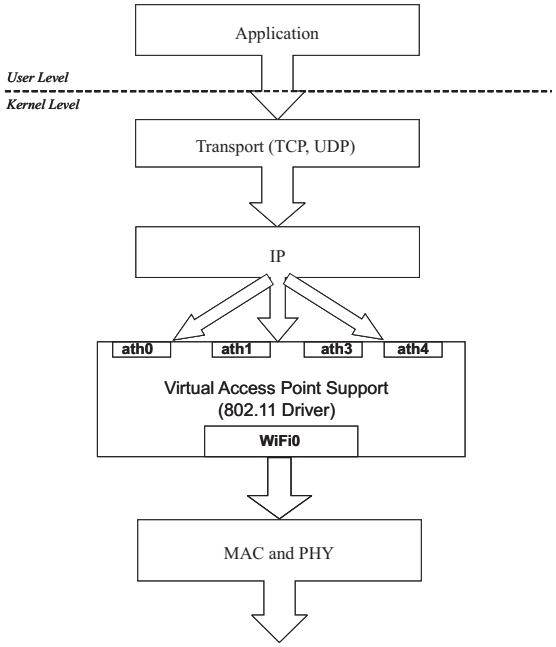


Figure 4.6: Position of the VAP creation and maintenance architecture as a part of the network stack.

multiple AP abstraction to the higher layers though it is operating on a single lower layer. Hence all the protocols operating on the machine are agnostic to the presence of the abstraction. In order to understand the performance shown in the results section we provide a brief overview of the new driver along with possible causes for seeing a degraded performance with multiple VAPs. The details mentioned in this part are based on the information available in the MADWIFI driver source code [25].

VAP Creation: On a command from the user tool (such as *wlanconfig* or *iwconfig*), IOCTL calls are made to the driver which creates a VAP as mentioned in [26]. The IOCTL calls are redirected from the kernel to specific API defined in the driver which results in allocation of required resources (including the assignment of MAC addresses, setting up queues) and marking entries in the drivers node table.

Receiving Information: A brief overview of how MAC frames are received are shown in Figure 4.7. Every time a MAC frame is detected, it triggers an interrupt to the driver. The servicing of this interrupt results in the buffering of the incoming frame by the driver. Once the entire frame is detected, the driver tries to match the MAC

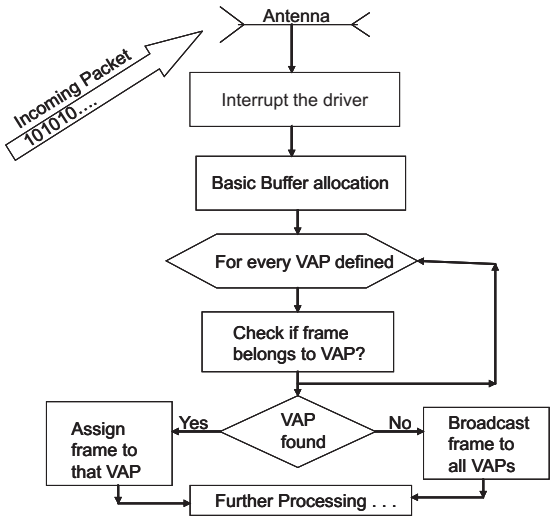
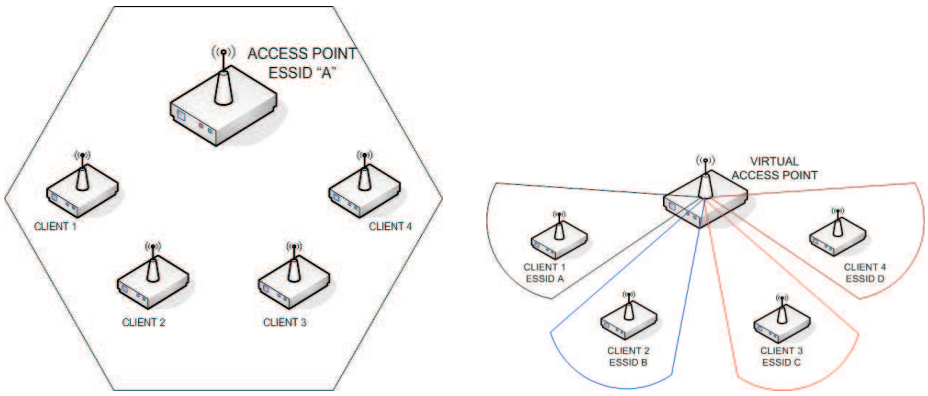


Figure 4.7: VAP execution flowchart based on the MADWIFI driver.

address in the frame with the MAC address of the virtual access points running on the interface. All the virtual interfaces have unique MAC addresses. If the MAC address of the incoming frame does not match that of any VAP, the frame is simply sent across to all the VAP interfaces on the assumption that if the frame does not belong on a VAP it will be dropped anyways. The comparison and the post processing of the MAC frames accounts for some extra overhead which corresponds to the overheads seen with VAPs. A virtual access point (VAP) is a relatively new concept and the performance with a virtualized driver has not been characterized. We start by comparing the performance of a VAP with a physical access point even before we can compare it with space sharing of the channel.

4.4.2 Virtual Access Point Overhead

Before we evaluate the benefits of using VAPs, we consider it important to determine the overheads of maintaining state of multiple networks at a single hardware device. The experimental setup for comparison is as shown in Figure 4.8(a) and Figure 4.8(b). Figure 4.8(a) shows a setup with one AP and all four clients within the same network. Figure 4.8(b) has the same nodes. However, each of the clients now belongs to a different logical network created by the VAPs. Results are evaluated for both uplink



(a) A physical access point and four clients (b) Four virtual access points and their individual clients

Figure 4.8: Experimental setup for performance evaluation with physical and virtual access points.

Parameter	*Value*
Channel Rate	36Mb/sec
Aggregate Offered Load	50Mb/sec
Experiment Duration	5 Minutes
Averaging Duration	Per Second
Operation Mode	802.11a
Traffic type	Uplink
Chipset	ATHEROS
Driver	Madwifi(0.9.3.1)

Figure 4.9: Experimental Parameters Used With ORBIT Nodes

and downlink performance with a saturated channel and equal offered load per client. Other experiment parameters were maintained as shown in Figure 4.9.

Figure 4.10 plots the observed per client throughput for uplink and downlink traffic. Performance of a single client with a single access point is taken as a reference for comparison. Key observations that can be made from the results are:

- As with any time sharing approach, the entire bandwidth (which is seen in the scenario with 1 client) is now shared across 4 clients.
- Uplink traffic sees a slight deterioration in performance with both the AP and the VAP as compared to the reference flow with 1 client.
- There is no added deterioration with uplink traffic using VAPs for having clients

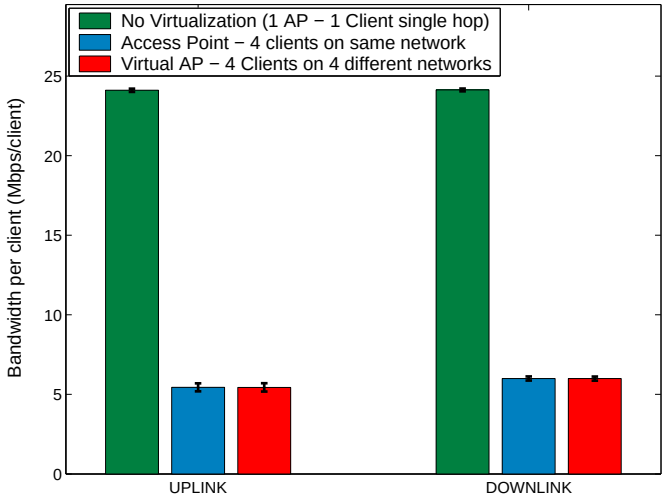


Figure 4.10: Impact of virtualizing using VAP based time-sharing approach.

on multiple networks, as compared to an AP with all clients in one network. Hence, we can conclude that the deterioration seen in both cases which leads to a net channel throughput of 21.76$Mbps$ as compared to 24.11$Mbps$ is due to the increased channel contention overhead.

- Downlink overheads for both AP and VAP with 4 clients are neglibible as compared to that with a single client.

- Error bars for both cases show little variance in throughput.

Thus empirical evaluations and a study of the source code reveals no significant overhead for running VAPs on a single node. This suggest that experiments evaluating aggregate throughput with test setups running a single AP or multiple VAP should generate comparable results with the channel utilization being determined by the number of clients. Based on this conclusion, we can now compare the performance of virtualization with VAP and that with space separation.

