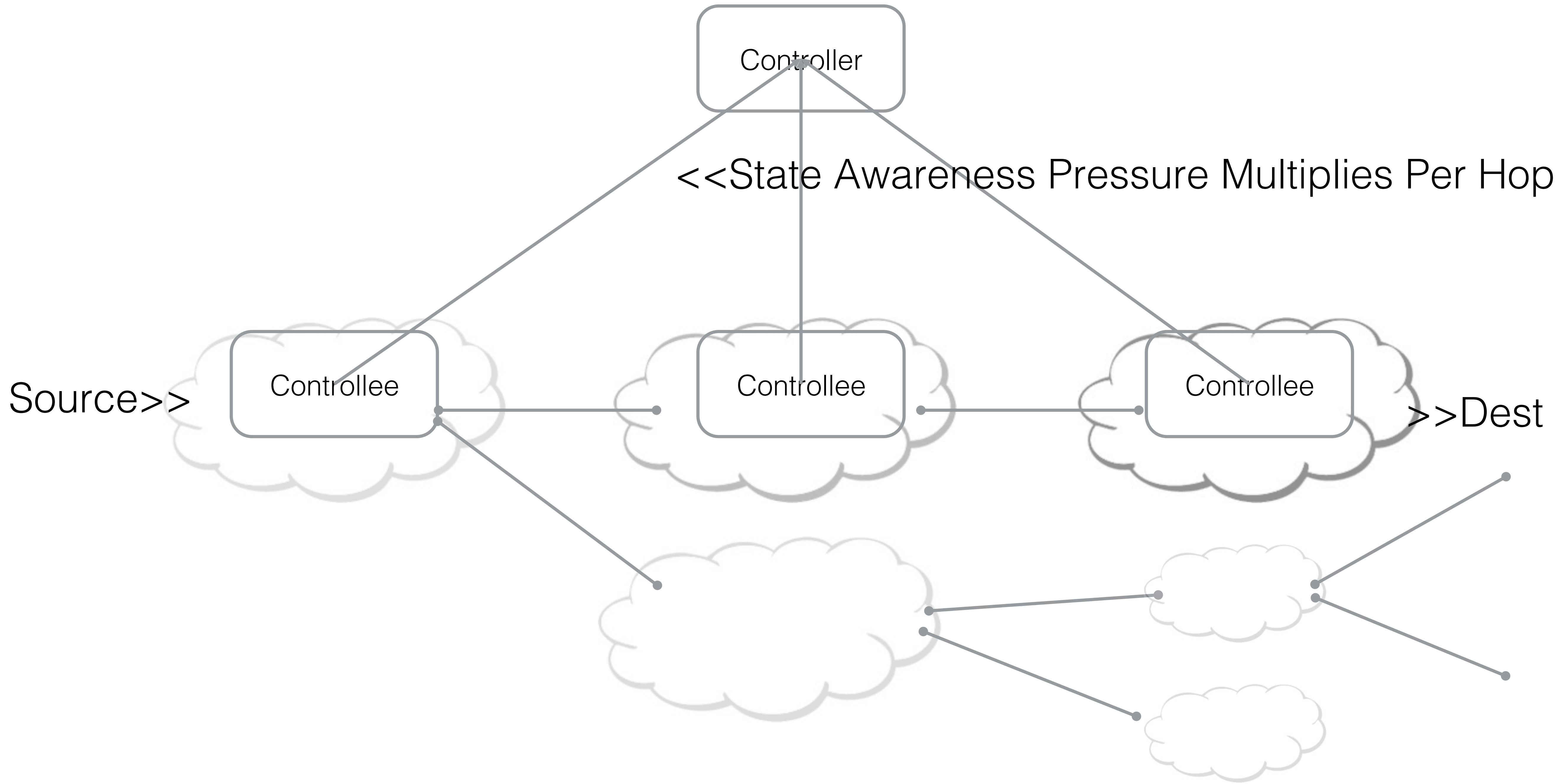
Consolidating Network Virtualization Indirections

Sharon Barkai IETF91 SDNRG

- Network virtualization structure and dynamic programmability progressed beyond "Controller"
- On its own "Controller" is not a scale structure, it has exponential complexity and cap issues per net-hop
- The scaled structure is achieved using scoped virtualization indirection interfaces (NVI)

The SDN Scale Challenge



NVI:

A given well known direct interface is enhanced with a programable standard based i direction construct seamless to **both** legacy ends

NVI1 ID-Location

- Source <=> Network <=> Dest
- NVE <=> Underlay <=> NVE
- Allows endpoints to show-up anywhere, no prefix "zip-code" zoning limitation, datacenter for hosting, nfv..

