# InstaGENI Design Document

Rick McGeer, Jessica Blaine, Nicki Watts, N. Krishnan
HP Labs

Andy Bavier
Princeton University

Rob Ricci
University of Utah

Joe Mambretti
Northwestern University, StarLight

# Overview

## Design Philosophy

The InstaGENI design philosophy is oriented to providing the highest possible level of capability, flexibility, and option variability for experimenters while maintaining exceptionally high levels of operational utility, reliability, redundancy, and cost effectiveness for resource site managers. To achieve these goals, it is important to address three design components of this distributed environment, a) core foundation infrastructure resources b) management and control resource environments and c) experimental environments. The core foundation resources consist of the components that essentially constitute a large scale, highly distributed, integrated instrument. Such components include racks, power feeds, compute, storage and switching devices, optical and copper cables, PDUs, and basic campus, regional, national and international network resources. This distributed instrument will constitute a type of integrated container that will host two sets of environments. One set consists of management and control resource environments that provide the frameworks required to support all capabilities necessary to create a very wide range of experimental environments, including environments that instantiate their own management and control planes. The overall management and control frameworks will allow the highest possible latitude for designing, implementing, and operating specialized experimental environments.

## Minimization of Risk and a Tiered Design

The GENI Project Office and the InstaGENI design team recognize that there is a significant tension between the need to deploy the InstaGENI racks in the mesoscale GENI deployment, an operational infrastructure, and the need to provide new experimental services for GENI researchers. For this reason, the InstaGENI team and the GPO have chosen a tiered design, where foundational services ship with the rack on the control node and higher-level services are offered in swappable images on the experiment nodes.

The native Aggregate Managers on the InstaGENI racks are ProtoGENI (for image distribution and computation) and FOAM (to implement a hypervised OpenFlow network). However, ProtoGENI cannot match the high scalability of PlanetLab due to PlanetLab's very lightweight virtualization technology. To overcome this, the PlanetLab base node image will be offered as a standard image on the InstaGENI racks. When swapped in, PlanetLab nodes on the InstaGENI racks will federate with a centralized PlanetLab Aggregate Manager (IG-PLC) and will offer long-running slices. In the extreme, all experiment nodes on the rack will federate as PlanetLab nodes, in which case the control node will simply function as a local FOAM controller unless and until the ProtoGENI controller reclaims the Planetlab nodes

We note that this tiered design strategy has been used successfully in ProtoGENI for several years, has a long antecedent in PlanetLab itself, a precursor to GENI. Many services which are ordinarily thought of as part of the kernel of a PlanetLab-like

system are realized as PlanetLab slices. In a similar manner, the tiered design strategy of InstaGENI offers a proven, robust core for an infrastructure deployment while offering advanced services in the images.

## Architectural Principles

Currently a macro trend in architectural design is one that emphasizes the highest levels of abstraction possible, as can be seen in emerging techniques for Platforms-As-A-Service (PaaS), Software-As-A-Service (SaaS), Infrastructure-As-A-Service (IaaS), any Resource-As-A-Service (XaaS), etc. InstaGENI has been designed to allow the leveraging of this trend so that the InstaGENI environment can be continually expanded and improved for the foreseeable future.

The GENI Aggregate Managers operate at multiple-levels of the anything-as-a-service hierarchy. FOAM offers Networks-as-a-Service. ProtoGENI offers Hardware-as-a-Service (dedicated physical machines or, at a minimum, dedicated cores) and virtual machines as a service; and PlanetLab offers Containers-as-a-Service. Higher-end possibilities can emerge on this common base. For example, Seattle offers Platform-as-a-Service on top of the PlanetLab and ProtoGENI bases, or standalone.

Since Hardware-as-a-Service and Network-as-a-Service are foundational elements, the InstaGENI rack offers ProtoGENI and FOAM as the native Aggregate Managers. The Control Node functions as a FOAM and ProtoGENI controller, offering ProtoGENI slices of physical and virtual machines, and FOAM virtual networks as primitive elements of the InstaGENI stack. Higher-level Aggregate Managers, including ORCA and PlanetLab, as well as Storage-as-a-Service, Database-as-a-Service, and Platform-as-a-Service can be layered on top of the basic functions.

## Integration with GENI Aggregate Managers

As mentioned above, the InstaGENI rack views GENI Aggregate Managers as implementing different levels in the as-a-Service hierarchy. The base layer is ProtoGENI and FOAM. The InstaGENI rack functions as a small ProtoGENI cluster with OpenFlow networking based on FOAM, and experimenters can allocate ProtoGENI slices just as they do at other ProtoGENI sites. However, the InstaGENI ProtoGENI installation is designed to federate; experimenters will be able to allocate federatable slices across multiple InstaGENI sites, and across multiple non-InstaGENI GENI sites.

InstaGENI will federate with PlanetLab in the manner described above. The basic PlanetLab node image will ship with InstaGENI. When a PlanetLab node is allocated, it federates with a private MyPLC instance called IG-PLC. IG-PLC will export a single instance of the GENI AM API and federate with PlanetLab Central and ProtoGENI.

## Network Connectivity and Stitching

As noted, InstaGENI is comprised of three major components, and consequently, there are three classes of major considerations for network connectivity, comprised

of sets of options for each of the major components. The InstaGENI design is sufficiently flexible to accommodate all major potential options. These are described in the section of this document on “Wide Area Connectivity.” That section also describes the networks that will be used for management, control and data planes and for WAN resources used as part of experimental environments. The topic of "Network Stitching" is covered in this document in the section with that title.

### Site Provision of IP Addresses

Both PlanetLab and ProtoGENI offer multiple virtual containers per node, each containing a GENI sliver. In PlanetLab’s and ProtoGENI’s networking model, a sliver can be associated with one or more virtual interfaces that connect it to the network in the following ways:

- *Private IP address with NAT translation*. The virtual interface inside a sliver has a private IP address and a NAT is used to translate between the private address and the node’s public IP addresses.
- *Public IP address with direct connectivity.* The virtual interface inside a sliver is addressable by a unique IP address. Each virtual interface can also be addressed at L2 by a unique MAC address.
- *Public or private IP address with tagged VLAN.* The virtual interface inside a sliver can be associated with a VLAN.