4.5 Space Division Multiple Access(SDMA) on ORBIT

The idea behind SDMA slicing is to allocate a unique set of resources to the users partitioned in space. Individual experiments are mapped to different subsets of the testbed, termed as slices, to ensure minimum possible interference. The ORBIT wireless testbed

is located in a 20 meter x 20 meter space and the testbed nodes are in close physical proximity of one another. Under these conditions, partitioning the resources in space to avoid interference would not be practically possible. This holds true for most of the emulator testbeds. "Artificial stretching" of the distance is achieved by controlling transmission power of the nodes and using noise injection to emulate barriers between the nodes of different experiments. Our experiments explore the possibility for virtualizing the ORBIT grid using SDMA by controlling power in addition to providing spatial separation. In an artificially stretched SDMA, all the experiments are multiplexed on the same channel and hence face CSMA-based contentions with the nodes belonging to other experiments.

4.6 VAP versus SDMA

Strictly, SDMA slicing is different from other virtualization schemes. Unlike in most of the virtualization approaches, each node of a SDMA slice is entirely assigned to the user and there is no logical partitioning of the testbed nodes. Many wireless experiments evaluate modified MAC and PHY layers protocols, and hence sharing nodes with other experiments is not possible. Such scenarios necessitate the need to allot a slice to each experiment. In our setup for SDMA, experiments that use a MAC and PHY layer which is compatible with the IEEE 802.11 standard would operate successfully in a virtualized environment. The SDMA experiments will not be restricted to fixed star topologies like in the case for the VAP. More generic topologies like ad-hoc are possible to execute in a SDMA environment. However, due to spatial constraints in testbeds, incorporating arbitrary topologies in the slice allocated to the experiment may be challenging or impossible for some experiments.

Unlike in the SDMA based approach, the VAP performs logical partitioning of a physical node (Access Point in this case). This partitioning implies a need for isolation between the different experiment instances on the same physical node. Though there are several techniques that virtualize the resources of a single node by supporting multiple concurrent instances of Operating Systems (OS), there is penalty in terms of added complexity and performance loss. Moreover, sharing a node with other experiments

limits the capabilities to modify the MAC/PHY layer parameters. The VAP however conserves network resources and is not limited by the space constraints of the testbed. With VAP, creating topologies is more efficient and easier than SDMA.

To summarize:

- SDMA and VAP approaches provide channel multiplexing options for virtualization of a wireless testbed and hence qualify for supporting long running concurrent experiments.
- In the SDMA approach, for every new experiment, a "slice" of the testbed resources is assigned to the experiment. The challenge is to assign the topologies of the different experiments as far apart as possible. In the VAP case, for every new experiment, a new ESSID is created on the VAP. This is similar to creating new VLAN for each experiment.
- In both the approaches, the new slice (experiment) will share the same radio channel and access to it is controlled by CSMA mechanism. Therefore there is bound to be undesirable effects of traffic of one or more experiments on the other experiments.
- It should be noted that in the case of VAP, if the experiments use downlink traffic, the transmission is time scheduled. Hence there are no CSMA based contentions. This case is different from the scenarios with VAP uplink and SDMA, where experiments face CSMA based channel contention with other experiments. In a later section, we discuss the difference in performance of these scenarios. However to make a fair comparison between VAP and SDMA, we characterize the VAP with uplink traffic only.

Figure 4.11 depicts the topology for our comparisons. We compare the performance of both virtualization schemes by mapping four co-existing experiments. Each individual experiments consist of an APclient single hop wireless. In the case of VAP based virtualization, we use a VAP with 4 clients with each experiment mapped to a different

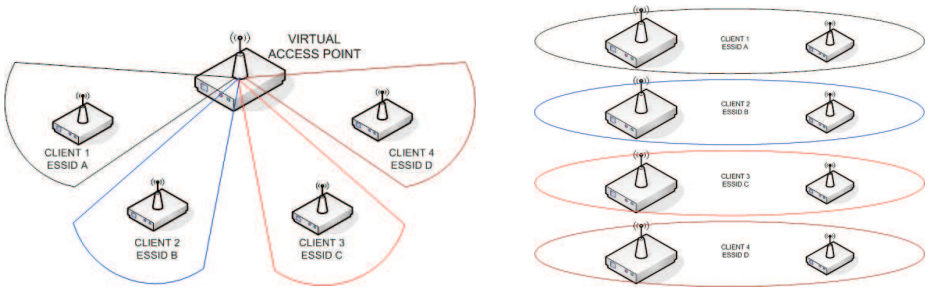


Figure 4.11: Logical VAP and SDMA topologies.

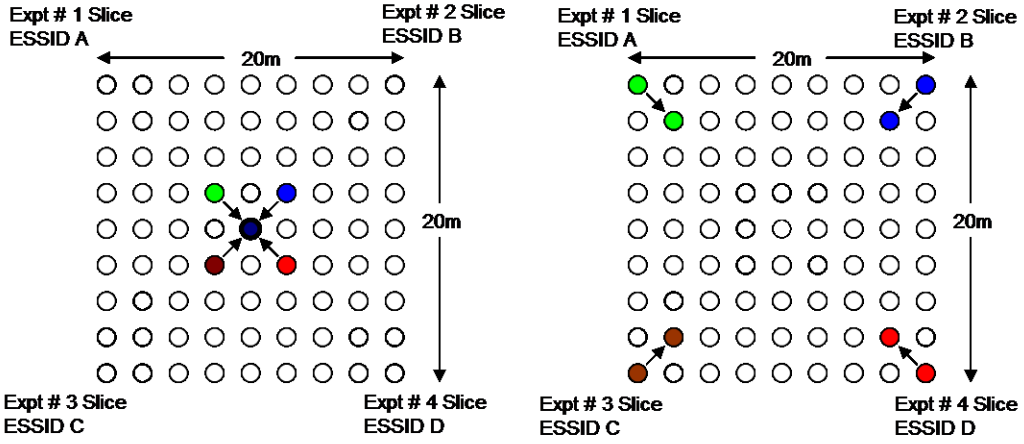


Figure 4.12: (LEFT) Topology for VAP-based virtualization. (RIGHT) Topology for investigation of SDMA-based virtualization on ORBIT testbed.

ESSID. In the SDMA scenario, we consider pairwise communication between 4 APs with 1 client each placed on the four corners of the ORBIT grid to ensure maximum spatial separation. In addition, the transmission power is reduced to the minimum possible value to ensure minimum interference between the experiments. Figure 4.12 shows the experiment set-up for the comparison between VAP and SDMA slicing on the ORBIT testbed.

4.6.1 Throughput Comparisons

The throughput performance is the key metric to keep in mind when performing virtualization of a networking system. In this section, we study the performance and throughput variations for the two virtualizing schemes: SDMA and VAP.

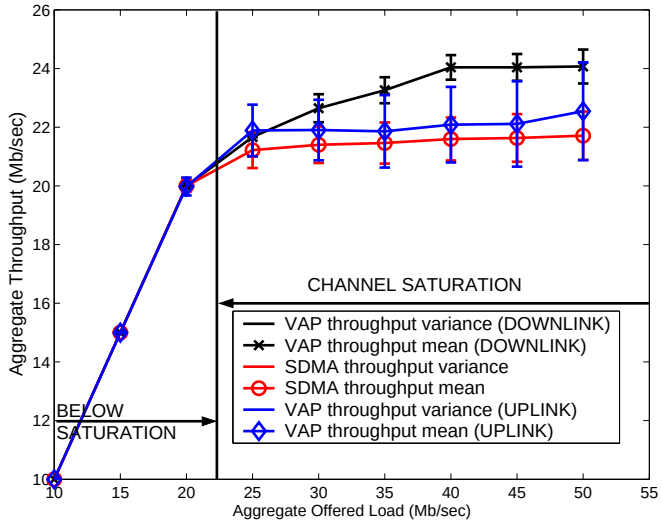


Figure 4.13: A comparison of available bandwidth with offered load for SDMA and VAP based virtualization schemes supporting four concurrent experiments.

Variation with offered load:

Performance comparison of the VAP versus space separation (SDMA) uses the experiment setup as shown in Figure 4.11. Figure 4.12 depicts the topology for the two scenarios arranged on the ORBIT testbed. We compare the performance of both virtualization schemes by mapping four co-existing experiments. Each individual experiments consist of an AP-client single hop wireless.

Figure 4.13 shows the results for the aggregate throughput for virtualized experiments with varying offered load. Initially the offered load is kept low to show results below saturation. In this case we observe that both SDMA and VAP have a comparable performance. However, as the offered load is pushed into the saturation limits of the channel, there is a clear difference in the throughput. While collecting results we have taken care to disable the MAC frame aggregation which is a default feature of the wireless driver while operating in the saturation region.