NVI2 Sub-Service

- Source <=> Network <=> Dest
- NVE <=> Underlay <=> NVE
- SFC <SFFs & Functions> SFC
- Allows seamless chaining of middle-boxes between source and dest, seamless to source, dest, functions

NVI3 Class-Instance

- Source <=> Network <=> Dest

⇅

- NVE <=> Underlay <=> NVE

⇅

- SFC <SFFs & VIPs> SFC

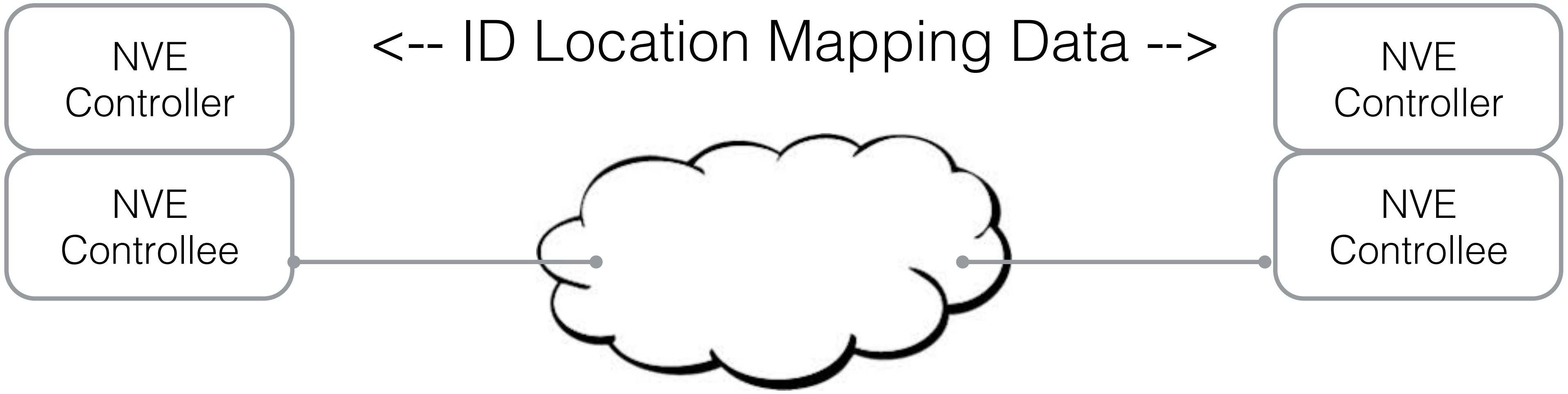
⇅

- Elastically Allocated Actual IPs

- These (recursive) indirection constructs are not a "free-format" or anything goes flow programming, But they do help scale SDN per each one NVE, SFF, ADC ..**How**?

- A controller-contrôlée pair can govern each such mechanism, this scales consistently, as long as the **global** context can be shared between the pairs

