Title: Running RINA on GENI

Overview:
Recursive InterNetwork Architecture (RINA) is a new network architecture based on the fundamental principle that networking is Inter-Process Communication (IPC). RINA separates mechanisms and policies, and enables policy-based dynamic service composition. This tutorial introduces the basic elements of the Recursive InterNetwork Architecture (RINA) (http://csr.bu.edu/rina/) and describes how to setup RINA experiments on GENI nodes.

Prerequisites:

- A basic understanding of GENI concepts such as those covered in the Introduction to GENI and Experimentation using GENI tutorial (http://groups.geni.net/geni/wiki/GEC19Agenda/IntroToGENI).

- Comfortable with running your own experiments on GENI resources.

- Verify that you are able to log into the GENI Portal (https://portal.geni.net).

- A basic familiarity with using a UNIX command line.

- A laptop with Mac OS or Linux OS. For Windows users, install Cygwin, or a recent version of Virtual Box (https://www.virtualbox.org/) with an Ubuntu image.

- A basic understanding of the Recursive InterNetwork Architecture (RINA: http://csr.bu.edu/rina).

Tools:

- GENI Portal.
- UNIX command line.

Where to get help:

- For support and general questions about ProtoRINA, please contact the ProtoRINA team at *protorina-developers@cs.bu.edu*.
- You are welcome to subscribe to ProtoRINA users mailing list at http://cs-mailman.bu.edu/mailman/listinfo/protorina-users

Experiment 4: Routing

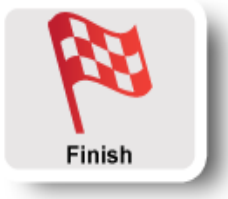
Tutorial Instructions:

Part1: Design/Setup

- **Design the Experiment**
- **Establish the Environment**

Part2: Execute

- **Login to VMs**
- **Import configuration files**
- **Start IDD and DNS**
- **Start Node1**
- **Start Node2**
- **Start Node3**
- **Start Node4**
- **Stop Node2**
- **Analysis**



Part3: Finish

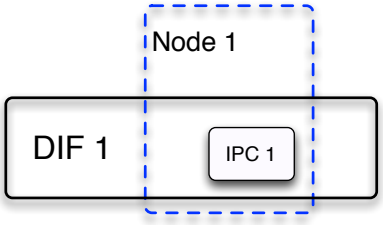
Experiment 4: Routing

Part1: Design/Setup:

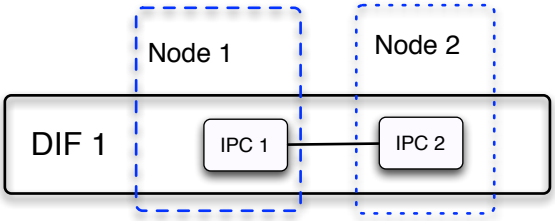
1. Design the Experiment

In this experiment, we use four RINA nodes (Node1, Node2, Node3 and Node4).

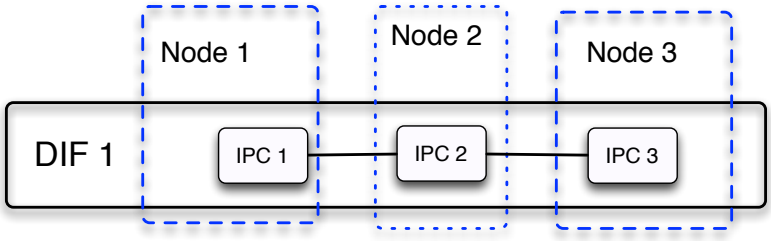
First we start Node1. It has one IPC process (IPC1) belonging to DIF1.



