Understanding the AM API using Content Centric Networking

1. Design the Experiment

a. In today's experiment you will use resources at the aggregate listed on the worksheet. If you don't have a worksheet use *Clemson InstaGENI (aka instageni.clemson.edu)*

2. Establish the Environment

2.1 Pre-work: Ensure SSH keys are setup

Verify that you have at least one public key associated with your account. To do that, after you login to the portal check under your Profile, under the `SSH keys` tab. If you do not have SSH keys associated yet, please follow the instructions on that tab of the Portal.

2.2 Configure Omni

a. Login to the GENI Portal
b. Click on the `Profile tab` link on the top of your screen. Then click on the `Configure omni` tab under `PROFILE`.

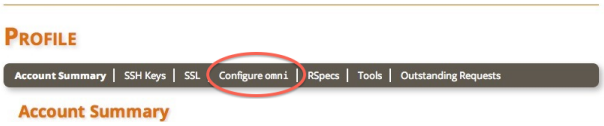

Figure 2-1 Click on the *Configure omni* tab under *Profile*.

c. Click on the `Download your omni data` button under step 2.

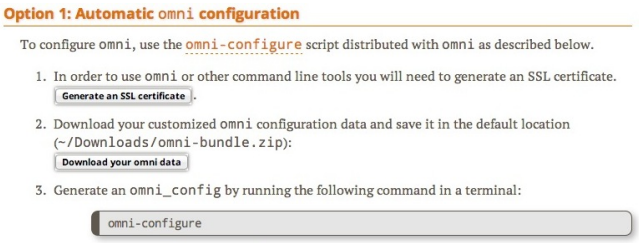

Figure 2-2 *Download your omni data* under step 2.

d. If this is the first time you try to access your GENI certificate you will have to generate one. Click on the `generate a certificate` link.

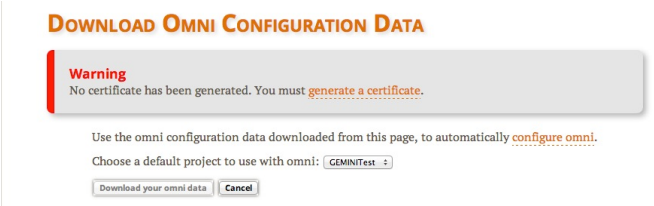

Figure 2-3 Click on *Generate a certificate*.

e. Unless you really understand how SSL certificates work, choose the simple option. Click on the `Generate Combined Certificate and Key File` button and then click on `Close`. You will be taken back to the download page with the warning. Reload the page to enable the download button.

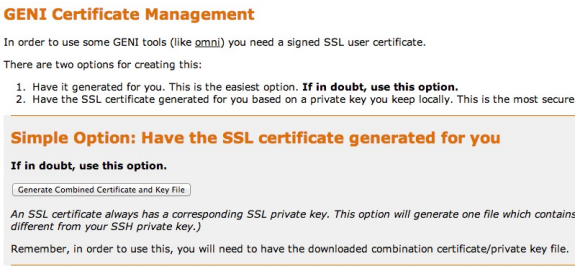

Figure 2-4 Click on *Generate Combined Certificate and Key File*.

f. If you are a member of more than one project, select which project you would like to be the default one for running experiments in GENI. You can always change the project that is used by the `-r` command line option of Omni. Then click on `Download your omni data`.

Figure 2-5 Click on *Download your omni data*.

g. The bundle will be saved at `~/Downloads/omni.bundle`
h. Open a terminal window and type:

```
omni-configure
```

The cert and key files you need will be installed in the appropriate folders.

3. Obtain Resources

3.1 Create a slice

Create a slice using `omni` and the slice name of your choice. From now on that slice name will be referred to as `SLICENAME`.

```
$ omni createslice SLICENAME
```

3.2. Load a simple topology in Jacks

For this exercise, we will edit an existing RSpec file. Start by loading this predefined topology into Jacks.

a. In the Portal, launch Jacks for this slice.
b. From the **Choose RSpec** menu (see figure), select the **URL** button.

