Getting Started with GENI: A User Tutorial

Jonathon Duerig, Robert Ricci, Leigh Stoller, Matt Strum, Gary Wong
University of Utah
ricci@cs.utah.edu

Charles Carpenter, Zongming Fei, James Griffioen, Hussamuddin Nasir, Jeremy Reed, Xiongqi Wu
University of Kentucky
griff@netlab.uky.edu

This article is an editorial note submitted to CCR. It has NOT been peer reviewed.
The authors take full responsibility for this article's technical content. Comments can be posted through CCR Online.

ABSTRACT

GENI, the Global Environment for Network Innovations, is a National Science Foundation project to create a "virtual laboratory at the frontiers of network science and engineering for exploring future internets at scale." It provides researchers, educators, and students with resources that they can use to build their own networks that span the country and—through federation—the world. GENI enables experimenters to try out bold new network architectures and designs for networked systems, and to deploy and evaluate these systems on a diverse set of resources over a large footprint.

This tutorial is a starting point for running experiments on GENI. It provides an overview of GENI and covers the process of creating a network and running a simple experiment using two tools: the Flack GUI and the INSTOOLS instrumentation service.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Network Topology; C.2.3 [**Network Operation**]: Network Management, Network Monitoring

General Terms

Experimentation, Management, Measurement

Keywords

GENI, Testbed, Virtualization, Instrumentation

1. OVERVIEW OF GENI

GENI, the Global Environment for Network Innovations, is a National Science Foundation project to create a "virtual laboratory at the frontiers of network science and engineering for exploring future internets at scale" (see `http://www.geni.net/`). GENI uses virtualization technology to "*slice*" network resources among multiple virtual networks; many researchers can share the same physical substrate, each running his or her own networking stack within a different slice. GENI includes a wide and constantly-growing range of resources including virtual and physical PCs, national backbone links and routers, programmable routers, mobile nodes, sensor nodes, etc. In addition to a network platform, the GENI project has also developed a large set of tools that help users to design, deploy, conduct, and monitor their experiments. This tutorial will cover two such tools: the Flack slice creation GUI (`http://www.protogeni.net/flack2`) and the INSTOOLS instrumentation and monitoring system (`http://www.netlab.uky.edu/p/instools/`). Pointers to other network resources and tools provided by GENI can be found in Section 5.

GENI is a *federated* facility: it is built from a number of *aggregates*, each of which offers computation, storage, or networking resources (or a combination of all three). Each aggregate contributes resources to the GENI user community, but maintains its own autonomy. Current aggregates in GENI include clusters, fiber networks, wireless deployments, campus networks, and PCs connected to edge networks. Most aggregates provide communication channels to the Internet, allowing them to interact with networks outside of GENI, and to provide services to end users.

While most aggregates have their own custom user interfaces, they all offer a uniform API called the *GENI Aggregate Manager (AM) API*. The AM API provides the ability to *discover* the set of resources (called components) that an aggregate has available, and *allocate* those resources to users. Several user tools—such as the Flack GUI used in this tutorial—have been built on the AM API.

To run an experiment or service in GENI, a user creates a *slice*: a container for the set of resources that comprise the user's network. Slices are generally isolated from one another, typically using virtualization technology; though the details of this isolation differ from aggregate to aggregate, the result is that many different experiments, using different protocols, architectures, or parameters, can be explored simultaneously on top of the same physical substrate. For example, servers may be sliced using virtual machines, and Ethernet links may be sliced using VLANs. A slice may exist entirely within a single aggregate, or it may span aggregates. The lifetime of a slice varies from a few hours to many months depending on the experiment being conducted.

In GENI, the definition of an "experiment" is not set by the system, but is up to the experimenter. An experimenter may start with small, controlled experiments using synthetic workloads, much like the experiment described in this tutorial. As wrinkles in the system under test are ironed out, these experiments can expand in scale and scope, running on more devices with greater hardware and geographical diversity. Many parts of GENI allow experimenters to intentionally cause failures or degrade their network in a controlled way in order to study the robustness of their system. Finally, when the experimenter is confident in their system, they may open it to end users for a live deployment study.