How Does NVI Scale SDN



Source>>

NVE Controller

NVE Controllee

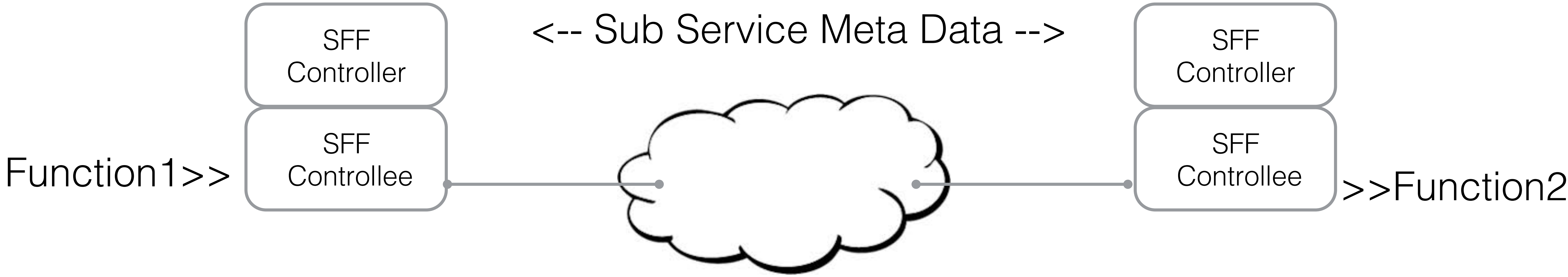
<-- ID Location Mapping Data -->

NVE Controller

NVE Controllee

>>Dest

How Does NVI Scale SDN



<-- Sub Service Meta Data -->

SFF Controller

SFF Controllee

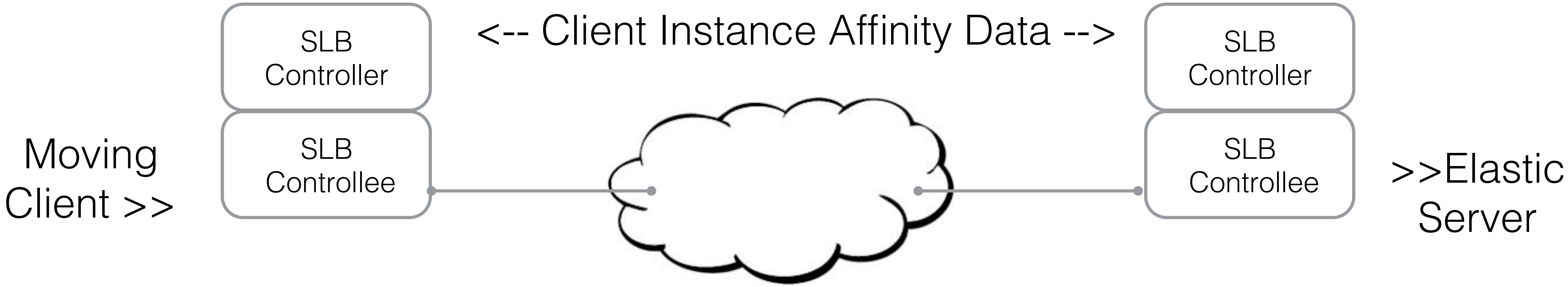
Function1>>

SFF Controller

SFF Controllee

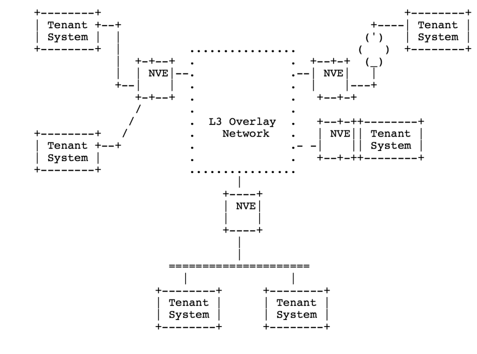
>>Function2

How Does NVI Scale SDN



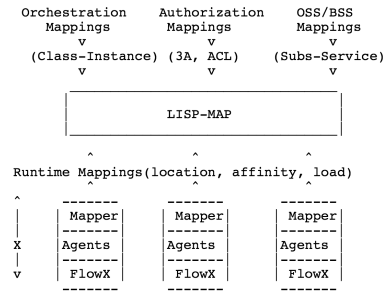
However

- Implementing and deploying each of these NVI RFCs separately will result in extreme inefficiencies
- Though solutions will be dynamic and utilized, the multi-hop high latency factor may be critical

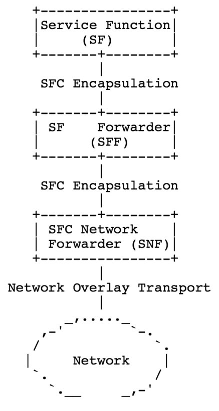


The dotted line indicates a network connection (i.e., IP).

Figure 1: NVO3 Generic Reference Model



Identity-Location Overlay



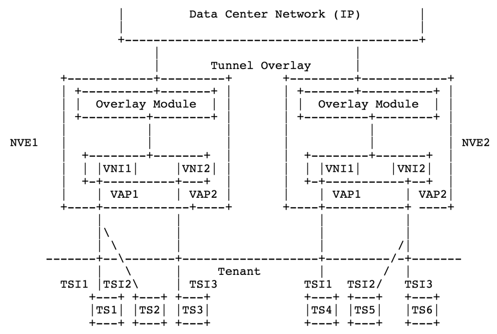


Figure 2: NVE Reference Model

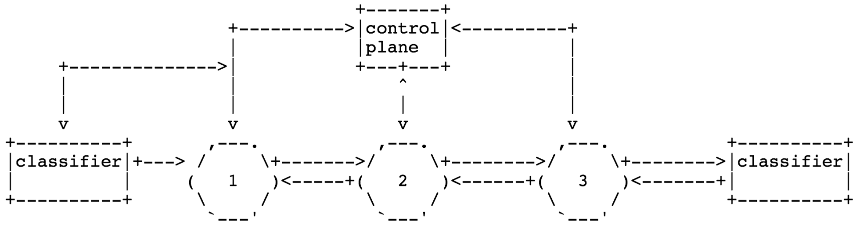
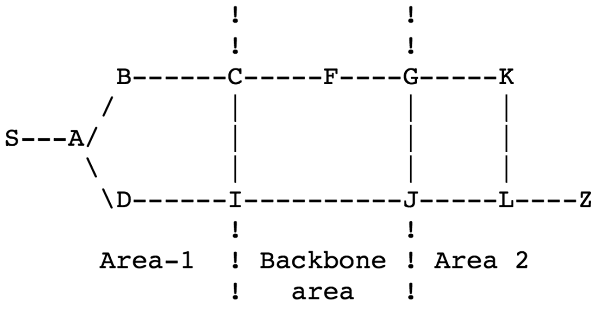
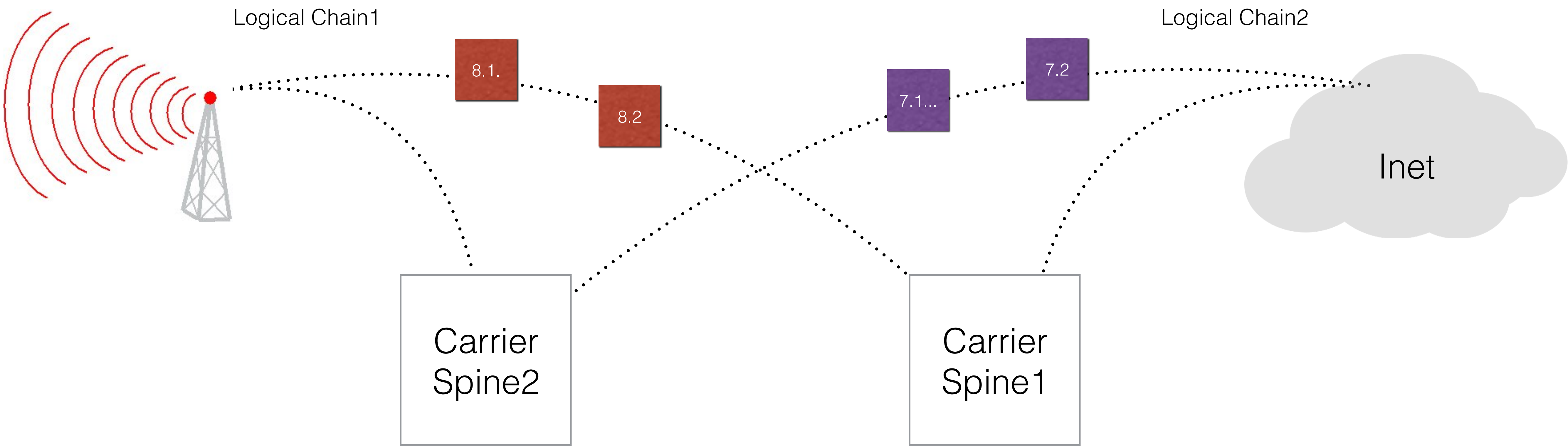