The difference in performance observed in Figure 4.13 is due to physical layer capture [27]. Capture is the phenomenon by which a receiver is able correctly decode one of the many simultaneously colliding packets due to relatively high signal to noise ratio. Physical layer capture can be detected either by sniffing packets from the channel with

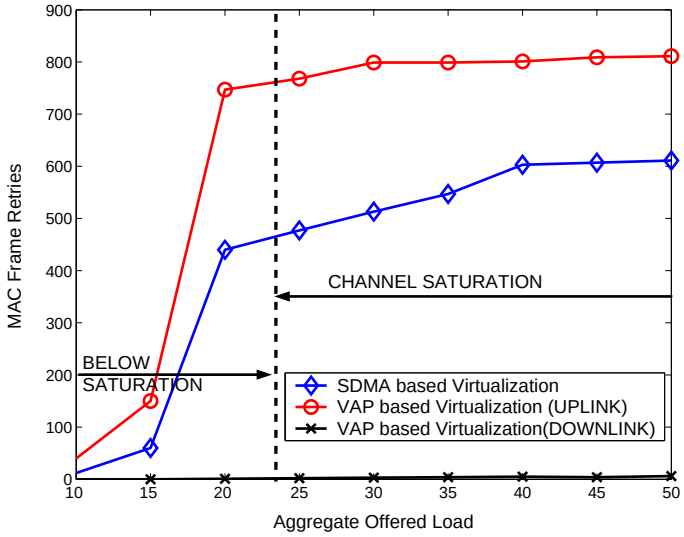


Figure 4.14: A comparison of number of MAC frame retries for SDMA and VAP based virtualization schemes supporting four concurrent experiments.

multiple sniffers (since the sniffers themselves are susceptible to capture) or by comparing the number of MAC retries with a case without capture. Figure 4.14 shows the the aggregate number of MAC retries with the VAP and the SDMA case. It is clearly seen that the number of MAC retries with SDMA were significantly lesser than with VAP since the receivers are able to decode colliding packets due to capture.

One more observation made from Figure 4.13 is if the experiments consists of pure downlink traffic (i.e., traffic from Access Point to client), the bandwidth provided by the VAP exceeds that of the SDMA scenario. This performance increase is attributed to the fact that the VAP has a time-scheduled downlink transmission unlike the SDMA scenario in which the four Access Points have to contend for the channel with CSMA basic access. Figure 4.14 confirms this behavior showing that there are minimal collisions with VAP operating in the downlink since there are no MAC collisions. Comparing VAP downlink and SDMA setting is not fair as the latter faces CSMA contentions on the downlink. Hence, our comparison study is restricted to VAP uplink and the SDMA setting.

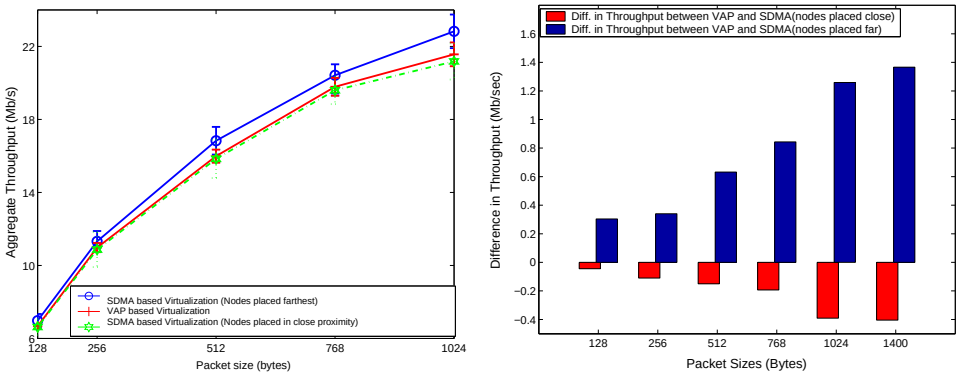


Figure 4.15: A comparison of available bandwidth for SDMA and VAP showing the effect of space and transmission power control.

Variation with packet size:

Packet sizes in a saturated channel determine both the MAC and physical layer overhead as well as the aggregate channel access time. Smaller the packet sizes lesser the net throughput and decreased efficiency and vice-versa. The goal of these set of experiments is to test if varying packet sizes have similar effect on performance with both the VAP and SDMA approach.

To determine the effect of node positioning on the capture effect with SDMA, we measure SDMA performance with two setups : (a) The nodes of the experiments are setup far from each other as with the conventional setting (Figure 4.12). (b) the experiments are setup near to each other (Figure 4.16). For each experiment run packet sizes were varied and the aggregate throughput was measured. Figure 4.15 shows the results of the experiments. We plot the difference in throughput of each of the SDMA setups from the VAP experiment to show the performance gains.

The general trend for both setups follows intuition where performance is poor for small packet sizes and vice versa. However, SDMA setup with nodes placed far away had the advantage of decreased interference and improved performance with higher capture. The positive increase in difference in throughput shows that the benefits of capture increase with packet sizes. The SDMA setting without spatial separation shows a degraded performance as compared to the VAP setting. The MAC-ACKS in the downlink see lesser interference and collisions in the VAP due to time scheduled

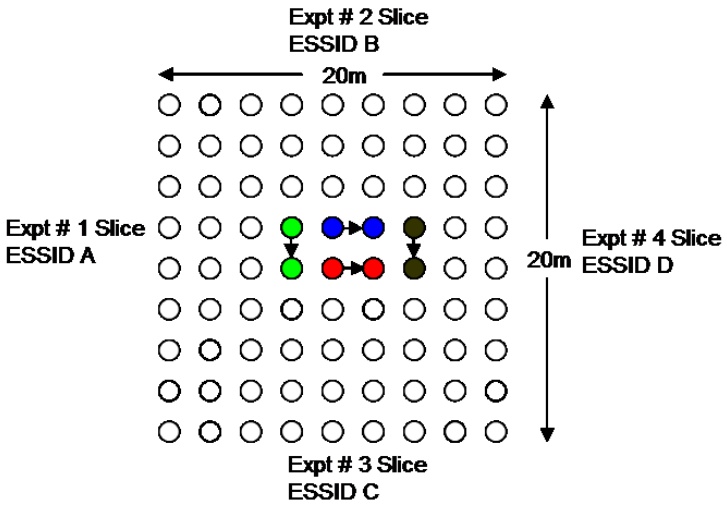


Figure 4.16: Topology for investigating SDMA with nodes placed close to each other on ORBIT.

downlink transmission and hence the setting has a better performance as compared to the SDMA without spatial correlation. As the packet size increases this difference is even more pronounced since the effect of a collision is more pronounced for larger packet sizes.

TCP Throughput Variations:

Testing the performance over a TCP traffic may not yield the right results for our comparison since it may involve the effect of the higher layers in the protocol itself like the rate and error control with TCP. However several wireless experiments involve evaluation over TCP. Figure 4.17 shows the time variation of TCP performance for VAP and SDMA scenarios. The variations in throughput are more for TCP flows than in the case with UDP flows for all the scenarios due to additional TCP-ACK traffic in the downlink. As expected, the variations in throughput are more for the SDMA case than the VAP attributed to the additional amount of downlink contention. Again SDMA scores high due to the capture effect.

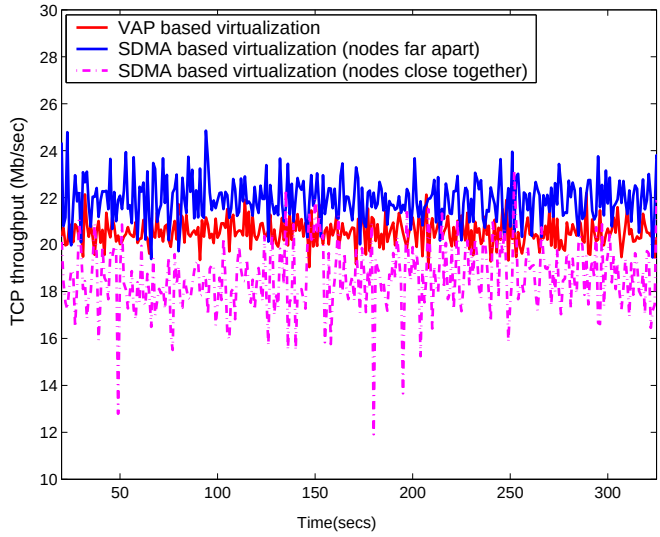


Figure 4.17: A comparison of the available TCP bandwidth for SDMA and VAP based virtualization schemes supporting four concurrent experiments.