Because the set of resources offered by GENI is so diverse, GENI does not prescribe a common method for accessing or programming the resources in slices once they are allocated, preferring instead to expose each component's natural access method and programming interfaces. Many GENI resources are real or virtual PCs, running Linux or an operating system of the experimenter's choice. These components are often accessed using `ssh` and run code compiled against standard Linux libraries. Others have specialized interfaces: for example, slices on many campus networks are controlled using the OpenFlow controller protocol [2].

The remainder of this tutorial describes how to run a simple experiment using the diverse set of resources provided by ProtoGENI aggregates (`http://www.protogeni.net/`), one collection of aggregates in the larger GENI ecosystem. Section 2 covers the process of getting a GENI account, reserving a set of resources using the Flack GENI GUI, and installing software on them. Section 3 then explains the process of running instrumentation tools in that slice as a way to monitor its progress and archive the data for later use. Section 4 walks through a sample experiment, showing how to allocate resources to it and set it up. Finally, Section 5 provides pointers to additional reading materials and online resources.

2. USING GENI

For this tutorial, we will use the Flack GENI GUI to discover, reserve, access, and use a GENI slice. Flack can be found at `http://protogeni.net/flack2`.[1] Because Flack was developed as part of the ProtoGENI project, our examples focus on resources provided by ProtoGENI aggregates. Flack also supports the GENI AM API, meaning it can also be used to reserve resources from other aggregates such as PlanetLab (`http://planet-lab.org`); we encourage the reader to experiment with resources from other aggregates in addition to the ones described in this tutorial. We begin by describing the steps needed to obtain a GENI account, followed by the process of defining a slice with Flack.

2.1 Obtaining a GENI Account

To use GENI resources, you must have a relationship with an *Identity Provider (IdP)* who will vouch for your identity and a *Slice Authority (SA)* who takes responsibility for your actions on GENI. In practice, these two entities are commonly combined, meaning that you may simply obtain an account on one of the federated facilities that make up GENI. Accounts registered at one facility are typically accepted GENI-wide. For this tutorial, we recommend visiting `http://www.emulab.net` and using the "Request Account" link. If you are the first person from your team to request an account, you will need to start a new "project"; otherwise, join a project that a colleague or instructor has already created. When filling out the account creation form, be sure to provide a pass phrase for your GENI certificate and supply an `ssh` public key; both of these are required to use GENI. Note that project applications on Emulab go through a basic review process, so your account may not be ready immediately. When it is, you will receive email, and a set of GENI certificates and credentials will automatically be created for you. Flack is able to retrieve these credentials from Emulab sites; during this tutorial, you will not have to deal with them yourself. However, they are available to you directly should you wish to use other user tools or build your own.

2.2 Finding GENI Resources

You can use Flack to discover what GENI resources are available, select resources to use, and then instantiate your slice. Flack is a Flash application, so you will need a recent version of the Flash plugin (10.2+) installed in your browser. After installing Flash, navigate to `http://protogeni.net/flack2`.

You can select "Tutorial" to get an online tutorial similar to the one described in this paper. The online tutorials contain additional screenshots to help guide you. If you are still waiting for your account to be approved, or are not yet ready to apply for one, you can use the "View Resources" button to explore some of GENI's resources. However, this mode is "read only", and you will be unable to create slices or reserve resources.

To allocate resources and run experiments, use the "Log in" button. Flack can download the users' GENI credentials from the authority where the user has an account. Select the Emulab site that you used in the previous step (likely `emulab.net`), then click the "Download" button near the top of the window. You will need to be logged into your authority to retrieve your credentials; if you are not already logged in, Flack will direct you to your authoritavite web site where you should login. After logging in, return to the Flack tab and click "Download" to try again to download your credentials. If the download is successful, you will see your SSL certificate appear in the login window. There is one final phase of authentication: the SSL certificate is stored in an encrypted form, and you will need to type the passphrase you used when applying for your account in order to decrypt and use your certificate.

After verifying your passphrase, Flack begins sending out requests to discover the set of resources known to ProtoGENI. This may take a few minutes, as it involves contacting each aggregate to request an "advertisement" of the resources the aggregate offers. As aggregates respond with their advertisements, you will see icons appear on the map.

