Ανάπτυξη εγχειριδίου χρήσης της πειραματικής υποδομής NITOS

Σίμος Ιορδανίδης

Πανεπιστήμιο Θεσσαλίας
Τμήμα Μηχανικών Η/Υ Τηλεπικοινωνιών και Δικτύων

Στόχος

- Ο χρήστης της πειραματικής υποδομής του NITOS να μπορεί μέσω απλών βημάτων να μάθει πως να χρησιμοποιεί αυτήν και βασικές τεχνολογίες **(WiFi-WiMax-Openflow)**

Πώς υλοποιήθηκε ;

- **<u>Χρήση δυο εργαλείων:</u>**

➔ Sphinx

➔ Github

- Το Sphinx είναι ένα εργαλέιο που μας επιτρέπει εύκολα και έξυπνα να κάνουμε documentation
- Λειτουργεί με απλό κώδικα σε text files
- Μεταφράζει το σύνολο των text files σε διάφορα formats(HTML,LaTex,PDF)

Πλεονεκτήματα

- Απλό στην εκμάθηση και χρήση.
- Δεν χρειάζεται να ασχοληθούμε εξ'ολοκλήρου με HTML/CSS αφού δημιουργούνται αυτόματα.
- Παρέχει πολλά HTML Themes.
- ➔ Στο συγκεκριμένο εγχειρίδιο έγινε χρήση του default theme.

Παράδειγμα

```
.. test_nit documentation master file, created by
   sphinx-quickstart on Sun Apr  6 12:53:31 2014.
   You can adapt this file completely to your liking, but it should at least
   contain the root `toctree` directive.


.. image:: _static/header-bg.png
   :width: 30%
   :height: 100px
.. image:: _static/NITlab_logo_layers.png
   :width: 30%
   :height: 100px
.. image:: _static/header-bg.png
   :width: 30%
   :height: 100px
```

NITlab documentation
====================================

First Steps on using NITOS

.. toctree::
   :maxdepth: 2

   account_create
   get_reserve
   connectNITOS
   loadOMF
   runOMF
   control_Nodes_Status
   save_OMFimage
   video_tutorials

OMF Example

.. toctree::
   :maxdepth: 2

   omf_example

Using Wifi

.. toctree::
   :maxdepth: 2

   wireless_example

Using WiMax

.. toctree::
   :maxdepth: 2

   wiMax

Using Openflow

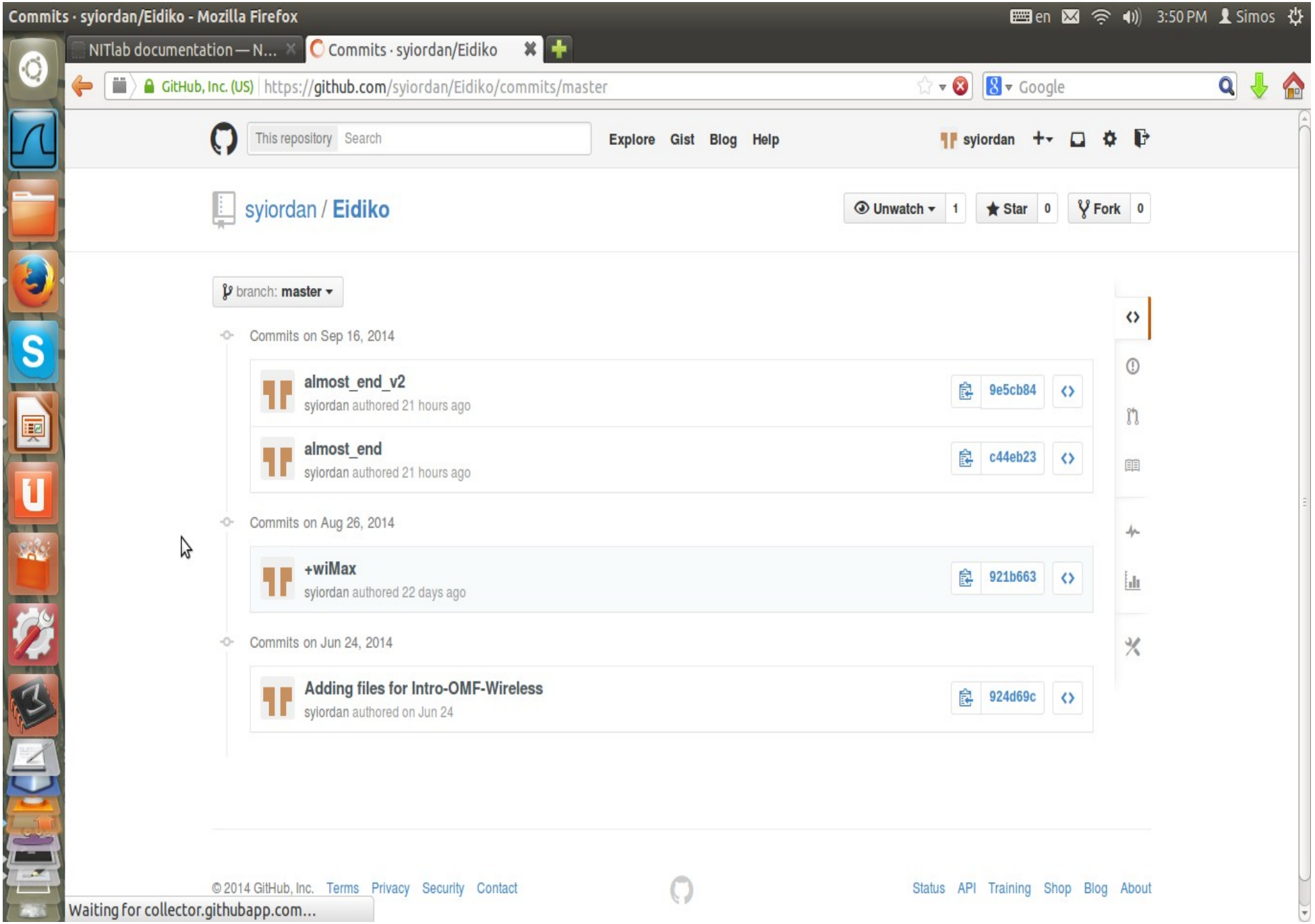
.. toctree::
   :maxdepth: 2

   Openflow

- Χρησιμοποιώντας την εντολή **$make html** στο τερματικό μας....έχουμε έτοιμο το html αρχείο.

GitHub

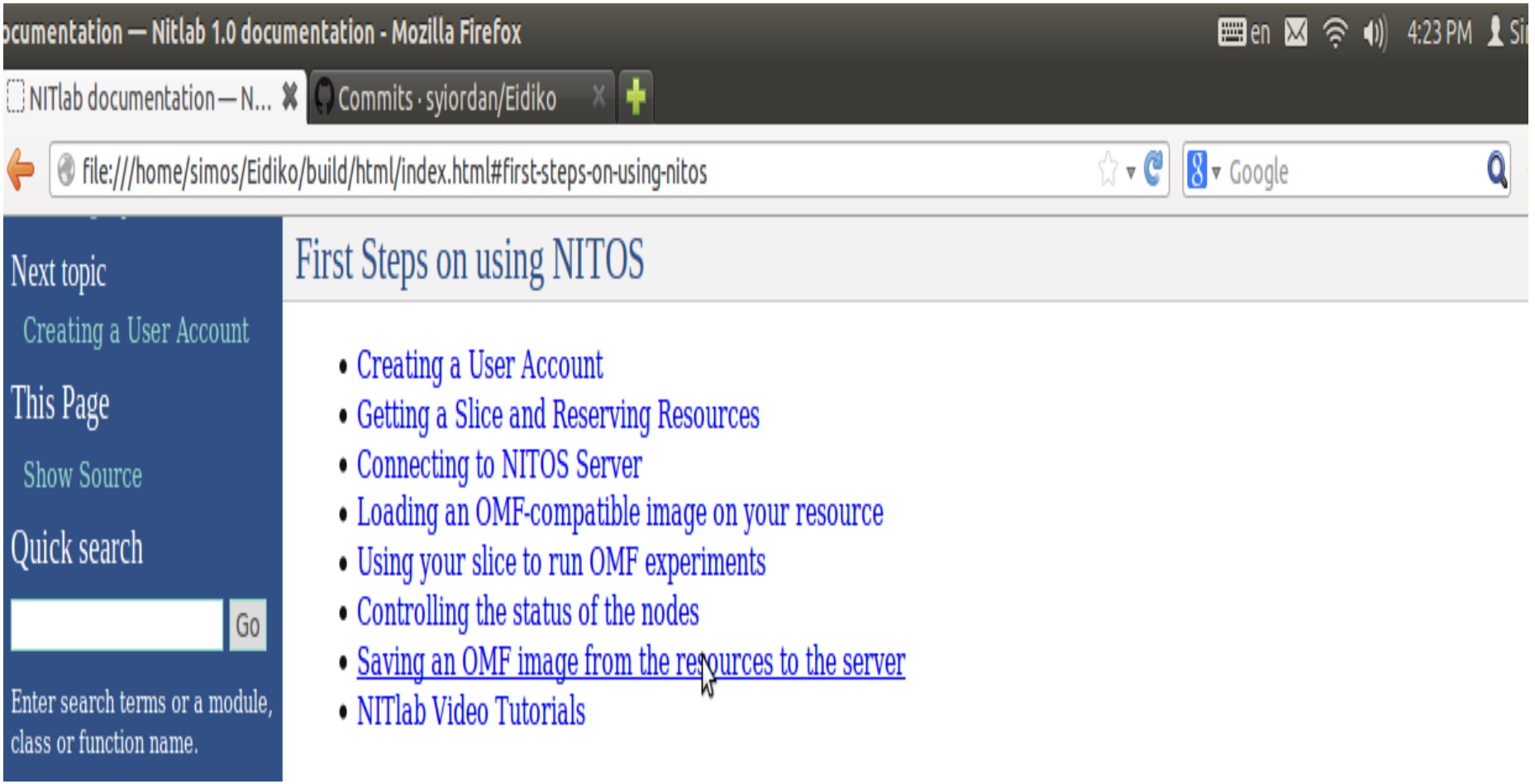
- Social repository για projects ανοιχτού κώδικα που βασίζονται πάνω στο Git version control system.
- Εύκολη διαμοίραση κώδικα και συνεργασία σε projects.
- Χρησιμοποιήθηκε για versioning στο documentation.


Commits · syiordan/Eidiko - Mozilla Firefox
en
3:50 PM
Simos
NITlab documentation — N...
Commits · syiordan/Eidiko
GitHub, Inc. (US)
https://github.com/syiordan/Eidiko/commits/master
Google
This repository
Search
Explore
Gist
Blog
Help
syiordan
syiordan / Eidiko
Unwatch
1
Star
0
Fork
0
branch: master
Commits on Sep 16, 2014
almost_end_v2
syiordan authored 21 hours ago
9e5cb84
almost_end
syiordan authored 21 hours ago
c44eb23
Commits on Aug 26, 2014
+wiMax
syiordan authored 22 days ago
921b663
Commits on Jun 24, 2014
Adding files for Intro-OMF-Wireless
syiordan authored on Jun 24
924d69c
© 2014 GitHub, Inc.
Terms
Privacy
Security
Contact
Status
API
Training
Shop
Blog
About
Waiting for collector.githubapp.com...

Δομή Εγχειριδίου

- Χωρίζεται σε 5 τμήματα:
  - **First steps on using Nitos**
  - **OMF Example**
  - **Using WiFi**
  - **Using WiMax**
  - **Using Openflow**

First Steps on using Nitos



- Στα συγκεκριμένα sections περιγράφεται αναλυτικά πως ο χρηστής μπορεί να μάθει βασικά στοιχεία για την χρήση του testbed όπως να :

- Δημιουργήσει λογαριασμό στο NitLab
- Δεσμέυσει slice και nodes
- Συνδεθεί στον Server
- Φορτώνει OMF-images στους κόμβους και να τρεχει OMF πειράματα.

