Tutorial: Advanced Topics in Networking Experiments using GENI

Niky Riga, Sarah Edwards
GENI Project Office

26 July 2011

Sponsored by the National Science Foundation

Outline

- Overview
- ProtoGeni resources
- MyPLC resources
- OpenFlow Resources
- Wrap Up

Tutorial Overview

- Learn what is happening behind the scenes
  - How is an experiment setup in GENI?
  - What are the key participants?
- Why is this useful?
  - Understand what might go wrong
  - Make more complicated experiments, build tools
  - Be brave and use resources that are not integrated yet

Looking behind the scenes Disclaimer

- GENI is evolving fast
  - What is true today, might change tomorrow

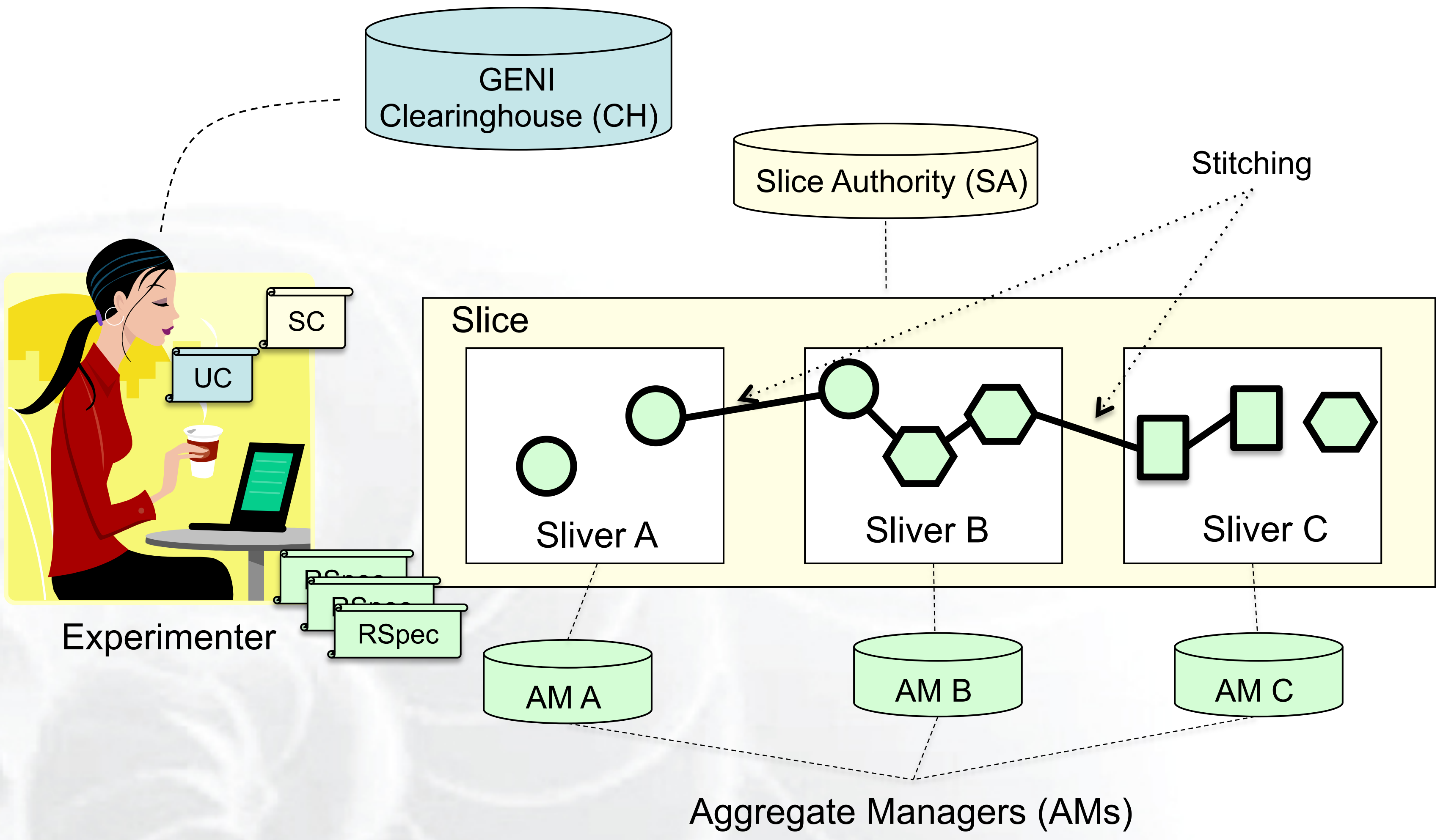
- Use tutorial resources as a pointer

- Best place to get up-to-date info is the GENI wiki

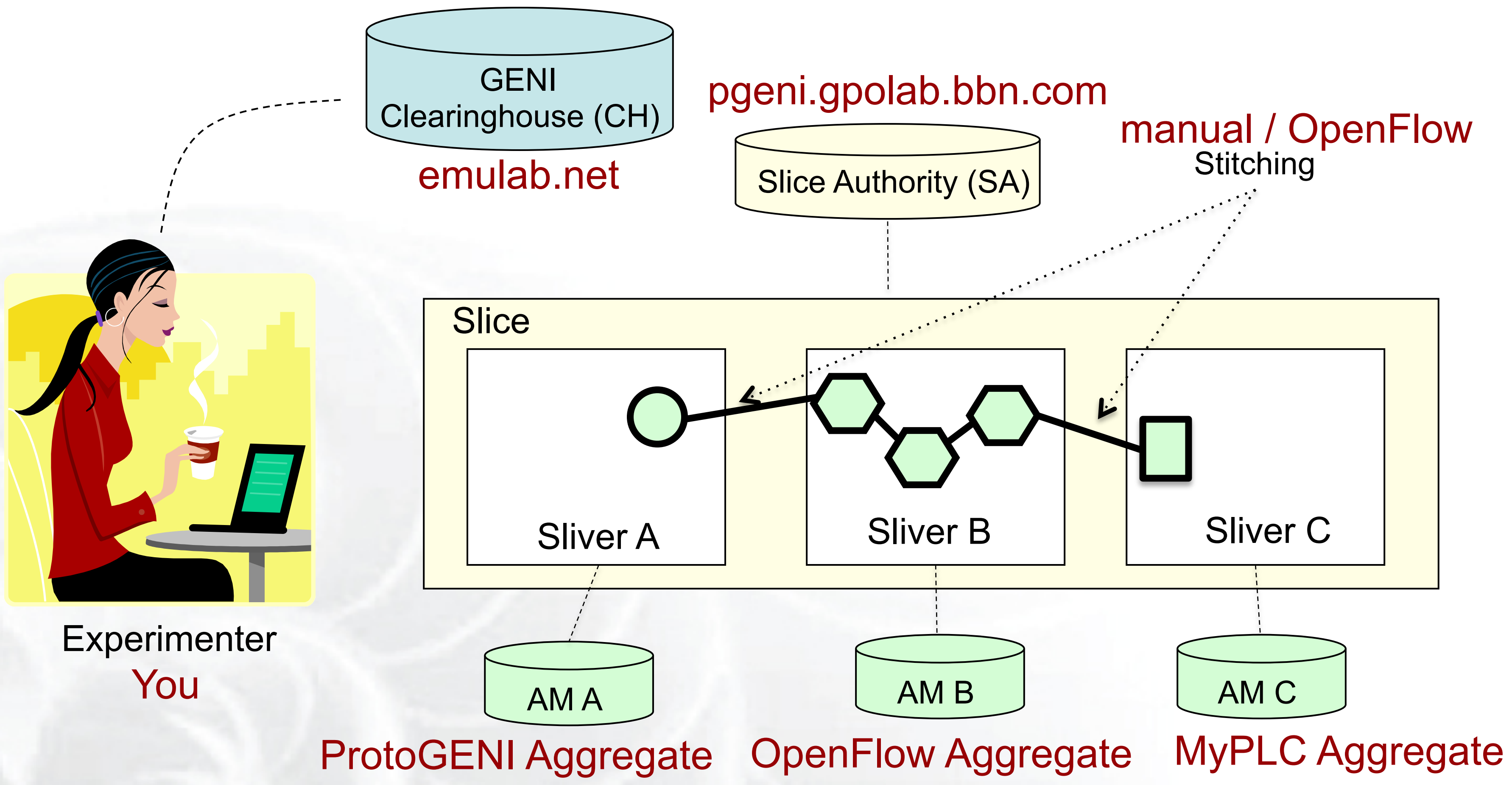
http://groups.geni.net/geni/wiki/GeniExperiments

- Email us with questions (help@geni.net)