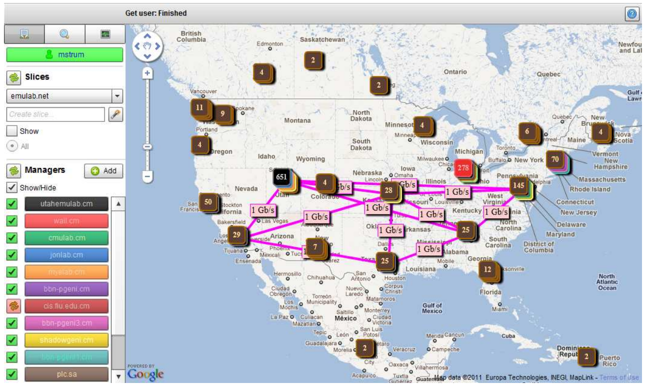

Figure 1: Screenshot of Flack, showing the map view of resources on the right, and a list of the user's slices and GENI's aggregate managers on the left.

GENI includes aggregates from all over the world, and Flack's main view is a map of those resources (shown in Figure 1). Each icon lists the number of resources available at that location. Where icons overlap, they are shown as a stack on the map, with the number on the stack indicating how many nodes exist at that location. The aggregate a resource belongs to is indicated by the icon's color; this corresponds to the colors in the Aggregate Mangers list on the left side of the Flack window.

We encourage the reader to take some time to explore this view and the information that it offers. Clicking an icon in the map will list the set of resources available at that location. Individual nodes will display their type information, what operating system images (if any) may be loaded on that node, its network interfaces, and the connectivity between those interfaces and the rest of the GENI network. Many nodes permit exclusive use, allowing you access to the "raw hardware". Other nodes are sliced or sliceable using a variety of virtualization technologies. Most nodes are PCs, but some are switches, NetFPGA cards, hardware routers, or other kinds of networking equipment that can be incorporated into experiments. Should you wish to learn more about the details of the GENI APIs

[1]This tutorial uses version 2 of Flack, as found at the URL above. The latest version of Flack can aways be found at `http://protogeni.net/flack`.

themselves and their associated data structures, icons on the node windows and elsewhere allow you to see the raw representations maintained internally by Flack.

2.3 Creating a Slice

Flack goes beyond simply examining the resources available in GENI; it is also used to create slices, to allocate resources to them, and to manipulate those slices once they have been created.

To create a new, empty slice, type a name in the text box in the upper left corner labeled "Create slice...", then click the wand to the right. Slice names must be unique. We recommend including your user name in the slice name to avoid collisions.

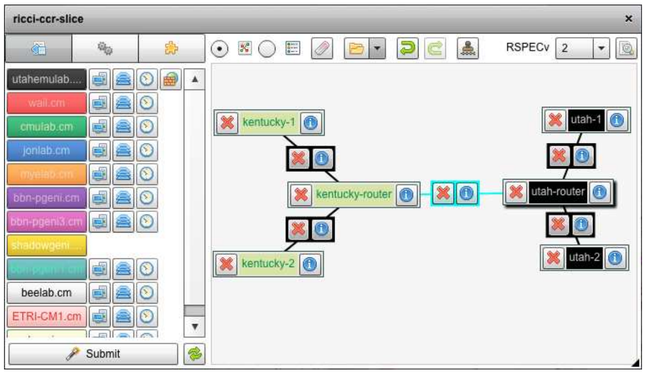

Figure 2: Screenshot of the Flack slice window, showing canvas on the right and node palette on the left.

Resources can be added to the slice through the "slice window" which appears automatically after clicking the wand. (You can also bring it up for an existing slice by clicking on the slice's name in the main Flack window.) The "slice window" (shown in Figure 2) features a canvas for drawing the desired network. The nodes to drag onto this canvas come from the palette to its left. There are three ways to drag resources onto the canvas.

The first method selects *arbitrary nodes* from an aggregate, thereby avoiding the need to identify the specific resources to be used. For example, the Utah aggregate and the Kentucky aggregate both contain a large number of general purpose nodes (PCs) that are roughly equivalent, so it does not matter which node from the aggregate is used. To enable this style of node selection, the palette starts out with a list of all known aggregates. Next to the aggregate names are icons representing arbitrary nodes of type: "raw," "exclusive VM," "shared VM," and others. Dragging any of these icons onto the canvas requests an abitrary node of that type from the specified aggregate.