Creating a User Account — Nitlab 1.0 documentation - Mozilla Firefox

Creating a User Account — ... | Commits · syiordan/Eidiko

file:///home/simos/Eidiko/build/html/account_create.html

Previous topic
NITlab documentation
Next topic
Getting a Slice and Reserving Resources
This Page
Show Source
Quick search
Go
Enter search terms or a module, class or function name.

Creating a User Account

The first step you need to do before you can use NITOS is to create your NITlab account. This process is simple. Underneath the login boxes, click the **"Create an account"** link. Fill in the required information. Once you click the **"Register"** button, a confirmation mail will come to you and when you confirm it, your request will be sent to the administrator for approval. When the administrator approves the creation of your account, you will receive a confirmation email.

Example:

1. Visit http://nitlab.inf.uth.gr/NITlab/
2. Click "Create an account"
3. **Name**: Simos Iordanidis
   Username: syiordan
   Password: 1234
   Confirm Password: 1234
   Email Address: syiordan@inf.uth.gr
   Confirm email Adress: syiordan@inf.uth.gr

   Click Register
4. Confirm the mail you've received from administrator. The administrator will approve the creation of your account.

You are ready!You are a member of NITOS.At this time, you may continue with the next topic.

Getting a Slice and Reserving Resources — Nitlab 1.0 documentation - Mozilla Firefox

Getting a Slice and Reservin... | Commits · syiordan/Eidiko

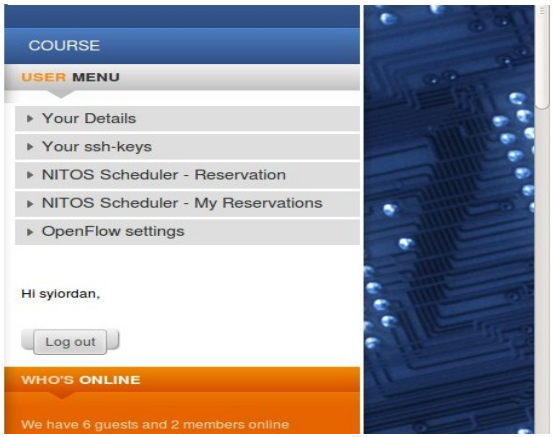
file:///home/simos/Eidiko/build/html/get_reserve.html

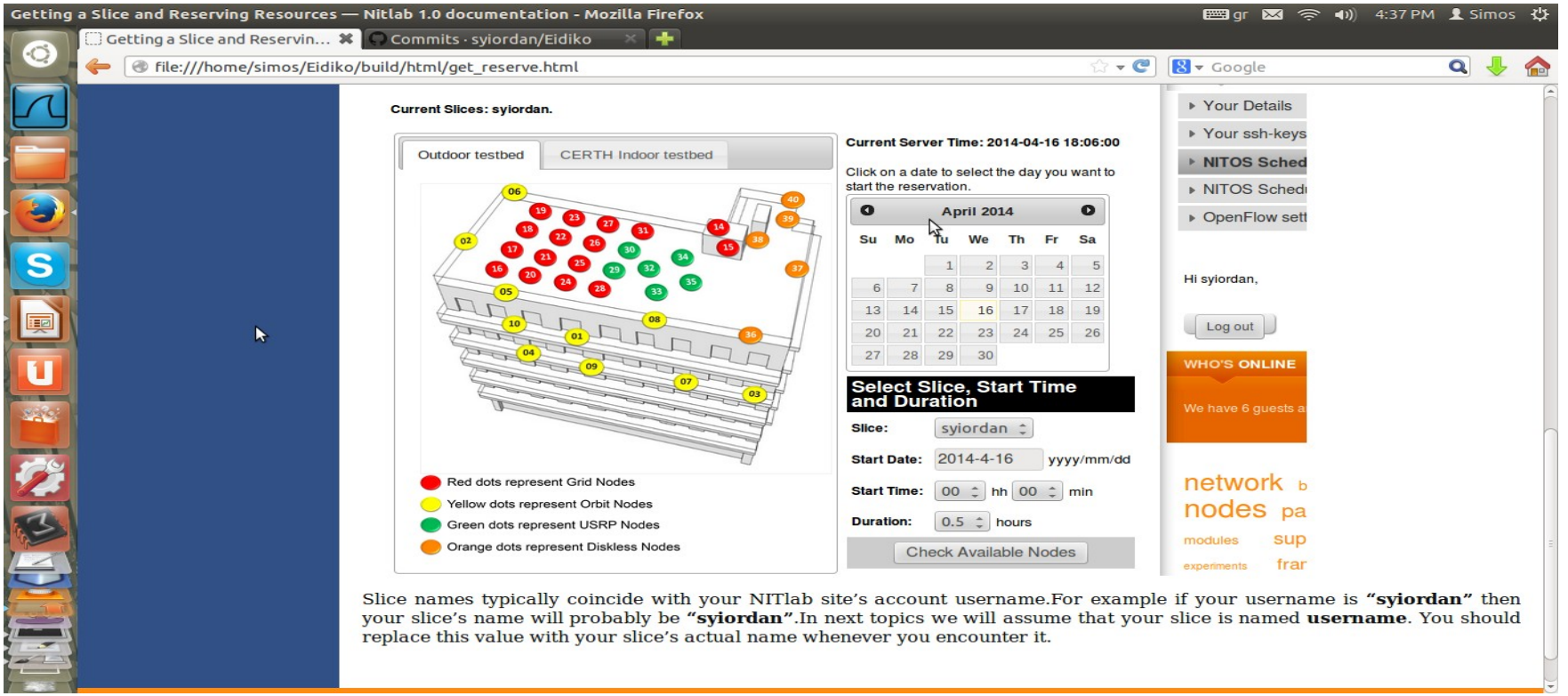
Previous topic
Creating a User Account
Next topic
Connecting to NITOS Server
This Page
Show Source
Quick search
Go
Enter search terms or a module, class or function name.

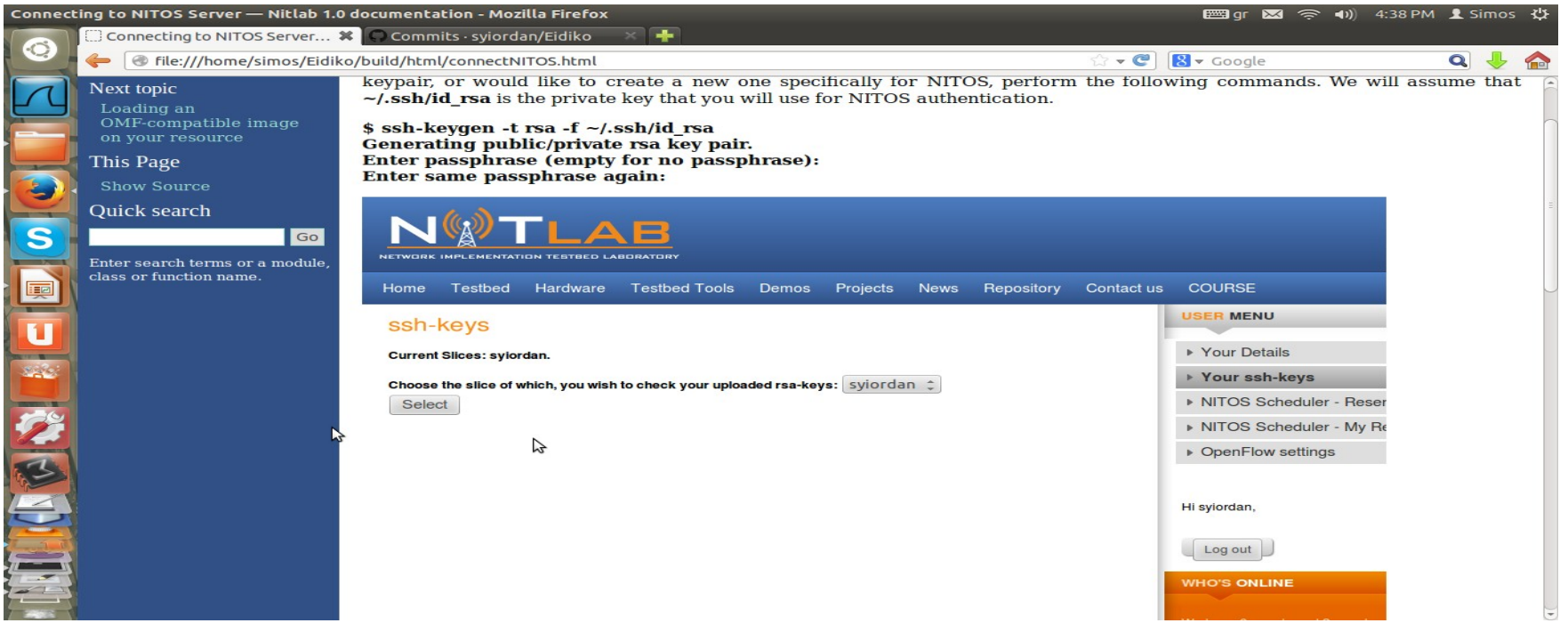
Getting a Slice and Reserving Resources

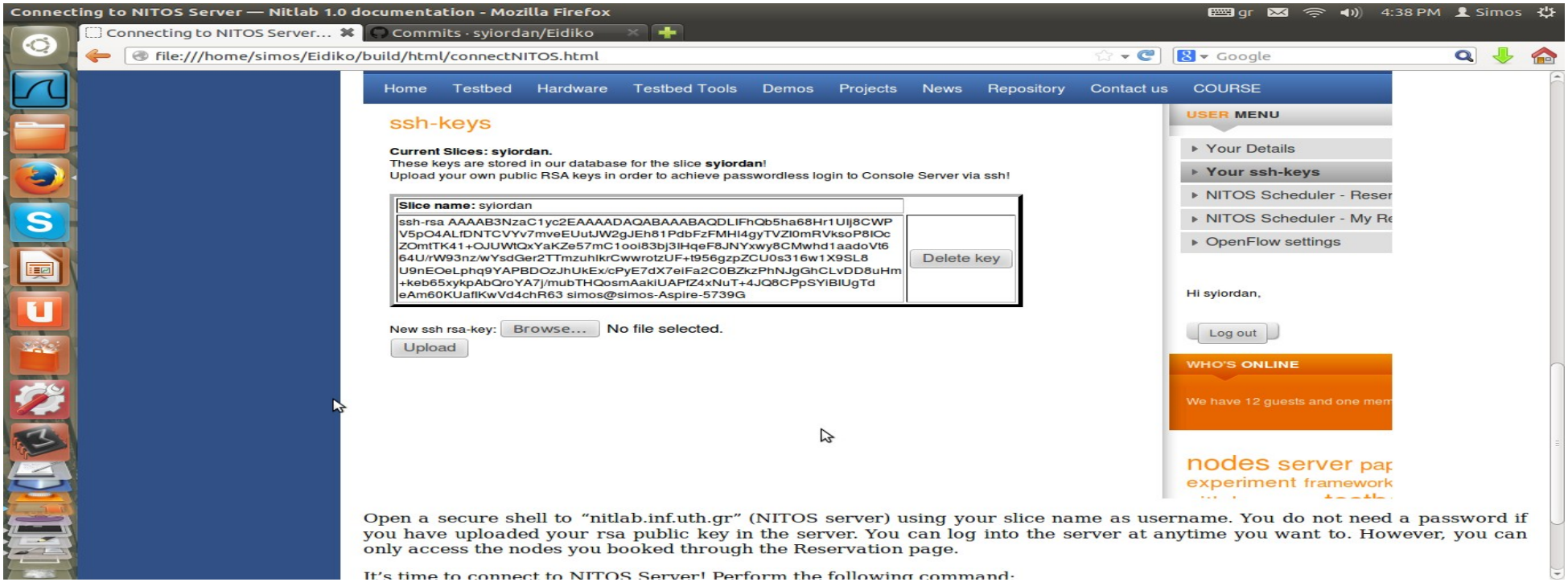
All access to NITOS resources occurs through the "slice" abstraction.

