GMOC Topology-Entity Data Exchange Format Specification

Version 0.1

Camilo Viecco – GMOC Senior Programmer

Introduction

The GENI Meta-Operations Center (GMOC), located at Indiana University, will design and craft the protocols and processes needed to facilitate operational data exchange among GENI's federated infrastructure. The first is to provide a global view of GENI-wide resources and the second is to provide a mechanism for emergency shutdown of sections or GENI. GMOC should also be able to provide real time status of GENI elements and to provide historic views into these status and network configurations.

GMOC has written a minimum requirements document for data to be exchanged between aggregate managers and GMOC. The objective of this document is to provide the specification of how this data will be formated for exchange. This document does not specify the mechanisms and/or protocols for the actual transmission of the document.

Requirements

The use case for the data exchange scenario is that the aggregate manager will be able to generate from their internal representation of the GENI infrastructure. The data exchange interface should be able to collect all but the usage statistics as specified in the Strawman Document. This exchange format should provide a 'snapshot' of the network as known by each data aggregator and/or clearinghouse.

There are several topology description languages/formats available. However none has the concept of slice nor a simple way to create arbitrary circuit hierarchies (starting from a top-bottom) manner. Thus GMOC has created an ontology to describe topologies as defined by the internal format specification.

Assumptions

We will use the ontology as defined in the GMOC internal format specification. Further we make assumptions For the internal topology an entity information the assumptions have been made in two key areas: identifiers for the entities and the relationships between entities.

Identifiers, encodings, and field sizes

We assume that at slices and devices are uniquely identified in GENI-wide by a human readable name. Names character set can be any unicode representable character set, but they must be encoded using UTF-8. Names are limited in size to 128 octets (bytes).

Principals are uniquely identified GENI-wide by a primary email address. Principals given names and last names are limited to 40 bytes. The email address is limited to 128 bytes.

Locations are uniquely identified (in the scope of each exchange document) by a human readable name.The minimal specification for a location is either the tuple (city, state_province, mail_code, country) or by the tuple (longitude and latitude).

Organizations are uniquely identified (in the scope of each exchange document) by a human readably name. These names are limited to 60 bytes.

Device's interfaces can be uniquely identified within a device by a device specific unique name. It is assumed that this name binding will be remain unique for subsequent documents as long as there are no changes in configuration of either the interface or the device.

Data model

In our data model every network device is considered a device. Devices can have a single parent device. The graph of the parent-hood for devices is a forest (a set of trees) .Slivers are modeled as virtual devices, that is a device with a parent device. Slices can be associated with both slivers and full devices. Each device can be associated with one sliver at most, this the graph of the relationship of slivers and devices is another forest.

A circuit is any network connection, between two or more devices. Circuits refer to any layer in the network stack and can be connected to any interface. Circuits can be build a multiplicity of other circuits. A circuit can be part of multiple circuits. The graph of circuit relationships is a disjoint set of directed acyclic graphs.

Data format

The date exchange format is defined using the relax-NG compact syntax as follows:

```
datatypes xsd = "http://www.w3.org/2001/XMLSchema-datatypes"
grammar {

start = element geni_aggregate {geni_aggregate-content}

geni_aggregate-content =
  attribute name {text},
  attribute public_key{text}?,
  element location {location-content}+,
  element contact  {contact-content}+,
  element organization {organization-content}+,
  element point_of_presence {pop-content}+,
  element device {device-content}+,
  element slice {slice-content}*,
  element net_topology {net_topology-content}

  location-content =
     attribute name {text},
     ( element address {address-content} |
       element geo_location {geo_location-content} |
       ( element address {address-content} ,
```

```
        element geo_location {geo_location-content} ) )

  address-content =
     attribute address {text}?,
     attribute city    {text},
     attribute province {text},
     attribute country  {text}

  geo_location-content =
     attribute latitude {xsd:double},
     attribute longitude {xsd:double}

  contact-content=
     attribute email_address {text},
     attribute last_name {text},
     attribute given_names {text},
     attribute phone {text}?,
     attribute organization_name {text}?

  organization-content=
     attribute name{text},
     element   primary_contact_email {text},
     element   location_name {text},
     element   parent_organization_name {text}?,
     element   url {text}?

  pop-content =
      attribute name{text},
      attribute location_name{text},
      element  operator_org_name{text}?,
      element  admin_org_name{text}?

  administrative_state-content =
      attribute state {"Planning"|"Provisioning"|"Available"|"NormalOperation"|
"Maintenance" |"Unknown" | "Decomissioned"}

  operational_state-content =
      attribute state {"Up" | "Degraded" | "Down" | "Unknown"}

  device-content =
     attribute name {text},
     element  device_location {device_location-content},
     element  operator_org_name {text},
     element  admin_org_name {text}?,
     element  device_type {text},
     element  sw_version {text}?,
     element  hw_version {text}?,
     element  operational_state {operational_state-content}?,
     element  administrative_state {administrative_state-content}?,
     element  interface {interface-content}*

  device_location-content =
      element pop_name {text} |
      element parent_device_name {text}
```

```
  interface-content =
      attribute name {text},
      element contracted_bw {xsd:double}?,
      element max_bps       {xsd:double}?,
      element administrative_state {administrative_state-content}?,
      element net_addr {net_addr-content}*

  net_addr-content =
      element net_addr_type {text},
      element addr{text},
      element netmask{text}

  slice-content =
      attribute name {text},
      element   operator_org_name {text},
      element   primary_contact_email {text},
      element   device_names {text}+

  net_topology-content =
      element  network {network-content}+,
      element  circuit {circuit-content}+,
      element  circuit_hierarchy {circuit_hierarchy-content}*

  network-content =
      attribute name {text},
      element   operator_org_name {text}?,
      element   admin_org_name {text}?

  circuit-content =
      attribute name {text},
      attribute circuit_type {text},
      element   channel {xsd:integer}?,
      element   reserved_bw {xsd:integer}?,
      element   vlan {xsd:integer}?,
      element   circuit_endpoint {circuit_endpoint-content}*

  circuit_endpoint-content =
      attribute   device_name {text},
      attribute   interface_name {text}

  circuit_hierarchy-content =
      element   upper_circuit_name {text},
      element   lower_circuit_name {text}
 }
```

Semantic Validation

The relax-ng schema validates the data from a document perspective. However any file must also pass the following semantic validation rules.

1. Every referenced email address must be part of a defined contact.

2. Every location_name must reference a defined location.
3. Every referenced organization name must be part of a defined organization.
4. Every referenced pop name must reference a defined pop.
5. Every referenced device name must reference a defined device.
6. Every referenced interface must reference a defined interface.

If a document is valid and pass the semantic validation rules then applications should be able to parse the document.

References

1. GMOC Discussion Strawman – Operational Dataset. January 2009.http://groups.geni.net/geni/attachment/wiki/GENIMetaOps/operational_dataset_v31.pdf.
2. Relax NG Website. http://www.relaxng.org
3. Internal Format specification.