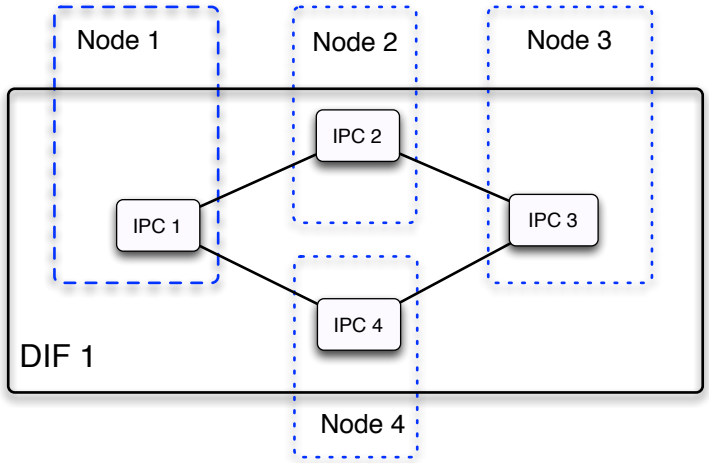
Second we start Node2. It has one IPC process (IPC2), and IPC2 will join DIF1 through IPC1.



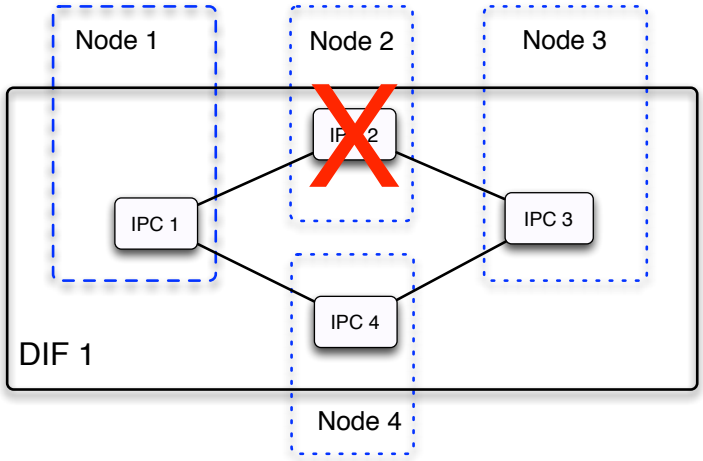
Third we start Node3. It has one IPC process (IPC3), and IPC3 will join DIF1 through IPC2. The next hop of IPC1 to IPC3 is IPC2.



Fourth we start Node4. It has one IPC process (IPC4), and IPC4 will join DIF1 through IPC1. Also IPC4 and IPC3 are configured to be neighbors. Now there are two paths of the same cost in DIF1 between IPC1 and IPC3. IPC2 is the next hop of IPC1 to IPC3.



Last we terminate Node2, and the path from IPC1 to IPC3 through IPC2 is broken. Then after the routing information is updated in DIF1, the route between IPC 1 and IPC 3 adapts to the path IPC1- IPC4- IPC3.



2. Establish the Environment

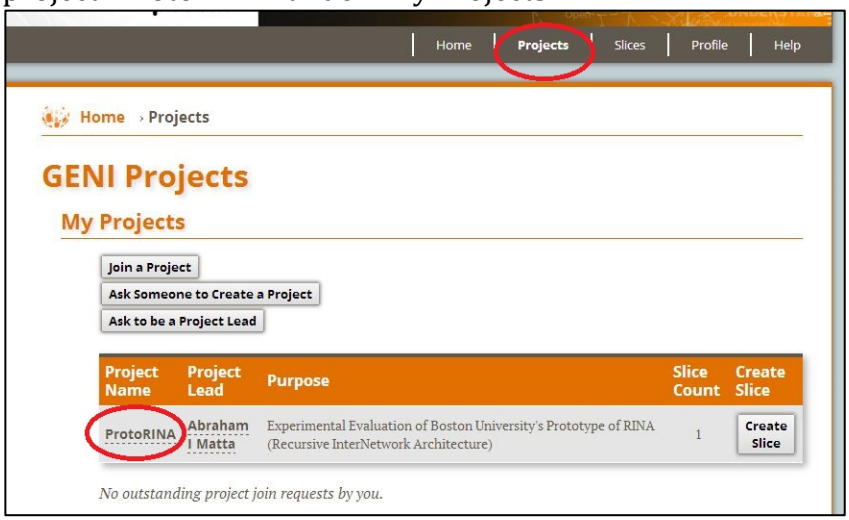
Skip this section if you have already established your environment for RINA

