GMIP: Design of the GENI Management Information Plane for resource and network management, *v.2.0*

Karen Sollins
Deliverable: INFOSUBSTR S3.b
Date: August 26, 2011

Introduction

In this project we address the issue of the infrastructure required to manage the GENI facilities. As outlined in our previous paper, **A Network Management Information Plane for GENI: the architectural challenges**, the issues in management of such a distributed, federated, layered facility will require a rich information infrastructure. The challenges range from availability of the information both horizontally in support of distribution and federation, as well as vertically because management is necessary within each layer, and between the layers defined by the GENI architecture. In addition, because the environment reflects a federation of not only resources but also administrative controls, usage policies with respect to the management capabilities must be an integral aspect of the management "system", and hence an acceptable and usable authentication, policy and authorization capability is necessary.

The capability on which we focus in this work is the information plane required to achieve network management in this context. We call this the GENI Management Information Plane or GMIP. We recognize that network management tools reflect an increasing and changing set of capabilities and tools, as our understanding and requirements for management expand. Hence, we focus on what is required to support that expanding and evolving set of tools: the information substrate on which they all must sit and which each may use for both the basis of its operation and the storage of its results.

To this end, we identified in that earlier paper five key design principles for the information substrate system:

- *Minimization of duplicate information collection*: in general we recognize that duplicate data collection is likely to be wasteful perhaps both in terms of storage and possibly in terms of bandwidth to deliver it to storage facilities. We note that there are several reasons for duplicate information collection, but in general suggest minimizing such duplication.
- *Wide and easy availability of information on demand*: by standardizing and enabling requests for information which is sharable, network management can be improved by allowing for capabilities to build on each other.
- *Minimization of usage of resources*: because the resources used for network management in general come at a cost to the communication the network is providing, it is important to be parsimonious with usage of storage, communication and computation resources.
- *Customary information transformation should be widely possible and available*: whether in order to reduce use of resources or to support restrictive security, privacy, or

confidentiality policies, it should be possible to abstract, summarize, and sample from existing information to reduced versions of it.

- *Definition and enforcement of policy controls*: It will be necessary to not only express the kinds of policies that will be required across the boundaries of the components of the federation, but also have confidence in the enforcement of those policies, as network management information is made available across those same boundaries.

This document will first clarify the more specific purposes of the information substrate of a GENI network management system, the GENI Management Information Plane or GMIP. It will then provide a high-level description of the components and the system design. That is followed by more detail on each of the components, the internal and external interfaces, and concludes with an evaluation of the assumptions, benefits and risks.

Purpose

As argued in our earlier document, we can postulate that network management problems may span both the multiple layers of the GENI architecture and boundaries that define the component federants. The problem we are addressing with an information plane for network management is to enable a unified approach to addressing the multiple-layer, multiple federant management issues. There are three key aspects to providing this unified approach: making the information accessible (both storage and finding the appropriate information), understanding the information, and providing acceptable access control. Thus the problem we are addressing here is providing effective and controlled access to the information required to solve network management problems.

When one examines both the existing capabilities and the nature of the potential problems that network management may be called upon to address, one realizes that there is a mismatch. In the GENI architecture, there are capabilities placed at two layers, as demonstrated in Figure 1 of our previous document, the Ops and Management Plane, which provides low-level operational facilities management, without an adequate approach to federation, and the Measurement Plane, which provides information exclusively to the individual experimental networks created by the Slice Plane. First, there is little or no capability for access-controlled management across a set of federants. The current GMOC approach is simply to collect all data that might be made available from all federants without any access control policy. Second, although unwanted behaviors in experiments may be the result of issues at any of the three operational layers, the experiment itself, the Slice Plane and the Operations Plane, there is no ability for an analysis or management tool to drill down through these layers to analyze and perhaps address problems in behaviors of experiments. Such a drill across federants or down is dependent on access to the information that reflects the behaviors, contexts, and configurations that together reflect what the experimenter is seeing. It is important to note that these same kinds of controlled access requirements may be important to the operators. An experimenter may collect experimental behavior measurements that themselves in some perhaps restricted form may help the systems operators and administrators diagnose and address problems they are seeing. Problems may become apparent to either the experimenter or the network operators. Evidence for problems may begin with an experiment apparently not operating as expected. This may be due to something within the experiment itself, something in the Slice Plane, or something at the lower level network plane, where the current "Ops and Management Plane" exists. In fact, problems may arise due to interactions across those layers or between federated components. This complex web of problems argues that a common, controlled information substrate is needed for effective GENI network management or the GENI Management Information Plane, GMIP.

