Cisco Open Network Environment
Software Defined Networking and a new world of custom network applications

Ric Pruss, Principal Engineer
Network Operating Systems Technology Group, Cisco Systems Inc.

September 6, 2012

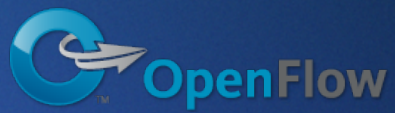
Why VMware Paid $1.26B for 70 Software Engineers

‘Having the services of those 100 engineers turned out to be incredibly valuable. They did something that really transformed the industry and they gave rise to an asset that’s worth plus-or-minus $40 billion.’
— Paul Maritz

Wired Article on 08.29.12

"…In the SDN architecture, the control and data planes are decoupled, network intelligence and state are logically centralized, and the underlying network infrastructure is abstracted from the applications…"

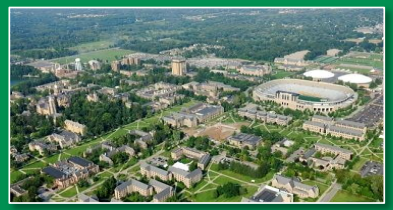
https://www.opennetworking.org/images/stories/downloads/white-papers/wp-sdn-newnorm.pdf



"…open standard that enables researchers to run experimental protocols in campus networks. Provides standard hook for researchers to run experiments, without exposing internal working of vendor devices……"

http://www.openflow.org/wp/learnmore/

Customer Insights discussions on SDN



Research/Academia

- Experimental OpenFlow/SDN components for production networks

➢ **Network "Slicing"**

Massively Scalable Data Center

- Customize with Programmatic APIs to provide deep insight into network traffic

➢ **Network flow management**

Cloud

- Automated provisioning and programmable overlay

➢ **Scalable Multi-tenancy"**

Service Providers

- Policy-based control and analytics to optimize and monetize service delivery

➢ **Agile service delivery**

Enterprise

- Virtualization of workloads, VDI, Orchestration of security profiles

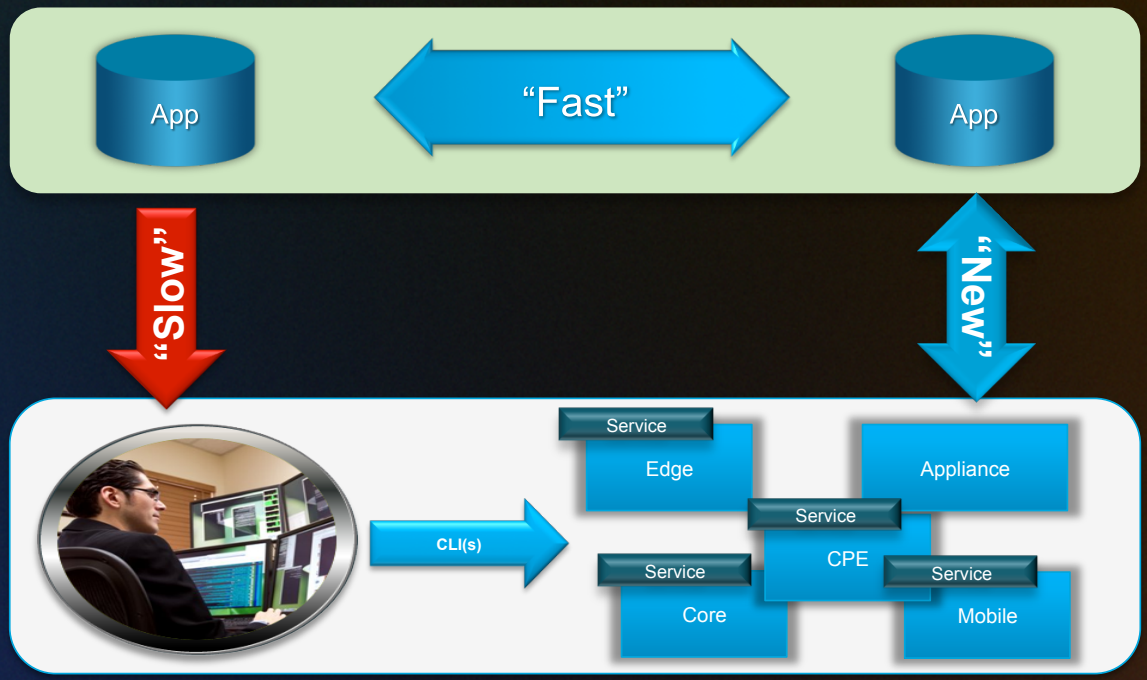
➢ **Private Cloud Automation**

Diverse Functionality Required Across Segments

Towards Programmatic Interfaces to the Network

Approaching Today's Application Developer Dilemma

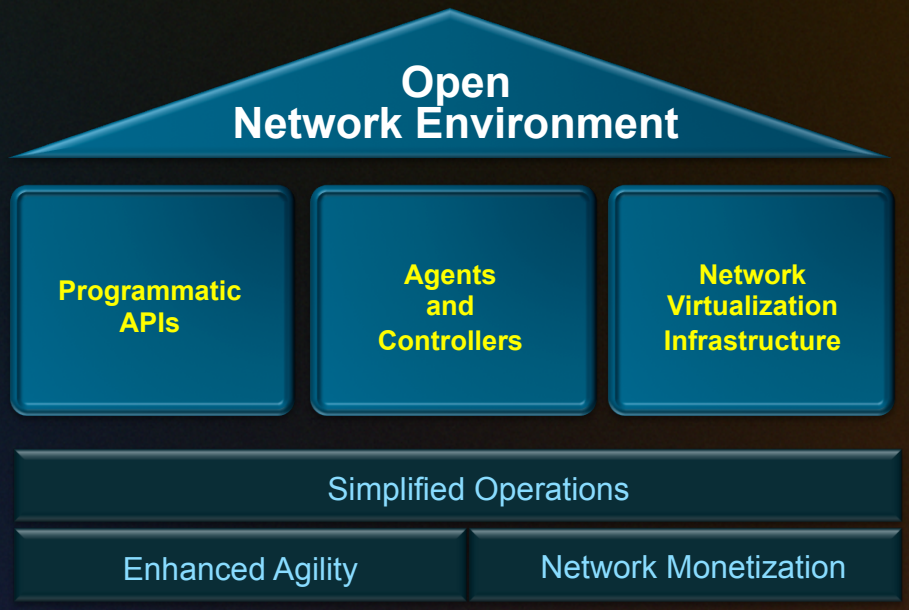
- Many Network Applications today:
  - OTT – for speed and agility
  - Avoid network interaction – complex and slow innovation
- New Model for Network Applications
  - Keep speed and agility
  - Full-duplex interaction with the network across multiple planes – extract, control, leverage network state



A New Programming Paradigm is Needed

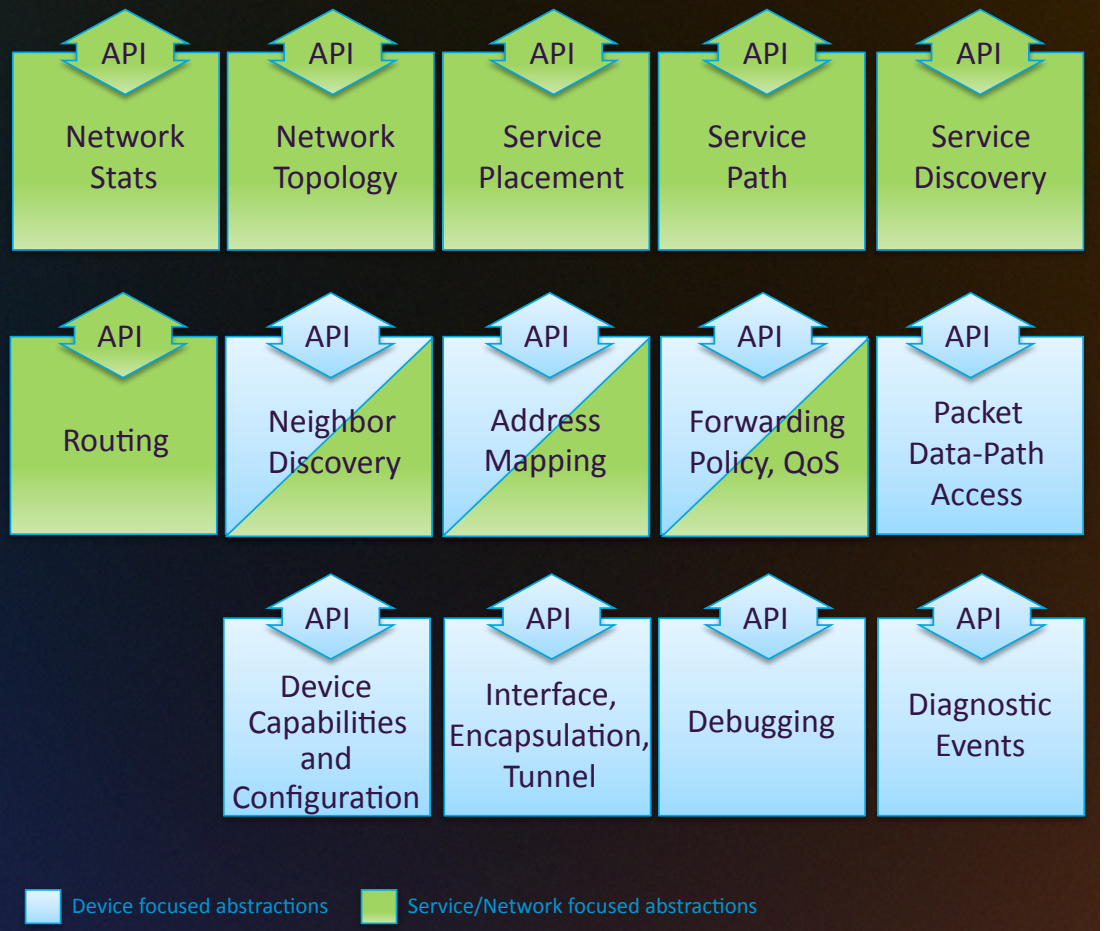
Common Concepts – The Open Network Environment
Approaching a Definition

