GENI Multi-Technology and Multi-Point Layer-2 Stitching Architecture and Design

1 Introduction

This document describes an architecture design to support two advanced features: multi-technology stitching and layer-2 multi-point stitching. This design is an extension to the GENI Network Stitching architecture design that was developed in Spirals 3 and 4. Purpose of these advanced features is to support general stitching requests and cover all the current stitching use cases in the GENI community.

Previous GENI Network Stitching architecture design were focused on Point-to-Point (P2P) VLAN stitching. The key feature is to allow client to request a link that consists of two interfaces belonging to different aggregates. Stitching Computation Service (SCS) will expand that into a multi-aggregate stitching path and represent it in XML in the stitching extension element. SCS handles network connectivity, VLAN availability and VLAN translation constraints. It also generates workflow data that tell how the aggregates and hops depend on each other. This helps client construct dynamic workflow to instantiate the stitching path.

Emerging use cases from GENI community require connectivity technologies beyond VLAN. Top of the technology list are GRE and MPLS tunnels, VXLAN and OpenFlow. This document will address stitching of these technologies under their specific use cases.

Under the previous GENI Network Stitching architecture, we have already been stitching multiple P2P VLAN paths in a single request. Emerging use cases, however, require multi-point (MP) layer-2 stitching, which means to create layer-2 broadcast bridge for interfaces from more than two aggregates. In more general cases, we need to handle multiple P2P multi-technology links and multiple MP layer-2 links in a single request. This requires extending the SCS functions.

2 Multi-Technology Multi-Layer Stitching

Experimenters in the GENI community are exploring many technologies based on their research agenda. This design will not enumerate all technologies. Instead, we extract the most desirable ones, examine them in use cases and find the common solutions. The enhancements discussed in the following sections are based on additions to the current GENI Stitching Schema version 2, which is available via this URL:

- www.geni.net/resources/rspec/ext/stitch/2/stitch-schema.xsd

2.1 Stitching with Layer-3 and Layer-2 Tunnels

Layer-3 and layer-2 tunnels are the most desired alternatives to direct VLAN stitching when experimenters build "sliced" networks across multiple aggregates, especially where only IP connectivity is available. GRE, L2TP and MPLS are common protocols to carry the tunnels across an IP network.

Use GRE as an example. The layer-3 GRE tunnel creates an IP link between source and remote points. Layer-3 traffic will be routed across the tunnel just as on a single-hop IP link. A typical use case is to use the tunnel to connect virtual nodes at layer-3 and

distribute slice specific (often private) IP routes within the experiment scope. This use case is shown at top of Figure 1.

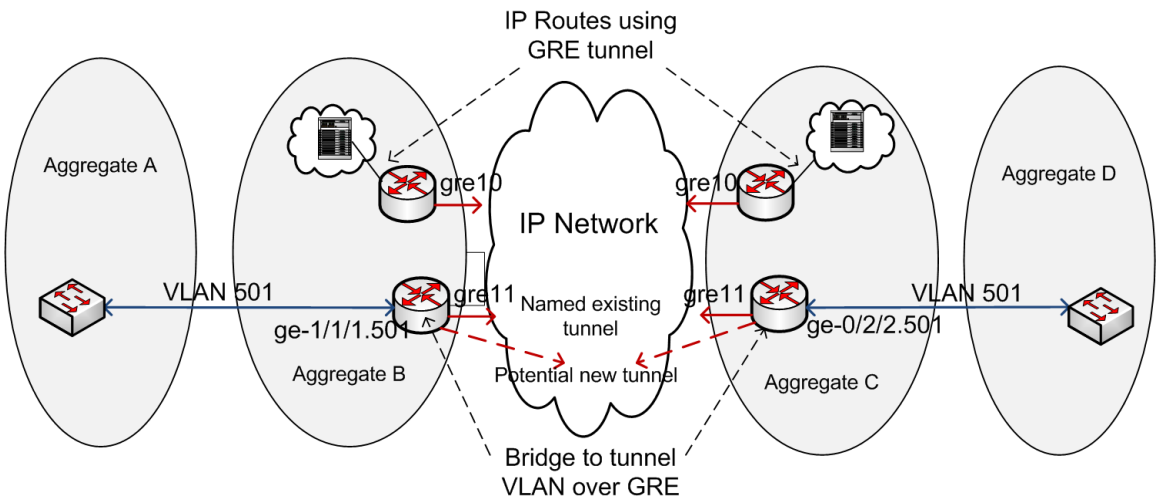

Figure 1: Stitching with Layer-3 and Layer-2 tunnels.

