ExoGENI: A Multi-Domain Infrastructure-as-a-Service Testbed*

Ilia Baldine, Yufeng Xin, Anirban Mandal, Paul Ruth, and Chris Heerman
RENCI

Jeff Chase
Department of Computer Science
Duke University

May 19, 2012

Abstract

NSF's GENI program seeks to enable experiments that run within virtual network topologies built-to-order from testbed infrastructure offered by multiple providers (domains). GENI is often viewed as a network testbed integration effort, but behind it is an ambitious vision for multi-domain infrastructure-as-a-service (IaaS). This paper presents ExoGENI, a new GENI testbed that links GENI to two advances in virtual infrastructure services outside of GENI: open cloud computing (OpenStack) and dynamic circuit fabrics. ExoGENI orchestrates a federation of independent cloud sites and circuit providers through their native IaaS interfaces, and links them to other GENI tools and resources.

The ExoGENI deployment consists of cloud site "racks" on host campuses within the US, linked with national research networks and other circuit networks through programmable exchange points. The ExoGENI sites and control software are enabled for software-defined networking using OpenFlow. ExoGENI offers a powerful unified hosting platform for deeply networked, multi-domain, multi-site cloud applications. We intend that ExoGENI will seed a larger, evolving platform linking other third-party cloud sites, transport networks, and other infrastructure services, and that it will enable real-world deployment of innovative distributed services and new visions of a Future Internet.

1 Introduction

ExoGENI is a new testbed at the intersection of networking and cloud computing, funded through NSF's Global Environment for Network Innovation (GENI) project. GENI is the major US program to develop and deploy integated network testbeds. The ExoGENI testbed is designed to support research and innovation in networking, operating systems, distributed systems, future Internet architectures, and deeply networked, data-intensive cloud computing. The testbed can also serve as a platform for novel applications and services, e.g., for the US IGNITE initiative. The initial deployment is scheduled to become operational in late 2012.

ExoGENI is based on an extended Infrastructure-as-a-Service (IaaS) cloud model with orchestrated provisioning across sites. Each ExoGENI site is a private IaaS cloud using a standard cloud stack to manage a pool of servers. The sites federate by delegating certain functions for identity management, authorization, and resource management to common coordinator services. This structure enables a network of private clouds to

*This paper is a reformatted version of a paper appearing in the *Proceedings of the 8th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom)*, Thessaloniki, Greece, June 2012. TridentCom proceedings are published by ICST, which holds the copyright for this work on their terms. This work is supported by the US National Science Foundation through the GENI initiative and NSF awards OCI-1032873, CNS-0910653, and CNS-0720829; by IBM and NetApp; and by the State of North Carolina through RENCI.

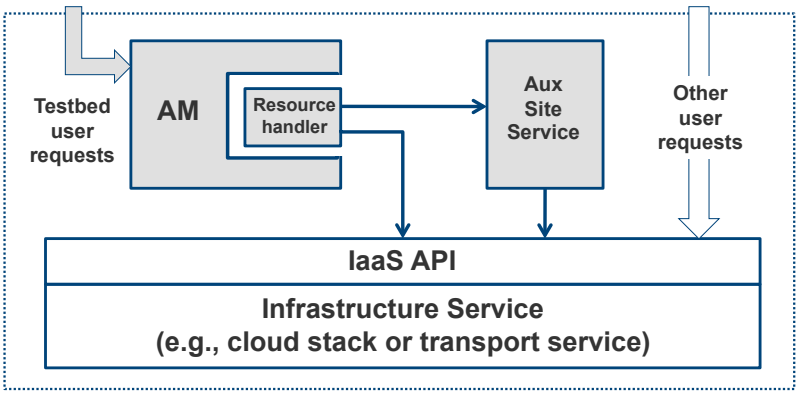

Figure 1: Structure of a resource provider or *aggregate*. Each provider runs a native infrastructure service (IaaS) of its choice, which may serve requests from local users through a native API. To join a federation the aggregate is fronted with a generic Aggregate Manager (AM) service. The AM validates user requests against local policy and serves them by invoking the native IaaS API through resource-specific plugin *handlers*. A handler may rely on other auxiliary services for some functions, e.g., image loading, OpenFlow authorization, or network proxying.

operate as a hybrid community cloud. Thus ExoGENI is an example of a *multi-domain* or *federated* cloud system, which some have called an intercloud.

ExoGENI combines this structure with a high degree of control over networking functions: OpenFlow networking within each site, multi-homed cloud servers that can act as virtual routers, site connectivity to national circuit backbone fabrics through host campus networks, and linkages to international circuits through programmable exchange points. The project aims to enhance US research cyberinfrastructure capabilities in four inter-related ways:

- **The missing link.** ExoGENI interconnects clouds to dynamic circuit fabrics, enabling a range of networked cloud applications and services, including data-intensive interaction, distributed data sharing, geo-replication, alternative packet networks, and location-aware services.
- **On-ramps to advanced network fabrics.** ExoGENI shows how to use campus clouds to bridge from campus networks to national transport network fabrics, overcoming a key limitation identified by NSF's CF21 vision. ExoGENI cloud sites can act as *virtual colocation centers* that offer on-demand cloud services adjacent to fabric access points. Sites at fabric intersection points can also act as *virtual network exchanges* to bridge "air gaps" between fabrics stemming from lack of direct connectivity or incompatible circuit interfaces.
- **Cloud peering and data mobility.** ExoGENI enhances the potential for peering and sharing of private clouds. It offers a means to bring data and computation together by migrating datasets to compute sites or placing computation close to data at rest.
- **Networking as a service.** ExoGENI brings flexible network configuration to cloud computing. It also enables experimental deployments of new packet networking models over a flexible link substrate. Built-to-order virtual networks can implement routing overlays using IP or other packet-layer protocols. Testbed users may deploy custom node operating systems with alternative networking stacks into their nodes, and use OpenFlow datapaths and/or virtual routers to implement new network services at the cloud edge and at network intersection points.