- Open Network Environment – Complementing the Intelligent Network
  - *Preserve what is working*: Resiliency, Scale and Security, Comprehensive feature-set
  - *Evolve for Emerging Requirements*: Operational Simplicity, Programmability, Application-awareness
- The Open Network Environment integrates with existing infrastructure
  - Software Defined Network concepts are a component of the Open Network Environment
  - The OpenFlow protocol can be used to link agents and controllers, and as such is component of SDN as well



Approaching abstractions for Networking

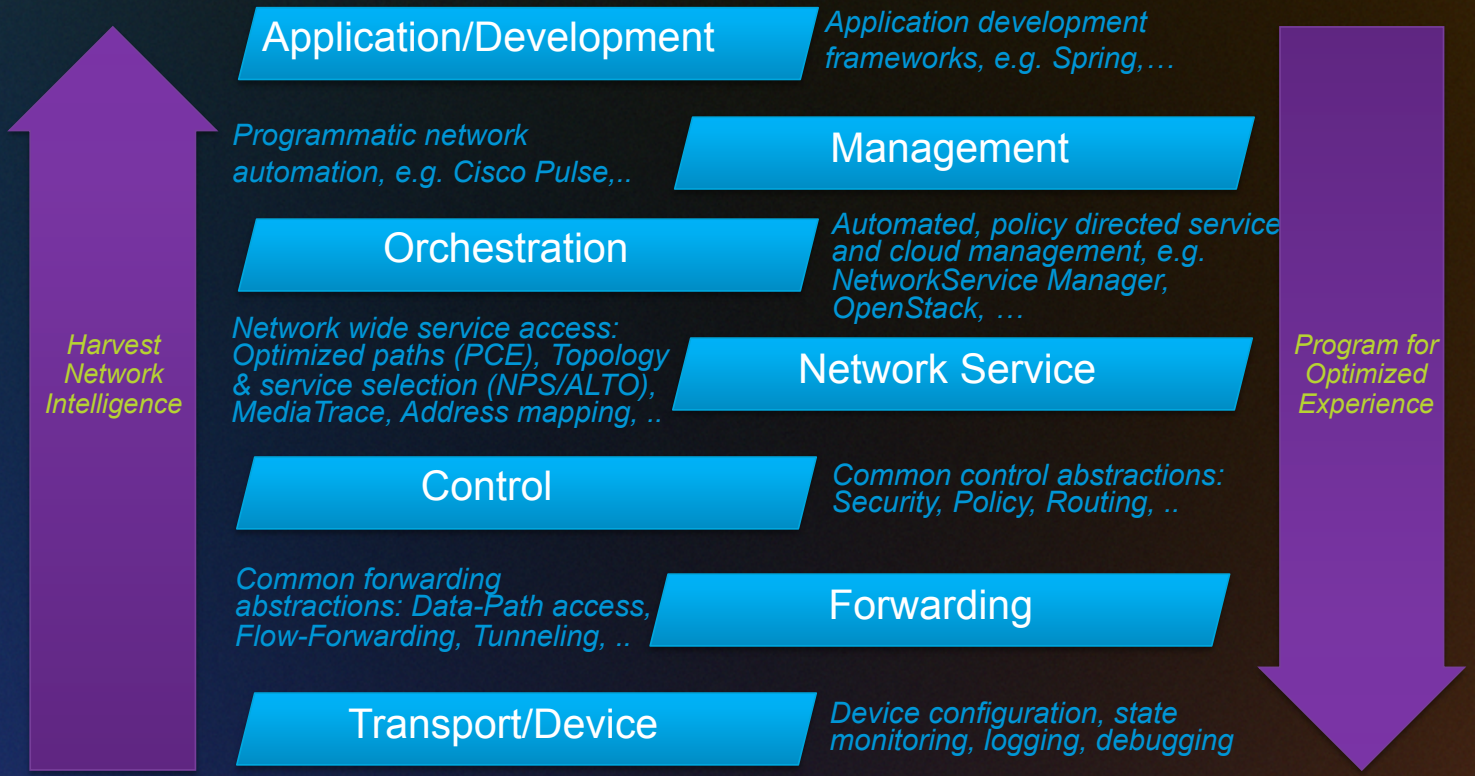
- Data-plane Abstractions – ISO Layering
  - Data plane abstractions key to Internet's success
- Abstractions for the other planes (control, services, management, orchestration,..)
  - … are missing
- Define network abstractions and associated APIs
  - Enable a holistic Network Programming model
  - Leverage and extend infrastructure at pace of the business
  - Deploy common applications across all devices
  - Extend/upgrade/add features without upgrading the network operating system



Programmatic Network Access – Multiple Layers

Full-Duplex access to the network at multiple layers and networking planes

- Enable API platform kit across all platforms, to integrate with development environments
- Accelerate development of network applications: Completely integrated stack from device to network
- Multiple deployment modes (local and remote APIs)
- Multiple Language Support (C, Java, …)
- Integrate with customer development environment to deliver enhanced functionality
- Reduced time to market by leveraging common platform for building services



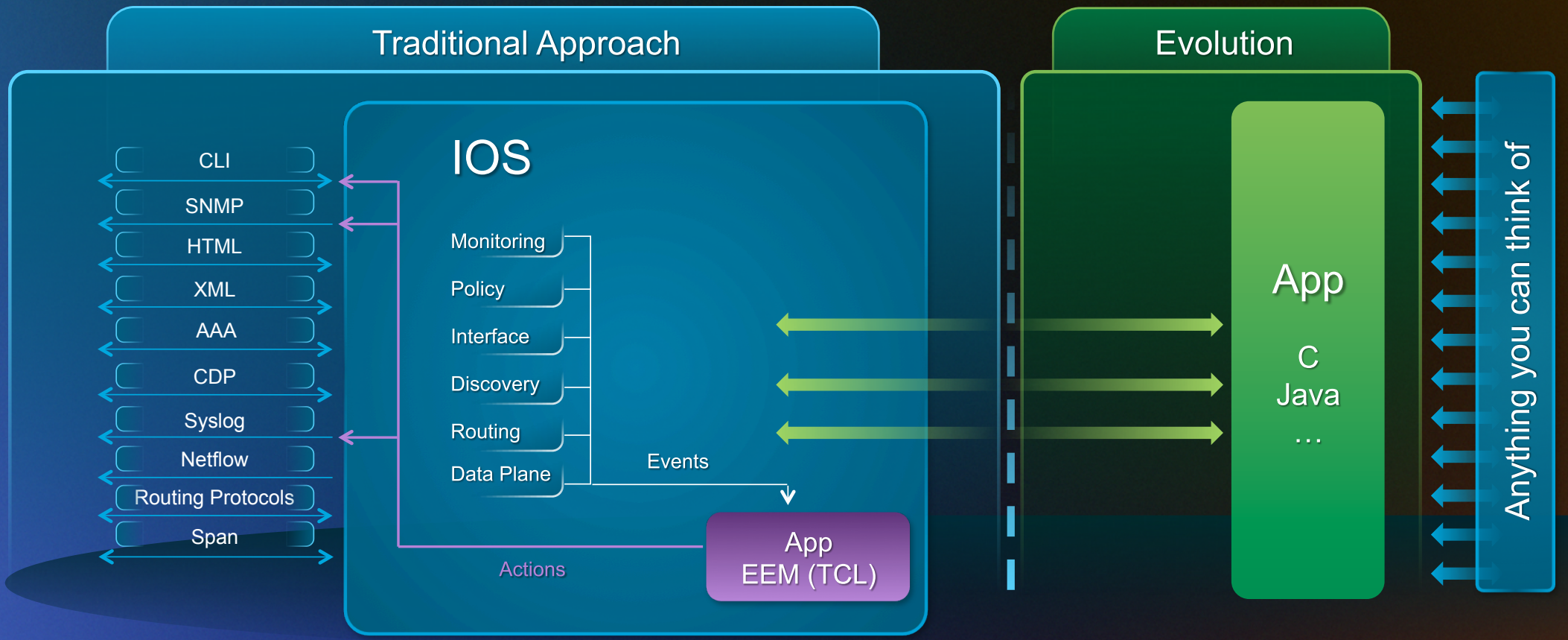
APIs make Abstractions available to Programmers

Example: Cisco's onePK (one Programming Kit) – Get your build on!