There is nothing that new or surprising about these methods of connecting VMs to the network, as they are supported by most modern VM environments (e.g., VirtualBox, VMware).

Access to the virtual containers (for example, ssh and other services) will be available from the public Internet if and only if, a large enough routable subnet is provided by the hosting site. ProtoGENI will allocate the available IP space on a first come first basis to virtual containers. Once the space has been exhausted (or if no space is provided), containers will have non-routable IP addresses. When a container has a non-routable IP, users will be able to access their containers by first logging into the local ProtoGENI file server node, where users will have accounts. Once logged into the file server, they can ssh into their containers. If allowed by the site hosting the rack, InstaGENI will also provide Network Address Translation (NAT) so that outgoing traffic can reach the public Internet.

Both the control node and the experiment nodes will have separately-routed iLO cards for node control. It is strongly recommended that the control node iLO card have a publicly-routable IP address. Thus, the minimum number of routable IP addresses for the rack is expected to be three: one for the control node, one for the control node iLO card, and one for FOAM.

### No Warranty

All software and hardware that ships with the InstaGENI racks come with no warranty of any sort. In particular, hardware in the InstaGENI rack is provided to GENI at an estimated 62% discount from list price; one of the sacrifices for this steep discount is the warranty HP generally provides on hardware. The InstaGENI team, in cooperation with the GENI Project Office, will attempt to replace defective hardware as budget and circumstance permit.

## Hardware Overview

The base design of the InstaGENI rack consists of five experiment nodes, one control node, an OpenFlow switch for internal routing and data plane connectivity to the Mesoscale infrastructure and thence to the Internet, and a small control plane switch/router for control plane communication with the Mesoscale infrastructure and the Internet.

The InstaGENI rack has been designed for expandability, while providing standalone functionality capable of running most ProtoGENI experiment or an exceptionally capable PlanetLab site. As with all designs, the result is a compromise. The summarized list of parts is shown here:

| Part |
|---|
| **Boss Node: ProLiant DL360G7, quad-core, single-socket, 12 GB Ram, 4 TB Disk (RAID) dual NIC** |
| **Experiment Node: ProLiant DL360G7, six-core, dual-socket,48GB Ram, 1TB Disk, dual NIC** |
| **HP ProCurve 6600, 48 1 Gb/s ports, 4 10 Gb/s ports** |
| **HP ProCurve 2610, 24 10/100 Mb/s ports, 2 1 Gb/s ports** |
| **42U Rack with power supply** |
| **HP PDU** |

The base computation node is the HP ProLiant DL360 G7, used for both experiment and control nodes. The control node features the Intel Xeon E5672, a quad-core, 3.2GHz processor. The experiment node features the Intel Xeon X5650, a six-core processor, 3.20 GHz. The DL360 G7 offers dual sockets. We will only use one socket in the control node, and both sockets in the experimental nodes. We therefore project 60 experimental cores/rack (depending on whether a quad- or six-core chip is chosen) and four cores in the control node. The DL360 G7 features the Intel 5520 chipset and 12 MB of L3 cache.

Experiment nodes are configured for images and transient storage: hence disk (1 TB/node) is relatively light. Permanent user and image storage is on the Control Node, with features 4 TB/disk in a RAID array. The memory controller is the HP Smart Array P410i/256 MB Controller (RAID 0/1/1+0/5/5+0). All disks are 7200 RPM Hot Plug SATA 2.5". Nodes in InstaGENI racks have local disk rather than a SAN: this enables isolation, when required, by allocating an entire physical node to a single slice, avoiding contention for disk or controller resources.

The experiment nodes and switch have been designed for promiscuous, rather than high-performance networking. The DL360 G7 features two HP NC382i Dual Port Multifunction Gigabit Server Adapters (four ports total) with TCP/IP Offload Engine, including support for Accelerated iSCSI.

The control node has been spec'd to 12 GB of memory, the preferred amount for the FOAM Aggregate Manager. This is packaged in three 4 GB DDR3 1333 MHz DIMMs. The experiment node has been spec'd to 48 GB of memory, or 4 GB/core.. All configurations are packaged as 6 8 GB DDR3 1333 MHz DIMMS..

The nodes may be extended by the use of two PCI express cards.
It is anticipated that the nodes will require remote monitoring and management. Hence all nodes, experiment and control, will ship with HP integrated Lights Out Advanced remote management, version 3 (iLO3). HP iLO Advanced delivers always-available remote server management that's always available. HP iLO Advanced also enables simplified server setup, power and thermal optimization, plus embedded health monitoring, with both remote Windows and Linux support.

### Networking

The primary network device shipped with the InstaGENI rack is the HP ProCurve (now E-Series) 6600 switch. The 6600 is a 1U switch optimized for data center optimization, and supports OpenFlow. The 6600 series includes 1U 10/100/1000Base-T and 10-GbE SFP+ stackables enhanced for server edge connectivity with front-to-back cooling, redundant hot-swappable power, and redundant hot-swappable fans. The 6600 supports the OpenFlow protocol using the OpenFlow HP 6600 1.09n firmware, and has been deployed on the Indiana University OpenFlow trial network. Our configuration features 48 1 Gb/s ports and 4 10 Gb/s ports.