Definition of high-level components: system design

The discussion and challenges identified above suggest that the management information plane must be orthogonal to the layers of the architecture, and thus accessible to all of them and support distributed administration to meet the requirements of federation and performance.

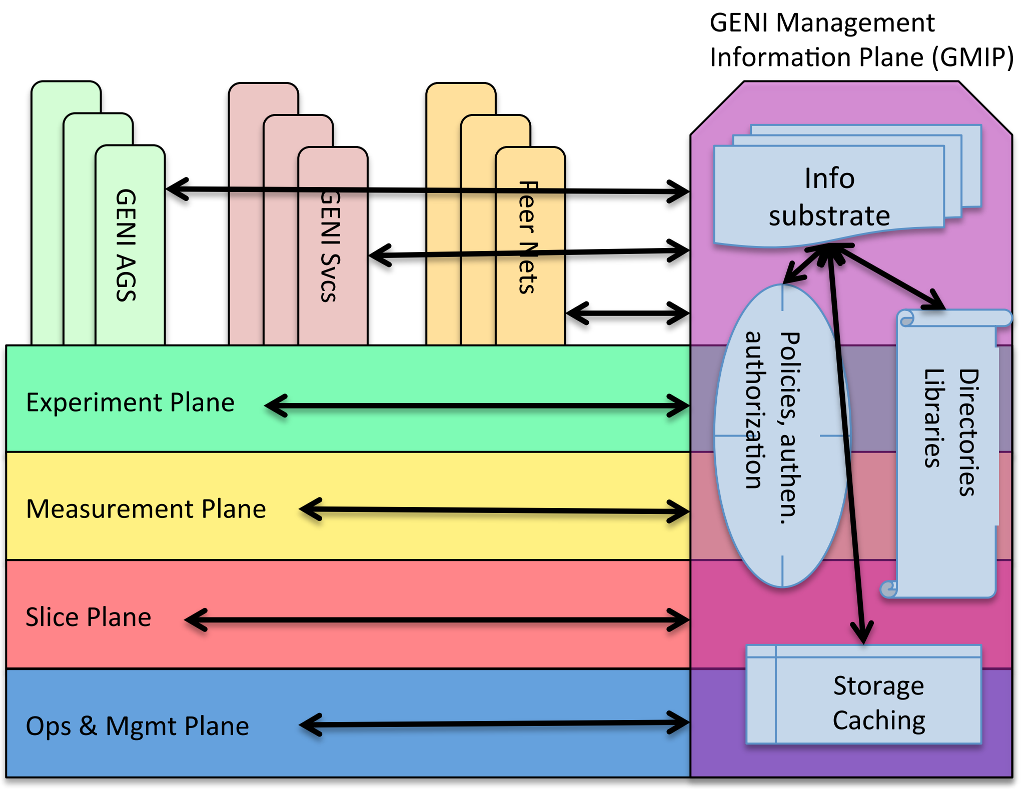

Figure 1: GENI Architecture with GENI Management Information Plane (GMIP), providing access to and from all layers and all elements of the GENI network.

As depicted in Figure 1, we identify four key high-level components of the GMIP

1. **Information substrate** the information substrate, that supports named information objects and the metadata that provides association of each object with its type/kind/class in the ontology, its provenance, its ownership and the policies that control access to it;
2. **Policies, authentication, and authorization:** the access control element that supports access control policies, the infrastructure for principal identification, and authentication, and authorization enforcement;
3. **Directories and Libraries:** the information and ontology directory services, that support finding target information and the ontology definitions by which they are labeled;
4. **Storage and caching:** the substrate that provides longevity of information and efficient delivery of it, through storage and caching.

The choice of these four elements of the system is derived directly from the architectural challenges and our specific design principles.

In this context, we can identify four primary kinds of activities that are necessary for such a subsystem to function effectively. First, the information ontology needs to be defined, presumably using an editing tool appropriate to the ontology language selected. The ontology will define the types or kinds of individual information objects, and, as needed, the ability to reason about relationships among the kinds of information objects. Second, there will need to be registration of authorized principals. Third particular information objects need to be both registered in a directory or lookup service and stored as needed in the storage and caching subsystem along with their access control policies. Finally, a user or application on presentation of authentication, authorization or credential information might seek out and acquire access to one or more stored information objects. The user or application seeking access may reside in any of the architectural layers and in any of the federated component systems, which implies that each of these subsystems must be designed to run across both vertical and horizontal components.