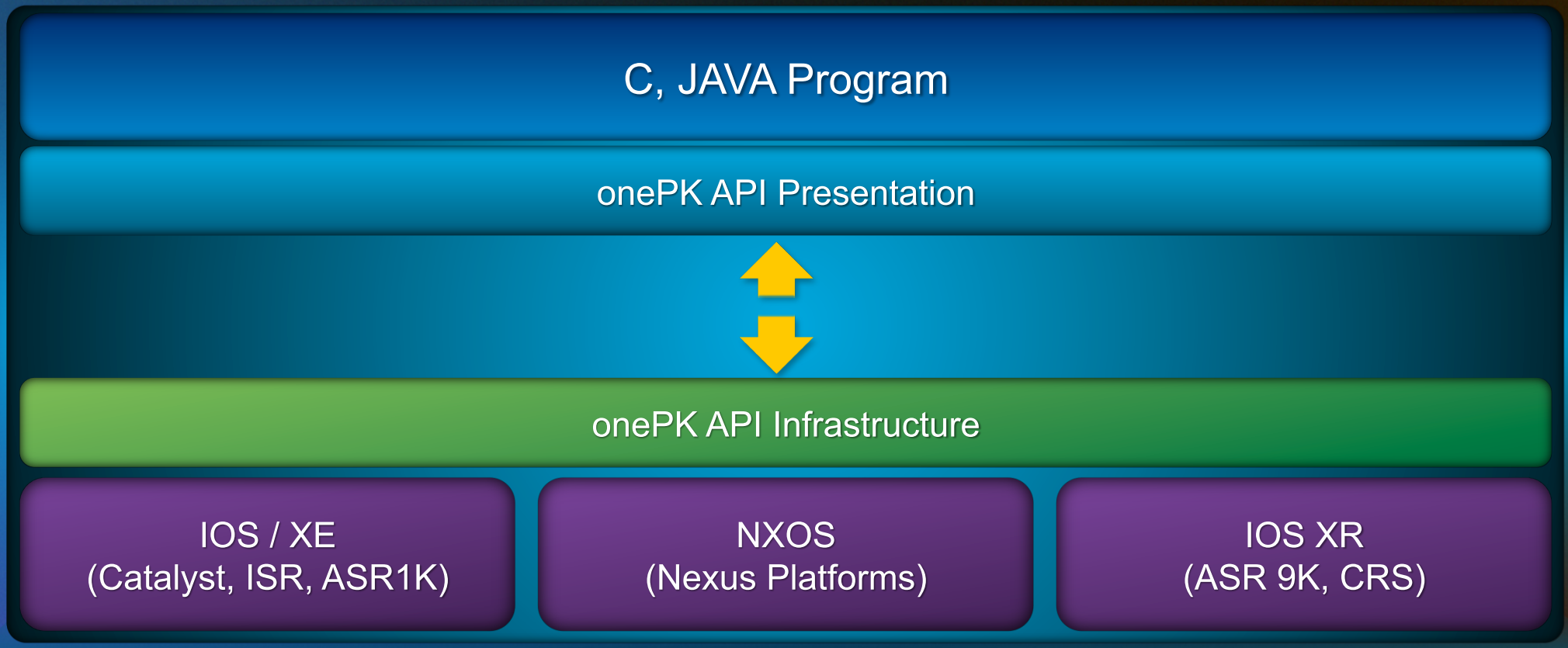
Flexible development environment to:

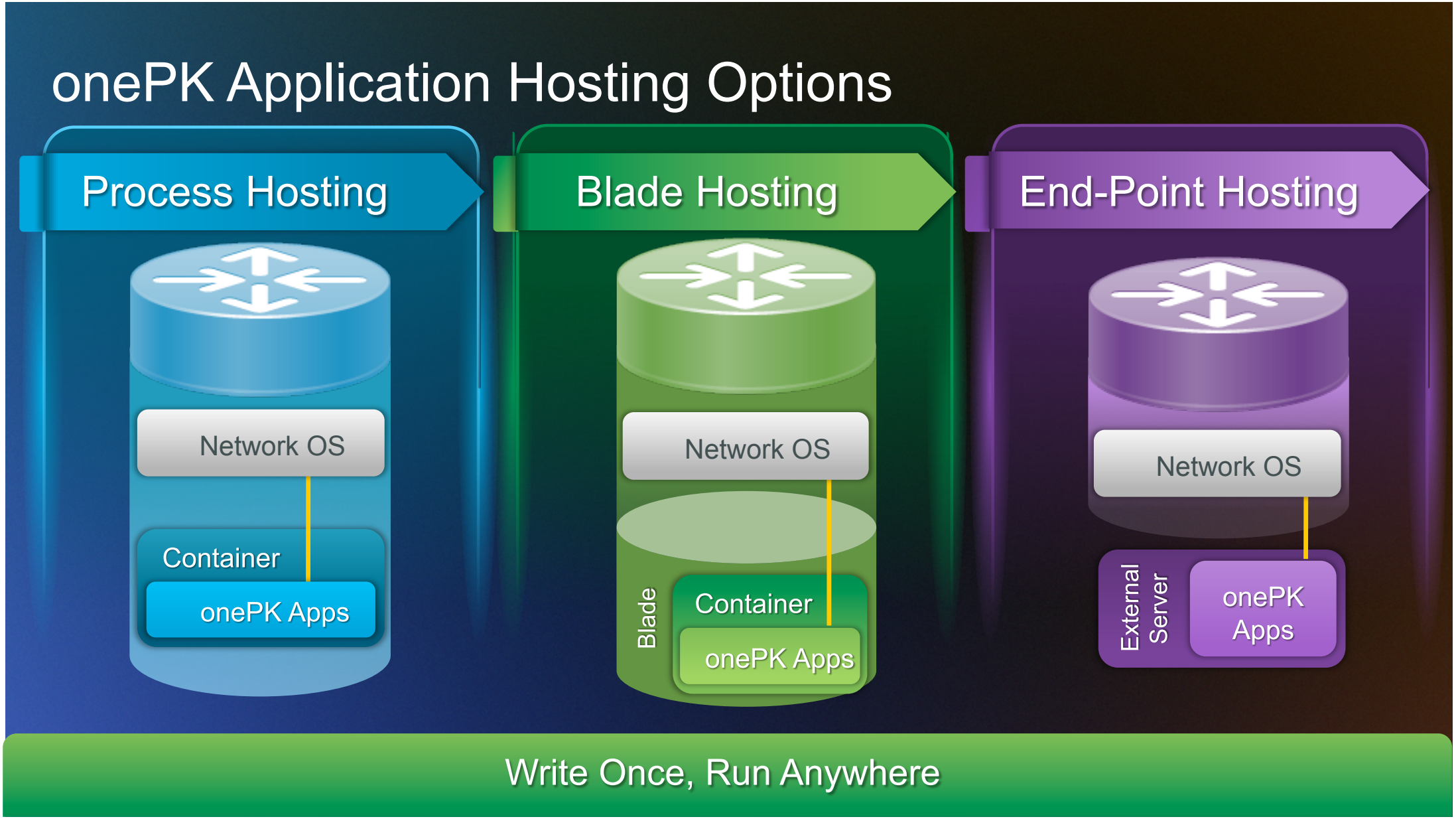
- Innovate
- Extend
- Automate
- Customize
- Enhance
- Modify

Evolving How We Interact With The Network Operating System



onePK Architecture




onePK Application Hosting Options
Process Hosting
Network OS
Container
onePK Apps
Blade Hosting
Network OS
Blade
Container
onePK Apps
End-Point Hosting
Network OS
External Server
onePK Apps
Write Once, Run Anywhere