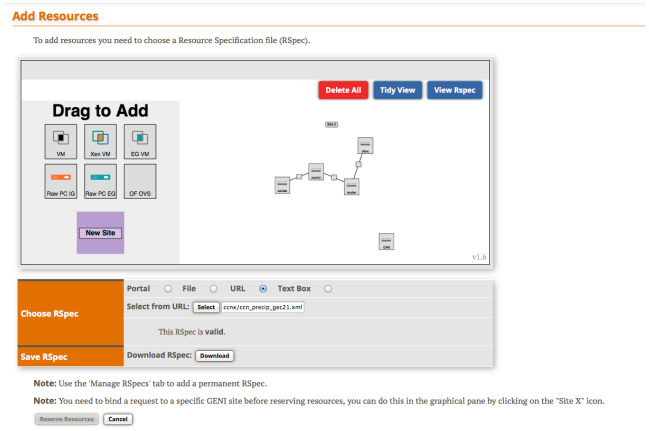

Figure 3-1 *Import an RSpec into Jacks.*

c. Enter the URL for the RSpec:
`http://www.gpolab.bbn.com/experiment-support/ccnx/ccn_precip_gec21.xml`
then click **Select**.
d. After you click **Select**, a network topology should appear on the canvas.

3.3. Modify the RSpec to automatically install and execute CCNX software

For this experiment, we need to install the following software on the nodes:

- The CCNX software (ccnx-0.6.1-F15-x64.tar.gz)
- Scripts that set up the CCNX software (ccnx-setup.tar.gz)
- Scripts used to pull atmospheric precipitation data using the CCNX protocol (ccnx-atmos-F15-x64.tar.gz)

When the nodes start up, we need the following scripts to be executed:

- Script that sets up the node (node-setup)
- Script that sets up the ccnx protocol (ccnx-setup)
- Script that setup up ccnx protocol routes (add-precip-routes)

We automate the installation and running of the software using install and execute scripts in the RSpec.

a. Click on the **collab node** (see figure) to manipulate details of the node configuration.

When you open the node information, you should see a new pane with fields to fill in like in the figure.

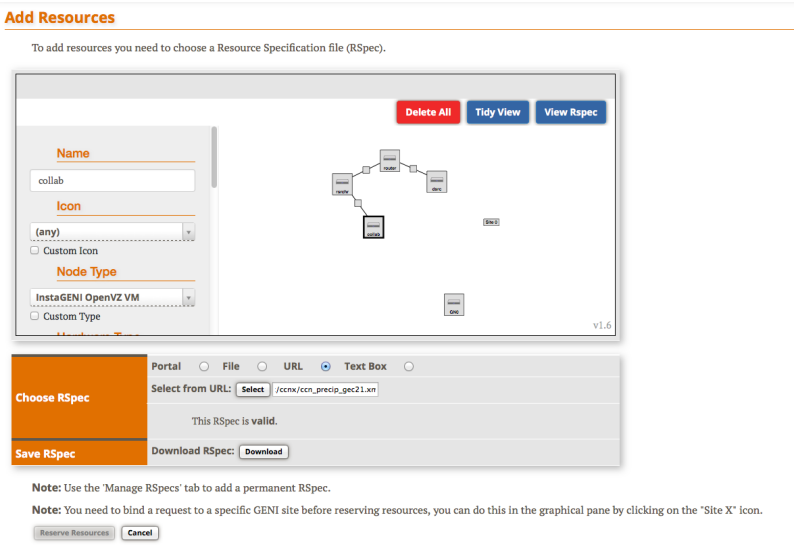

Figure 3-2 *Edit the nodes*

b. The way we will request installation of the proper software and execution of our experiment is to add *install* and *execute* services.