2.1. Pre-work:

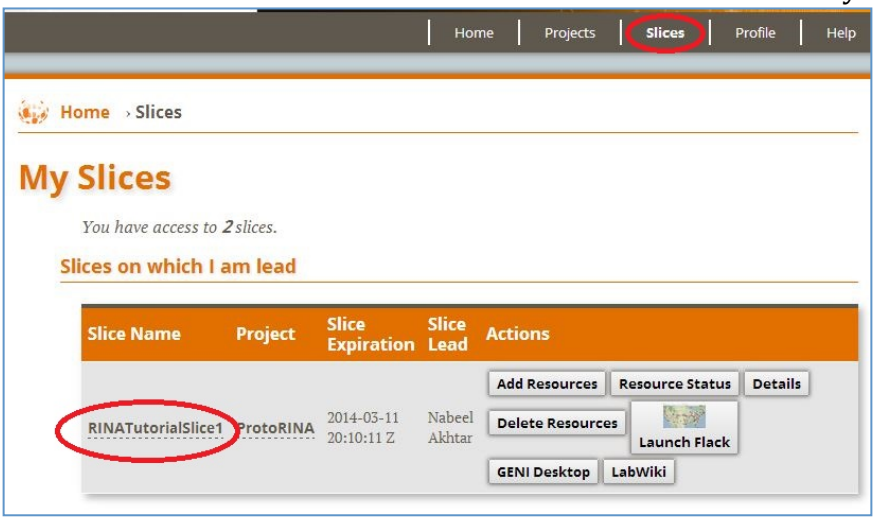
- Ensure SSH keys are setup. If your SSH keys are not setup before, do the following steps:
  - Generate an SSH Private Key on the *Profile* page on the GENI portal
  - Download Private Key to *~/Downloads*
  - Open terminal and execute
    $ mv ~/Downloads/id_geni_ssh_rsa ~/.ssh/.
    $ chmod 0600 ~/.ssh/id_geni_ssh_rsa
    $ ssh-add ~/.ssh/id_geni_ssh_rsa

- Ensure you are part of the project "ProtoRINA".

- To check this, go to portal.geni.net -> Projects. You should be able to see project “ProtoRINA” under “My Projects”.



- Ensure that you are added to the slice by the lead.
  - To check this, go to portal.geni.net -> Slices. You should be able to see slice “RINATutorialSlice1” or “RINATutorialSlice2” under “My Slices”.