Yes, it is secure

Security Five Ways

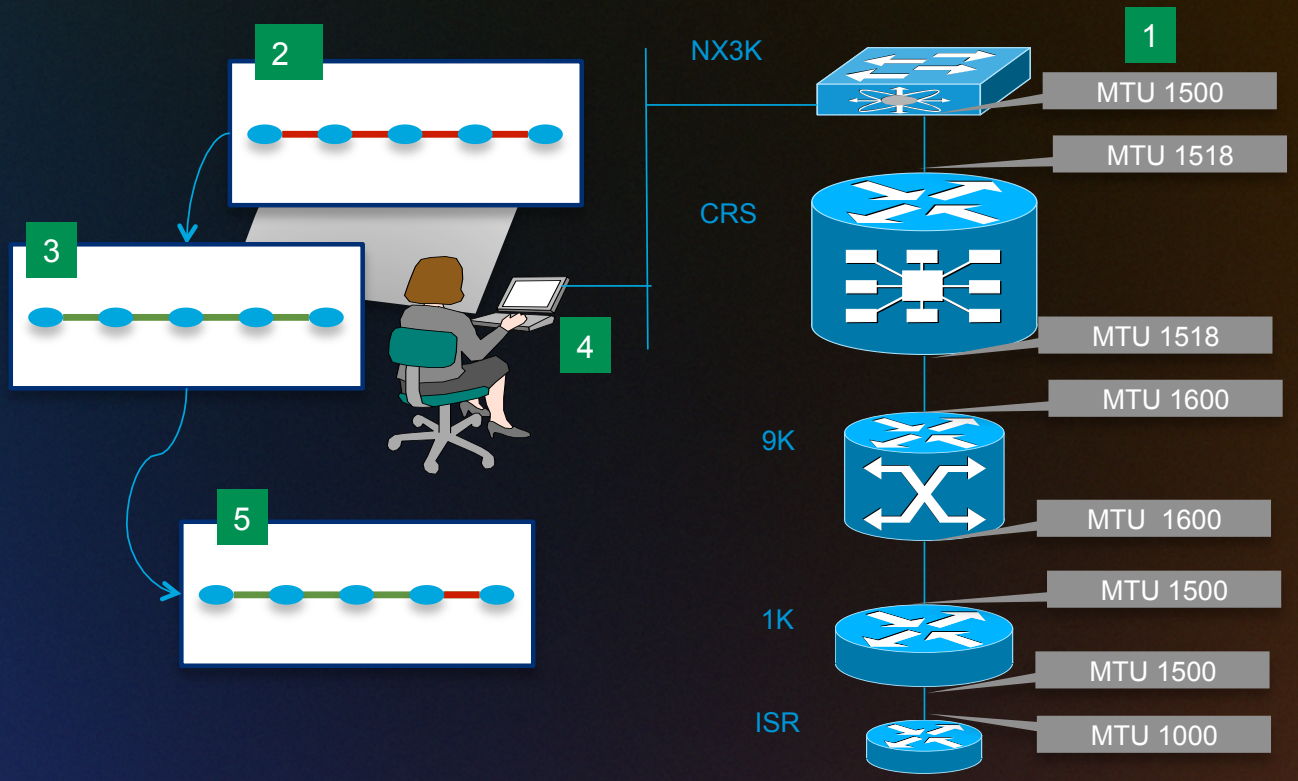
onePK APIs are Grouped in Service Sets

Base Service Set	Description
Data Path	Provides packet delivery service to application: Copy, Punt, Inject
Policy	Provides filtering (NBAR, ACL), classification (Class-maps, Policy-maps), actions (Marking, Policing, Queuing, Copy, Punt) and applying policies to interfaces on network elements
Routing	Read RIB routes, add/remove routes, receive RIB notifications
Element	Get element properties, CPU/memory statistics, network interfaces, element and interface events
Discovery	L3 topology and local service discovery
Utility	Syslog events notification, Path tracing capabilities (ingress/egress and interface stats, next-hop info, etc.)
Developer	Debug capability, CLI extension which allows application to extend/integrate application's CLIs with network element

Example: Simplified Management

Problem: Misconfigurations cause network outages, degrade performance, impact SLAs.
Value proposition: Get, set, and detect configuration changes via cross-platform API

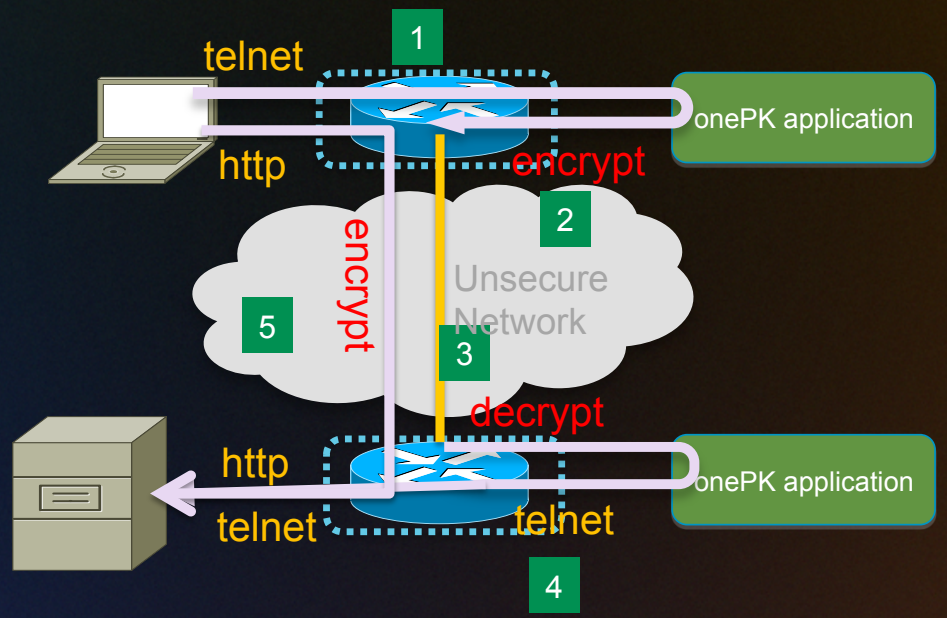
1. Network begins with mismatched parameters on either side of link (e.g. MTU)
2. Application checks parameters on either side and identifies mismatches (red lines)
3. Application sets parameters to match (lines turn green)
4. Application registers for events related to parameters change.
5. Users logs into console and manually changes parameter. Topology indicates change.



Example: Custom Encryption

Problem: Customers want custom encryption on specific traffic types
Value proposition: Punt traffic of interest, encrypt, and re-inject.

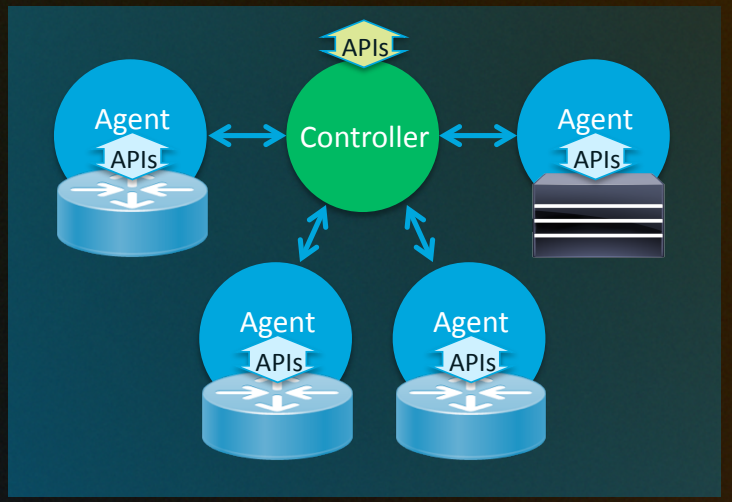
1. Policy APIs on ingress router are set to punt telnet and syslog to app
2. App encrypts punted traffic and re-injects into data path.
3. Policy APIs on egress router punt telnet and syslog to app
4. App decrypts punted traffic and re-injects into data path.
5. Traffic that does not match policy passes through unencrypted.

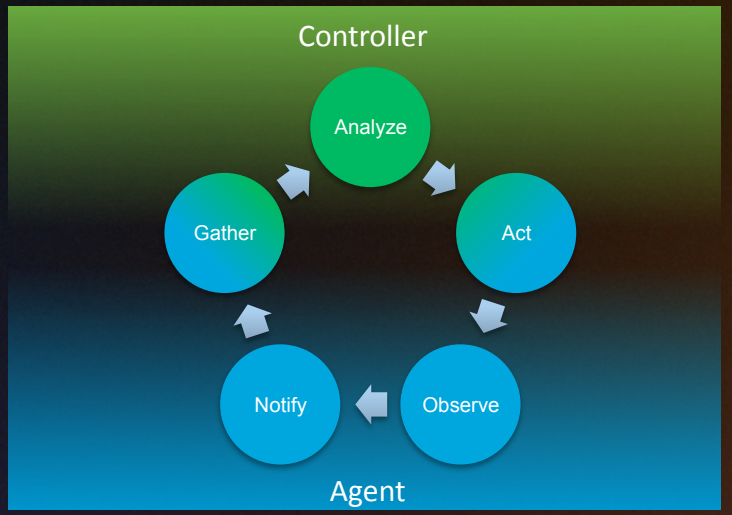


Agents and Controllers

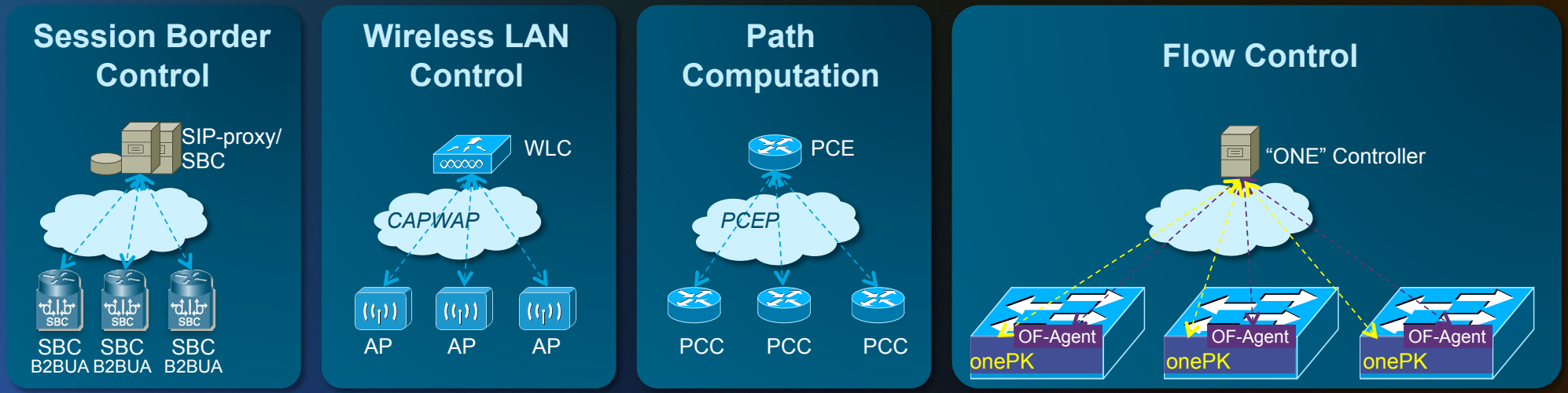
Consolidate State Across Multiple Network Elements