Terminology



For the Tutorial

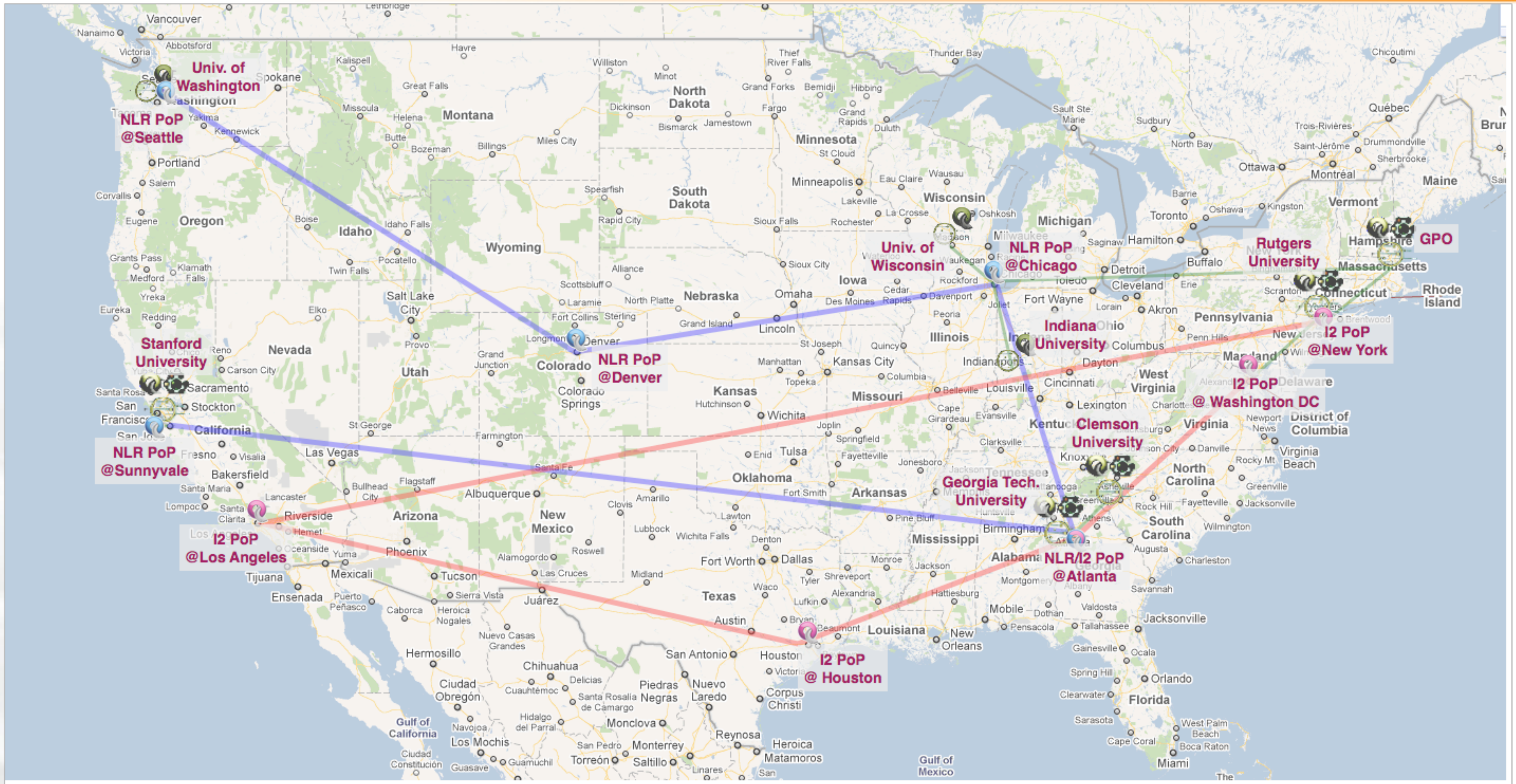


OpenFlow Mesoscale Overview
(code name Tango GENI)

OpenFlow Mesoscale deployment :

- is a prototype GENI infrastructure
- spans multiple sites connected over Layer 2
  - 2 backbone, 7 regionals, 8 campuses
- is open to experimenters that want to gain early access to a Layer 2 infrastructure that combines multiple aggregates.
- includes :
  - OpenFlow aggregates
  - Private PlanetLab aggregates (MyPLC)
  - ProtoGeni aggregates

Where are the tutorial resources?



10 OpenFlow AM (2 backbones (NLR, I2) + 8 campuses)

8 MyPLC AM

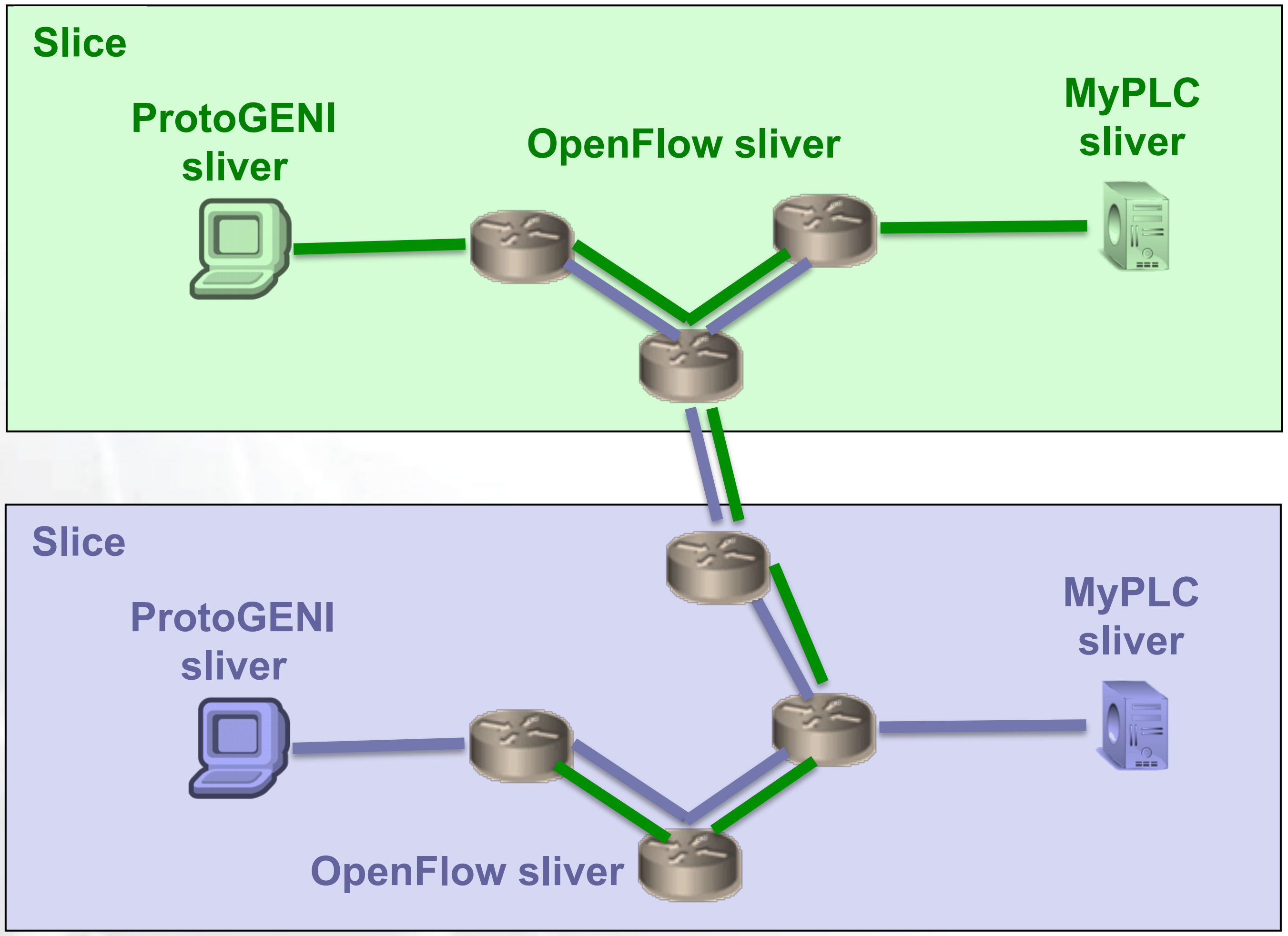
- Clemson, GaTech, GPO, Indiana, Rutgers, Stanford, Wisconsin, Washington

2 ProtoGENI AM (GPO, Utah)

Infrastructure setup

- Separate control and data plane
  - Control plane over commodity Internet
  - Data plane is Layer 2 over GENI backbone
- 2 VLANs over the same resources, providing different topologies
- All hosts have one interface directly connected to an OpenFlow switch
- 10 OpenFlow switches in the backbone

Today's GENI Experiment



geni
Exploring Networks of the Future

Omni: Resource Reservation tool

- A command line experimenter tool
- Create slices and slivers using the GENI AM API
- Written in and scriptable from Python
- Use existing accounts
  - ProtoGENI
  - PlanetLab

- **Works with aggregates that implement the GENI AM API**
  - ProtoGENI, PlanetLab, OpenFlow, …