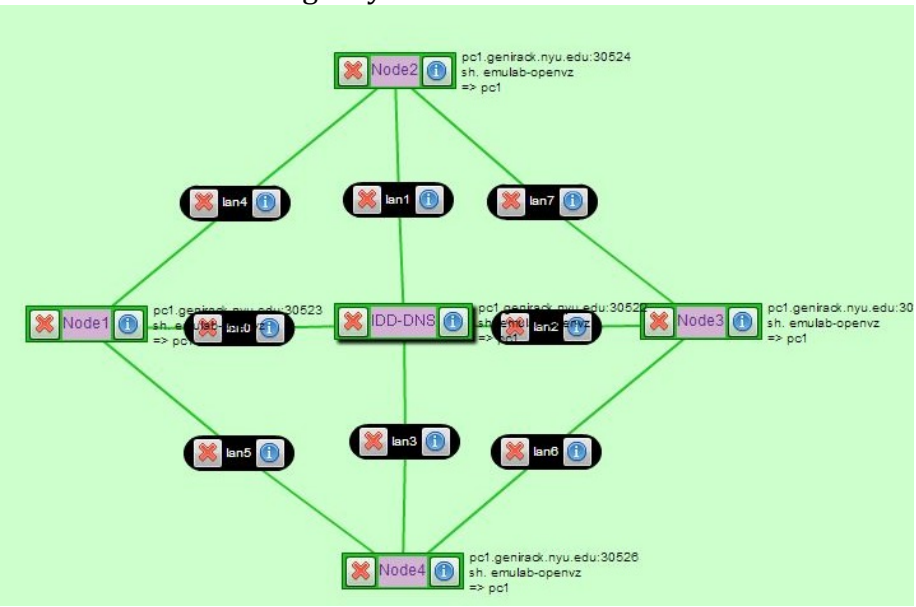


2.2. Check Resources:

- Click on “Launch Flack” on your slice. That might take a few minutes to display the whole GUI.

DO NOT DELETE RESOURCES!!!

- You should see the following on your Flack:

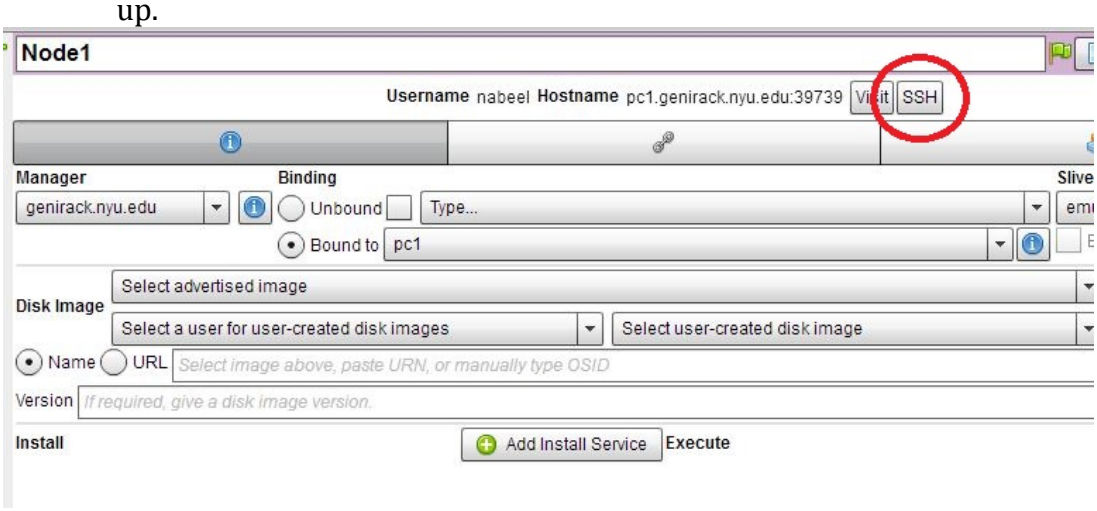


Part2: Execute:

1. **Login to VMs**
   In this experiment, we will use five VMs (named Node1, Node2, Node3, Node 4 and IDD-DNS). You can see these five VMs on Flack.

   a) To login to a VM, click on  on "Node1" as shown below:

   b) Now, Click on SSH link as shown below. A new terminal window will pop up.



To ssh from the command line, do the following (substituting with values shown on the screen).

```
ssh USERNAME@HOSTNAME -p PORT
```

   c) Go back to Flack, and ssh to the other four VMs (Node2, Node3, Node4 and IDD-DNS) following the instructions given above.
   d) Make sure you open two terminal windows for "IDD-DNS". We will run the IDD process and DNS process separately using two terminal windows.

You should have 6 terminal windows now: One for Node1, one for Node2, one for Node3, one for Node4 and two for IDD-DNS.

The IDD process provides RINA directory services

2. Import configuration files

Now you will import configuration files to your home directory for five VMs respectively.