4.6.2 Delay-Jitter Comparisons

Experimenters often use delay as a metric measured for performance of an experimental setup. Jitter, defined as the variance of delay is also an important metric in the performance of real time traffic such as voice or video. We will compare the effect of VAP and SDMA-based virtualization on both observed delay and jitter per experiment.

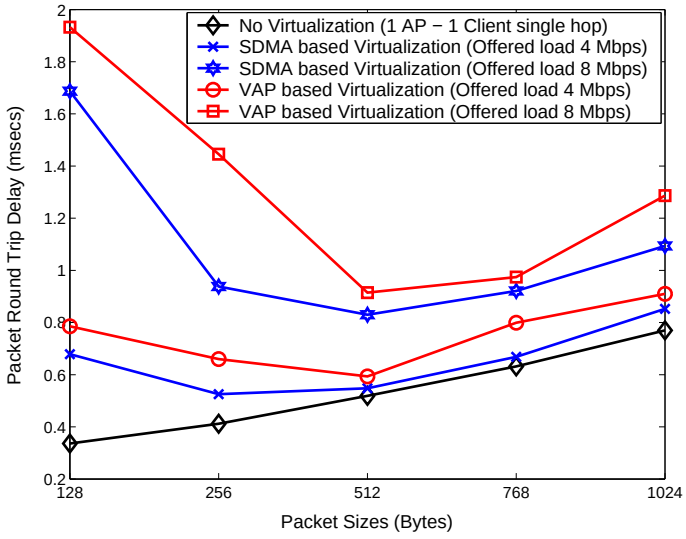


Figure 4.18: Round trip delay variations with packet size for VAP and SDMA based virtualization schemes as compared to the non-virtualized scenario.

The experiment setup for delay and jitter measurements is the same as before, shown in Figure 4.12. Figure 4.18 shows the round trip delay measurements for the following cases:

1. No Virtualization
2. SDMA with different offered loads
3. VAP with different offered loads

We use two different offered loads to test the deterioration in delays with varying offered loads. With no virtualization, experiments get a linear increase in delay with packet sizes due to increase in transmission times. This deduction is based on the assumption that the individual experiments have a one hop wireless topology with single flows. Hence there are no CSMA contentions. However, in the case of virtualization, experimenters have a V-shaped curve for delay results. The nodes of every experiment face CSMA contentions with nodes from other experiments. Delay values decrease with packet size for smaller packets as the CSMA contentions decreases with lesser number of packets. However for large packet sizes, the transmission and queueing times are more prominent than CSMA contentions and the delay follows a similar trend. The per-packet delays for SDMA experiments are lower as a result of capture effect. Capture ensures that the MAC frames are received despite collision, which lowers the net MAC retries (Figure 4.14) for getting a packet across and consequently the queueing delays.

Figure 4.19 shows the round trip jitter as a function of different packet sizes and offered load. The trend for jitter follows the same pattern as that for delay i.e., high for small packets, decreases for bigger sizes and slightly increases for the biggest packets sizes. However, unlike delay, the jitter decreases with packet size for no virtualization scenario. Since we measure RTT jitter, there is contention even with one hop, single flow topologies. Hence, as the packet size increases, for a constant offered load the number of contending packets decrease resulting in decreased jitter.

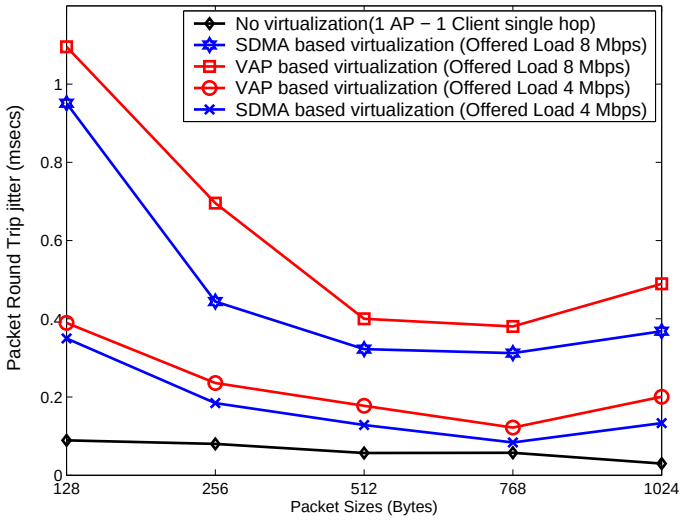


Figure 4.19: Round trip Jitter measurements for video delivery with different approaches.

4.7 Inter-Experiment Interference Illustrations

Ensuring isolation between experiments is one of the most important goals of wireless virtualization. Often it is seen that abuse of resource by one device sharing a resource leads to a deterioration in performance for other experiments sharing the platform. We will elaborate the consequences of these inter-experiment effects with time and space separation for virtualization and suggest approaches (described in the next section) to mitigate the same.

In this section, we use the same experiment set-up as used in the throughput, delay and jitter characterization of VAP and SDMA based virtualization schemes. The experiment setup is shown in Figure 4.12.

4.7.1 Channel Saturation

We define two cases to show the effect of channel saturation in VAP based virtualization on the experimental results. In the first case, initially all experiments have a uniform offered load of 5 Mb/sec with the channel operating below saturation. The offered load of the experiments are increased at distinct time intervals for a duration of about 25 secs. The increase is sufficient to take the channel into saturation. Figure 4.20 shows the

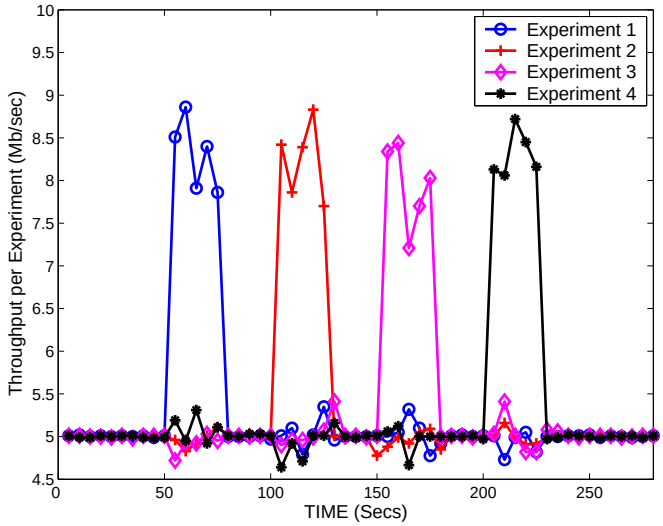


Figure 4.20: Effect on the performance of experiments when one of the experiment pumps traffic to push channel into saturation.

effect of channel saturation on the other experiments on the same VAP. The packet sizes of all experiments were kept the same. In the second case, the packet size of one of the experiment is relatively smaller compared to other experiments on the same VAP. In this scenario, it can be observed from Figure 4.21 that the throughput for the experiment using smaller packet size drops in addition to increase in variance. The CSMA algorithm works fairly at steady state in terms of number of packet transmissions. Since the experiment with smaller packet size will have relatively more number of MAC frames to transmit than the experiments with larger packet sizes, the former suffers from drop in number of MAC transmissions.

4.7.2 Throughput Coupling

In this section we study the transient behavior of the experiments using VAP and SDMA based virtualization. It gives a clearer picture on how the different experiments of the two approaches interact when sharing a common wireless channel. To quantify the inter-experiment effects, we define a coupling factor between virtualized experiments as:

$$\sigma_{(nv\_num,\ \ v\_num)} = \frac{(T_{non-virtualized} - T_{virtualized})}{T_{non-virtualized}} \tag{4.1}$$

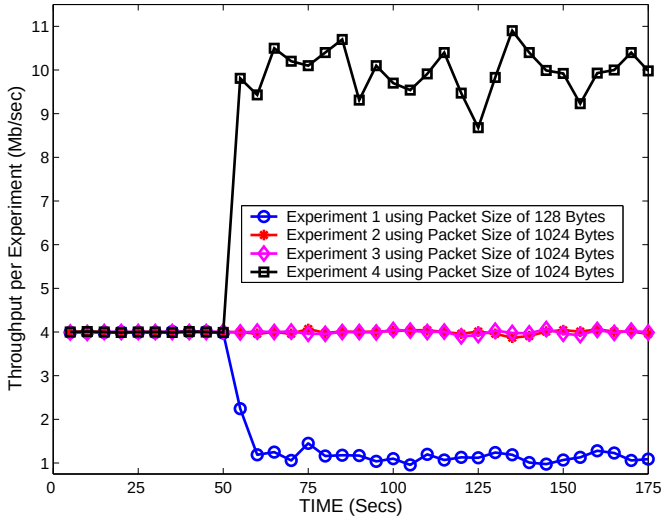


Figure 4.21: Effect on the performance of experiments using smaller packet sizes when one of the experiments pumps traffic in VAP based virtualization.