The NITlab Administrator is in charge of managing your slice. You will need to contact him directly and ask him/her to either create a new slice for you, or add your user account to an existing slice.Once your account has been associated to a slice, you will be able to view the **Reservation page** and the **"Your ssh-keys"** page, at the User Menu on the right of the page.



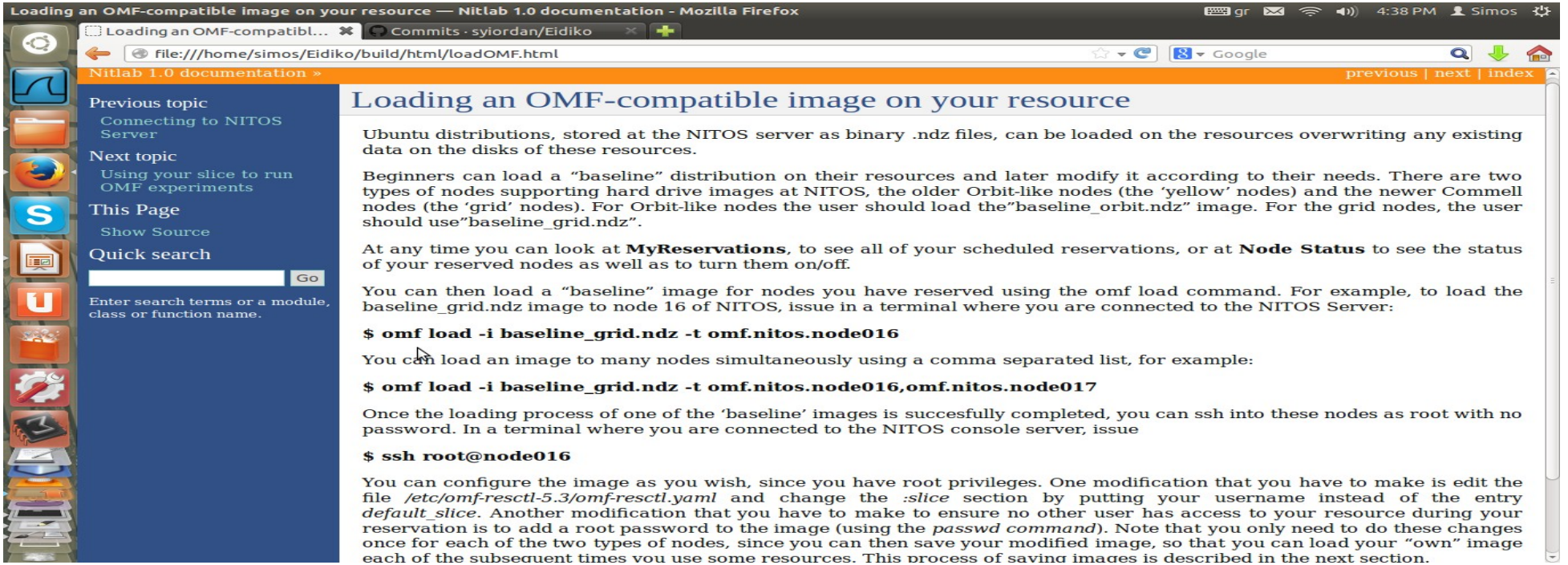

Getting a Slice and Reserving Resources — Nitlab 1.0 documentation - Mozilla Firefox
Getting a Slice and Reservin...
Commits · syiordan/Eidiko
file:///home/simos/Eidiko/build/html/get_reserve.html
Current Slices: syiordan.
Outdoor testbed
CERTH Indoor testbed
Red dots represent Grid Nodes
Yellow dots represent Orbit Nodes
Green dots represent USRP Nodes
Orange dots represent Diskless Nodes
Current Server Time: 2014-04-16 18:06:00
Click on a date to select the day you want to start the reservation.
April 2014
Select Slice, Start Time and Duration
Slice: syiordan
Start Date: 2014-4-16 yyyy/mm/dd
Start Time: 00 hh 00 min
Duration: 0.5 hours
Check Available Nodes
Your Details
Your ssh-keys
NITOS Sched
NITOS Sched
OpenFlow sett
Hi syiordan,
Log out
WHO'S ONLINE
We have 6 guests a
Slice names typically coincide with your NITlab site's account username.For example if your username is "syiordan" then your slice's name will probably be "syiordan".In next topics we will assume that your slice is named username. You should replace this value with your slice's actual name whenever you encounter it.


Connecting to NITOS Server — Nitlab 1.0 documentation - Mozilla Firefox
Connecting to NITOS Server...
Commits · syiordan/Eidiko
file:///home/simos/Eidiko/build/html/connectNITOS.html
Next topic
Loading an OMF-compatible image on your resource
This Page
Show Source
Quick search
Go
Enter search terms or a module, class or function name.
keypair, or would like to create a new one specifically for NITOS, perform the following commands. We will assume that ~/.ssh/id_rsa is the private key that you will use for NITOS authentication.
$ ssh-keygen -t rsa -f ~/.ssh/id_rsa
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
NITLAB
NETWORK IMPLEMENTATION TESTBED LABORATORY
Home Testbed Hardware Testbed Tools Demos Projects News Repository Contact us COURSE
ssh-keys
Current Slices: syiordan.
Choose the slice of which, you wish to check your uploaded rsa-keys: syiordan
Select
USER MENU
Your Details
Your ssh-keys
NITOS Scheduler - Reser
NITOS Scheduler - My Re
OpenFlow settings
Hi syiordan,
Log out
WHO'S ONLINE



Open a secure shell to "nitlab.inf.uth.gr" (NITOS server) using your slice name as username. You do not need a password if you have uploaded your rsa public key in the server. You can log into the server at anytime you want to. However, you can only access the nodes you booked through the Reservation page.

It's time to connect to NITOS Server! Perform the following command:



Ubuntu distributions, stored at the NITOS server as binary .ndz files, can be loaded on the resources overwriting any existing data on the disks of these resources.

Beginners can load a "baseline" distribution on their resources and later modify it according to their needs. There are two types of nodes supporting hard drive images at NITOS, the older Orbit-like nodes (the 'yellow' nodes) and the newer Commell nodes (the 'grid' nodes). For Orbit-like nodes the user should load the"baseline_orbit.ndz" image. For the grid nodes, the user should use"baseline_grid.ndz".

At any time you can look at **MyReservations**, to see all of your scheduled reservations, or at **Node Status** to see the status of your reserved nodes as well as to turn them on/off.

You can then load a "baseline" image for nodes you have reserved using the omf load command. For example, to load the baseline_grid.ndz image to node 16 of NITOS, issue in a terminal where you are connected to the NITOS Server:

$ omf load -i baseline_grid.ndz -t omf.nitos.node016

You can load an image to many nodes simultaneously using a comma separated list, for example:

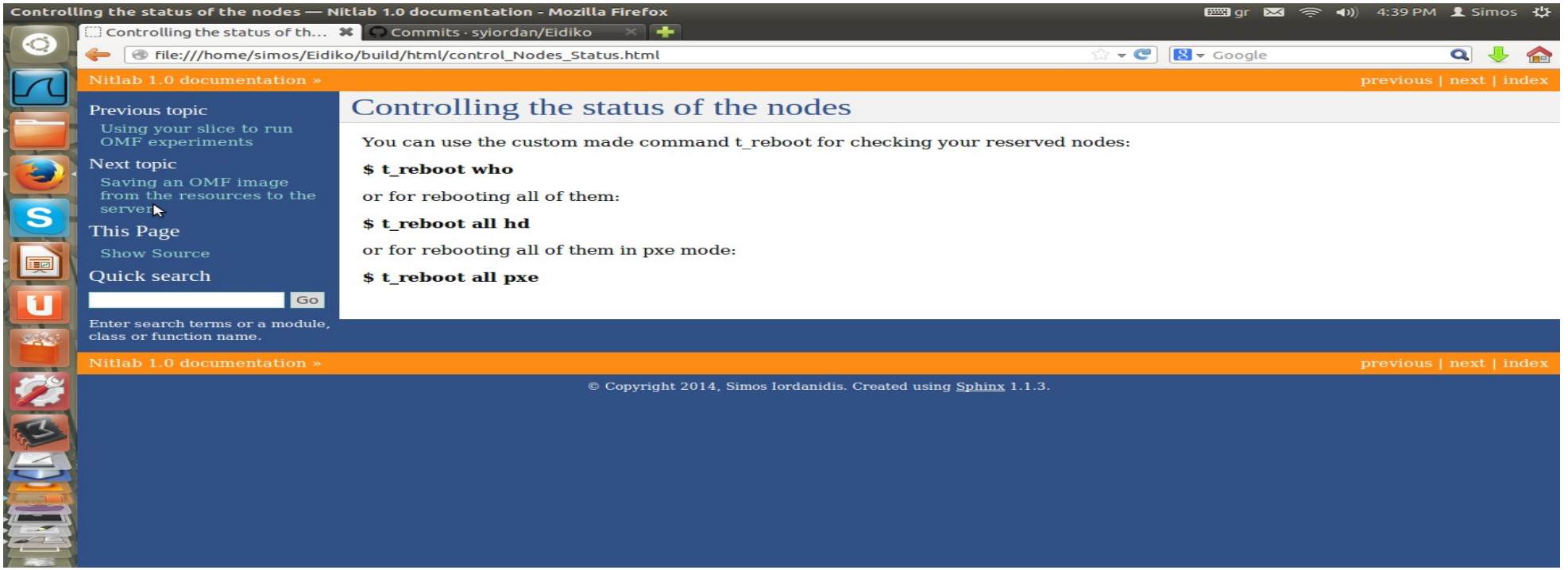
$ omf load -i baseline_grid.ndz -t omf.nitos.node016,omf.nitos.node017

Once the loading process of one of the 'baseline' images is succesfully completed, you can ssh into these nodes as root with no password. In a terminal where you are connected to the NITOS console server, issue

$ ssh root@node016

You can configure the image as you wish, since you have root privileges. One modification that you have to make is edit the file */etc/omf-resctl-5.3/omf-resctl.yaml* and change the *:slice* section by putting your username instead of the entry *default_slice*. Another modification that you have to make to ensure no other user has access to your resource during your reservation is to add a root password to the image (using the *passwd command*). Note that you only need to do these changes once for each of the two types of nodes, since you can then save your modified image, so that you can load your "own" image each of the subsequent times you use some resources. This process of saving images is described in the next section.


Using your slice to run OMF experiments — Nitlab 1.0 documentation - Mozilla Firefox
Using your slice to run OMF ...
Commits · syiordan/Eidiko
file:///home/simos/Eidiko/build/html/runOMF.html
Google
Nitlab 1.0 documentation »
previous | next | index
Previous topic
Loading an OMF-compatible image on your resource
Next topic
Controlling the status of the nodes
This Page
Show Source
Quick search
Enter search terms or a module, class or function name.
Using your slice to run OMF experiments
There are two ways to use your slice in order to run OMF experiments during your reservation. The first way is to install the OMF Experiment Controller locally at your PC and configure it appropriately and the second way is to issue OMF commands when logged in the server giving your slice as an option. We strongly recommend the second option, since every necessary compoment is in place in our Server, so we describe just the second method below. Running OMF Experiment Controller from the server
You can run OMF experiments while logged in the NITOS console server with your account. All you have to do is use the –slice username option when running the omf exec command (replacing username with your actual username).
Nitlab 1.0 documentation »
previous | next | index
© Copyright 2014, Simos Iordanidis. Created using Sphinx 1.1.3.