Protocols will need to be designed for each kind of interaction. There will need to be a general GMIP protocol for interactions between new or existing GENI layers and management tools and the GMIP itself. In addition, protocols will need to be defined to support the communication among the subsystems and between elements in each subsystem. This document does not go into that level of detail, because, until there is agreement that both the information plane and its elements are necessary and correctly outlined, designing specific protocols is a futile activity.

Low level design for each element: component design issues

In this section we focus on each of the elements of the GMIP, with reference to their relationships, but with a focus on each one independently.

The Information substrate

This subsystem is all about supporting the basic information objects, assuring that access is provided as possible and under access control constraints. It is called upon with the mutual agreement of the directory services and the access control services to provide the appropriate information objects. If it is required, the information service may provide transformational filtering on the information before it is provided. To do this, it is likely to call on the ontology library as well. Thus, the information substrate will do two basic operations, take an information object along with its metadata, such as its definition in the ontology, the applicable access constraining policies, provenance, and other metadata and make sure that its assigned identities are registered in the directory or directories, and its ontology and policy types are in the appropriate libraries. It will then also keep track of or arrange that the object is stored appropriately through communication with the supporting storage and caching subsystem. Since the information substrate contains no state itself, but derives or stores it in the other services, it can be as widely replicated as is useful.

Directories and Libraries

The directories and libraries provide the ability to look up entities that are needed. We divide them into two categories, ontology definitions and information objects. It is important to recognize here that both of these require two forms of “naming”, an attribute-based, user-friendly and a globally unique identifier scheme. The schemes for these are not in the component design, but must be part of the overall design of the system. For attribute-based naming to be effective there must be agreement on the attributes that are useful and their definitions. For globally unique identification there must be agreement on both a scheme and how it will be devised to guarantee uniqueness. Both of these are topics for which many effective schemes work, so we do not choose one here, but note that they are required. Issues such as distributed vs. centralized definitions, timeliness, internationalization, etc. are important aspects of choosing solutions in this space.

The ontology library

The ontology library will have at least one associated ontology editor and two forms of insertion/lookup. The first is attribute-based, in order to provide application and user-friendly insertion and lookup. This lookup service may respond with zero, one or more than one matching ontology definitions. The second is identifier-based, so that the type of an information object can be captured in the metadata of the object in a globally unique form. In this case, the library will respond with zero or one definition. We do not expect this library to be extremely dynamic, nor large, relatively speaking. Hence it is recommended, within the bounds of privacy concerns, that it be fully replicated, for easy access from anywhere.

For the ontology language we recommend that the RelaxNG [Oasis] schema/type language be used as a starting place, because of its simplicity and the fact that it is already being explored within GENI in the Instrumentation and Measurement activities for measurement data descriptions. It has both an XML and more compact non-XML representation. This suggestion is also consistent with the use of XML for the definition of RSpecs, using the Protogeni PGV2 schema. There are at least two kinds of questions that must be examined with respect to the choice of a language. The first is whether it is expressive enough and the second is whether it is compatible with other ongoing work. This choice is compatible with some of the ongoing work within GENI, but may be less so with respect to other less traditional and non-wired communities. For this work, we propose that a simple choice be made first, without eliminating the possibility that the choice will need to be re-evaluated with experience.

The object directory

The object directory, along with the storage system, is a central element of the whole GMIPS service. Without the means of registering and finding information objects, the information plane will not exist. The object directory will also have two forms, user-friendly, attribute-based lookup and globally unique identity based lookup. In general, we expect a significant scaling problem in this service, in part because GENI itself may grow, but more importantly, because this service is about network behaviors, some of which may be used to solve immediate problems, but some of which may be important for long term predictions and analysis, suggesting that the information may need to remain available for an unpredictably long time. This suggests possible exponential growth. Hence, we suggest looking toward DHT approaches, and perhaps over time, layered DHT approaches for resolution of the identifiers. These provide a combination of distribution, resiliency to failures, and efficient management of the information. The CAN [RFH+01] approach is a good candidate for an attribute based DHT. Chord [SMK+01] might be a particularly good choice for the pure, global identifier scheme. Note that these resolution approaches would be only that and distinct from the actual storage; storage and caching are likely to be under the direct management of the authorities that retain control of the information. The community needs to decide whether access control policies need to be applied only to the information itself, or to both the information and the existence of identities. If the latter is the case, the directory service will need to interact with the policy, authentication and authorization service.

Policies, authentication, and authorization

At the core of policy, authentication and authorization subsystem are technologies that GENI has already been exploring and integrating, Shibboleth and ABAC.