$\sigma_{(nv\_num,v\_num)}$ indicates the coupling between non-virtualized experiment nv_num and virtualized experiment v_num. $T_{non-virtualized}$ and $T_{virtualized}$ represent the throughput of the experiments in the non-virtualized and virtualized. σ takes values between 0 and 1. A σ of 0 indicates an ideal experiment setup where there is no interference between experiments while a σ of 1 indicates complete interference of one experiment with the others.

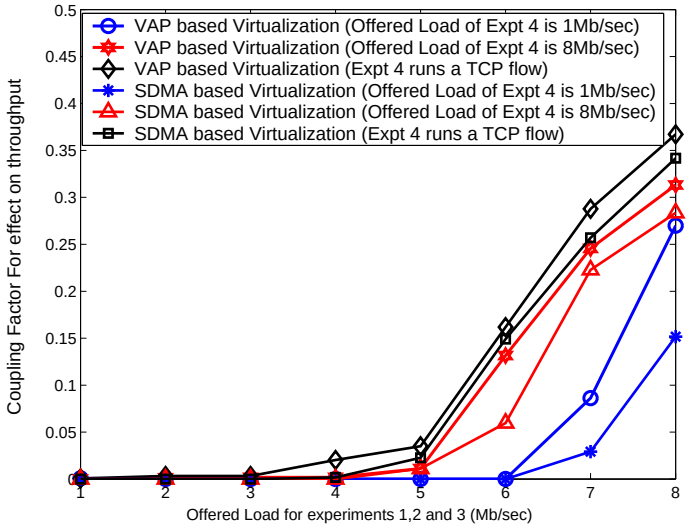


Figure 4.22: Coupling Factor for effect on throughput of experiments due to traffic from other experiments.

In the scenario with four concurrent experiments for both VAP and SDMA, we observe the impact of the fourth experiment on the first three experiments for different traffic scenarios of the fourth experiment. In the first plot, we show the coupling factor for the first three experiments with varying offered loads for both VAP and SDMA-based approaches. The packet size used by all four experiments was set to 1024 bytes. The plot of the throughput coupling factor is shown in Figure 4.22:

- In the initial runs we keep the offered load of the fourth experiment at 1 Mbps and find the coupling factors for both virtualization schemes is negligible for low offered loads and start to become prominent after the offered loads for the three experiments crosses 6 Mbps. The channel is driven into saturation and effects the performance of the experiments. The effect is less for SDMA, since the performance of SDMA is superior to the VAP.

- In the case where the offered load of the fourth experiment is about 8 Mbps the channel saturates at lower values of offered loads of the first three experiments and therefore the coupling factor is higher.

- In the case where the fourth experiment uses TCP, the coupling on other experiments observed is relatively higher than that with UDP. TCP flow pumps traffic at the maximum possible rate and its effect is more significant on the other experiments than that observed with a UDP flow. This increase can also be accounted by the overhead of the TCP-ACK traffic that increases the amount of contention among the different experiment flows.

In the second plot, we show the coupling factor for the first three experiments with varying packet sizes for both VAP and SDMA-based approaches. The plot of the throughput coupling factor is shown in Figure 4.23:

- The degree of inter-experiment effect on an experiment decreases as the packet size of the experiment increases

- Experiments with small packets have lesser effects on other experiments than experiments with higher packet sizes.

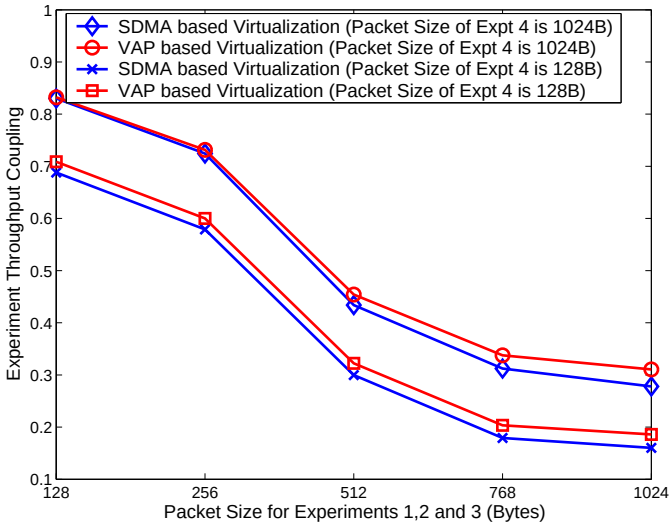


Figure 4.23: Coupling Factor for effect on throughput of experiments due to traffic from other experiments.

- In this case too, SDMA performs better than VAP due to capture effect. Capture effect becomes more prominent as the packet sizes of the experiments increase.

4.7.3 Jitter Coupling

Similar to throughput results, the experimental measurements of packet jitter is affected by traffic from other experiments. We investigate jitter coupling in VAP and SDMA-based virtualization approaches by streaming a video from a client to an AP as a part of one experiment and running UDP flows as part of the other three experiment. Figure 4.24 shows the plot for jitter coupling factor values for videos of different bit-rates for VAP and SDMA-based virtualization scenarios. The jitter coupling factor was calculated using Equation 4.1. The jitter values are calculated for a real-time experiment that streams videos of different bit-rates from a client to an AP. With no virtualization, it was observed that the jitter of the video does not depend upon its bit-rate. However, in the virtualized case as the bit-rate increases the jitter value increases. Moreover, the jitter values of the video increase as the channel approaches saturation due to increase in the offered load of the other 3 UDP experiments. Similar to throughput results, the jitter coupling is more for the VAP setting as compared to that with the SDMA virtualization.

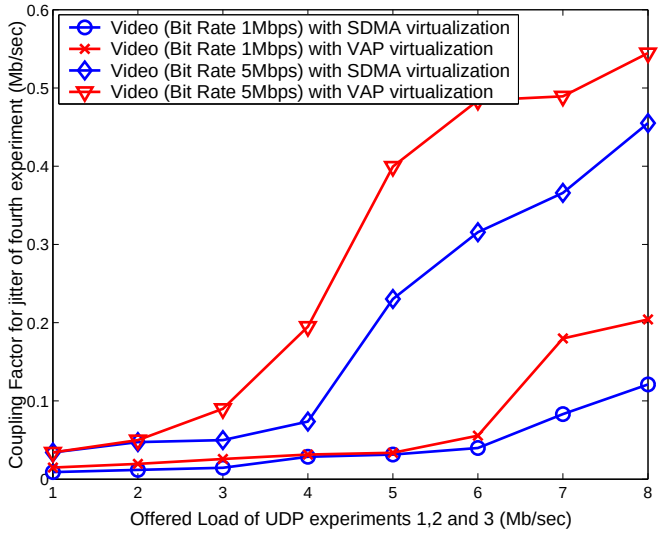


Figure 4.24: Effect on jitter measurements in channel multiplexing virtualization approaches.

4.8 Proof-of-Concept Integration Experiment with Policy Management for VAP based Virtualizations

This proof-of-concept experiment demonstrates PlanetLab-ORBIT integration with use of VAP based virtualization.

Aim : The goal of this experiment is to show some preliminary results that can be obtained with a typical Integrated wired-wireless experiment. We also show that if a VAP based approach is used for virtualization on the grid, there are concerns with the performance of one experiment affecting the other. We follow this analysis with our proposed solution to the interference problem.

Topology and setup: Figure 4.25 shows our experimental setup. It consists of three UDP-CBR traffic flows belonging to three independent experiments which are being sent by servers running on PlanetLab nodes to their respective clients running on the ORBIT grid. The forth flow is a video streaming from one of the Planetlab nodes to the clients running on the ORBIT grid. To setup this configuration of nodes an experiment script to similar to the one shown in Figure 2.5 was used.