Figure 3: Service Function Chaining Architecture

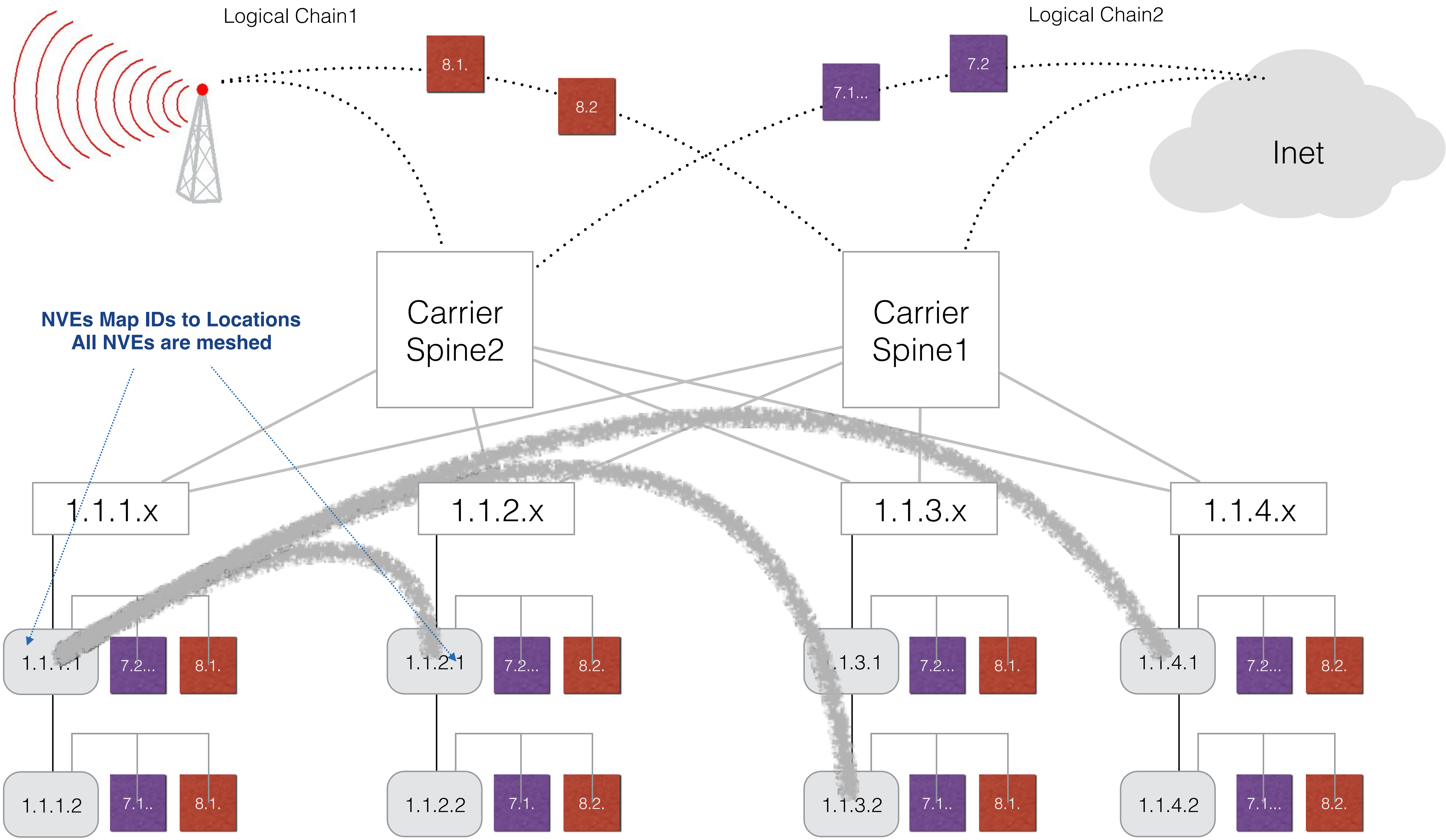


Example: Programming the network for Carrier-Services between the Access and the next peering provider network