For each VM, copy your group files to your local directory as shown below. You can find your group number in the worksheet provided to you.

```
cp -r /users/Tutorial/ master/group1/ ~
```

Make sure you copy data for your own group provided in the handout.

3. Start IDD and DNS

We will start the IDD process and DNS process now.

a) To start DNS, go to directory with the DNS properties file using one of the IDD-DNS terminal windows:

```
cd group1/IDD-DNS/DNS/
```

Now run the following command to run DNS:

```
java -jar /users/Tutorial/master/IDD-DNS/DNS/DNS.jar dns.properties
```

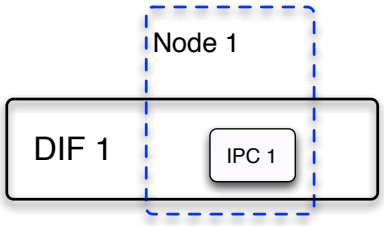
b) To run IDD, go to directory with the IDD properties file using the second IDD-DNS terminal window:

```
cd group1/IDD-DNS/IDD/
```

Now run the following command to run IDD:

```
java -jar /users/Tutorial/master/IDD-DNS/IDD/IDD.jar idd.properties
```

4. Start Node1



Now we will start Node1 process on the first VM (Node1).

a) In the terminal for Node1, go to directory for Node1 as shown below:

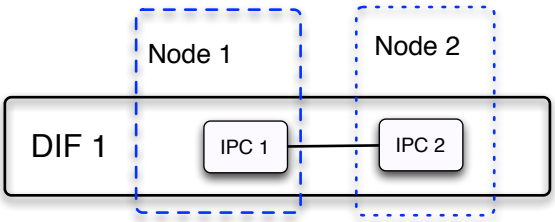
```
cd group1/Configurations/exp4/Node1/
```

b) Start Node1 process using the following command:

```
java -jar /users/Tutorial/master/RINANode.jar Node1.properties
```

At this point, your Node1 should be up and running.

5. Start Node2



We start Node 2 process. It has one IPC process (IPC2), and IPC2 joins DIF1 through IPC1.

a) In the terminal for Node2, go to directory for Node2 as shown below:

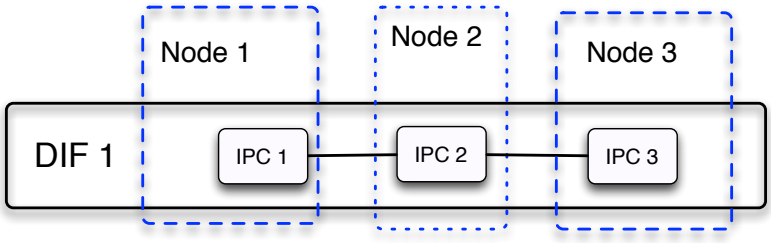
```
cd group1/Configurations/exp4/Node2
```

b) Start Node2 process using the following command:

```
java -jar /users/Tutorial/master/RINANode.jar Node2.properties
```

At this point, your Node2 should be up and running, and it will join DIF1 through IPC1 on Node1.

6. Start Node3



In this step, we start Node3. It has one IPC process (IPC3), and IPC3 joins DIF1 through IPC2.

a) In the terminal for Node3, go to directory for Node3 as shown below:

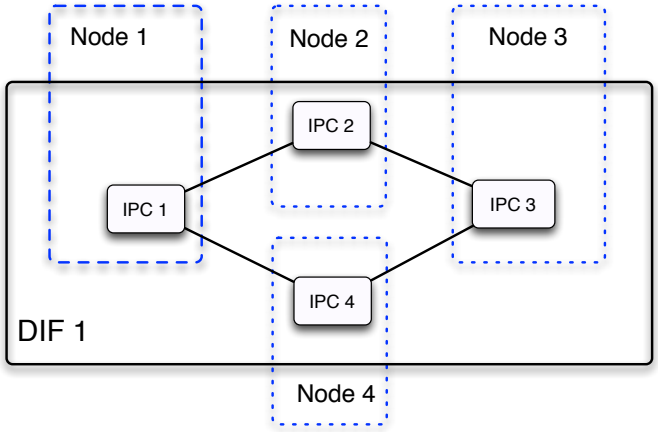
```
cd group1/Configurations/exp4/Node3
```

b) Start Node3 process using the following command:

```
java -jar /users/Tutorial/master/RINANode.jar Node3.properties
```

7. Start Node4

In this step, we start Node4. Node4 has one IPC process (IPC4), and IPC4 joins DIF1 through IPC1. Also IPC4 and IPC3 are configured to be neighbors.



a) In the terminal for Node4, go to directory for Node4 as shown below:

```
cd group1/Configurations/exp4/Node4
```

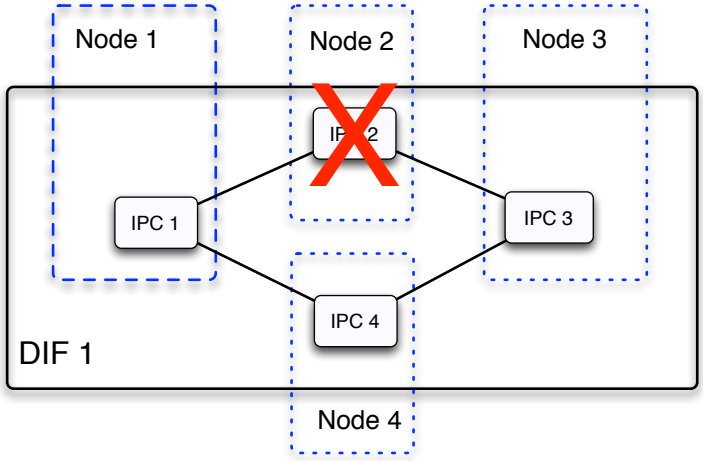
b) Start Node4 using the following command:

```
java -jar /users/Tutorial/master/RINANode.jar Node4.properties
```

Now there are two paths of the same cost in DIF1 between IPC1 and IPC3.

8. Stop Node2

Now we stop Node2 process, and the path from IPC1 to IPC3 through IPC2 is broken.



In the terminal for Node2, press “Ctrl +c” on the keyboard to terminate Node2 process. Now after the routing information is updated in DIF1, the route between IPC1 and IPC3 adapts to the path IPC1 - IPC4 - IPC3.

9. Analysis

Now we will analyze route adaptation by parsing the log file at Node1.