Figure 4.26 shows a plot of the three UDP traffic flows as seen at the receiver on

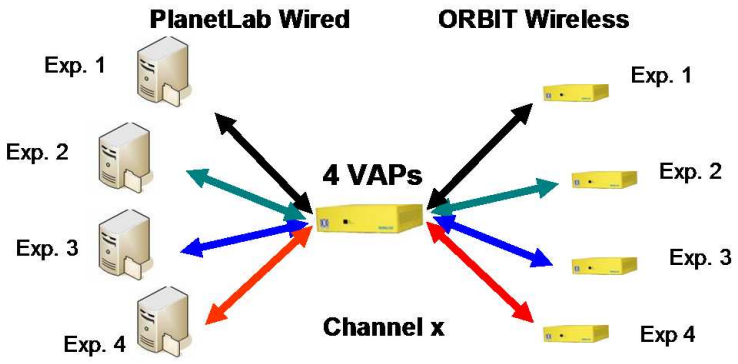


Figure 4.25: Experiment layout where nodes are added from PlanetLab while the VAP support from the 802.11 linux drivers is exploited for running multiple networks from a physical AP.

the wireless client node. The offered load for two of these experiments is increased as a function of time. As long as the aggregate offered load is below saturation, all three flows have a fair share of the throughput. Figure 4.29(a) shows the video seen at the client node based on the traffic of the forth experiment in the setup. It can be observed that since there are very few packet drops and low congestion in both the wired and wireless network, the video is clear.

As the offered load for each of the experiments is increased with time, the aggregate traffic on the wireless network reaches saturation. Figure 4.26 shows that in saturation the net throughput seen for the three flows is highly variant in time. Moreover, the picture in Figure 4.29(b) shows that the video suffers considerably when the wireless channel is in saturation. The increased distortion in the video quality may be attributed to the increased levels of jitter and dropped packets with the video flow. To prevent such situations where the performance of one experiment affects the other we introduce a policy manager to incorporate the use of traffic control with the experiments. In addition to traffic control, the policy manager performs two other functions:

- Admission control: To make a decision on allowing or denying an experiment to share a VAP with other experiments depending on its bandwidth requirements.
- Assigning Bandwidth: To allot the different experiments a maximum bandwidth value based on the number of experiments on a single VAP and their bandwidth requirements.

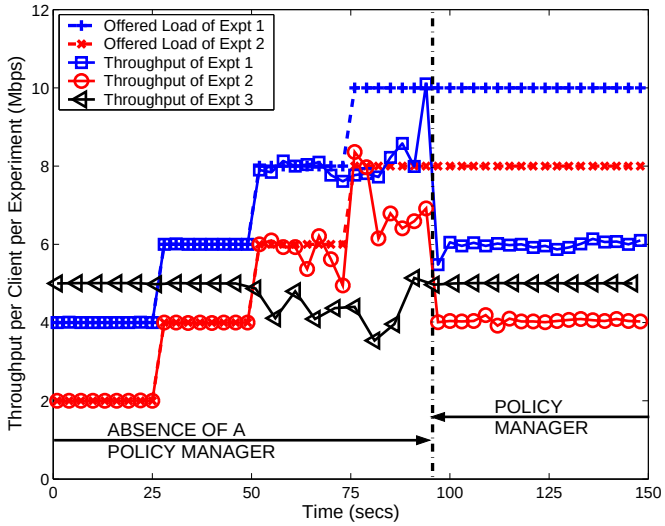


Figure 4.26: Throughput (Mbps) seen at the wireless VAP where manual intervention rate limits flows to stop channel saturation.

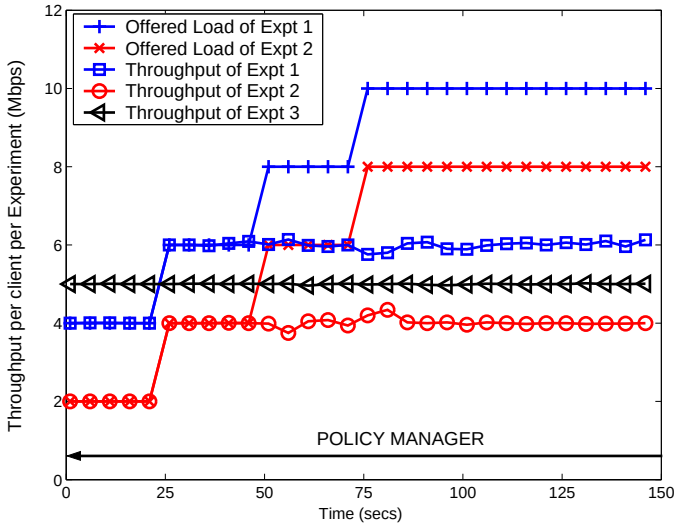


Figure 4.27: Throughput (Mbps) seen at the wireless VAP where individual flow rates are pre-decided by the policy manager. Channel never saturates.

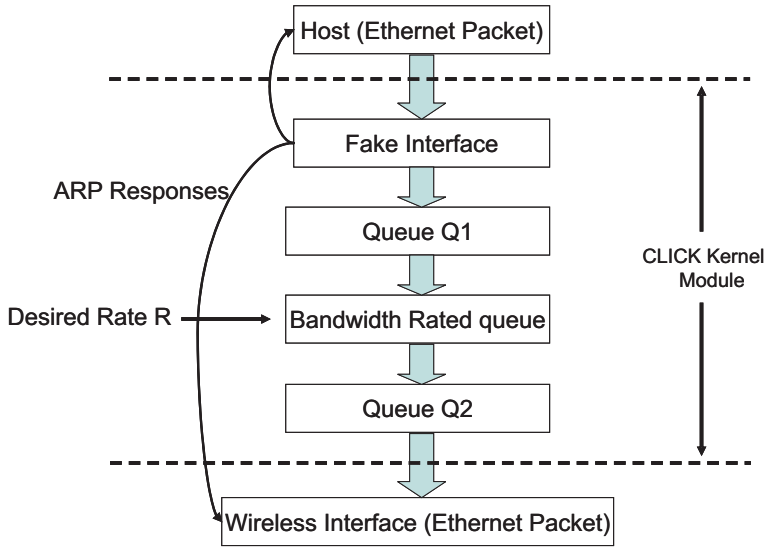


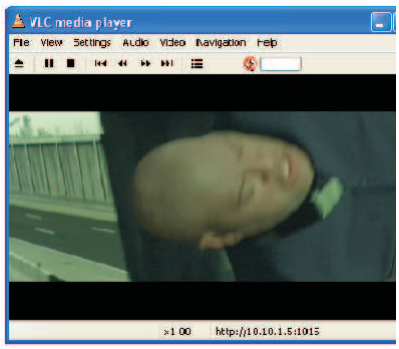
Figure 4.28: Click Modular Router Elements for Bandwidth Shaping.

The Policy managed could be integrated with the experiment scheduling and resource tracking mechanisms to ensure that each of the experiments get a fair share of the resources.

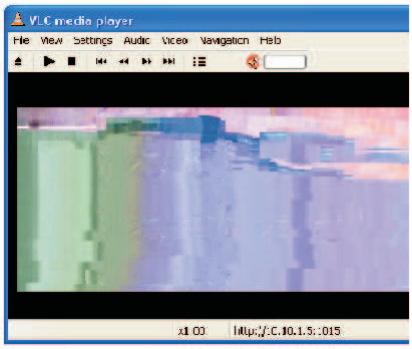
The Policy manager will work in one of the following modes to enforce traffic control among the different experiments sharing a common VAP:

- Dynamic intervention based control: Figure 4.26 shows a typical scenario with manual intervention in an experiment. Initially as the aggregate offered load is increased, the experiments are pushed into saturation. However, with manual intervention it is possible to rate limit the traffic flows. We make use of Click Modular Router [28] as a tool to implement bandwidth shaping. The CLICK elements used in our implementation for bandwidth shaping are shown in Figure 4.28. The CLICK configuration file with these elements defined was installed in the Linux kernel.

- Pre-Enforced based control: It is also possible to have a policy manager to limit the maximum share of throughput of the experiments, even before they are started. Figure 4.27 shows the results with a policy manager. Both the UDP experiments with high offered loads are rate limited to their assigned throughput values even before they reach channel saturation.



(a) Below saturation video at the start of the experiment



(b) The data flows saturate the channel, thereby resulting in deterioration of the video quality

(c) Video performance after traffic shaping with manual intervention. Availability of sufficient bandwidth restores the video quality

Figure 4.29: Video performance of experiment 4 as the other 3 experiments progress with increasing offered loads. Video is observed by forwarding the traffic from the client node to an observation node in the grid.

4.9 Measure to increase performance: MAC layer lumping

In virutalized approaches such as SDMA and VAP, the channel is multiplexed between the various experiments. There is a need to deploy measures that increase the bandwidth of the wireless link. One such feasible solution is lumping of MAC frames. This concept, implemented in the latest MADWIFI driver, follows the steps shown in Figure 4.30. IP packets received within certain threshold of inter-arrival time are transmitted as a single combined MAC frame. This phenomenon results in reduction in the number of MAC frames contending for the channel for the same offered load and increase in the size of the MAC frames reducing the protocol overhead. As seen from Figure 4.30, the overhead of lumping is an additional MAC header for the lumped MAC frame i.e., a MAC header for every two MAC frames.