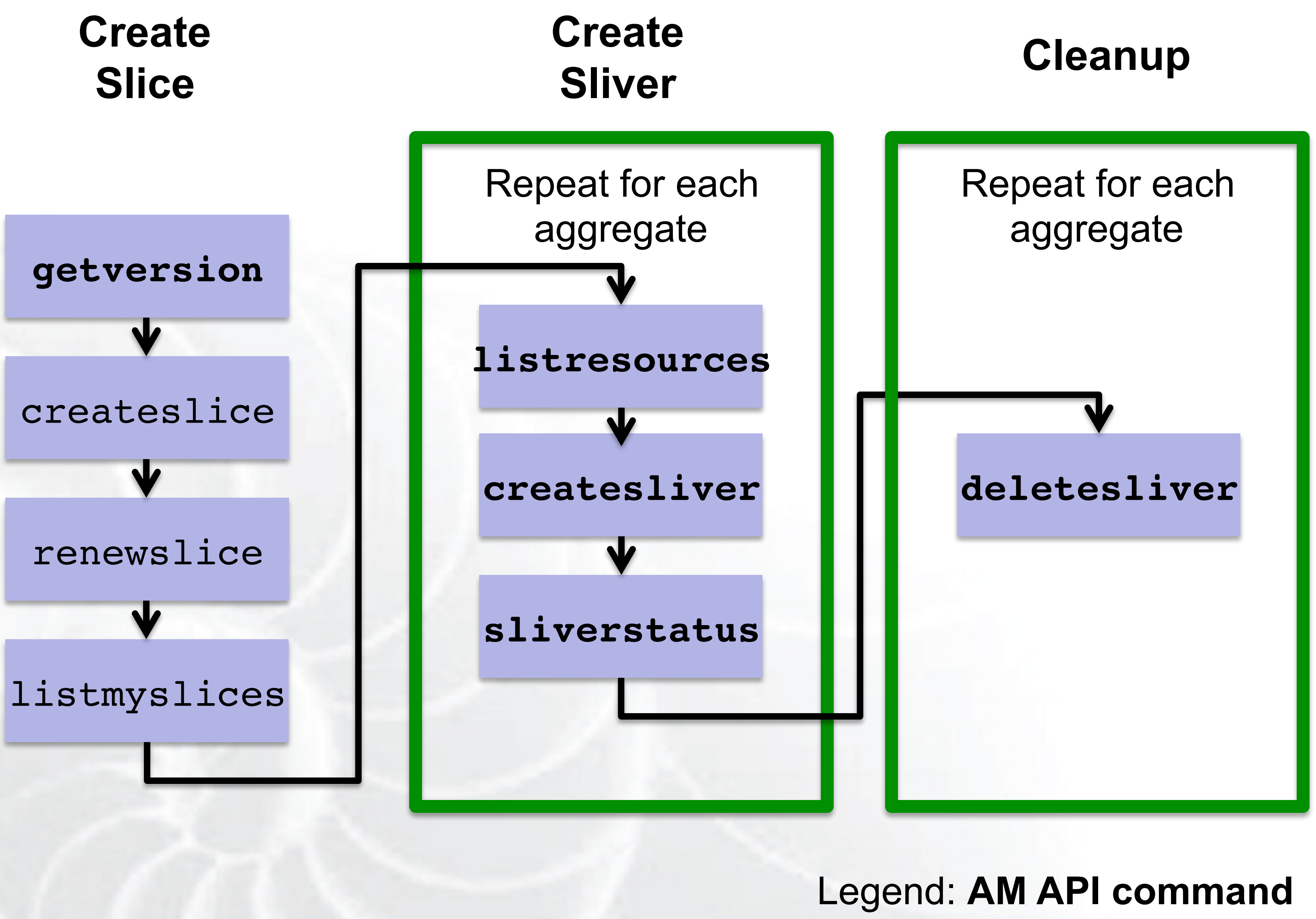
```
$ omni.py createsliver aliceslice myRSpec.xml
INFO:omni:Loading config file omni_config
INFO:omni:Using control framework pgeni
INFO:omni:Slice urn:publicid:IDN+pgeni.gpolab.
          expires within 1 day on 2011-07-07
INFO:omni:Creating sliver(s) from rspec file
INFO:omni:Writing result of createsliver for
INFO:omni:Writing to ‘aliceslice-manifest-rspe
INFO:omni: -----------------------------------
INFO:omni: Completed createsliver:

  Options as run:
                aggregate: https://www.emulab.
                framework: pgeni
                native: True

  Args: createsliver aliceslice myRSpec.xml

  Result Summary: Slice urn:publicid:IDN+pgeni
Reserved resources on https://www.emulab.net/p
  Saved createsliver results to aliceslice-man
INFO:omni: ===================================
```

http://trac.gpolab.bbn.com/gcf/wiki/Omni

AM API

- **Common GENI Aggregate API** which allows experimenters to manage resources
- AM API Commands:
  - GetVersion
  - ListResources
  - CreateSliver
  - DeleteSliver
  - SliverStatus
  - RenewSliver
  - Shutdown

Omni Command Workflow



Legend: **AM API command**

Configure omni_config

```
[omni]
default_cf = pgeni    # Tutorial accounts are on GPO's PG
users = alice         # alice's keys loaded on the VM to allow login

# ---------- Users ----------
[alice]
urn = urn:publicid:IDN+pgeni.gpolab.bbn.com+user+alice
# Really important to get the keys correct!!!
keys = ~/omni_tutorial/ssh/alice_key.pub     #key to load on VM

# ---------- Frameworks ----------
[pgeni]
type = pg
ch = https://www.emulab.net:443/protogeni/xmlrpc/ch
sa = https://www.pgeni.gpolab.bbn.com:443/protogeni/xmlrpc/sa

# Tutorial certificate and key
cert = ~/omni_tutorial/ssh/alice_cert_ct.pem
key = ~/omni_tutorial/ssh/alice_cert_ct.pem
```

Getting Help

- `omni.py –h`

  Lists all commands and their arguments

  Lists all command line options

  Lists Omni version

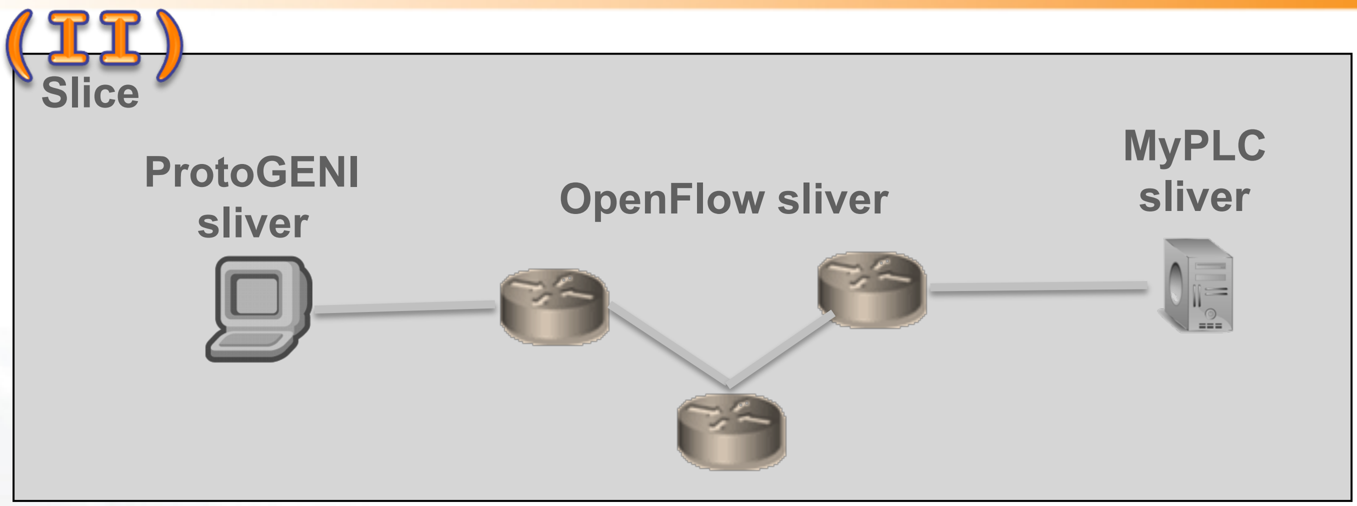
  Lists url to find out more information about Omni

- Omni Troubleshooting page: http://trac.gpolab.bbn.com/gcf/wiki/OmniTroubleShoot

Before we start

- Many people will be accessing the resources, so some calls might fail. Wait a bit and try again!
- Omni is a command line tool, copy-paste is your friend
- You can copy-paste between your computer and the VM.