1. Stop Node1, Node2, Node3 and Node4.
   You can terminate node processes by pressing “Ctrl+c” on the keyboard.

DO NOT CLOSE TERMINAL WINDOWS

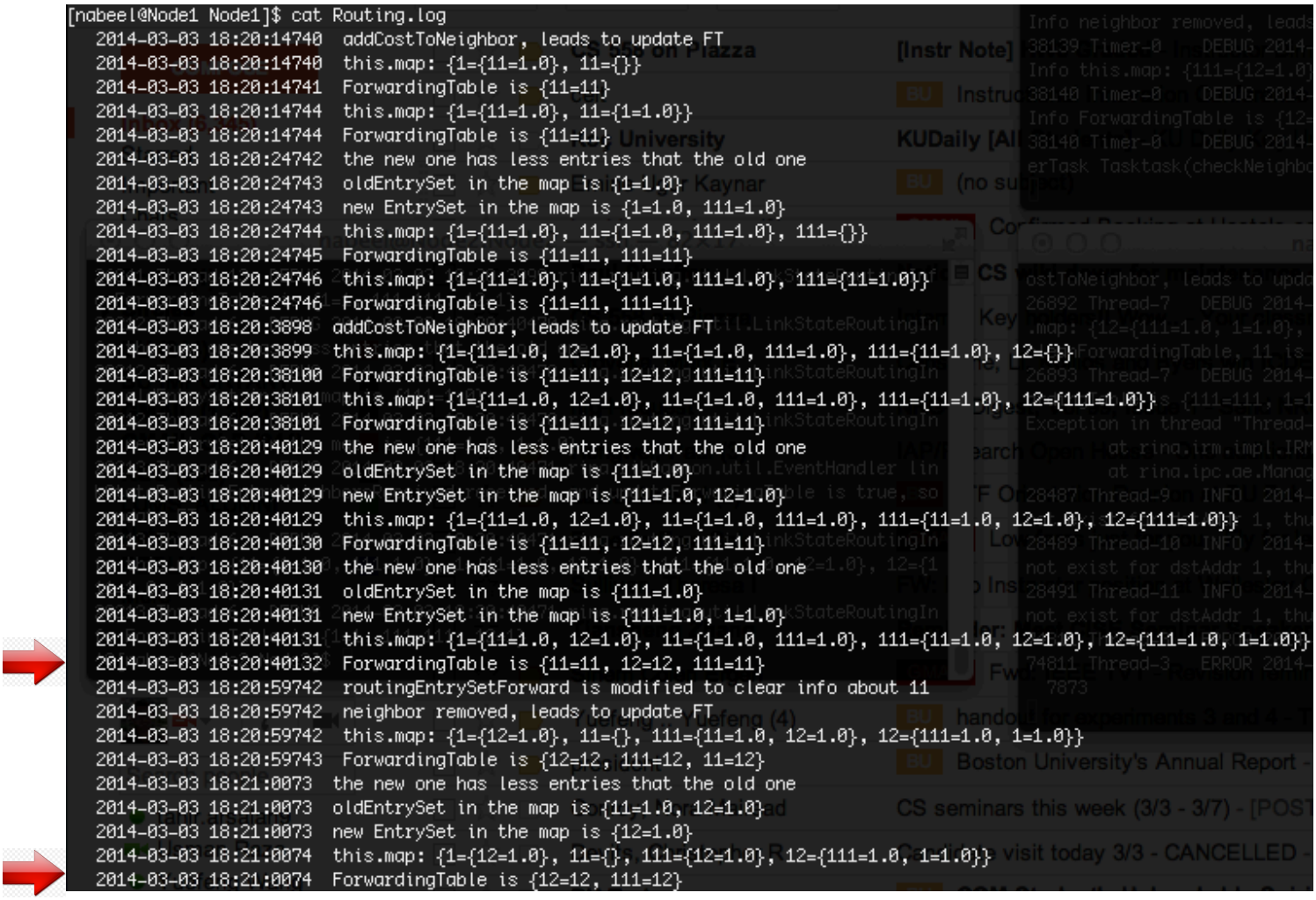
2. We will look at the routing information. Type the following in the terminal for Node1 to parse the log file.

```
/users/Tutorial/master/RINALogChecker.sh -i rinaBU.log -o Routing
```