This paper gives an overview of the ExoGENI testbed and its control software. The research contribution of this paper is to summarize the design principles of ExoGENI resulting from our experience in developing

the testbed software (Section 2). Section 3 explains the rack site structure and interconnection architecture for the testbed. Section 4 outlines the integration of multi-domain IaaS with the GENI architecture as it currently stands. The software described in this paper has been implemented, deployed and demonstrated at various events.

2 A Testbed of Federated IaaS Providers

ExoGENI supports virtual infrastructure resources, which are instances of "fundamental computing resources, such as processing, storage, and networks" according to the NIST definition of Infrastructure-as-a-Service [17]. Testbed users may instantiate and program a virtual topology consisting of virtual machines (VMs), programmable switch datapaths, and virtual network links based on Ethernet standards. The deployment is based on an evolving set of technologies including point-to-point Ethernet circuits, OpenFlow-enabled hybrid Ethernet switches, and standard cloud computing software—OpenStack and xCAT [8].

The "Exo" (outside) prefix reflects our view of how GENI will evolve and what capabilities are needed to deliver on the promise of GENI to "explore networks of the future at scale". GENI is evolving alongside cloud technologies and open network control systems whose functions and goals overlap with GENI. The rate of investment in developing and deploying these systems is quite a bit more than an order of magnitude larger than the GENI effort.

One purpose of ExoGENI is to define a path to leverage these technologies and substrates in the GENI project. At the same time, GENI control software offers new ways to combine and extend them as a unified deployment platform for advances in network science and engineering. ExoGENI shows how GENI control software can leverage IaaS advances in a way that addresses important orchestration challenges for networked cloud computing.

The Exo prefix captures four related principles illustrated in Figure 1:

E1 **Decouple infrastructure control from orchestration.** Each provider domain (*aggregate*) runs a generic front-end service (an Aggregate Manager or AM) that exports the testbed APIs. The AM cooperates with other services in the federation, and invokes a back-end infrastructure service to manage the resources in the domain.

E2 **Use off-the-shelf software and IaaS services for infrastructure control.** Standard IaaS software and services offer a ready back-end solution to instantiate and release virtual resources in cloud sites, circuit services, and other virtual infrastructure services. The generic AM interfaces to these standard APIs using plugin *handler* modules.

E3 **Leverage shared third-party substrates through their native IaaS interfaces.** This compatibility with standard back-end infrastructure control services offers a path to bring independent resource providers into the federation. The provider deploys an off-the-shelf IaaS service and "wraps" it with an AM to link it into the testbed federation.

E4 **Enable substrate owners to contribute resources on their own terms.** Participating providers are autonomous: they are empowered to approve or deny any request according to their policies. Providers allocate virtual resources with attached QoS properties for defined intervals; the callers determine what resources to request and how to expose them to applications. Resource allotments are visible to both parties and are controlled by the providers. These principles are similar to those put forward for the *exokernel* extensible operating system [15] developed at MIT in the 1990s; the exo name also pays homage to that project [13].

Based on these principles ExoGENI provides a framework to incorporate "outside" resources and infrastructure services into a federation and to orchestrate their operation. For example, providers may deploy new cloud sites using open-source cloud stacks, such as Eucalyptus [18] or OpenStack, which support the de facto

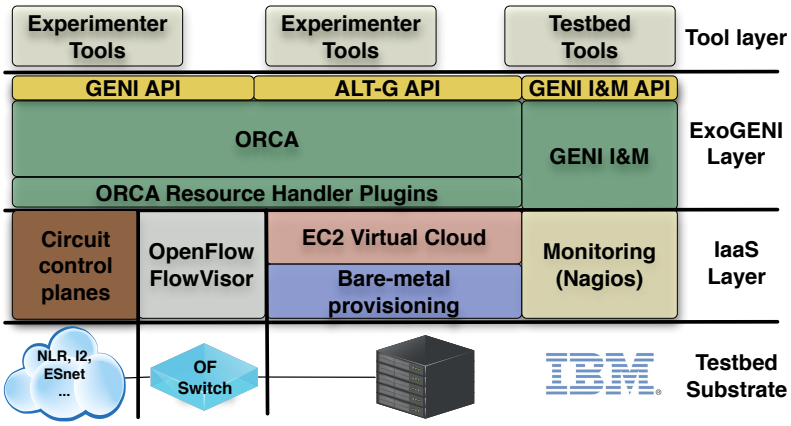

Figure 2: Conceptual view of the ExoGENI software stack. An IaaS layer consists of standard off-the shelf software and services for server clouds, network transport services, and OpenFlow networking, which control the testbed substrate. Testbed users and tools access the testbed resources through GENI APIs and an alternative API (labeled Alt-G in the figure) based on semantic resource models. The ORCA resource leasing system tracks resource allocation and orchestrates calls to *handler* plugins, which invoke the APIs for services in the IaaS layer. The monitoring system has a similar structure.

standard Amazon EC2 IaaS cloud API. Common APIs such as OSCARS [10] are also emerging for transport network circuit services. Providers may deploy these and other systems independently; once a system is deployed we can install a front-end orchestration service (AM) to link it into the federation without interfering with its other functions and users. The AM may be operated by the provider itself or by a delegate or authorized client.