- Some network delivered functionality benefits from logically centralized coordination across multiple network devices
  - Functionality typically domain, task, or customer specific
  - Typically multiple Controller-Agent pairs are combined for a network solution
- Controller
  - Process on a device, interacting with a set of devices using a set of APIs or protocols
  - Offer a control interface/API
- Agent
  - Process or library on a device, leverages device APIs to deliver a task/domain specific function
- Controller-Agent Pairs offer APIs which integrate into the overall Network API suite





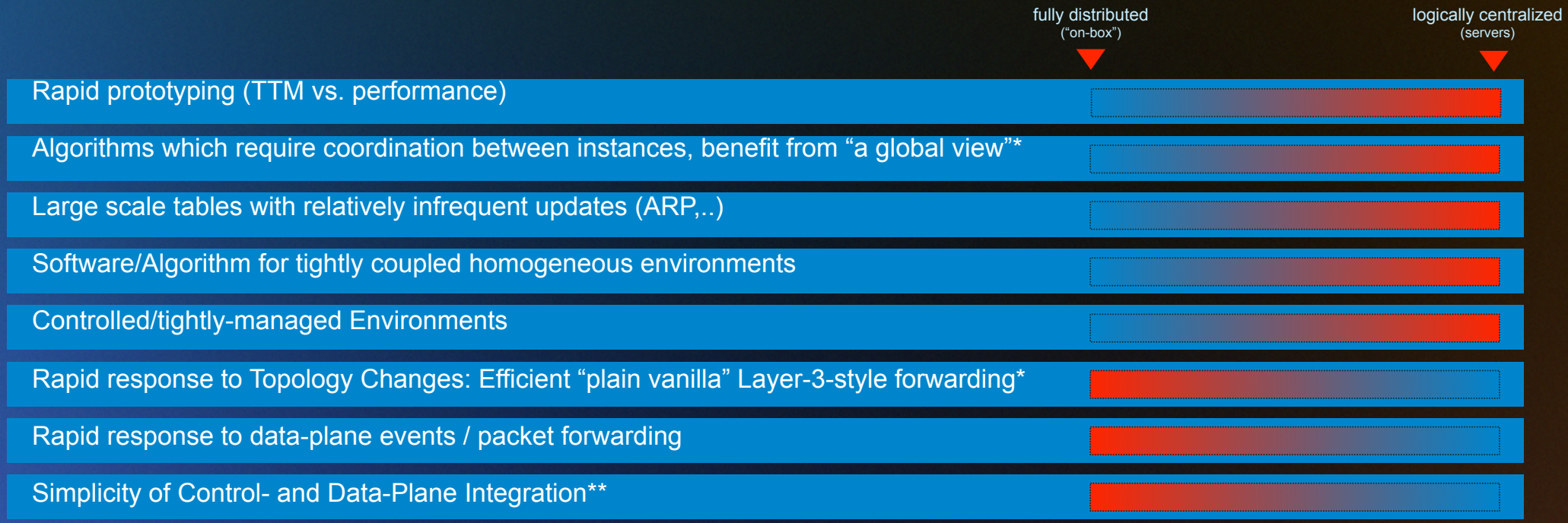
Agents and Controllers – Task Specific Sets



- Networking already leverages a great breath of Agents and Controllers
  - Current Agent-Controller pairs always serve a specific task (or set of tasks) in a specific domain
- System Design: Trade-off between Agent-Controller and Fully Distributed Control
  - Control loop requirements differ per function/service and deployment domain
  - "As loose as possible, as tight as needed"
  - Latency, Scalability, Robustness, Consistency, Availability

Evolving the Control Plane Environment

Exploring the tradeoff between Agents and Controllers – and fully distributed Control

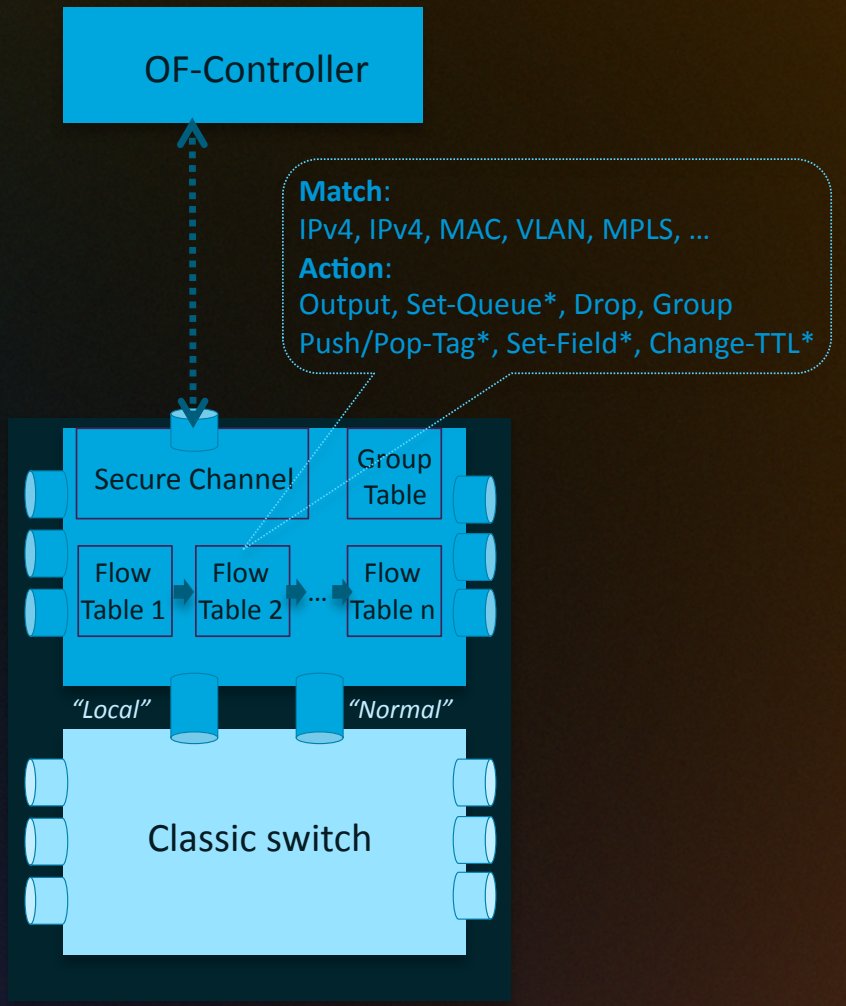


** Past experience (e.g. PSTN AIN, Softswitches/IMS, SBC): CP/DP split requires complex protocols between CP and DP.

* See also: Martin Casado's Blog: http://networkheresy.wordpress.com/2011/11/17/is-openflowsdn-good-at-forwarding/

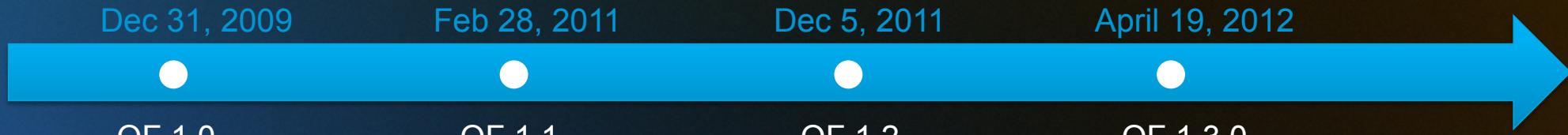
Controllers and Agents
OpenFlow as a Technology

- Original Motivation
  - Research/Academia to experiment with new control paradigms
- Base Assumption
  - Providing reasonable abstractions for control requires the control system topology to be decoupled from the physical network topology
- OpenFlow Components
  - *Application Layer Protocol*: OF-Protocol
  - *Device Model*: OF-Device Model (abstraction of a device with Ethernet interfaces and a set of forwarding capabilities)
  - *Transport Protocol*: Connection between OF-Controller and OF-Device*
- Observation:
  - OF-Controller and OF-Device need pre-established IP-connectivity



* TLS, TCP – OF 1.3.0 introduces auxiliary connections, which can use TCP, TLS, DTLS, or UDP.