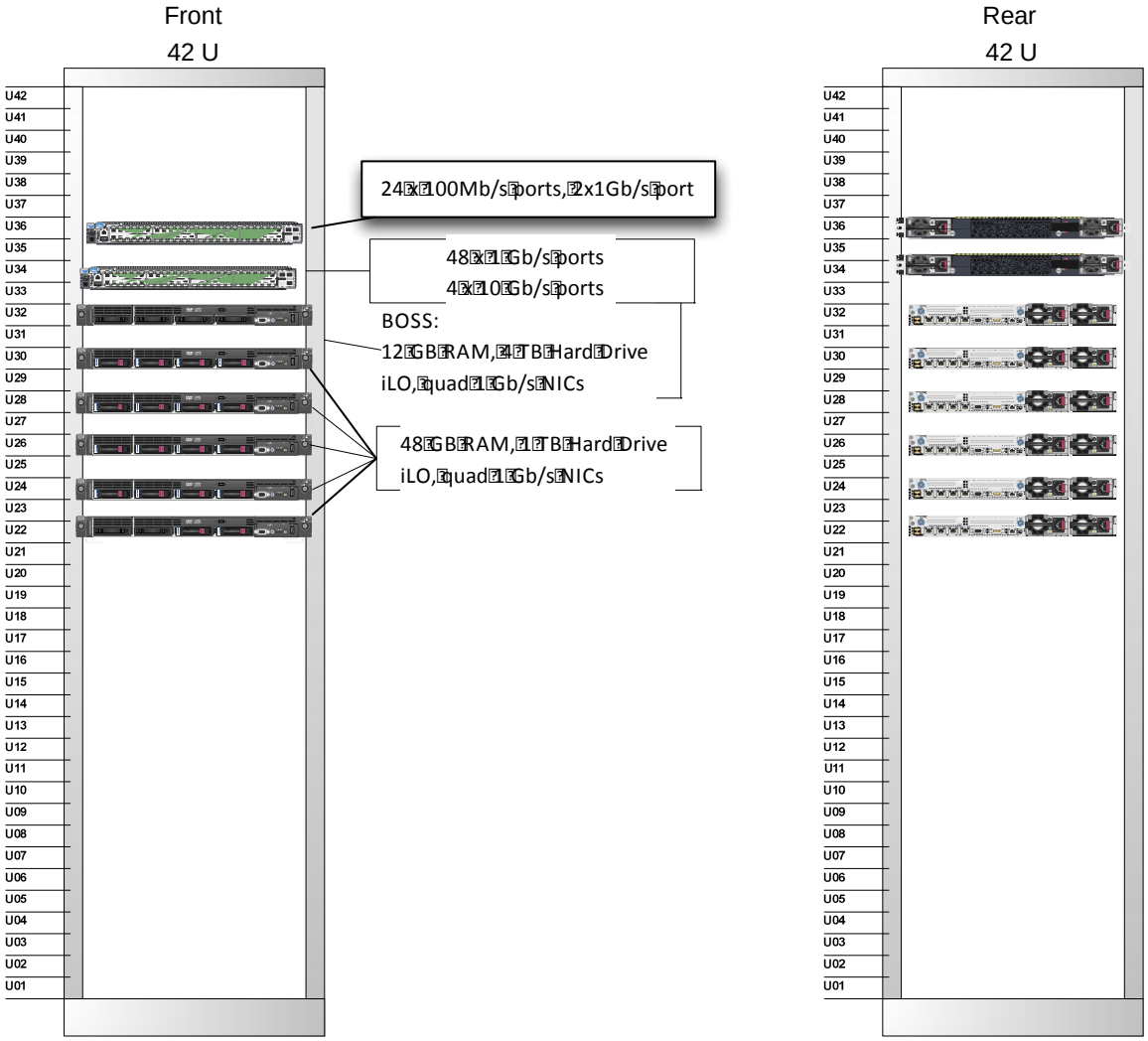
The control connection for the wide area goes through the HP 2610-24 switch. The ProCurve 2610-24 provides 24 10/100Base-TX connectivity, and includes two dual personality (RJ-45 10/100/1000 or mini-GBIC) slots for Gigabit uplink connectivity. An optional redundant external power supply also is available to provide redundancy in the event of a power supply failure. The 2610 switch will also carry the six (one control Node and five experiment nodes) iLO connections.

## Bill of Materials

| Heart Item # | Quantity | Model # | Description |
|---|---|---|---|
| 0100 | 1 | AF002A | HP Universal Rack 10642 G2 Shock Rack |
| | | AF002A | |
| | 1 | 001 | BASE RACKING |
| | 0 | | HP DL360G7 CTO Chassis [#4] |
| 0200 | 1 | 579237-B21 | HP DL360G7 CTO Chassis |
| 0201 | 1 | 588072-L21 | HP E5620 DL360G7 FIO Kit |
| 0202 | 3 | 593339-B21 | HP 4GB 1Rx4 PC3-10600R-9 Kit |
| 0203 | 1 | 516966-B21 | HP DL360 G6/G7 SFF HD Bkpln Kit |
| 0204 | 4 | 625609-B21 | HP 1TB 3G SATA 7.2k 2.5in MDL HDD |
| 0205 | 1 | 534562-B21 | HP 1G Flash Backed Cache |
| 0206 | 1 | 503296-B21 | HP 460W CS Gold Ht Plg Pwr Supply Kit |
| 0207 | 1 | 512485-B21 | HP iLO Adv 1-Svr incl 1yr TS&U SW |
| 0300 | 1 | H4396C | HP Contract Service to be ordered |
| | 0 | | HP DL360G7 CTO Chassis [#3] |
| 0400 | 5 | 579237-B21 | HP DL360G7 CTO Chassis |
| 0401 | 5 | 588066-L21 | HP X5650 DL360G6/G7 FIO Kit |
| 0402 | 5 | 588066-B21 | HP X5650 DL360G6/G7 Kit |
| 0403 | 30 | 500662-B21 | HP 8GB 2Rx4 PC3-10600R-9 Kit |
| 0404 | 5 | 516966-B21 | HP DL360 G6/G7 SFF HD Bkpln Kit |
| 0405 | 5 | 625609-B21 | HP 1TB 3G SATA 7.2k 2.5in MDL HDD |
| 0407 | 5 | 503296-B21 | HP 460W CS Gold Ht Plg Pwr Supply Kit |
| 0408 | 5 | 512485-B21 | HP iLO Adv 1-Svr incl 1yr TS&U SW |
| 0500 | 1 | H4396C | HP Contract Service to be ordered |
| 0600 | 1 | J9452AZ | HP 6600-48G-4XG Factory Integ Switch |
| 0601 | 1 | J9269AZ | HP 6600 Fact Integ Switch Power Supply |
| 0700 | 1 | J9085AZ | HP 2610-24 Factory Integrated Switch |
| 0800 | 1 | AB469A | HP Factory Rackmount Shelf Kit |
| 0900 | 1 | J9583AZ | HP X410 1U Integ Univ 4-post Rck Mnt Kit |
| 1000 | 1 | AF054A | HP 10642 G2 Sidepanel Kit |
| 1100 | 1 | 464794-B21 | HP ProCurve 5400 Series Rail Kit |
| 1200 | 1 | 252663-B24 | HP 16A High Voltage Modular PDU |
| 1201 | 1 | AF593A | HP 3.6m C19 Nema L6-20P NA/JP Pwr Crd |