Let's start …



I. Configure Omni

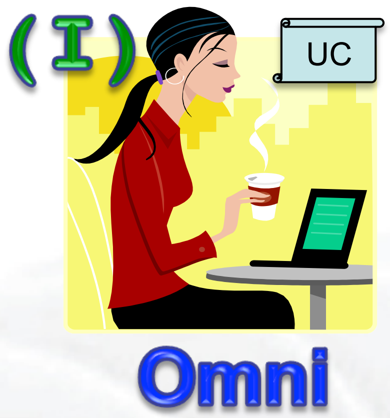
```
        omni.py getversion
```

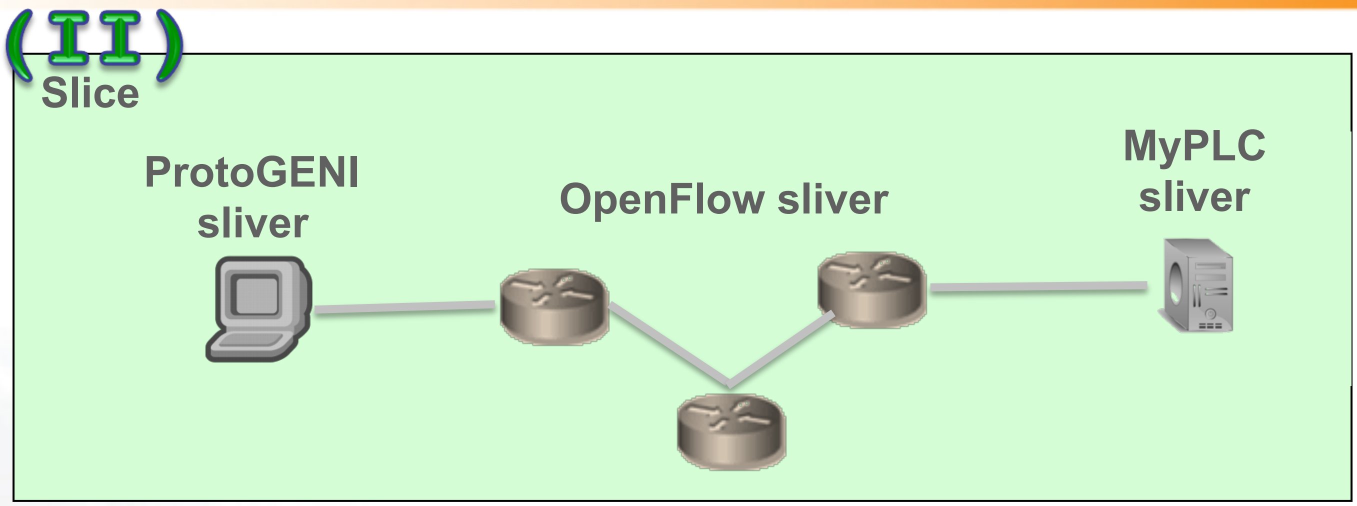
II. Make a slice

```
        omni.py createslice slicename
        omni.py renewslice slicename date
        omni.py listmyslices username
```

Note: `-a aggregateUrl` to specify an aggregate manager
`-o` to save the output to a file

… now …





I. Configure Omni

```
        omni.py getversion
```

II. Make a slice

```
        omni.py createslice slicename
        omni.py renewslice slicename date
        omni.py listmyslices username
```

Note: `-a` *`aggregateUrl`* to specify an aggregate manager
`-o` to save the output to a file

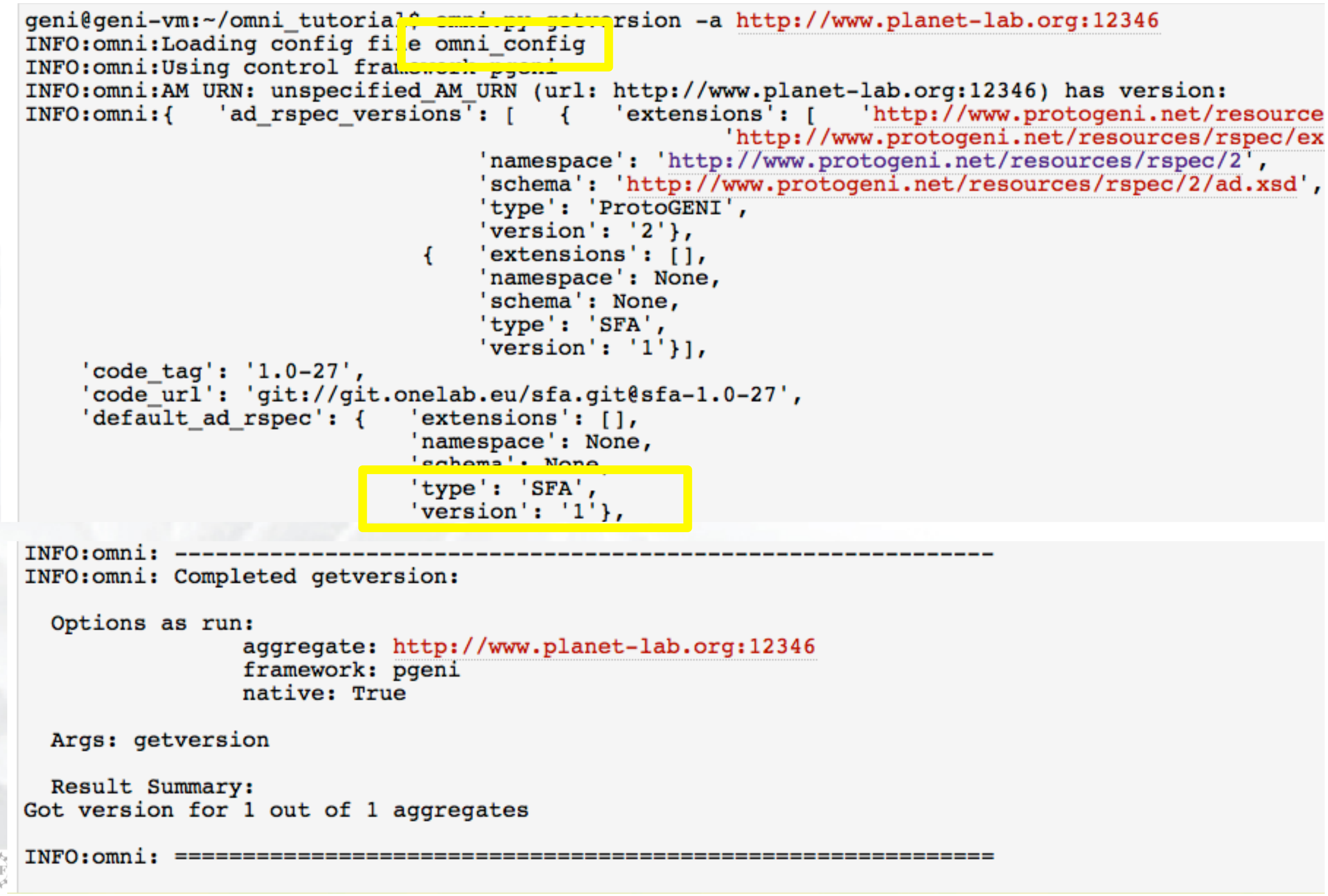
Request Resources

Resource Specification Document (RSpec)

- XML document that describes resources
  - hosts, links, switches, etc
- today 4 different RSpec versions are used
  - **ProtoGENI RSpec v2**
  - ProtoGENI RSpec v0.2
  - PlanetLab RSpecs (SFA)
  - OpenFlow RSpecs

```
<?xml version="1.0" encoding="UTF-8"?>
<rspec xmlns="http://www.protogeni.net/
resources/rspec/2"
       xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
       xsi:schemaLocation="http://
www.protogeni.net/resources/rspec/2
                           http://
www.protogeni.net/resources/rspec/2/
request.xsd” type="request" >
  <node client_id="my-node"
        exclusive="false">
    <sliver_type name="emulab-openvz" />
  </node>
</rspec>
```

omni.py getversion

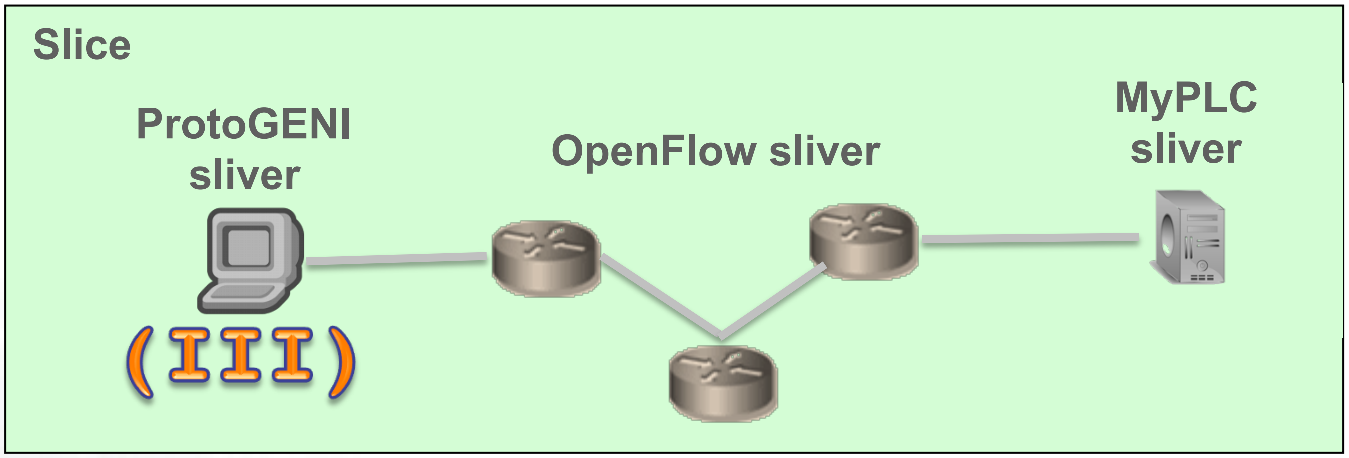


```
geni@geni-vm:~/omni_tutorial$ omni.py getversion -a http://www.planet-lab.org:12346
INFO:omni:Loading config file omni_config
INFO:omni:Using control framework pgeni
INFO:omni:AM URN: unspecified_AM_URN (url: http://www.planet-lab.org:12346) has version:
INFO:omni:{   'ad_rspec_versions': [   {   'extensions': [   'http://www.protogeni.net/resource
                                                          'http://www.protogeni.net/resources/rspec/ex
                                  'namespace': 'http://www.protogeni.net/resources/rspec/2',
                                  'schema': 'http://www.protogeni.net/resources/rspec/2/ad.xsd',
                                  'type': 'ProtoGENI',
                                  'version': '2'},
                              {   'extensions': [],
                                  'namespace': None,
                                  'schema': None,
                                  'type': 'SFA',
                                  'version': '1'}],
    'code_tag': '1.0-27',
    'code_url': 'git://git.onelab.eu/sfa.git@sfa-1.0-27',
    'default_ad_rspec': {   'extensions': [],
                            'namespace': None,
                            'schema': None,
                            'type': 'SFA',
                            'version': '1'},

INFO:omni: ------------------------------------------------------------
INFO:omni: Completed getversion:

  Options as run:
                aggregate: http://www.planet-lab.org:12346
                framework: pgeni
                native: True

  Args: getversion

  Result Summary:
Got version for 1 out of 1 aggregates

INFO:omni: ============================================================
```