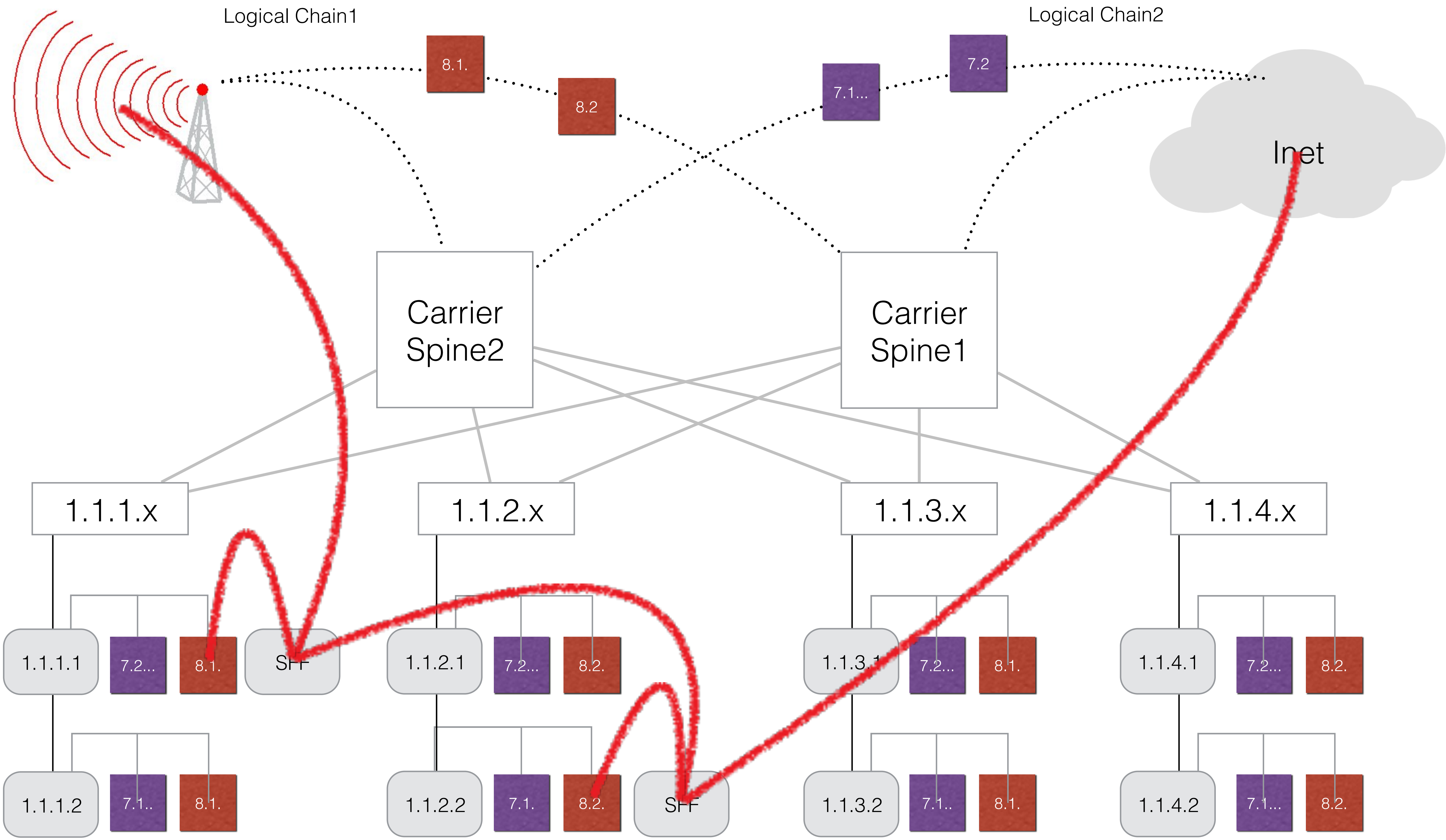


- Mobility Management - stationary context for moving phones
- Subscriber Management - Quality/class of service queuing, AAA
- TCP Optimization - jitter buffers and window scaling utilization
- Video caching and transcoding - steer and redirect video streams
- Filters and protections - firewalls, parental control, honey pots
- Analytics - records of TCP flows, flow quality, HTTP flow tracking
- Header enrichment - profiling users, revenue share, 800 data flows
- ...