Shibboleth is already being used in the GENI context. It provides a single-sign-on, federated approach that allows each institution to use its own technologies and policies for allowing access privileges. It is based on standards such as SAML, and permits a user to gain access to resources, based on a verified identity handled by the person's home institution. In turn, any third party resource owner can set policies for authorized use of the resources, based on the certification provided by the home institution. Shibboleth is being used and extended to meet the requirements of GENI to organize and recognize different kinds of GENI participants.

ABAC or Attributed Based Access Control provides a richer access control approach in which rules are explicit and reasoning about how or why permission is granted or denied is exposed. This allows for later review, forensics and so forth. It also supports a richer decision-making context than Shibboleth. When we consider our objectives in this work, we see that the kinds of policies that might be important may need to be extended from the basic question of yes-or-no access, to decisions about restricted or tempered access, or contextual access, may require both a more complex policy space and the ability to reason about more complex situations and policies. As a simple example, one might allow access to internal aggregate operational information to a student who is actually running a valid experiment in a valid project that appears to have failed, but only if the information can be anonymized adequately. Not only might one want to reason about this, but later it may be important to understand the reasoning, and be assured that only the anonymized information access was provided.

All this argues for several components in this subsystem. The first is Shibboleth, as described above. This may be enhanced with ABAC. A third element of the subsystem is a library of policies. As with the ontology definitions, policy definitions are likely to be too large and complex to be included in an object directly, but might effectively be included by identifier. The identifier then can be found in a policy library. Notice that the policy library itself will need to provide adequate security and privacy, so that it can also only be modified or updated with adequate authorization. In addition, a policy editor is likely to be an important tool, in addition to a tool for viewing and analyzing the reasoning that ABAC derives.

In terms of distribution, Shibboleth is already built as a completely federated system. Each element of the federation can be virtual or not, and can be as distributed or centralized as it chooses. This is also true of ABAC. Since the policies themselves will be specific to sets of information under some specific administrative control (aggregate, slice, or experiment), the policies about them might well be managed as locally or globally as the underlying information itself. As with the ontology library, with adequate privacy and access control policies implemented on the policy library itself, it might be widely distributed and replicated, if that appears to be useful.

Storage and Caching

The storage system is the service that provides persistent for the information objects. It must meet several design criteria:

- Reliability
- Availability
- Efficiency
- Local control

We will address each of these separately.

<u>Reliability</u>: The storage system must provide reasonable guarantees that, if it agrees to store an object, it will do so. It is possible that for different kinds of information different models of reliability will be important. As a simple example, the information about the behavior of a core router may be much more "important" in a general way than the behavior of a home wireless base station, so the model of reliability may be quite different. The results of monitoring the core resources of a large ISP may be stored in a storage system with a great deal of redundancy and a long life-time UPS, while the information about the home wireless hub may simply be stored on a disk attached to that hub. A simple, basic storage system might expose its structure and the guarantees that each component of the system provides, leaving the decisions about where to store objects to the information substrate and eventually in the hands of the users of the whole system. A more sophisticated storage system might reason about the information through a set of attributes and requirements provided by the user to the information system, allowing either the information system or the storage system to store information based on a set of behavioral and policy decisions about the requirements for that information.

Availability: We identify here two distinct issues with respect to availability: (1) availability in the face of failures of specific resources in the network, and (2) availability in the face of misbehavior of a key element of the network that has an extremely wide impact. We will consider each of these separately.

One of the challenges of information storage in a system that is designed to support network management is that the information about a resource is particularly valuable when the resource itself and perhaps its supporting infrastructure are not available. To address this first challenge to availability, we argue that a small number of reliable copies of the information should be available at locations selected to reflect independent failure probabilities from each other. This suggests an intelligent storage scheme that reasons about the centrality, in terms of importance, of the information, to a proposition for replicated placement of that information. Although one might think that everything should be replicated, it is important to understand and take into account the costs of doing that. Such an approach must tradeoff the additional costs of such replication including not only the duplicated storage, but also the network load in distributing the information and any requirements on synchronizing the copies if the data is mutable or streamed.

Our second challenge with respect to availability of information derives from the complex nature of the design of a network, in which the core resources can have extremely wide impact. Hence when something is not working correctly in the core, the number and distribution of reports of failures becomes widespread. To address this latter of a potential implosion of failure reports and analysis and redress requests, we argue for wide replication of the information, in order to avoid further detrimental traffic in the region of such a failure. To achieve this, a dynamic caching scheme will be needed as part of the storage system. It is sometimes possible to recognize *a priori* information that is likely to be "popular" in this sense, but too often failures and critical elements are not recognized until after the fact. To allow the GMIP to respond adequately to such central failures or misbehaviors, a dynamic caching system will be necessary. Three key elements of such a subsystem are (1) adequate capacity, (2) an algorithm for both recognizing when further caching may be needed and for placing information in the caches in locations that are inferred to help mitigate the situation, and (3) an algorithm for timing out or replacing information as it becomes either stale or less important.

