

## **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](mailto: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 2: Flow Allocation

### Tutorial Instructions:



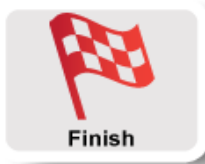
#### Part1: Design/Setup

- Design the Experiment
- Establish the Environment



#### Part2: Execute

- Login to VMs
- Import configuration files
- Start IDD and DNS
- Start Node1
- Start Node2
- Analysis



#### Part3: Finish

## Experiment 2: Flow Allocation

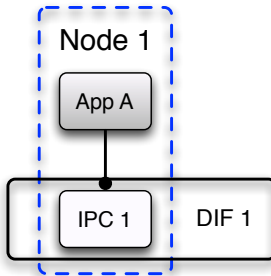
### Part1: Design/Setup:



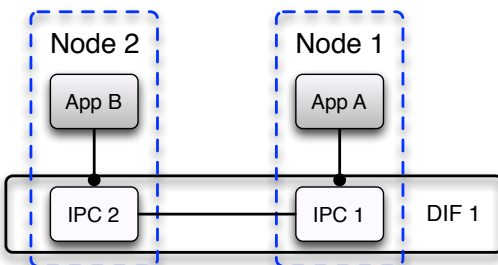
#### 1. Design the Experiment

In this experiment, we use two RINA nodes (Node1 and Node2), and each has one application process and one IPC process.

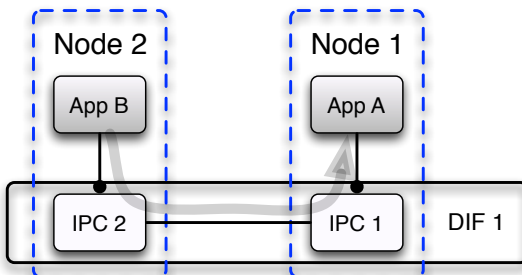
First we start Node1. IPC1 is the only member of DIF1, and AppA uses IPC1 as its underlying IPC process.



Second we start Node2, and IPC2 joins DIF1 through IPC1. AppB uses IPC2 as its underlying IPC process.



Third AppB allocates a connection to AppA using the communication service provided by the underlying DIF1. Once the connection is created, AppB keeps sending simple messages to AppA.



## 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”.

Project Name	Project Lead	Purpose	Slice Count	Create Slice
ProtoRINA	Abraham J. Matta	Experimental Evaluation of Boston University's Prototype of RINA (Recursive InterNetwork Architecture)	1	Create Slice

- 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”.

Slice Name	Project	Slice Expiration	Slice Lead	Actions
RINATutorialSlice1	ProtoRINA	2014-03-11 20:10:11 Z	Nabeel Akhtar	Add Resources, Resource Status, Details, Delete Resources, Launch Flask, GENI Desktop, LabWiki

## 2.2. Check Resources:

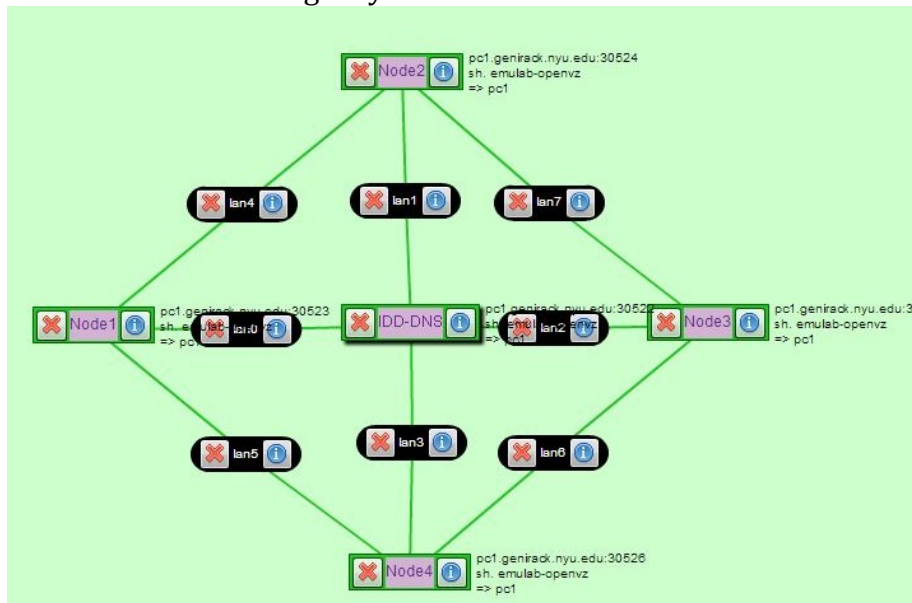
- Click on “Launch Flack” on your slice. That might take a few minutes to display the whole GUI.