Selecting a "raw" node allocates an entire physical PC to the slice, providing the ability to load any operating system and control the node with root access. The "VM" types give you a virtual environment which can either be exclusive or shared. "Exclusive VM" nodes are hosted on physical nodes which are not shared between slices, so they are useful when you want to minimize the potential for interference from other users. "Shared VM" nodes run on physical hardware shared by multiple simultaneous slices.

The second method of node selection allows you to request *specific* nodes from an aggregate manager. Clicking on the name of an aggregate in the palette will cause the palette to change to a list of the nodes available from that aggregate. Nodes with an "X" next to them are already reserved (or are unavailable for some other reason); other nodes have icons next to them according to whether they are available for raw, exclusive, or shared allocation, as described above. Any available node can be dragged onto the canvas.

The third method of node selection is *geographical* allocation. Dragging a marker from the map onto the palette populates the palette with the list of nodes at that location. The desired node from the specified location can then be placed on the canvas.

After dragging nodes onto the canvas, you can examine and modify their settings by clicking the "i" (information) button on the node. This reveals a dialog box in which you can change which physical node will be used, change the sharing and virtualization options, or bind your node to a particular type of hardware. Some nodes allow you to specify which operating system image will be used. You can also specify a URL with a tarball to be automatically downloaded and installed on the node, and commands to be executed when the node boots. The tarball is installed before the startup command is run, which means that you can set a node to automatically install and then run custom software.

Hovering the mouse cursor near any node on the canvas will start creation of a link; click and drag the resulting line to another node to complete it. Flack will select a link type based on the nodes you are trying to connect. If the nodes are from the same aggregate, a VLAN will be created. These VLANs are conservatively allocated and operate at the Ethernet layer which provides more flexibility for experimentation. If the nodes are from different aggregates, Flack needs to stitch them together. The most basic kind of stitching is a GRE tunnel which uses IP-in-IP encapsulation to create a virtualized link across the commodity Internet.

Like nodes, links have an "i" button that displays the details of the link and allows its properties to be edited. Link types can be modified, such as changing a VLAN to a tunnel. It also allows editing of IP addresses and the interfaces to which the links connect. By default, Flack tries to use a different subnet for each link.

Drawing a topology on the canvas does not, in itself, cause resources to be allocated. To instantiate the topology of selected components, use the "Submit" button in the slice window. Once allocation begins, the background will change color to indicate that resources have been allocated but are not yet ready. While Flack is working, the status bar at the top indicates which operations are in progress. The dashboard to the left of the canvas shows the status of allocations at each aggregate. For the curious, clicking on the "tasks" button will show the details of the GENI calls made by Flack. As resources become ready, nodes' outlines will change color to indicate their status. Once the slice is allocated and all resources are ready, the background and borders will turn green.

2.4 Accessing and Using Your Slice

After the slice is created and ready, you can access your resources over the commodity Internet. The DNS name of a resource (needed to `ssh` to the resource) can be found by clicking on the "i" information button on any of the nodes. The information window now displays more information than was available before slice creation finished. For example, if you did not request specific nodes when creating the slice, the DNS names of the specific resources that were allocated will be shown.

To log into the nodes in the slice, one typically uses `ssh`. Flack sets up the nodes using the public key(s) that you supplied when setting up your account at Emulab. Assuming the associated private key has been loaded into your `ssh` client, you can login using the user name and host found near the top of the node information window. Once you are logged in, you are free to perform any operation you want: because slices are isolated from one another, and nodes and links are returned to a "clean" state in between uses, your actions will have limited impact on others using GENI.

3. MONITORING GENI

Because the GENI network is built from conventional networking infrastructure (e.g., physical routers and links), it can be monitored using conventional network monitoring tools similar to those used by network operation centers today (e.g., tools to gather SNMP data). We refer to this type of monitoring as *Global Monitoring* because it monitors the performance of the GENI testbed as a whole. Global monitoring is of great interest to the GENI testbed operators such as aggregate operators and the GENI Meta Operations Center, GMOC (`http://gmoc.grnoc.iu.edu/`).