OpenFlow Evolution



OF 1.0

- Single Table
- L2, IPv4 focused matching

OF 1.1

- Multiple Tables
- MPLS, VLAN matching
- Groups: {Any-,Multi-}cast
- ECMP

OF 1.2

- IPv6
- Flexible-length matching

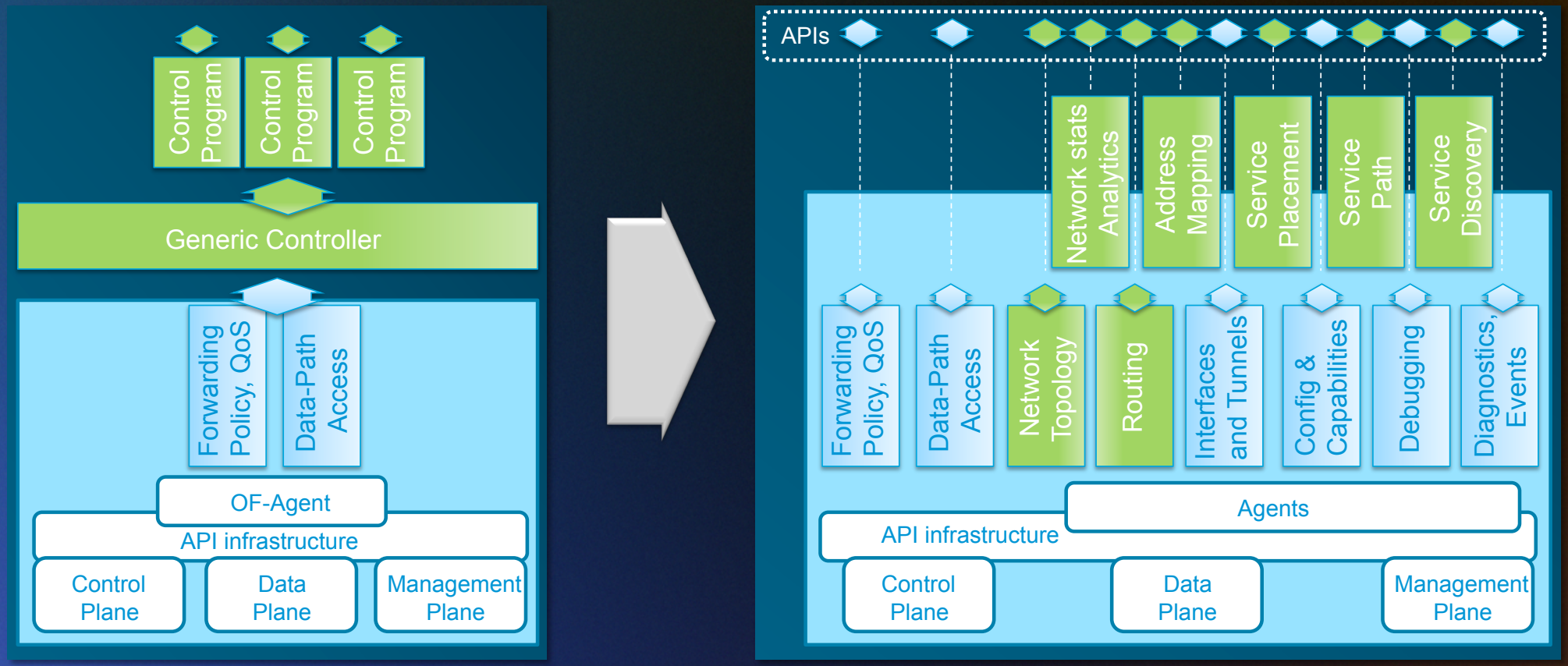
OF 1.3.0

- 802.1ah PBB
- Multiple parallel channels between Switch and Controller

A few topics of ongoing work

High availability model for device and controller (state re-sync etc.)
Hardware friendly switch model – "typed tables" → New Forwarding Abstractions WG
Security model (granular access control)
L3-forwarding model
Enhanced Statistics
Management infrastructure (evolution of OF-CONFIG)
Testing and certification framework
Hybrid device/network deployment capability (→ Hybrid WG)

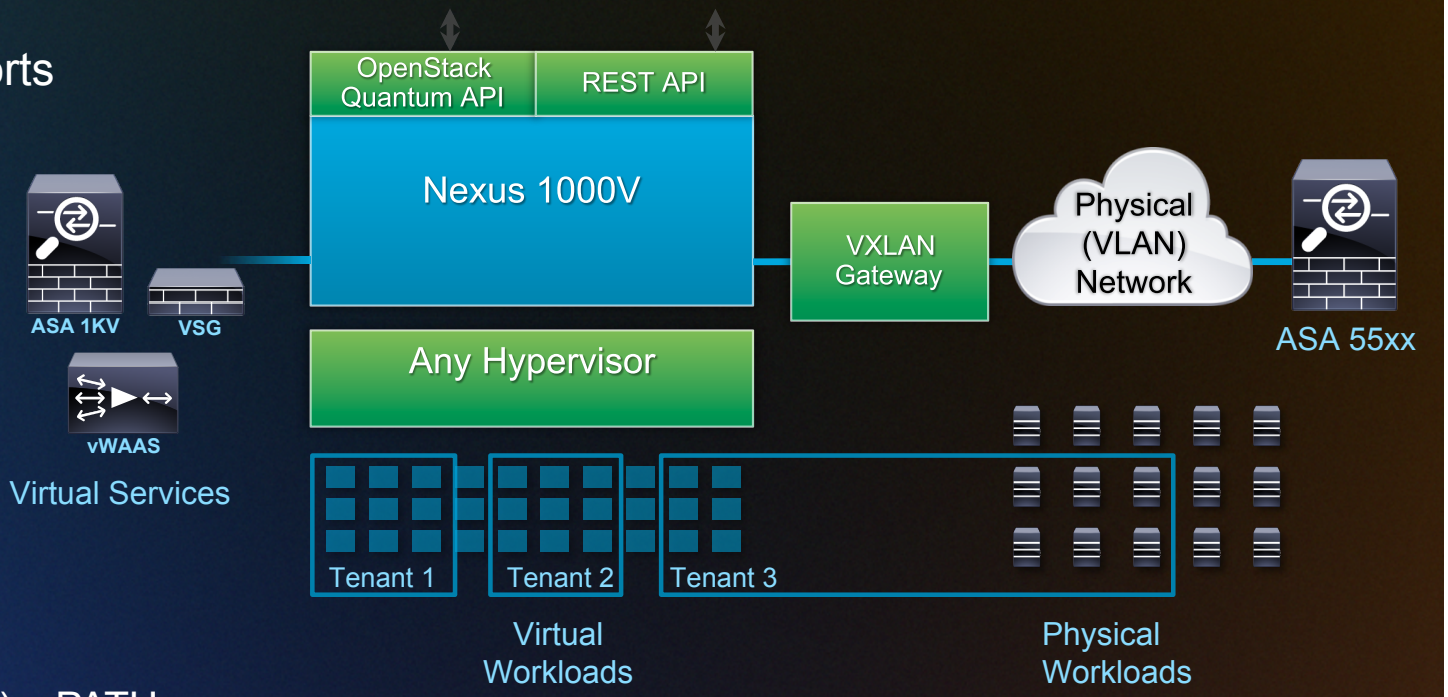
Evolve the early SDN Model…
… acknowledge the need for diverse abstractions



Virtual Overlay Networks

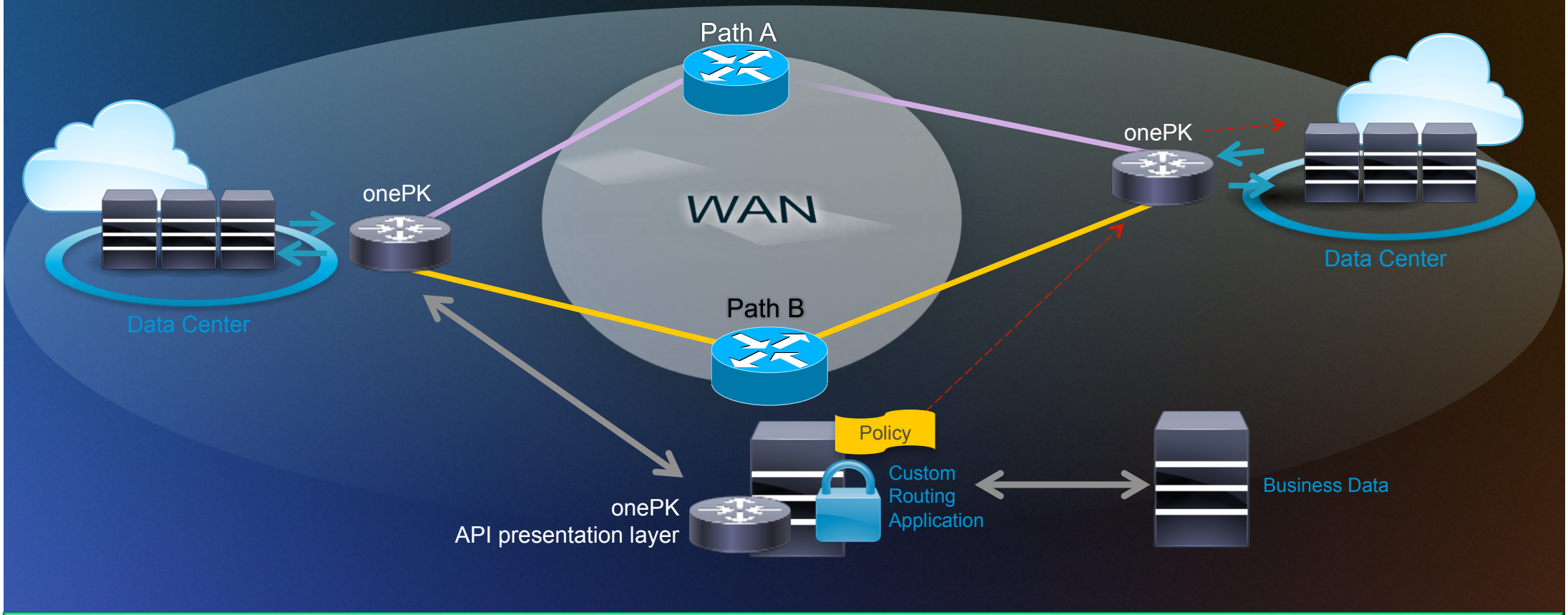
Example: Virtual Overlay Networks and Services with Nexus 1000V

- Large scale L2 domains:
  Tens of thousands of virtual ports
- Common APIs
  Incl. OpenStack Quantum API's for orchestration
- Scalable DC segmentation and addressing
  VXLAN
- Virtual service appliances and service chaining/traffic steering
  VSG (cloud-ready security), vWAAS (application acceleration), vPATH
- Multi-hypervisor platform support: ESX, Hyper-V, OpenSource Hypervisors
- Physical and Virtual: VXLAN to VLAN Gateway



Example: Custom Routing

Data Center Traffic Forwarding Based on a Custom Algorithm



Unique Data Forwarding Algorithm Highly Optimized for the Network Operator's Application

Custom Routing Application

- Business Problem

  Network operator needs to direct traffic using unique or external decision criteria; e.g. route long lived elephant flows, backup traffic using the lowest $ cost path, or have trading information follow the shortest delay path

- Solution

  Custom route application built and deployed using onePK, communicating directly with the forwarding plane.