2.1 ExoGENI Control Software

The control software for ExoGENI was developed and refined in an ongoing collaboration between RENCI and Duke University to create a GENI testbed "island" around the Breakable Experimental Network [1] (BEN), a multi-layer optical testbed built by the State of North Carolina and managed by RENCI. The project grew into a more comprehensive effort to support a federated IaaS system linking BEN and other infrastructure systems under orchestrated control and a common authorization framework. Some early results from the project have been reported in previous publications [3, 2, 24, 16].

An initial goal of the project (2008) was to build a native circuit service for BEN based on the Open Resource Control Architecture (ORCA [5]). ORCA is an outgrowth of earlier projects in networked cloud computing at Duke, funded by NSF and IBM. It is based on the SHARP federation model [9] and the plugin architecture of the Shirako resource leasing core [14], with extensions for automated control policies added for Automat [25]. The project developed a set of plugin modules for ORCA, which are used in ExoGENI.

Building the native BEN circuit service presented an interesting test case for the ORCA control framework. We built policy plugins that plan requested paths through the BEN network by issuing queries on *semantic resource models*, which are logic-based declarative descriptions of the network expressed in an extended variant of the Network Description Language (NDL [11, 7, 12]). We also built handler plugins that manage paths by forming and issuing commands to network devices over the BEN management network, based on path descriptions generated by the queries.

BEN is one of several circuit providers for ExoGENI. We later implemented new ORCA plugins to interface to external circuit APIs: National LambdaRail's Sherpa FrameNet service and OSCARS, which is used in the national circuit fabrics ESnet and Internet2 ION.

We also extended other handlers to drive EC2 cloud APIs. With the emergence of Eucalyptus [18] we focused our development on integration with standard EC2-compatible cloud stacks, replacing our older cloud software called Cluster-on-Demand [6]. ExoGENI rack sites now use these plugins to drive the OpenStack cloud service.

These steps led us to the ExoGENI software architecture, which uses ORCA to orchestrate a set of participating virtual infrastructure providers through their native IaaS interfaces. Figure 2 depicts the software stack. The ORCA portions of the stack run in different ORCA-based servers: ORCA is a toolkit for building aggregate managers (AMs) for the rack sites and other providers, together with related coordination services (Section 4).

Users and their tools invoke testbed APIs to instantiate and program virtual resources from participating providers. A *slice* is a set of virtual resources under common user control. A slice may serve as a container or execution context to host an application or network service. An ExoGENI slice may contain a network topology with programmable nodes and links—a virtual distributed environment [21]. The links in the topology comprise the slice *dataplane*. Software running within a slice may manage its dataplane as a private packet network using IP or alternative protocol suites at the discretion of the slice owner.

A slice may span multiple sites and link to other GENI resources or other external resources as permitted by peering and interconnection agreements. ExoGENI slices are isolated: they interact with the outside world through controlled interfaces, and the resources in a slice may have quality-of-service properties defined by the providers.

2.2 Relationship to other GENI Testbeds

ExoGENI is significant in part because it offers our first opportunity to evaluate the federated IaaS model in a production testbed. The ExoGENI principles represent a departure in the GENI effort, whose current standards evolved from testbeds that were established and accepted by the research community at the start of the GENI program in 2007: PlanetLab [19], Emulab [23], and ORBIT [20]. Each testbed developed its own control software to manage substrates that are permanently dedicated to that testbed and under the direct control of its central testbed authority.

The ExoGENI testbed is the first GENI-funded substrate whose control software departs from that model and instead uses standard virtual infrastructure services, which may be deployed and administered independently and/or shared with other uses. We intend that ExoGENI will serve as a nucleus for a larger, evolving federation that encourages participation from independent cloud sites, transport networks, and testbed providers, beyond the core GENI-funded substrate. An important goal of the project is to provide a foundation for organic and sustainable growth of a networked intercloud through a flexible federation model that allows private cloud sites and other services to interconnect and share resources on their own terms.

The ExoGENI model offers potential to grow the power of the testbed as infrastructure providers join and their capabilities continue to advance. In time these advances may enable not just real deployment of innovative distributed services but also new visions of a Future Internet.

This goal requires an architecture that supports and encourages federation of sites and providers and exposes their raw IaaS capabilities, including QoS capabilities, to testbed users through common APIs. It requires a different structure from the GENI predecessors, whose primary uses have been to evaluate new ideas under controlled conditions (for Emulab and ORBIT) and to measure the public Internet as it currently exists (for PlanetLab). PlanetLab has enabled development of innovative distributed services in the real world, but it is limited as a deployment platform because it supports only best-effort resource allocation, limits use of modified kernel software to user-mode virtualization, and depends on the existing Internet for its dataplane.

