OpenADN: A Case for Open Application Delivery Networking

Subharthi Paul, **Raj Jain**, **Jianli Pan**
Washington University in Saint Louis
{Pauls, jain, jp10}@cse.wustl.edu

Jay Iyer, **Dave Oran**
Cisco Systems
{jiyer,oran}@cisco.com

International Conference on Computer Communications and Networking (ICCCN 2013)
Nassau, Bahamas, July 30-August 2, 2013

These slides and audio/video recordings are available at:
http://www.cse.wustl.edu/~jain/talks/ad_ic3np.htm

©2013 Raj Jain

Overview

1. Application Delivery in a Data Center
2. Application Delivery in a Multi-Cloud Environment
3. Our Solution: OpenADN
4. OpenADN Design Issues
5. OpenADN Design

©2013 Raj Jain

Clouds and Mobile Apps

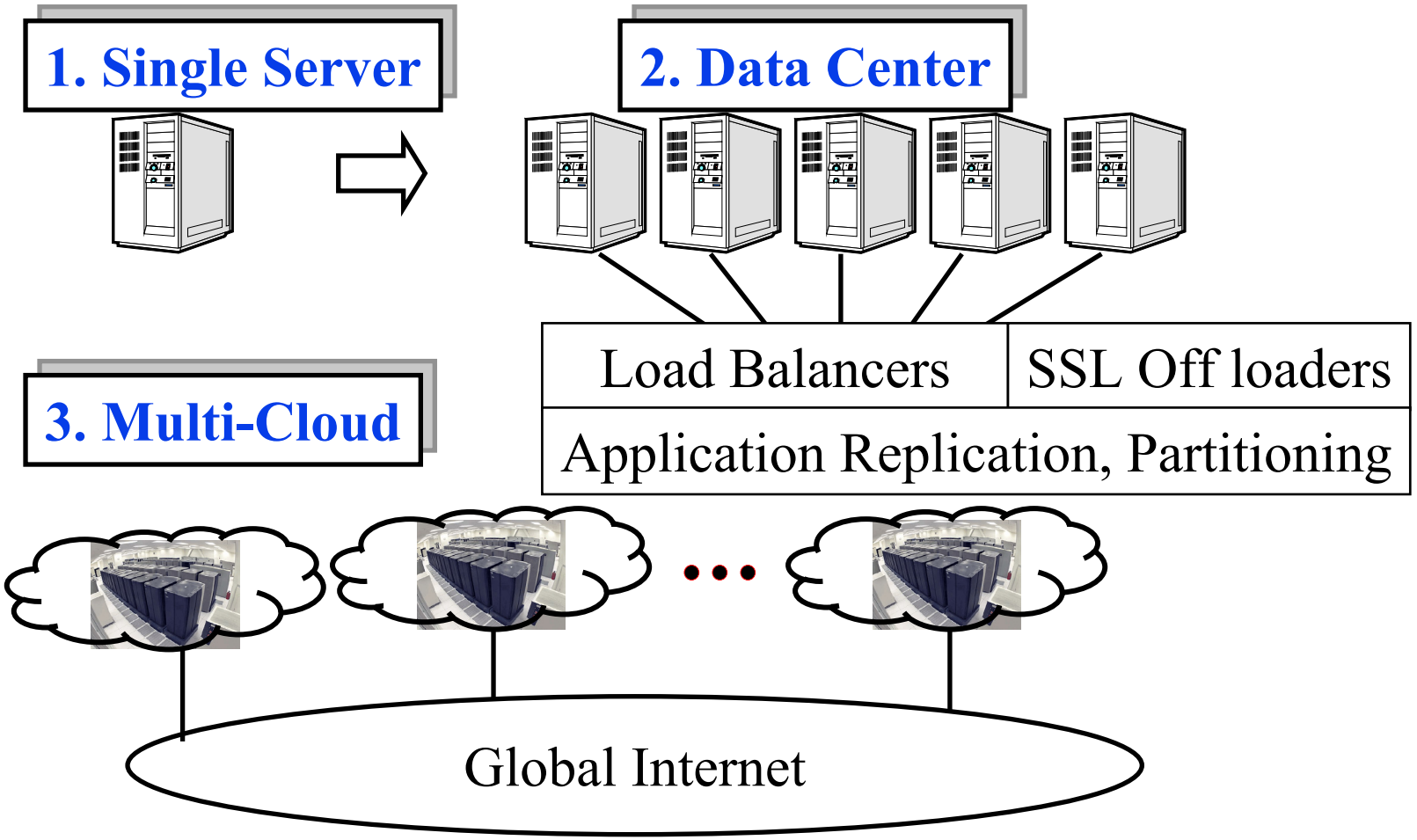
- August 25, 2006: Amazon announced EC2
  ⇒ Birth of Cloud Computing in reality
  (Prior theoretical concepts of computing as a utility)
- *Web Services To Drive Future Growth For Amazon* ($2B in 2012, $7B in 2019)
  - Forbes, Aug 12, 2012