3. This will generate a log file for all routing information on Node1. To display this log file, type following:

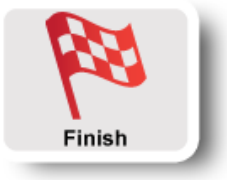
```
cat Routing.log
```

Sample output is shown below.
Note that IPC1, IPC2, IPC3 and IPC4 have been assigned address (1, 11, 111, 12) respectively in DIF1. Observe that the next hop from IPC1 to IPC3 (address: 111) changes from IPC2 (address: 11) to IPC4 (address:12) once IPC2 fails.



```
[nabeel@Node1 Node1]$ cat Routing.log
   2014-03-03 18:20:14740  addCostToNeighbor, leads to update FT
   2014-03-03 18:20:14740  this.map: {1={11=1.0}, 11={}}
   2014-03-03 18:20:14741  ForwardingTable is {11=11}
   2014-03-03 18:20:14744  this.map: {1={11=1.0}, 11={1=1.0}}
   2014-03-03 18:20:14744  ForwardingTable is {11=11}
   2014-03-03 18:20:24742  the new one has less entries that the old one
   2014-03-03 18:20:24743  oldEntrySet in the map is {1=1.0}
   2014-03-03 18:20:24743  new EntrySet in the map is {1=1.0, 111=1.0}
   2014-03-03 18:20:24744  this.map: {1={11=1.0}, 11={1=1.0, 111=1.0}, 111={}}
   2014-03-03 18:20:24745  ForwardingTable is {11=11, 111=11}
   2014-03-03 18:20:24746  this.map: {1={11=1.0}, 11={1=1.0, 111=1.0}, 111={11=1.0}}
   2014-03-03 18:20:24746  ForwardingTable is {11=11, 111=11}
   2014-03-03 18:20:3898  addCostToNeighbor, leads to update FT
   2014-03-03 18:20:3899  this.map: {1={11=1.0, 12=1.0}, 11={1=1.0, 111=1.0}, 111={11=1.0}, 12={}}
   2014-03-03 18:20:38100  ForwardingTable is {11=11, 12=12, 111=11}
   2014-03-03 18:20:38101  this.map: {1={11=1.0, 12=1.0}, 11={1=1.0, 111=1.0}, 111={11=1.0}, 12={111=1.0}}
   2014-03-03 18:20:38101  ForwardingTable is {11=11, 12=12, 111=11}
   2014-03-03 18:20:40129  the new one has less entries that the old one
   2014-03-03 18:20:40129  oldEntrySet in the map is {11=1.0}
   2014-03-03 18:20:40129  new EntrySet in the map is {11=1.0, 12=1.0}
   2014-03-03 18:20:40129  this.map: {1={11=1.0, 12=1.0}, 11={1=1.0, 111=1.0}, 111={11=1.0, 12=1.0}, 12={111=1.0}}
   2014-03-03 18:20:40130  ForwardingTable is {11=11, 12=12, 111=11}
   2014-03-03 18:20:40130  the new one has less entries that the old one
   2014-03-03 18:20:40131  oldEntrySet in the map is {111=1.0}
   2014-03-03 18:20:40131  new EntrySet in the map is {111=1.0, 1=1.0}
   2014-03-03 18:20:40131  this.map: {1={11=1.0, 12=1.0}, 11={1=1.0, 111=1.0}, 111={11=1.0, 12=1.0}, 12={111=1.0, 1=1.0}}
   2014-03-03 18:20:40132  ForwardingTable is {11=11, 12=12, 111=11}
   2014-03-03 18:20:59742  routingEntrySetForward is modified to clear info about 11
   2014-03-03 18:20:59742  neighbor removed, leads to update FT
   2014-03-03 18:20:59742  this.map: {1={12=1.0}, 11={}, 111={11=1.0, 12=1.0}, 12={111=1.0, 1=1.0}}
   2014-03-03 18:20:59743  ForwardingTable is {12=12, 111=12, 11=12}
   2014-03-03 18:21:0073  the new one has less entries that the old one
   2014-03-03 18:21:0073  oldEntrySet in the map is {11=1.0, 12=1.0}
   2014-03-03 18:21:0073  new EntrySet in the map is {12=1.0}
   2014-03-03 18:21:0074  this.map: {1={12=1.0}, 11={}, 111={12=1.0}, 12={111=1.0, 1=1.0}}
   2014-03-03 18:21:0074  ForwardingTable is {12=12, 111=12}
```

Part3: Finish:



You are done with the last experiment. Make sure you terminate all the processes on Node1, Node2, Node3, Node4 and IDD-DNS nodes.

YOU CAN CLOSE ALL TERMINAL WINDOWS NOW.