SDN Multi-Domain

Architecture Thoughts

Global Lambda Integrated Facility (GLIF)
Technical Working Group Meeting #19
Honolulu, Hawaii
January 16, 2013

Tom Lehman
Mid-Atlantic Crossroads
University of Maryland

Perspective/Reference Points/Assumptions

- SDN is an abstract, undefined term which basically just means that we can do things dynamically in the network via an API
- OpenFlow is one type of API and/or control mechanism which can be part of an SDN domain
- There will be other APIs and control mechanisms which will be part of SDN
- We can design a Multi-Domain SDN solution by considering similar things to what we need to consider for any Multi-Domain Service systems
- Administrative domain demarcations will remain at base level - recursion and slicing will be used to present users with something different

Perspective/Reference Points/Assumptions

- **From a user perspective we will be provisioning "services", which need to be defined.**
  - OpenFlow Service Example: user uses and API to get a FlowTable rule inserted in their favorite OpenFlow Network which gives them some vlan and mac space, and then they fire up their own OpenFlow Controller to create slices
  - MultiPoint Topololgy Service Example: user gets a multi-point topology over which they run their own applications.
    - internal mechanism may be via OpenFlow, or may be via other mechanisms

Key Architectural Considerations

- **SDN Multi-Domain Service may be more accurate term then SDN Inter-Domain**
- **OpenFlow is both a control plane and a data plane**
- **The data plane is unique as compared to other data planes we have dealt with:**
  - flowspaces can cover alot of areas and unique combinations
- **At the OpenFlow control plane level, we also have options:**
  - let users run their own openflow controller and talk to network flowvisor
  - just provide user services thru an API, with OpenFlow being the internal mechanism to get things done

Key Architectural Items

- **User Services Definition**
- **Controller Service API**
  - Tree vs Chain messaging
  - Real-Time resource identification (multi-round negotiation protocols)
  - **schemas (syntax, semantics, use cases) for service and resource descriptions**
- **Topology Service**
  - Export/Distribution (realtime vs static)
  - **schemas (syntax, semantics, use cases) for resource descriptions**
- **Computation Service**
  - Resource/Path Computation
- **Common set of schemas for topology descriptions and service request/responses**
- **Authentication/Authorization Features**
  - needed for Service Requests and Topology Viewing

•These are the most important things that need to be done first
•This is the language for describing services and resources