- June 29, 2007: Apple announced iPhone
  ⇒ Birth of Mobile Internet, Mobile Apps
  - Almost all services are now mobile apps: Google, Facebook, Bank of America, …
  - Almost all services need to be global (World is flat)
  - Almost all services use cloud computing

Networks need to support efficient service setup and delivery

©2013 Raj Jain

Service Center Evolution

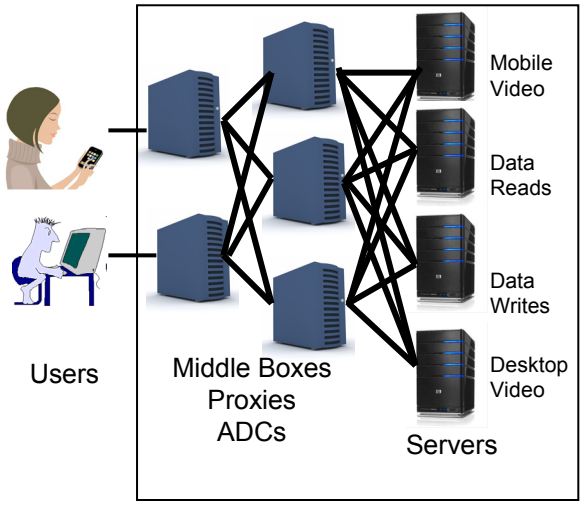


Need to make the global Internet look like a data center

©2013 Raj Jain

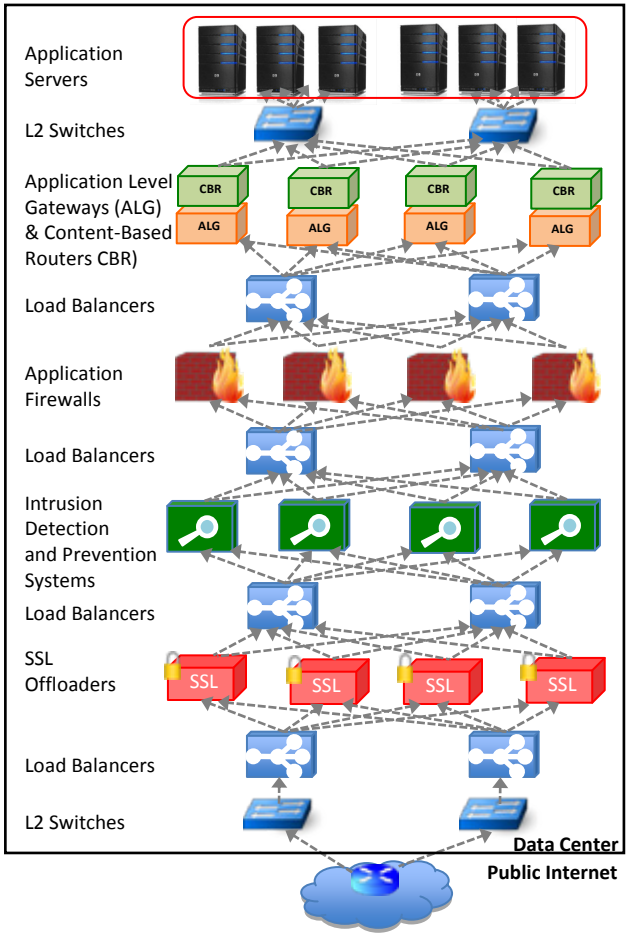
Application Delivery in a Data Center

- **Replication**: Performance and Fault Tolerance
  - ✓ If Load on S1 >0.5, send to S2
  - ✓ If link to US broken, send to UK
- **Content-Based Partitioning**:
  - ➢ Video messages to Server S1
  - ➢ Accounting to Server S2
- **Context Based Partitioning**:
  - ➢ Application Context: Different API calls
    - ✓ Reads to S1, Writes to S2
  - ➢ User Context:
    - ✓ If Windows Phone user, send to S1
    - ✓ If laptop user, send to HD, send to S2
- **Multi-Segment**: User-ISP Proxy-Load Balancer-Firewall-Server



Application Deployment Environment

- Application logic in servers
- Security (firewall, intrusion detection, SSL offload) in middle boxes
- Performance optimization (WAN optimizers, content caches) middleboxes
- Application-level policy routing (APR): Partitioning and replication middleboxes