Slice Name	Project	Slice Expiration	Slice Lead	Actions
RINATutorialSlice1	ProtoRINA	2014-02-27 22:49:32 UTC	Nabeel Akhtar	<a href="#">Add Resources</a> <a href="#">Resource Status</a> <a href="#">Details</a> <a href="#">Delete Resources</a> <a href="#">Launch Flack</a> <a href="#">GENI Desktop</a> <a href="#">LabWiki</a>



DO NOT DELETE RESOURCES!!!

- You should see the following on your Flack.



## Part2: Execute:




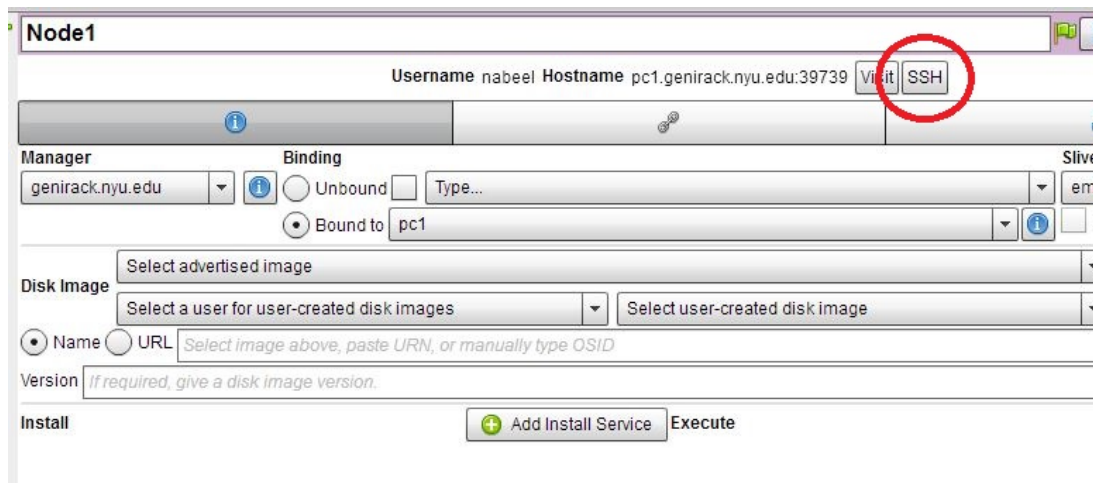
### 1. Login to VMs

In this experiment, we will use three VMs (named Node1, Node2 and IDD-DNS). You can see these three VMs on Flack.


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



- b) Now, Click on  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 two VMs (Node2 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 4 terminal windows now: one for Node1, one for Node2 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 three 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.



Each RINA node process and IPC process has a configuration file which specifies their properties. User can try different settings by changing these configuration files. For RINA node, it includes information about application processes and IPC processes that are residing on this node. For an IPC process, it specifies properties such as underlying DIF information, enrollment policy, routing policies.

The following are part of Node2's configuration file. You can find the file in:

```
group1/Configurations/exp2/Node2/Node2.properties
```

```
service.name = simpleApp
application.name = App
application.instance = B
simpleApp.dstApName = App
simpleApp.dstApInstance = A
simpleApp.num = 10
simpleApp.frequency = 1
underlyingIPC.1 = IPC2
```

Node2 has an application process (AppB) on it, which is a simple application. AppB will send 10 messages to another application process (AppA), one message every second, once there is a connection between them.