  Unique data forwarding algorithm highly optimized for the network operator's application

- Approach (for e.g. Latency based routing) leveraging onePK

  (1) Retrieve Network Topology using onePK Discovery Service Set

  (2) Measure Link Latency through onePK programmatic interface to EEM and IPSLA

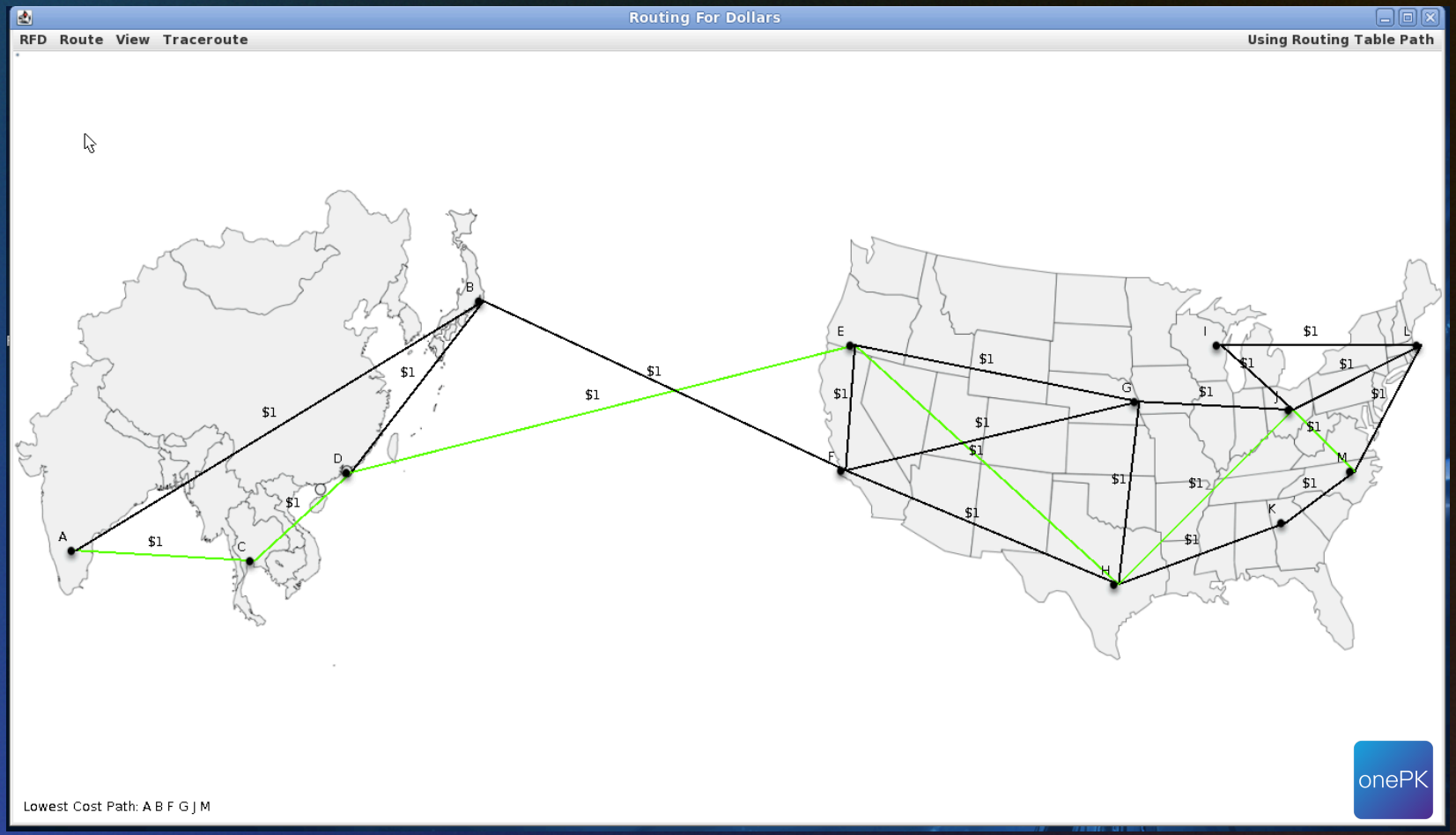
  (3) Compute optimal routing information (here: Public domain version of Dijkstra with latency as metric)

  (4) Install optimal routes in routers using onePK Routing Service Set, use Policy Service Set to classify traffic which should follow optimal delay route

  … In case of loss of connectivity to the "custom routing app", fall back to normal (e.g. EIGRP) routing