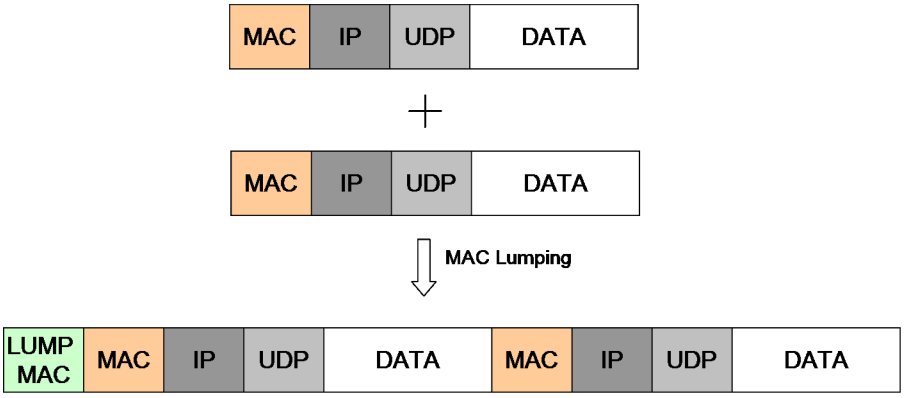


Figure 4.30: Implementation of MAC layer lumping in the Madwifi Driver.

We show the increase in performance with MAC lumping enabled using the VAP based virtualized network topology in Figure 4.8(b). Plots for aggregate throughput and number of packet transmissions with offered load for lumping enabled and disabled cases are shown in Figure 4.31. Lumping activates when the channel nears saturation ie. packet queueing at MAC layer triggers lumping. The line graph shows the increase in performance with lumping. With MAC lumping, the number of MAC frames transmitted decreases by half.

Variation of aggregate throughput with packet sizes for lumping as opposed to the

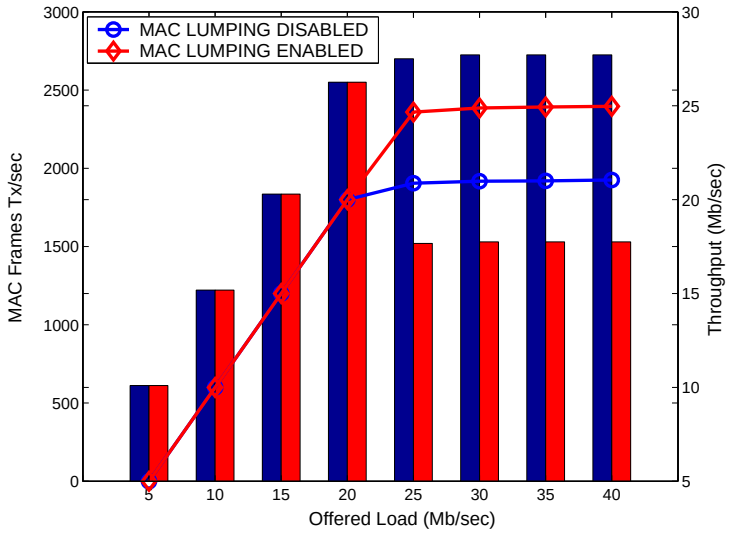


Figure 4.31: Performance Improvements with MAC layer Lumping for VAP based Virtualization.

normal scenario shown in Figure 4.32. It can be seen that the performance improvement by lumping increases with packet size and eventually the increase in performance becomes constant towards large packet sizes. For small packets, the lumping overhead occupies a higher percentage of the packet size. Hence, the improvement in performance with lumping as opposed to without lumping increases with packet size.

Figure 4.33 shows the channel utilization for 802.11a with MAC layer lumping. We show scenarios with PHY rate of 36Mb/sec and 54Mb/sec. It is clear from the figure that MAC lumping improves the channel utilization by reducing the percentage of the various protocol overheads. However, the MAC overhead increases for the case of lumping. However, the percentage of increase in MAC overhead is less than the percentage of increase in channel utilization for data transmission. As seen before, the lumping overhead is more prominent in smaller packet sizes and hence the effective increase in performance increases with packet sizes for MAC lumping.

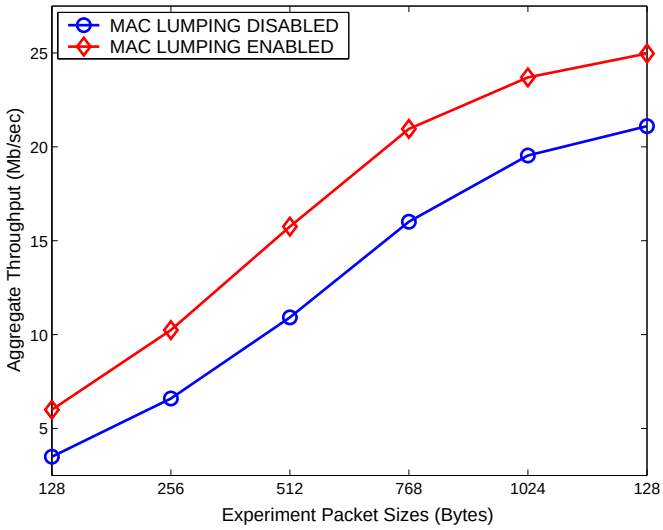


Figure 4.32: MAC layer lumping throughput improvements with packet sizes.

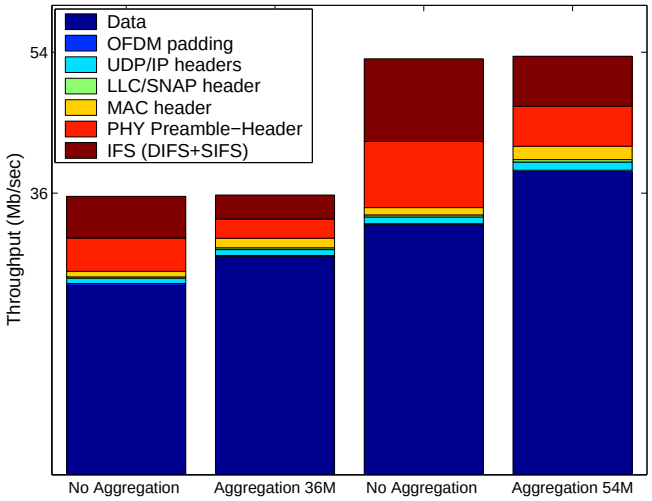


Figure 4.33: Channel Utilization improvement with MAC layer lumping.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

The unified designs presented in this thesis should serve as a practical foundation for wired/wireless integration in future heterogeneous testbeds. A common framework for control and management of heterogeneous testbed infrastructure will lead to easier and faster experimentation. Our proof of concept experiments demonstrate the effectiveness in terms of the ease of experimental deployment and the overall usefulness of this integrated framework.

In this work we have also discussed the challenges that would be encountered in designing Virtualization techniques for wireless virtualization for more efficient use of the testbed resources. We have suggested and evaluated feasibility of techniques that could facilitate virtualization for supporting concurrent experiments in terms of empirical results. However, the wireless experiments have a wide range of requirements and may not be supported by all types of virtualization. We have listed some experiments based on their requirements for certain experimental metrics. Figure 5.1 lists these requirements along with the methods of virtualization that can support the respective experiment constraint as a part of our qualitative comparison of SDMA and VAP-based approaches.

Every experiment would have a stringent requirement on either throughput/packet loss or delay/jitter bounds. In addition some experiments may have requirements related to channel assignment and mobility-handoffs. The conclusions are based on the experimental results shown in the previous sections. We list some of the important points where SDMA and FDMA slicing scores over VAP based virtualization scheme:

1. Resource Control (Rate, Power) Power adjustments may be a problem if SDMA is provided using schemes like artificial stretching as described previously.

2. Admission control and Qos control such as adjusting MAC retransmission etc.

3. Adjusting MAC level parameters like CSMA threshold, disabling ACKs etc.

4. Experiments involving frequency control cannot be carried on FDMA slices.

5. Cross Layer Experimentation involving MAC layers can be performed in SDMA and FDMA slices. In SDMA and FDMA slices, the experiment nodes are not shared between experiments. However, in the case of VAP, the MAC layer stack of the Access Point is shared between experiments.