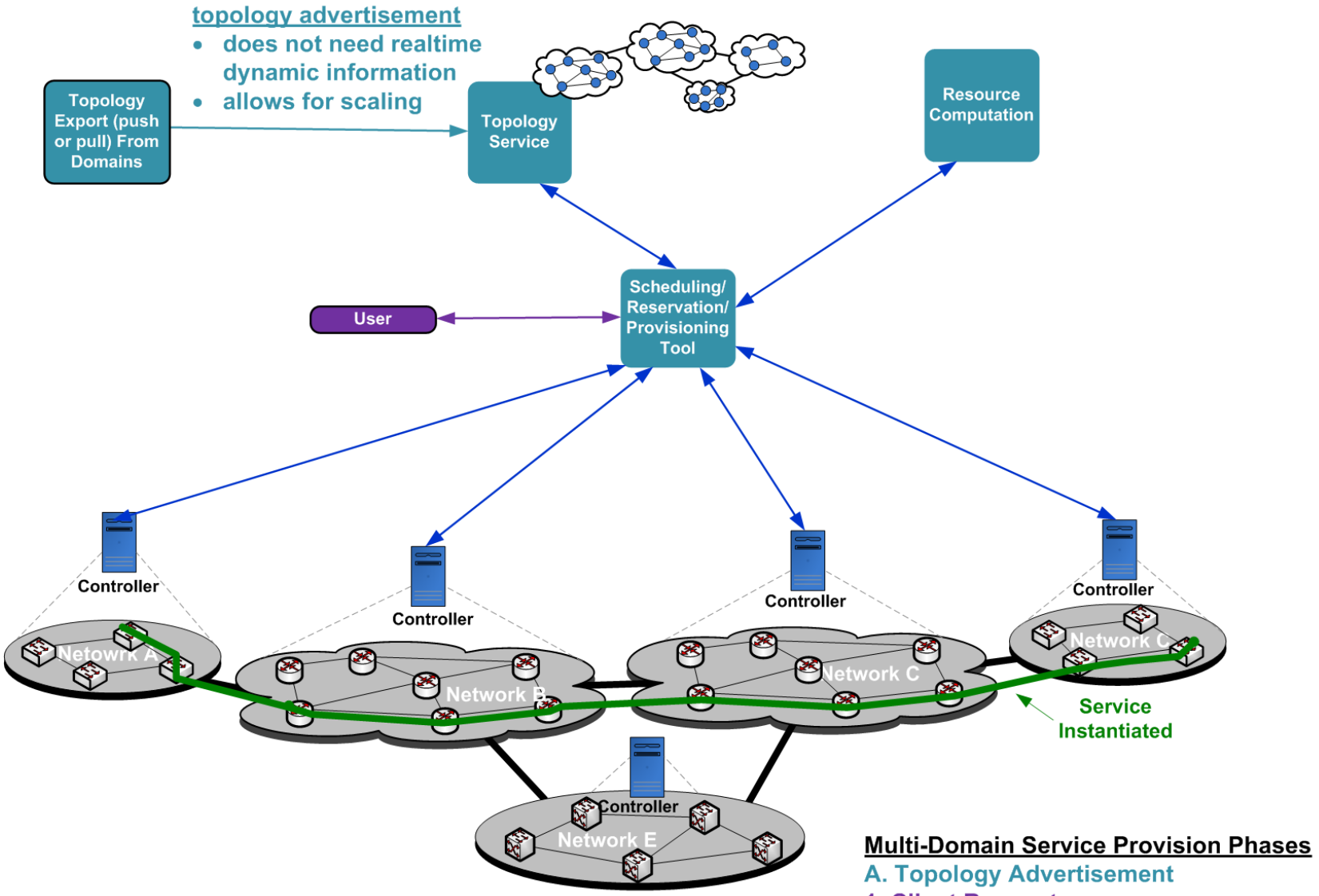
Key Architectural Items

- **There are multiple architectures and many protocols which can make this work**
- **All of the protocols and schemas in discussion today could be used as part of this Architecture solution/implementation**
- **But the architecture/system is more then just a protocol**
  - many details must be specified and implemented associated with all the architecture components
- **not sure who will design/implement/test/support the full system?**
  - it would be helpful if there was more coordination/development synergy across the various projects working on these issues

One Architectural Approach
(based on experience with OSCARS, DRAGON, GENI)

- Centralized at the Intra-Domain level for resource management and service provisioning
- Distributed at the Multi-Domain level for resource management and service provisioning
- External topology distribution systems must not require large/frequent dynamic data export (scalability and stability issues)
- Resource identification for real-time service provision can only be done by local domain systems
- Multi-domain service provision based on tree mode protocols which include real-time negotiation/multi-phase commit features

