Cisco live!

# What You Make Possible

TOMORROW
starts here.

CISCO

# Deploying Carrier Grade IPv6 using CGSE

BPKSPG-2604

TOMORROW starts here.

# Agenda

- CGv6 Overview
- Introduction to the CGSE
- Configurations
  - NAT44
  - NAT64
  - 6rd
  - DS-Lite
- Deployment Options
- IPv6 Economics
- Q & A

# CGv6 Overview

# Terminology

- CGv6 - Carrier Grade IPv6.
  - Cisco's IPv6 Transition package.
- CGSE - Carrier Grade Service Engine.
  - A service PLIM for the CRS-1/3, provides IPv6 Transition solutions.
- CGN - Carrier Grade NAT.
  - High performance & scale NAT44 for service providers.
- NAT - Network Address Translation. (Historically stateless and 1:1)
- NAPT/PAT – Network Address Port Translation, Port Address Translation. (NAT or NAT44 in this session means NAPT or PAT.)
- NAT-PT – NAT Protocol Translation, the original (and now deprecated way) to do IPv6 to IPv4 translation. Now NAT64

# CGv6 Terminology

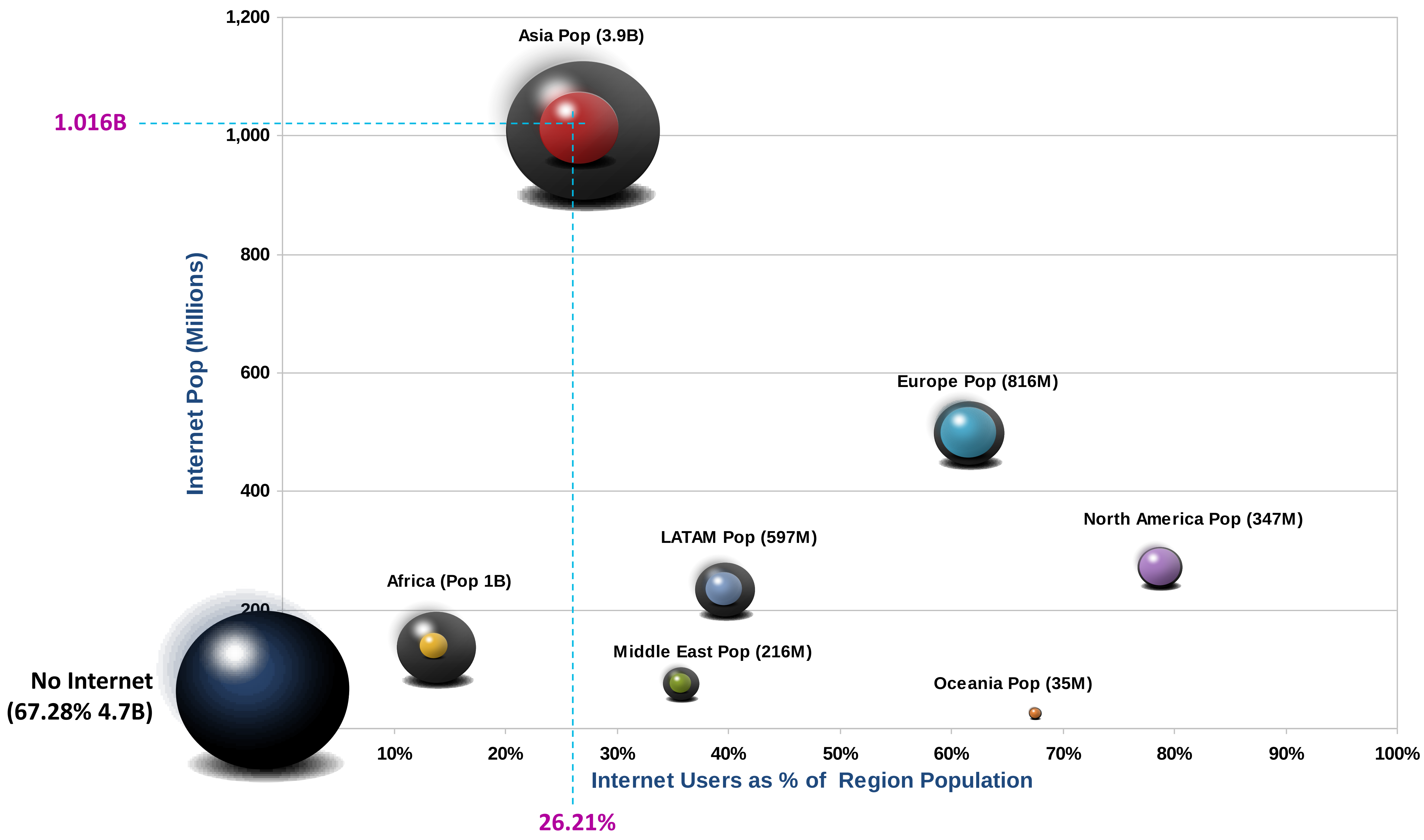
- Stateless NAT64
  Translation from IPv6 to IPv4 in a stateless manner, 1:1 (NAT)
- Stateful NAT64
  Translation from IPv6 to IPv4 in a stateful manner, N:1 (NAPT)
- 6rd
  IPv6 rapid deployment, provide Dual Stack over IPv4 only access
- DS-Lite
  Dual Stack Lite, provide Dual Stack over IPv6 only access
  - Watch for MAP-E and MAP-T IDs emerging from IETF and support on CGSE

# Why IPv6 ?

## You have heard it all before

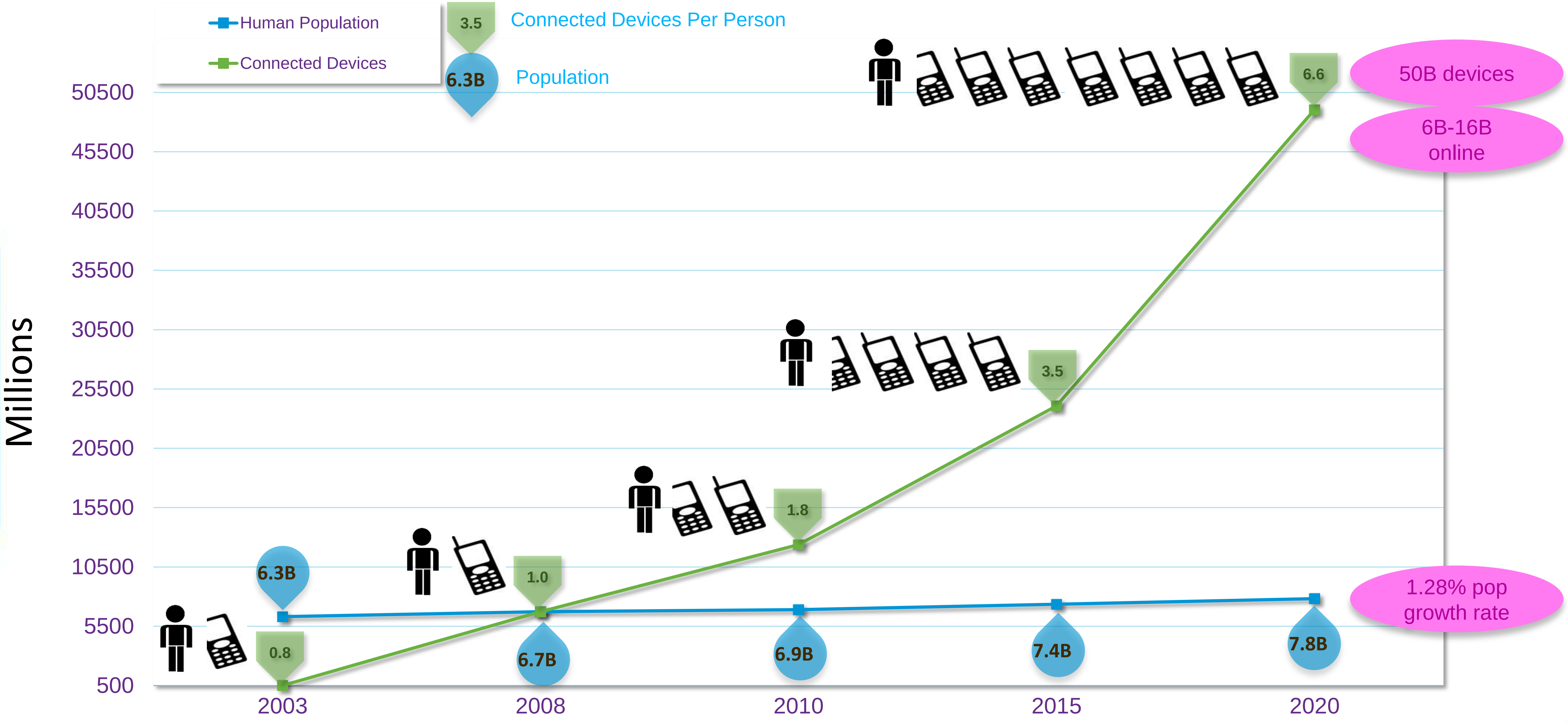
- IANA and the RIRs have run out of IPv4 address
- There is ongoing growth
- Consumers are generally ambivalent
  - Do not/should not care whether IPv4 or IPv6 broadband delivery
- IPv4 address trading markets are starting to appear
  - Growth, fragmentation, and identity verification of the IPv4 routing table is inevitable

# Internet Usage By World Region




Source: http://www.internetworldstats.com/stats.htm Dec 2011

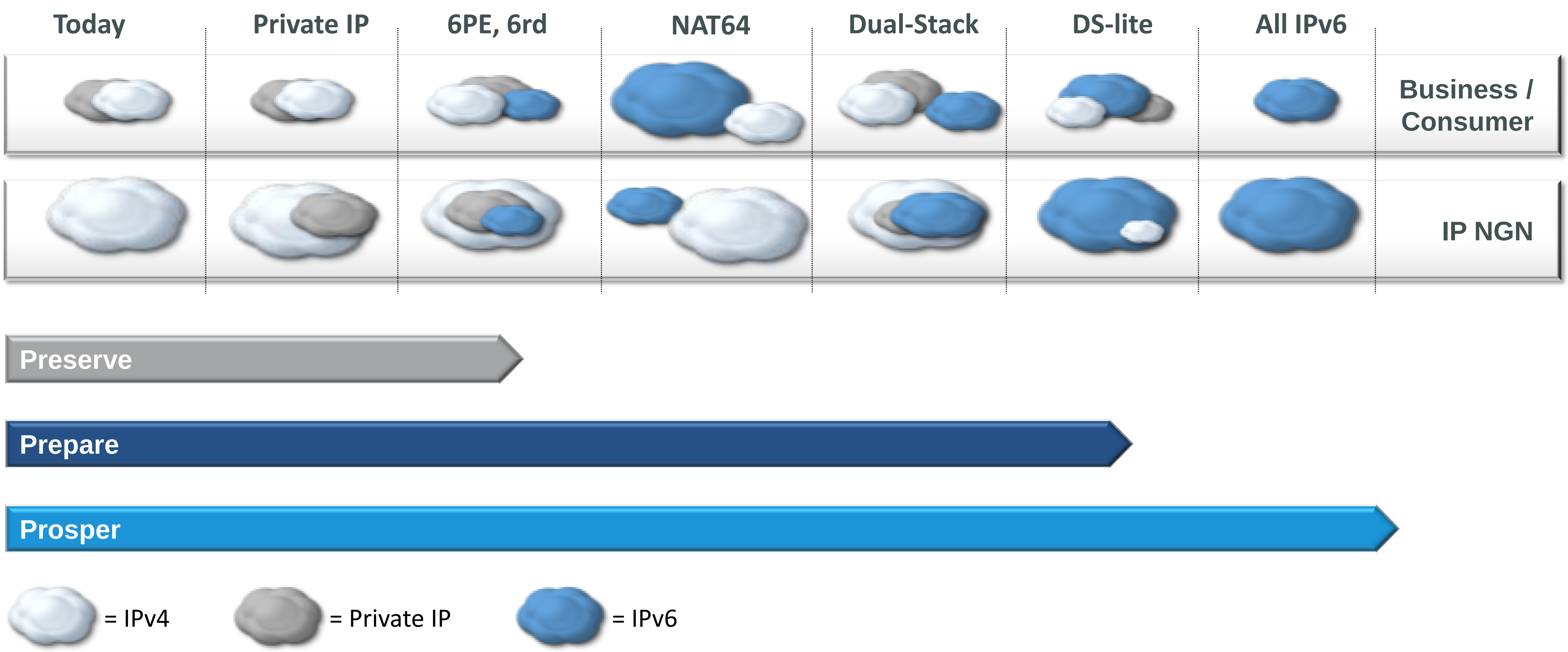
# Internet Usage By Device




Source: Cisco IBSG projections
UN Economic & Social Affairs http://www.un.org/esa/population/publications/longrange2/WorldPop2300final.pdf

# CGv6 Framework – 3 Tiered Approach

## Enabling an Orderly, Incremental Transition

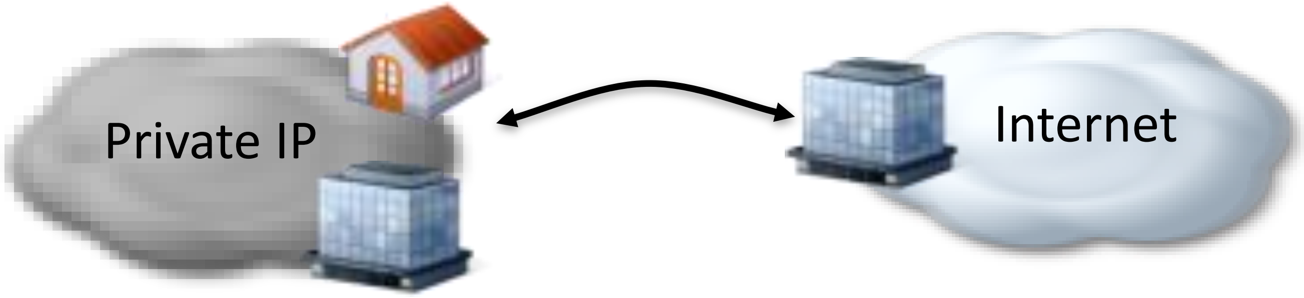



http://www.cisco.com/go/cgv6/

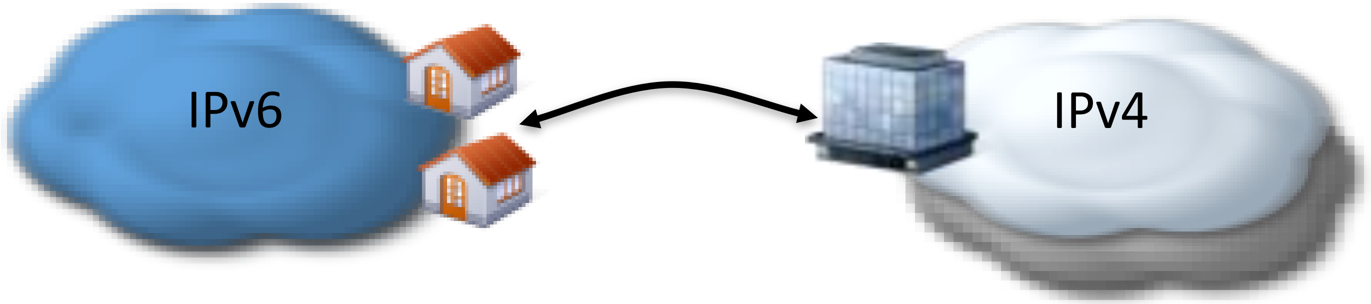
# CGv6 Framework – 3 Tiered Approach

## Translation & Tunnelling

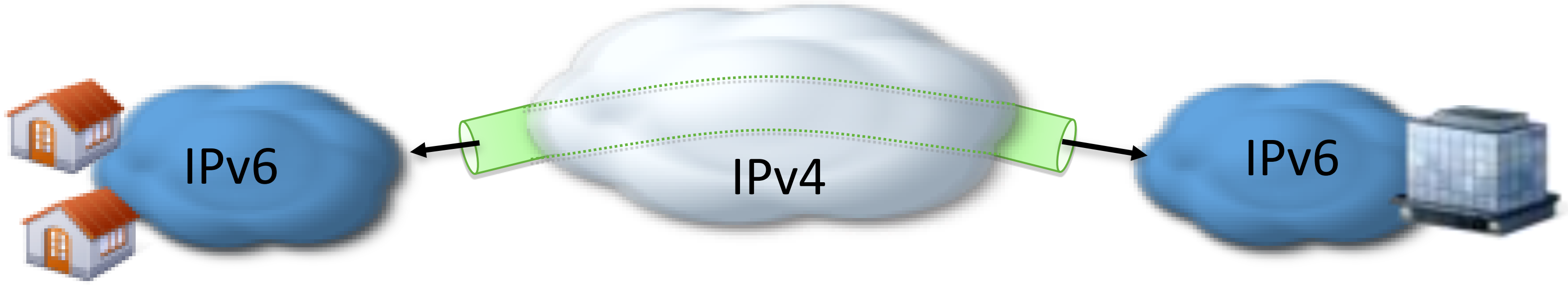
Carrier Grade IPv4 NAT




IPv6/IPv4 Translation, BEHAVE working group

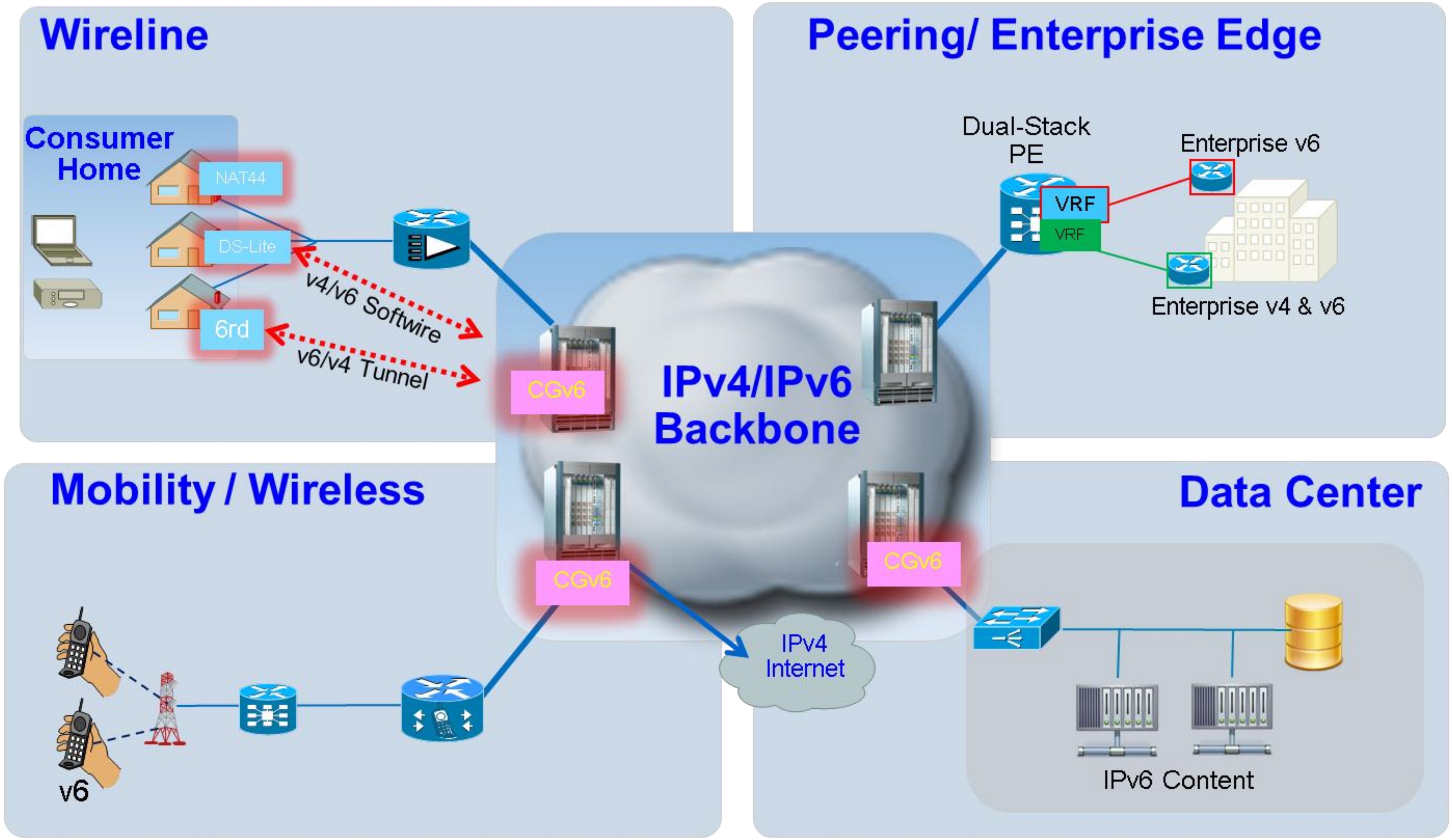



IPv6 over IPv4, and IPv4 over IPv6, Softwire Working Group



# IPv6 "Backbone-First" Solutions

## CGv6 Extends IPv6 Connectivity and Services



# Introduction to the CGSE

# Carrier Grade Services Engine (CGSE)

## Introduction

Cisco CGSE

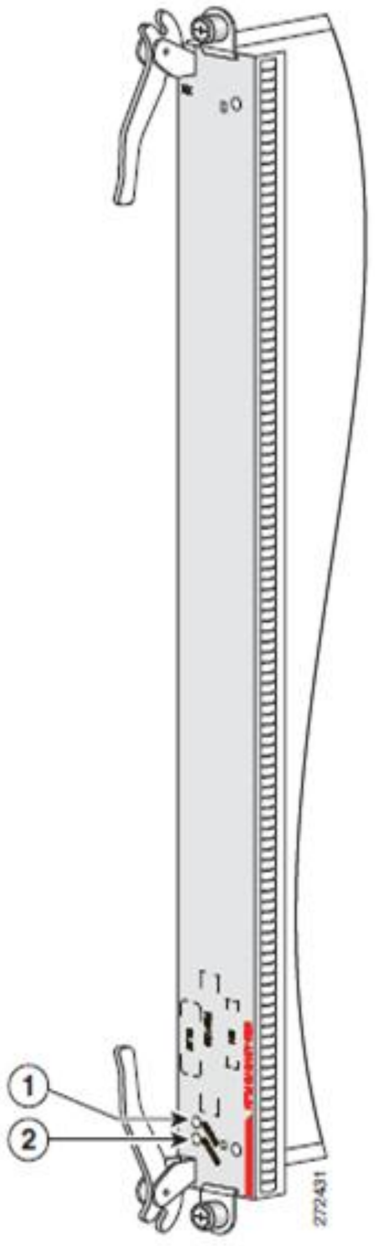
- **CGv6: Translation** (NAT44, NAT64)
  **Tunnelling** (6rd, DS-Lite)
- **20+ million** active translations
- **100s of thousands** of subscribers
- **1+ million** connections per second
- **20Gb/s** of throughput **per CGSE**