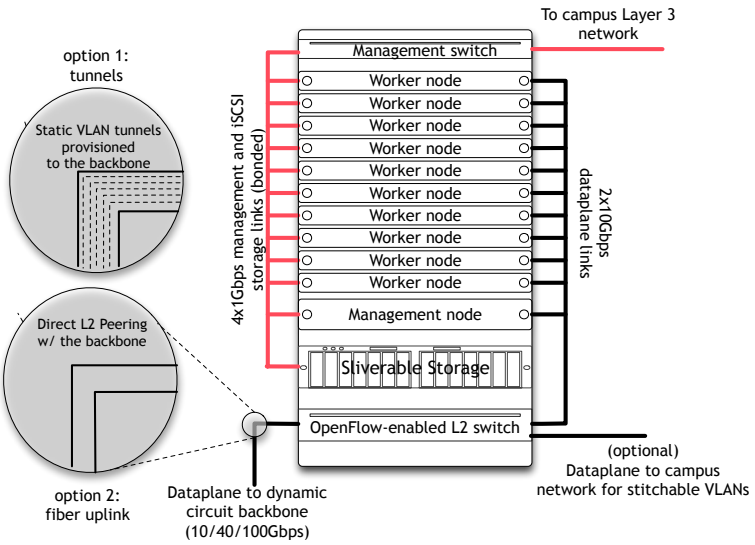

Figure 3: Structure of an ExoGENI site rack for the initial deployment. Each rack has low-bandwidth IP connectivity for management and a high-bandwidth hybrid OpenFlow switch for the slice dataplanes. The site ORCA server controls L2 dataplane connections among local nodes and external circuits.

3 ExoGENI Services and Interconnections

Each ExoGENI cloud site includes a packaged rack with a small cloud server cluster and an integrated OpenFlow network, built by our industry partner IBM. The initial funding will deploy 14 rack sites at universities and research labs across the United States. Each of the sites is capable of supporting about 100 virtual machines, based on an EC2-compatible IaaS cloud service (OpenStack with Linux/KVM).

Figure 3 depicts the rack components and connections. The nodes in the initial racks are x3650 M3 and M4 IBM servers. The *worker nodes* are the server substrate for dynamic provisioning of nodes for slices. A single *management node* (head node) runs the control servers for the site, including the OpenStack head and ORCA servers. The rack also includes an iSCSI storage appliance for images, instrumentation data, and other needs.

All components are connected to a *management switch*, which has an L3 connection to the campus network and from there to the public Internet. This switch is used for intra-site access to the iSCSI storage, for remote management by the testbed operator (RENCI) through a VPN appliance (not shown), and for slice connectivity to the public Internet. Each worker has multiple 1Gbs ports for these uses.

A separate *dataplane* switch carries experiment traffic on the slice dataplanes. It is the termination point for L2 links to external circuit providers and to VLANs or subnets on the host campus network, if permitted by the host campus. The initial racks have an IBM/BNT G8264R 10G/40G OpenFlow-enabled hybrid L2 switch with VLAN support. Each worker node has two 10Gbps links to the dataplane switch.

Slice owners can access their nodes over public IP through a management interface. Access is by root *ssh* with a public key specified by the slice owner. Nodes may also have public IP addresses if permitted by the host campus. Public IP access is managed by OpenStack and proxied through its head node.

3.1 Circuit Backbones

Each circuit provider offers a point-to-point Ethernet service among specified points-of-presence on its network. ExoGENI can use dynamic circuit services offered by major national fabrics (NLR, Internet2, ESnet) through their native APIs (e.g., OSCARS). Two rack sites (RENCI and the *exo-dev* development cluster at Duke

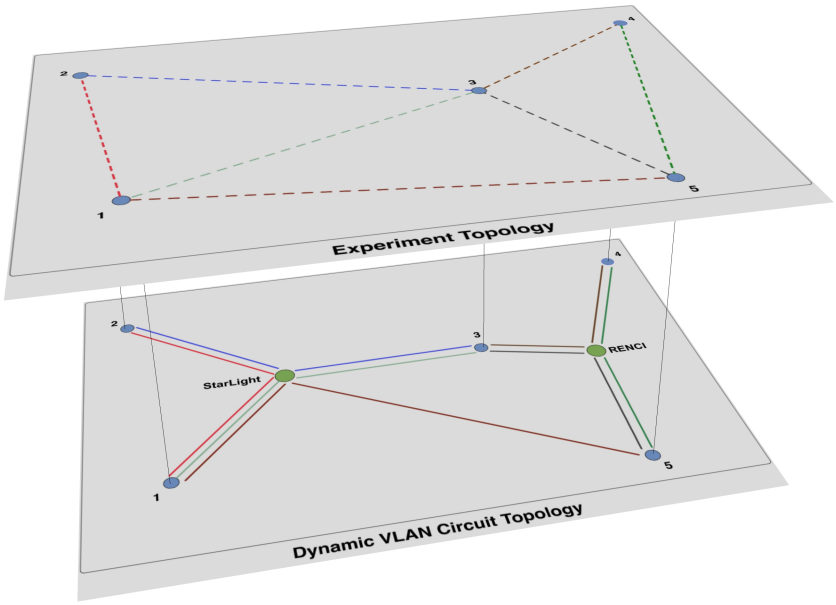

Figure 4: A virtual topology mapped to a circuit fabric substrate with network exchange points. The exchanges are ExoGENI aggregates that stitch adjacent circuits into end-to-end paths with VLAN tag remapping. The exchanges bridge gaps among multiple circuit providers and reduce the need for common VLAN tags for circuits that span rack sites.