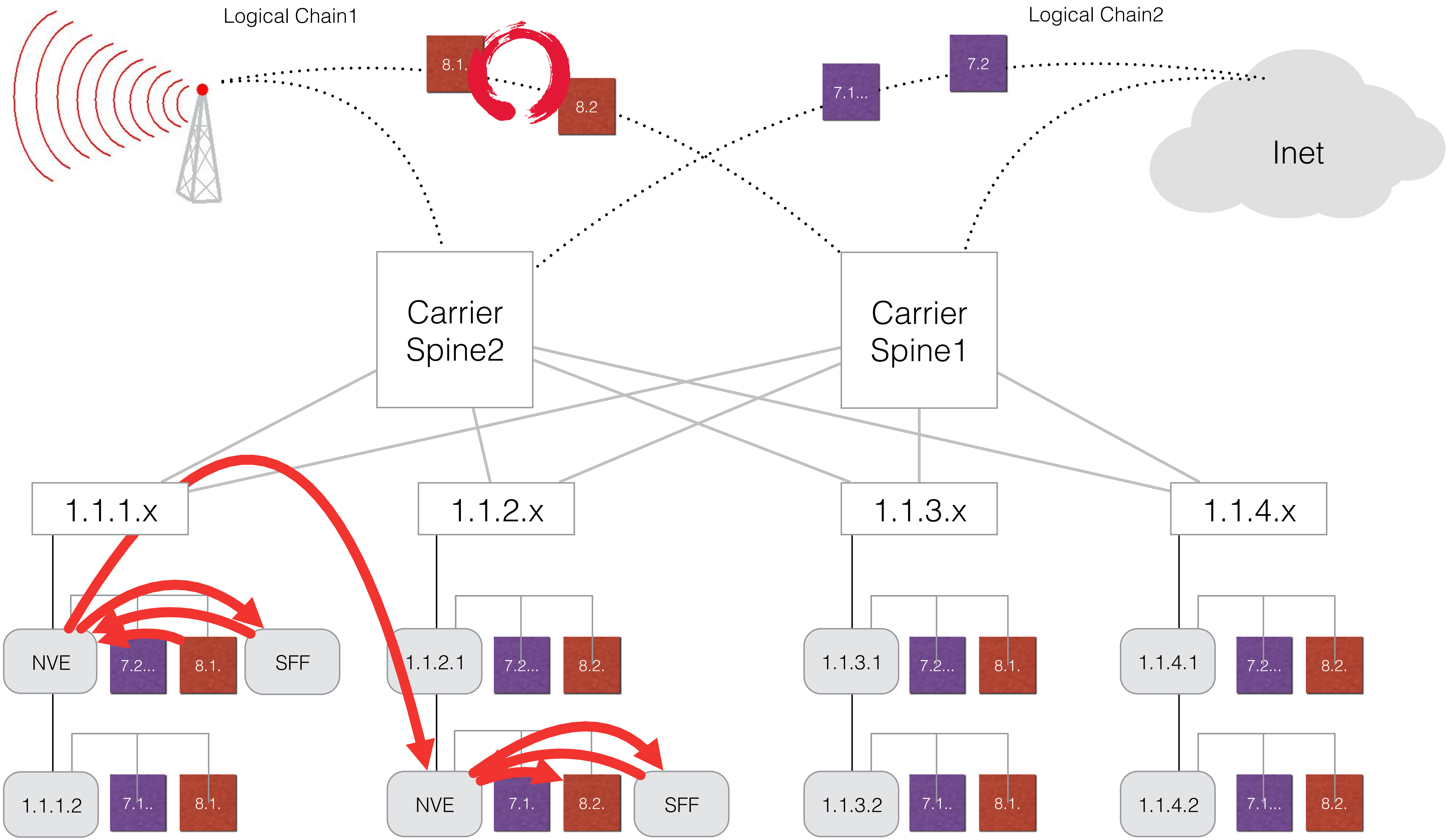
NVEs Separate the scaling of Resources & the scaling of Locations Any resource can be instantiated on any rack, racks still scale by subnets



SFFs implement a service with function chains, map Flow Instances to functional-middle-box virtualized resources