Controlling the status of the nodes — Nitlab 1.0 documentation - Mozilla Firefox
Controlling the status of th...
Commits · syiordan/Eidiko
file:///home/simos/Eidiko/build/html/control_Nodes_Status.html
Google
Nitlab 1.0 documentation »
previous | next | index
Previous topic
Using your slice to run OMF experiments
Next topic
Saving an OMF image from the resources to the server
This Page
Show Source
Quick search
Go
Enter search terms or a module, class or function name.
Controlling the status of the nodes
You can use the custom made command t_reboot for checking your reserved nodes:
$ t_reboot who
or for rebooting all of them:
$ t_reboot all hd
or for rebooting all of them in pxe mode:
$ t_reboot all pxe
Nitlab 1.0 documentation »
previous | next | index
© Copyright 2014, Simos Iordanidis. Created using Sphinx 1.1.3.

Saving an OMF image from the resources to the server — Nitlab 1.0 documentation - Mozilla Firefox

Saving an OMF image from ... | Commits · syiordan/Eidiko

file:///home/simos/Eidiko/build/html/save_OMFimage.html

Previous topic
Controlling the status of the nodes
Next topic
NITlab Video Tutorials
This Page
Show Source
Quick search
Go
Enter search terms or a module, class or function name.

Saving an OMF image from the resources to the server

You can save your modified image, in order to maintain the changes you've made and be able to use it again issuing the omf save command. For example, issuing

$ omf save -n omf.nitos.node005

will save the current image of node 5 to the server. The image is saved as an *.ndz* file in the directory */var/lib/omf-images-5.4* of the NITOS server with a name containing the date and time of the `*omf save*` command. You can then load this image to your resources using the omf load command and the name of the image. You can also rename the image, so that its name is more meaningful to you.

NITlab Video Tutorials — Nitlab 1.0 documentation - Mozilla Firefox

NITlab Video Tutorials — N... | Commits · syiordan/Eidiko

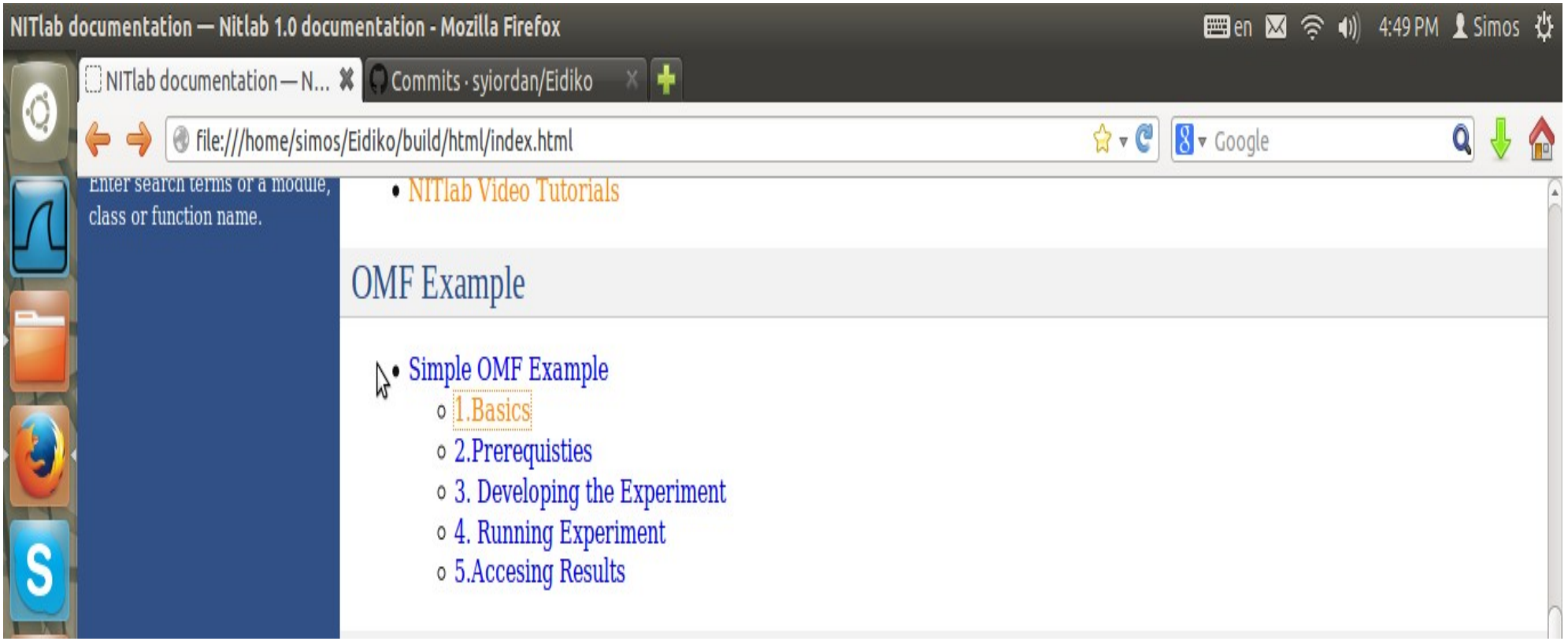
file:///home/simos/Eidiko/build/html/video_tutorials.html

Previous topic
Saving an OMF image from the resources to the server
Next topic
Simple OMF Example
This Page
Show Source
Quick search
Go
Enter search terms or a module, class or function name.

NITlab Video Tutorials

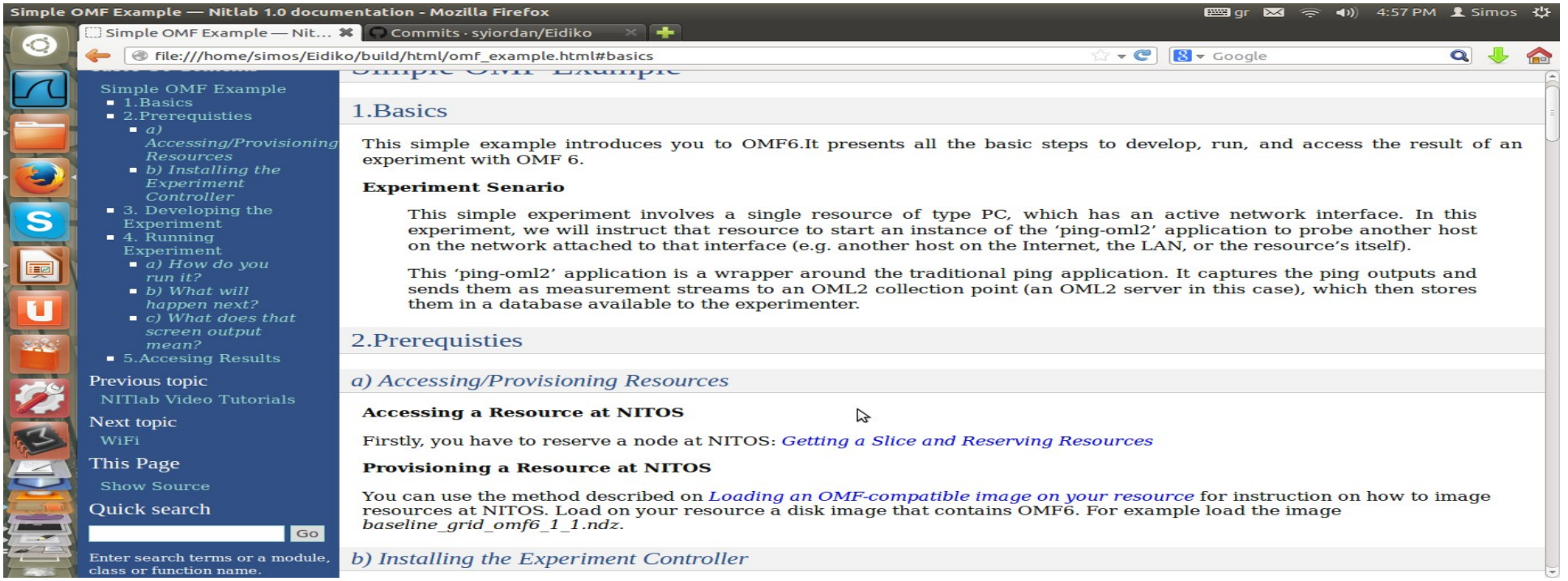
Video1: How to register at NITOS testbed

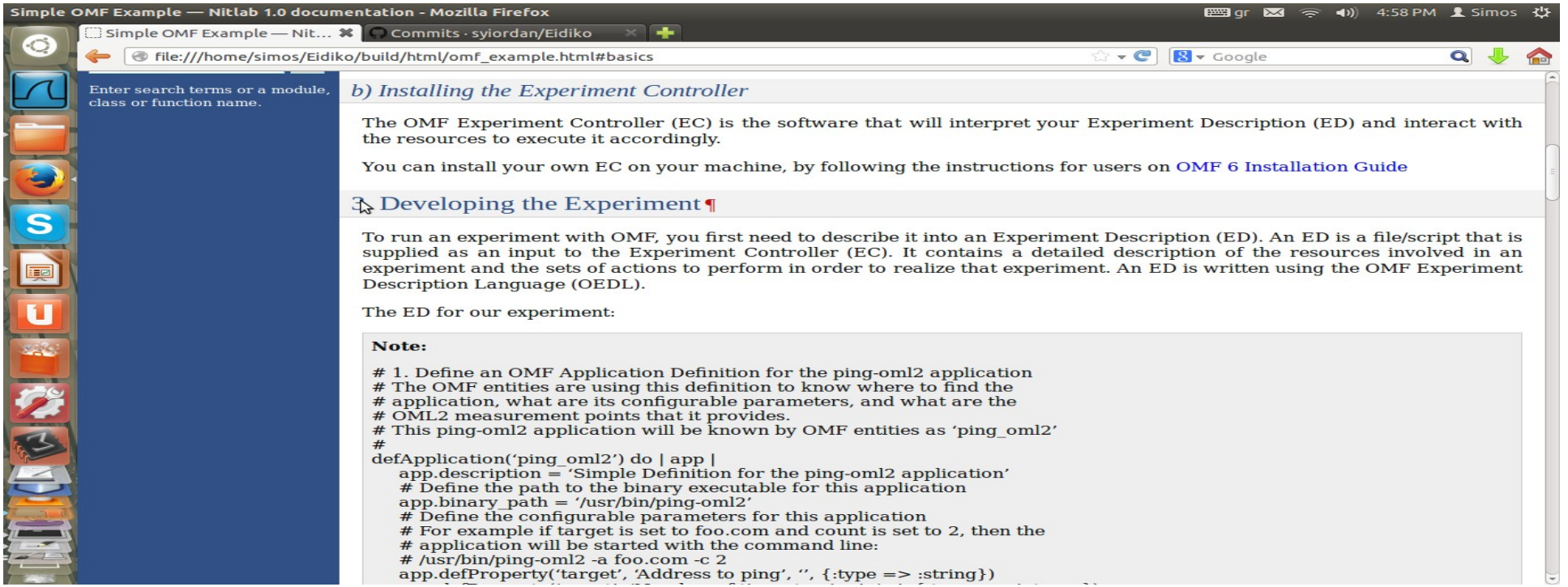
OMF Example


NITlab documentation — Nitlab 1.0 documentation - Mozilla Firefox
en
4:49 PM
Simos
NITlab documentation — N...
Commits · syiordan/Eidiko
file:///home/simos/Eidiko/build/html/index.html
Google
Enter search terms or a module, class or function name.
NITlab Video Tutorials
OMF Example
Simple OMF Example
1.Basics
2.Prerequisties
3. Developing the Experiment
4. Running Experiment
5.Accesing Results

•Στα συγκεκριμένα sections περιγράφεται αναλυτικά πως ο χρήστης μπορεί να εξοικειωθεί με το OMF με το να :

- Αναπτύξει ένα απλό πείραμα.
- Να τρέξει αυτό το πείραμα
- Να έχει πρόσβαση στα αποτελέσματα του πειράματος.