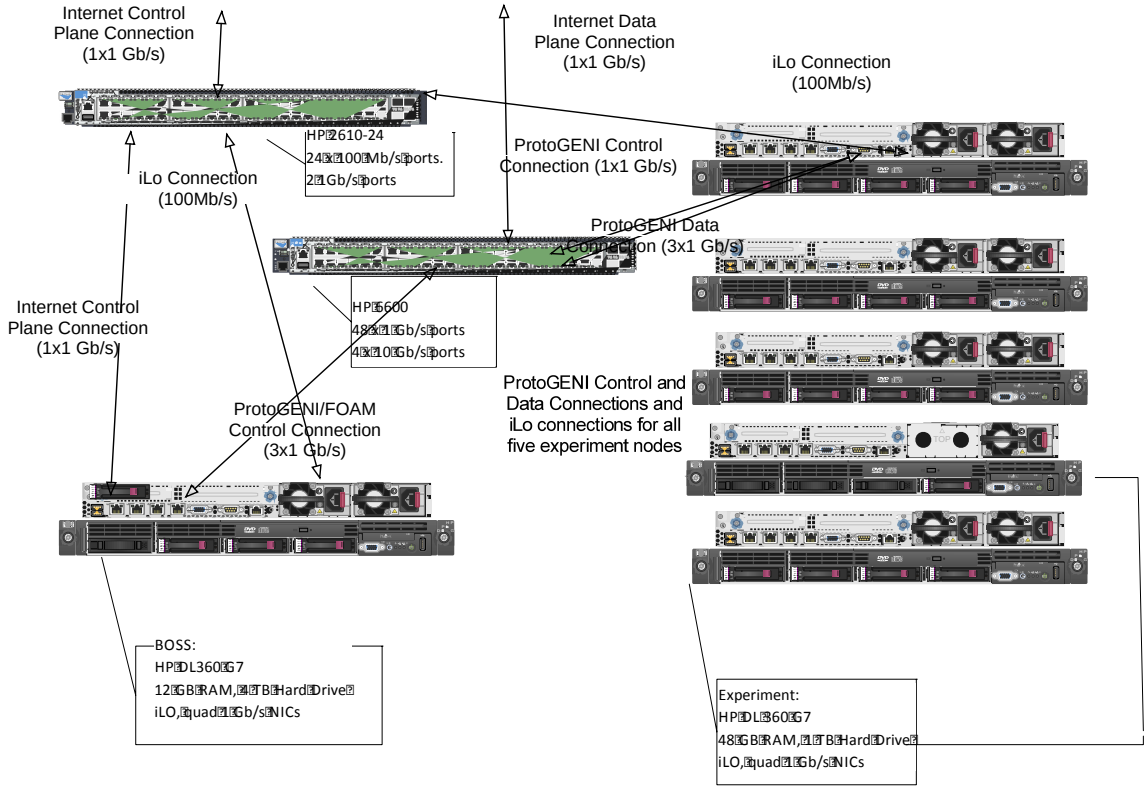
## Rack Diagram

Here, we show the rack diagram for the InstaGENI rack



## Topology Diagram

Here, we show the topology diagram for the InstaGENI rack, showing connections between the key components



## Software Stack Overview

Management and control functions for nodes in InstaGENI racks will be provided primarily by the ProtoGENI software stack. Each rack will have its own installation of the control software, and will be capable of operating as an independent unit. (The integration of the rack into the larger GENI infrastructure is described in later sections.)

The "Control Node" in each rack will run Xen. This allows multiple pieces of control software to run side-by-side in different virtual machines, with potentially different operating systems and administrative control. This configuration will also ease the deployment, backup, and update of the control software. Initially, there will be three such virtual machines:

- One will act as an Emulab/ProtoGENI "boss" node: this is a database, web, and GENI API server, and also manages boot services for the nodes
- A second will act as a local fileserver, and give users shells so that they can manage and manipulate the data on the fileserver even if they have no current sliver. This VM can also act as a gateway for remote logins to sites that do not have sufficient IP addresses to give every experiment node a publicly-routable address
- A third will run FOAM to control the OpenFlow resources on the in-rack switch

More control VMs may be added in the future as needed.

### Node Control and Imaging

The "experiment" nodes in the rack will be managed by the normal ProtoGENI/Emulab software stack, which provides boot services, account creation, experimental management, etc. Users will have full control over physical hosts, including loading new operating system images and making changes to the kernel, in particular the network stack. The ProtoGENI/Emulab software uses a combination of network booting, locked down BIOS, and power cycling to ensure that nodes can be returned to the control of the facility and to a clean state, meaning that accidental or intentional changes that render a node's operating system unbootable or cut off from the network can be corrected. Nodes will be "scrubbed" between uses: after a sliver is terminated, the node will be re-loaded with a "clean" image for the next user.