However, unlike conventional networks, GENI enables experimenters to create and operate *their own networks (slices)*. In that sense, experimenters are network operators and need to monitor and observe what is going on inside their network. In fact, experimenters are primarily interested in the behavior and performance of their experiment with little—or at least less—desire to know about the performance of the GENI testbed as a whole. Unfortunately, relatively little attention has been given to the problem of helping experimenters (users) set up and operate the instrumentation and measurement infrastructure needed to monitor their experiments.

To address this problem, the INSTOOLS project has been developing *slice-specific monitoring* infrastructure that is designed to capture, record, and display information within a slice. Although users may opt to make available some or all of their information to globally-shared monitoring services and repositories (e.g., the GMOC), the primary focus is on developing a slice-specific monitoring framework.

Providing each experiment with its own instrumentation and measurement infrastructure has several benefits. First, it can be tailored to the topology of each experiment, rather than requiring all experiments to get their monitoring data from a single shared instrumentation and measurement infrastructure. Second, it means that an experiment's measurement data can be kept private to the experiment—never being stored or processed by resources outside of the experiment. Third, the monitoring load imposed by experiments is distributed to the nodes, rather than placing the monitoring burden on a single shared infrastructure node. Moreover, monitoring traffic can be kept local to the experiment.

3.1 Instrumentation Infrastructure

Setting up the infrastructure to monitor a conventional network can be as time consuming as setting up the network itself. In GENI, where users can set up and tear down networks on the timescale of minutes, automated mechanisms are needed to assist in setting up the monitoring infrastructure. The INSTOOLS software is specifically designed to make it easy for users to "instrument" their experiments with monitoring software. To make it easy for users to instrumentize their slice, INSTOOLS will automatically deploy, configure, and run monitoring software or services on the resources that comprise the slice (called *Monitoring Points (MPs)*), and it will also set up servers (called *Measurement Controllers (MCs)*) to collect the information captured at MPs, process the information, and make it available to the experimenter for presentation via graphical user interfaces (see Figure 3). By default, the INSTOOLS software will monitor a wide range of commonly used network performance metrics, but it can be configured to collect more or less information based on the experimenter's needs.

The MCs for an experiment are automatically added to the user's slice and initialized by INSTOOLS—in fact they will automatically show up on the Flack canvas when the experiment is "instrumentized". Because experiments can span multiple aggregates that are each under different authoritative control and may offer different types of resources, the INSTOOLS architecture creates (at least) one MC for each aggregate in the experiment (slice). Creating one MC for each aggregate ensures that the measurement plane traffic stays within an aggregate, improving scalability. It also enables MCs to offer aggregate-specific monitoring features that are tailored to the type of resources offered by the aggregate. In general, INSTOOLS will create the MC by adding a PC to the slice and then loading it with a custom operating system designed specifically for that aggregate that has been pre-configured with the INSTOOLS software. In addition to adding an MC, INSTOOLS sets up the measurement channels/links between the MPs and the MC so that the measurement plane (to the greatest extent possible) does not interfere with the slice's data plane.

Because an experiment may have multiple MCs, each collecting measurement data on behalf of the user, INSTOOLS provides a interface called a *Portal* that allows a user to easily view and control all the data being collected by all of the MCs. The portal provides a "one-stop shop" for users to interact with the instrumentation infrastructure that has been deployed in their slice. Moreover, the portal can redirect users to archival services where archived copies of their measurement data are stored and possibly shared with others (see Section 3.2). The INSTOOLS architecture is illustrated in Figure 3.