- Builds upon the proven performance of the Cisco CRS platform
- High-capacity, carrier-class SP platform with Cisco IOS-XR

Cisco CRS

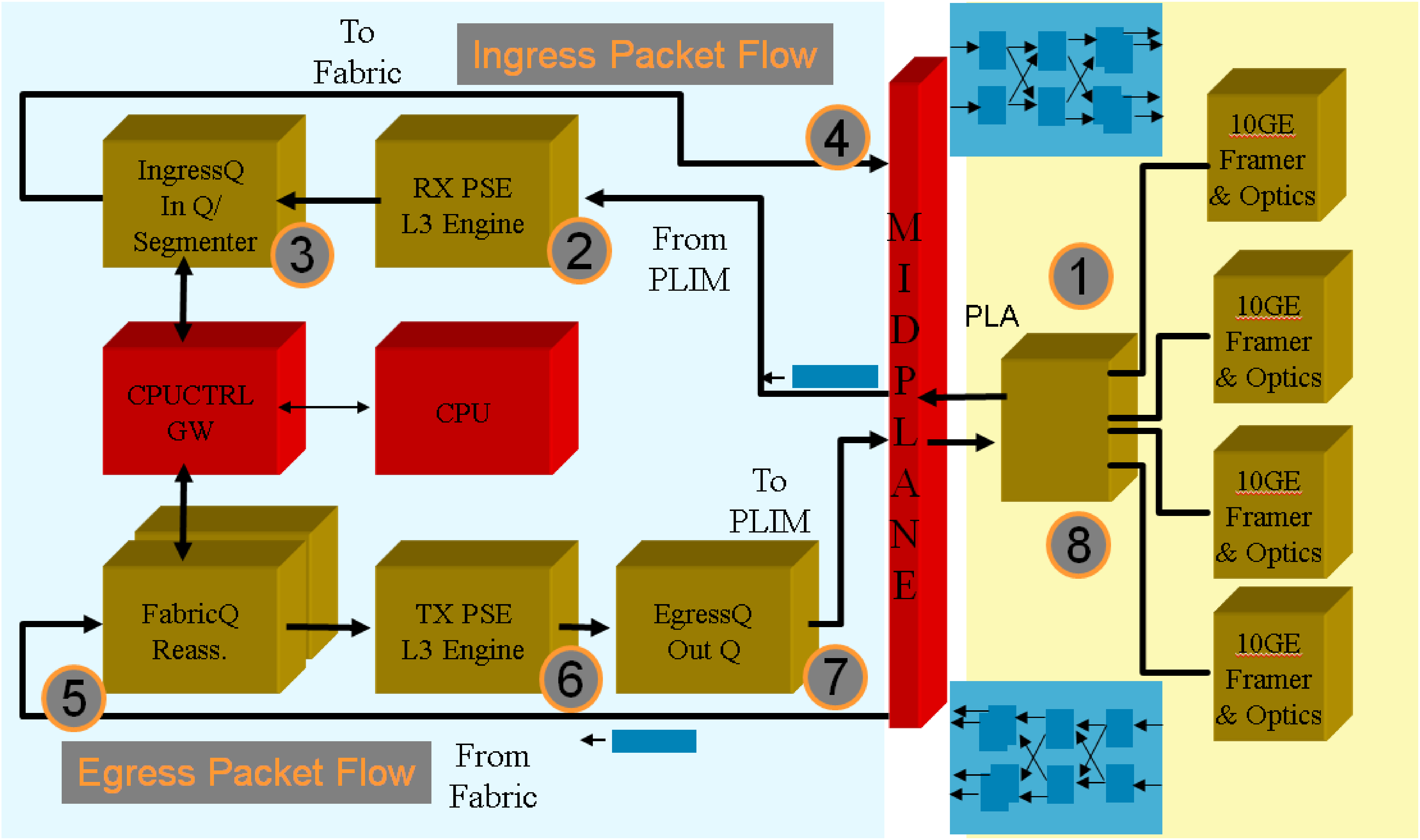
# CGSE Overview

- CGv6 function resides on CGSE PLIM
- Paired with CRS-MSC-40G-B, CRS-MSC-20G-B, CRS-MSC and FP-40 (IOS-XR 4.1.1 onwards),
- Does not support pairing  with MSC-140, or FP-140
- No external interfaces
- Four 16-core Octeon MIPs CPUs, 64 CPU cores
- Standard interface to MSC, 20 Gbps of throughput (per CGSE)
- IOS XR on MSC, Linux on Octeon CPUs
- Thermal Restrictions for CGSE in CRS-1 16 slot
  - Upper Bay: CGSE; Lower Bay: anything (including another CGSE)
  - Lower Bay: CGSE; Upper bay: CGSE only




| | |
|---|---|
| 1 | ActiveLED |
| 2 | Status LED |

# CRS Overview







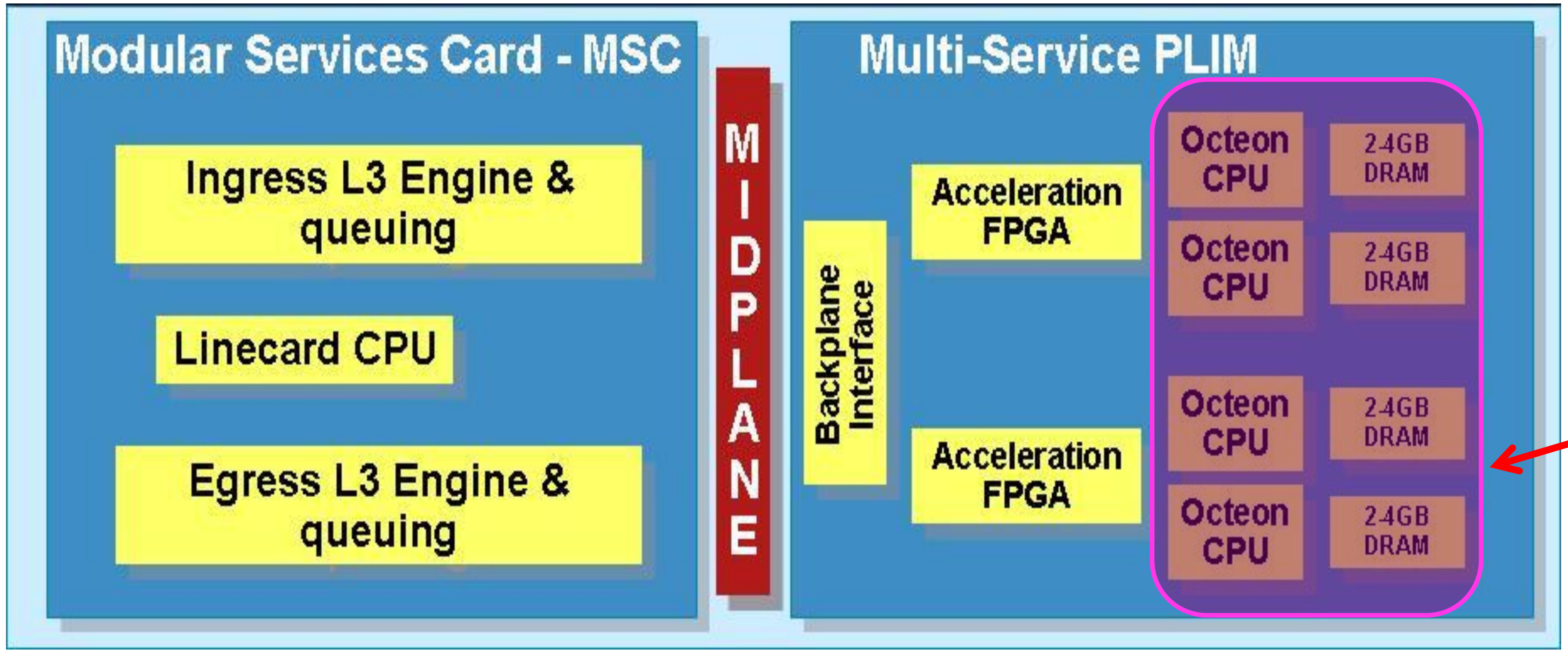
Cisco live!

# CRS Terminology

- PLIM
  - Physical Layer Interface Module – The ‘interface’ portion of a Linecard, paired with an MSC or FP
- PLA
  - PLIM ASIC
- PSE
  - The Packet Switching Engine (also referred to as ‘Metro’) is the primary L3 feature processing ASIC
- MSC
  - Modular Services Card
- FP(40)
  - Forwarding Processor
- PIE
  - Package Installation Envelope. An installable software file

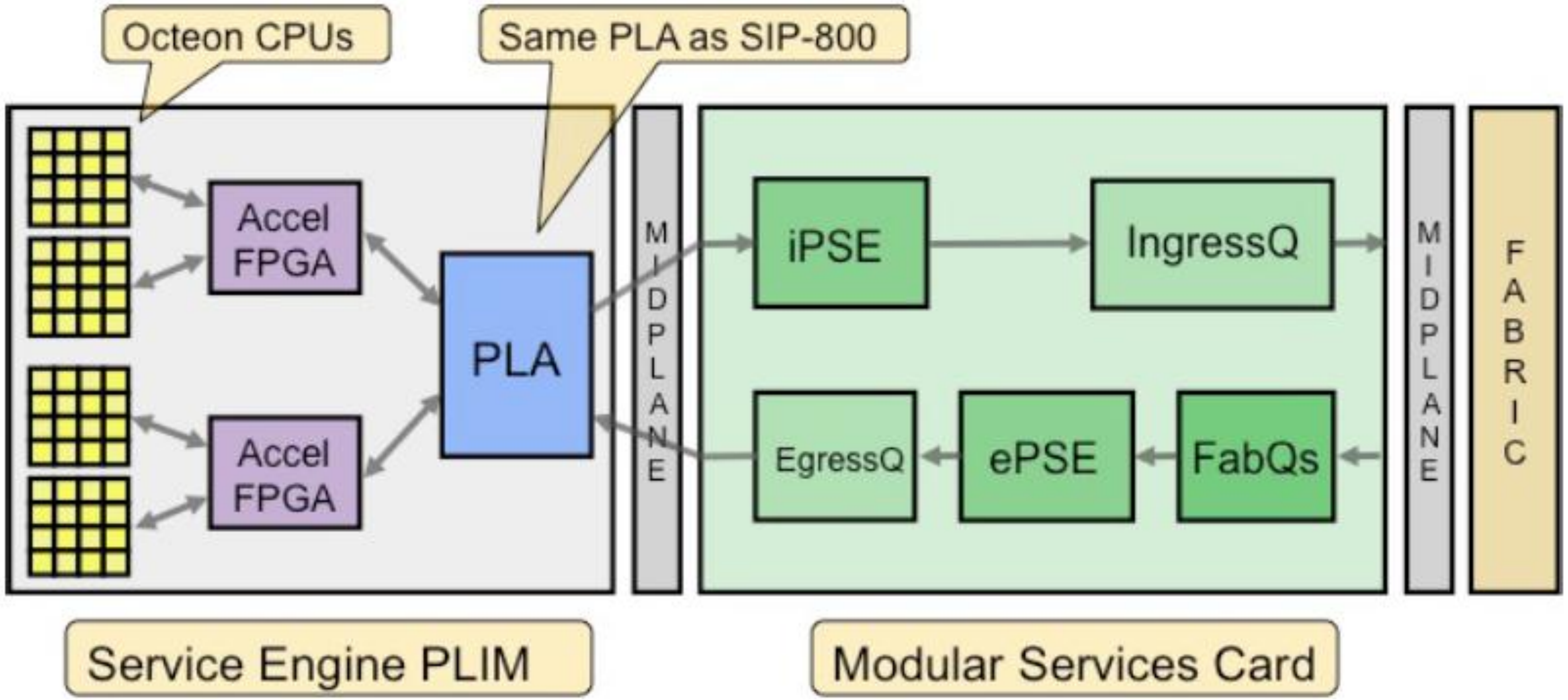
# CRS CGSE Hardware Overview

- Multi-CPU Processor bank, 4 Cavium Octeaon CPU Complexes
- 16x Cores Per CPU Complex, 64x Total CPU Cores
- 2-8 Gig Memroy for each Octeon complex
- 20G bandwidth between CGSE and CRS-1

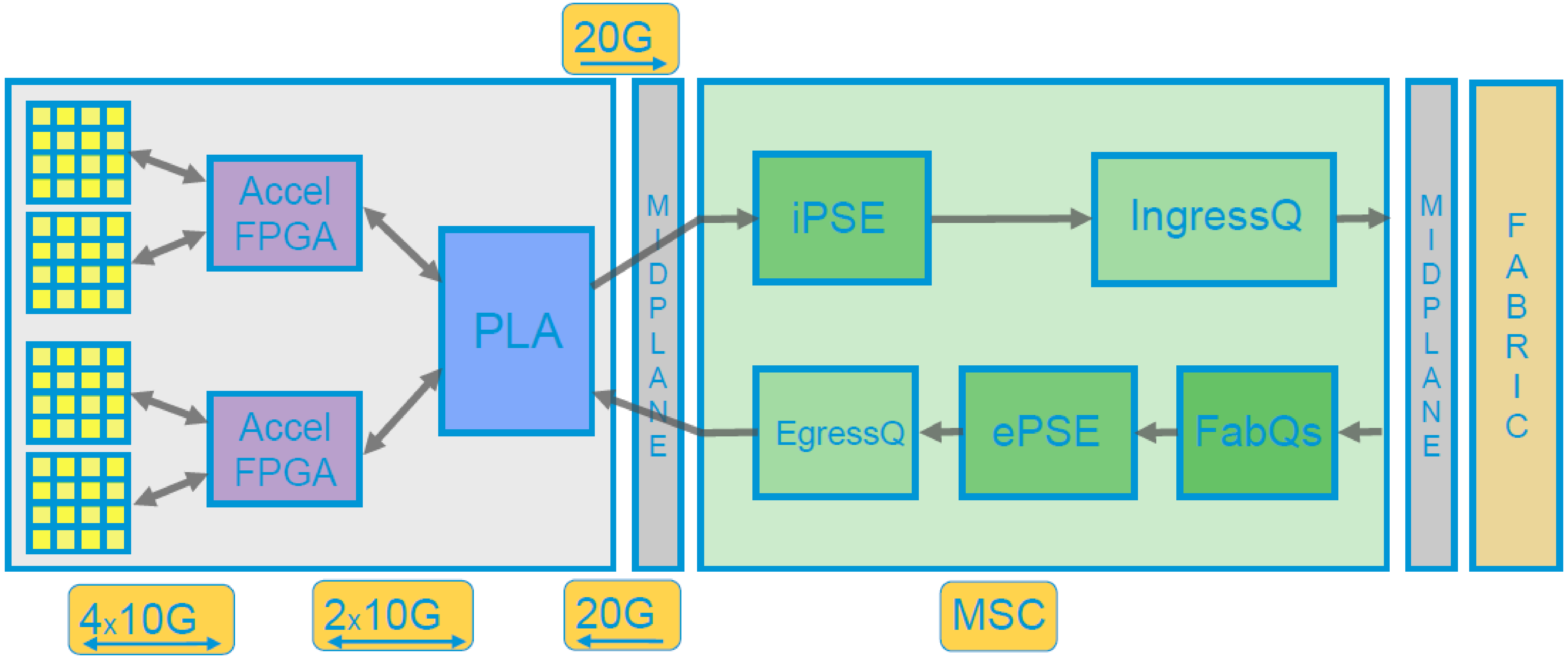



CGN Service Types run here

# CRS-CGSE-PLIM / CRS-MSC Architecture



# CRS-CGSE-PLIM / CRS-MSC Bandwidth



# CGSE Infrastructure for CGv6 Apps

## Application Support

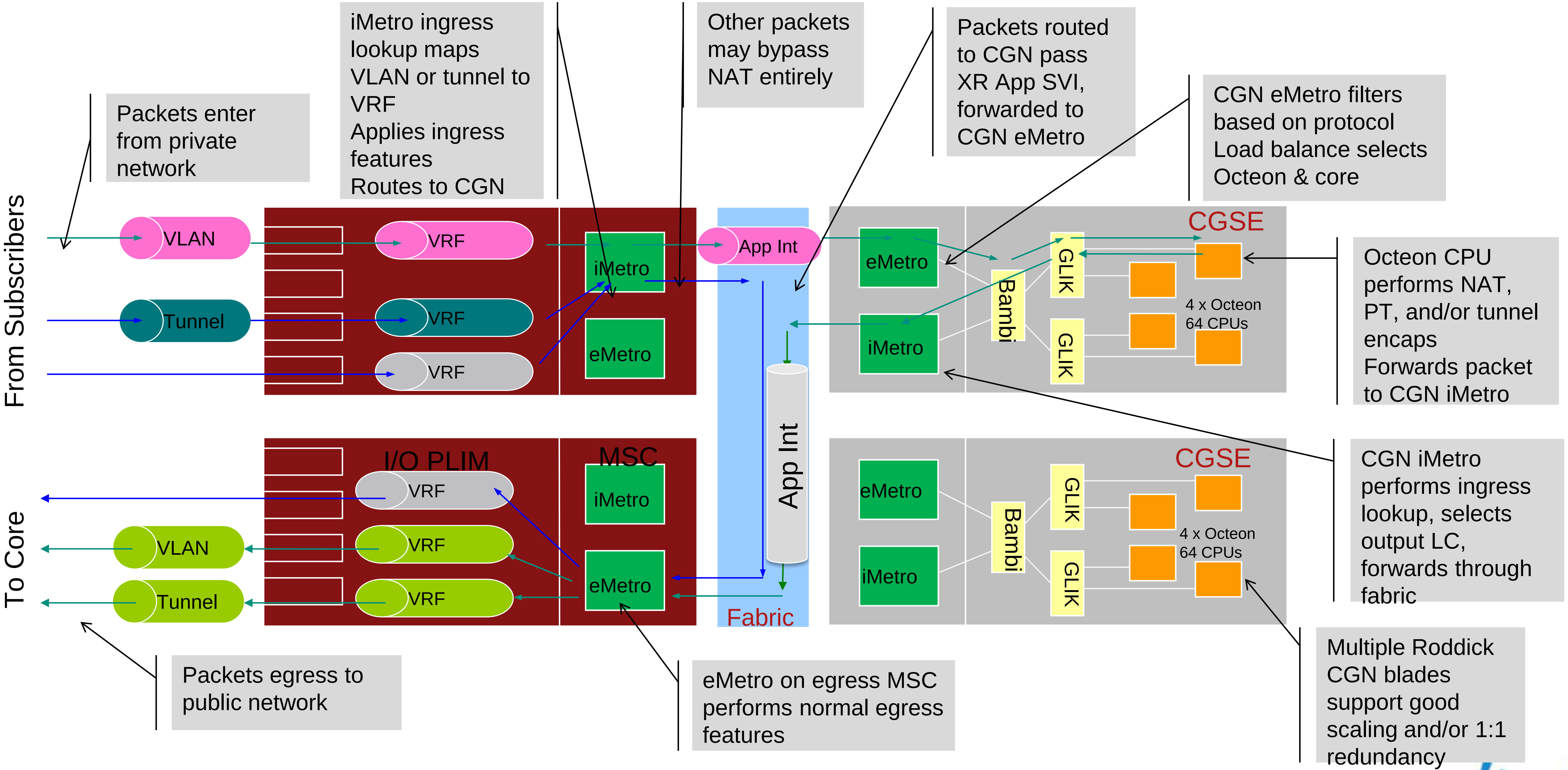
- Infrastructure allows multiple CGv6 applications to co-exist
  - NAT44, NAT64 Stateless/Stateful, 6rd etc.
- Logging support for every translation
- All 64 cores run one instance of the application
- Egress Forwarding engine (eMetro) does classification & load balancing
- Integrated IOS-XR CLI to configure CGv6 applications

# CGSE Infrastructure for CGv6 Apps

## Application Redundancy Support

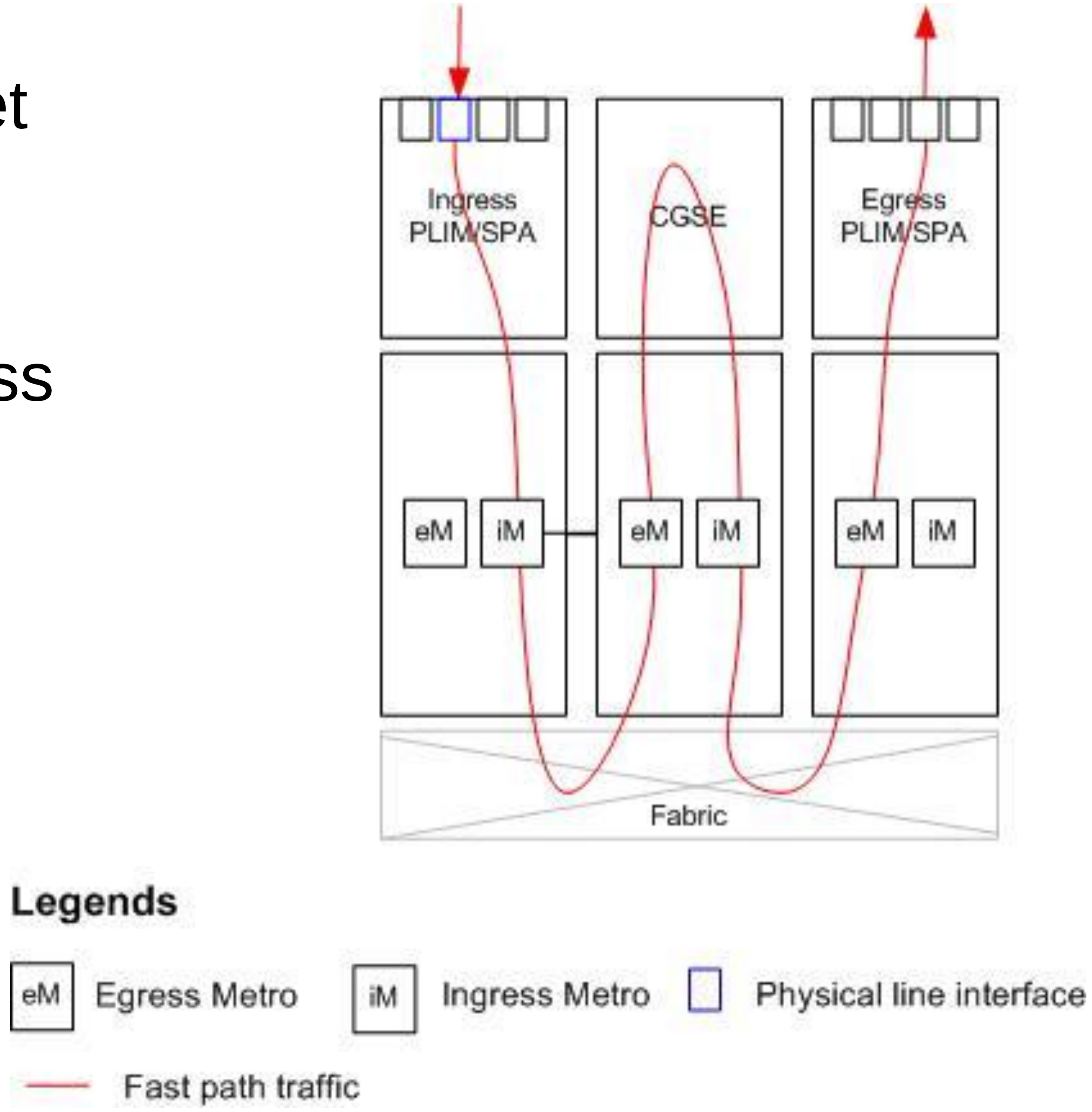
- Application failures are logged and core dumps saved in RP hard disk
- Data path, control path, core to core online diagnostics
- Active / Standby CGSE table sync-ups for warm standby
- Failure detection and switchover to standby (max 1s traffic loss)
- Debugging support : ltrace, counters, packet dumps, per core debugging, show and debug commands