ProtoGENI v2 RSpecs

```
<rspec type="request" xsi:schemaLocation="… " xmlns="http://
www.protogeni.net/resources/rspec/2">
  <node client_id="…" component_manager_id="urn:…"
component_id="urn:…" component_name="…" exclusive="true">
   <sliver_type name="raw-pc">
     <disk_image name="urn:…">
   </sliver_type>
   <services>
     <execute command="…" shell="…" />
     <install install_path="…"
              url="…"
              file_type="…"/>
   </services>
  </node>
</rspec>
```

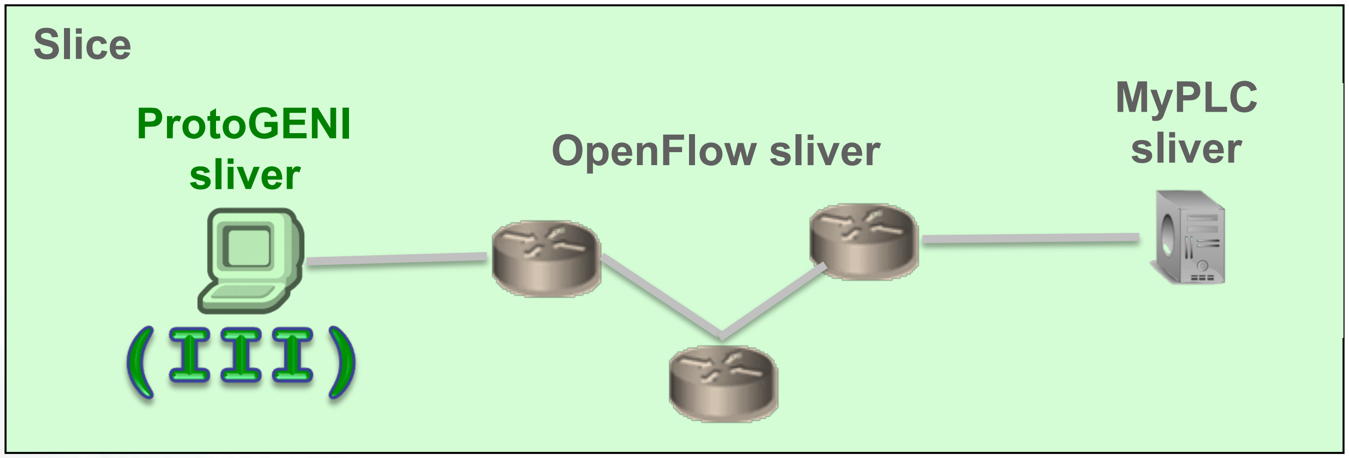
… continue …



III. Make a ProtoGENI sliver

```
omni.py createsliver slicename reqRSpec
omni.py sliverstatus slicename
```

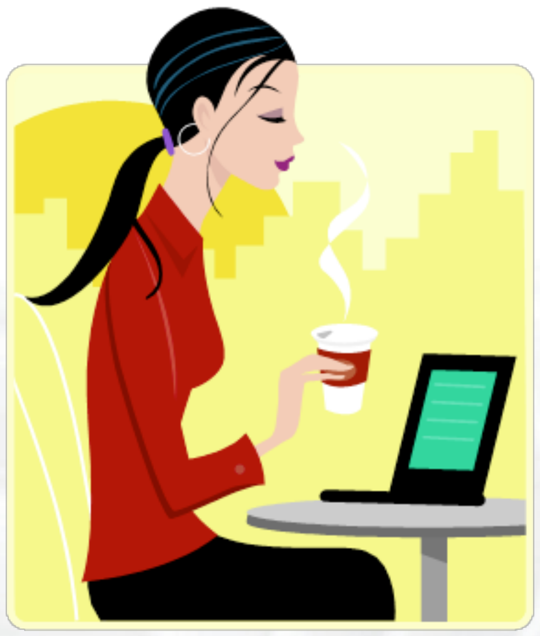
… now …

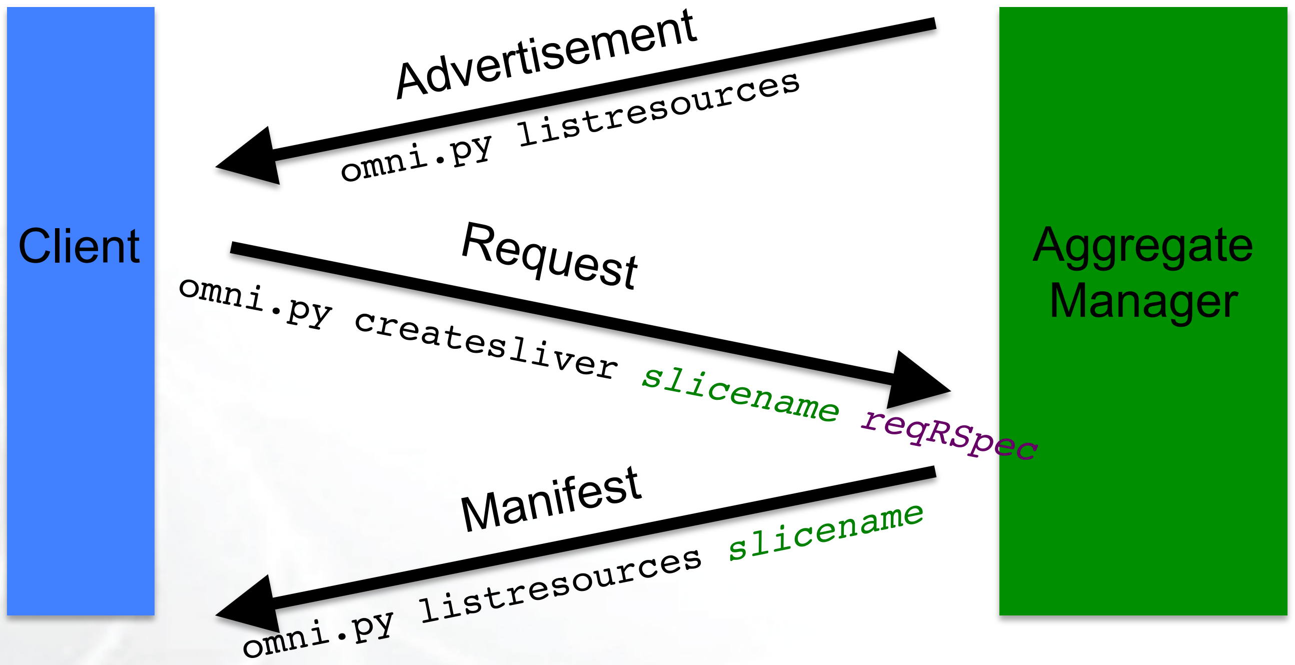


III. Make a ProtoGENI sliver

```
omni.py createsliver slicename reqRSpec
omni.py sliverstatus slicename
```

The Three Types of RSpecs





Advertisement RSpec : What does the AM have?

Request RSpec : What does the Experimenter want?

Manifest RSpec: What does the Experimenter have?

Too many RSpecs ….