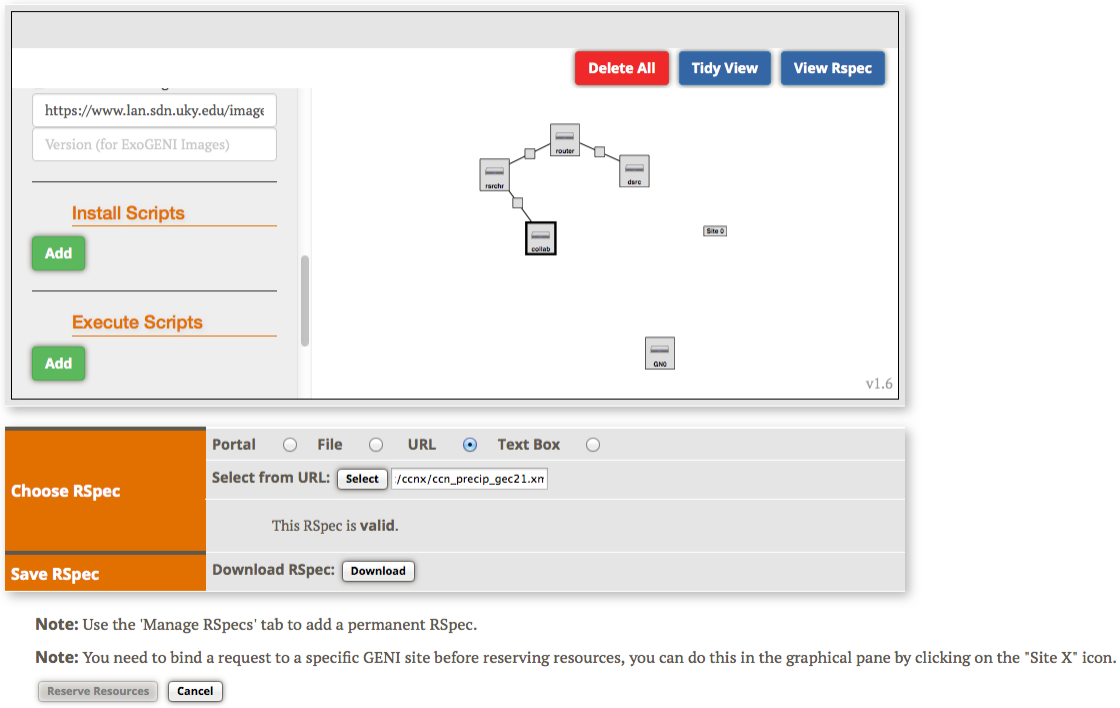

Figure 3-3 *Specify the install and execute scripts*

c. First, select "Add" under the **Install Scripts** sections as shown. In the box in the **URL** field enter (cut-and-paste):

   `http://www.gpolab.bbn.com/experiment-support/ccnx/ccnx-0.6.1-F15-x64.tar.gz`

   and enter

   `/`

   in the box under it (labeled 'Install Path:').

   Now, select "Add" under the **Install Scripts** again. This time add:

   `http://www.gpolab.bbn.com/experiment-support/ccnx/ccnx-atmos-F15-x64.tar.gz`

   and enter

   `/`

   in the box under it (labeled 'Install Path:').

   Similarly, add a third install service:

   `http://www.gpolab.bbn.com/experiment-support/ccnx/ccnx-setup.tar.gz`

   and enter

   `/tmp`

   in the box under it (labeled 'Install Path:'). **Note the different install directory this time.**

d. In a similar manner, use the "Add" button under the **Execute Scripts** section to add three execute services on this node:

```
cd /tmp/ccnx-setup &&  ./node-setup
cd /tmp/ccnx-setup &&  ./add-precip-routes rsrchr
cd /tmp/ccnx-setup && ./ccnx-setup router 4
```

Be very careful when entering this information -- these commands will not be executed yet, so it will be some time before you will see any relevant error messages if there is a mistake here.

e. You *DO NOT* have to specify install and execute scripts for the other nodes as they have already been done for you. You can check this by clicking on the icons for these nodes.