Images for OSes popular with network researchers, including at least two Linux distributions and FreeBSD, will be provided. Users may customize these images and

make their own snapshots. Installation "from scratch" of other operating systems will be possible, but will likely involve significant expertise on the part of the experimenter and manual intervention on the part of the rack administrators. Users making images in this fashion will be strongly encouraged to do so on the InstaGENI installation at the University of Utah, where the most assistance will be available. The operating system images provided by InstaGENI will be as "generic" (that is, capable of running on a variety of hardware that might be added to the racks) as possible. However, if sites hosting racks wish to add hardware that is not supported by these "generic" images, local administrators will bear responsibility for producing customized images.

One use of this capability will be to boot nodes into images that support other control frameworks: likely uses of this capability will be to create slivers that act as full PlanetLab nodes or OpenFlow controllers.

In addition to raw hardware nodes, ProtoGENI will also provide the ability to create multiple virtual machines (VMs) on the experimental nodes. ProtoGENI supports this in two forms; in the first, an experimenter can allocate a dedicated physical machine, and then slice that into any number of virtual containers. All of the containers are part of the one slice that is being run by the user. In the second form, one or more of the physical nodes are placed into "shared" node, which allows multiple users to allocate containers alongside other experimenters. Typically, nodes running in shared mode exhibit better utilization, but ProtoGENI will support either option. Physical nodes may be dynamically moved in and out of the "shared pool" at any time (though moving a node out of the shared pool destroys any slivers running on it.) InstaGENI racks will allocate at least one node per rack as a shared host; more nodes may be moved into this pool as required.

The slicing technology used for ProtoGENI virtual hosts is OpenVZ. Under this technology, multiple "containers" share the same Linux kernel, but have considerable freedom. In particular, each container gets a separate network stack, enabling it to access VLAN devices, set routing tables, etc. Users cannot, however, modify the kernel or the network stack; users who need such capabilities will be directed to use physical hosts. The major advantage of OpenVZ for slicing is that it is very lightweight. In contrast to VM technologies that do not share a single kernel, container virtualization has been shown to scale to hundreds of slivers per host (heavily dependent on the resource needs of individual slivers, of course.)

### Network Control

Within each rack, experimenters will have access to the "raw" network interfaces on nodes allocated to their slices. VLANs will be created on the rack's switch to instantiate links requested in users' RSpecs. A small number of VLAN numbers will be reserved for control purposes, leaving over 4,000 available to experiments. Using 802.1q tagging, each physical interface will have the ability, if requested, to act as many virtual interfaces, making use of many VLANs. With the exception of stitching

(described in later sections), user traffic within racks will be segregated by VLAN. The switch we have selected is capable of providing full line-rate service to all ports simultaneously, avoiding artifacts due to interference between experiments.

OpenFlow will be separately enabled or disabled for individual VLANs, a feature supported by the ProCurve switch we have selected for the racks. VLANs requested by users will default to having OpenFlow disabled. Request RSpecs will be able to request OpenFlow for particular VLANs; in this case, the OpenFlow controller for the VLAN will be pointed to the local FOAM instance as its controller, and users will contact FOAM to set up flowspace, controllers, etc. for their VLANs.

A single switch will be shared for experiment traffic and control traffic, so experimenters will be able to enable OpenFlow only on the VLANs that are part of their slices; OpenFlow will not be enabled on VLANs used for control traffic or connections to campus or wide-area networks.

Network ports that are not currently in use for slices or control purposes will be disabled in order to reduce the possibility for traffic to inadvertently enter or exit the network.

ProtoGENI virtual containers also permit the experimental network interfaces to be virtualized so that links and lans may be formed with other containers or physical nodes in the local rack. This is accomplished via the use of tagged VLANs and virtual network interfaces inside the containers. Note that ProtoGENI does not permit a particular physical interface to be oversubscribed; users must specify how much bandwidth they intend to use; once all of the bandwidth is allocated, that physical interface is no longer available or new containers. Bandwidth limits are enforced via the use of traffic shaping rules in the outer host environment. In addition to VLANs between nodes, ProtoGENI also supports GRE tunnels that can be used to form point to point links between nodes residing in different locations.

Connectivity between racks and with other components of GENI will be covered in more detail later in this document.

### Rack Installation Process

Software installation for the ProtoGENI control nodes will be accomplished through virtual machine images. The Xen instance on the control node will first have basic configuration (such as its IP address) set by local administrators. Generic control nodes images, to run inside the Xen VM, will be provided by the ProtoGENI team and will be customized local administrators. In particular, ProtoGENI has developed software that allows the local administrators to fill in a configuration file describing the local network environment (such as IP addresses, routers, DNS severs, etc.), and to convert a functioning ProtoGENI control server to use these addresses. This functionality can also be used to move an InstaGENI rack to another part of the hosting institution's network, if needed. A default Xen image running the FOAM controller will also be supplied.

Once a rack is wired up, software that will be developed for the ProtoGENI stack will test connections by enabling all switch ports, booting all nodes, and sending specially crafted packets on all interfaces. Learned MAC addresses will be harvested from the switches and compared against the specified wiring list. This will detect mis-wired ports and potentially failed interfaces, so that they can be corrected. The ongoing health of the network will be monitored by running Emulab's 'linktest' program after each slice is created; this program tests the actual configured topology against the experimenter's requests.

### Update Process

InstaGENI racks' control software will be updated frequently and in accordance with an announced schedule to keep up to date on GENI functionality and security patches; the "frequent update" strategy has proved effective on the Utah ProtoGENI site, which rarely suffers downtime due to software updates. All updates will be tested first on the InstaGENI rack at the University of Utah for a minimum of one week before being rolled out to other sites. All racks will receive at least one week of warning before software updates, and updates may be postponed in the face of upcoming paper deadlines, course projects, and other high-priority events. Most updates involve no disruption of running slices; updates that do carry this risk will be announced ahead of time to the GENI community and scheduled for specific (off-peak) times.