Middlebox Deployment

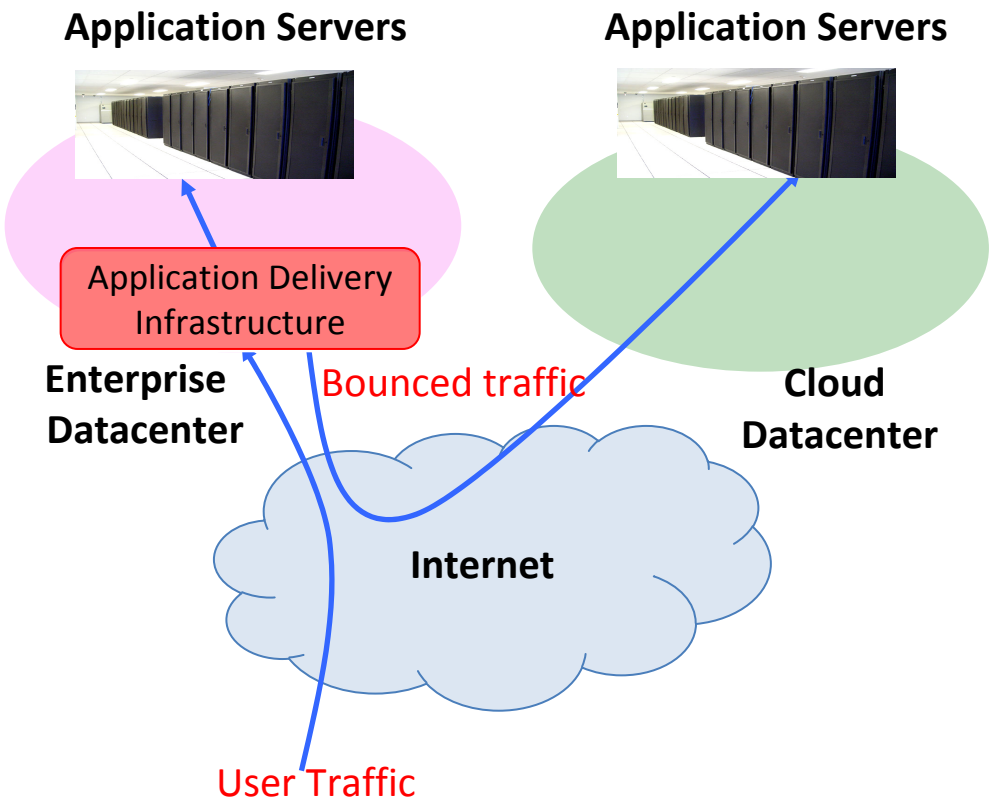
- Number of middleboxes (Application Delivery Controllers) is comparable to the number of routers

Appliance Type	Number
Firewalls	166
NIDS	127
Conferencing/Media Gateways	110
Load Balancers	67
Proxy Caches	66
VPN devices	45
WAN optimizers	44
Voice Gateways	11
Middleboxes total	636
Routers	~ 900

- Market size of optimization ADCs will grow from 1.5B in 2009 to $2.24B in 2013 [17]
- Security appliances will grow from $1.5B in 2010 to $10B in 2016 [13]

©2013 Raj Jain

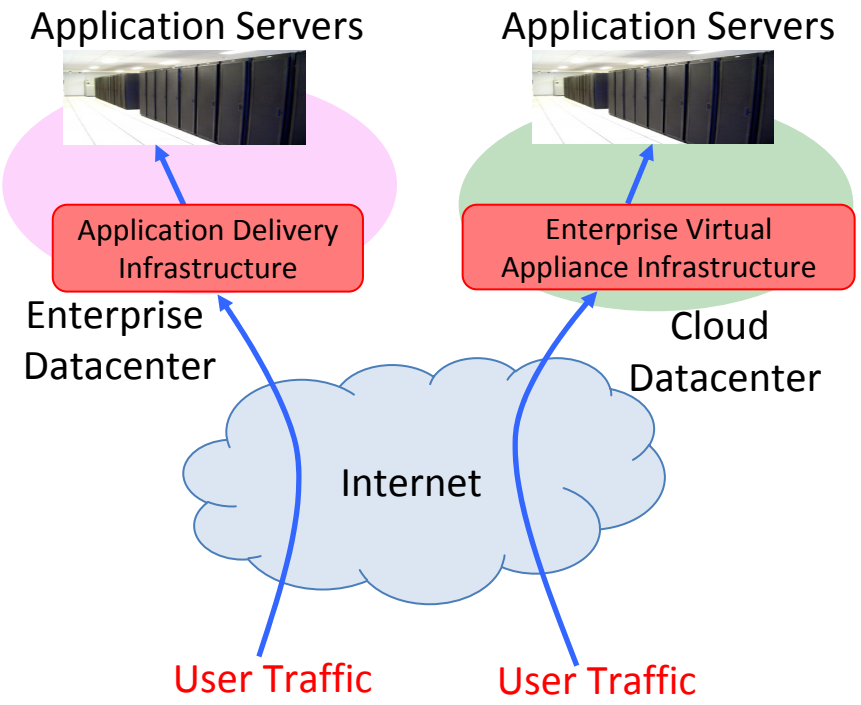
Single-Cloud Failover Deployment



- Under usage spikes and failures, some of the application servers are replicated in cloud. Traffic is bounced through middleboxes in enterprise data centers.