- There is an art in writing well formed RSpecs
- Do not try to write one from scratch
  - Find example RSpecs and use them as your base
  - Use tools, like Flack, to generate sample RSpecs for you
  - When appropriate modify advertisement RSpecs

PlanetLab: Modifying an ad RSpec

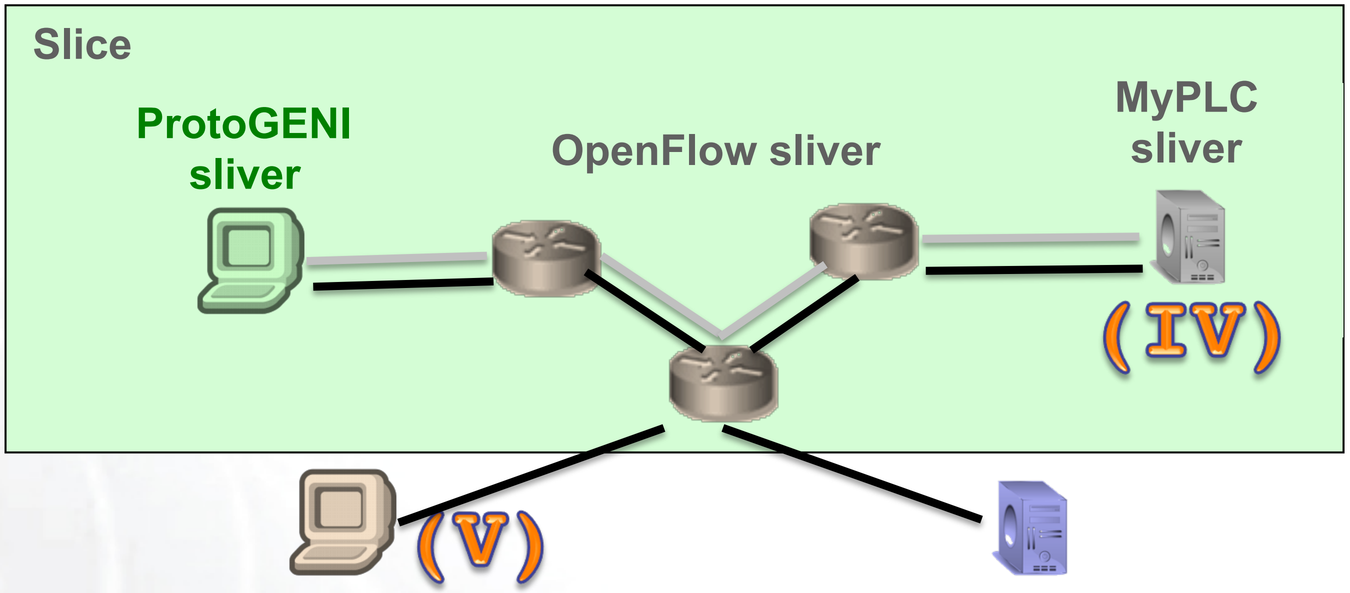
```
<RSpec type="SFA">
  <network name="plc">
    <site id="s1">
      <name>MyPLC</name>
      <node id="n1">
        <hostname> host1.geni.net </hostname>
        <sliver/>
      </node>
      <node id="n2">
        <hostname> host2.geni.net </hostname>
      </node>
    </site>
  </network>
</RSpec>
```

Insert a **<sliver/>** tag in the node tag, for the node you want to reserve

Scripting Omni

- You can write custom Python scripts
  - Call existing Omni functions
  - Parse the Output
- Example: getMyLogin.py
  - Calls sliverstatus
  - Parses output of sliverstatus
  - Determines ssh command to log into node
- More examples distributed with Omni

… continue …



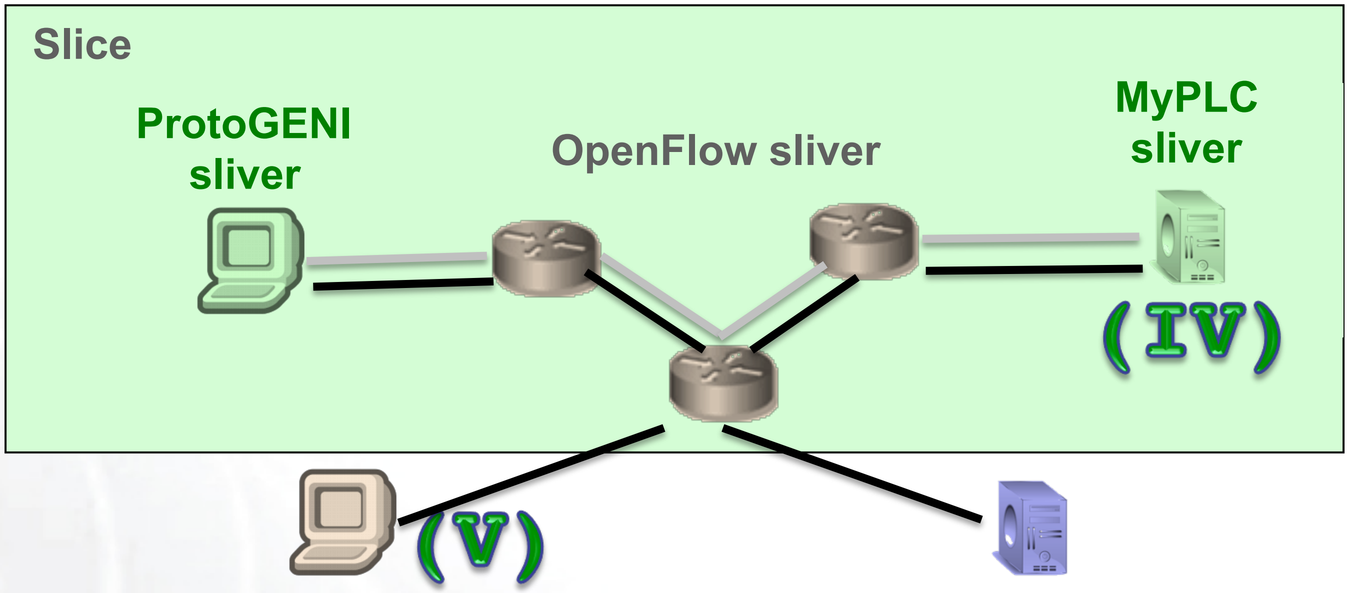
IV. Make a MyPLC sliver

```
omni.py listresources [slicename]
omni.py createsliver slicename reqRSpec
```

Note : After creating your sliver wait for **1 minute** before trying to login. If you can't login for more than **5 minutes** something is wrong

V. Run Layer 2 Ping

… now …



IV. Make a MyPLC sliver

```
omni.py listresources [slicename]
omni.py createsliver slicename reqRSpec
```

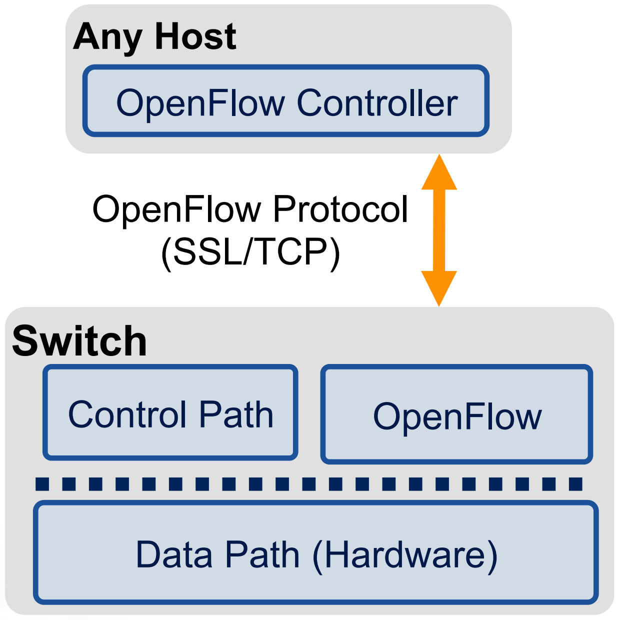
Note : After creating your sliver wait for **1 minute** before trying to login. If you can't login for more than **5 minutes** something is wrong

V. Run Layer 2 Ping

OpenFlow resources

OpenFlow is an API
- Controls how packets are forwarded
- Implementable on COTS hardware
- Make deployed networks programmable