Simple OMF Example — Nitlab 1.0 documentation - Mozilla Firefox
4:57 PM Simos
Simple OMF Example — Nit...
Commits · syiordan/Eidiko
file:///home/simos/Eidiko/build/html/omf_example.html#basics
Google
Simple OMF Example
1.Basics
2.Prerequisties
a) Accessing/Provisioning Resources
b) Installing the Experiment Controller
3. Developing the Experiment
4. Running Experiment
a) How do you run it?
b) What will happen next?
c) What does that screen output mean?
5.Accesing Results
Previous topic
NITlab Video Tutorials
Next topic
WiFi
This Page
Show Source
Quick search
Go
Enter search terms or a module, class or function name.
1.Basics
This simple example introduces you to OMF6.It presents all the basic steps to develop, run, and access the result of an experiment with OMF 6.
Experiment Senario
This simple experiment involves a single resource of type PC, which has an active network interface. In this experiment, we will instruct that resource to start an instance of the 'ping-oml2' application to probe another host on the network attached to that interface (e.g. another host on the Internet, the LAN, or the resource's itself).
This 'ping-oml2' application is a wrapper around the traditional ping application. It captures the ping outputs and sends them as measurement streams to an OML2 collection point (an OML2 server in this case), which then stores them in a database available to the experimenter.
2.Prerequisties
a) Accessing/Provisioning Resources
Accessing a Resource at NITOS
Firstly, you have to reserve a node at NITOS: Getting a Slice and Reserving Resources
Provisioning a Resource at NITOS
You can use the method described on Loading an OMF-compatible image on your resource for instruction on how to image resources at NITOS. Load on your resource a disk image that contains OMF6. For example load the image baseline_grid_omf6_1_1.ndz.
b) Installing the Experiment Controller


Simple OMF Example — Nitlab 1.0 documentation - Mozilla Firefox
4:58 PM Simos
Simple OMF Example — Nit...
Commits · syiordan/Eidiko
file:///home/simos/Eidiko/build/html/omf_example.html#basics
Google
Enter search terms or a module, class or function name.
b) Installing the Experiment Controller
The OMF Experiment Controller (EC) is the software that will interpret your Experiment Description (ED) and interact with the resources to execute it accordingly.
You can install your own EC on your machine, by following the instructions for users on OMF 6 Installation Guide
3. Developing the Experiment
To run an experiment with OMF, you first need to describe it into an Experiment Description (ED). An ED is a file/script that is supplied as an input to the Experiment Controller (EC). It contains a detailed description of the resources involved in an experiment and the sets of actions to perform in order to realize that experiment. An ED is written using the OMF Experiment Description Language (OEDL).
The ED for our experiment:
Note:
1. Define an OMF Application Definition for the ping-oml2 application
The OMF entities are using this definition to know where to find the
application, what are its configurable parameters, and what are the
OML2 measurement points that it provides.
This ping-oml2 application will be known by OMF entities as 'ping_oml2'
#
defApplication('ping_oml2') do | app |
app.description = 'Simple Definition for the ping-oml2 application'
# Define the path to the binary executable for this application
app.binary_path = '/usr/bin/ping-oml2'
# Define the configurable parameters for this application
# For example if target is set to foo.com and count is set to 2, then the
# application will be started with the command line:
# /usr/bin/ping-oml2 -a foo.com -c 2
app.defProperty('target', 'Address to ping', '', {:type => :string})

```
# Here we define only one group (Sender), which has only one resource in it
#
defGroup('Sender', 'node016') do | g |
    # Associate the application ping_oml2 defined above to each resources
    # in this group
    g.addApplication("ping_oml2") do | app |
        # Configure the parameters for the ping_oml2 application
        app.setProperty('target', '8.8.8.8')
        app.setProperty('count', 3)
        # Request the ping_oml2 application to collect measurement samples
        # from the 'ping' measuremnt point (as defined above), and send them
        # to an OML2 collection point
        app.measure('ping', :samples => 1)
    end
end

# 3. Define the sequence of tasks to perform when the event
# "all resources are up and all applications are install" is being triggered
#
onEvent(:ALL_UP_AND_INSTALLED) do | event |
    # Print some information message
    info "This is my first OMF experiment"
    # Start all the Applications associated to all the Groups
    allGroups.startApplications
    # Wait for 5 sec
    # Stop all the Applications associated to all the Groups
    after 15 do
        allGroups.stopApplications
    end
    # Tell the Experiment Controller to terminate the experiment now
    after 10 do
        Experiment.done
    end
end
```

Simple OMF Example — Nitlab 1.0 documentation - Mozilla Firefox
Simple OMF Example — Nit...
Commits · syiordan/Eidiko
file:///home/simos/Eidiko/build/html/omf_example.html#basics
Google
4:58 PM Simos

4. Running Experiment

a) How do you run it?

To run your experiment you have to:

- **save its description in a file on your computer, thus either**
  - cut-and-paste the above ED listing into a new file named 'tutorial000.rb'
  - download the ED directly: ED_script
- **open a terminal and navigate to the folder/directory where you saved that file**
- **start the EC software and tell it to execute the experiment described in your ED file, using the command line:**

  omf_ec -u xmpp://usr:pwd@my_xmpp.com exec –oml_uri tcp:srv:port tutorial000.rb

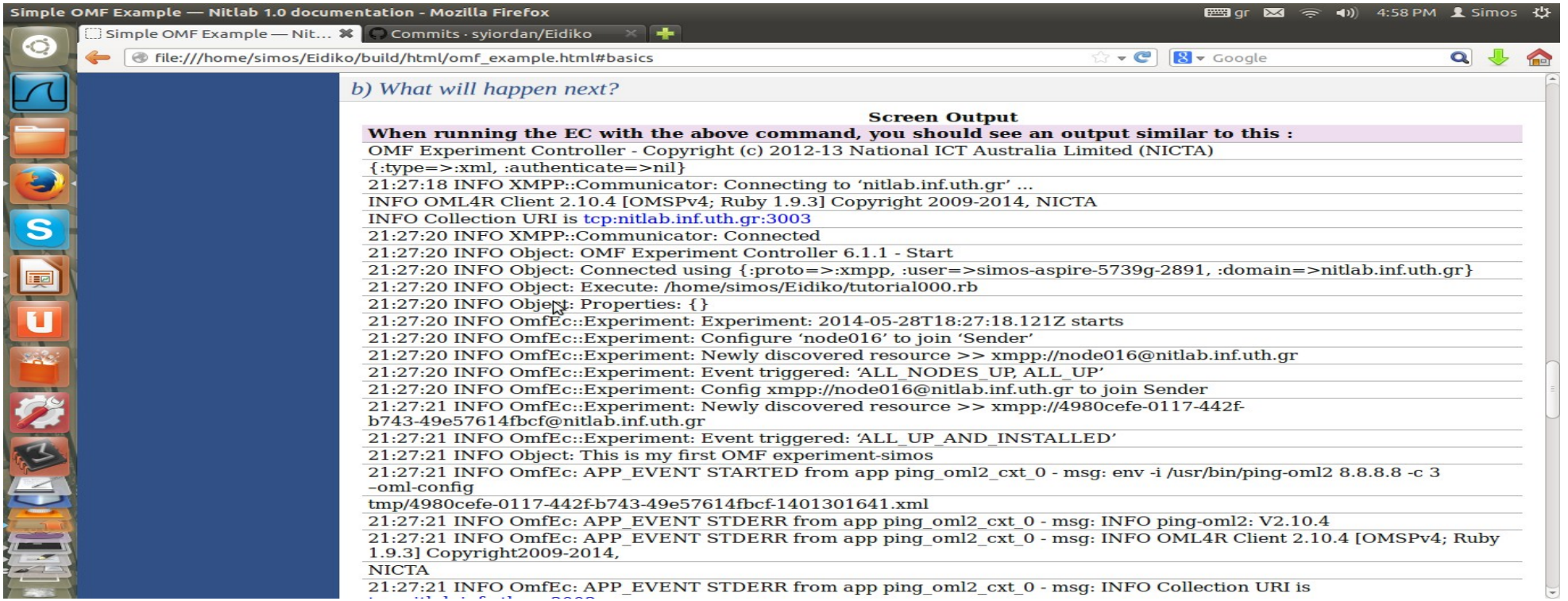
  - replace xmpp://usr:pwd@srv with the credentials for your user on the xmpp pubsub server that is used to communicate with the resources
  - replace tcp:srv:port with the hostname/IP and port of the OML2 server which will collect the experiment's measurement
- **So for our example, xmpp server and OML server run at nitlab.inf.uth.gr. Then you would use the command:**

  omf_ec -u xmpp://nitlab.inf.uth.gr exec –oml_uri tcp:nitlab.inf.uth.gr:3003 tutorial000.rb

If you would like to know more about the other options of the OMF EC software please run the commands:

```
omf_ec help
omf_ec help exec
```

b) What will happen next?

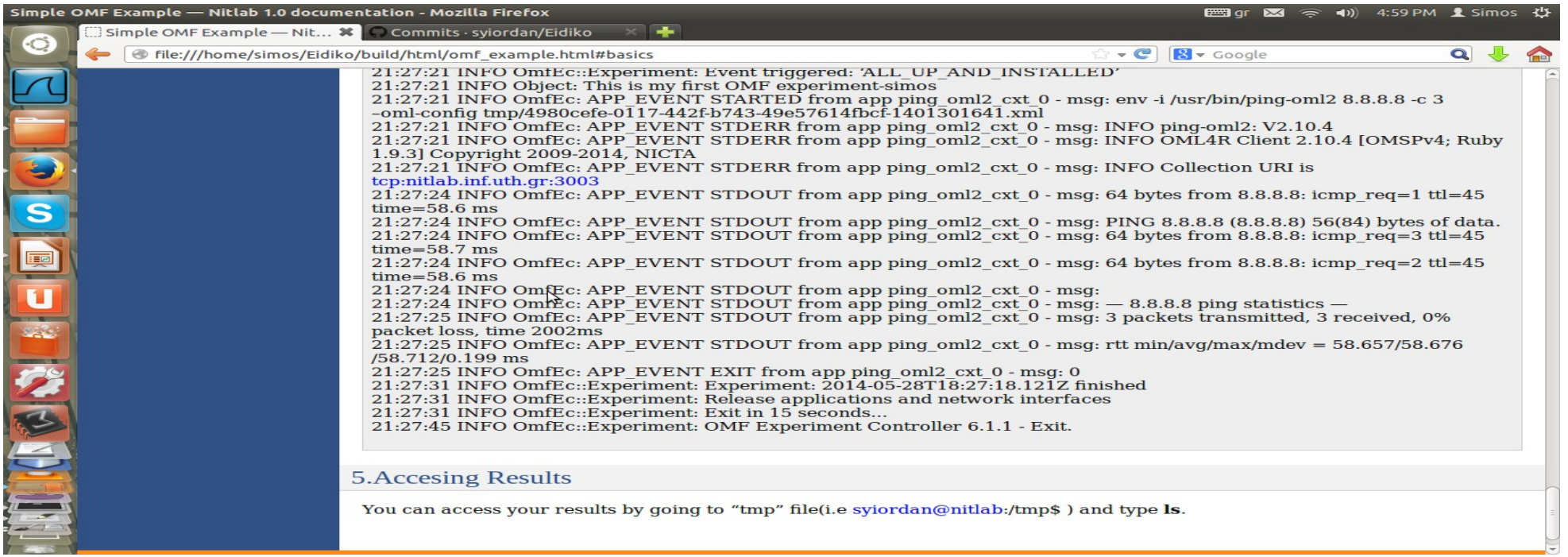


b) What will happen next?