©2013 Raj Jain

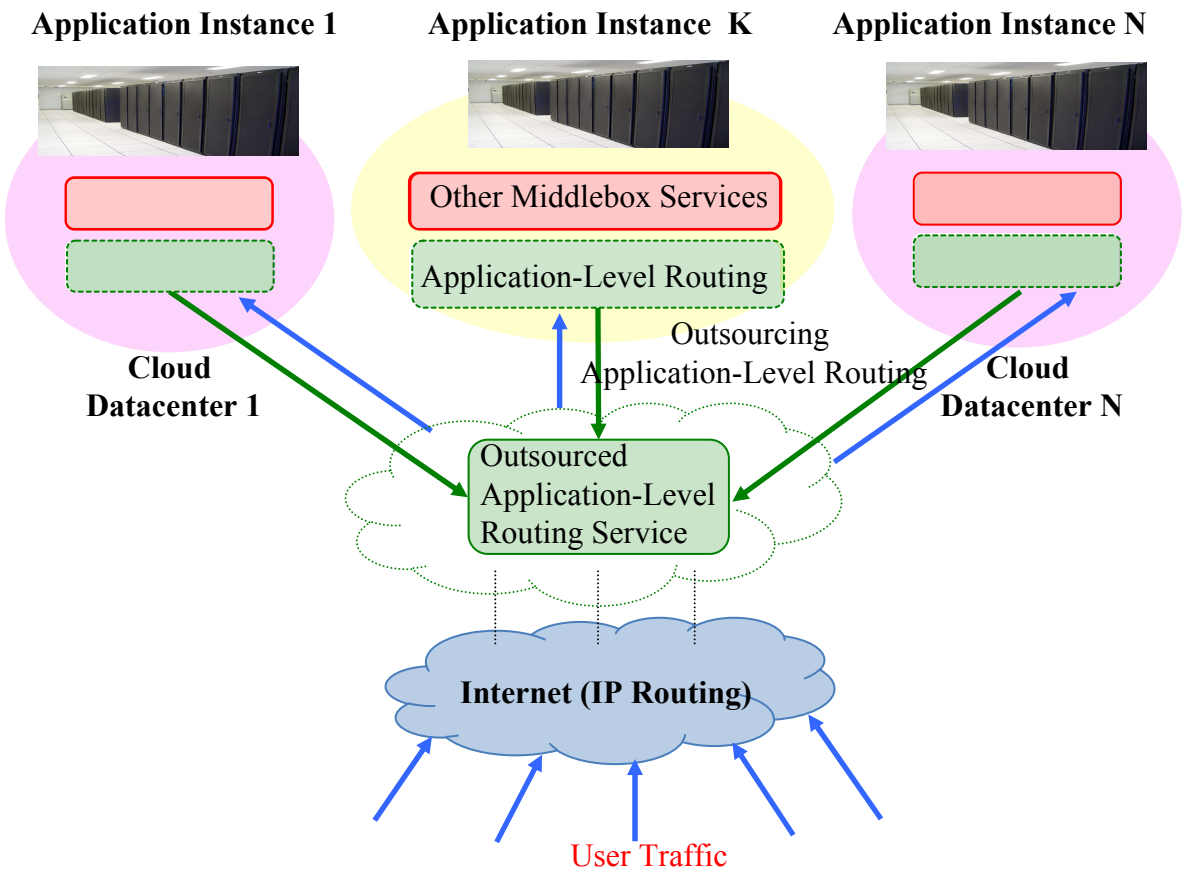
Independent Cloud Deployment



- Virtual appliances are used
- Non-standard techniques (e.g., changing link weights) used to route traffic in datacenters are not available in clouds since networks are not visible to ASPs.