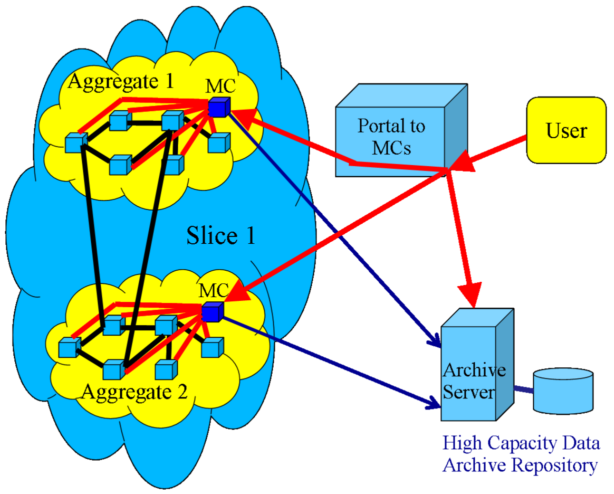

Figure 3: The INSTOOLS Instrumentation Infrastructure

To enable automatic data capture at MPs, INSTOOLS enhances the software installed on a node by adding a variety of off-the-shelf and custom monitoring software/services, and then modifies the experiment creation code in the control framework to configure and launch the added monitoring software on each node. The monitoring software that we add to each node includes an SNMP [1] daemon, various off-the-shelf system/network management programs (e.g., `tcpdump`, NetFlow, etc.), custom monitoring code based on the `pcap` library to collect certain packet statistics not captured by the SNMP daemon, and a remote access daemon to execute the capture software. In addition to capturing network information, INSTOOLS also captures operating system information such as CPU load, memory load, routing table configurations, ARP caches, loaded modules, etc. These data are captured via standard operating system tools like `ps`, `vmstat`, etc., and also via SNMP. Table 1 summarizes the set of tables and graphs that INSTOOLS captures by default.

3.2 Archiving Measurement Data

To make it easy for users to save the measurement data they have collected for future analysis, INSTOOLS provides three ways to

Tables about a node	ARP table IP address table Loaded kernel modules table Process table Routing tables Active TCP connections Active UDP listeners
Graphs about a node	Node CPU utilization graph Node memory utilization graph Node total traffic graph Node IMCP traffic graph Node total TCP traffic graph Node total UDP traffic graph Node total IP traffic graph
Traffic graphs about a link	byte count graph for link interface traffic packet count graph for link unicast traffic
Port/Protocol-specific graphs	Netflow graphs for a long list of ports/protocols including: "169," "255," ah, eigrp, esp, icmp, ipinip, normal, skip, tcp, udp, aim, bo2k, dc, dns, edonkey, ftp, gnutella, http, https, imap, irc, kazaa, nntp, pop3, real, smtp, ssh, and telnet

Table 1: Information automatically captured by INSTOOLS

archive data: (1) the Univ. of Kentucky Archive Service (UKAS), (2) the CNRI Archive Service, and (3) the GMOC Archive Service.

The UKAS is the most powerful of the services because it not only provides a repository for data, it also provides a computational environment for viewing archives that recreates the same look-and-feel the user had when viewing the live data. In particular, it runs the same content management system used for the live view, allowing it to display the same web pages that were offered by the MC at the time the archive was taken.

The CNRI archive service supports the concept of "workspaces" which can then be archived to long-term storage. INSTOOLS supports the ability to store measurement data in the slice's "workspace" and then adds the metadata needed to move the workspace data into permanent storage. The archive files can be accessed via the CNRI web interface at `http://mda.doregistry.org/`.

In the case of the GMOC archive service, the collected measurement data are first transformed into a GMOC-specific format and then archived at the GMOC site. Similar to the CNRI service, INSTOOLS generates the necessary metadata for the GMOC archive service and the archived data can be retrieved from the GMOC site at `http://gmoc-dev.grnoc.iu.edu/api-demo/data`.

4. AN EXAMPLE EXPERIMENT

Having described the basic functionality of Flack and INSTOOLS, we are now ready to walk through an example experiment that uses Flack and INSTOOLS to create and monitor a GENI slice.

Consider the topology shown on the Flack canvas in Figure 4, which consists of two (black) nodes selected from the Utah aggregate manager (labeled "*source*" and "*r2*") and one (light green) node selected from the Kentucky (UK) aggregate manager (labeled "*r1*"). The three nodes form a clique, with the two Utah nodes connected by a VLAN and the UK node connected to the Utah nodes via GRE tunnels over the commodity Internet.