Screen Output

When running the EC with the above command, you should see an output similar to this :

```
OMF Experiment Controller - Copyright (c) 2012-13 National ICT Australia Limited (NICTA)
{:type=>:xml, :authenticate=>nil}
21:27:18 INFO XMPP::Communicator: Connecting to 'nitlab.inf.uth.gr' ...
INFO OML4R Client 2.10.4 [OMSPv4; Ruby 1.9.3] Copyright 2009-2014, NICTA
INFO Collection URI is tcp:nitlab.inf.uth.gr:3003
21:27:20 INFO XMPP::Communicator: Connected
21:27:20 INFO Object: OMF Experiment Controller 6.1.1 - Start
21:27:20 INFO Object: Connected using {:proto=>:xmpp, :user=>simos-aspire-5739g-2891, :domain=>nitlab.inf.uth.gr}
21:27:20 INFO Object: Execute: /home/simos/Eidiko/tutorial000.rb
21:27:20 INFO Object: Properties: {}
21:27:20 INFO OmfEc::Experiment: Experiment: 2014-05-28T18:27:18.121Z starts
21:27:20 INFO OmfEc::Experiment: Configure 'node016' to join 'Sender'
21:27:20 INFO OmfEc::Experiment: Newly discovered resource >> xmpp://node016@nitlab.inf.uth.gr
21:27:20 INFO OmfEc::Experiment: Event triggered: 'ALL_NODES_UP, ALL_UP'
21:27:20 INFO OmfEc::Experiment: Config xmpp://node016@nitlab.inf.uth.gr to join Sender
21:27:21 INFO OmfEc::Experiment: Newly discovered resource >> xmpp://4980cefe-0117-442f-
b743-49e57614fbcf@nitlab.inf.uth.gr
21:27:21 INFO OmfEc::Experiment: Event triggered: 'ALL_UP_AND_INSTALLED'
21:27:21 INFO Object: This is my first OMF experiment-simos
21:27:21 INFO OmfEc: APP_EVENT STARTED from app ping_oml2_cxt_0 - msg: env -i /usr/bin/ping-oml2 8.8.8.8 -c 3
–oml-config
tmp/4980cefe-0117-442f-b743-49e57614fbcf-1401301641.xml
21:27:21 INFO OmfEc: APP_EVENT STDERR from app ping_oml2_cxt_0 - msg: INFO ping-oml2: V2.10.4
21:27:21 INFO OmfEc: APP_EVENT STDERR from app ping_oml2_cxt_0 - msg: INFO OML4R Client 2.10.4 [OMSPv4; Ruby
1.9.3] Copyright2009-2014,
NICTA
21:27:21 INFO OmfEc: APP_EVENT STDERR from app ping_oml2_cxt_0 - msg: INFO Collection URI is
```

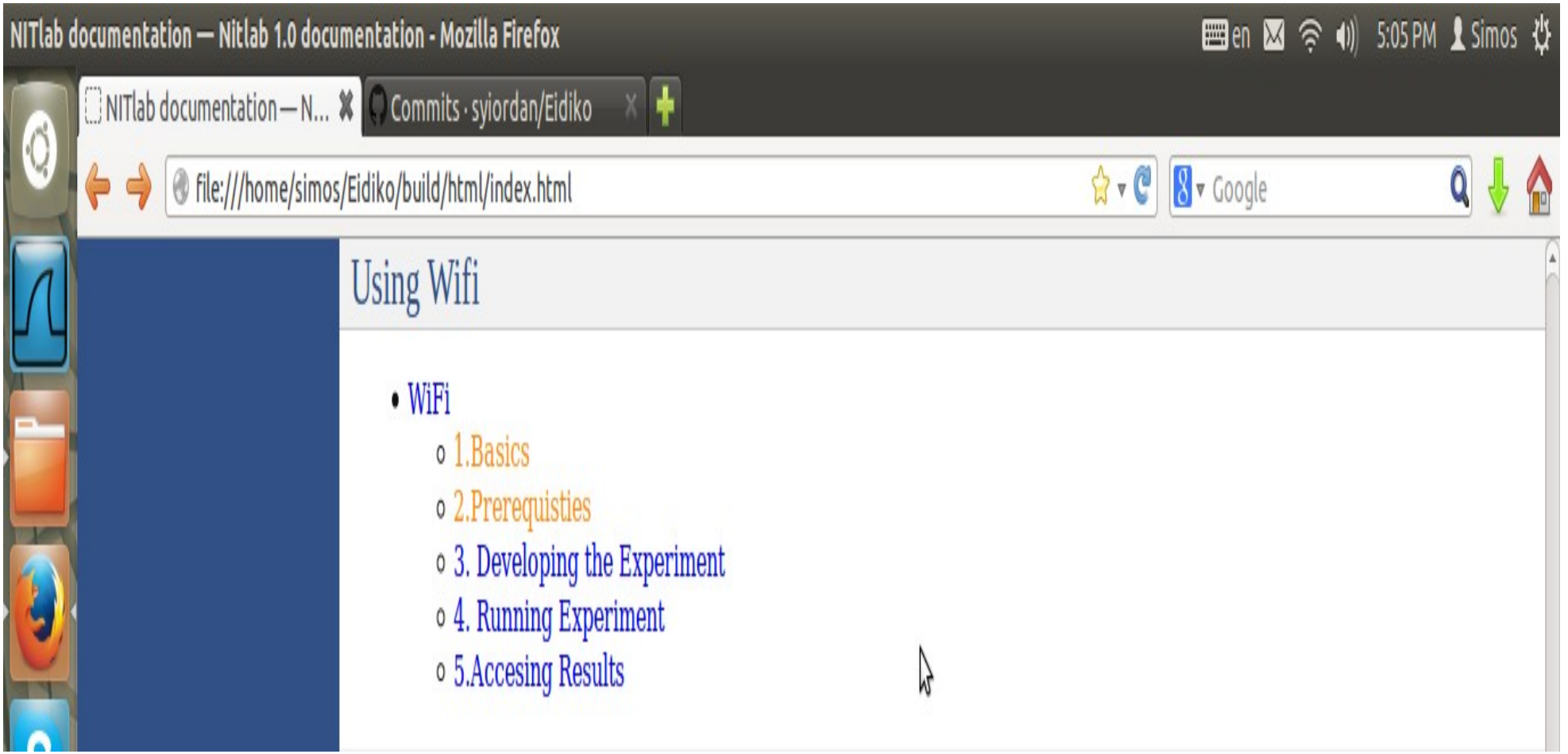



```
21:27:21 INFO OmfEc::Experiment: Event triggered: 'ALL_UP_AND_INSTALLED'
21:27:21 INFO Object: This is my first OMF experiment-simos
21:27:21 INFO OmfEc: APP_EVENT STARTED from app ping_oml2_cxt_0 - msg: env -i /usr/bin/ping-oml2 8.8.8.8 -c 3
–oml-config tmp/4980cefe-0117-442f-b743-49e57614fbcf-1401301641.xml
21:27:21 INFO OmfEc: APP_EVENT STDERR from app ping_oml2_cxt_0 - msg: INFO ping-oml2: V2.10.4
21:27:21 INFO OmfEc: APP_EVENT STDERR from app ping_oml2_cxt_0 - msg: INFO OML4R Client 2.10.4 [OMSPv4; Ruby
1.9.3] Copyright 2009-2014, NICTA
21:27:21 INFO OmfEc: APP_EVENT STDERR from app ping_oml2_cxt_0 - msg: INFO Collection URI is
tcp:nitlab.inf.uth.gr:3003
21:27:24 INFO OmfEc: APP_EVENT STDOUT from app ping_oml2_cxt_0 - msg: 64 bytes from 8.8.8.8: icmp_req=1 ttl=45
time=58.6 ms
21:27:24 INFO OmfEc: APP_EVENT STDOUT from app ping_oml2_cxt_0 - msg: PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
21:27:24 INFO OmfEc: APP_EVENT STDOUT from app ping_oml2_cxt_0 - msg: 64 bytes from 8.8.8.8: icmp_req=3 ttl=45
time=58.7 ms
21:27:24 INFO OmfEc: APP_EVENT STDOUT from app ping_oml2_cxt_0 - msg: 64 bytes from 8.8.8.8: icmp_req=2 ttl=45
time=58.6 ms
21:27:24 INFO OmfEc: APP_EVENT STDOUT from app ping_oml2_cxt_0 - msg:
21:27:24 INFO OmfEc: APP_EVENT STDOUT from app ping_oml2_cxt_0 - msg: — 8.8.8.8 ping statistics —
21:27:25 INFO OmfEc: APP_EVENT STDOUT from app ping_oml2_cxt_0 - msg: 3 packets transmitted, 3 received, 0%
packet loss, time 2002ms
21:27:25 INFO OmfEc: APP_EVENT STDOUT from app ping_oml2_cxt_0 - msg: rtt min/avg/max/mdev = 58.657/58.676
/58.712/0.199 ms
21:27:25 INFO OmfEc: APP_EVENT EXIT from app ping_oml2_cxt_0 - msg: 0
21:27:31 INFO OmfEc::Experiment: Experiment: 2014-05-28T18:27:18.121Z finished
21:27:31 INFO OmfEc::Experiment: Release applications and network interfaces
21:27:31 INFO OmfEc::Experiment: Exit in 15 seconds...
21:27:45 INFO OmfEc::Experiment: OMF Experiment Controller 6.1.1 - Exit.
```

5.Accesing Results

You can access your results by going to "tmp" file(i.e syiordan@nitlab:/tmp$) and type **ls**.

Using WiFi


NITlab documentation — Nitlab 1.0 documentation - Mozilla Firefox
en
5:05 PM
Simos
NITlab documentation — N...
Commits · syiordan/Eidiko
file:///home/simos/Eidiko/build/html/index.html
Google
Using Wifi
WiFi
1.Basics
2.Prerequisties
3. Developing the Experiment
4. Running Experiment
5.Accesing Results

- Στα συγκεκριμένα sections περιγράφεται αναλυτικά στα αντίστοιχα βήματα με πρίν πως ο χρήστης μπορεί να εξοικειωθεί με την WiFi τεχνολογία με το να :

- Αναπτύξει ένα απλό wireless OMF πείραμα που γεννά κίνηση μεταξύ 2 κόμβων πάνω απο ασύρματο κανάλι.
- Να τρέξει αυτό το πείραμα
- Να έχει πρόσβαση στα αποτελέσματα του πειράματος.

Using WiMax


Using WiMax
WiMax
1.Basics
2.Prerequisties
3.Experimental Topology
4.Connecting to the WiMAX Network
5.Set up experimentation properties of the BS
6.Developing the experiment
7.Running Experiment
8.Accessing Results

- Στα συγκεκριμένα sections περιγράφεται αναλυτικά στα πως ο χρήστης μπορεί να εξοικειωθεί με την WiMax τεχνολογία με το να :

- Αποκτήσει εικόνα της WiMax Topology.
- Μάθει να συνδέεται στον κατάλληλο Server και στο WiMax Network.
- Μάθει να χειρίζεται παραμέτρους του Base Station.
- Αναπτύξει ένα απλό WiMax OMF πείραμα που γεννά κίνηση μεταξύ 2 κόμβων.
- Να τρέξει αυτό το πείραμα
- Να έχει πρόσβαση στα αποτελέσματα του πειράματος.

1.Basics

Firstly,you will see how to connect and use WiMax.You'll also see how to develop and run a simple experiment where the WiMAX interface is setup and generate traffic between two nodes.

2.Prerequisties

Accessing/Provisioning Resources

Accessing a Resource at NITOS

Firstly, you have to reserve a node at NITOS: *Getting a Slice and Reserving Resources*

In order to use the WiMAX testbed resources offered by the NITOS facility, you will have to reserve the WiMAX base station and the nodes with WIMAX connectivity, offered by the testbed (**Icarus nodes41-49**).

Reserving the base station will render rights to the experimenter to access the WiMAXrf services of the testbed, used to configure and setup the base station at the experimenter's will.

After your reservation has started,login to Server.

Warning: You have to connect to **Nitos Server2**!

You can use your NITlab account to access the Nitos Server2.Just open a secure shell,write **ssh slice_name@nitlab2.inf.uth.gr** (i.e syiordan@nitlab2.inf.uth.gr) and log into the server.