University) have direct links to the BEN network, which connects to 10Gbps ports on NLR FrameNet and I2/ION.

Rack sites connect to the circuit backbones either through dedicted fiber or a static pool of pre-provisioned VLAN tunnels that traverse campus networks and/or regional networks (RONs). The host campus presents these circuit VLANs to the rack dataplane switch. ExoGENI coordinator services use these various L2 capabilities to construct end-to-end circuit paths requested for a slice dataplane, passing through intermediate network exchange points as needed to remap VLAN tags or bridge gaps between multiple circuit providers (Section 3.2). Authorized slices may also link to other VLANs entering the dataplane switch from the campus network or backbone, such as the GENI Meso-Scale OpenFlow Waves on Internet2 and NLR.

3.2 Network Exchange Points

ExoGENI has connectivity to a wider range of circuit providers through two network exchange points: a RENCI-operated exchange on BEN and the StarLight facility in Chicago, which lies at the intersection of various national and international networks. RENCI has deployed switches with VLAN tag translation (Cisco 6509) to these exchange points, each controlled by an AM.

We use the exchange points to stitch end-to-end circuits that span multiple transport networks with no common inter-domain circuit service, i.e., an "air gap" between circuit fabrics. The exchanges can also remap VLAN tags to link end-to-end circuits between edge sites, for use when no tag remapping option is available along the direct path and the sites have no common VLAN tags available in their static pools. We have demonstrated use of the exchanges to bridge connections among different circuit providers and to interconnect member sites of GENI's "Cluster-D" through various regional networks and NLR. Some sites without NLR access have dedicated connectivity to StarLight through regional or commercial providers and can bridge circuits to other networks through StarLight.

Figure 4 shows an example of how VLAN translation at fixed exchange points can enable more dynamic

slice topologies for ExoGENI. The figure shows a slice's virtual topology in the top plane and the underlying circuits and their remapping in the bottom plane.

ExoGENI will use its point-of-presence at StarLight to enable dynamic connectivity to other GENI substrates, ESnet, Internet2, NLR, and DOE's 100 Gbps ANI testbed (via a planned rack site at NERSC/LBL). StarLight also links ExoGENI to various international testbed partners. For example, we will have the opportunity to connect to our partners from Fraunhofer FOKUS and their Teagle control framework [4] via StarLight and GEANT.

3.3 Site Software

Each ExoGENI rack site is reconfigurable at the physical layer, providing a flexible substrate for repeatable experiments. Most testbed users will use a virtual machine (VM) service (OpenStack), but we also have bare-metal imaging based on the open-source xCAT provisioning system [8], which is developed and maintained by IBM. Currently we use xCAT only to deploy the OpenStack worker nodes, and do not expose it through the rack AM. Programmatic bare-metal imaging is on our roadmap for performance-critical applications such as virtual routers.

The head node runs the OpenStack head and various network proxies and auxiliary services for GENI and ORCA. These services include a native ORCA-based AM for the site, and a second ORCA-based server that proxies the GENI APIs (Section 4). Each rack AM includes a cloud handler plugin to invoke EC2/OpenStack APIs and an *ImageProxy* server to obtain node images named by a URL in the request. An *image* file specifies a canned operating system and application stack selected by the user. ImageProxy is a stand-alone caching server that enables the cloud site to import images on demand from the network (Section 3.4).

The ORCA cloud handler also invokes a cloud service extension with a command set to instantiate dataplane interfaces on VMs when they are requested, stitch interfaces to adjacent virtual links, and configure interface properties such as a layer-3 address and netmask. This extension is known as *NEuca*: we implemented it for Eucalyptus before porting it to OpenStack/Nova. We plan to integrate it with the OpenStack/Quantum framework as it develops. Both NEuca and ImageProxy are independent of ORCA.

Each rack also runs OpenFlow control services, including a FlowVisor [22] proxy to mediate access from OpenFlow controllers to OpenFlow datapaths on the rack dataplane switch. An ExoGENI slice may designate an OpenFlow controller to manage traffic on its local dataplane links; the ORCA cloud handler invokes the proxy to authorize the controller to manage traffic on VLANs assigned to those links.

3.4 Image Management

A key orchestration challenge for multi-domain networked clouds is uniform management of images to program the node instances. With standard IaaS cloud stacks following the Amazon EC2 model each cloud site requires some local user to pre-register each image with the site's cloud service, which then generates an image token that is local to that site. Networked clouds need a way to manage images across the member sites of a federation.

In our approach the creator of an image registers it at some shared image depository and names it by a URL. A request to instantiate a VM names the image by a URL and a content hash for validation. The AM's cloud handler plugin passes the image URL and hash to the local *ImageProxy* server. The ImageProxy fetches and caches any image components if they are not already cached, and registers the image with the local cloud service if it is not already registered. It then returns a local token that the AM cloud handler may use to name the image to the local cloud service when it requests a VM instance.

In principle, ImageProxy can work with any image server that supports image fetch by URL, e.g, the various Virtual Appliance Marketplaces operating on the web. It also supports BitTorrent URLs to enable scalable content swarming of images across many cloud sites.