For the purposes of our example, assume the goal of the experiment is to test routing changes. The *source* node will generate UDP

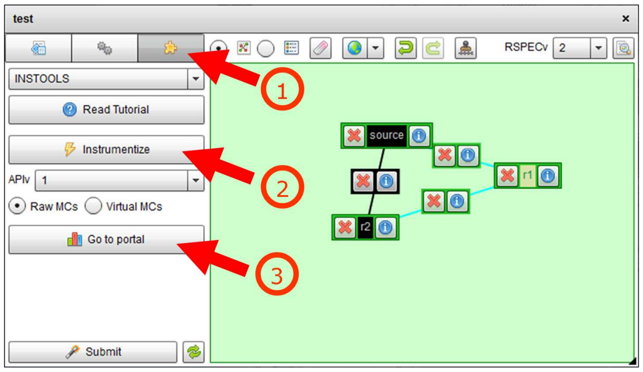

Figure 4: Flack canvas showing topology of the example experiment and sequence of steps for instrumentizing the slice.

traffic to both *r1* and *r2*. Periodically, we will modify the routing table on *source* so that it routes traffic solely through one of its outgoing interfaces or routes traffic out both interfaces at the same time. We will use a tarball that contains the traffic generator and the scripts to manipulate the routes, and will start them automatically using a startup script. Using INSTOOLS, we will be able to observe the traffic and the periodic routing changes.

Begin by creating a new slice in Flack as described in Section 2.3. On the slice window, rather than drawing the topology, use the drop-down "Import from..." menu near the top of the window, and select "Import from the web." You will be prompted for a URL; enter `http://protogeni.net/tutorial.xml`. This is a "*Resource Specification* (RSPEC)" prepared specifically for this tutorial. It contains the topology, tarball URL, and startup commands described above. After it is loaded, we encourage you to explore it by using the information buttons on the nodes and links.

When you are ready, click the "Submit" button to have Flack reserve the resources shown on the canvas. The canvas will change yellow to indicate that the reservation is in progress, and after a few minutes, will turn green when they are ready for use.

4.1 Monitoring the Slice

Once the reservation is finished, the slice can be instrumentized as shown in Figure 4 by selecting the plugins tab (step 1) and clicking on the "Instrumentize" button (step 2). The instrumentation infrastructure will be created, with measurement software distributed and configured on experimental nodes with one MC added to each aggregate. When the instrumentation process has finished, clicking on the "Go to portal" button (step 3 in Figure 4) opens a browser window to the INSTOOLS portal for the experiment.

The portal, shown in Figure 5, allows users to access all of their measurement data via a single interface. The main window of the portal provides a map view overlaid with the resources and network topology used in the slice. If resources on the map are visually too close to one another to be distinguished from one another, the portal provides the ability to "Expand" the view to clearly show the resources—presenting more of a logical view of the topology. You will note that in addition to the slice's resources, two measurement controllers (one for each aggregate), "MCutahemulab" and "MCukgeni" have been added for the purposes of collecting the measurement data.

Click on a node or a link—a dialog box will appear, giving you a menu of measurements that the portal can display. For example, in Figure 6, clicking on node *r2* brings up a control form that shows