Provisioning a Resource at NITOS

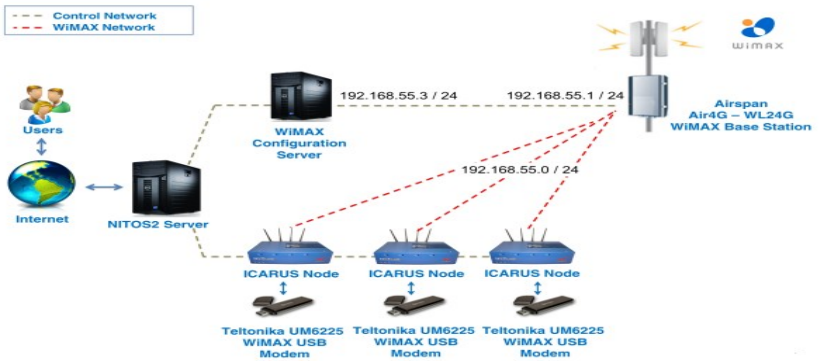
You can use the method described on *Loading an OMF-compatible image on your resource* for instruction on how to image resources at NITOS. Load on your resource a disk image that is compatible for WiMax experimentation such as **fgre_2014.ndz**.

WiMax — Nitlab 1.0 documentation - Mozilla Firefox
WiMax — Nitlab 1.0 docum... | Commits · syiordan/Eidiko | wifi nitos topology - Αναζήτ...
file:///home/simos/Eidiko/build/html/wiMax.html#basics
wifi nitos topology

3.Experimental Topology

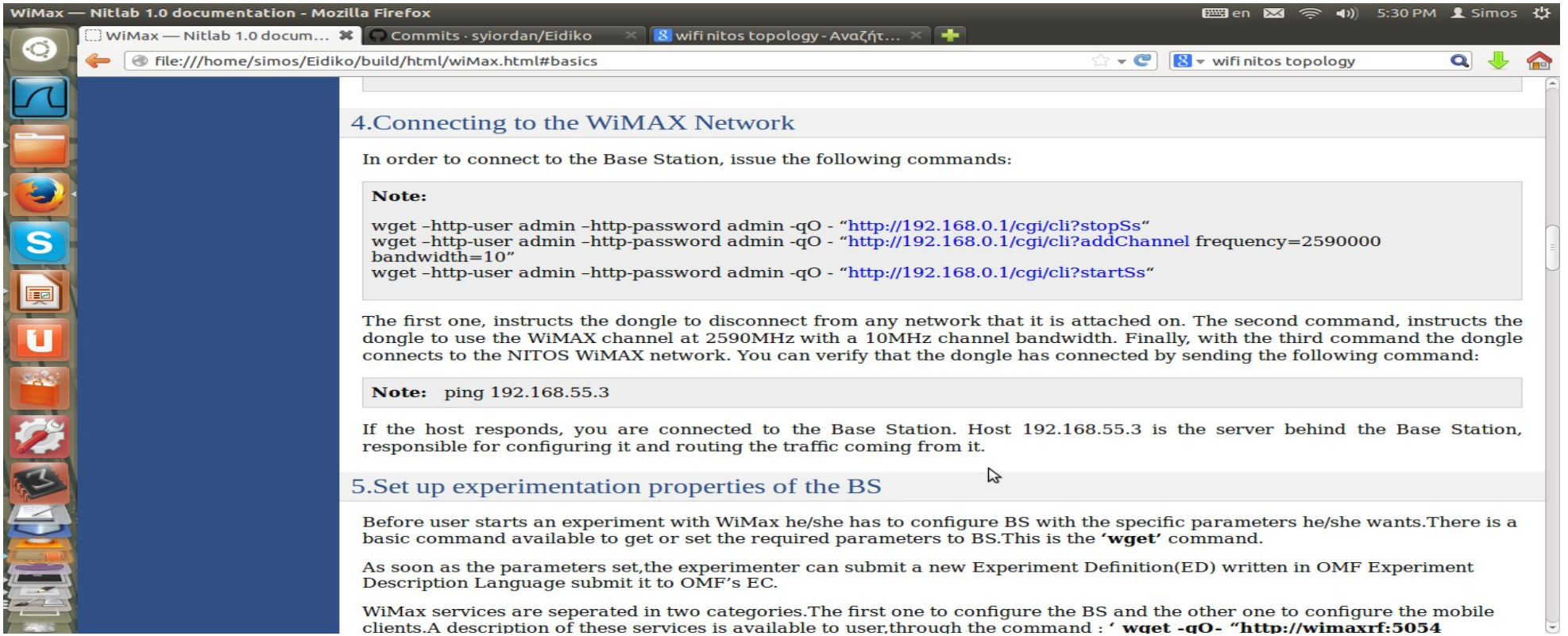
The topology that we are examining in this tutorial is a central Base Station unit, able to route requests from an internal WiMAX network to the internet.

- The Base Station is using the 192.168.55.1 IP address
- Multiple clients use the 192.168.55.0/24 subnet to communicate with the Base Station
- Packets routed through the Base Station are sent over the internet through a NAT translation



However, the devices that we currently use do not allow us to interface directly the WiMAX device. If you send the following command

root@node044:~# ifconfig tel0



4.Connecting to the WiMAX Network

In order to connect to the Base Station, issue the following commands:

Note:

wget –http-user admin –http-password admin -qO - "http://192.168.0.1/cgi/cli?stopSs"
wget –http-user admin –http-password admin -qO - "http://192.168.0.1/cgi/cli?addChannel frequency=2590000 bandwidth=10"
wget –http-user admin –http-password admin -qO - "http://192.168.0.1/cgi/cli?startSs"

The first one, instructs the dongle to disconnect from any network that it is attached on. The second command, instructs the dongle to use the WiMAX channel at 2590MHz with a 10MHz channel bandwidth. Finally, with the third command the dongle connects to the NITOS WiMAX network. You can verify that the dongle has connected by sending the following command:

Note: ping 192.168.55.3

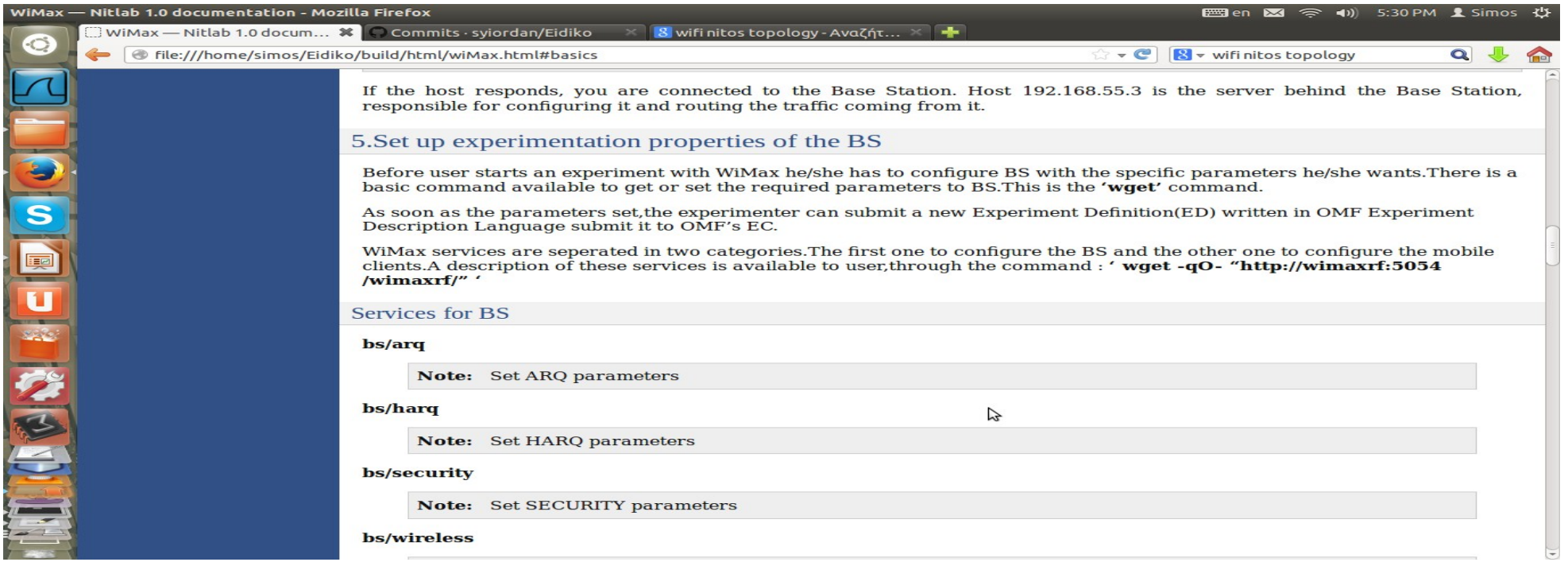
If the host responds, you are connected to the Base Station. Host 192.168.55.3 is the server behind the Base Station, responsible for configuring it and routing the traffic coming from it.

5.Set up experimentation properties of the BS

Before user starts an experiment with WiMax he/she has to configure BS with the specific parameters he/she wants.There is a basic command available to get or set the required parameters to BS.This is the **'wget'** command.

As soon as the parameters set,the experimenter can submit a new Experiment Definition(ED) written in OMF Experiment Description Language submit it to OMF's EC.

WiMax services are seperated in two categories.The first one to configure the BS and the other one to configure the mobile clients.A description of these services is available to user,through the command : ' **wget -qO- "http://wimaxrf:5054**



If the host responds, you are connected to the Base Station. Host 192.168.55.3 is the server behind the Base Station, responsible for configuring it and routing the traffic coming from it.

5.Set up experimentation properties of the BS

Before user starts an experiment with WiMax he/she has to configure BS with the specific parameters he/she wants.There is a basic command available to get or set the required parameters to BS.This is the **'wget'** command.

As soon as the parameters set,the experimenter can submit a new Experiment Definition(ED) written in OMF Experiment Description Language submit it to OMF's EC.

WiMax services are seperated in two categories.The first one to configure the BS and the other one to configure the mobile clients.A description of these services is available to user,through the command : ' **wget -qO- "http://wimaxrf:5054 /wimaxrf/"** '

Services for BS

bs/arq

Note: Set ARQ parameters

bs/harq

Note: Set HARQ parameters

bs/security

Note: Set SECURITY parameters

bs/wireless

Since the BS is back to its default settings, you can start experimenting with the WiMAX testbed.

6.Developing the experiment

To run an experiment with OMF, you first need to describe it into an Experiment Description (ED). An ED is a file/script that is supplied as an input to the Experiment Controller (EC). It contains a detailed description of the resources involved in an experiment and the sets of actions to perform in order to realize that experiment. An ED is written using the OMF Experiment Description Language (OEDL).