## 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 following command to start DNS:

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

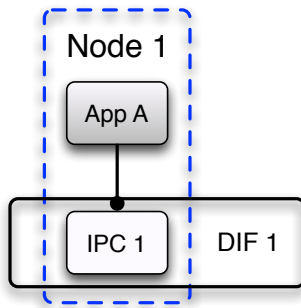
- b) To run IDD, go to directory with the IDD properties file using the second IDD-DNS terminal windows:

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

Now run the following command to start IDD:

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

#### 4. Start Node1



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

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

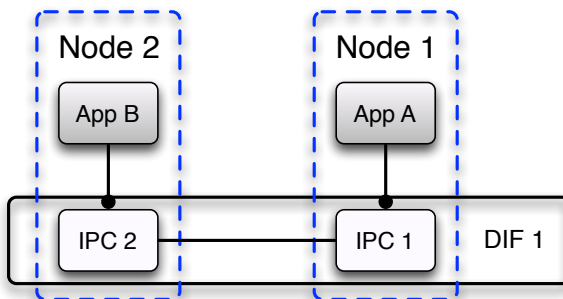
```
cd group1/Configurations/exp2/Node1/
```

- 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



Now we will start Node2 process and begin the enrollment for IPC2.

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

```
cd group1/Configurations/exp2/Node2
```

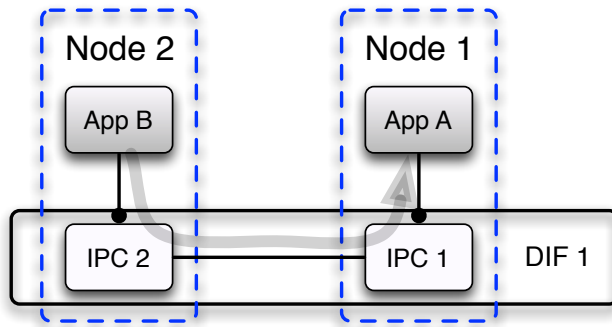
- b) Start Node2 process using the following command:

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

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



Then AppB will allocate a connection to AppA using the communication service provided by the underlying DIF1. Once the connection is created, AppB keeps sending simple messages to AppA.



## 5. Analysis

Now we will analyze application traffic by parsing the log file at Node2.

- a) Stop Node1 and Node2.

You can terminate the node process by pressing “Ctrl+c” on the keyboard.



**DO NOT CLOSE TERMINAL WINDOWS**

- b) We will look at the simple messages sent from AppB to AppA. Type the following in the terminal for Node1 to parse the log file.

```
/users/Tutorial/master/LogChecker.sh -i rinaBU.log -o SimpleApp
```

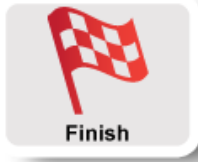
- c) This will generate a log file for all the simple application messages. To display this log file, type the following:

```
cat SimpleApp.log
```

Sample output is shown below:

```
[nabeel@Node1 Node1]$ cat SimpleApp.log
2014-03-03 15:44:34477 Regular handler started
2014-03-03 15:44:34600 Message received, and content is Hello, world: 0
2014-03-03 15:44:35562 Message received, and content is Hello, world: 1
2014-03-03 15:44:36564 Message received, and content is Hello, world: 2
2014-03-03 15:44:37565 Message received, and content is Hello, world: 3
2014-03-03 15:44:38566 Message received, and content is Hello, world: 4
2014-03-03 15:44:39628 Message received, and content is Hello, world: 5
2014-03-03 15:44:40599 Message received, and content is Hello, world: 6
2014-03-03 15:44:41600 Message received, and content is Hello, world: 7
2014-03-03 15:44:42601 Message received, and content is Hello, world: 8
2014-03-03 15:44:43603 Message received, and content is Hello, world: 9
```

### Part3: Finish:



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



**DO NOT CLOSE TERMINAL WINDOWS**

Let's proceed to the third experiment. We will use the same resources for the third experiment.