A snapshot of the "boss" VM will be taken before upgrades are undertaken, so that in the case of update problems, the control node can be returned to a working state quickly.

Institutions hosting InstaGENI racks will be strongly discouraged from modifying the control software so that updates can be applied uniformly across all racks. Administrators of modified racks will take responsibility for updating and maintaining those racks.

Backwards compatibility with the two previous versions of the GENI APIs will be preserved at all times to avoid the need for "flag days."

## Wide-Area Connectivity

The InstaGENI design required close consideration of three major classes of WAN connectivity. One class of WAN resources consists of those that constitute core foundation infrastructure, including those that support management planes, control planes and data planes beyond the support provided by the local rack network, which includes support provided by the local site network, the campus network, the regional network and national and international networks.

A second class of WAN connectivity consists of the actual management plane, control plane, and data plane channels, which will be supported by the core infrastructure.

A third class of connectivity consists of the networks that are created, managed and controlled by experimenters.

One set of resources that constitute part of the core foundation infrastructure includes those that support management planes, control planes and data planes provided by the local rack network, the site network, the campus network, the regional network and national and international networks. The InstaGENI design is based on an assumption that, in general, the WAN core foundation resources will be fairly similar and static. Also, the rack based interface for these capabilities will be fairly uniform. However, there are multiple current options for the design and implementation of individual campus network resources, including those that enable resource segmentation, a critically important attribute especially for research experimentation, which requires reproducibility. Consequently, an expectation is that basic connections to the InstaGENI racks will be customized for individual sites. Also, consideration will vary depending on local ownership and operations procedures. For example, some university research groups and CS departments manage their own networks, while others rely on division level or integrated campus-wide networks. In any case, the InstaGENI design is sufficiently flexible to accommodate all major potential options. One consideration is ensuring capabilities for resource segmentation, which is a critical attribute, especially for research experimentation that is intended to be reproduced. Also, segmentation provides a means from isolating the research environment from restrictive campus networking resource allocation schemes (e.g., rate limiting mechanisms, highly restrictive firewalls, overly conservative congestion controls, large flow filters, etc.) Segmentation also enhances security for the environment. Segmentation can be achieved in various ways. Perhaps the most optimal is dedicated optical fiber provisioned from an edge switch/router to the InstaGENI site, a technique that has been used effectively on a number of campuses. Another potential option is one or more dedicated or dynamically allocated VLANs from an edge switch/router to the InstaGENI site. Other options include using dedicated L2 Ethernet channels,

OpenFlow based paths, QinQ, MACinMAC, MPLS, various IP tunneling techniques, etc. Participants in the InstaGENI initiative will work with appropriate individuals at local sites to design required campus networking infrastructure and to provide guidance on interfacing with regional and national R&E networks.

The InstaGENI rack will provide support for a 10 Gbps port to support an uplink channel. Although not all campuses will initially be able to support 10 Gbps external WAN capacity, they should be able to do so in future years, and 1 G connections would be too restrictive.

### Networks Supporting Management, Control, and Data Planes

The InstaGENI management and control framework required to communicate with the rack equipment will be supported by a secure commodity Internet connection (e.g., using IPSec) utilizing the local campus network. This channel will primarily be used by facility management. In addition, if required, secondary, backup, private network can be implemented using the campus network, and its external connections to the national R&E networks.

The InstaGENI framework data plane will use the campus network, regional networks, and the national R&E networks to support transport of general resource utilization statistics as well as resource status monitoring. This data plane will also be used to support software updates.

<<WARNING later discussion determined that software updates are over the control plane>>>

The InstaGENI experimental environment will provide options for creating WAN networks based on dynamic or static VLANs provisioned over the core foundation channels noted above. In addition, options will be provided to provision dynamic or static L3 VPNs and tunnels. To enable rich connectivity, including dynamic connectivity among InstaGENI sites, the InstaGENI site at the StarLight communications exchange facility, which supports all national and international R&E and federal agency networks, will have options for implementing multiple cross connections to existing GENI facilities as well as to all major experimental national and international network testbeds, including those supported by Global Lambda Integrated Facility (GLIF) and GLORIAD, several regional networks, and national and international R&E networks and federal agency networks. StarLight will also support a direct InstaGENI-ExoGENI connection (ORCA is supported at StarLight). In addition, StarLight supports multiple L2 dynamic provision tools, including all those used by major R&E and federal agency networks, such as OSCARs, Sherpa, DYNES, Fenius, Argia, and others.

### OpenFlow

Experimenters will have access to OpenFlow capabilities that are inherent within the InstaGENI switch configuration. Currently, options for basing slices on VLANs or OpenFlow are mutually exclusive. However, this issue is being addressed by various development groups and when hybrid options are available they will be implemented within the InstaGENI environment. (See below: OpenFlow Integration).

External OpenFlow provisioning requires special consideration. To date, almost all OpenFlow testbeds have implemented a single controller to manage all of the OpenFlow switches within that environment. This approach will be used in the initial InstaGENI implementations.

However, because this approach requires all OpenFlow switches to connect to a single controller, scaling can become problematic because individual controllers can only support a finite number of OpenFlow switches, beyond which traffic parameters exhibit degradation. Also, this approach is vulnerable to problems if the controller encounters a fault. In addition, this approach does not address requirements for multi-domain topology discovery. Consequently, these situations are being addressed by emerging architecture that supports multi-controller environments across many OpenFlow domains, each with an individual controller, but implemented within a multi-domain federation. This architecture is currently being used to support a major international OpenFlow testbed at the StarLight facility. As the InstaGENI environment expands, it will implement this type of interdomain OpenFlow architecture.

Experimental connections to the public Internet will require special considerations, involving policies and procedures agreed to in advance by the GPO, campus representatives, the InstaGENI participants, and the experimenters

## GENI Integration

Because the primary control software for InstaGENI racks will be the ProtoGENI system, they will be fully compliant with all new versions of the GENI APIs. They will also support ProtoGENI's native APIs.