The ED for our experiment:

Note:

```
defProperty('runtime',20,"Time in second for the experiment is to run")
defProperty('client',"192.168.55.48","IP address of iperf server")
defProperty('interval', "1", "Interval of Iperf measurements")
defProperty('sender', 'omf.nitos.node043', "ID of sender node")
defProperty('receiver', 'omf.nitos.node048', "ID of sender node")

defGroup('Sender', property.sender) do | node |
    node.net.t0.channel = "2590000,10"
    node.addApplication("test:app:iperf-5.4") do | app |
        app.setProperty('client', property.client)
        app.setProperty('interval', property.interval)
        app.setProperty('time', property.runtime)
        app.setProperty('udp', true)
        app.measure('transfer', :samples =>1)
    end
end

defGroup('Receiver', property.receiver) do | node |
    node.net.t0.channel = "2590000,10"
    node.addApplication("test:app:iperf-5.4") do | app |
        app.setProperty('server', true)
```

WiMax — Nitlab 1.0 documentation - Mozilla Firefox
WiMax — Nitlab 1.0 docum... | Commits · syiordan/Eidiko | wifi nitos topology - Αναζήτ...
file:///home/simos/Eidiko/build/html/wiMax.html#basics
wifi nitos topology

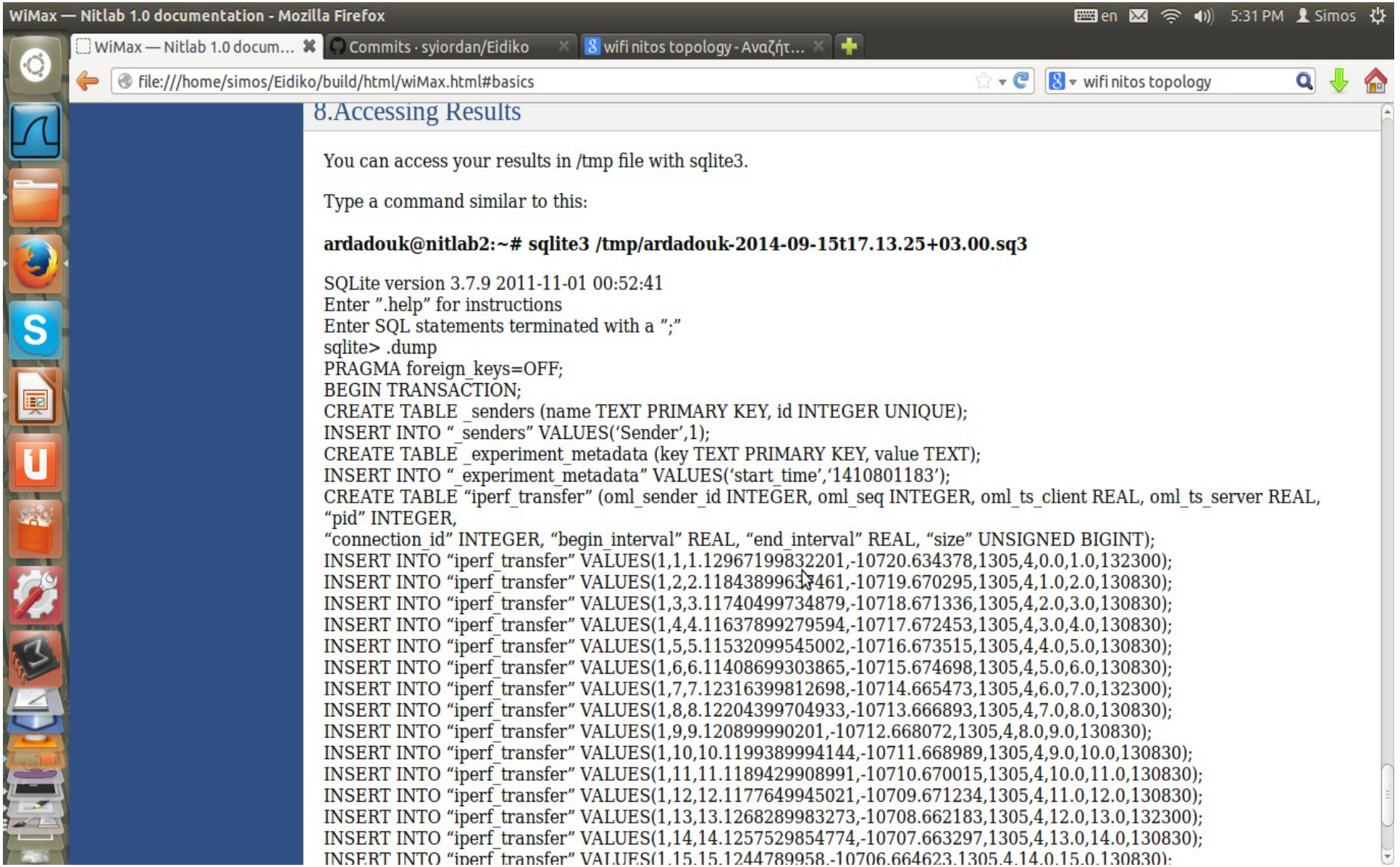
7.Running Experiment

You can run it from the testbed server with the following command:

omf exec experiment.rb -slice your_username

Screen Output

When running the EC with the above command, you should see an output similar to this :
INFO NodeHandler: OMF Experiment Controller 5.4 (git 3105a52)
INFO NodeHandler: Slice ID: ardadouk
INFO NodeHandler: Experiment ID: ardadouk-2014-09-15t17.13.25+03.00
INFO NodeHandler: Message authentication is disabled
INFO Experiment: load system:exp:stdlib
INFO property.resetDelay: resetDelay = 90 (Fixnum)
INFO property.resetTries: resetTries = 1 (Fixnum)
INFO Experiment: load system:exp:eventlib
INFO Experiment: load wimax_experiment.rb
INFO property.runtime: runtime = 20 (Fixnum)
INFO property.client: client = 192.168.55.47" (String)"
INFO property.interval: interval = 1" (String)"
INFO property.sender: sender = omf.nitos.node046" (String)"
INFO property.receiver: receiver = omf.nitos.node047" (String)"
INFO Topology: Loading topology 'omf.nitos.node046'.
INFO Topology: Loading topology 'omf.nitos.node047'.
INFO Experiment: Switching ON resources which are OFF
INFO ALL_UP_AND_INSTALLED: Event triggered. Starting the associated tasks.
INFO exp: Request from Experiment Script: Wait for 50s....
INFO exp: This is an iperf experiment using a teltonika modem
INFO exp: Request from Experiment Script: Wait for 20s....


WiMax — Nitlab 1.0 documentation - Mozilla Firefox
en
5:31 PM
Simos
WiMax — Nitlab 1.0 docum...
Commits · syiordan/Eidiko
wifi nitos topology - Αναζήτ...
file:///home/simos/Eidiko/build/html/wiMax.html#basics
wifi nitos topology
8.Accessing Results
You can access your results in /tmp file with sqlite3.
Type a command similar to this:
ardadouk@nitlab2:~# sqlite3 /tmp/ardadouk-2014-09-15t17.13.25+03.00.sq3
SQLite version 3.7.9 2011-11-01 00:52:41
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> .dump
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE _senders (name TEXT PRIMARY KEY, id INTEGER UNIQUE);
INSERT INTO "_senders" VALUES('Sender',1);
CREATE TABLE _experiment_metadata (key TEXT PRIMARY KEY, value TEXT);
INSERT INTO "_experiment_metadata" VALUES('start_time','1410801183');
CREATE TABLE "iperf_transfer" (oml_sender_id INTEGER, oml_seq INTEGER, oml_ts_client REAL, oml_ts_server REAL,
"pid" INTEGER,
"connection_id" INTEGER, "begin_interval" REAL, "end_interval" REAL, "size" UNSIGNED BIGINT);
INSERT INTO "iperf_transfer" VALUES(1,1,1.12967199832201,-10720.634378,1305,4,0.0,1.0,132300);
INSERT INTO "iperf_transfer" VALUES(1,2,2.11843899637461,-10719.670295,1305,4,1.0,2.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,3,3.11740499734879,-10718.671336,1305,4,2.0,3.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,4,4.11637899279594,-10717.672453,1305,4,3.0,4.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,5,5.11532099545002,-10716.673515,1305,4,4.0,5.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,6,6.11408699303865,-10715.674698,1305,4,5.0,6.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,7,7.12316399812698,-10714.665473,1305,4,6.0,7.0,132300);
INSERT INTO "iperf_transfer" VALUES(1,8,8.12204399704933,-10713.666893,1305,4,7.0,8.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,9,9.120899990201,-10712.668072,1305,4,8.0,9.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,10,10.1199389994144,-10711.668989,1305,4,9.0,10.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,11,11.1189429908991,-10710.670015,1305,4,10.0,11.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,12,12.1177649945021,-10709.671234,1305,4,11.0,12.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,13,13.1268289983273,-10708.662183,1305,4,12.0,13.0,132300);
INSERT INTO "iperf_transfer" VALUES(1,14,14.1257529854774,-10707.663297,1305,4,13.0,14.0,130830);
INSERT INTO "iperf_transfer" VALUES(1,15,15.1244789958,-10706.664623,1305,4,14.0,15.0,130830);

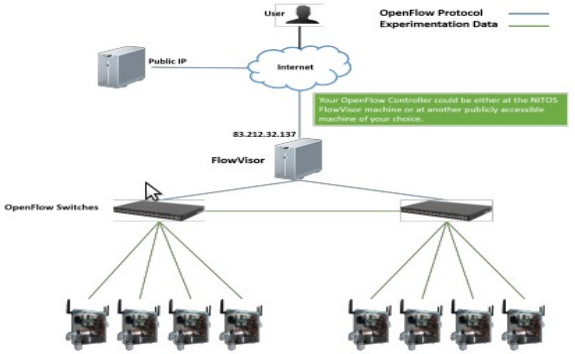
Using Openflow

- Στα συγκεκριμένα sections περιγράφεται αναλυτικά πως ο χρήστης μπορεί να εξοικειωθεί με την OpenFlow τεχνολογία με το να :

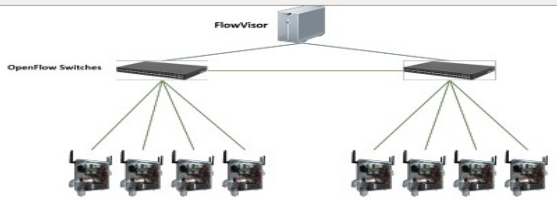
- Αποκτήσει εικόνα για την OpenFlow τεχνολογία.
- Αποκτήσει εικόνα για τα OpenFlow Settings.

NITOS facility provides remote access to OpenFlow switches, enabling the user to create an OpenFlow slice, related to an experiment slice that has already available. The OpenFlow slice is implemented by the FlowVisor, a Stanford's software tool that creates slices on the OpenFlow switches, enabling the parallel access to them by separate users. In a transparent way from the user perspective, the switches ports, which are used by the user's reserved nodes, are assigned to the user's OpenFlow slice. The whole process is orchestrated by the NITOS scheduler, configuring appropriately the FlowVisor at the beggining of a new reservation slot.

In summary, the remote user of the NITOS facility is able to reserve nodes for his slice (through the NITOS scheduler) and run an OpenFlow experiment with use of this slice. In particular, the testbed provides transparently an abstract OpenFlow switch for his slice, that conceptually is equivalent to a physical OpenFlow switch that includes only the ports of the nodes that he has reserved. The OpenFlow controller that defines the functionality of this abstract switch, should listen both on the host machine and the TCP port that are illustrated in the OpenFlow settings, in the User Menu on the right of the page (requires registration). The following figure illustrates the NITOS OpenFlow testbed topology, as well as the slicing mechanism.



Openflow — Nitlab 1.0 documentation - Mozilla Firefox
Openflow — Nitlab 1.0 doc... | Commits · syiordan/Eidiko | wifi nitos topology - Αναζήτ...
file:///home/simos/Eidiko/build/html/openflow.html#openflow-testbed-topology
wifi nitos topology



2)OpenFlow settings

Current Slices: syiordan.

Along with the reservation of NITOS nodes, the related ports of the OpenFlow switches are also included in the corresponding experimentation slice. Please provide the IP address of the appropriate OpenFlow controller to perform an OpenFlow experiment with the use of this slice.

The current IP addresses of the OpenFlow controllers of each of your slices are depicted in the following table:

syiordan
IP:
Port:

In case that one of the above IP addresses is 127.0.0.1, you can raise your OpenFlow controller at the NITOS FlowVisor server machine at 83.212.32.137. You are able to connect to the FlowVisor server machine through ssh, using the same ssh-keys that you use to connect to the NITOS server machine.

Μελλοντικά ζητήματα

- Το εγχειρίδιο μπορεί να εμπλουτιστεί και με άλλες κατηγορίες,όπως **LTE Experimentation** και **Software Defined Radio.**

- Αγγλική παρουσίαση του εγχειριδίου.

Τι αποκόμισα...

- Βασικές γνώσεις για την χρήση του Nitos Testbed και τον πειραματισμό με τις διάφορες τεχνολογίες.
- Εκμάθηση εργαλείων(Sphinx,Github).
- Εκμάθηση βασικών στοιχείων HTML/CSS

Πηγές

- http://nitlab.inf.uth.gr/NITlab/
- http://mytestbed.net/
- http://sphinx-doc.org/
- https://github.com/

Ερωτήσεις