Multi-Domain SDN Architecture



Multi-Domain Service Provision Phases

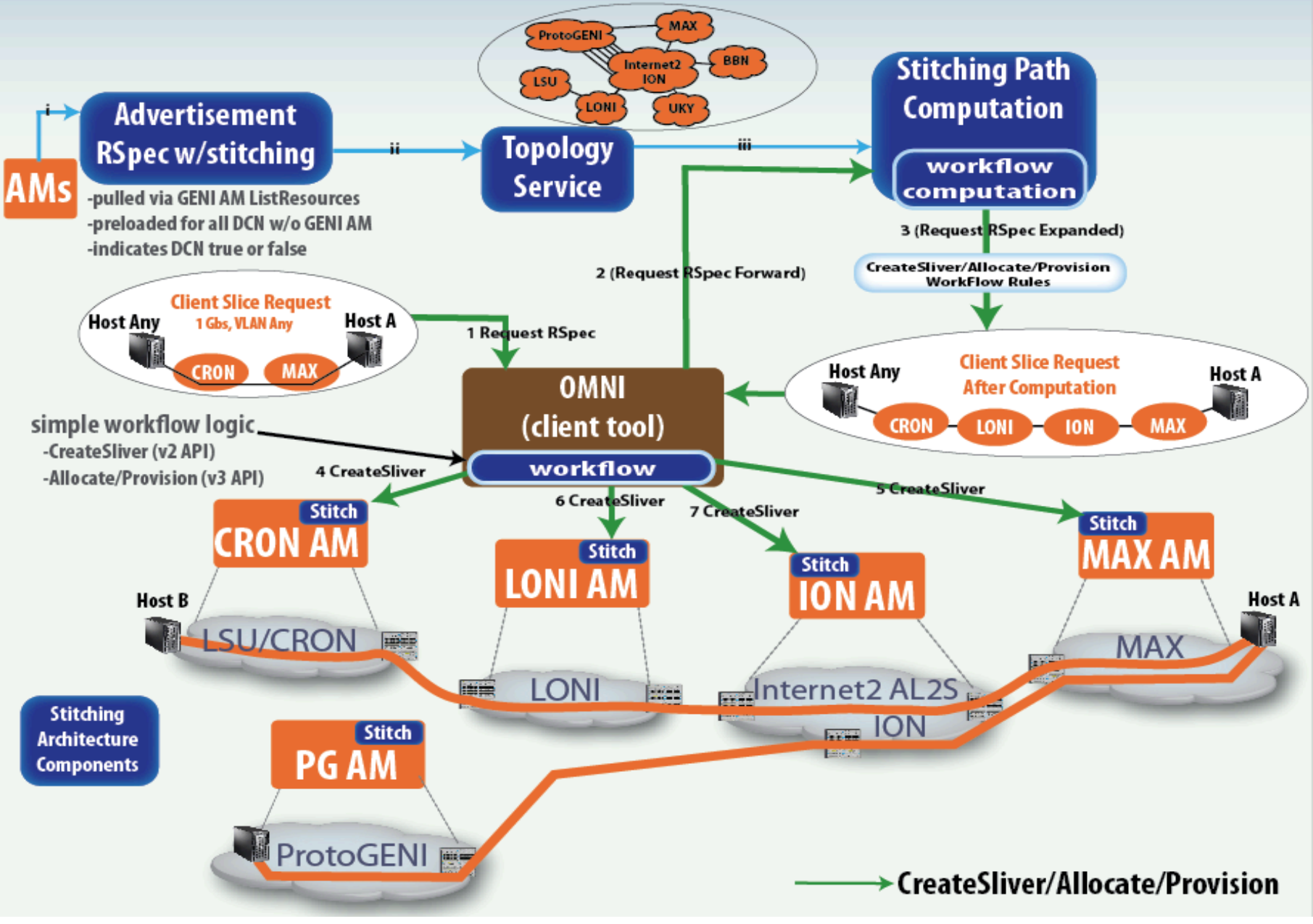
A. Topology Advertisement

1. Client Request

2. Resource Scheduling (Multi-Phase)

3. Service Instantiation

GENI Network Stitching Architecture



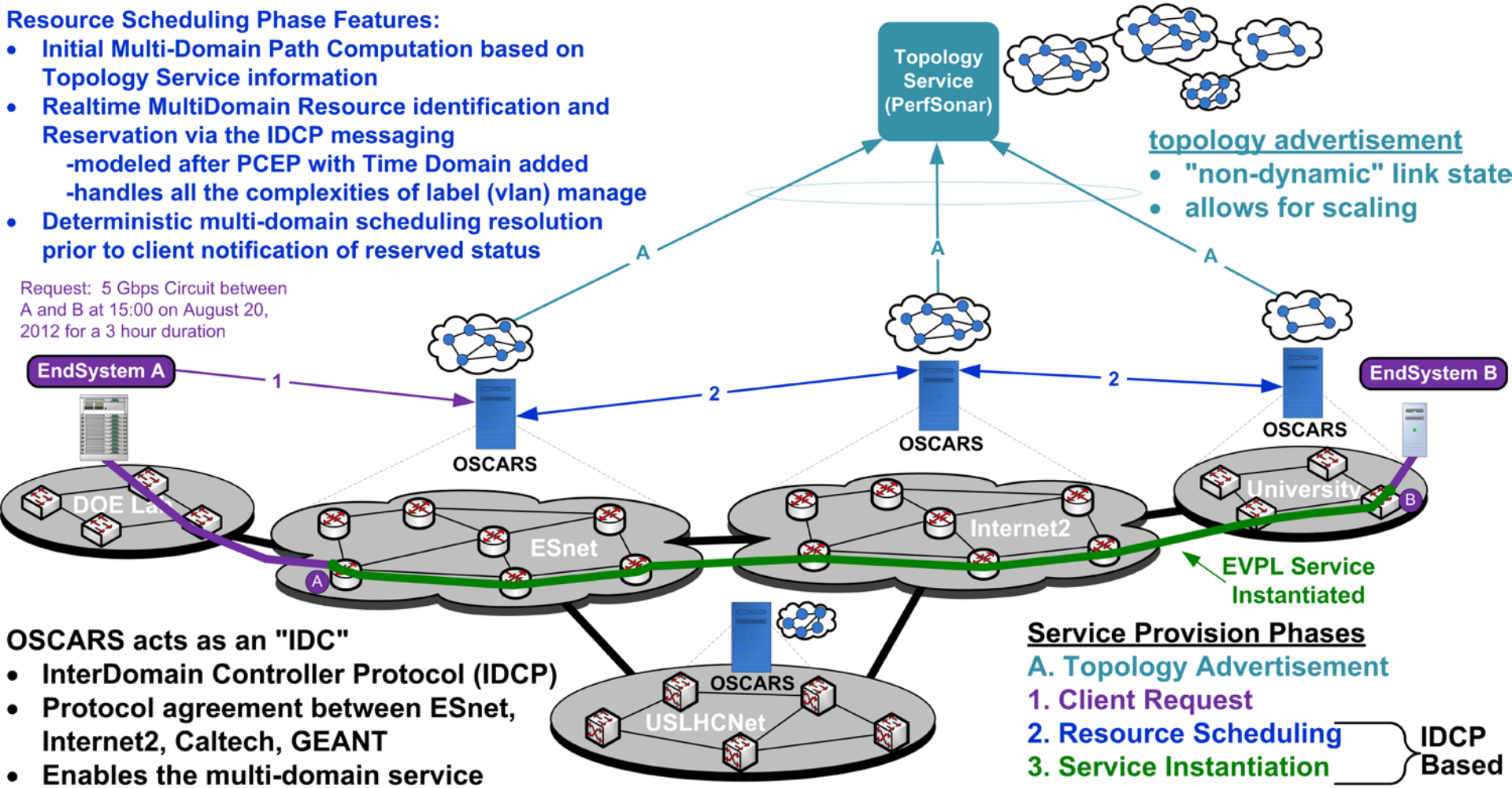
OSCARS/IDCP

The "service" is Ethernet Virtual Private Line (EVPL) with dedicated bandwidth

Different networks use different technologies to instantiate (MPLS, SONET, Native Ethernet, WDM)

Resource Scheduling Phase Features:

- **Initial Multi-Domain Path Computation based on Topology Service information**
- **Realtime MultiDomain Resource identification and Reservation via the IDCP messaging**
  - **-modeled after PCEP with Time Domain added**
  - **-handles all the complexities of label (vlan) manage**
- **Deterministic multi-domain scheduling resolution prior to client notification of reserved status**



OSCARS acts as an "IDC"

- **InterDomain Controller Protocol (IDCP)**
- **Protocol agreement between ESnet, Internet2, Caltech, GEANT**
- **Enables the multi-domain service**

Service Provision Phases

A. Topology Advertisement
1. Client Request
2. Resource Scheduling
3. Service Instantiation

IDCP Based