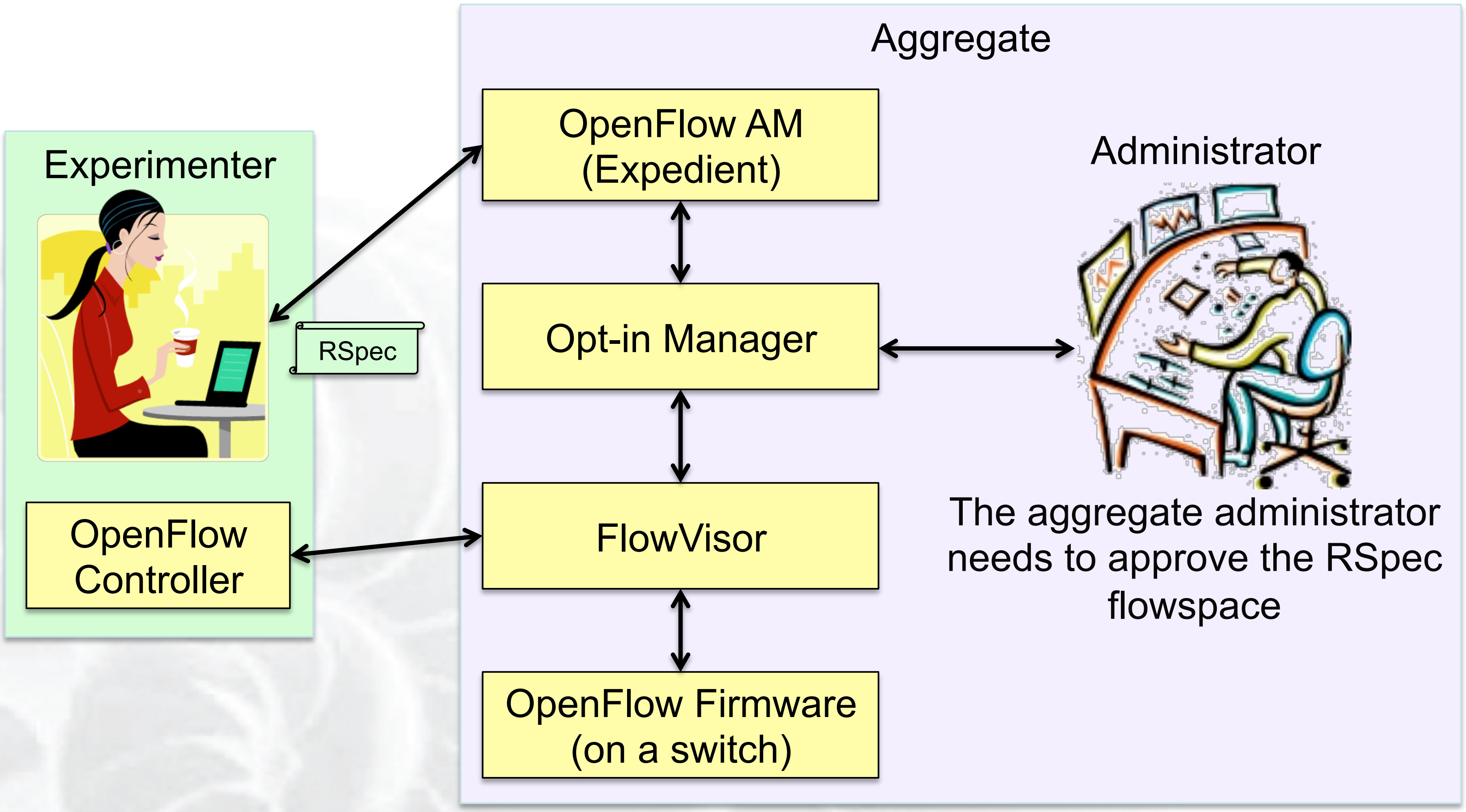
FlowSpace describes packet flows :
- Layer 1: Incoming port on switch
- Layer 2: Ethernet src/dst addr, type, vlanid
- Layer 3: IP src/dst addr, protocol, ToS
- Layer 4: TCP/UDP src/dst port



An experimenter can control multiple FlowSpaces

Modified slide from : http://www.deutsche-telekom-laboratories.de/~robert/GENI-Experimenters-Workshop.ppt

OpenFlow in GENI



The aggregate administrator needs to approve the RSpec flowspace

OpenFlow RSpec

```
<resv_rspec type="openflow" version="2">
  <user … />
  <project … />
  <slice controller_url="tcp:host:port” expiry="1326139200"/>
  <flowspace>
    <switch urn=“urn: … ”/>
    <port urn=“urn: … ”/>
    <pkt_field from=”…" to=”…"/>
</flowspace>>
```

expiry is in Unix timestamp

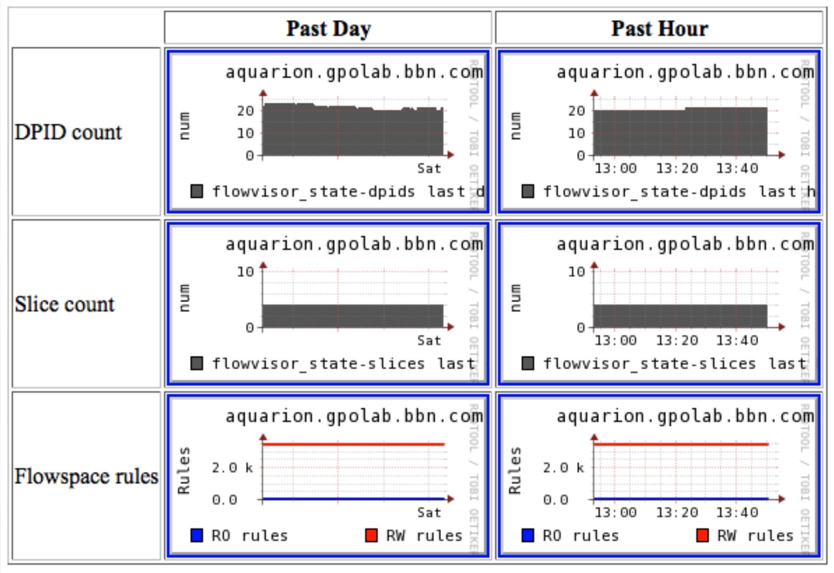
pkt_field can be :

- dl_src, dl_dst, dl_type, dl_vlan
- nw_src, nw_dst, nw_proto, nw_tos
- tp_src, tp_dst

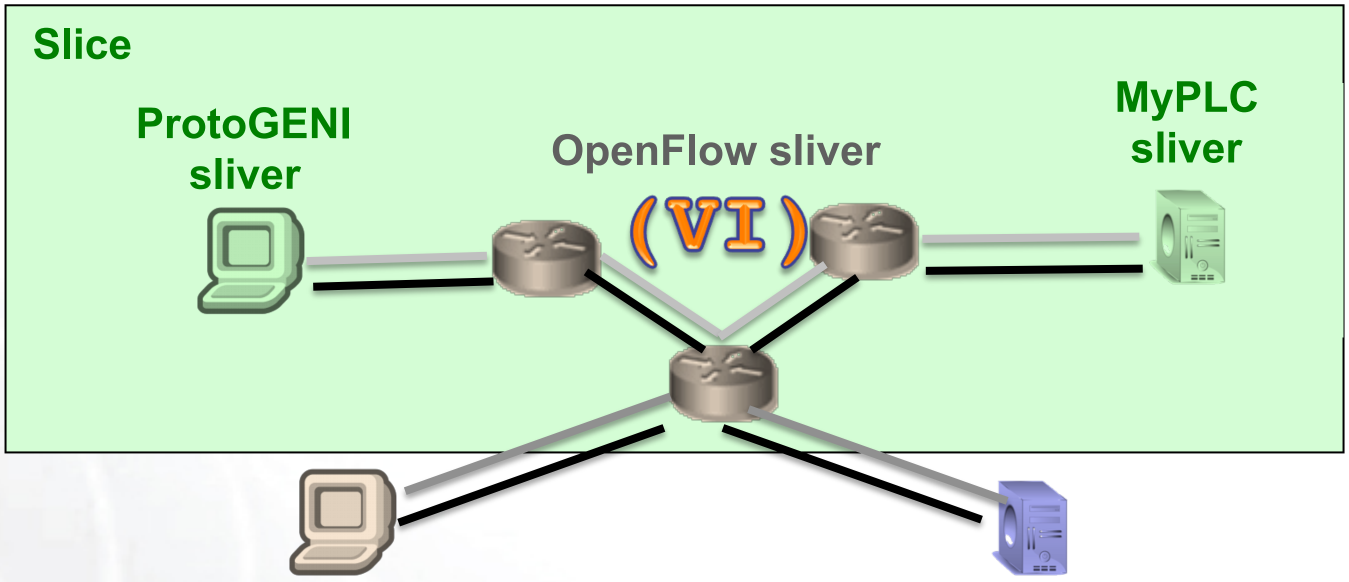
Monitoring

- Each Aggregate is being monitored
- Statistics are collected at GMOC (GENI Meta-operation center)
- Statistics graphed, look at :

  http://monitor.gpolab.bbn.com/tango/



… continue …



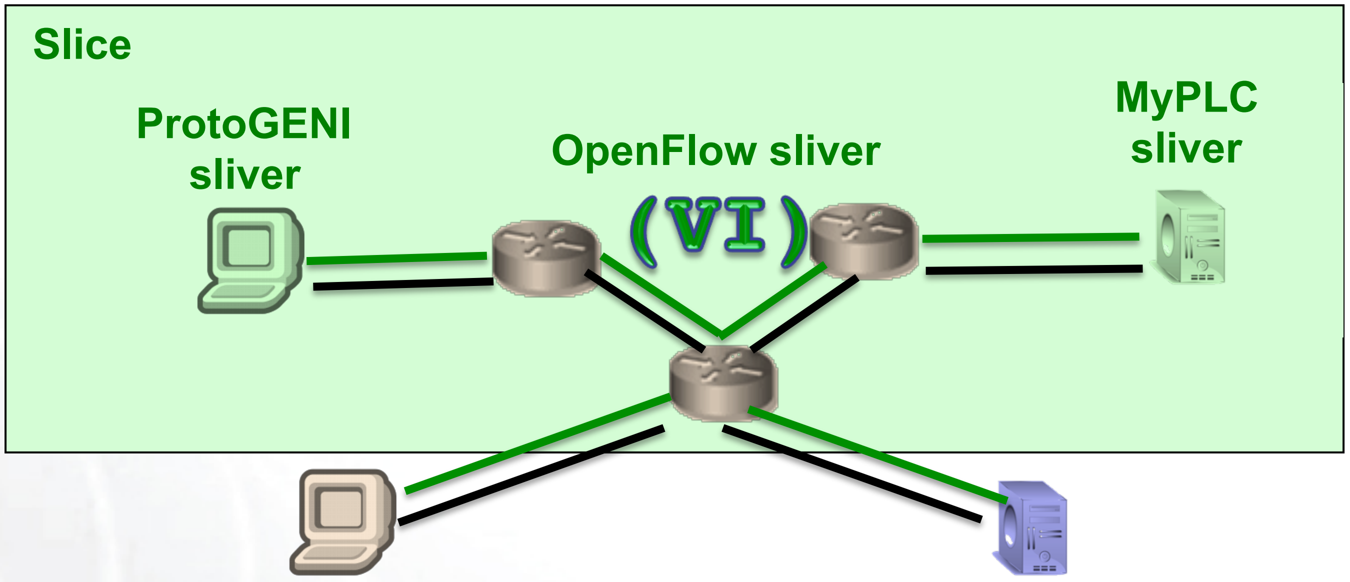
VI. Make an OpenFlow sliver

```
omni.py createsliver slicename reqRSpec
```

After you create your sliver, ask us to opt-in your traffic.