©2013 Raj Jain

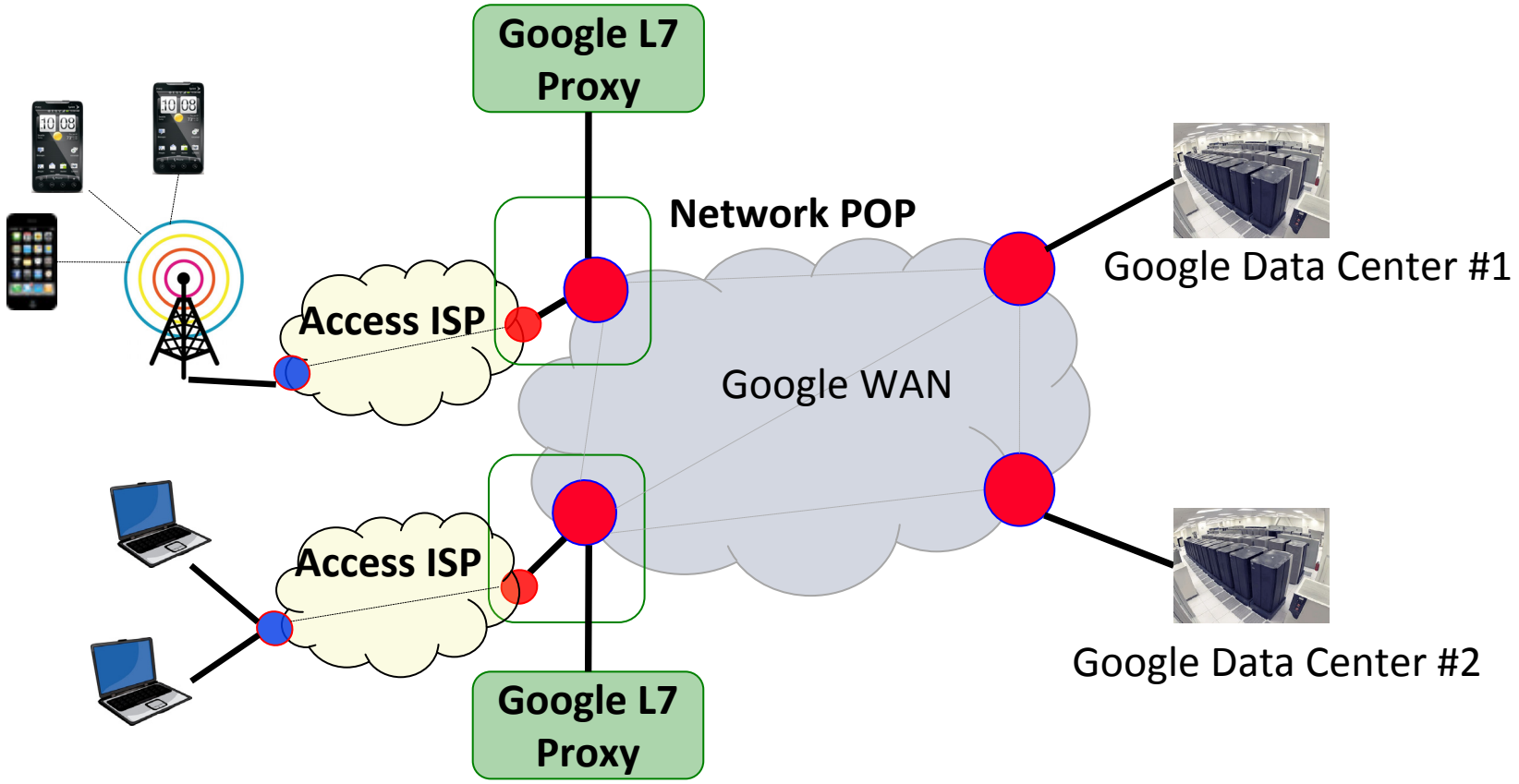
Multi-Cloud Deployments



- Need a globally distributed front-end service is required for application partitioning