Interestingly, additional support would be required for the VAP to provide different wireless statistics for its virtual interfaces to the higher layers to facilitate cross layer experimentation. This is important if a cross layer experiment is using one of the Virtual AP interface. Currently, statistics like RSSI, MAC retires etc. are reported for the underlying physical interface and no distinct information for the virtual interfaces are reported. Cross Layer protocols relying on MAC and physical layer information would fail to run on the VAP without this support.

We have discussed various virtualization schemes and their limitations in regard to experiment requirements. We now present some detail on scalability issues for the schemes:

FDMA

1. It is limited by the number of orthogonal frequencies (3 for 802.11b and 12 for 802.11 a/g).

2. Switching time between frequencies. This can be avoided by use of multiple cards. However the latter approach is subject to co-channel interference.

SDMA

1. Space Constraints

2. Number of nodes in each partition/slice.

3. If making use of Artificial stretching: Granularity of control on range of Noise sources and transmission power of nodes.

VAP

1. Virtualization scheme only applies to Access points hence restricted to fixed star topology.

2. Limited by number of concurrent experiments. Present facility supports four experiments per VAP.

5.2 Future Work

We have discussed a few techniques towards wireless virtualization of emulators and classified them with respect to the experiments needs. As a result, testbeds should employ scheduling algorithms that operate different spatial slices in different modes, such that the current experiment mix is efficiently distributed across the testbed. Hence work is needed in direction of design and implementation of resource manager.

Experiment requirements	*FDMA based Slicing*	*SDMA based Slicing*	*VAP based Virtualization*
MAC experiments	Yes (no frequency control)	Yes	Yes but Cross-Layer experiments may require extra support.
Very small delay/jitter bounds(<10ms)	Yes	Possible depending on traffic on other experiments.	Possible depending on traffic on other experiments.
Small to medium delay/jitter bounds(100ms)	Yes	Yes depending on traffic on other experiments.	Yes depending on traffic on other experiments.
High Throughput (*Loaded*)	Yes	Yes depending on the desired through-put and no. of experiments sharing the resource.	Yes depending on the desired through-put and no. of experiments sharing the resource. Traffic Shaping scheme guarantees bandwidth.
Low Packet Losses	Yes	Yes	Yes
Experiment Topology	No restriction	No restriction but requires a complicated algorithm to incorporate arbitrary topologies in the slice allocated to the different experiments due to spatial constraints in the testbed	Fixed Star Topology.
Channel Switching	Restricted depending on availability of channel	Yes	No, the channel is shared by other experiments on the same VAP.
Mobility and hand off	Yes	Yes	Yes

Figure 5.1: Mapping Types of experiments to the various Virtualization techniques.

References

[1] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh, "Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols," IEEE Wireless Communications and Networking Conference, March 2005. [Online]. Available: http://www.orbit-lab.org/download/publications/Orbit_Overview.pdf

[2] L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, B. White, J. Lepreau, and A. Joglekar, "An integrated experimental environment for distributed systems and networks." Proceedings of the 4th Symposium on Operating System Design and Implementation OSDI, 2002.

[3] "An integrated experimental environment for distributed systems and networks," 2002. [Online]. Available: http://www.isi.edu/deter

[4] A. Bevier, M. Bowman, B. Chun, D. Culler, S. Karlin, L. Peterson, T. Roscoe, T. Spalink, and M. Wawroniak, "An open platform for developing, deploying and accessing planetary-scale services." [Online]. Available: https://www.planet-lab.org/

[5] "Technical document on wireless virtualization," *Wireless Working Group-GENI*, September 2006. [Online]. Available: www.geni.net/GDD/GDD-06-17.pdf

[6] "Us nsf - national science foundation." [Online]. Available: http://www.nsf.gov/

[7] "Geni design principles." [Online]. Available: http://www.geni.net/design principles.pdf

[8] M. Singh, I. Seskar, M. Ott, P. Kamat, and M. Ott, "Orbit measurements framework and library (oml): Motivations, design,implementation, and features." IEEE Tridentcom, Feb 2005.

[9] A. Bevier, M. Bowman, B. Chun, D. Culler, S. Karlin, L. Peterson, T. Roscoe, T. Spalink, and M. Wawroniak, "Operating system support for planetary-scale network services," In proceedings of the NSDI., San Francisco, California, 2004. [Online]. Available: http://www.cs.princeton.edu/ acb/nsdi04/paper.pdf

[10] M. Huang, "Vnet: Planetlab virtualized network access." [Online]. Available: http://www.planet-lab.org/PDN/PDN-05-029

[11] B. Chun, "Pssh/pscp tool." [Online]. Available: http://www.theether.org/

[12] "S-cube project webpage," HP. [Online]. Available: http://networking.hpl.hp.com/s-cube/

[13] "Yum tool," HP. [Online]. Available: http://linux.duke.edu/projects/yum/

[14] D. Raychaudhuri and M. Gerla, "New architectures and disruptive technologies for the future internet: The wireless, mobile and sensor network perspective." Report of NSF Wireless Mobile Planning Group (WMPG) Workshop, August 2005.

[15] B. Aboba, "Virtual access points," IEEE document 802.11-03/154r1. [Online]. Available: http://www.drizzle.com/ aboba/IEEE/11-03-154r1-I-Virtual-Access-Points.doc

[16] J. Dike, "A user-mode port of the linux kernel," 5th Annual Linux Showcase and Conference, Oakland, California, 2001.

[17] S. Banerji, "Time based virtualization of an 802.11-based wireless facility." [Online]. Available: www.geni.net/docs/banerji.pdf

[18] "Ntpd: Network time protocol." [Online]. Available: http://dast.nlanr.net/Projects/Iperf/

[19] *Supplement to IEEE Standard for Information technology Telecommunications and information exchange between systems Local and metropolitan area networks Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: High-speed Physical Layer in the 5 GHZ Band*, IEEE-SA Standards Board IEEE Std 802.11a-1999 (Supplement to IEEE Std 802.11-1999), September 1999.

[20] "Iperf 1.7.0: The tcp/udp bandwidth measurement tool." [Online]. Available: http://www.ntp.org/

[21] "Ping tool." [Online]. Available: http://linux.die.net/man/8/ping

[22] "Videolan vlc media player." [Online]. Available: http://www.videolan.org/

[23] "Manpage of tcpdump tool." [Online]. Available: http://www.tcpdump.org/

[24] "Ethereal tool." [Online]. Available: http://www.ethereal.com/

[25] "Madwifi driver." [Online]. Available: www.madwifi.org

[26] "Creating virtual ap on madwifi," IEEE document 802.11-03/154r1. [Online]. Available: http://madwifi.org/wiki/UserDocs/MultipleInterfaces

[27] S. Ganu, K. Ramachandran, M. Gruteser, and I. Seskar, "Methods for restoring mac layer fairness in ieee 802.11 networks with physical layer capture," 5th Annual Linux Showcase and Conference. REALMAN, in conjunction with Mobihoc, 2006.

[28] "Click modular router," HP. [Online]. Available: http://read.cs.ucla.edu/click/

[29] C. Doerr, A. Sheth, M. Neufeld, J. Fifield, and D. Grunwald, "Softmac: Flexible wireless research platform." Hot Topics in Networking (HotNets-IV), 2005.

[30] D. Culler, L. Peterson, T. Anderson, and T. Roscoe, "A blueprint for introducing disruptive technology into the internet." First Workshop on Hot Topics in Networking (HotNets-I), 2002.

[31] "Wisconsin advanced internet laboratory." [Online]. Available: http://wail.cs.wisc.edu

[32] "Implementing the emulab-planetlab portal: Experiences and lessons learned." Proceedings of the First Workshop on Real, Large Distributed Systems, 2004.

[33] A. Bavier, N. Feamster, M. Huang, L. Peterson, and J. Rexford, "In vini veritas: Realistic and controlled network experimentation," In ACM SIGCOMM, vol. 36, no. 4, San Francisco, California, October 2006, pp. 3–14. [Online]. Available: http://www.vini-veritas.net/papers

[34] Y. Koh, C. Pu, S. Bhatia, and C. Consel, "Efficient packet processing in user-level operating systems: A study of uml," *Proceedings of the 31st IEEE Conference on Local Computer Networks LCN* , 2006.

[35] E. Bugnion, S. Devine, and M. Rosenblum, "Virtualization system including a virtual machine monitor for a computer with a segmented architecture," in US Patent, 6397242, 1998. [Online]. Available: http://www.theether.org/

[36] B. D. K. Fraser S. Hand T. Harris A. Ho R. Neugebauer I. Pratt P. Barham and A. Warfield, "Xen and the art of virtualization," SOSP, 2003.

[37] "Generic routing encapsulation over clns networks," RFC3147 IETF draft of the networking working group.