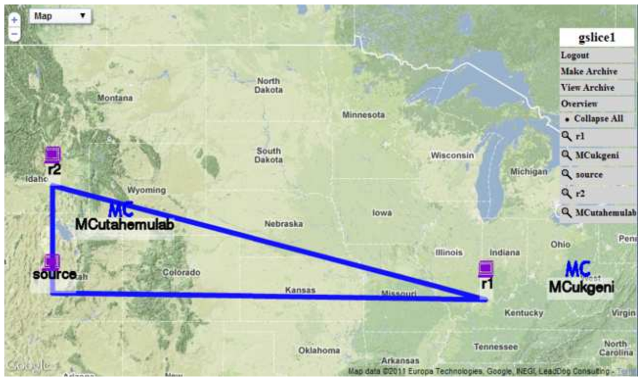

Figure 5: The Portal Interface

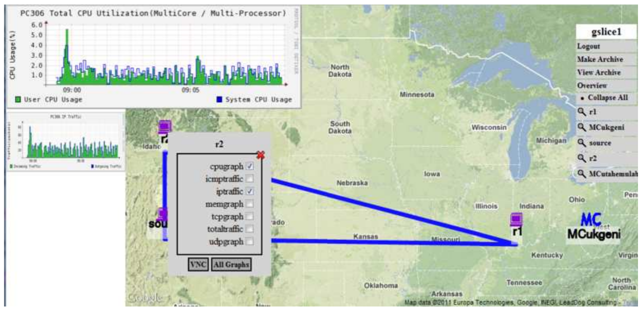

Figure 6: Measurement data visualized at the portal

checkboxes for displaying commonly used measurement graphs, a link to the machine's VNC connection for remote access, and a link to the Live View page that contains all of the node's graphs. Selecting a particular graph from the list will cause that graph to appear on the left hand side of the portal in a readable, but small, format. Hovering the mouse over the graph will enlarge it for a better view. Traffic graphs for a link can be viewed by clicking on a link and selecting the desired graph. Clicking on an MC node brings up a popup that allows one to go to the "Main Page" of the MC which provides a live (realtime) view of all the data being collected by that MC. For example, Figure 7 shows a live MC page that depicts the traffic loads for several different types of packets passing through a node over time.

Open traffic graphs for the experiment's three links by clicking on each link, then selecting one of the checkboxes. You will see UDP traffic generated by the simple traffic generator that has been installed on the nodes. As the startup script varies the routes, the amount of traffic carried on each link will periodically change.

4.2 Extending and Deleting the Slice

By default, Flack only requests resources for a period of 2 hours—after this time, they will be released from your slice. Of course, many experiments require more time. The "slice" tab in the Slice window (which opens automatically after you use the "Submit" button to allocate resources) includes buttons to extend the slice's lifetime. When you are finished with your slice, the "slice" tab also includes a "delete" button. If you close Flack or the Slice window, the main Flack window has a list of slices on the left side. Clicking on any slice will re-open the slice window and populate it with the currently allocated resources.

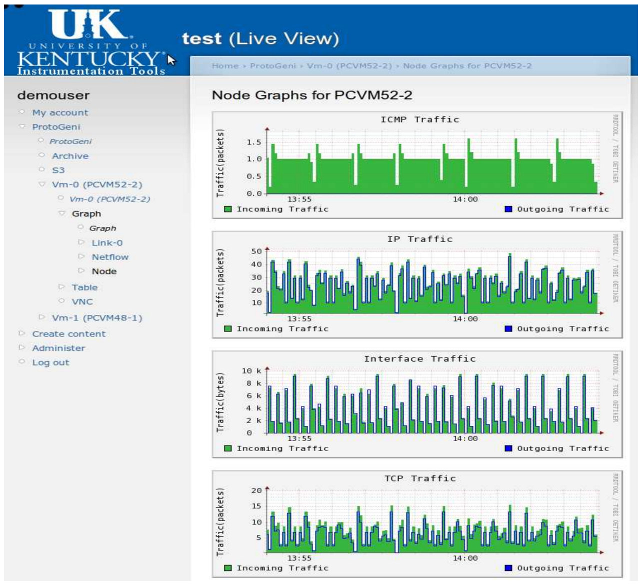

Figure 7: Live view of the traffic on a node

5. GOING FURTHER

The example experiment covered in this tutorial is only the beginning of what you can do with GENI. To explore GENI further, we recommend the following resources:

- The `protogeni-users` mailing list, for support and general discussion of ProtoGENI: `http://groups.google.com/group/protogeni-users`
- The ProtoGENI website, for more documentation and examples, including source code and command line tools: `http://www.protogeni.net/`
- The GENI experimenter portal, for links to other GENI resources and tools: `http://groups.geni.net/geni/wiki/ExperimenterPortal`

6. ACKNOWLEDGEMENTS

Our thanks go to the many individuals at universities, companies, the NSF, and the GENI Project Office who have worked to make GENI a reality; they are too numerous to name here.

This material is based upon work supported in part by the National Science Foundation under grants CNS-0834243 and CNS-0723248. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of BBN Technologies Corp, the GENI Project Office, or the National Science Foundation.

7. REFERENCES

[1] J. Case, R. Mundy, D. Partain, and B. Stewart. Introduction and Applicability Statements for Internet Standard Management Framework. RFC 3410, Internet Engineering Task Force, December 2002.

[2] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38:69–74, March 2008.