©2013 Raj Jain

Google WAN

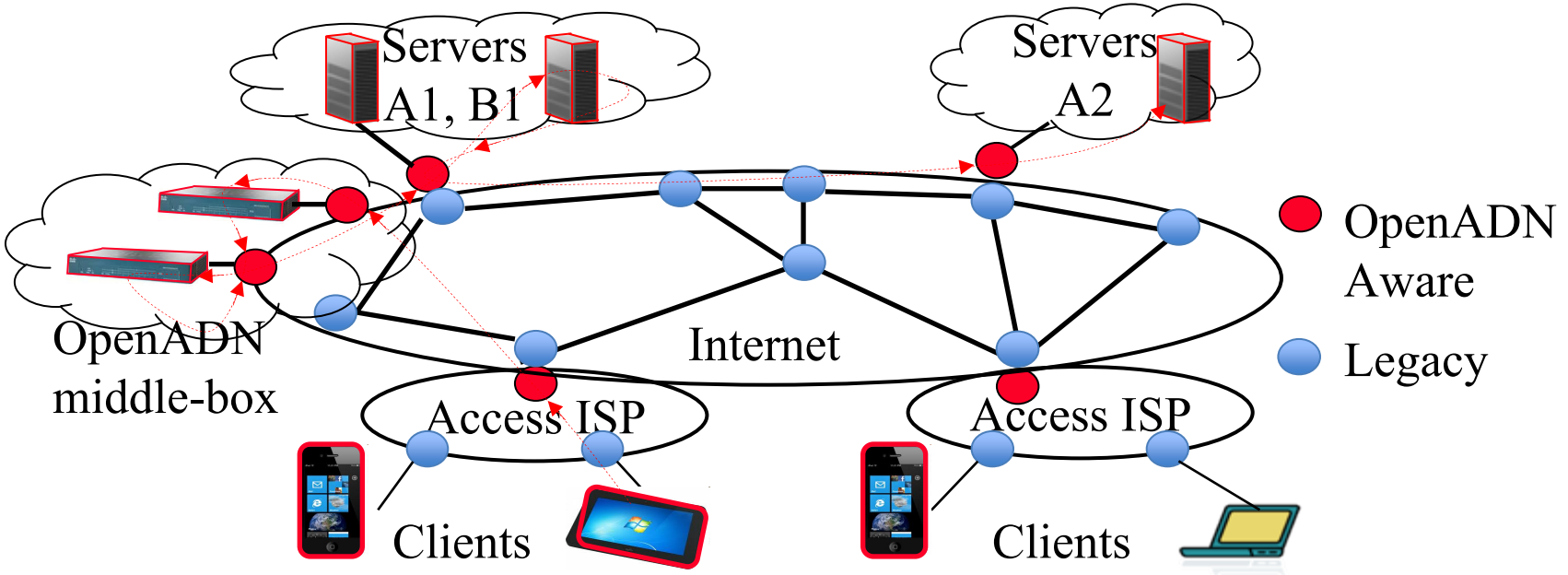


- Google appliances in Tier 3 ISPs
- Details of Google WAN are not public
- ISPs can not use it: L7 proxies require data visibility

©2013 Raj Jain

Our Solution: OpenADN

- Open Application Delivery Networking Platform
  Platform = OpenADN aware clients, servers, switches, and middle-boxes
- Allows Application Service Providers (ASPs) to quickly setup services on Internet using cloud computing⇒ Global datacenter



Design Issues

1. Who will implement? ASP or ISP?
   - Neither Application nor networking ⇒ Middle
   - Application specific but need performance similar to networking
   - ASPs can extend applications
     or ISPs can provide application specific routing
     by providing programmability
2. Middleboxes are deployed in a chain
   - User to SSL offloader to IDS to Firewall to Content based router to load balancer to Application Server
   - Multiple TCP Segments

©2013 Raj Jain

Design Issues (Cont)

3. Each TCP segment ends in a "**Waypoint**"
   - Waypoint = middlebox or server
4. A connection from one waypoint instance to the next waypoint instance is called a "**stream**"
5. Switching context: Application partitioning based on content, application context, networking context, or user context
   - Need to put meta-tags in the header that help waypoints correctly route the packets
6. Sender and Receiver Policies: Receivers may be services.

©2013 Raj Jain

Design Issues (Cont)

7. Data Privacy: Need a way for ISPs to implement this without looking at the data
8. Dynamic Application Deployment State: ISPs need to know where and how many waypoints are up

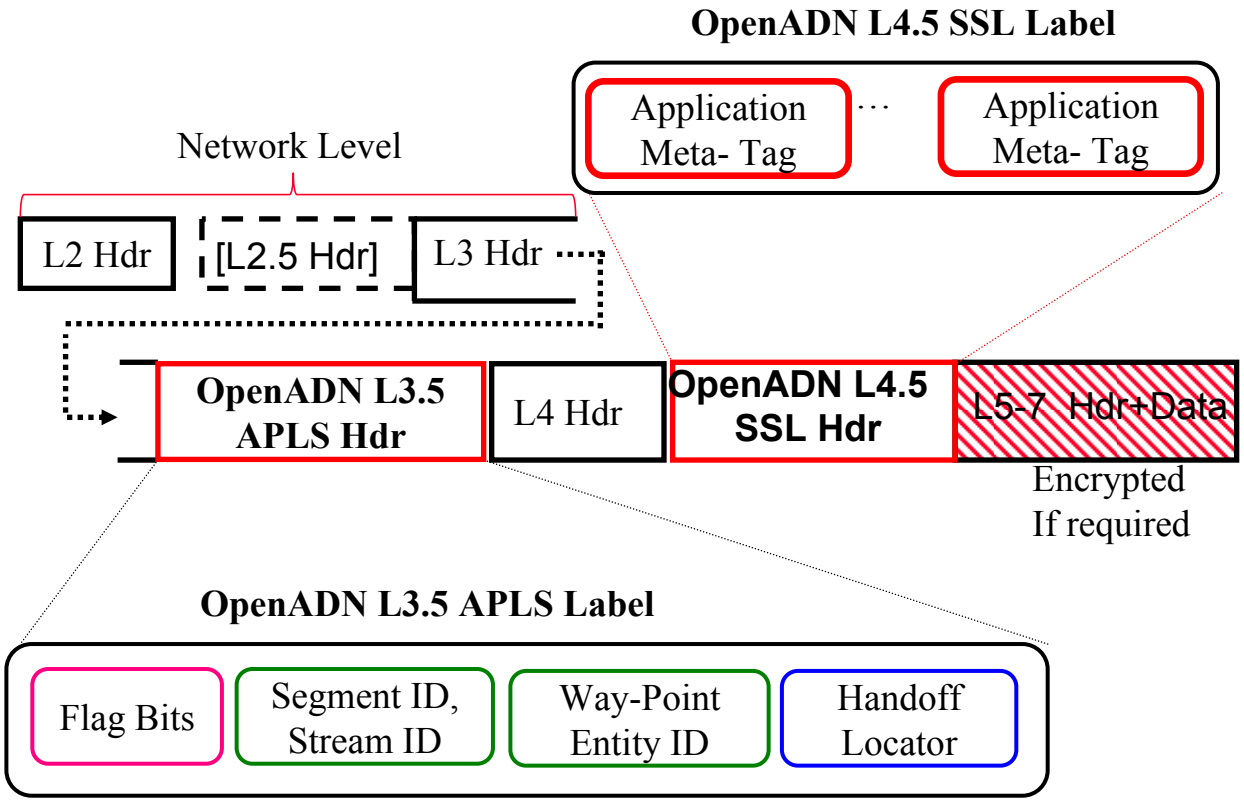
©2013 Raj Jain

Design Approach

1. Application Delivery Networking (ADN) layer between the networking and higher layers
2. The packets require classification and routing based on content
3. Classification is done in ASP trusted entity since it needs access to data and encoded in a meta-tag