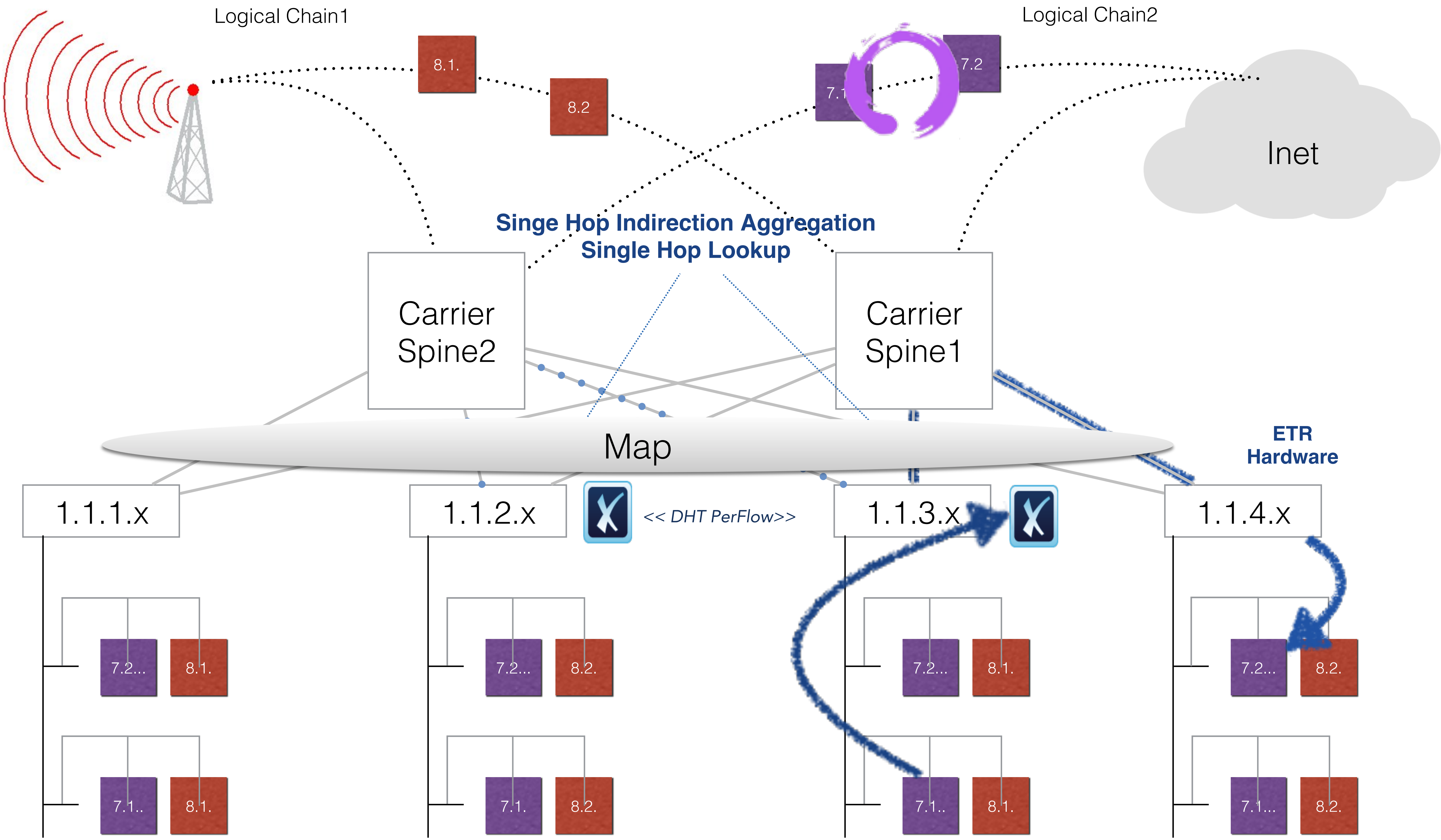
Worst Case "Hot Potato" per functional hop From F8.1 to F8.2 instances: NVE - NVE - SFF1 - NVE - NVE - SFF2 - NVE - NVE - SLB - NVE - NVE



Map-Reduce NVI at Consolidated Ingress

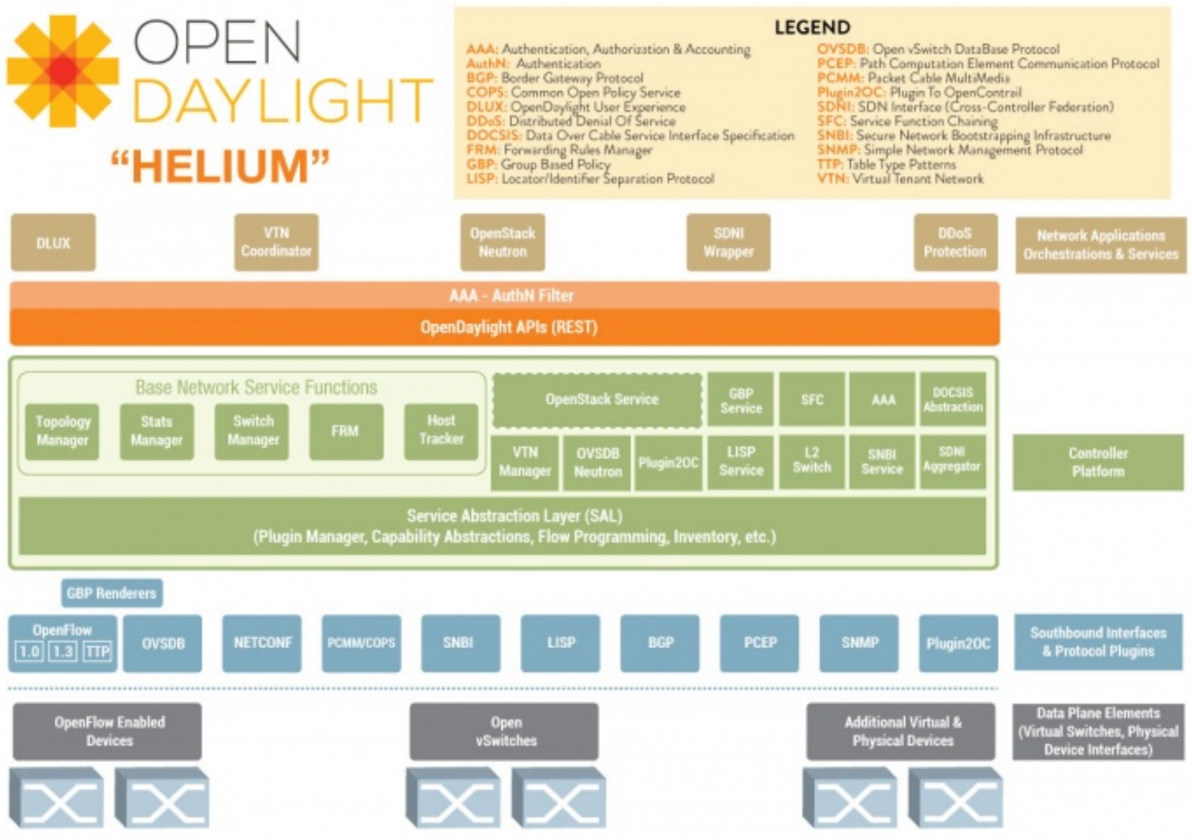
- Source <=> Network <=> Dest
- NVE <=> Underlay <=> NVE
- SFC <SFFs & VIPs> SFC
- Map&Encap Decap ActualIP