To advertise a layer-3 stitching link that is capable of GRE or other form of tunneling, the following technology specific information in Switching Capability Descriptor (SCD) need to be added under *link* / *SwitchingCapabilityDescriptors* / *switchingCapabilitySpecificInfo* element.

Schema for Psc_Tunnel technology specific information under SCD:

```
<xs:complexType name="SwitchingCapabilitySpecificInfo_Psc_Tunnel">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="capability" type="xs:string"/>
    <xs:element minOccurs="1" maxOccurs="1" name="tunnelType" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="interfaceMTU" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="localIPAddress" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="tunnelName" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

tunnelType could be “gre”, “l2tp”, “mpls” or “l2tpv3”. *localIPaddress* is the routable IP that the tunnel terminates on at the local node. *tunnelName* refers to an already existing tunnel. A tunneling stitching link can advertise either *localIPaddress* or *tunnelName* or both. When only *localIPaddress* shows up, a new tunnel has to be created. When only *tunnelName* shows up, the traffic must the existing tunnel. With both, we can use the existing tunnel if it connects to the designated remote or create a new one.

Layer-2 tunneling is a more common use case. It tunnels layer-2 traffic, typically VLANs, across underlying IP network such that both ends see each other as in direct layer-2 peering. This use case is shown at bottom of Figure 1.

The layer-2 tunneling over IP is instantiated by creating a tunnel "bridge" function and adding both layer-3 (IP) interface and layer-2 (VLAN) interface in the bridge. To advertise this capability, we need to have both *L2SC* and *PSC_Tunnel* SCDs under the same link. An example is below.

Example advertisement for VLAN+GRE tunnel stitching link:

```
<link id="urn:publicid:IDN+example.org+interface+router1:ge-1/1/1">
<!--skip some parameters-->

  <switchingCapabilityDescriptor>
    <switchingcapType>psc</switchingcapType>
    <encodingType>packet</encodingType>
    <switchingCapabilitySpecificInfo>
      <switchingCapabilitySpecificInfo_Psc_Tunnel>
        <tunnelType>gre</tunnelType>
        <localIPAddress>206.196.1.1</localIPAddress>
        <tunnelName>gre10</tunneName>
      </switchingCapabilitySpecificInfo_Psc_Tunnel>
    </switchingCapabilitySpecificInfo>
  </switchingCapabilityDescriptor>

  <switchingCapabilityDescriptor>
    <switchingcapType>l2sc</switchingcapType>
    <encodingType>ethernet</encodingType>
    <switchingCapabilitySpecificInfo>
      <switchingCapabilitySpecificInfo_L2sc>
        <vlanRangeAvailability>2-1000</vlanRangeAvailability>
        <vlanTranslation>false</vlanTranslation>
      </switchingCapabilitySpecificInfo_L2sc>
    </switchingCapabilitySpecificInfo>
  </switchingCapabilityDescriptor>
</link>
```

The first SCD for the *PSC_Tunnel* type represents the lower tunnel over IP layer. The second SCD of *L2SC* type represents the upper Ethernet layer where a range of VLANs that could be put through the underlying tunnel. The same representation is also used in the stitching extension for Request and Manifest RSpecs. Use VLAN+GRE as an example. Presence of the *tunnelName* element indicates that a tunnel is already instantiated and available. Aggregate Manager can use this information to determine whether to use an existing GRE tunnel or create a new one. When creating a new one, AM can use the local IP address as source and the one in next hop as remote. Also it knows which VLAN should be created and bridged to the GRE tunnel.

1.1. VXLAN and NVGRE Stitching

In some use cases, people want to explore emerging layer-2 technologies like VXLAN and NVGRE etc. as alternative to VLAN, especially for cloud networking experiments. VXLAN uses 24-bit Virtual Network Identifier (VNI) while NVGRE uses 24-bit Tenant Network Identifier (TNI). Both allow to scale network segmentation better than the 12-bit VLAN tag. Both technologies tunnel over IP networks, especially when connecting to

outside of its local area or domain. From path computation and instantiation perspective, they can be treated the same as VLAN. To deal with the minor difference from VLAN in segmentation identifiers, we add the *switchingCapabilitySpecificInfo_L2sc_Generic* definition in place of the *switchingCapabilitySpecificInfo_L2sc* that was dedicated to VLAN type SCD in the Stitching Extension v2 schema.

Schema for L2sc_Tunnel technology specific information under SCD:

```
<xs:complexType name="SwitchingCapabilitySpecificInfo_L2sc_Generic">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="capability" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="interfaceMTU" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="labelRangeAvailability" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="suggestedLabelRange" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="labelTranslation" type="xs:boolean"/>
  </xs:sequence>
</xs:complexType>
```

Here *labelType* could be “vlan”, “vxlan” and “nvgre” etc. This means this definition also covers *switchingCapabilitySpecificInfo_Lsc*. For backward compatibility, we will still keep the latter in the new schema.

1.2. OpenFlow Stitching

People in the GENI community have been using OpenFlow in multi-aggregate slices. Previously we treat it as “payload” and transparent to stitching. We see two use cases that require us to represent OpenFlow explicitly in stitching computation and stitching instantiation. In the first case, a layer-2 stitching link carries a range of VLANs. Some of them are regular VLANs. The others carry OpenFlow traffic that needs to be associated with experimenter controllers. The AM needs to differentiate these two kinds of VLANs on such “hybrid” layer-2 links. The solution is to add a second SCD that has *switchingType=“openflow”* and *encoding=“vlan”*, in addition to the first SCD that has *switchingType=“l2sc”* and encoding=“*ethernet*”. The technology specific information of the *openflow*/*vlan* SCD will be either *switchingCapabilitySpecificInfo_Lsc* or *switchingCapabilitySpecificInfo_Lsc_Generic* just like in the *l2sc/ethernet* SCD. It should be noted that the VLAN (or label) ranges of the two SCDs must not overlap. An example definition is below.

Example advertisement for "hybrid" OpenFlow VLAN stitching link:

```
<link id="urn:publicid:IDN+example.org+interface+router2:ge-1/1/2">
<!--skip some parameters-->

  <switchingCapabilityDescriptor>
    <switchingcapType>l2sc</switchingcapType>
    <encodingType>ethernet</encodingType>
    <switchingCapabilitySpecificInfo>
      <switchingCapabilitySpecificInfo_L2sc>
        <vlanRangeAvailability>2-1000</vlanRangeAvailability>
        <vlanTranslation>false</vlanTranslation>
      </switchingCapabilitySpecificInfo_L2sc>
    </switchingCapabilitySpecificInfo>
  </switchingCapabilityDescriptor>

  <switchingCapabilityDescriptor>
    <switchingcapType>openflow</switchingcapType>
    <encodingType>vlan</encodingType>
    <switchingCapabilitySpecificInfo>
      <switchingCapabilitySpecificInfo_L2sc>
        <vlanRangeAvailability>1001-1100</vlanRangeAvailability>
      </switchingCapabilitySpecificInfo_L2sc>
    </switchingCapabilitySpecificInfo>
  </switchingCapabilityDescriptor>
</link>
```

In the other case that is expected for future use, OpenFlow is not stitched based on VLAN segmentation but in a more native way through flowspace slicing. For this purpose, we introduce a new SCD with *switchingType="openflow"* and encoding="*flowspace*". A new type *switchingCapabilitySpecificInfo_Flowspace* is defined below.

Schema for L2sc_Tunnel technology specific information under SCD:

```
 <xs:complexType name="SwitchingCapabilitySpecificInfo_Flowspace">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="capability" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="availableFlowspaceMatch" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="suggestedFlowspaceMatch" type="xs:string"/>
  </xs:sequence>
 </xs:complexType>
```

Each flowspace "match" string in *availableFlowspaceMatch* and *suggestedFlowspaceMatch* represents a flowspace in advertisement or a slice in request and manifest. Format of the "match" string will be defined in future.

1.3. Support for Arbitrary Layers

The above design for multi-technology multi-layer stitching has covered almost all use cases in current GENI practice. For cross-layer stitching, we need more than one SCD in defining a stitching link. We propose to have the following as a convention.

When two switchingCapabilityDescriptor elements are present under the same link, the first one represents the lower layer and the second one represents the upper layer. The implication is that an upper layer connection can be adapted / tunneled through a lower layer connection.

However, there could still be corner cases that operators and experimenters want to try more complex technologies that require more than two layers. Two such cases are illustrated in Figure 2. In the left stack, VLAN is tunneled over IP layer. Then OpenFlow is carried in VLAN segmentation. In the right stack, layer-2 is the lower layer that provides VLANs to carry both OpenFlow and IP traffic in a mix. Both stacks require three SCDs under the same link. However, it is impossible to infer an upper-to-lower layer relationship by implication. The solution is to add a so-called Interface Adjustment Capability Descriptor (IACD), a definition borrowed from GMPLS standard. IACD explicitly defines how one layer is related to the other through upper-to-lower layer adaptation. As a corner case without immediate need, we only consider IACD an architecture component and leave its definition to future work.

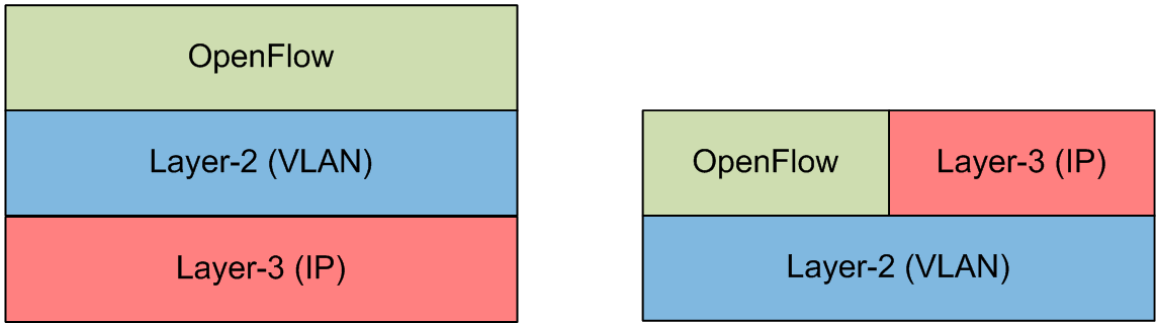

Figure 2: Illustration of complex stitching layers.

2. Multi-Point Layer-2 Stitching

2.1. Model and RSpec Schema

Figure 3 shows the model of Multi-Point Layer-2 (MP-L2) stitching over GENI infrastructures. In this design we model any MP-L2 stitching request as to build a <link> where all the <interface> elements in the link need to be bridged over a common layer-2 broadcast domain, which often means a single VLAN or connected VLANs that translate tags although other layer-2 technologies are also supported.

In this model, interfaces from the same aggregate are grouped as a single Terminal of the broadcast domain. The work by stitching is to connect all the Terminals in a Tree structure to form a broadcast domain. As the terminals are subset of global nodes, creating an optimal MP-L2 broadcast domain becomes a Steiner Minimum Tree (SMT) problem. The problem need to be solved under both VLAN (or other Layer-2 label) and bandwidth constraints. It has NP-Complete complexity and requires developing new heuristic algorithms.

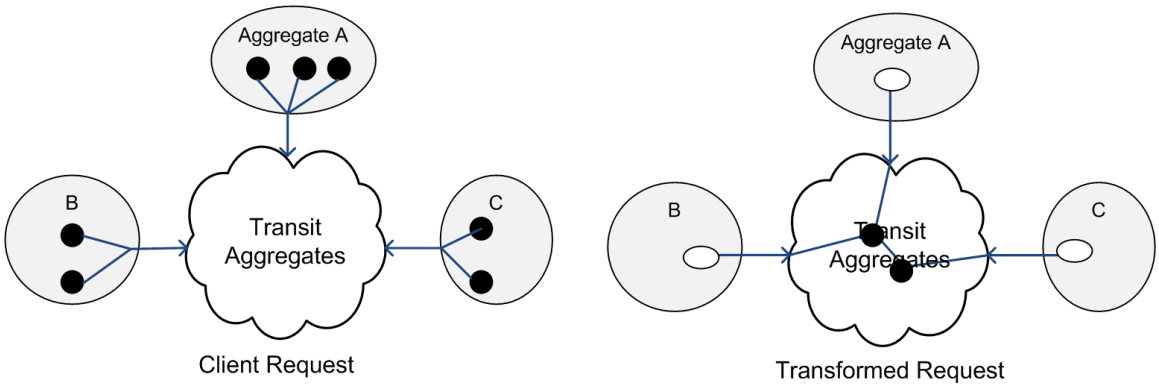

Figure 3: Illustration of MP-L2 stitching model.

The current GENI RSPEC v3 and Stitching Extension v2 schemas are able to describe arbitrary network topologies. To advertise MP-L2 bridging capability, we propose to utilize the existing <capabilities> elements to add a new capability:

<capability>mp-l2-bridging</capability>

This can be added to either under <aggregate> or <node> depending on the scope of advertisement for this capability.

To ask for an MP-L2 stitching path, a client needs no format change to Request RSpec but simply adds all the interfaces it wants to bridge to under the same *<link>* in the RSpec main body.

An expanded Request RSpec sent to AM needs to include a Stitching Extension that represents the MP-L2 bridging at certain path hops. So does the Manifest RSpec replied from the AM. The current Stitching Extension v2 schema is able to accommodate that using a stitching *<path>* that contains only *<link>* type *<hop>* elements. An example is shown in Figure 4, where a 3-way bridge needs to be created in Intenret2 AL2S aggregate that involves edge links on three different nodes: rtr.wash, rtr.newy and rtr.salt. <path> (a) and <path> (b) are two different but equivalent representation of the same MP-L2 stitching path.