# CGN Packet Walk-Through



# Packet Path

- Data Plane packet path for packet using CGSE service
- For ingress traffic ServiceApp interfaces are bound to the Ingress PLIM/SPAs iMetro
- For egress traffic ServiceApp interfaces push into Fabric



# CGv6 on CGSE

- SP-Class Performance/Scale
  - 20M Translations (combined)
  - 1M connection setups/sec
  - 10G full-duplex performance
  - <=250 microseconds latency
- NAT behaviour compliance
  - RFC4787, RFC5382, RFC5508
- NAT64 compliance
  - RFC6052, RFC6145, RFC6146
- 6rd compliance
  - RFC5969
- DS-Lite compliance
  - RFC6333

- CGv6 Bypass
- Netflow9 logging
- Syslog logging
- VRF based traffic diversion
- Static Port Forwarding
- TCP/UDP/ICMP Timers
- 1 + 1 Warm Standby
- Active FTP ALG
- Lawful Intercept

# CGSE Configuration

# Bring up the CGSE Board

- CGN PIE for version of IOS-XR needs to be installed

```
router(admin)#sh install active summary
Active Packages:
 disk0:hfr-cgn-3.9.3
```

- Control connections to the CGSE are via a single ServiceInfra Interface (per CGSE) & an IPv4 address of local significance.

```
router(config)#
interface ServiceInfra1
 ipv4 address 4.3.2.1 255.255.255.252
 service-location 0/2/CPU0
```

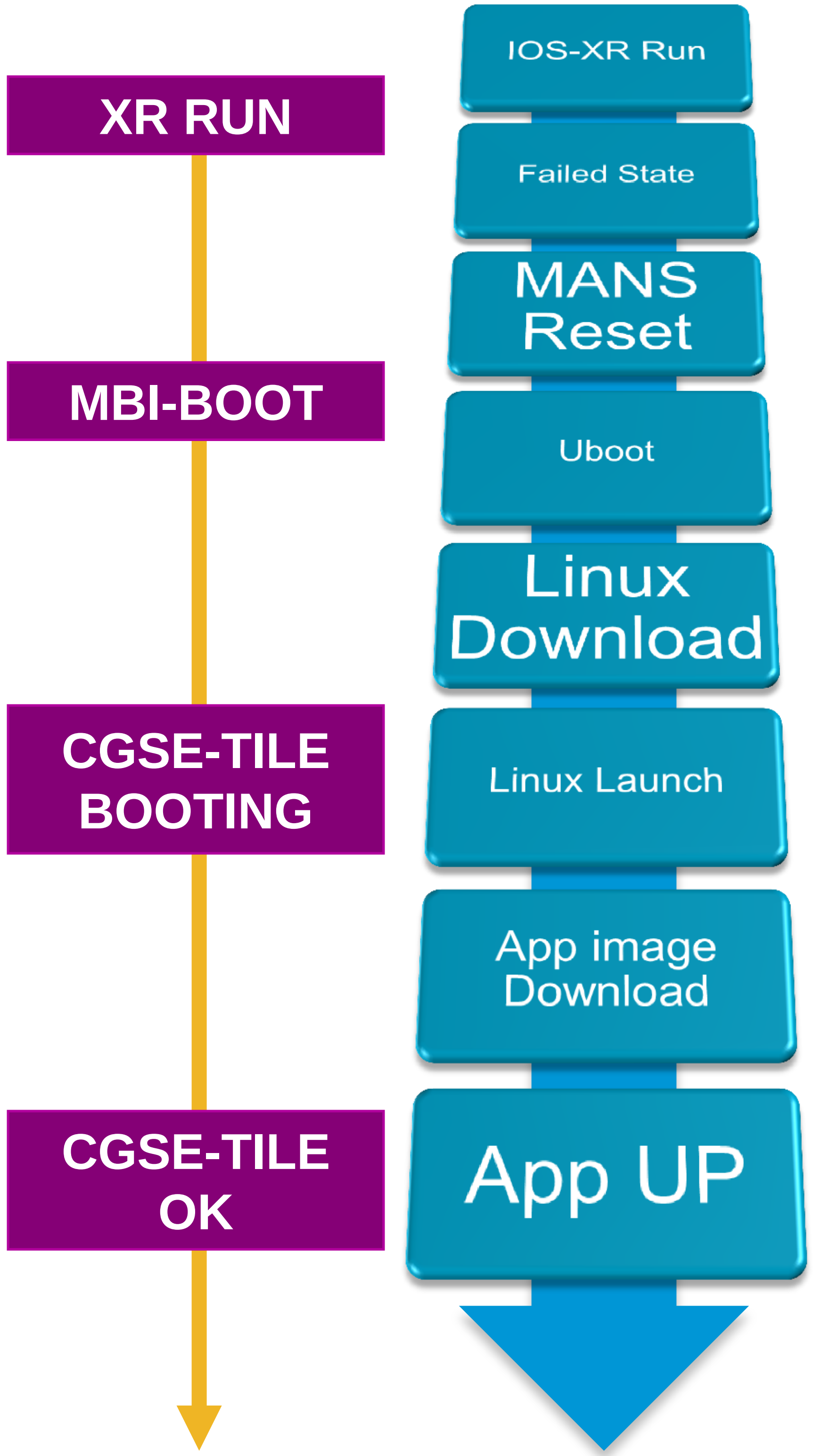
- Specify the service role ( cgn ) for the given CGSE location

```
router(config)#
hw-module service cgn location 0/2/CPU0
```

- You need to reload the card. It may take ~15min

```
router#
hw-module location 0/2/CPU0 reload
WARNING: This will take the requested node out of service.
Do you wish to continue?[confirm(y/n)] y
```

# CGSE Boot Process




Service/ CGN Pie not installed

Service/ CGN Pie installed without role config

Takes Master Octeon out of Reset

Sends Doorbell to indicate bootloader downloaded (Successful Uboot)

Linux Download will start and boot params

Linux launch happens on master octeon which downloads linux on Slave

Linux UP Doorbell , App image gets downloaded via TFTP and launched

- PLIM Services process monitors various boot stages.
  - Packaged with ‘comp-hfr-mini.vm’
- 3 retries after which card will put into Failed State

# Service Instance Configuration

- Service Instance is the highest level configuration structure
  – Represents the CGSE card or primary/backup CGSE pair
  – Common redundancy model is 1:1 warm standby
  – One ServiceInfra interface per Service Instance – control path

```
service cgn cgn1
 service-location preferred-active 0/X/CPU0 preferred-standby 0/Y/CPU0
```

- "Service-Type" specific instance is the child structure

  Includes specific configuration for apps running within Service Instance

  Service Types supported *(NAT44, NAT64, 6rd BR & DS-Lite)*

```
service cgn cgn1
 service-type nat64 [stateless|stateful] nat64-xx
   (SL-NAT64 specific config)
 service-type nat44 nat44
   (NAT44 specific config)
 service-type tunnel v6rd 6rd
   (6rd specific config)
 service-type ds-lite ds-l
   (6rd specific config)
```

**XX** = free text/user definable

# ServiceApp Interfaces

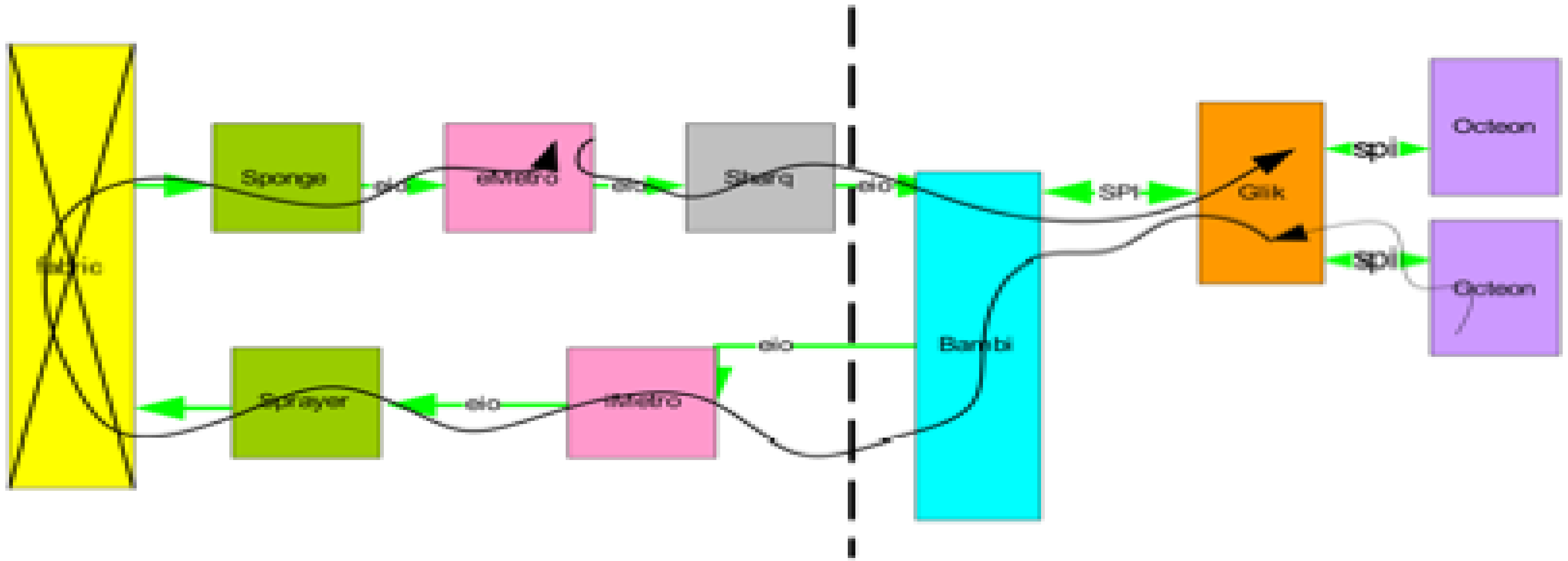
## Logical interfaces/paths between CGSE apps and rest of router

- Treated like regular interfaces from a routing standpoint
  - ServiceApps will go down if the CGSE goes down
  - Can be used to signal availability of CGSE (advertise ServiceApp into IGP)
  - NAT applications will use local static routing to steer traffic into CGSE
- Routing example from NAT44
  - Default route to CGSE in Inside VRF
  - ServiceApp is configured with 80.1.1.1/24
  - Traffic routed to other addresses on 80.1.1.0/24 go to the CGSE
  - Static routes can use interface name, next hop, or both

```
!
interface ServiceApp1
 vrf CGSE-Inside
 ipv4 address 80.1.1.1/24
 service cgn demo service-type nat44
!
```

```
!
router static
 vrf CGSE-Inside
  address-family ipv4 unicast
(option A) 0.0.0.0/0 ServiceApp1
(option B) 0.0.0.0/0 80.1.1.2
(option C) 0.0.0.0/0 ServiceApp1 80.1.1.2
!
```

# Data Path Fault Monitoring

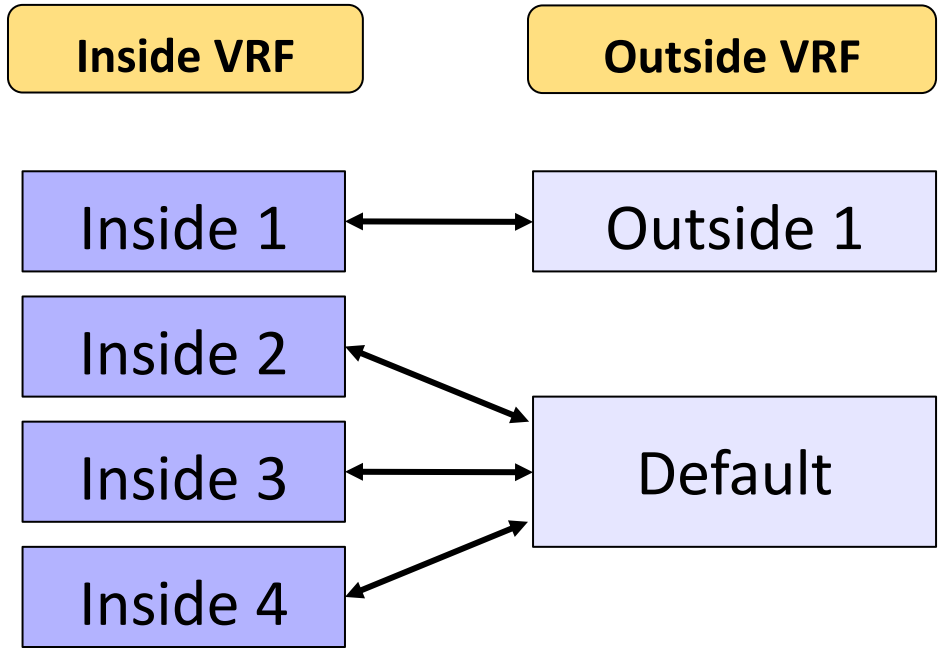



- Each Octeon forms a UDP packet with a destination address in the ServiceInfra subnet range
- The UDP destination port number will identify the Octeon
- The timeout for each packet is 150ms and 3 timeouts cause a fault

# NAT44 Configuration

# NAT44 Deployment Notes

- One NAT44 Instance per CGN (per primary/backup card pair)
- Scaling via multiple pools & VRFs within the NAT44 instance
- Separated VRF model – inside & outside of NAT in different VRFs
  - Outside can be ‘default’ VRF or named, Inside must be a named VRF
  - Each Inside VRF maps to one Outside VRF
  - Multiple inside VRFs may map to same outside VRF
- Src based bypassing (Need ACL Based Forwarding)
- Retrieving NAT Statistics
  - IOS-XR CLI
  - Netflow v9
  - XML
  - ANA (Check support)
- Max IPv4 Outside pool per CGSE is /16
- Max number of subscribers per CGSE is 1 Million

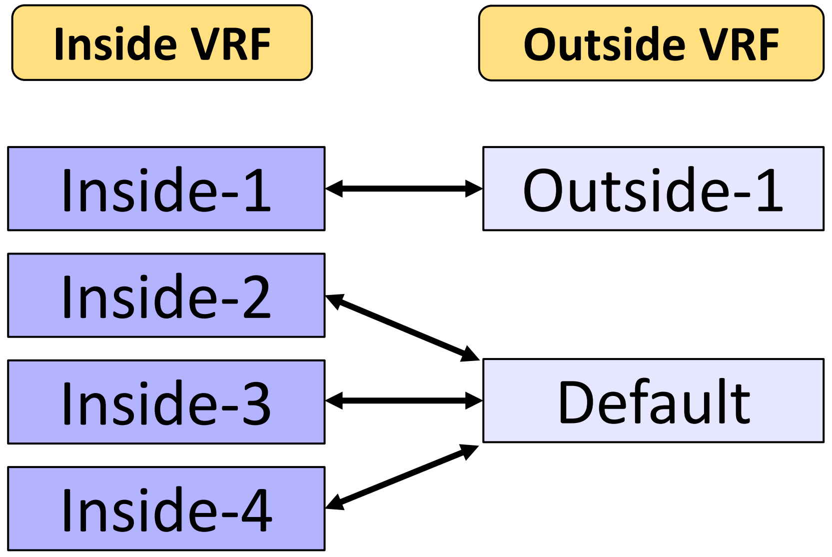


# NAT44 on CGSE

## Setting up the ServiceApps

```
!
interface ServiceApp1
 vrf Inside-1
 ipv4 address 192.168.1.1 255.255.255.252
 service cgn cgn1 service-type nat44
!
interface ServiceApp2
 vrf Inside-2
 ipv4 address 192.168.2.1 255.255.255.252
 service cgn cgn1 service-type nat44
!
interface ServiceApp3
 vrf Inside-3
 ipv4 address 192.168.3.1 255.255.255.252
 service cgn cgn1 service-type nat44
!
interface ServiceApp4
 vrf Inside-4
 ipv4 address 192.168.4.1 255.255.255.252
 service cgn cgn1 service-type nat44
!
```

```
!
interface ServiceApp10
 vrf Outside-1
 ipv4 address 192.168.10.1 255.255.255.252
 service cgn cgn1 service-type nat44
!
interface ServiceApp11
 ipv4 address 192.168.11.1 255.255.255.252
 service cgn cgn1 service-type nat44
!
```

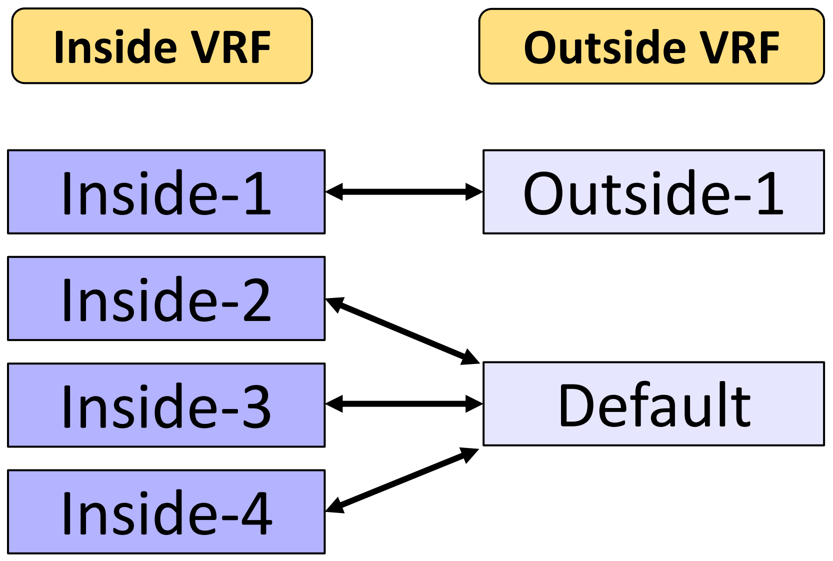



**xx** = free text/user definable

# NAT44 Service-Type Specific Instances

Setting up the service specific instances and routing

```
!
service cgn cgn1
 service-type nat44 nat44-1
  inside-vrf Inside-1
   map outside-vrf Outside-1 address-pool 52.52.0.0/16
  inside-vrf Inside-2
   map address-pool 52.52.0.0/16
  inside-vrf Inside-3
   map address-pool 53.53.32.64/26
  inside-vrf Inside-4
   map address-pool 53.53.18.0/24
!
router static
 address-family ipv4 unicast
  52.52.0.0/16 serviceApp11
  53.53.32.64/26 serviceApp11
  53.53.18.0/24 serviceApp11
 vrf Outside-1
  52.52.0.0/16 serviceApp10
 vrf Inside-1
   0.0.0.0/0 ServiceApp1
 vrf Inside-2
   0.0.0.0/0 ServiceApp2
......
!
```




**xx** = free text/user definable

# NAT44 Options

- portlimit
  - Gives the max number of ports allowed for an user (default 100)
- alg [ActiveFTP|rtsp]
  - Support ‘active’ FTP and or RTSP through the NAT function
- protocol
  - Specify protocol (UDP/TCP/ICMP) specific timeouts
  - Specify MSS
- refresh-direction Outbound
  - If set, session timer will only be reset if the in2out packets are flowing
- filtering-policy
  - If set, address dependent filtering is enabled
- dynamic-port-range start
  - Provides the start port for selecting outside port for dynamic sessions (default is 1024 onwards)
- tcp-policy
  - If enabled, drop non syn packets until session is established (i.e syn received from both sides) .