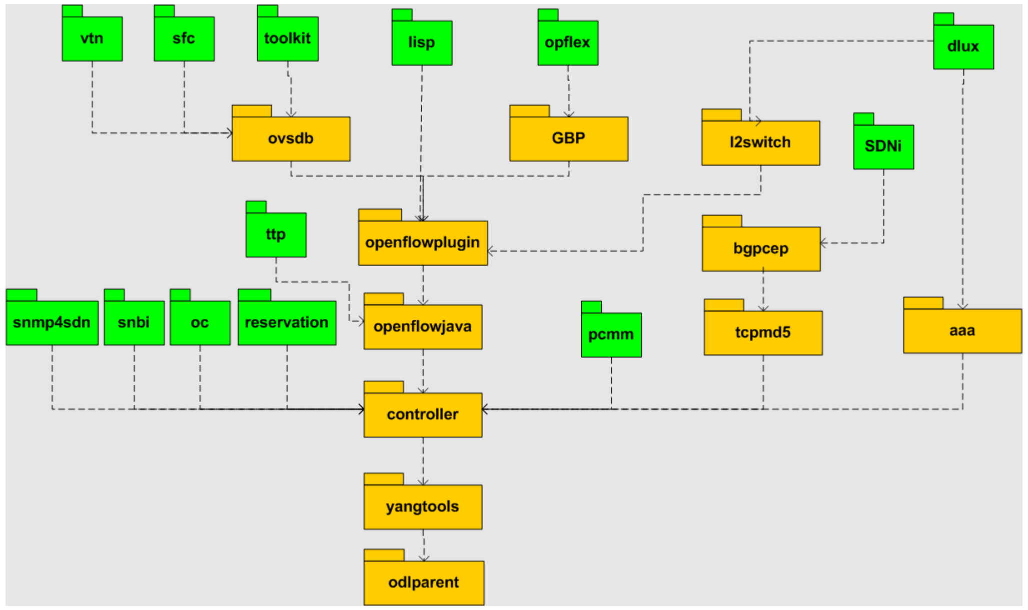
Best Case: Single Hop ToR-XTR consolidation, Single Hop ToR-DHT lookup



By Sharing Controller API and Global Context

- Global context: Mapping Authority for ID-Location, (mappable) Service Meta-Data for middleBoxes, and (mappable) Source-VIP affinity for load-balancing, (mappable) landmarks for segment-routing etc.


OPEN
DAYLIGHT
"HELIUM"
LEGEND
AAA: Authentication, Authorization & Accounting
AuthN: Authentication
BGP: Border Gateway Protocol
COPS: Common Open Policy Service
DLUX: OpenDaylight User Experience
DDoS: Distributed Denial Of Service
DOCSIS: Data Over Cable Service Interface Specification
FRM: Forwarding Rules Manager
GBP: Group Based Policy
LISP: Locator/Identifier Separation Protocol
OVSDB: Open vSwitch DataBase Protocol
PCEP: Path Computation Element Communication Protocol
PCMM: Packet Cable MultiMedia
Plugin2OC: Plugin To OpenContrail
SDNI: SDN Interface (Cross-Controller Federation)
SFC: Service Function Chaining
SNBI: Secure Network Bootstrapping Infrastructure
SNMP: Simple Network Management Protocol
TTP: Table Type Patterns
VTN: Virtual Tenant Network
DLUX
VTN Coordinator
OpenStack Neutron
SDNI Wrapper
DDoS Protection
Network Applications Orchestrations & Services
AAA - AuthN Filter
OpenDaylight APIs (REST)
Base Network Service Functions
Topology Manager
Stats Manager
Switch Manager
FRM
Host Tracker
OpenStack Service
GBP Service
SFC
AAA
DOCSIS Abstraction
VTN Manager
OVSDB Neutron
Plugin2OC
LISP Service
L2 Switch
SNBI Service
SDNI Aggregator
Controller Platform
Service Abstraction Layer (SAL)
(Plugin Manager, Capability Abstractions, Flow Programming, Inventory, etc.)
GBP Renderers
OpenFlow
1.0
1.3
TTP
OVSDB
NETCONF
PCMM/COPS
SNBI
LISP
BGP
PCEP
SNMP
Plugin2OC
Southbound Interfaces & Protocol Plugins
OpenFlow Enabled Devices
Open vSwitches
Additional Virtual & Physical Devices
Data Plane Elements (Virtual Switches, Physical Device Interfaces)



- Upon flow start or PacketIn a flow handler is dynamically selected from the controller lib
- The most specific flow handler will perform all source-dest-application VNI resolutions
- Including ID-Location, Subscriber-Service, Service-Instance.. n-tuple and specific VIP dependencies
- All resolutions must use RFC based modules so next hop NVI aggregation is interoperable

Thank You