©2013 Raj Jain

OpenADN Label



- Layer 3.5: Application Label Switching (APLS)
  Hop-by-hop transport between waypoints in a segment
- Layer 4.5: Segment Switching Layer (SSL)
  Between application segments

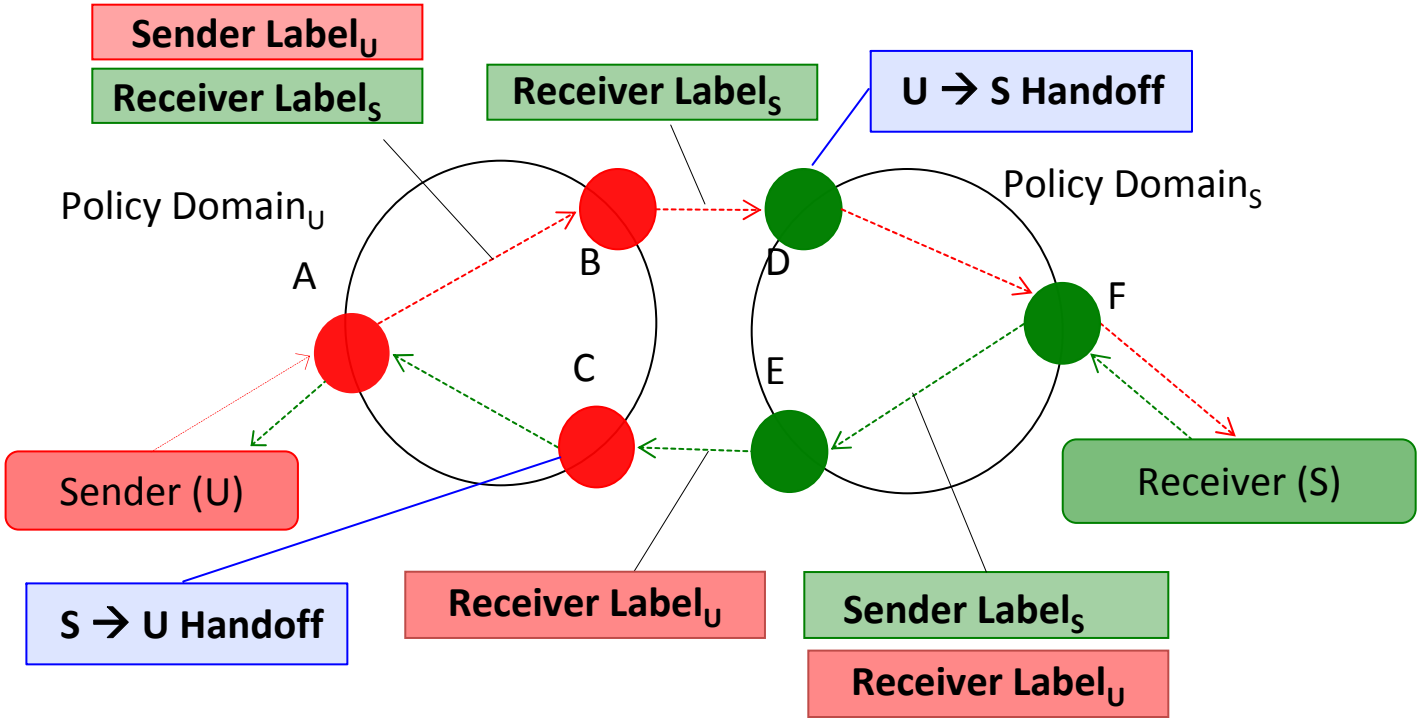
©2013 Raj Jain

OpenADN Labels (Cont)

- Layer 4.5 Label: Stack of meta-tags – one for each segment. At the egress of a segment, a meta-tag is popped and used during the next segment
- Layer 3.5 Label:
  - <Segment ID, Stream ID>: Specific instance of an application segment
  - Waypoint ID: Previous or next Waypoint (as indicated by Flag bits)
  - Handoff Locator: Middlebox copies this to the destination IP address. Helps switch the packet to the next OpenADN switch