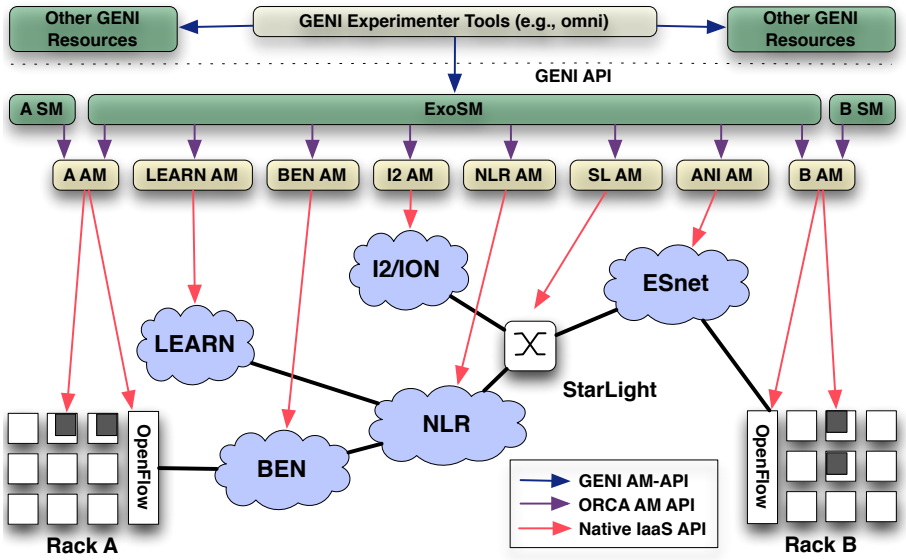

Figure 5: ExoGENI as a set of GENI aggregates. There is one ORCA Aggregate Manager (AM) for each ExoGENI site or other provider. For third-party IaaS services the AM authenticates using an ExoGENI-linked identity as a registered customer of the provider. Each ExoGENI site also runs an ORCA Slice Manager (SM) that proxies the GENI API, exposing the site as an independent aggregate within GENI. One or more ExoGENI-wide SMs (ExoSM) expose the entire ExoGENI testbed to GENI as a single GENI aggregate. The ExoSM uses native ORCA brokering, slicing, and stitching features to construct complete virtual topologies within ExoGENI slices.

4 ExoGENI and the GENI Federation

ExoGENI may be viewed as a group of resource providers (aggregates) within a larger GENI federation. ExoGENI itself is an instance of the GENI architecture and supports GENI APIs, and also supports additional native ORCA interfaces and capabilities that are not yet available through standard GENI APIs. This section outlines how ExoGENI integrates with GENI and extends the GENI federation model.

4.1 Aggregates

GENI aggregates implement standard GENI APIs for user tools to request resources. The ExoGENI AMs, as described in this paper, are orchestration servers built with the ORCA toolkit. They support internal ORCA protocols rather than the standard GENI aggregate APIs. The GENI APIs are evolving rapidly to support more advanced control and interconnection of slices and rich resource representations and credential formats. Rather than implementing the GENI API directly in the AMs, the initial ExoGENI deployment proxies them through a GENI plugin for separate ORCA-based servers called *Slice Managers* (SM), as described below.

This approach enables ExoGENI to present a secure and flexible interface to the GENI federation and to support standard user tooling for GENI. At the same time ExoGENI supports end-to-end slice construction across the ExoGENI aggregates, based on native ORCA capabilities. The AM operator interface also allows local policies that limit the local resources available to the testbed over time. This feature enables ExoGENI providers to hold back resources from the testbed for other uses, according to their own policies.

4.2 GENI Federation: Coordinators

GENI aggregates delegate certain powers and trust to *coordinator* services. The coordinators help aggregates to cooperate and function as a unified testbed. For example, GENI currently defines coordinators to endorse and monitor participating aggregates, authorize and monitor use of the testbed for approved projects, and manage user identities and their association with projects. The GENI coordinators are grouped together under the umbrella of a GENI *Clearinghouse*, but they act as a group of distinct services endorsed by a common GENI root authority.

The GENI federation architecture allows participating aggregates to choose for themselves whether to accept any given coordinator and what degree of trust to place in it. These choices are driven by federation governance structure. GENI has opted for a hierarchical governance structure for its initial trial deployment, for reasons of safety and simplicity. To join the GENI federation an aggregate must enter into certain agreements, including compliance with various policies and export of monitoring data to GENI coordinators.

The aggregates in the initial ExoGENI deployment will enter into mandated agreements with GENI and accept and trust all GENI coordinators. Specifically, ExoGENI trusts GENI-endorsed coordinators to certify users, issue keypairs to users, authorize projects, approve creation of new slices for projects, authorize users to operate on approved slices, and endorse other aggregates in GENI. The GENI coordinators in turn delegate some identity management functions to identity systems operated by other GENI testbeds and participating institutions (Shibboleth/inCommon).

4.3 ExoGENI Coordinators

ExoGENI provides additional coordinators and APIs for managing resources and configuring end-to-end virtual topologies within the ExoGENI testbed itself. The ExoGENI services are based on the ORCA control framework. Figure 5 depicts some of the key services and their interactions.

User requests enter an ORCA control system through a server called a *Slice Manager* (SM), which invokes the AMs to obtain the requested resources. In general, an ORCA slice manager runs on behalf of slice owners with no special trust from other services: the SM is in essence an extensible user tool that runs as a recoverable server. However, in ExoGENI the SMs function as coordinators that provide a trusted interface to the rest of the GENI federation. The ExoGENI AMs accept requests only from trusted SMs endorsed by the testbed root. The SMs run plugins that expose the GENI standard APIs for GENI users and tools. They are responsible for validating the GENI authorization for the slice and the user identity in each request, and translating between GENI RSpec-XML resource descriptions and the logic-based semantic resource models used in ORCA.