# NAT44 Configuration Options

```
!
service cgn cgn1
 service-type nat44 nat44-1
  portlimit 200
  alg ActiveFTP
  dynamic-port-range start 1
  tcp-policy
  inside-vrf Inside-1
   map outside-vrf Outside-1 address-pool 52.52.0.0/16
   protocol tcp
    session init timeout 300
    session active timeout 400
    mss 1200
!
```

# NAT64

# IPv6/IPv4 Translation

| Stateless | Stateful |
|---|---|
| 1:1 translation | N:1 translation |
| NAT | NAPT |
| Any Protocol | TCP, UDP, ICMP |
| Helps ensure end-to-end address transparency and scalability | Uses address overloading; hence lacks end-to-end address transparency |
| No state or bindings created on the translation | State or bindings created on every unique translation |
| Session can be initiated from either side | Session must be initiated from IPv6 side |
| Requires IPv4-translatable IPv6 address assignment (mandatory requirement) | No requirement for the characteristics of IPv6 address assignment |
| Requires either manual or Domain Host Configuration Protocol Version 6 (DHCPv6)-based address assignment for IPv6 hosts | Capability to choose any mode of IPv6 address assignment: manual, DHCPv6 or SLAAC (Stateless Address Auto-Configuration) |
| No IPv4 address savings (Just like NAT) | Saves IPv4 addresses |

# IPv6/IPv4 Translation Scenarios

| From | To | stateful | stateless |
|---|---|---|---|
| IPv6 Network | IPv4 Internet | ✓ | ✓ |
| IPv4 Internet | IPv6 Network | ✗ ** | ✓ |
| IPv6 Internet | IPv4 Network | ✓ | ✗ |
| IPv4 Network | IPv6 Internet | Not viable because too few IPv4 addresses | |

**Possible with nat64 v6v4 static mappings

# IPv6/IPv4 Translation Scenarios

| From | To | stateful | stateless |
|---|---|---|---|
| IPv6 Network | IPv4 Network | ✓ | ✓ |
| IPv4 Network | IPv6 Network | ✗ ** | ✓ |
| IPv6 Internet | IPv4 Internet | Cannot be done | |
| IPv4 Internet | IPv6 Internet | Cannot be done | |

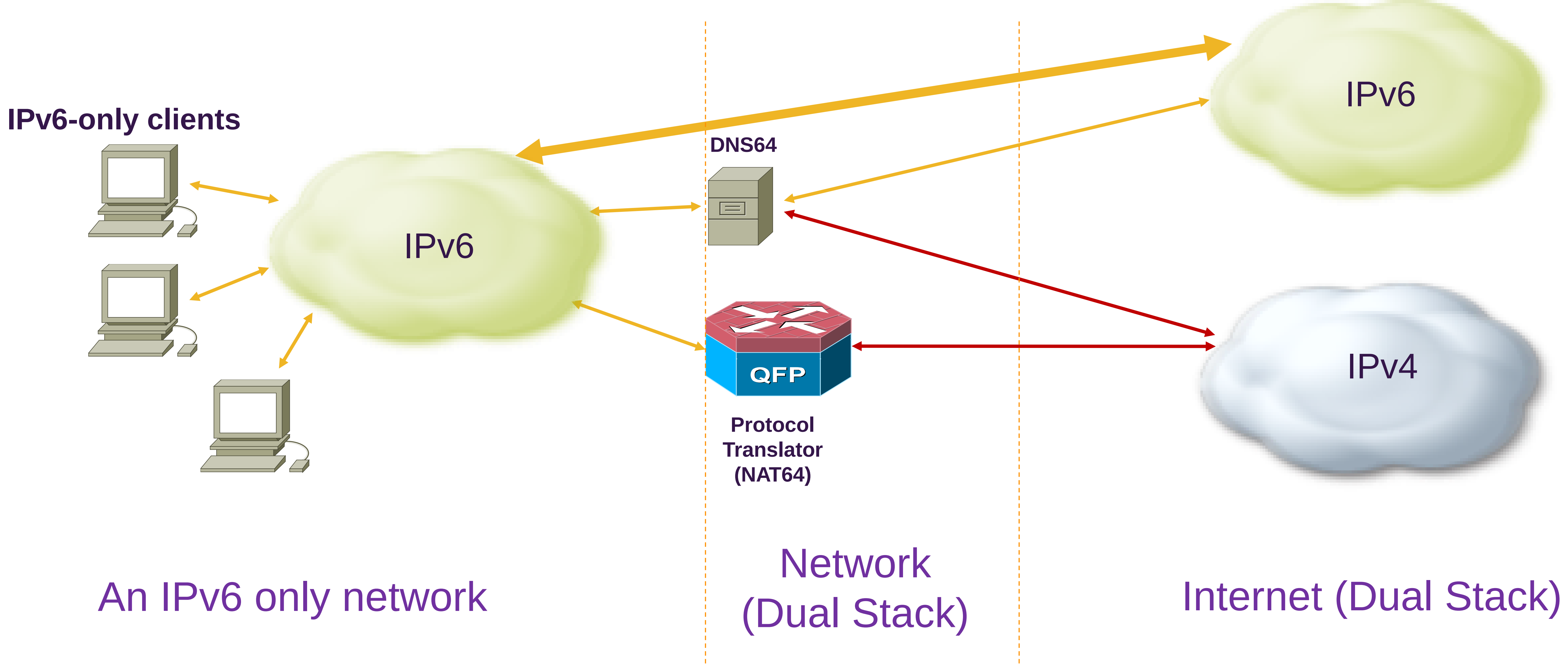
**Possible with nat64 v6v4 static mappings

# IPv6/IPv4 Translation: Two Scenarios

- Connecting an IPv6 network to the IPv4 Internet
  - You built an IPv6-only network, and want to access servers on the IPv4 Internet
  - Example:  IPv6-only mobile handsets

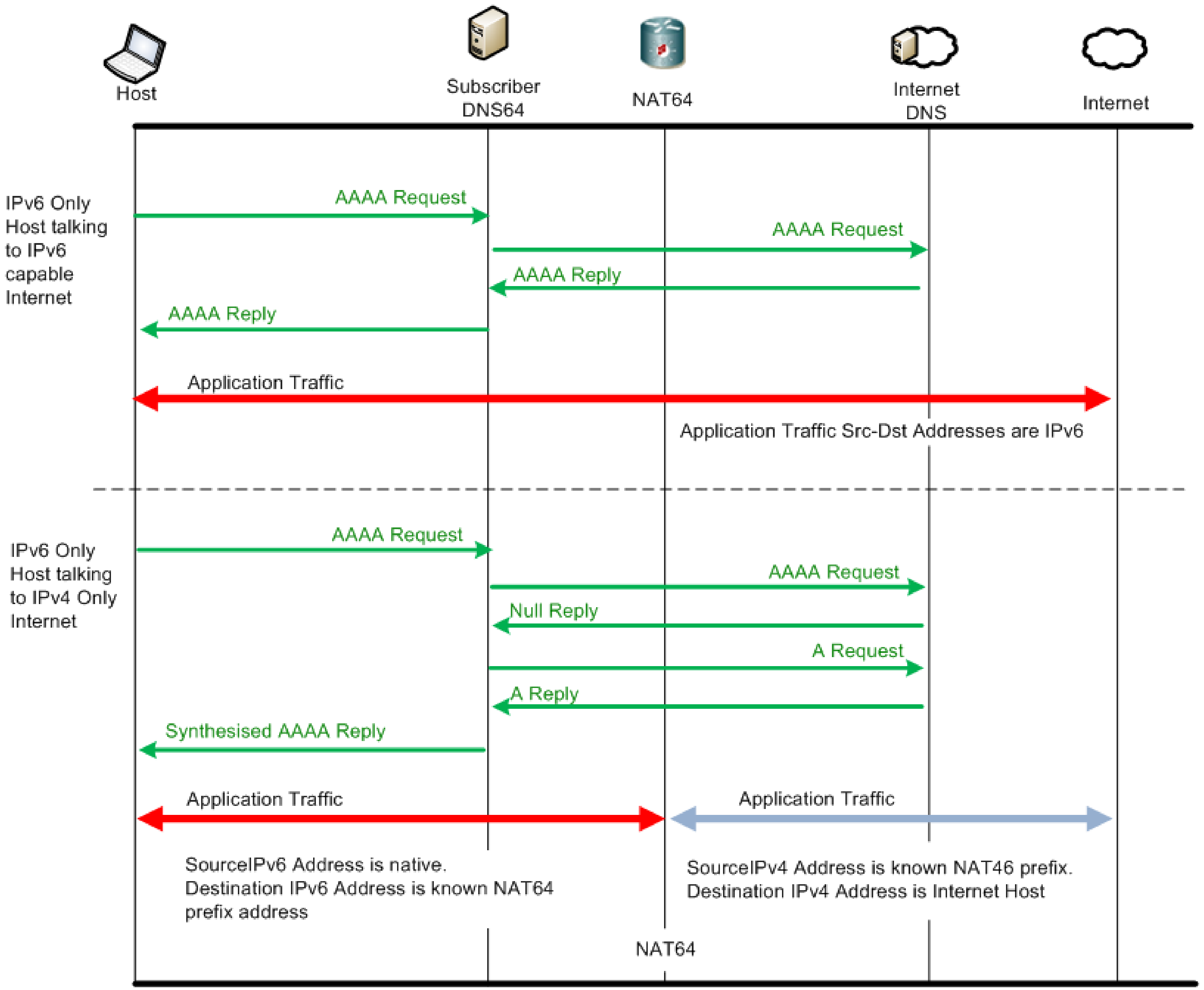
- Connecting the IPv6 Internet to an IPv4 network
  - You have IPv4 servers, and want them available to the IPv6 Internet
  - Example: IPv4-only datacentre (HTTP servers)

# Connecting an IPv6 Network to the IPv4 Internet




http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6553/white_paper_c11-676278.html

# DNS64 Flows




Synthesises AAAA records when AAAA are not present in the DNS

# DNS64

- Works for applications that do DNS queries

  http://www.example.com

  Well over 80% of applications.

- Breaks for applications that don't do DNS queries

  http://1.2.3.4

  SIP, RTSP, H.323, etc. – IP address literals

- Solutions:

  Application-level proxy for IP address literals (HTTP proxy)

  IPv6 application learns NAT64's prefix

# NAT64 Configuration – High Level Overview

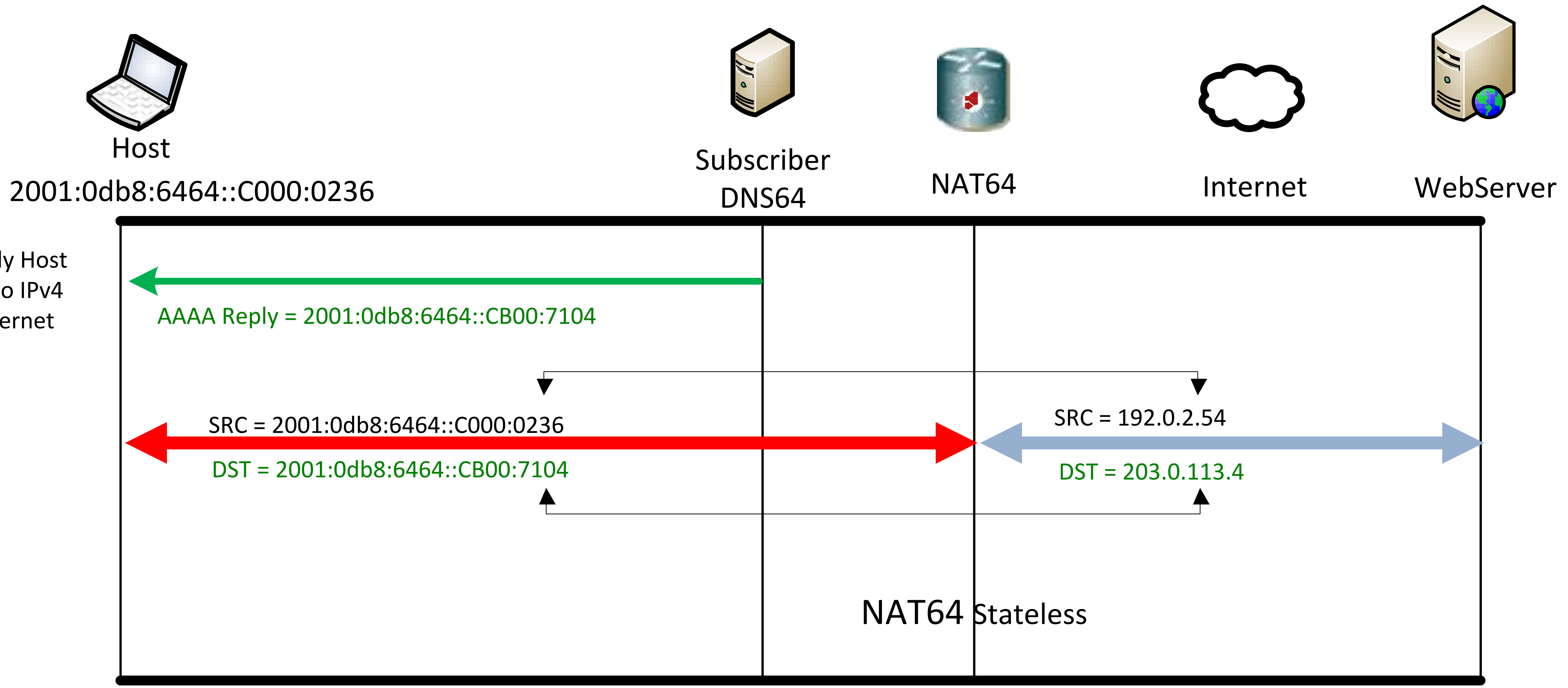
- The IPv4 traffic is diverted to the IPv4 ServiceApp
- The IPv6 traffic is diverted to the IPv6 ServiceApp
- One CGN instance per CGSE
- Multiple NAT64 instances per CGN instance (64 Max)
- Configuration tasks
  - Configure IPv4 and IPv6 Service Apps
  - Configure CGN instance
  - Configure NAT64 instances
  - Associate IPv4 and IPv6 ServiceApps to NAT64 instance
  - Configure routing of external IPv4 and internal IPv6 prefixes to ServiceApps

# Stateless NAT64

- Enables communication between IPv6 & IPv4 hosts
  - Performs packet translation between address families
  - Dual Stack using a single protocol (IPv4 is encoded inside IPv6)
- Limitations on deployment
  - Need to acquire IPv4 addresses (unless doing double NAT on IPv4)
  - Limited deployment scenarios
- Algorithmic mapping of addresses (no state maintained)
  - Limit of Stateless NAT64 is bandwidth
- Stateless NAT64 translates IP header only L4 and above remain intact (copied over)
- A Network Specific Prefix / NAT64 Prefix needs to be defined

# Stateless NAT64

IPv6 Only Host talking to IPv4 Only Internet




Key:

192.0.2.54 = C000:0236

203.0.113.4 = CB00:7104

# Stateless NAT64 on CGSE

```
!
interface ServiceApp4
 ipv4 address 2.0.0.1 255.255.255.0
 service cgn cgn1 service-type nat64 stateless
!
interface ServiceApp6
 ipv6 address 2001:db8:fe00::1/40
 service cgn cgn1 service-type nat64 stateless
!
service cgn cgn1
 service-location preferred-active 0/2/CPU0
 service-type nat64 stateless nat64_sl
  ipv6-prefix 2001:db8:6464::/96
  address-family ipv4
   interface ServiceApp4
  !
  address-family ipv6
   interface ServiceApp6
!
```

```
!
router static
  address-family ipv6 unicast
  2001:db8::/32 ServiceApp6
!
router static
  address-family ipv4 unicast
  100.2.0.0/16 serviceApp4
!
```

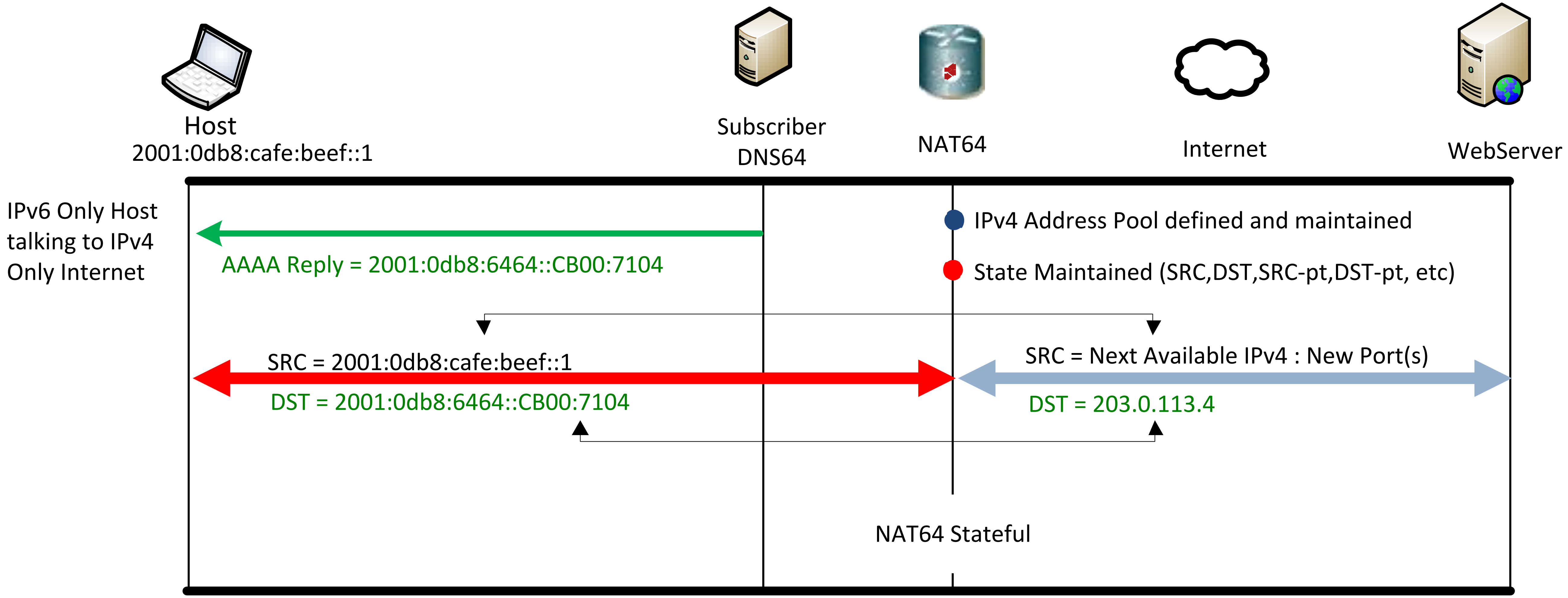
- Simple configuration
  - Configure ServiceApp interfaces
  - Configure CGN service type
  - Configure routing

**XX** = free text/user definable

# Stateful NAT64

- Enables communication between IPv6 & IPv4 hosts
  – Performs packet translation between address families
- Green-field (brand new) network wants to deploy IPv6 only
  – No need to acquire IPv4 addresses for each host
  – Pool of IPv4 addresses required for public Internet access
- Sessions will be initiated by IPv6 clients
- NAT64 translates IP & L4 header (will reset L4 information)
  – Limited to TCP / UDP / ICMP only
  – Algorithmic mapping of destination address
  – Stateful translation of source address and ports
  – Port contention issues
- A Network Specific Prefix / NAT64 Prefix needs to be defined
  – Can use 64:ff9b::/96 or any other prefix

# Stateful NAT64




Key:

203.0.113.4 = CB00:7104

# Stateful NAT64 on CGSE

```
!
interface serviceApp41
 ipv4 add 41.41.41.1/30
 service cgn cgn1 service-type nat64 stateful
!
interface serviceApp61
 ipv6 address 2001:db8:1000::/64
 service cgn cgn1 service-type nat64 stateful
!
service cgn cgn1
 service-location preferred-active 0/2/CPU0
  service-type nat64 stateful nat64_sf
  ipv6-prefix 2001:db8:6464::/96
  ipv4 address-pool 52.52.52.0/24
  ipv4 address-pool 53.53.53.0/24
  !
  address-family ipv4
   interface ServiceApp41
  address-family ipv6
   interface ServiceApp61
!
```

```
!
router static
  address-family ipv6 unicast
  2001:db8:6464::/48 ServiceApp61
!
router static
  address-family ipv4 unicast
  52.52.52.0/24 ServiceApp41
  53.53.53.0/24 ServiceApp41
!
```

- Simple configuration
  – Configure ServiceApp interfaces
  – Configure CGN service type
  – Configure routing

**XX** = free text/user definable

# Stateful NAT64 Options

- portlimit
  - Gives the max number of ports allowed for an user (default 100)
- dynamic-port-range start
  - Provides the start port for selecting outside port for dynamic sessions
- tcp-policy
  - If enabled, drop non syn packets until session is established (i.e syn received from both sides)
- refresh-direction Outbound
  - If set, session timer will only be reset if the in2out packets are flowing
- filtering-policy
  - If set, address dependent filtering is enabled
- IPv4 TOS Setting
  - By default IPv4 TOS field is copied from IPv6 Traffic Class field
- IPv6 Traffic Class Setting
  - By default IPv6 Traffic Class field is copied from IPv4 TOS field
- IPv4 DF override
  - When translating a IPv6 pkt with no Fragment Header IPv4 DF bit is made 1

# NAT64 Configuration Options

```
!
service cgn cgn1
 service-location preferred-active 0/2/CPU0
  service-type nat64 stateful nat64_sf
  portlimit 65535
  ipv6-prefix 2001:db8::/32
  ipv4 address-pool 52.52.52.0/24
  ipv4 address-pool 53.53.53.0/24
  dynamic-port-range start 1
  tcp-policy
  filter-policy
  fragment-timeout 10
  address-family ipv4
   tos 125
   interface ServiceApp41
   tcp mss 900
  address-family ipv6
   interface ServiceApp61
   traffic-class 130
   tcp mss 750
!
```

# IPv6/IPv4 Translation Issues

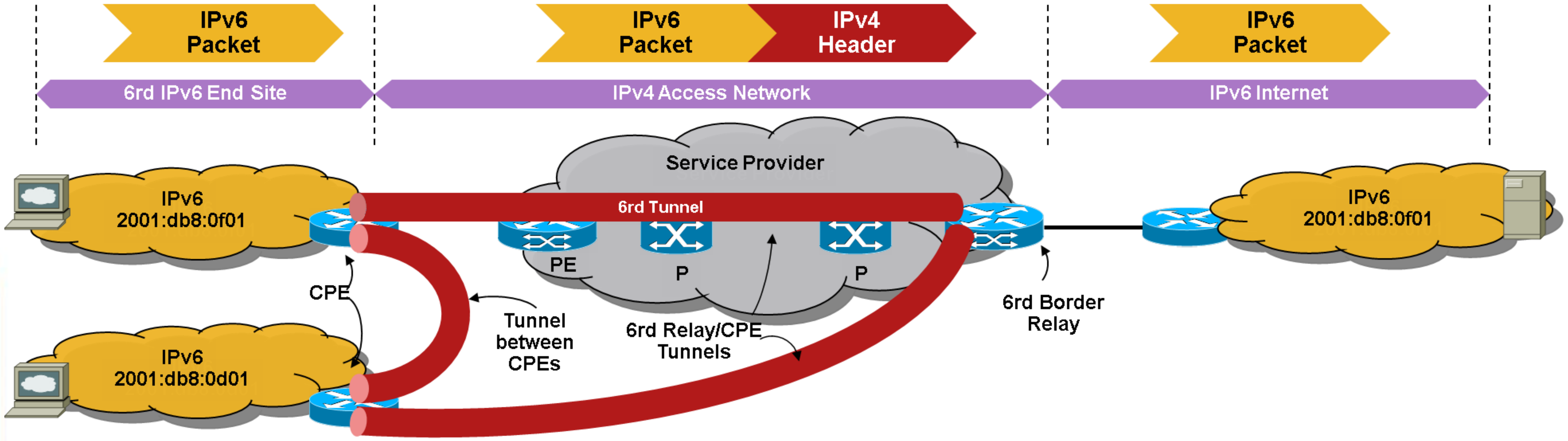
- IPv4 address literals
  - http://1.2.3.4, SIP, RTSP, etc.
- Application Layer Gateway, or application proxy
  - FTP (EPSV, PASV)
  - RTSP in mobile environments (3G)
  - Others applications?