©2013 Raj Jain

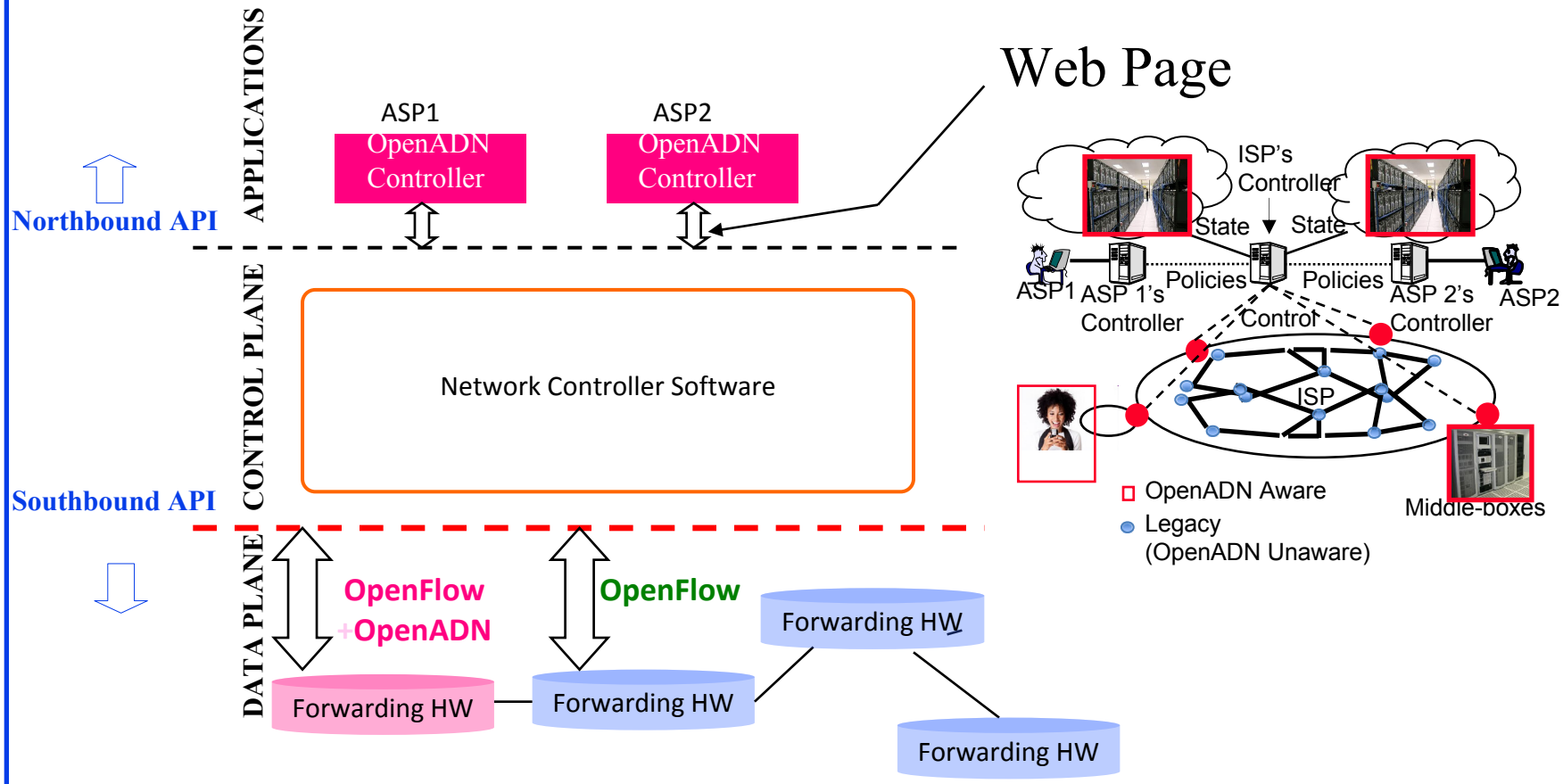
Sender and Receiver Policies



- Each packet has two labels: Sender Label, Receiver Label
- Sender label is popped at egress of sender domain and packet is sent to the ingress of the receiver domain

©2013 Raj Jain

OpenADN in SDN's Layered Abstractions



- SDN provides standardized mechanisms for distribution of control information
- OpenADN aware devices use enhanced OpenFlow

©2013 Raj Jain

Key Features of OpenADN

1. Edge devices only.
   Core network can be current TCP/IP based, OpenFlow or future SDN based
2. Coexistence (Backward compatibility):
   Old on New. New on Old
3. Incremental Deployment
4. Economic Incentive for first adopters
5. Resource owners (ISPs) keep complete control over their resources

Summary

1. Application delivery requires multiple segments between numerous middleboxes that are handled in an ad-hoc manner in datacenters
2. Distributing applications over a multi-cloud environment requires collaboration between ASPs and ISPs
3. OpenADN provides allows ISPs to provide application delivery and partitioning services without looking at the application data
4. Both ASPs and ISPs keep complete control over their resources by co-ordinating in the control plane using SDN.

©2013 Raj Jain