### ProtoGENI Integration

The InstaGENI racks will initially be registered as aggregate managers with the ProtoGENI clearinghouse. This means that they will be visible to, and usable from, existing tools that support the ProtoGENI APIs and clearinghouse; these tools include the ProtoGENI command line tools, Flack, the Kentucky Instrumentation Tools, and omni. Local administrators will be given several "policy knobs," which are currently implemented in the ProtoGENI codebase, in order to control access to the racks. These policy knobs allow the administrator to make the following simple policy decisions:

- Allow all GENI users access to the rack
- Allow GENI users to access the rack, but limit how many nodes each user may allocate at a time
- Block all external users (e.g. , those who do not have accounts registered on the particular rack) from using the rack
- Issue credential to specific users that allow them to bypass the policies above

We will add other policies of these types (e.g., user and resource restrictions) as required by sites. As GENI-wide policy becomes solidified and a central GENI clearinghouse brought up, the registration of the racks will be changed to the GENI clearinghouse. They will thus make use of the authorization and logging facilities that have been proposed for the Clearinghouse.

Initially, each rack will be given its own CA certificate; to establish trust with the rest of the GENI and ProtoGENI federations, a bundle of these certificates will be available from the ProtoGENI clearinghouse. ProtoGENI federates fetch this bundle nightly, so all current members of the ProtoGENI federation will, by default, accept the InstaGENI racks as members of the federation. Eventually, we anticipate that a "GENI Root" CA, signed by the GPO or NSF may be established; in that case, the InstaGENI racks could be issued certificates signed by this CA.

### PlanetLab Integration

The InstaGENI team will provide a PlanetLab node image that can be run on an InstaGENI rack: we call this the IG-PL node image. PlanetLab nodes use a container-based virtualization technology that provides an isolated Linux environment, rather than a standard VM, to a slice. Containers can offer better efficiency than VMs, particularly for I/O, because a hypervisor typically introduces an extra layer in the

software stack relative to a container-based OS. In the PlanetLab model, all slices run on an underlying shared kernel that slices cannot change. However it is possible to base the Linux environment offered to slices on different Linux distributions -- for example, a single node could support slices running several versions of Fedora, Ubuntu, and Debian Linux. For the IG-PL nodes, we plan at minimum to support both 32- and 64-bit slices running a recent Fedora build (e.g., Fedora 16).

The PlanetLab team is rapidly moving towards embracing Linux Containers (LXC) as the core virtualization technology. Since its inception PlanetLab has used Linux VServers for virtualization, but VServers have a number of limitations. From the standpoint of GENI, the most important is that VServers have a simple networking model based on L3 -- e.g., interfaces are mapped to VServers based on IP addresses, and it's not possible to customize the IP forwarding table or interface MAC address for each VServer. LXC is an alternative to Linux VServers currently being implemented by the Linux kernel community. LXC shares the same fundamental approach as Linux VServers, but unlike VServers it's incorporated into the mainline kernel (meaning no kernel patches), has a large developer base, and not least supports a richer networking model than VServers. Currently the PlanetLab team is debugging a prototype of a PlanetLab node based on LXC and expects that it will be ready to deploy in the next couple of months.

With the move to LXC, the PlanetLab team will extend PlanetLab's networking model to meet the GPO's stated requirements for GENI racks. The Linux kernel technology that makes this possible is network namespaces, which are integrated with LXC. Network namespaces provide each Linux container with its own view of the network. Within each container it is possible to customize many aspects of the network stack, including virtual device information such as IP and MAC address, IP forwarding rules, packet filtering rules, traffic shaping, TCP parameters, etc

Another important requirement for nodes running on a GENI rack is the ability to monitor resource consumption in close to real time. The IG-PL nodes will run slicestat, the low-level service that CoMon currently uses on PlanetLab to gather node statistics. Slicestat is a tiny sensor that takes a snapshot of current resource usage of active slices on a node. The GMOC will be able to access slicestat data on a well-known port on each node via HTTP. More information on slicestat can be found here:

http://codeen.cs.princeton.edu/slicestat/

IG-PL nodes will be managed using the PlanetLab control framework. The PlanetLab CF uses MyPLC to control nodes running the PlanetLab software stack. The SFA layer on top of MyPLC provides the GENI AM API interface to node resources. The PlanetLab team actively participates in efforts to define various GENI APIs and works to keep the SFA's implementation of the GENI AM API in sync

with community standards.

We propose a centralized control model whereby all IG-PL nodes (i.e., across all InstaGENI racks) are controlled and administered using a single instance of MyPLC and SFA. We'll refer to the central controller as IG-PLC. The local InstaGENI admin decides how many nodes will run the IG-PL image; then when an IG-PL node boots, it phones home to IG-PLC and appears in the list of resources exported by IG-PLC's Aggregate Manager. Users are able to allocate resources across all IG-PL nodes via the GENI AM API running at IG-PLC. This approach offers economies of scale with regard to administration similar to the public PlanetLab: the PlanetLab experts can directly troubleshoot problems across the entire platform and work with local admins to resolve them. We will initially host and staff IG-PLC at Princeton University; in the future it may make sense to migrate it to the GMOC or another support organization.

We will publish an IG-PL node image; under the model proposed above, that node image will be seeded with the information necessary to communicate with IG-PLC. As mentioned earlier, the transition of the IG-PL node image from VServers to LXC will happen after some InstaGENI racks have been deployed. We plan to perform initial installations with an image based on Linux VServers, in order to build out and test the IG-PL component of InstaGENI as thoroughly as possible. Once a LXC-based IG-PL image is ready we will publish the new image and encourage InstaGENI rack owners to upgrade to it.