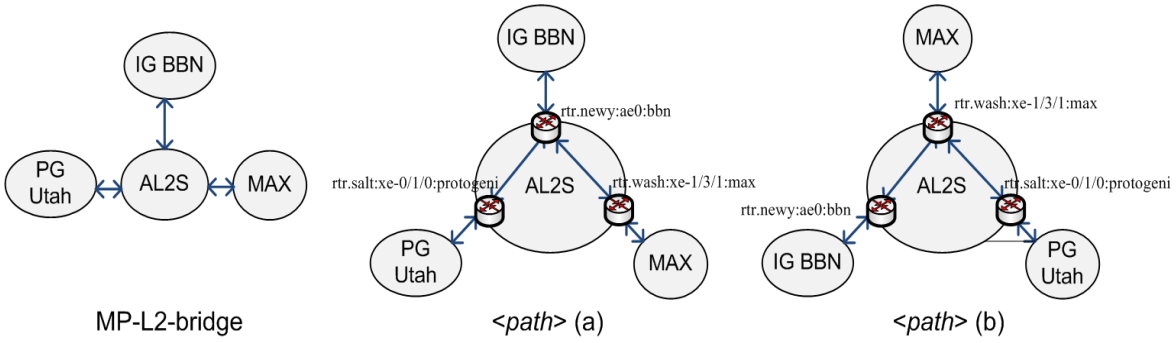

Figure 4: MP-L2 stitching path representation.

In the XML representation of <*path*> (a) in Figure 4, the *hop* "rtr.newy:ae0:bbn" has two *nextHop* elements referring to "rtr.salt:xe-0/1/0:protogeni" and "rtr.wash:xe-1/3/1:max" respectively. The AL2S AM can interpret this literally as "creating a VLAN bridge at rtr.newy and connect to both rtr.salt and rtr.wash" or roughly as "creating a VLAN bridge somewhere that connects these three interfaces at rtr.newy, rtr.salt and rtr.wash". The same applies to <*path*> (b). In the case AM interprets the <*path*> bridge roughly, <*path*> (a) and <*path*> (b) are exactly the same. In either case, there is no difference from the client's perspective.

2.2. MP-L2 Stitching Functions

In the augmented GENI Stitching Architecture, the following functions are required in support of the MP-L2 stitching.

- **SCS MP Path Computation** is required to extend the existing stitching computation service with algorithms to generate multi-point layer-2 bridge. New heuristic algorithms need to be developed to calculate an SMT to connect multiple terminal aggregates under layer-2 constraints and return expanded stitching extension with properly formatted <*path*> elements.

- **SCS and Client Workflow** for an MP-L2 path needs to be generated with upgraded stitching workflow heuristic. Both SCS and Client must understand that the workflow will not only work for P2P path but also for MP path.

- **Terminal Bridging:** MP-L2 stitching creates layer-2 broadcast domain that connects multiple terminal aggregates. Interfaces from same aggregate are all bridged to this broadcast domain by internal operation. We need to make sure individual Aggregate Managers have the function to execute the terminal bridging as transparent to the stitching workflow.

- **Transit Bridging:** Transit aggregate is where the MP-L2 bridging is created from the inter-aggregate stitching point of view. This requires a transit aggregate to understand the bridging representation in MP-L2 stitching path and execute the bridging instantiation. For example, AL2S AM needs to correctly interprets and instantiates the 3-way bridging in either form represented in Figure 4. There could be case that an aggregate is both terminal and transit in the same stitching path. In this case, it must be able to handle both functions independently.

3 General Stitching Requests and SCS Extension

A general stitching request will consists of nodes and links from arbitrary aggregates. There will be multiple links, each consisting of two or more interfaces that are defined under different nodes. After consolidating the interfaces from same aggregates, we get links that have two or more Terminals, which are the input for inter-aggregate stitching computation. A link with two Terminals is expanded by SCS into a P2P stitching path while a link with three or more Terminals is expanded by SCS into an MP stitching path.

In this design, we generalize it to the point that we can have multiple P2P multi-technology links and multiple MP-L2 links in a single request.

To handle the general stitching requests, SCS need to be extended as follows.

- SCS will have a "master" algorithm that dispatches the request to multiple computation workers. One worker will perform multi-P2P path computation. The others will each perform a single MP-L2 path computation. The "master" algorithm will combine the results into a single reply.

- SCS will compute workflow data for each individual P2P and MP-L2 paths. An extended function will merge the multi-path workflow data into a consolidated one so that client will have a single set of workflow data.

4 Advanced Stitching Architecture Considerations

This section presents an initial discussion of some possible future stitching architecture feature sets and deployment options. This is high level concept and summary information intended to provide some framework for further discussion and concept development. Depending on the results of these discussions, additional detail regarding these items may be included in future updates to this document.