# IPv6 Rapid Deployment (6rd) Configuration

# IPv6 Rapid Deployment (6rd) Overview

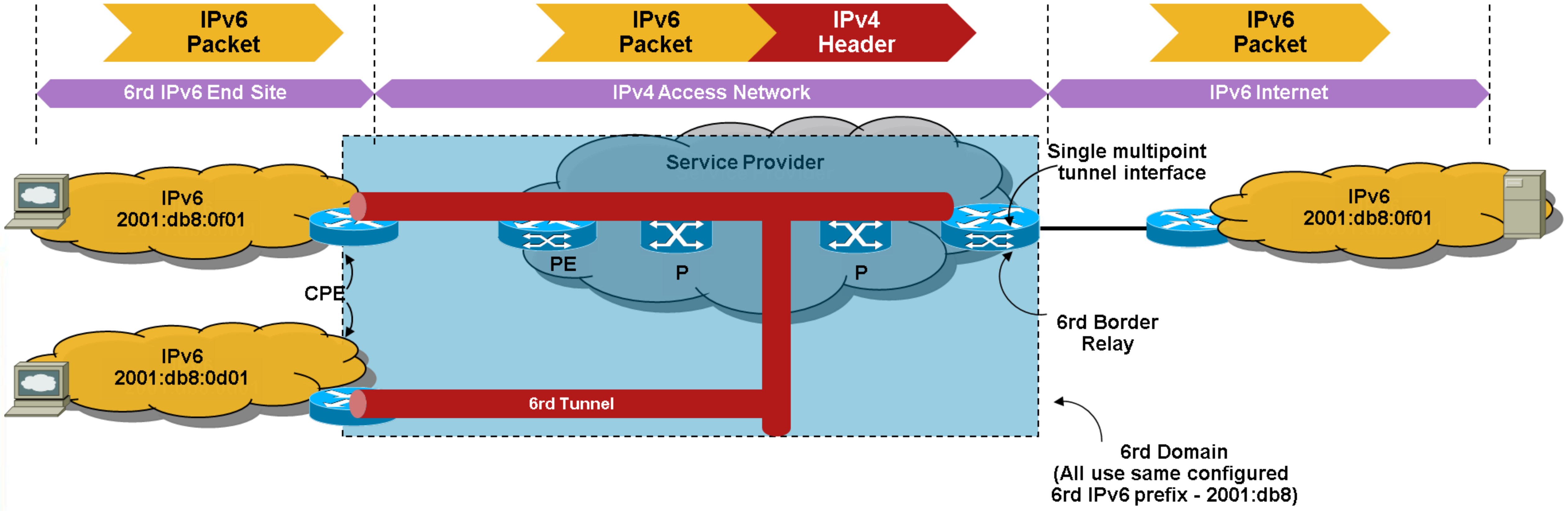
- 6rd is a tunnelling method specified in RFC 5969
  - Superset of 6to4 tunnelling [RFC3056]
  - 6rd utilises an SP's own IPv6 address prefix - avoids well-known prefix (2002::/16)
- Method of deploying IPv6 to end sites in an SP network where the access is IPv4
  - SP access and aggregation infrastructure is IPv4
  - End site is provided a dual stack service (LAN side)
  - 6rd overlay onto the Access/Aggregation network looks like multipoint network
- End sites share a common IPv6 prefix allocated by SP
- 6rd primarily supports IPv6 deployment to
  - A customer site (residential gateway)
  - **Requires CPE to support and be configured with 6rd function**
  - Doesn't interconnect the IPv4 with the IPv6 world, just provides a transport to IPv6

# 6rd Tunnels (RFC 5969)




- CPE = 6rd Residential Gateway
- Native dual-stack IP service to the end site
- Simple, stateless, automatic IPv6-in-IPv4 encap and decap functions
- Embedded IPv4 address needs to match IPv4 address in Tunnel header for security
- IPv6 traffic automatically follows IPv4 Routing (IPv4 address used as tunnel endpoint)
- BRs placed at IPv6 edge, addressed via anycast for load-balancing and resiliency

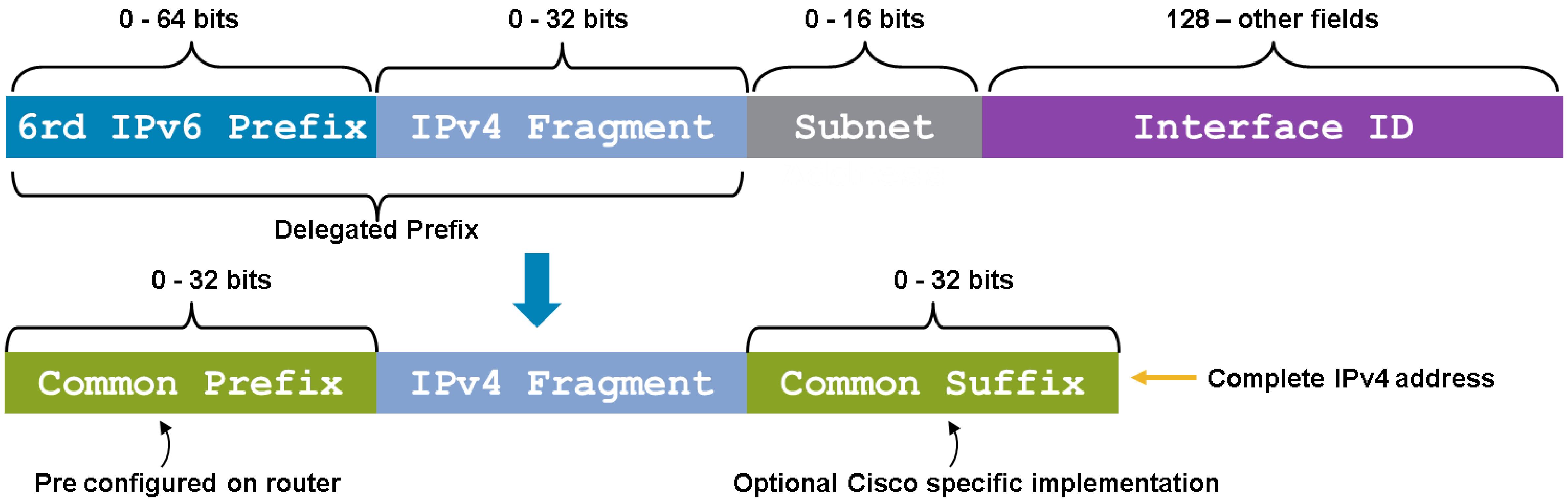
# 6rd Logical NBMA Behaviour




- 6rd views the IPv4 network as an NBMA link layer for IPv6
- Border Relay serves has a single multipoint interface
  - No per user state, serves all users in 6rd domain

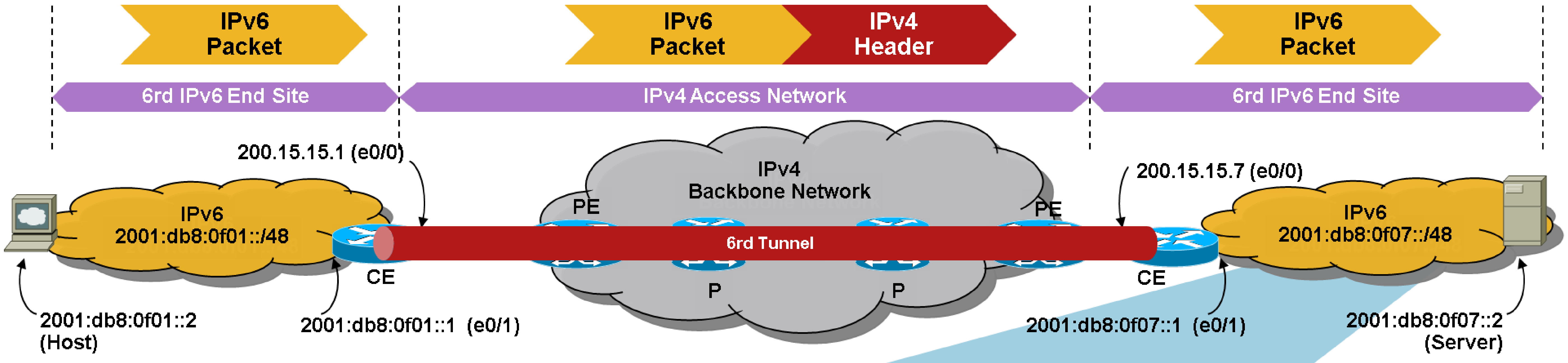
# 6rd Delegated Prefix

- Every customer site is assigned a 6rd delegated prefix
- Delegated prefix is created by
  - Combining the SP's 6rd prefix and all or part of the CE IPv4 address
- Not all 32 bits of IPv4 address need be carried
  - Common IPv4 prefix and suffix can be pre-configured

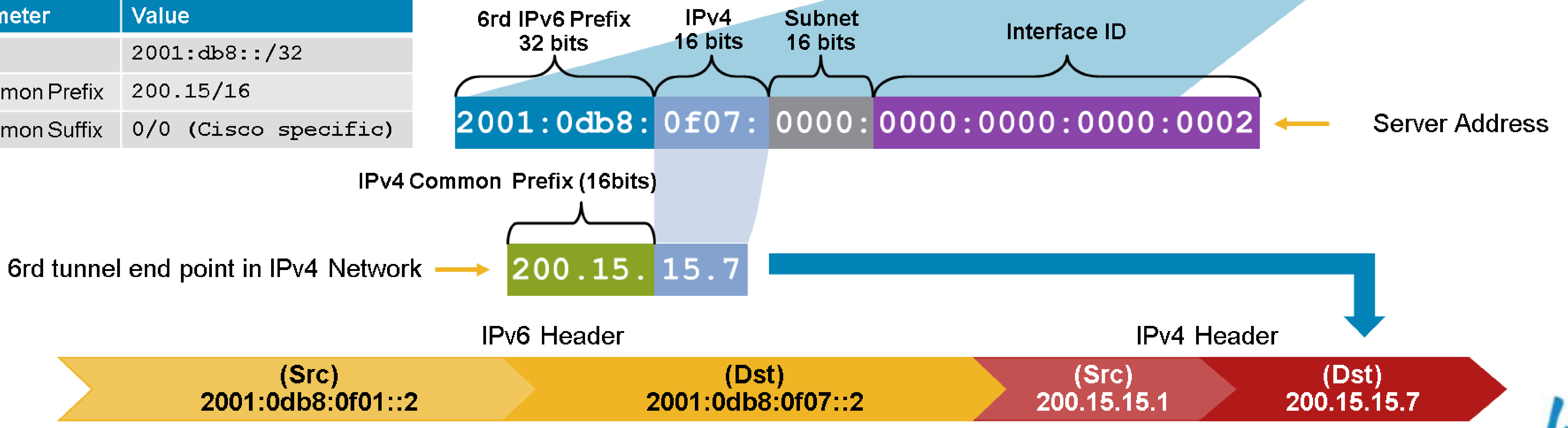


# Destination Dynamically Computed

## CPE to CPE within a 6rd domain

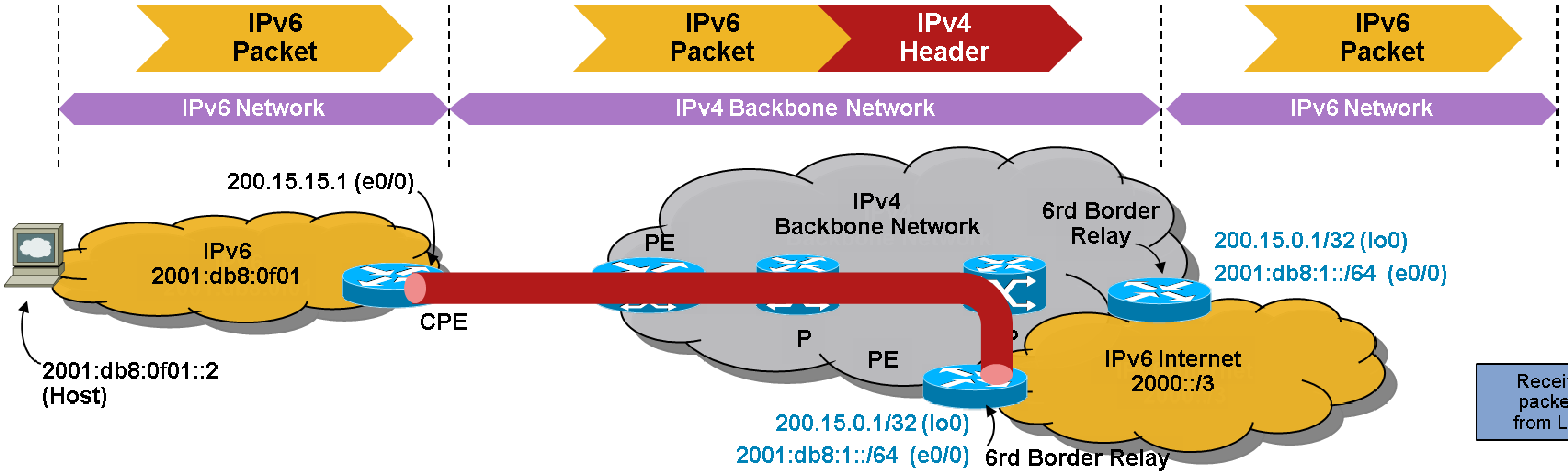


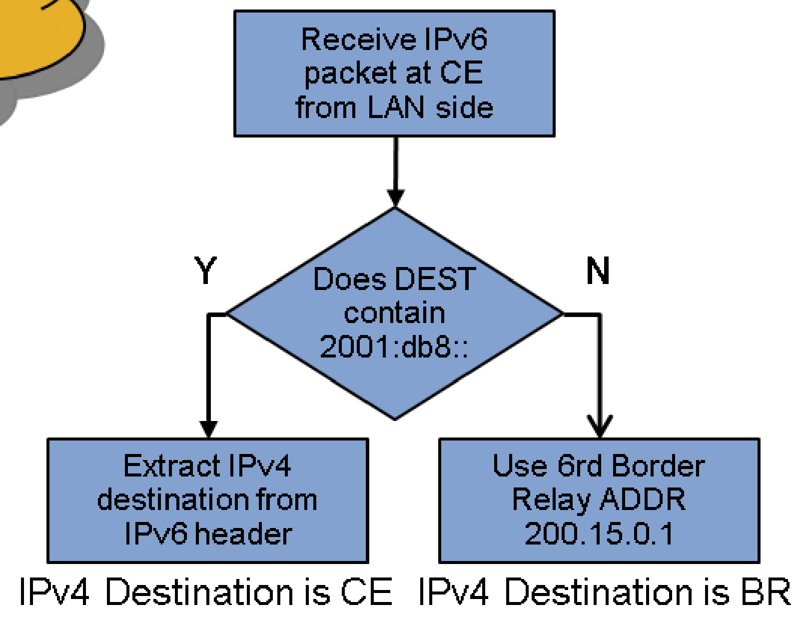

| 6rd Parameter | Value |
|---|---|
| 6rd Prefix | 2001:db8::/32 |
| IPv4 Common Prefix | 200.15/16 |
| IPv4 Common Suffix | 0/0 (Cisco specific) |



# Follow the White Rabbit

## Following default to the 6rd Border Relay




- 6rd Border Relay behaves as default route to IPv6 global Internet
- If IPv6 destination outside of 6rd prefix then tunnel packet to border relay
- 6rd is Stateless
  - Can use anycast addresses for 6rd BRs
  - Failover controlled by IGP, no effect on user traffic

# 6rd on CGSE

```
!
interface ServiceApp3
 ipv4 address 172.16.3.1 255.255.255.0
 service cgn cgn1 service-type tunnel v6rd
!
interface ServiceApp4
 ipv6 address 2001:db8::1/64
 service cgn cgn1 service-type tunnel v6rd
!
service cgn cgn1
 service-type tunnel v6rd 6rd
  br
   ipv6-prefix 2001:420:81::/56
   source-address 10.12.0.254
   ipv4 prefix length 24
   ipv4 suffix length 0
   unicast address 2001:420:81:fe::1
  !
  address-family ipv4
   interface ServiceApp3
  !
  address-family ipv6
   interface ServiceApp4
!
```

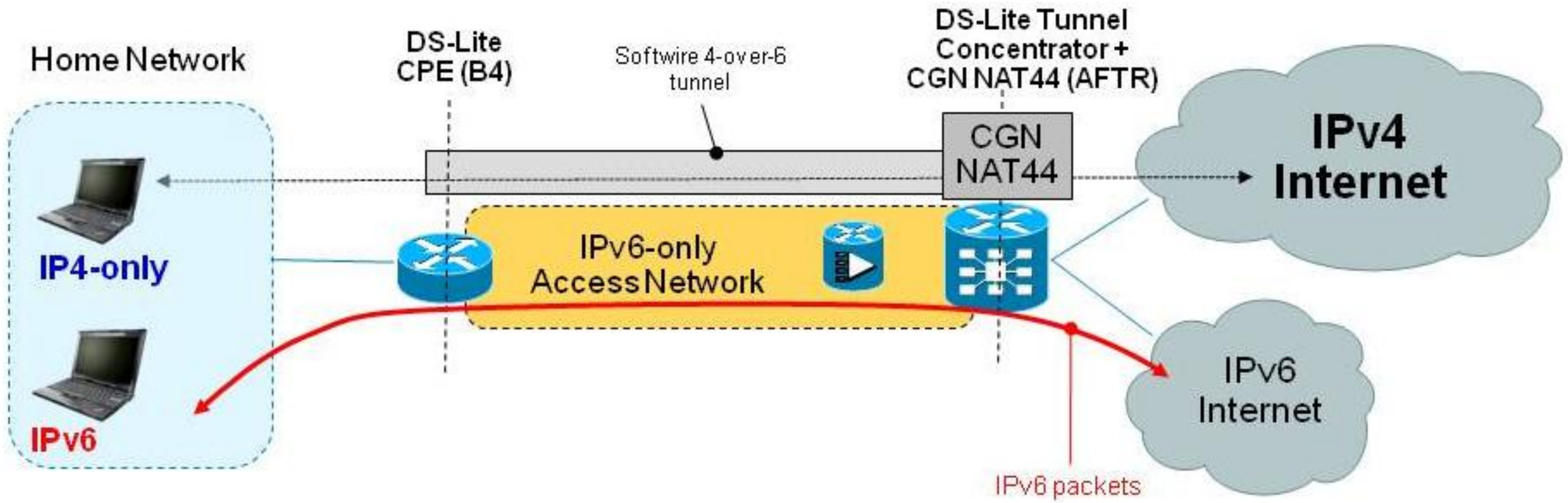
```
!
router static
 address-family ipv4 unicast
   10.12.0.254/32 vrf default ServiceApp3
!
```

- Simple configuration
  - Configure ServiceApp interfaces
  - Configure CGN service type
  - Configure routing

**XX** = free text/user definable

# DS-Lite

# DS-Lite




- Provides for a tunnelling solution for IPv4 client to access IPv4 servers via IPv6 access network. Conforms with draf-ieft-softwire-dual-stack-lite
- Supported from release 4.2.1 FCS
- Bulk Port Range and syslog support in addition to NetFlow
- Performance/Scale – 64 DS-Lite instances, 20Gbps, 20 mil sessions
- Successful showcase/interop at EuroCableLabs event

# DS-Lite Operation

- B4 - The CPE router that are attached to end hosts are referred to as B4 (Basic Bridging Broad-Band) element. IPv4 packets entering B4 are encapsulated using an IPv6 tunnel and sent to the AFTR. In the reverse direction tunnelled packets coming from the AFTR are de-capsulated and the IPv4 packets are routed to the end hosts.
- AFTR (Address Family Transition Router) element is the router that terminates the tunnel from the B4. It de-encapsulates the tunnelled IPv4 packet, NATs it and routes to the IPv4 Internet. In the reverse direction, IPv4 packets coming from the Internet are reverse NATed and the resultant IPv4 packets are sent the B4 using an IPv6 tunnel.
- IPv4 packet flow is shown using the Blue lines.
- IPv6 packet flow – this is shown using the Red lines. There is no tunnelling for these packets and are routed natively in the B4 and the AFTR routing elements.

# DS-Lite on CGSE

```
!
int serviceApp41
 ipv4 add 41.41.41.1/30
 service cgn cgn1 service-type ds-lite
!
int serviceApp61
 ipv6 address 2001:db8:1000::/64
 service cgn cgn1 service-type ds-lite
!
service cgn cgn1
 service-location preferred-active 0/2/CPU0
  service-type ds-lite ds-1
  map address-pool 52.52.52.0/24
  aftr-tunnel-endpoint-address 2001:db8:e0e:e01::
  !
  address-family ipv4
   interface ServiceApp41
  address-family ipv6
   interface ServiceApp61
!
```

```
!
router static
  address-family ipv6 unicast
  2001:db8:e0e:e01::/128 ServiceApp61
!
router static
  address-family ipv4 unicast
  52.52.52.0/24 ServiceApp41
!
```

- Simple configuration
  - Configure ServiceApp interfaces
  - Configure CGN service type
  - Configure routing

NOTE: 2001:db8:e0e:e01:: is the “aftr-tunnel-endpoint-address”

**XX** = free text/user definable

# DS-Lite Options

- portlimit
  – Gives the max number of ports allowed for an user (default 100)
- bulk-port-alloc
  – Enable bulk port allocation and set bulk size (used to reduce logging data volume)
- protocol
  – Specify protocol (UDP/TCP/ICMP) specific timeouts
  – Specify MSS

# DS-Lite Configuration Options

```
!
service cgn cgn1
 service-location preferred-active 0/2/CPU0
  service-type ds-lite ds-l
    portlimit 200
    bulk-port-alloc size 128
    map address-pool 52.52.52.0/24
    aftr-tunnel-endpoint-address 2001:db8:e0e:e01::
    address-family ipv4
      interface ServiceApp41
    address-family ipv6
      interface ServiceApp61
    protocol tcp
      session init timeout 300
      session active timeout 400
      mss 1200
!
```

# Summary of IPv4 Continuation & IPv6 Transition Technologies

| | CGN NAT44 | DS-Lite | 6RD | NAT64 SL | NAT64 SF |
|---|---|---|---|---|---|
| CPE | No CPE change | CPE change required | CPE change required | IPv4 or IPv6 hosts** | Only IPv6 hosts |
| IPv4 continuity | Yes | Yes | NAT44 Optional | N/A | N/A |
| IPv6 transition | N/A | Yes | Still requires IPv4 address. | Yes. IPv6 Only hosts still require IPv4 addresses | Yes |
| Access NW | IPv4/v6 | IPv6 | IPv4 | IPv6 | IPv6 |
| Stateful /Stateless | Stateful | Stateful | Stateless | Stateful | Stateful |

See draft-wing-nat-pt-replacement-comparison for more detail
** Any host can initiate the connection

# Logging

# Netflow v9 Logging

- Netflow logging for NAT44, NAT64 (Stateful) & DS-Lite
- With a default path MTU of 1500 bytes, one Netflow packet can hold ≈50 records.
- Generation is handled at the CPU core level.
- An event (new translation or deletion of an existing one) will trigger the creation of a NF packet but it's not sent directly.
  - If other events happen for the same core, records are added to the packet.
  - Packet is sent if we reach the MTU size or if we exceed one second.