### Network Stitching

Path stitching will be supported primarily through the implementation of static and dynamic VLANs across campus networks, regional networks, and national R&E networks. An expectation is that the large majority of paths will be based on VLANs. Extending VLANs across domains is a fairly straightforward process, as long as the domain participants agree on the number scheme used. However, at times a particular number or set of numbers is preallocated. Consequently, capabilities for VLAN translation (VLAN tag remapping) will be required. These VLAN translation capabilities will be provided at all InstaGENI sites and at the StarLight facility, where this function is currently already supported.

Internationally, stitching will utilize the emerging implementation on the GLIF of the Network Services Interface (NSI) protocol, which is being developed by the GLIF community and the Open Grid Forum, a standards organization. A software implementation of NSI has been implemented as a proof-of concept at multiple international exchange points and this implementation has been used for interdomain, multi-continent dynamic VLAN provisioning. A version of an NSI has been implemented as a persistent resource at the StarLight facility. Monitoring of this capability is based on perfSONAR.

### GENI Clearinghouse

When the GENI-wide clearinghouse becomes operational, the InstaGENI racks will move their registration from the ProtoGENI clearinghouse to the GENI clearinghouse. This will ensure that any allocation policies, auditing requirements, etc., required by the NSF and/or GPO are met. Racks will use the "AuthZ" service under discussion to enforce global policies and access conditions, and will use the planned logging service to ensure that all auditing information is made available to the GMOC in a secure, timely manner.

InstaGENI racks will initially be addressable as individual aggregates; that is, each will export the GENI AM API, and experimenter tools such as Flack will be able to contact the aggregates individually. Some proposed designs for the GENI clearinghouse place restrictions on contacting AMs in order to enforce GENI-wide policies. Proposed architectures include "forced proxying" through the Clearinghouse, organization of all GENI aggregates into an "Aggregate of Aggregates," or explicit sign-off of individual resource requests by the Clearinghouse. InstaGENI racks will support any of these configurations.

## OpenFlow Integration

Two types of OpenFlow integration will be supported: one for connections internal to a particular InstaGENI rack, and one for connections from the rack to external equipment (including other InstaGENI racks).

Inside of a rack, experimenters will be able to request OpenFlow on VLANs that their nodes are connected to. SNMP support in Emulab will be used to enable OpenFlow on a VLAN-by-VLAN basis, and all VLANs configured this way use FOAM as their controller. This will allow FOAM to enforce safety and isolation policies, such as limiting the switch CPU and memory resources available to each slice. In-progress extensions to the GENI RSpec will allow the ProtoGENI software and FOAM to exchange information about which VLANs and ports are currently allocated to which slice. FOAM will use this information to limit experimenters so that they can only manage flowspace for their own slices. FOAM can be configured using an RSpec extension with the GENI Aggregate Manager API.

## Remote Management Interface

Most administration of InstaGENI racks will be done through the Emulab/ProtoGENI web interface and via command line tools on the control node; physical access to the racks for administration is therefore not required.

All PCs in InstaGENI racks, including control nodes, will include HP's iLO technology, which includes power control and console access. This will allow both InstaGENI and local personnel to administer the nodes without requiring a physical presence. iLO console capabilities will be used for diagnosing faulty nodes (iLO continues to function in the presence of many type of hardware failures) and during the upgrade of control software.

Access to iLO on experiment nodes will be accomplished through the control nodes so that public IP addresses are not required. iLO on the control node itself will require a public address; this will enable remote administration and minimize downtime in the case of software failures (and many types of hardware failures) on the control node.

Full logs of resource allocations, including information about slices and users who requested them, will be available to the local administrators via a web interface. The raw data used in this interface will also be stored in a database on the control node, should local administrators wish to process this information in their own way. Using existing ProtoGENI APIs, the GMOC will be given credentials for each rack that allow

them to poll the rack for slice and sliver allocation status. When the GENI-wide clearinghouse is up and running, InstaGENI racks will be shift to using the logging service that the clearinghouse is expected to provide.

A web page will be provided that, given a source IP address on the public network and a time, will be provide information regarding what slice and user had control of that address at the time; this can be used by security offices and the GMOC to quickly identify the individual slice that may be the source of suspicious traffic to the Internet. The GMOC will also be given administrative accounts on the control node to assist in diagnosis, response, and post-mortem analysis of incidents.

InstaGENI personnel will maintain administrative accounts on each rack's control server in order to provide limited maintenance assistance to local administrators. InstaGENI personnel will also use this capability to assist with control software upgrades. While sites may, at their discretion, chose to deny administrative access to InstaGENI personnel, doing so will mean assuming all responsibility for the rack; InstaGENI will not provide support for racks configured in this manner.

## Emergency Stop

InstaGENI Racks will follow the Emergency Stop procedures outlined by the GMOC in their "Version 4" document, or newer versions as they become available.

Emergency stop of slices that are suspected of misbehavior will be provided through three interfaces:

- A web interface for rack administrators for cases in which they are made aware of misbehavior
- A GENI API call for use by the owner of a slice or the leader of a project, for cases when the slice may be compromised and used for purposes not intended by the experimenters
- A GENI API call for use by the GMOC, for cases when misbehavior is GENI-wide, is reported through GMOC channels, or occurs when local administrators are not reachable

The GMOC will be given a credential for each InstaGENI rack giving them full privileges to execute emergency shutdown on any slice. The GMOC will be the primary point of contact for any detected misbehavior that occurs after hours or on weekends or holidays.

Three levels of emergency stop will be provided:

- Cutting off the experiment from the control plane, but not the data plane: this is appropriate for cases in which as slice is having unwanted interactions with the outside Internet, but there is believed to be state within the slice worth preserving
- Powering off affected nodes and/or shutting down affected virtual machines
- Deletion of the slice and all associated slivers

When emergency stop is invoked on a slice, the "owner" of the slice is prevented from manipulating it further, and administrative action is required to complete the shutdown. This property can be used to preserve forensic evidence.

## Further References

Background material for this design, including site requirements, will be posted on the InstaGENI section of the GENI wiki.