4.1 GENI Stitching Service

Current stitching operations are based on user tools, such as OMNI, interacting with all the GENI Aggregates in a slice. A typical use case is that some of the Aggregates are "network resource only" aggregates such as ION, and eventually AL2S. It is possible to imagine a "GENI Stitching Service" that the existing user tools could utilize to take care of instantiating services on "network resource only" aggregates. Some key capabilities and features of such a "GENI Stitching Service" may be:

- Standalone GENI Stitching Service
- Some GENI Aggregates will be identified as "network-only" aggregates using the "capabilities" element in the stitching schema
- User tools can use this capability information to hand the entire (expanded) Request RSpec to the GENI Stitching Service
- GENI Stitching Service would provision (or reserve) only those aggregates labeled "network-only" in the multi-aggregate Request RSpec
- This would require the GENI Stitching Service to have a "Speaks For" capability
- An updated Manifest RSpec would be handed back to the User Tool
- The User Tool could use this information to update its Request RSpec and continue its interaction with the other (non network-only) aggregates
- The User Tool could also reverse the order and ask the GENI Stitching Service to take care of the "network-only" aggregates after the other aggregate resources were provisioned (or reserved)

- This type of GENI Stitching Service is probably more valuable in combination with the GENI AM APIv3 support when resource reservation (as opposed to just provision) capability is possible.

4.2 GENI Stitching Exchange Points

Current GENI Stitching operations typically involve crossing multiple campuses, regional, and/or wide area networks to establish layer2 connections between aggregates. At times there may be network resource contention, which can prevent specific stitching operations from being established. One method to increase the resources available for GENI Stitching may be to deploy one or more GENI Stitching Exchange Point(s). Some key capabilities and features of such a "GENI Stitching Exchange Point" may be:

- A single well-resourced Ethernet or Openflow switch under the control of a GENI AM
- This may be deployed within an existing exchange point, wide area network, regional network, or campus.
- Since a single well-resourced Ethernet or Openflow switch will have a non-blocking backplane, with all Layer2 and OpenFlow flowspace available, and is dedicated to GENI Stitching, the opportunity to reduce resource contention will be large
- The idea is that multi-aggregate GENI Slices can have topologies which all converge on a common GENI Stitching Exchange Point.
- The method by which a specific Aggregate gets connected to a GENI Stitching Exchange Point can vary. This may include static provisioning, dynamic provisioning thru other infrastructures, or some combination.
- It is imagined that GENI Stitching Exchange Point(s) would be an augmentation to the existing dynamic network infrastructures such as Internet2 AL2S/ION, and other similar capabilities.

From an architecture and implementation perspective a GENI Stitching Exchange Point is no different then a "network-only" aggregate. An Ethernet or Openflow switch under the control of a GENI Aggregate Manager can be a GENI Stitching Exchange Point. This is really a deployment consideration, where the idea is to place well-resourced GENI Stitching Exchange Points at strategic locations, to provide greater network resource availability. A regional network may find it advantageous to deploy a GENI Stitching Exchange Point somewhere within its infrastructure as a way to isolate the dynamic capabilities of GENI from the more static configurations of their production operations.

Another item to note is that from an architecture and implementation perspective, Internet2 AL2S with a GENI AM looks just like a large GENI Stitching Exchange Point. Or perhaps at GENI Stitching Exchange Point looks like a small AL2S with GENI AM. The key point is that the purpose of deploying something like a single switch in the form of a GENI Stitching Exchange Point is to expand resources available for stitching operations.