# Netflow v9 Configuration

- session-logging
  - Enabling this would send ‘destination’ info along with the translation info (Netflow records will be sent only when the destination information is also available). Default is to send only translation info (send when translation entry is created)
- path-mtu
  - MTU for the Netflow packet
- refresh-rate
  - template refresh time

```
!
service-type nat64 stateful nat64-sf
<snip>
!
  external-logging netflow version 9
   server
    address 90.1.1.1 port 99
!
```

```
!
service-type nat64 stateful nat64-sf
<snip>
!
  external-logging netflow version 9
   server
    address 90.1.1.1 port 99
    session-logging
    path-mtu 1476
!
```

# Syslog Logging

- Supported for NAT44 and DS-Lite
- Message needs to comply to RFC5424 format
- Bulk port allocation may be used to reduce the number of records generated.
  - Record generated when the block is assigned.
  - Record generated when a new block is necessary (when all ports are used).
  - Record generated when a block is released after the last session timed-out.

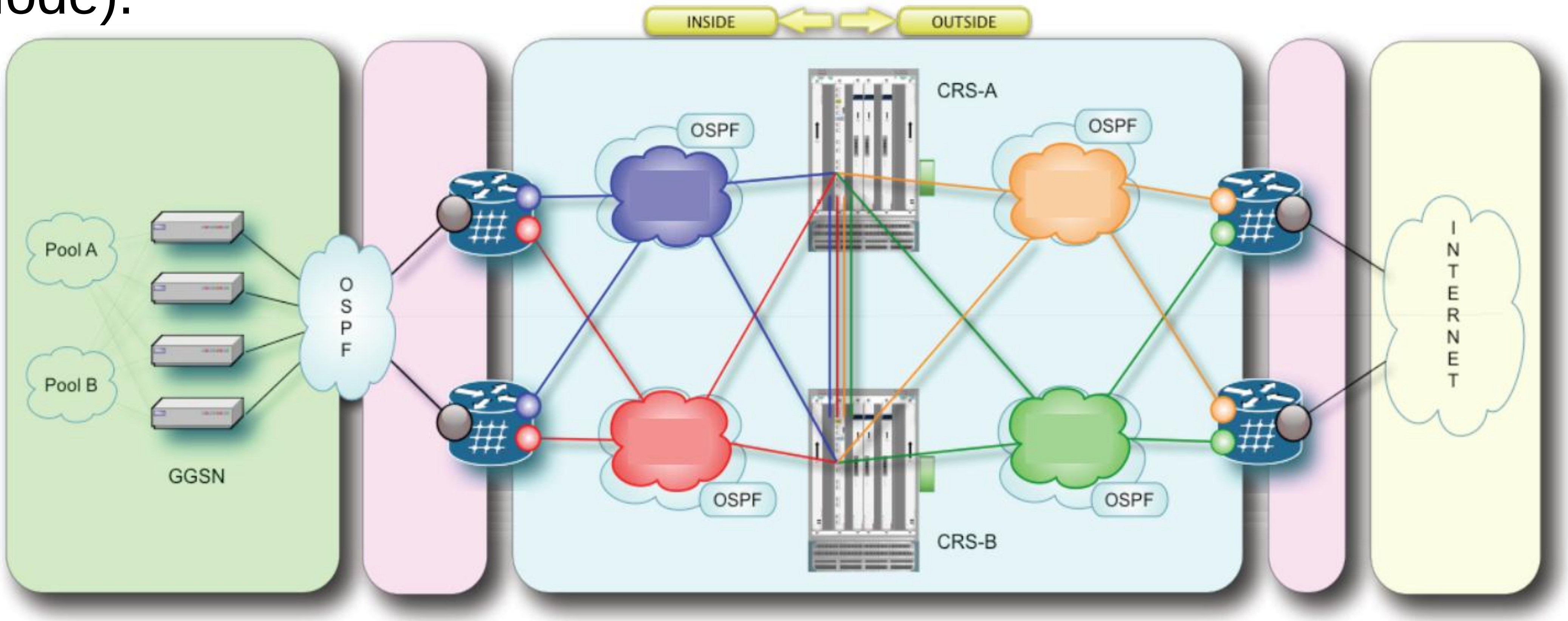
# Syslog Configuration

```
!
service-type nat44 nat44
<snip>
 !
  external-logging syslog
    server
      address 90.1.1.1 port 514
!
```

# CGSE Deployment Options

# CGSE Placement

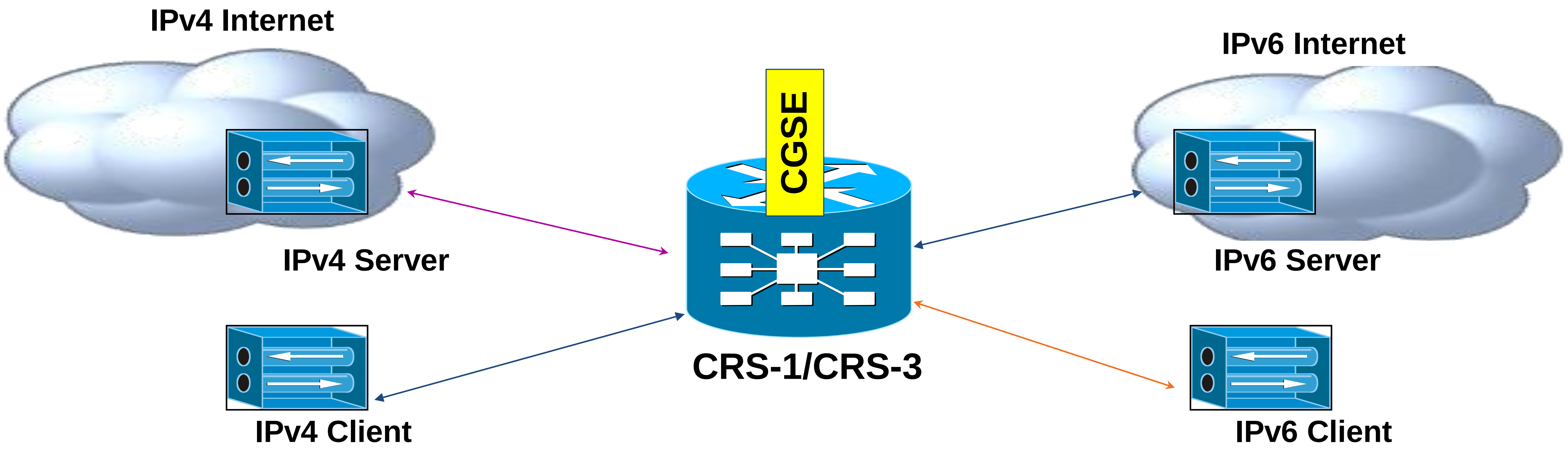
- Example of  Typical customer are using the CGSE to provide private/public IPv4 address translation for mobile customers behind GGSNs (Gateway GPRS Support Node).




Move to edge if:
- High bandwidth requirements
- Large volume of subs
- limited access to RFC1918 or translation space

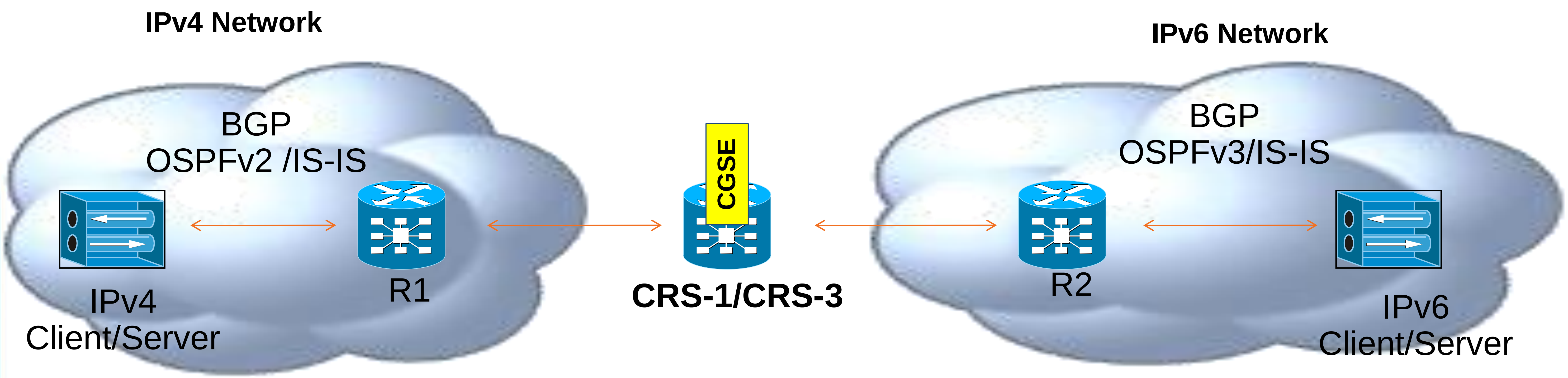
Move to core/Internet GW if:
- Low bandwidth requirements
- Low volume of subs
- Ample internal coverage of RFC1918 or translation space

# Case 1 – Basic CGSE




- An IPv6 network to IPv4 Internet & vice-versa
- IPv6 network to IPv4 network & vice-versa

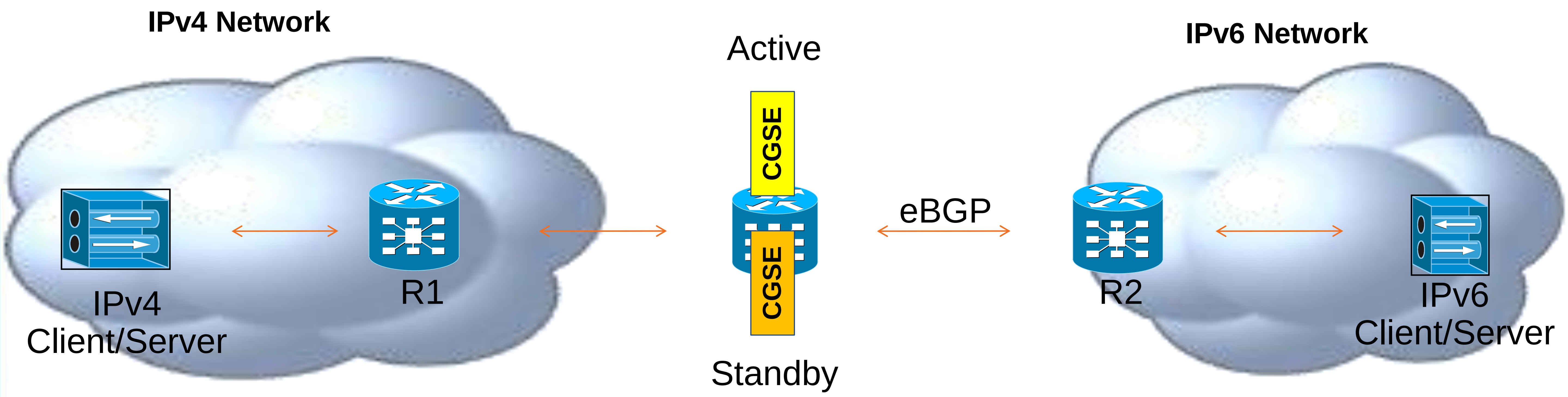
# Case 2 – CGSE Neighbouring (IGP/BGP)




- An IPv6 network to IPv4 Internet & vice-versa
- OSPFv2/IS-IS between CGSE & R1
- OSPFv3/IS-IS between CGSE & R2

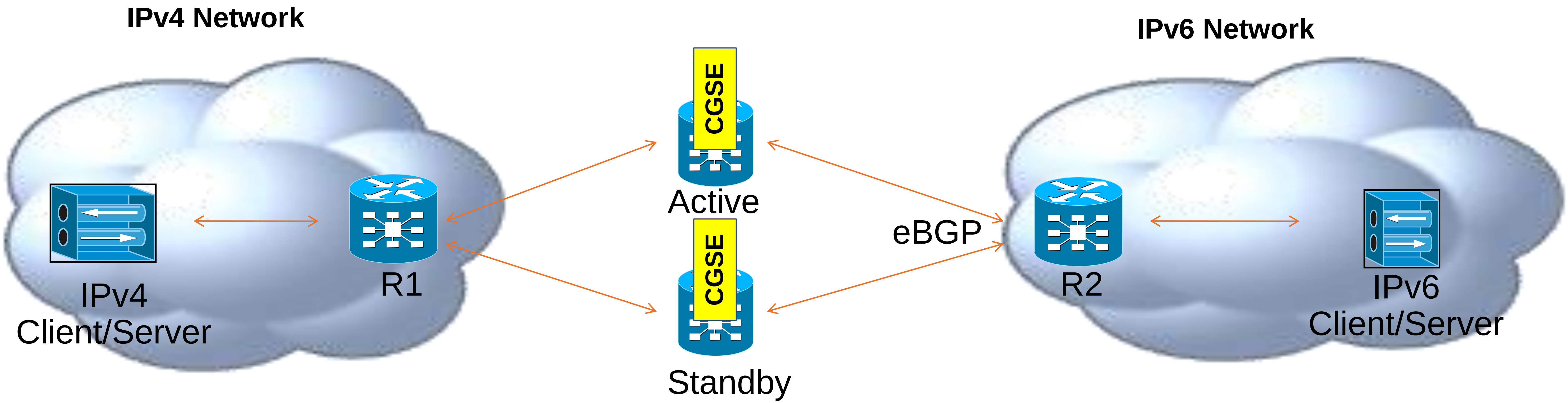
# Case 3 – Redundant CGSEs

## (Same chassis)



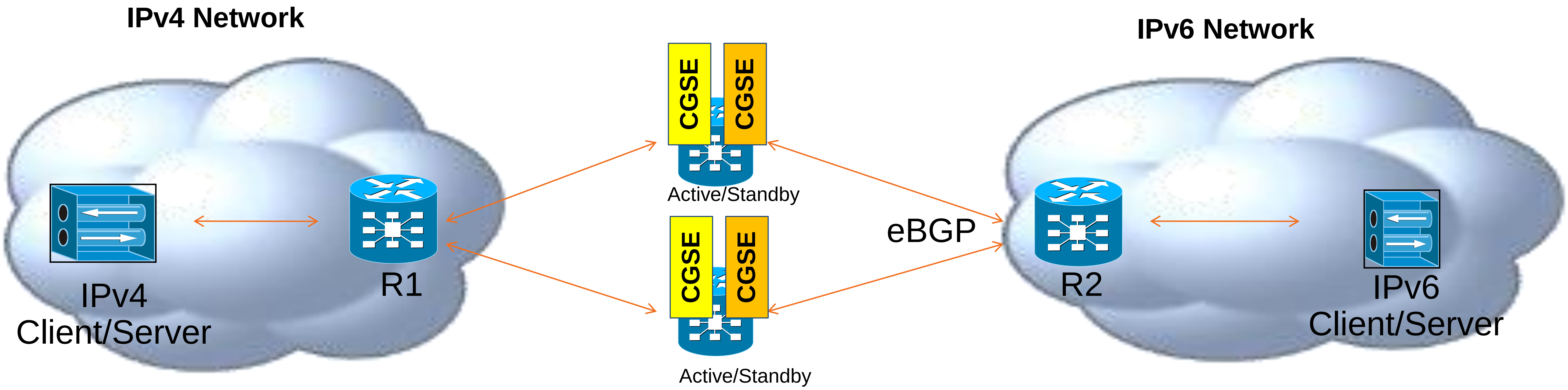

- An IPv6 network to IPv4 Internet & vice-versa
- Subscriber traffic follows best IP path.
- Static routes to IPv4 /IPv6 destination with metric assigned for Serviceapp interfaces
- Same NSP Prefix for both CGSEs

# Case 4 – Redundant CGSEs
## (multiple chassis deployment )




- An IPv6 network to IPv4 Internet & vice-versa
- Subscriber traffic follows best IP path.
- Same NSP prefix needs to be configured since it is stateless, synchronisation is not required.

# Case 5 – Redundant CGSEs
## (multiple chassis deployment)




- An IPv6 network to IPv4 Internet & vice-versa
- Subscriber traffic follows best IP path.
- Same NSP prefix needs to be configured since it is stateless, synchronisation is not required.

# Redundancy and High Availability (HA) Summary

- Intra-chassis Redundancy
  - 1:1 warm standby
  - Active/Active model with multiple active CGSEs
- Inter-chassis Redundancy
  - HSRP, IGPs, BGP, IPSLA, etc
- Fault Monitoring and High Availability (HA)
  - There are several fault monitoring paths:

    MSC CPU -> Octeon CPU

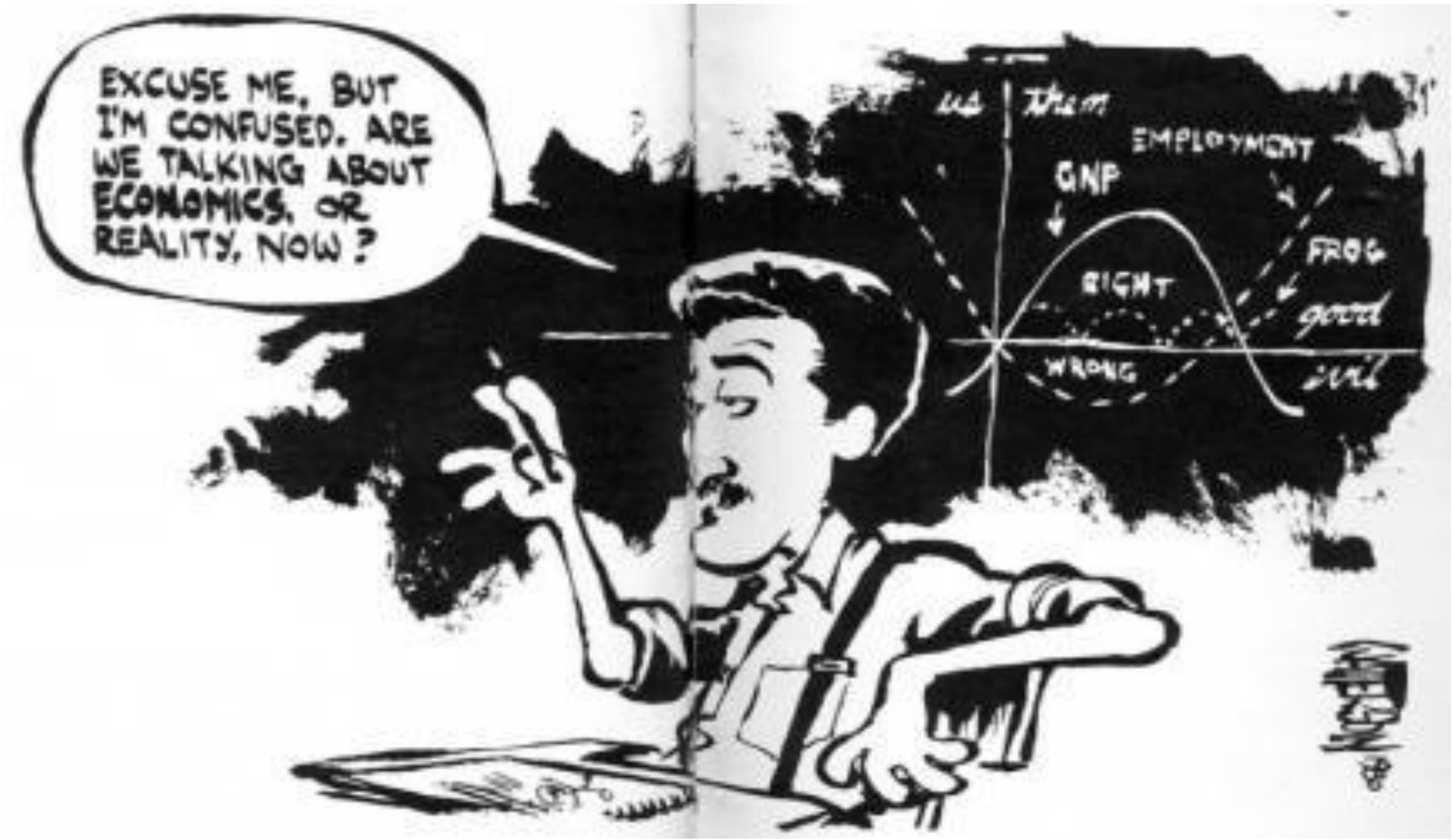
    Data Path (Octeon CPU -> Fabric -> Octeon CPU)
  - With default keep-alive timeouts, failures are detected within 500ms
  - HA Failures will result in a reload of the MSC+CGSE PLIM

# CGSE Resources

- CGSE Home Page
  - http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6553/brochure_c02-560497_ns1017_Networking_Solutions_Brochure.html
- Cisco CGv6 home page
  - http://www.cisco.com/go/CGv6
- Cisco Ipv6 home page
  - http://www.cisco.com/go/ipv6
- Cisco CRS home page
  - http://www.cisco.com/go/crs

# IPv6 Economics

# Realities





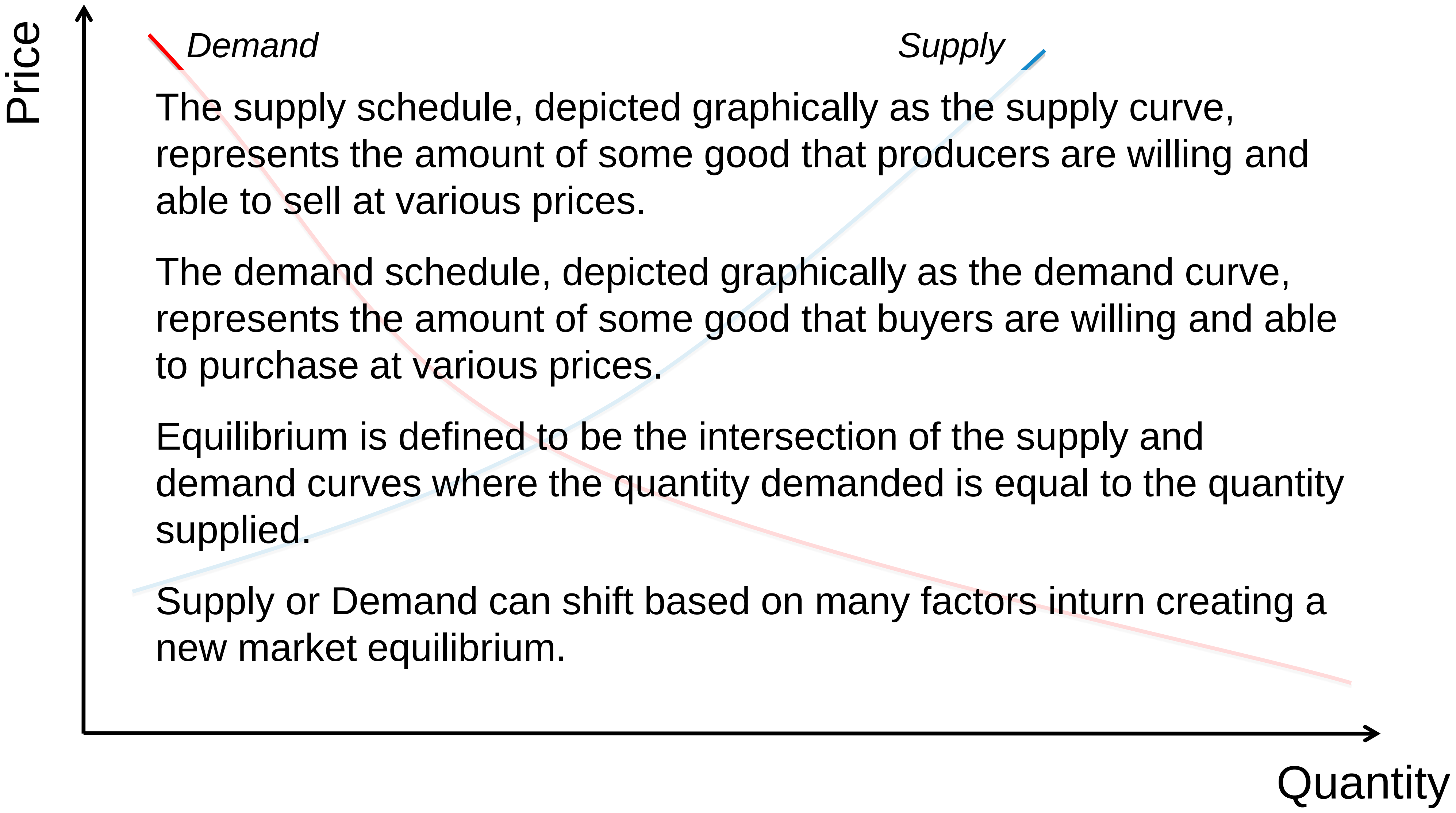
NOTE: I am not an economist.