Efficiency: Efficiency has to do with making the best use of limited resources. The resources include not only storage for both core storage and caching, but also network resources both to place the information in its storage locations and for accessing the information. As we have previously discussed, it is likely that both of these kinds of resources may be considered as a "tax" on each layer of the architecture. Each layer will use some of its resources to operate and manage itself, passing along the remainder of its resources to the layer above that it supports. Hence the more it uses for itself, the less it can provide to its supported layer. This argues that it should be doing a cost benefit-analysis and use only resources adequate to doing its job acceptably and no more. Hence, massive replication or wide-spread caching of all its monitoring and management information may be overkill and may penalize the layer that it is intending to serve.

Local control: Federation implies that a significant amount of control is retained locally. In GENI, we see this in the response of some of the aggregates to "Emergency stop". Some participants will(and do) say that although they are willing to contribute resources to GENI customers, they also have local customers, and simply because GENI decides that an emergency stop is important, that may not serve the non-GENI customers best interest and hence the aggregate may only take an emergency stop under advisement.

The same idea carries over to monitoring and management information. An aggregate or other federant is not always willing to provide all its operational information to anyone who asks. Reasons for this may include: a) that it is proprietary; b) that it might violate local privacy policies or laws; or c) that it may represent the federant in a light that it does not want made public. As suggested above, one part of controlling exposure of local information may be a set of tools for reducing, abstracting, or obfuscating some aspects of the information. A second is that organizations may want to retain physical control of the information, only exposing it under controlled and approved conditions. This must be possible to make cooperative network management acceptable to these players. Thus, the storage facility must support retention of control over the physical copies as needed by the federants. A completely open, shared "cloud" of storage will not adequately serve the requirements of controlled federation. Local control must be supported in the storage system, covering both the core storage subsystem and the caching subsystem

Summary and Evaluation

The GMIP system described above is designed to meet the design challenges and criteria described in our original document. It must be built utilizing technologies already in play or under consideration within GENI. Thus, for example, it should be integrated with the use of Shibboleth and ABAC already proposed for GENI to manage identity, authentication and authorization control. It should use, as much as possible, the same or closely related specification languages, so that users will not be expected to learn yet another different language syntax and possibly different editor.

What we have proposed here addresses the orthogonal challenges of the GENI architecture, of the cross multiple layer interoperability and demands of federation. In addition, it is designed to meet the five goals of:

- Minimization of duplication of effort: by providing an external, broadly accessible information service
- Wide and easy availability: through an integrated lookup and delivery service
- Minimization of usage of resources: by providing the opportunities to reason about the cost-benefit tradeoffs in terms of resource usage
- Customary information transformation: by providing a systems supported point at which this can occur under local control. A library of such transformation functions might serve as a valuable additional tool to vet and make them more widely available.
- Definition and enforcement of access control policies: by effective use of the combination of Shibboleth, ABAC and a policy language and library.

We recognize that there remain a number of challenging issues in this space, including for each subsystem how to organize the nodes in it in such a way that the nodes are findable by their peers and that any criteria by which selection of one or another can be shared across them, how to do things like the cost-benefit analysis of use of the various kinds of resources, the design of the id space for information objects, and a number of others. These are all problems that are approachable with fairly standard engineering approaches, but all would need to be addressed in the context of the architecture and requirements. Because of the early stages of this work, it is not worthwhile to put effort into designing them until and unless there is agreement on moving forward with the system as a whole.

At the highest level, this document is intended to demonstrate the feasibility of an information plane accessible across the GENI architectural layers and across the federation, in order to support the analysis and diagnosis of unexpected phenomena in the GENI context that may cross or be exposed across those boundaries. Without a common substrate and the capability of sharing and exposing information across those boundaries, addressing issues that either are exhibited in one layer although attributable in another, or that actually cross those boundaries, will be inaccessible.

References to Non-GENI work

[Oasis] Oasis, **RelaxNG,** http://www.oasis-open.org/committees/relax-ng/ and http://relaxng.org/ viewed Aug. 25, 2011.

[RFH+01] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, *A Scalable Content-Addressable Network,* **Proc. SIGCOMM**, Aug. 2001, San Diego, CA, USA.

[SMK+01] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, H. Balakrishnan, *Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications*, **Proc. SIGCOMM**, Aug. 2001, San Diego, CA, USA.