Internally to ExoGENI the SMs interact with other ORCA coordinators called *brokers*, which collect and share information about ExoGENI aggregates, including their advertised resources, services, and links to circuit providers. Each ORCA broker is trusted by some set of aggregates to advertise and offer shares of their resources. A broker may also coordinate resource allocation across its aggregates and guide or limit the flow of requests to the aggregates, e.g., based on scheduling policies and capacity constraints, as described in previous work on SHARP resource peering [9, 14]. Each request for resources is approved by a broker before the SM passes it to an AM.

There are two distinct SM configurations in ExoGENI. Each ExoGENI rack site runs an SM called a *site SM*. Each site SM exposes its rack site as a distinct GENI aggregate. Requests to a site SM can operate only on the local rack: each site SM uses a local broker with a share of local site resources.

In addition, a global ExoGENI SM called an *ExoSM* can access all aggregates within the ExoGENI testbed. There may be any number of ExoSMs, but the initial deployment has a single ExoSM. The ExoSM exposes ExoGENI as a single GENI aggregate. The ExoSM offers a unified view of the testbed, and supports virtual topology mapping and circuit path planning across the ExoGENI aggregates, including the circuit providers and network exchanges. User requests to ExoSM may request a complete slice topology spanning multiple

sites and circuit providers. The ExoSM coordinates construction and stitching of the end-to-end slice.

An ExoSM plans and sequences the resource requests and stitching actions to the AMs based on declarative semantic models that advertise the resources and capabilities of the participating aggregates [3, 24, 2]. The ExoSM obtains these domain models from a common testbed-wide broker (*ExoBroker*) that is accepted by all ExoGENI aggregates and receives all of their advertisements. The ExoBroker also facilitates allocation of common VLAN tags from static tunnel pools when needed.

4.4 Integration with GENI

GENI users and their tools choose for each slice whether to access the ExoGENI testbed as a single aggregate (through the ExoSM) or as a collection of distinct site aggregates (through the site SMs). External GENI tools can interact with ExoGENI site aggregates based on the current GENI architecture and API, in which aggregates are loosely coupled except for common authorization of users and slices. Each ExoGENI slice may also link to other GENI resources to the extent that the standard GENI tools and APIs support that interconnection.

At the same time, the ExoSMs and ExoBroker allow ExoGENI to offer capabilities for automated cross-aggregate topology embedding, stitching, and resource allocation within the ExoGENI testbed. These capabilities are currently unique within GENI. They are based on coordinator services, APIs, resource representations, and tools that are not part of a GENI standard. In particular, GENI defines no coordinators for resource management, so cooperation among GENI aggregates is based on direct interaction among AMs or exchanges through untrusted user tools. GENI is developing new extensions that would offer similar capabilities for automated configuration of cross-aggregate virtual networks.

ExoGENI also differs from current GENI practice with respect to the usage model for OpenFlow networks. GENI views an OpenFlow datapath as a separate aggregate that allocates the right to direct network traffic flows matching specified packet header (flowspace) patterns, which are approved manually by an administrator. In ExoGENI, OpenFlow is an integrated capability of the ExoGENI rack aggregates, rather than a distinct aggregate itself. ExoGENI slices may designate OpenFlow controllers to direct network traffic within the virtual network topology that makes up the slice's dataplane. ExoGENI is VLAN-sliced: each virtual link corresponds to a unique VLAN tag at any given point in the network. The handler plugins of the ExoGENI rack AMs authorize the controllers automatically, so that the designated controllers may install flow entries in the datapath for VLANs assigned to the slice's dataplane. We believe that this approach can generalize to other OpenFlow use cases in GENI and cloud networks.

5 Conclusion

This paper describes the design of the ExoGENI testbed, which addresses the goals of GENI by federating diverse virtual infrastructure services and providers. This approach offers a path to leverage IaaS advances and infrastructure deployments occurring outside of GENI. At the same time, it offers a path to bring GENI technologies to bear on key problems of interest outside of the GENI community: linking and peering cloud sites, deploying multi-site cloud applications, and controlling cloud network functions.

ExoGENI offers an architecture for federating cloud sites, linking them with advanced circuit fabrics, and deploying multi-domain virtual network topologies. The initial deployment combines off-the-shelf cloud stacks, integrated OpenFlow capability, linkages to national-footprint research networks and exchange points with international reach.

ExoGENI and its ORCA control framework enable construction of elastic Ethernet/OpenFlow networks across multiple clouds and circuit fabrics. Built-to-order virtual networks are suitable for flexible packet-layer overlays using IP or other protocols selected by the owner. IP overlays may be configured with routed

connections to the public Internet through gateways and flow switches. ExoGENI can also serve a broader role as a model and platform for future deeply networked cloud services and applications.

6 Acknowledgements

We thank RENCI, NSF, IBM, and the GENI Project Office (GPO) at BBN for their support. Many colleagues at GPO and other GENI projects have helped work through issues relating to ExoGENI.

References

[1] I. Baldine. Unique Optical Networking Facilities and Cross-Layer Networking. In *Proceedings of IEEE LEOS Summer Topicals Future Global Networks Workshop*, July 2009.