# ©

- I have plagiarised the following slides from Geoff Huston (www.potaroo.net)
- The interpretation and modification are entirely my own

# Economics 101

## The Supply Demand Schedule




The supply schedule, depicted graphically as the supply curve, represents the amount of some good that producers are willing and able to sell at various prices.

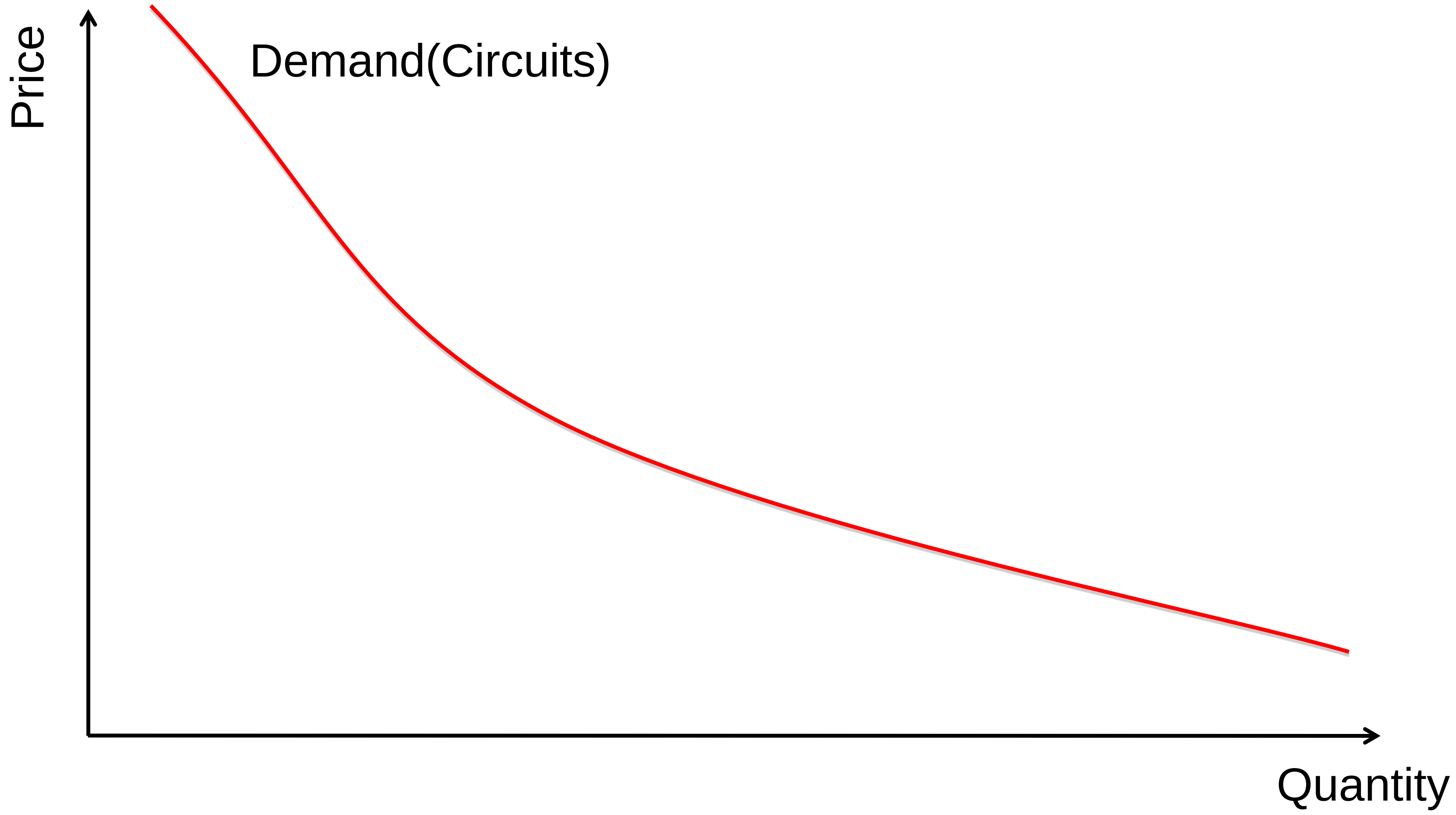
The demand schedule, depicted graphically as the demand curve, represents the amount of some good that buyers are willing and able to purchase at various prices.

Equilibrium is defined to be the intersection of the supply and demand curves where the quantity demanded is equal to the quantity supplied.

Supply or Demand can shift based on many factors inturn creating a new market equilibrium.

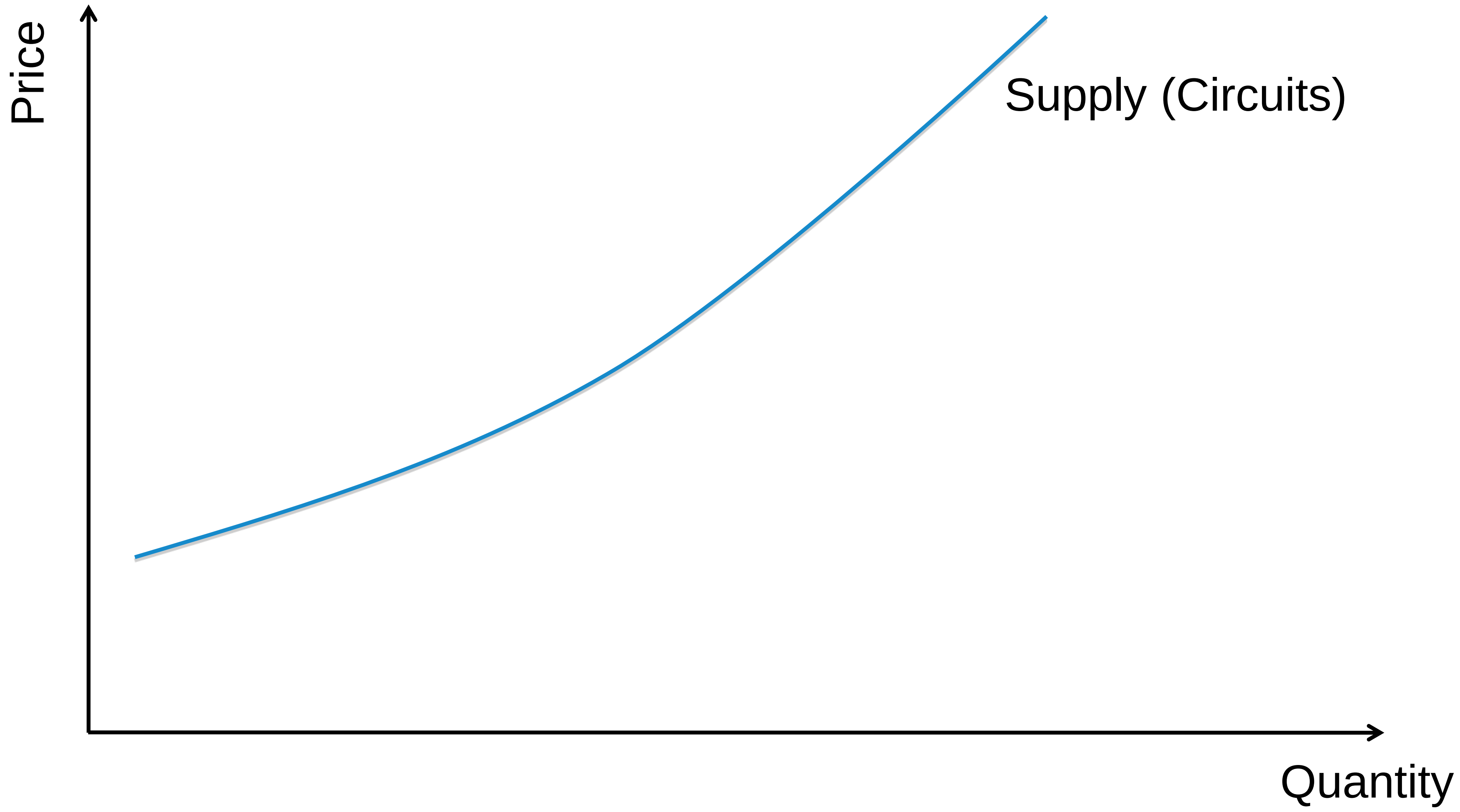
# The Demand Schedule

## Consumers



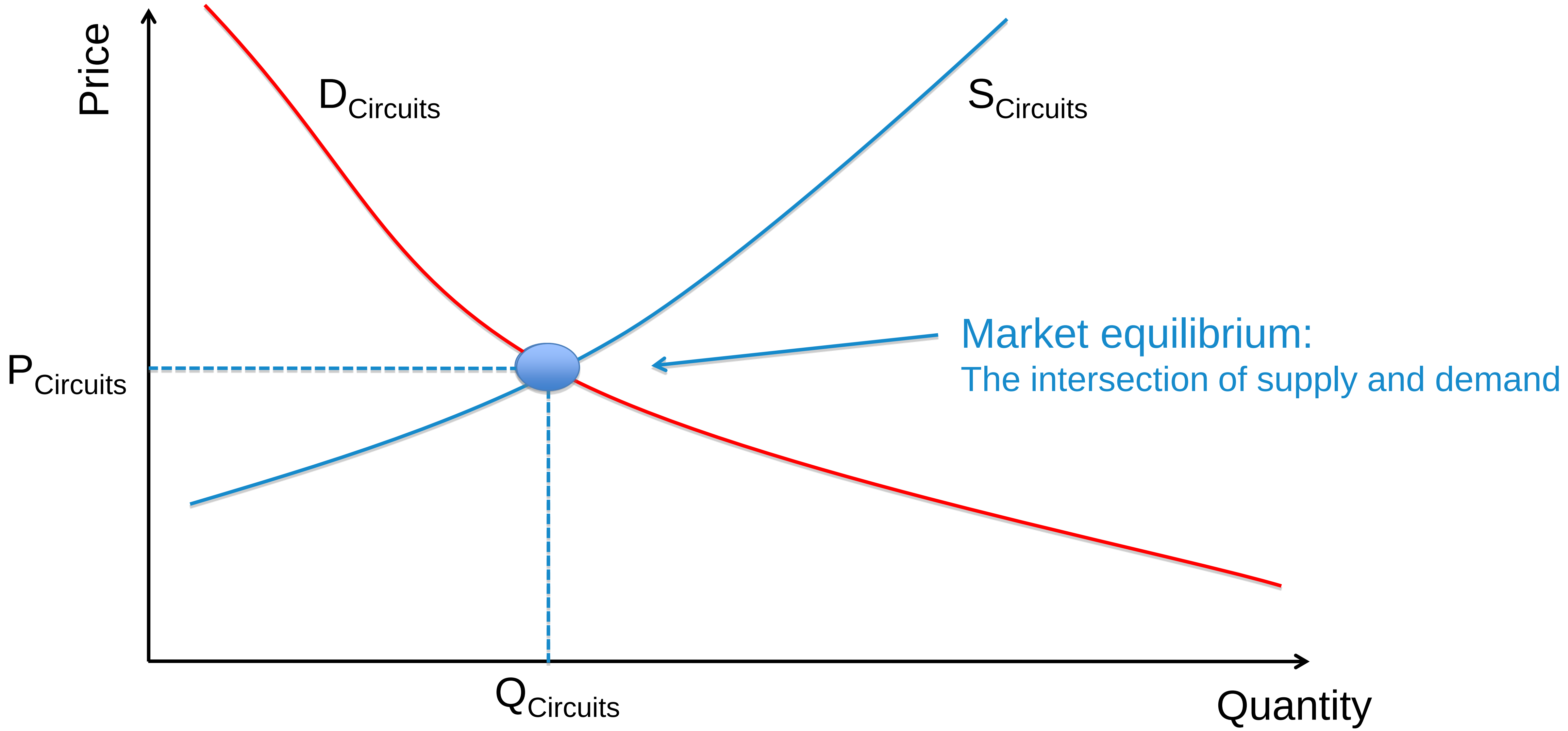
# The Supply Schedule

## Producers



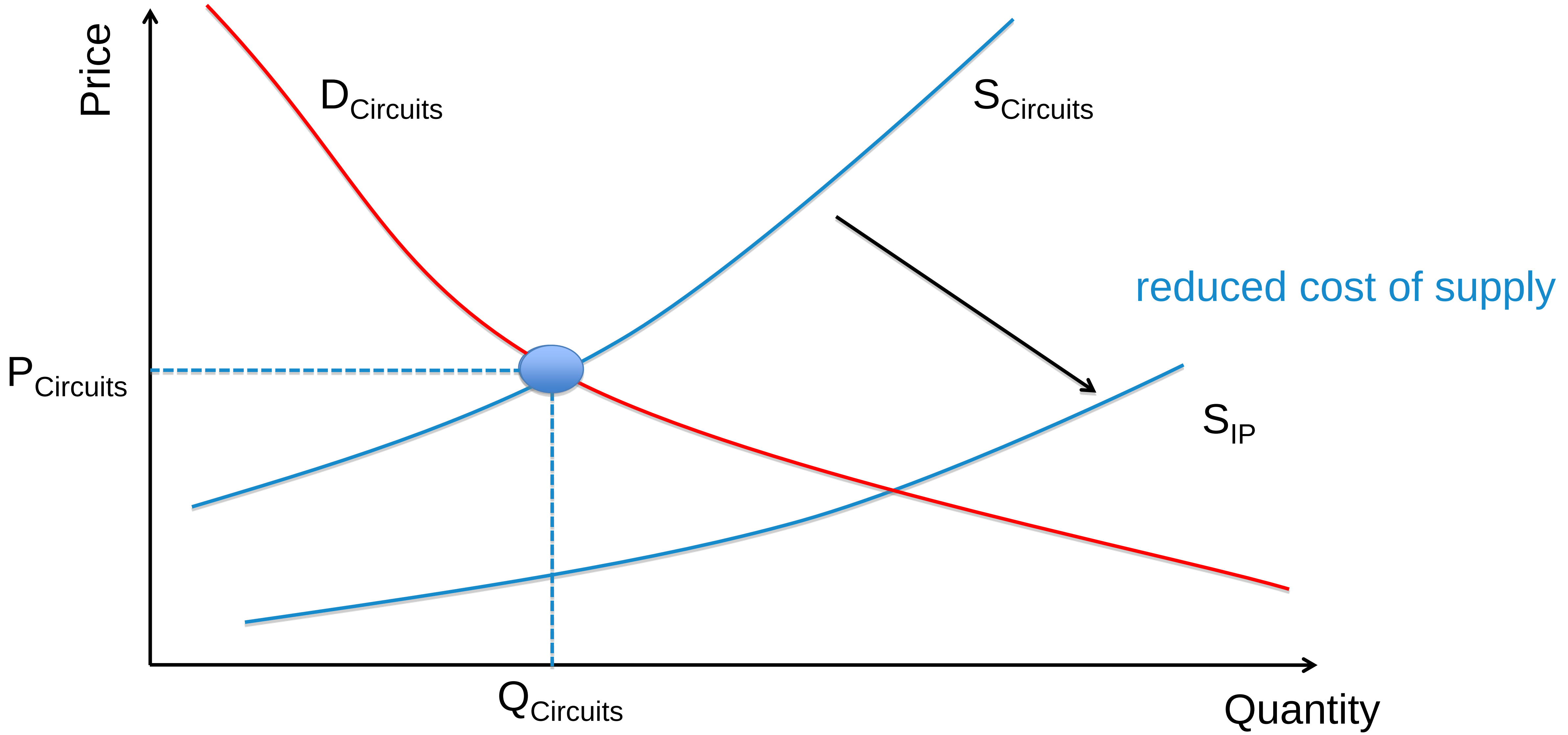
# The Supply Demand Schedule

## Equilibrium Point



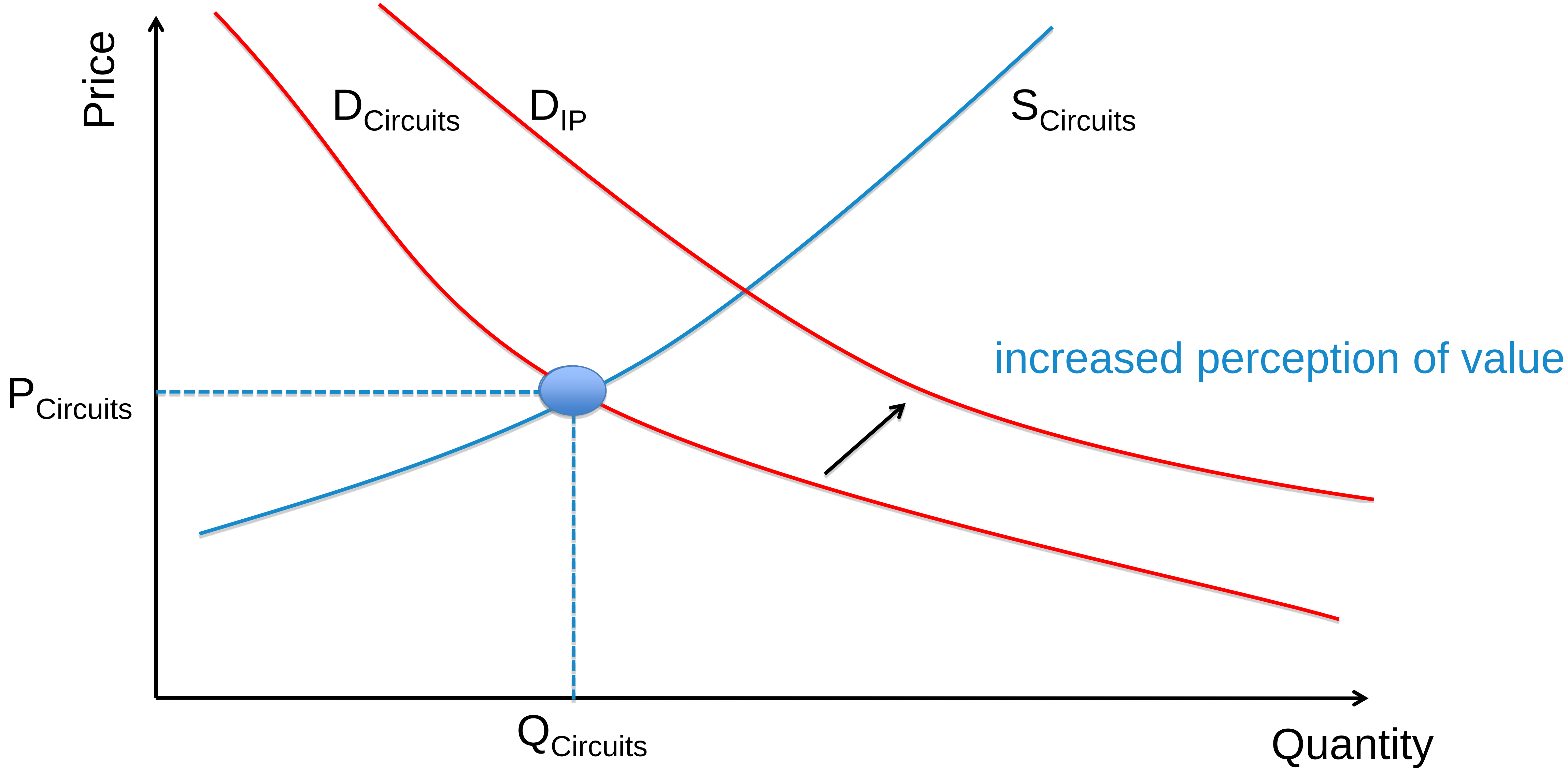
# The Supply Schedule Shift

## Circuits to Packets Supply



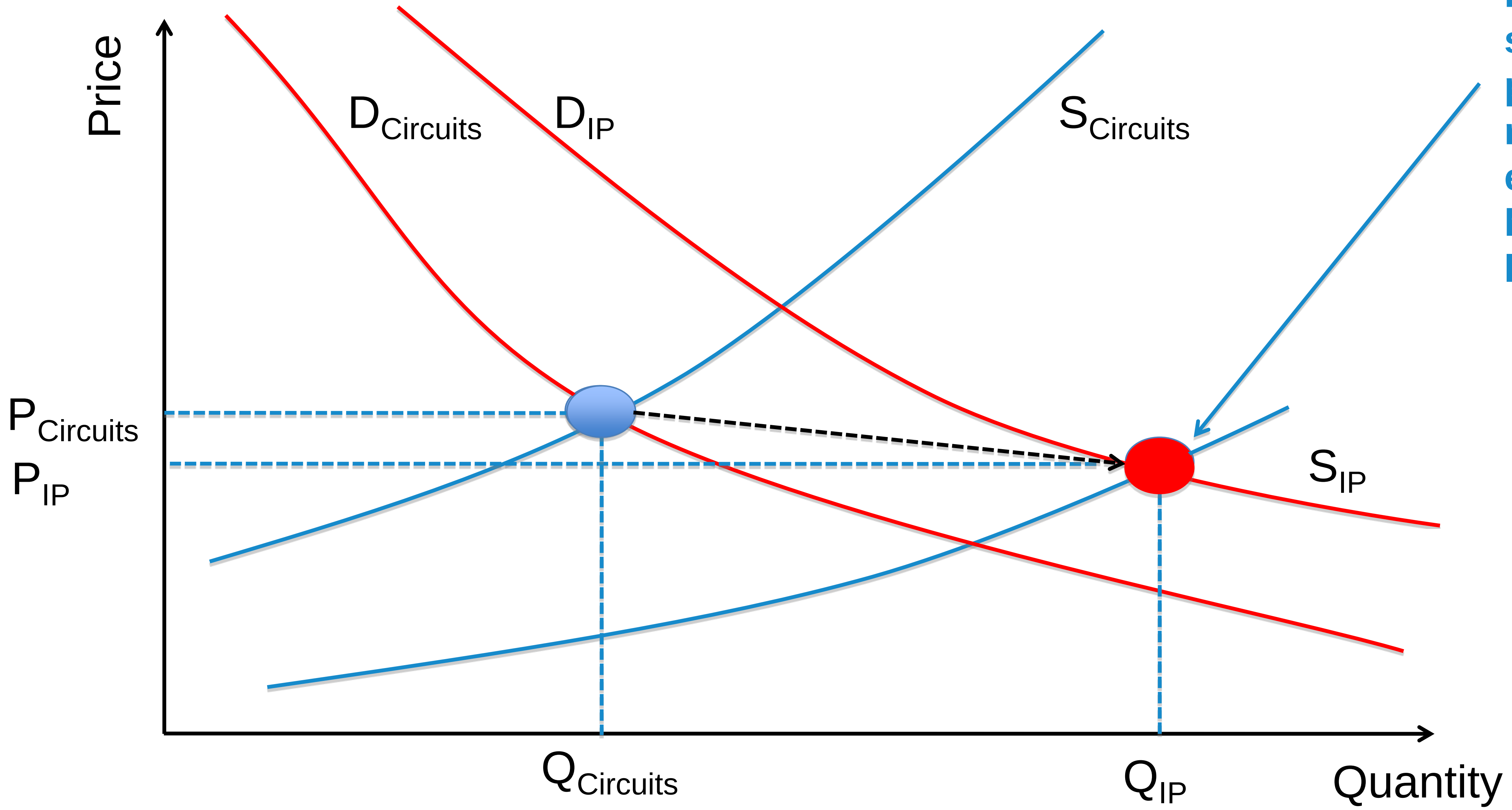
# The Demand Schedule Shift

## Circuits to Packets Demand



# The Supply Demand Schedule

## Circuits to Packets the new equilibrium




**reduced cost of supply, and increased perception of value, resulting in a new equilibrium point with higher quantity and lower unit price**

# IPv6 vs. IPv4

Are there any *competitive differentiators*?