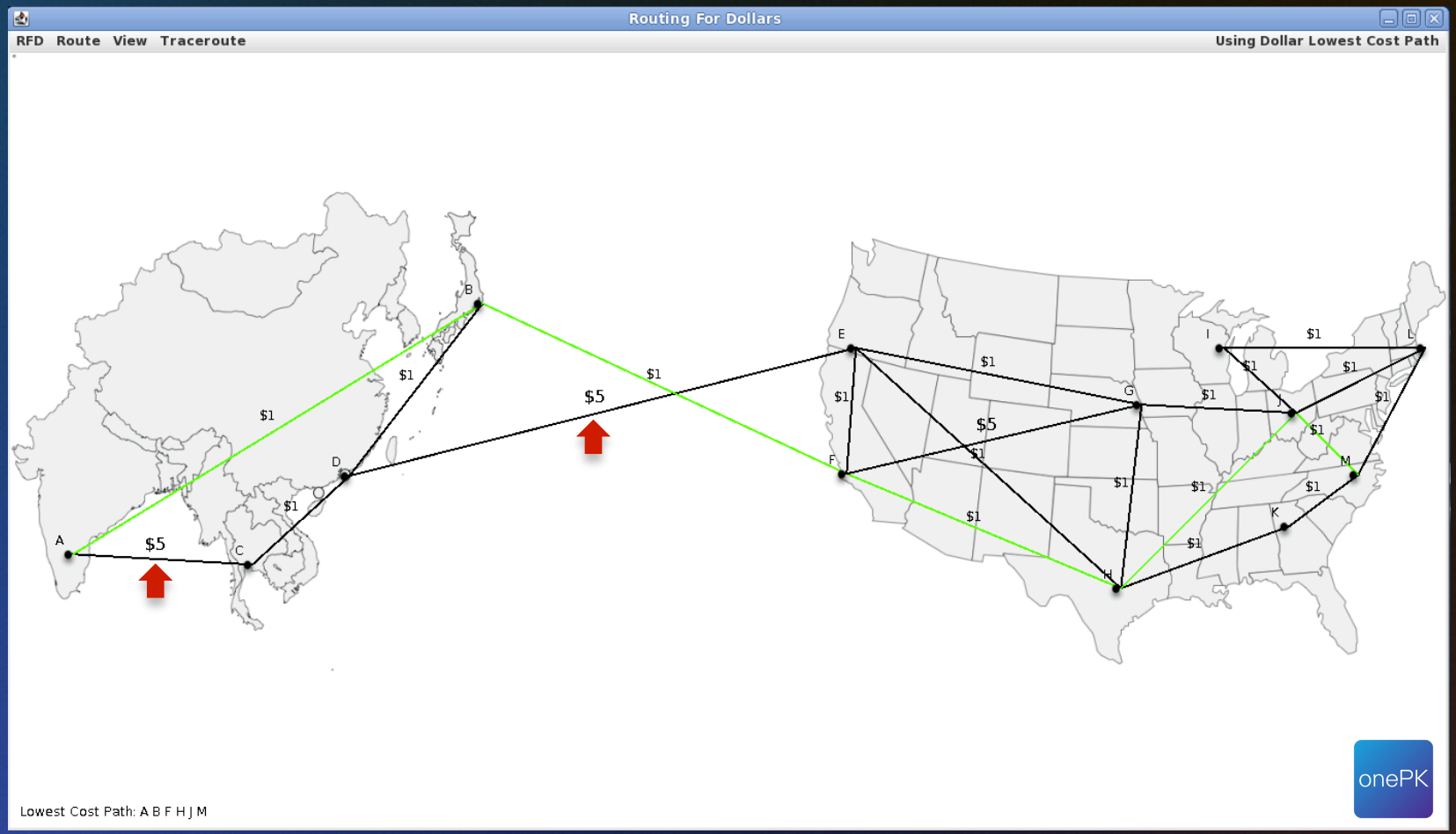
Custom Routing

Initial Setup: Default routing using EIGRP



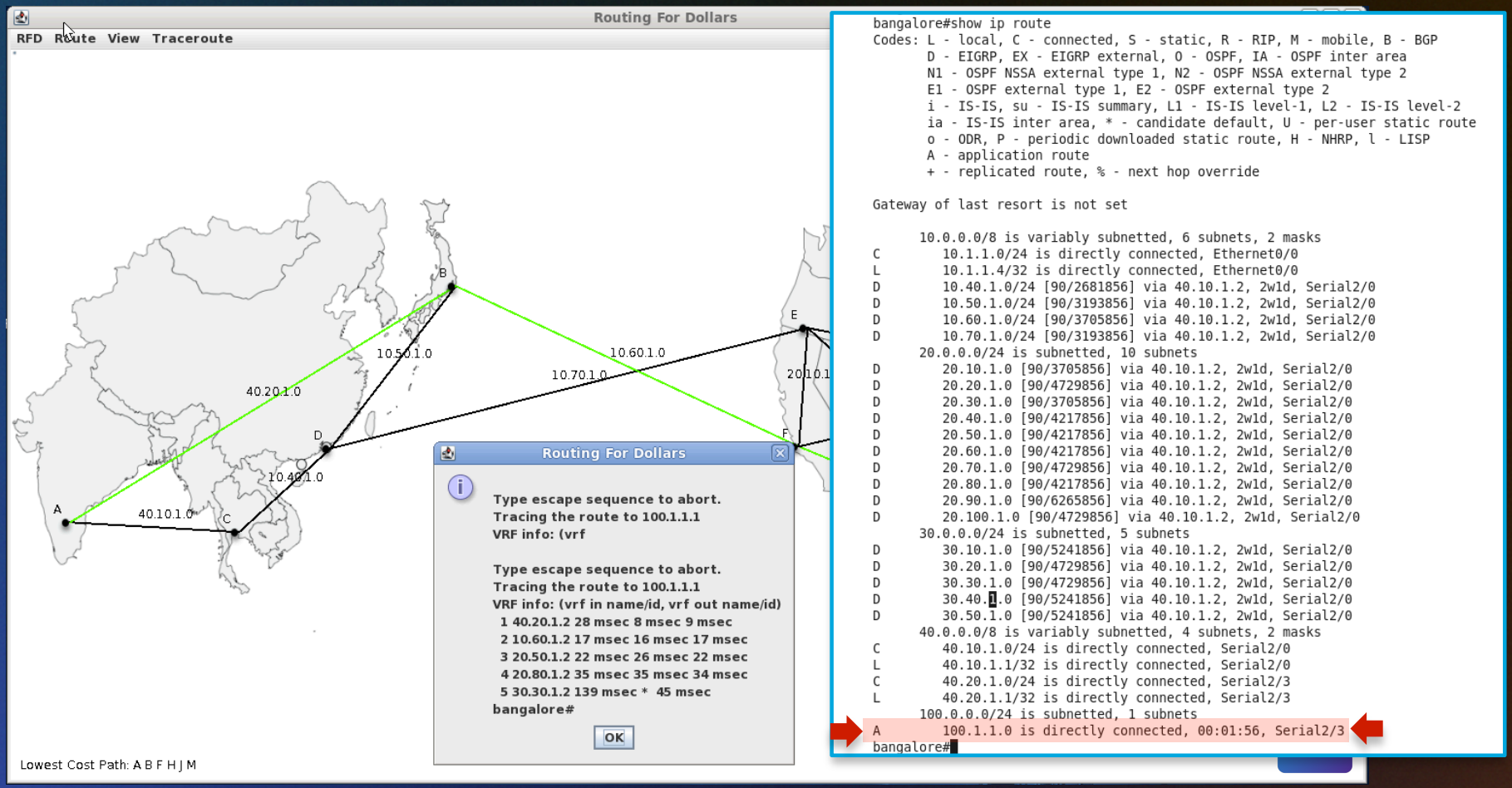
Custom Routing

Routing for Dollars: Application driven routes installed in network



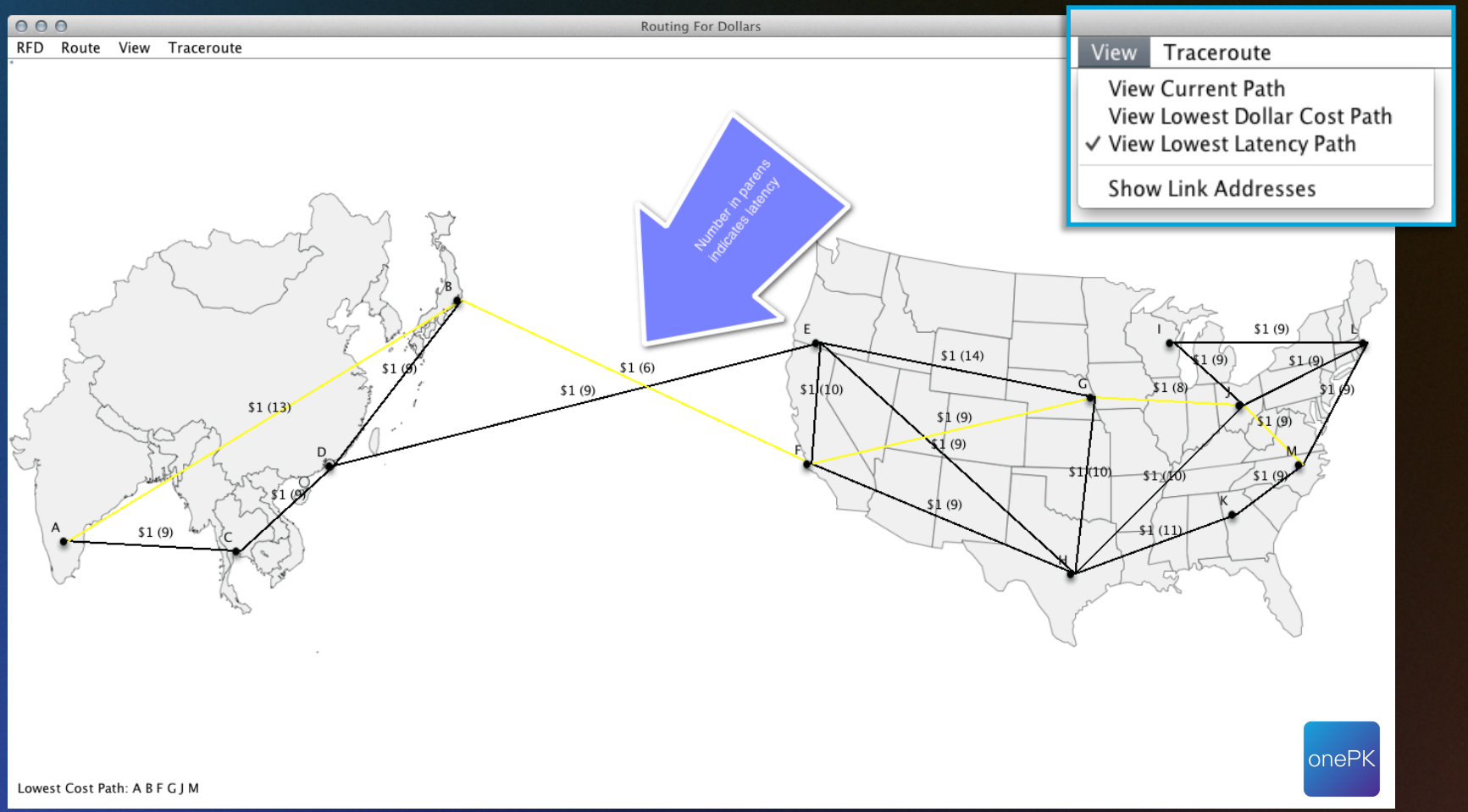
Custom Routing

Tracing the application installed route – using the developer and element services



Custom Routing – Another Example

Alternate Metrics: Measured Link Delay – Using EEM/IPSLA Service Set



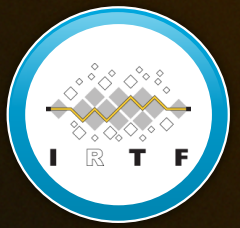
Custom Routing: Statistics

- Code Metrics
  - Total lines of code: 4700 (JAVA)
  - 40% SWING GUI
  - 20% Dijkstra's algorithm, lowest cost path determination
  - 25% Housekeeping: Node and link database
  - 15% Calls to onePK infrastructure + error checking
- Code increase to add "Latency based routing" on top of "Routing for Dollars"
  - 100 lines of code

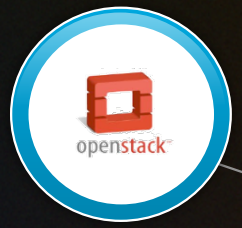
Industry Standards & Forums

802.1 Overlay Networking Projects

Open Network Research Center at Stanford University



Technical Advisory Group, Working Groups:
Config, Hybrid, Extensibility, Futures/FPMOD/OF2.0



Initiatives
Quantum
Donabe

Open Source Cloud Computing project

Overlay Working Groups:
NVO3, L2VPN, TRILL, L3VPN, LISP, PWE3
API Working Groups/BOFs
NETCONF, ALTO, CDNI, XMPP, SDNP, I2AEX
Controller Working Groups:
PCE, FORCES
New work items:
IRS – Interface to the Routing System

Closing Thoughts

- This is version 0.1 of SDN and API's
- Standards may follow
- Questions

- Ex-Cisco Sydney OS team – 7 great coders looking for a new gig

Thank you.