[2] I. Baldine, Y. Xin, D. Evans, C. Heermann, J. Chase, V. Marupadi, and A. Yumerefendi. The Missing Link: Putting the Network in Networked Cloud Computing. In *ICVCI: International Conference on the Virtual Computing Initiative (an IBM-sponsored workshop)*, 2009.

[3] I. Baldine, Y. Xin, A. Mandal, C. Heermann, J. Chase, V. Marupadi, A. Yumerefendi, and D. Irwin. Autonomic Cloud Network Orchestration: A GENI Perspective. In *2nd International Workshop on Management of Emerging Networks and Services (IEEE MENS '10), in conjunction with GLOBECOM'10*, Dec. 2010.

[4] N. Blum, T. Magedanz, F. Schreiner, and S. Wahle. A Research Infrastructure for SOA-based Service Delivery Frameworks. In *Proceedings of the 5th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom)*, Washington DC, USA, April 2009.

[5] J. Chase, L. Grit, D. Irwin, V. Marupadi, P. Shivam, and A. Yumerefendi. Beyond Virtual Data Centers: Toward an Open Resource Control Architecture. In *Selected Papers from the International Conference on the Virtual Computing Initiative (ICVCI)*, May 2007.

[6] J. S. Chase, D. E. Irwin, L. E. Grit, J. D. Moore, and S. E. Sprenkle. Dynamic Virtual Clusters in a Grid Site Manager. In *Proceedings of the 12th International Symposium on High Performance Distributed Computing (HPDC)*, June 2003.

[7] F. Dijkstra. *Framework for Path Finding in Multi-Layer Transport Networks.* PhD thesis, Universiteit van Amsterdam, 2009.

[8] E. Ford. From Clusters To Clouds: xCAT 2 Is Out Of The Bag. *Linux Magazine*, January 2009.

[9] Y. Fu, J. Chase, B. Chun, S. Schwab, and A. Vahdat. SHARP: An Architecture for Secure Resource Peering. In *Proceedings of the 19th ACM Symposium on Operating System Principles*, October 2003.

[10] C. Guok, D. Robertson, M. Thompson, J. Lee, B. Tierney, and W. Johnston. Intra and Interdomain Circuit Provisioning Using the OSCARS Reservation System. In *Proceedings of the 3rd International Conference on Broadband Communications, Networks and Systems (BROADNETS)*, 2006.

[11] J. Ham, F. Dijkstra, P. Grosso, R. Pol, A. Toonk, and C. Laat. A Distributed Topology Information System for Optical Networks Based on the Semantic Web. *Journal of Optical Switching and Networking*, 5(2-3), June 2008.

[12] J. V. Ham. *A Semantic Model for Complex Computer Networks.* PhD thesis, University of Amsterdam, April 2010.

[13] D. Irwin, J. Chase, L. Grit, and A. Yumerefendi. Underware: An Exokernel for the Internet? Technical report, Duke University Department of Computer Science, January 2007.

[14] D. Irwin, J. S. Chase, L. Grit, A. Yumerefendi, D. Becker, and K. G. Yocum. Sharing Networked Resources with Brokered Leases. In *Proceedings of the USENIX Technical Conference*, June 2006.

[15] M. F. Kaashoek, D. R. Engler, G. R. Ganger, H. M. Briceno, R. Hunt, D. Mazieres, T. Pinckney, R. Grimm, J. Janotti, and K. Mackenzie. Application Performance and Flexibility on Exokernel Systems. In *Proceedings of the Sixteenth Symposium on Operating Systems Principles (SOSP)*, October 1997.

[16] A. Mandal, Y. Xin, P. Ruth, C. Heerman, J. Chase, V. Orlikowski, and A. Yumerefendi. Provisioning and Evaluating Multi-Domain Networked Clouds for Hadoop-Based Applications. In *Proceedings of the 3rd International Conference on Cloud Computing Technologies and Science 2011 (IEEE Cloudcom '11)*, December 2011.

[17] P. Mell and T. Grance. The NIST Definition of Cloud Computing. Special Publication 800-145, Recommendations of the National Institute of Standards and Technology, September 2011.

[18] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov. The Eucalyptus Open-Source Cloud-Computing System. In *Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID)*, May 2009.

[19] L. Peterson, A. Bavier, M. E. Fiuczynski, and S. Muir. Experiences Building PlanetLab. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation (OSDI)*, November 2006.

[20] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh. Overview of the ORBIT Radio Grid Testbed for Evaluation of Next-Generation Wireless Network Protocols. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2005.

[21] P. Ruth, X. Jiang, D. Xu, and S. Goasguen. Virtual distributed environments in a shared infrastructure. *Computer*, 38(5):63–69, 2005.

[22] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar. Can the Production Network Be the Testbed? In *Proceedings of the Symposium on Operating System Design and Implementation (OSDI)*, October 2010.

[23] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar. An integrated experimental environment for distributed systems and networks. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI)*, pages 255–270, December 2002.

[24] Y. Xin, I. Baldine, A. Mandal, C. Heermann, J. Chase, and A. Yumerefendi. Embedding Virtual Topologies in Networked Clouds. In *6th ACM International Conference on Future Internet Technologies (CFI)*, June 2011.

[25] A. Yumerefendi, P. Shivam, D. Irwin, P. Gunda, L. Grit, A. Demberel, J. Chase, and S. Babu. Towards an Autonomic Computing Testbed. In *Workshop on Hot Topics in Autonomic Computing (HotAC)*, June 2007.