Look at your traffic being opted-in.

… now …



VI. Make an OpenFlow sliver

```
omni.py createsliver slicename reqRSpec
```

After you create your sliver, ask us to opt-in your traffic.

Look at your traffic being opted-in.

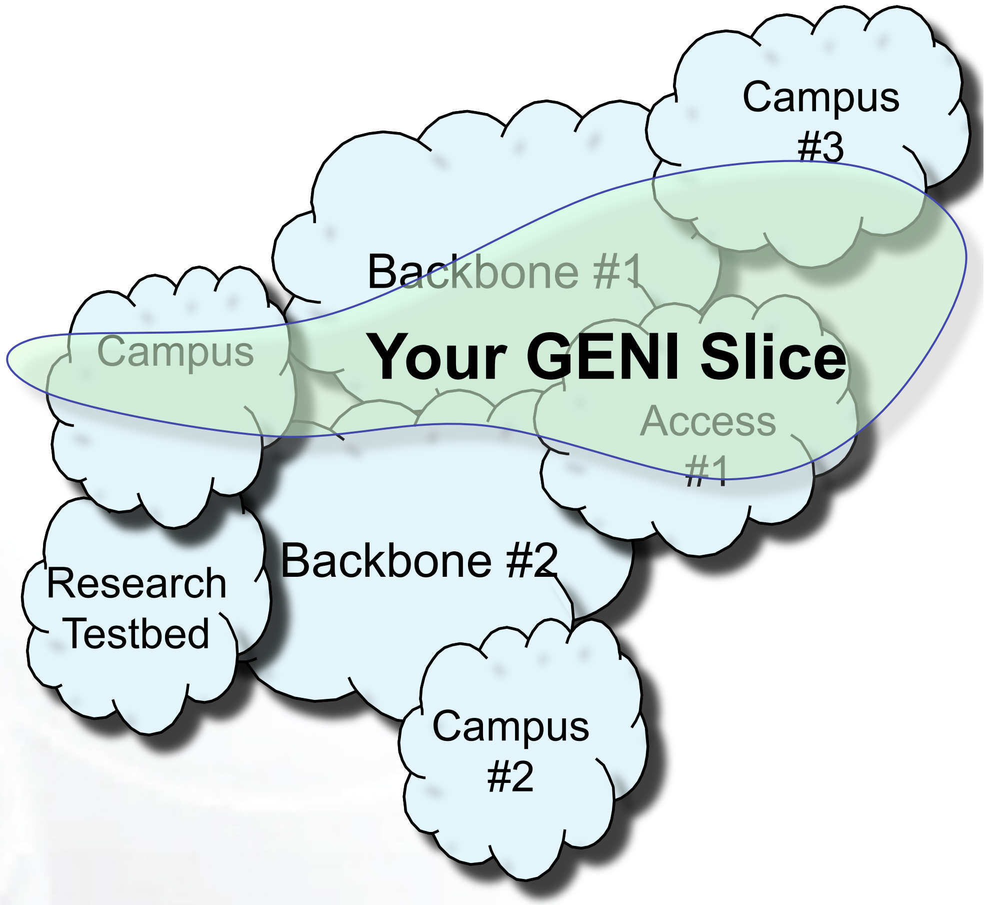
Cleanup

- When your experiment is done, you should always release your resources.
  - Archive your data
  - Delete all your slivers
    - OpenFlow slivers might outlive your slice, make sure you delete them before your slice expires
  - When appropriate delete your slice

Running Experiments on GENI

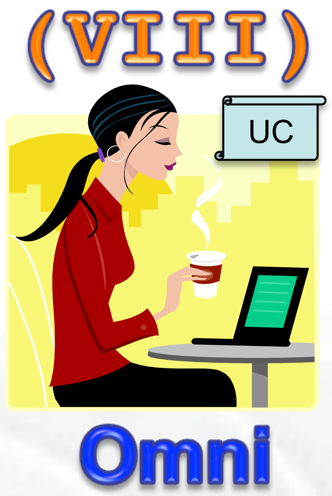
- Get an account to run experiments on GENI

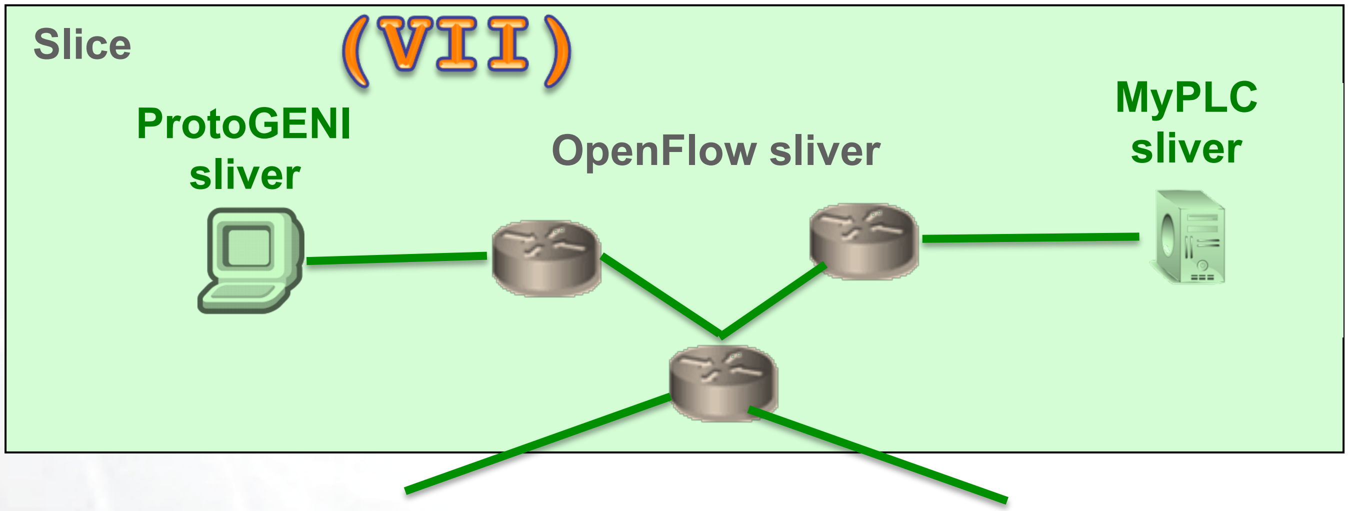
- Contact us at
  help@geni.net



- More information on Experimenter Portal:
  - http://groups.geni.net/geni/wiki/ExperimenterPortal

… continue …





VII. Cleanup resources

```
omni.py deletesliver slicename
```

VIII. Request your own account

– We will help you set up a ProtoGENI user account

… the end!

Omni **Omni**

Happy experimenting!

Backup Slides

Omni Resources

- Primary Information
  - `omni.py -h`
  - Omni Troubleshooting page: http://trac.gpolab.bbn.com/gcf/wiki/OmniTroubleShoot
  - For Omni specific help: gcf-dev@geni.net
  - For general GENI help: help@geni.net

- Omni Wiki (install instructions, documentation, bug reporting): http://trac.gpolab.bbn.com/gcf/wiki/Omni
- For an overview of GENI Experimentation using Omni:
  - http://groups.geni.net/geni/wiki/GENIExperimenter
- Example experiment walk-through:
  - http://groups.geni.net/geni/wiki/GENIExperimenter/ExperimentExample
- Example script walk-throughs:
  - http://trac.gpolab.bbn.com/gcf/wiki/OmniScriptingWithOptions and http://trac.gpolab.bbn.com/gcf/wiki/OmniScriptingExpiration

Omni Commands

- `omni.py getversion`
- `omni.py createslice` *`slicename`*
- `omni.py renewslice` *`slicename`* *`date`*
- `omni.py listmyslices` *`username`*
- `omni.py createsliver` *`slicename`* *`requestRSpec`*
- `omni.py sliverstatus` *`slicename`*
- `omni.py listresources` *`[slicename]`*
  `-t ProtoGENI 2` to request PGV2 Rspecs
- `omni.py deletesliver` *`slicename`*

Other Omni command line arguments

`-c` *`omni_config`* to use another `omni_config`

`-f` *`plc`* to use a different framework

`-t ProtoGENI 2` to specify the version of the Rspec