3.4. Export the modified request RSpec

Now we will pull back some of the covers and inspect exactly what Jacks has been doing for us when preparing the RSpecs for the experiments we design. Each node and link has a corresponding element in the RSpec, and the details of the component configuration (such as the install and execute services we requested above) are specified with attributes, or sometimes child elements, within those portions of the document.

a. From the *Add Resources* window (see figure), select the **View Rspec** button on the top right hand side. This will bring up a pane showing the current RSpec -- please take a moment to inspect it. The `<node>` and `<link>` elements contain the specification for the components we will request, and if you look closely, you will be able to see the install and execute service elements you added earlier. Click on the "View RSpec" button again to return to the main screen.

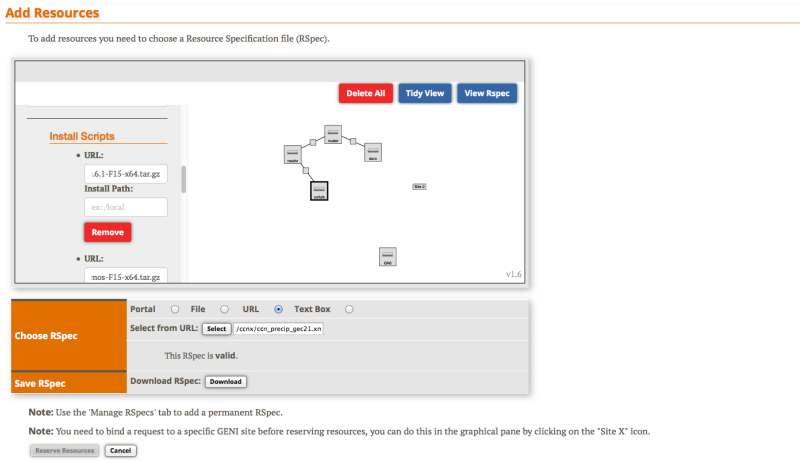

Figure 3-4 View and save the final request RSpec

b. Use the **Download** button (in the lower left part of the screen next to Save RSpec) to make a local copy of your RSpec with the name `rspec.xml`. We'll use this in the next step to demonstrate how other client tools also use RSpec files to communicate requests to aggregate managers.

3.5. Instantiate the new experiment using Omni

For this step, we'll change the approach a bit and switch to a new client tool, the command line Omni client.

From a terminal, please enter the command:

```
$ omni -a AM_NICKNAME createsliver SLICENAME RSPEC_FILE
```

where `AM_NICKNAME` is the nickname for your assigned aggregate manager and `SLICENAME` is the name of the slice you created earlier (both of these are given on your worksheet). `RSPEC_FILE` should be replaced with the filename of the RSpec you saved in step 3.4.

If all is well, Omni should give you a number of informational messages, such as:

```
INFO:omni:Loading config file /home/geni/.gcf/omni_config
```

It should quickly proceed to the point where it makes the request to the remote manager:

```
INFO:omni:Creating sliver(s) from rspec file /home/geni/Downloads/experiments.rspec for slice ..
```

This step can sometimes be time-consuming, so please be patient. If it succeeds, within a couple of minutes Omni should report:

```
INFO:omni: Completed createsliver:
```

and your resource reservation is complete!

Understanding the AM API using Content Centric Networking

4 Wait for resources to be ready

You can tell whether your nodes are *ready* by using a script built on `omni` called `readyToLogin`.

a. Please use the command:

```
readyToLogin -a AM_NICKNAME SLICENAME

where (as before) AM_NICKNAME and SLICENAME are your aggregate
manager nickname and your slice name (both found on your worksheet).
```

b. If it reports that the sliver is not yet ready (for example, it might say that the status is "changing"), then wait a minute or two and try again. Once everything is complete, `readyToLogin` will give output that should look something like this:

```
...
rschr's geni_status is: ready (am_status:ready)
User example logs in to rschr using:
        ssh  -p 32768 -i /Users/example/.ssh/geni_key_portal example@pc1.utah.geniracks.net
User example logs in to collar using:
        ssh -p 32769 -i /Users/example/.ssh/geni_key_portal example@pc1.utah.geniracks.net
...
```

5 Trying out the CCN protocol

The `install` and `execute` services requested in our RSpec have already started, and nodes in our experiment should be running the CCN (Content Centric Networking) protocol. Our experiment consists of:

- A data source (node `dsrc1` that holds precipitation data from the US National Oceanic and Atmospheric Administration (NOAA).
- A researcher node `rsrchr` that gets data from the data source
- A collaborator node `collab` that gets data from the researcher

Key features of the CCN protocol include:

- Data is accessed by name. In our case we use a program called client to get precipitation data by date range (e.g. precipitation between 1901/01/01 and 1901/01/02).
- All nodes cache data for a certain period of time. When a node receives a request for data, it checks its local cache. If the data is in it's cache, it returns that data. Otherwise, it forwards it on to its neighbor.

We verify this caching behavior by:

- Logging into the researcher node and using the client program to get precipitation data for a certain date range. The client displays how long it took to get the data.
- Retrieving the same data again and noting how we get it much faster since it comes out of a cache.
- Requesting data for different date ranges and seeing how long it took to retrieve the data.
- Requesting the data again and note it is retrieved much faster.

If you have time, you can repeat the above steps on the collaborator node.

Note: There is an optional part to this exercise that uses the GENI Desktop to visualize traffic on the links in our network. There you can visualize which data requests went all the way to the data source (node `dsrc1`) and which data requests were fulfilled from a node's cache.

5.1 Run the CCN application

1. Log into the node `rsrchr` using the `ssh` command returned by `readyToLogin`.
2. Once you are logged in, ask for precipitation data from 1 Jan 1902 to 2 Jan 1902:

```
$ /opt/ccnx-atmos/client.py
Start Date in YYYY/MM/DD? 1902/01/01
End Date in YYYY/MM/DD? 1902/01/02
```

3. You should see output that looks like:

```
Asking for /ndn/colostate.edu/netsec/pr_1902/01/01/00, Saving to pr_1902_01_01.
Time for pr_1902_01_01.tmp.nc 1.09802699089=
Asking for /ndn/colostate.edu/netsec/pr_1902/01/02/00, Saving to pr_1902_01_02.
Time for pr_1902_01_02.tmp.nc 4.65998315811=
Joining files..
Concat + write time 0.0735998153687
Wrote to pr_1902_1_1_1902_1_2.nc
```

   Note that it took about 1.1 and 4.7 seconds respectively to retrieve data for Jan 1 and Jan 2

4. Run the client again and request the same data. This time your output should look like:

```
Asking for /ndn/colostate.edu/netsec/pr_1902/01/01/00, Saving to pr_1902_01_01.
Time for pr_1902_01_01.tmp.nc 0.0423700809479=
Asking for /ndn/colostate.edu/netsec/pr_1902/01/02/00, Saving to pr_1902_01_02.
Time for pr_1902_01_02.tmp.nc 0.0388598442078=
Joining files..
Concat + write time 0.0237510204315
Wrote to pr_1902_1_1_1902_1_2.nc
```

   Notice how much faster the data was retrieved this time.
5. If time permits, log into the collaborator node `collab` and run queries from there. (Pick dates in 1901 or 1902.) Notice different data retrieval times depending on whether the data came from the datasource, the cache at `rsrchr`, or the local cache.

5.2 (Optional) Visualize experiment data flows

To use the GENI Desktop to visualize the data flows in your network, continue with the instructions here.

Setup

Next: Finish

Understanding the AM API using Content Centric Networking

7 Cleaning up

As in previous tutorials, it is always good practice to clean up slivers right away so that resources can be released and reused by other experimenters. While other tools (such as Flack and the GENI Portal) can also be used to clean up, no matter which software was used to establish the sliver in the first place, for this example we will demonstrate the clean up procedure using Omni.

The command to use is:

```
omni -a AM_NICKNAME deletesliver SLICENAME
```

where once again `AM_NICKNAME` is the aggregate manager nickname and `SLICENAME` is the name of your slice (both found on your worksheet).

A minute or so later, Omni should respond with:

```
INFO:omni: Completed deletesliver:
```

and some details of exactly what was deleted.

8. Archive Experiment

Congratulations! You have completed the exercise.

Introduction