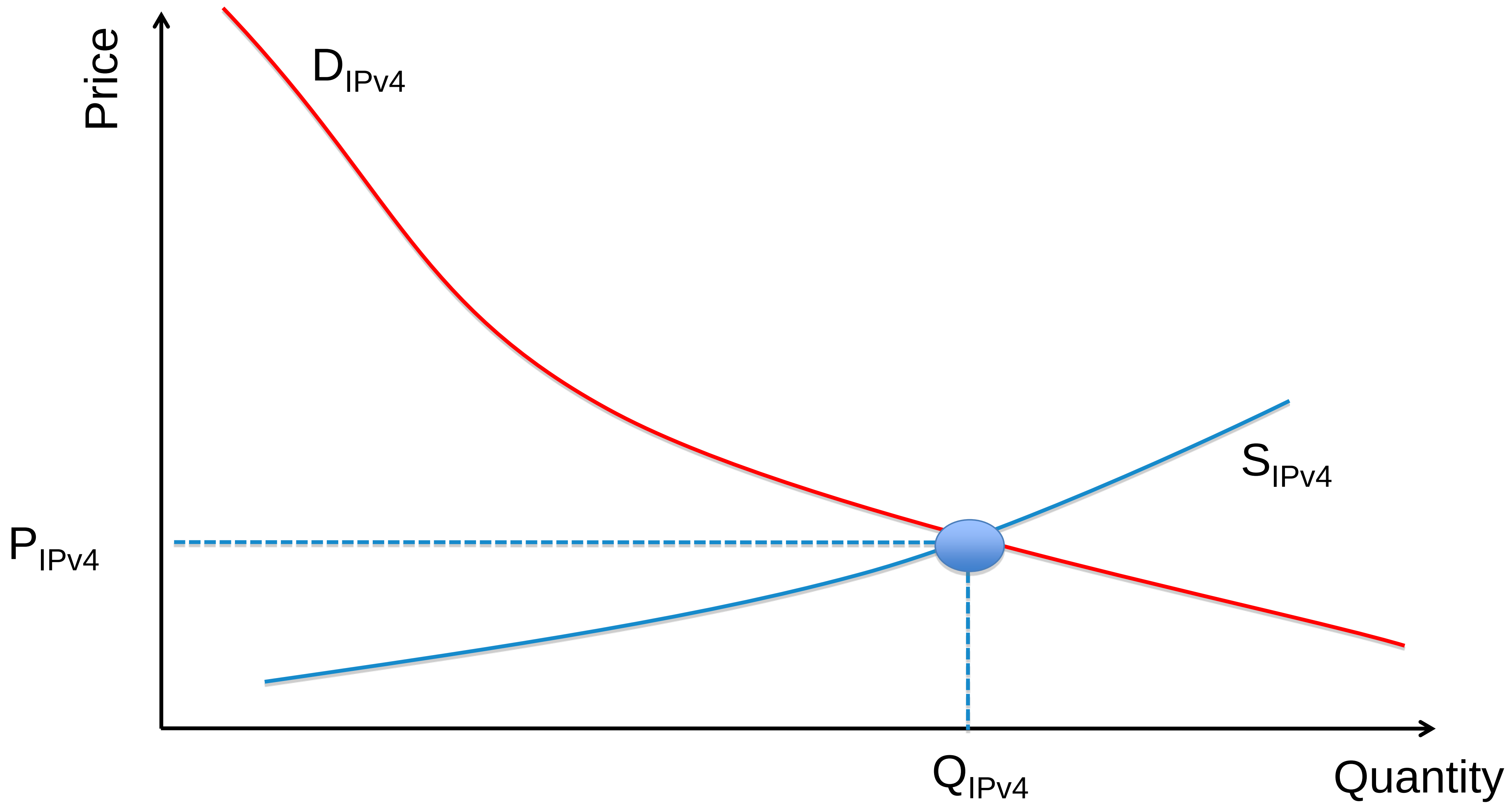
- Is the $cost_{v4}$ = $cost_{v6}$ ?
  - No, there is a cost associated with implementing IPv6
- Is the $functionality_{v4}$ = $functionality_{v6}$ ?
  - Yes, they are both transports, so no difference

Also:

- no inherent consumer-visible difference or demand
- hard to monetise IPv6
- adoption enables hedging by the provider against *future risk*

# The Supply Demand Schedule

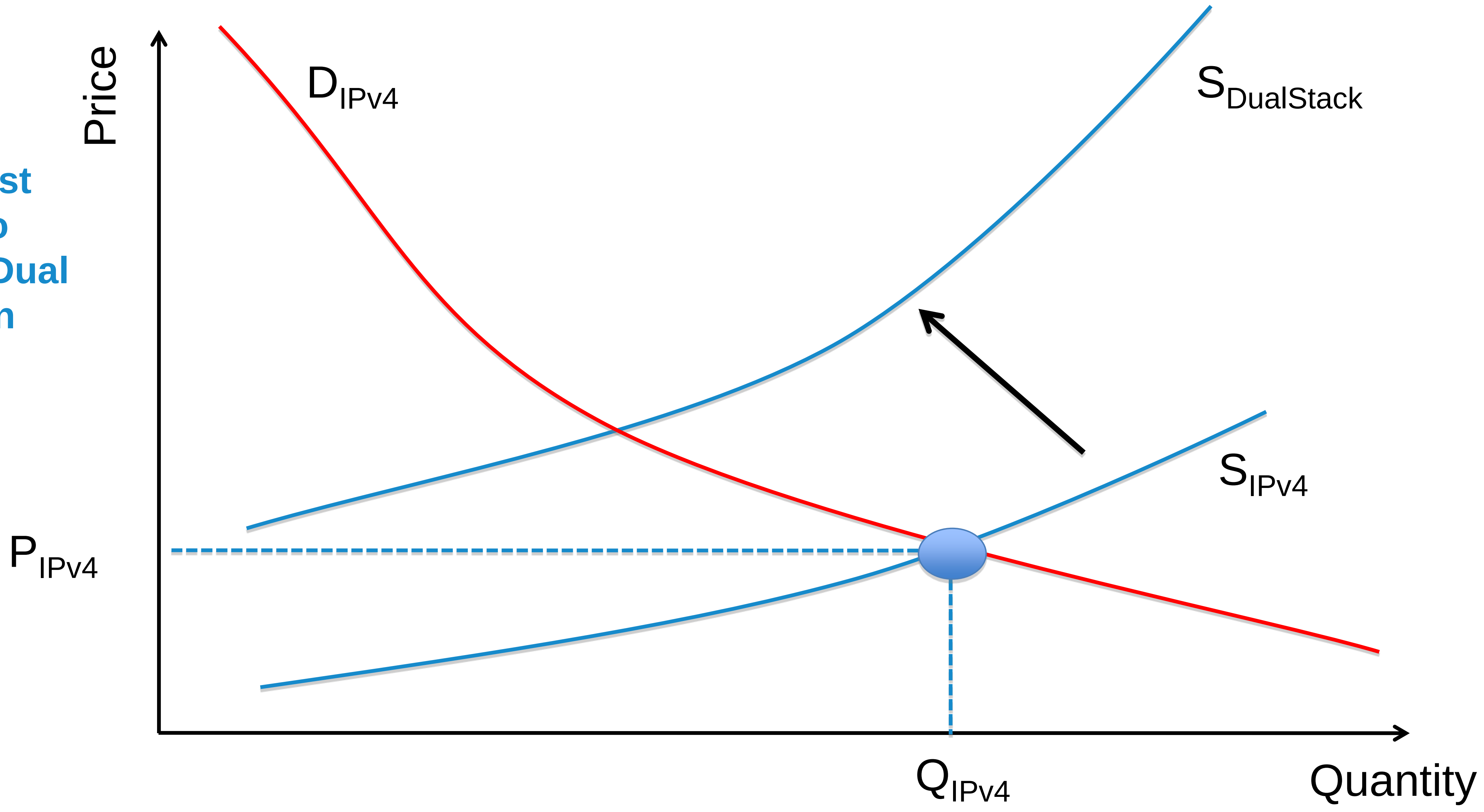
## Baseline - Existing IPv4



# The Supply Schedule Shift

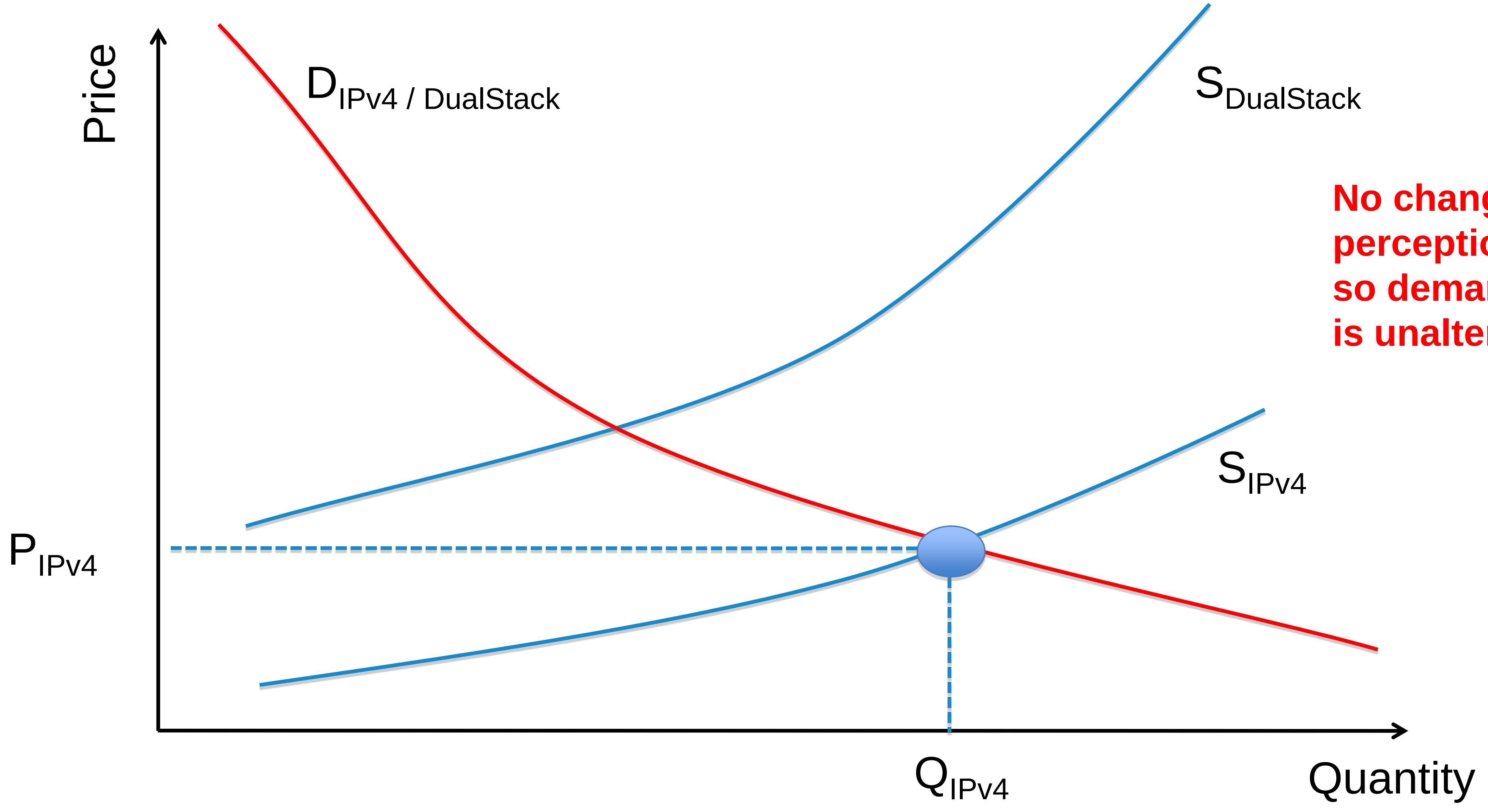
## Adopting Dual Stack Supply

**Supply side cost increase due to implementing Dual Stack operation**



# The Demand Schedule Shift

## Adopting Dual Stack Demand

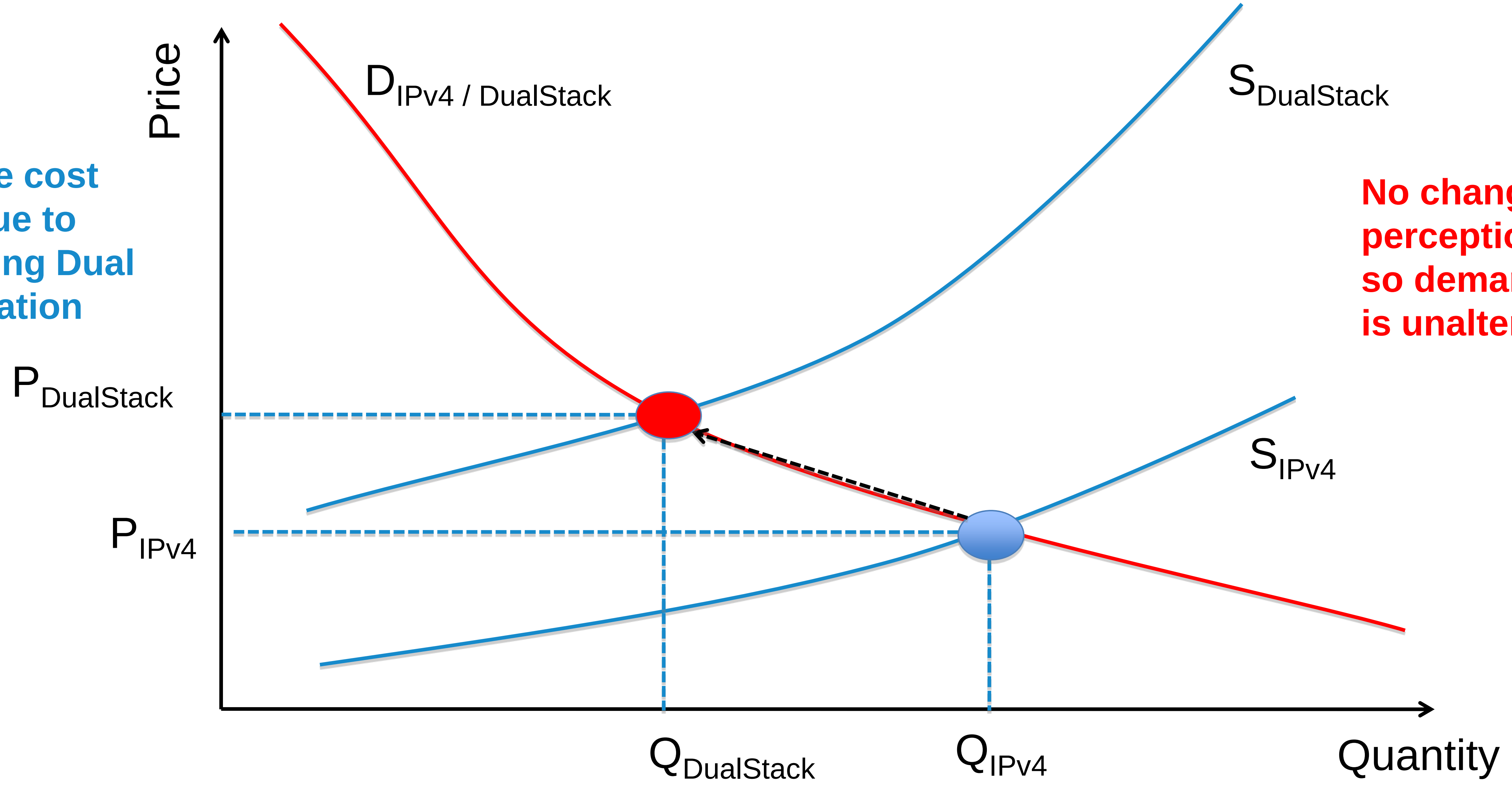



**No change in perception of value, so demand schedule is unaltered**

# The Supply Demand Schedule

## Adopting Dual Stack the new equilibrium

**Supply side cost increase due to implementing Dual Stack operation**




**No change in perception of value, so demand schedule is unaltered**

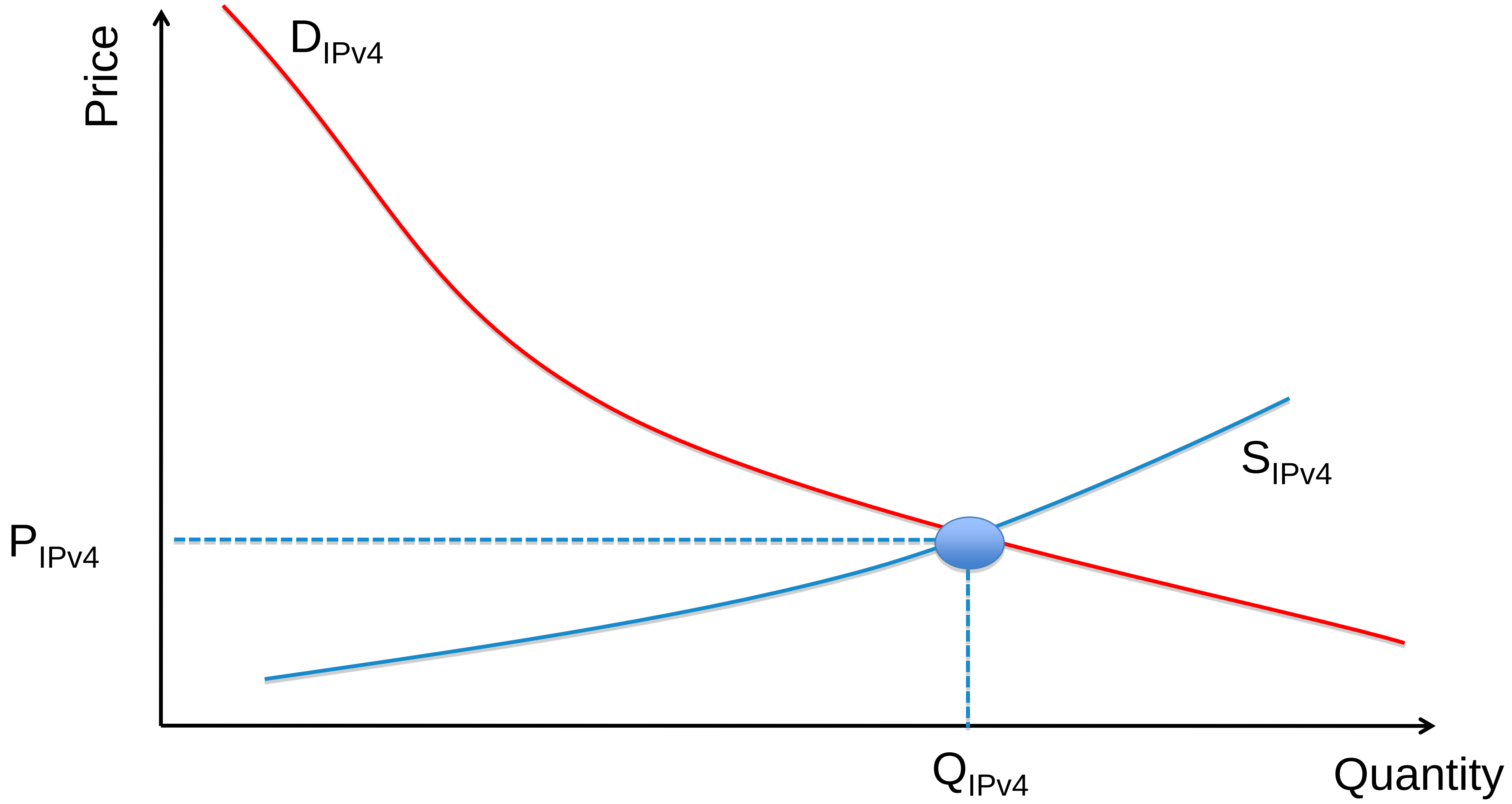
- **Equilibrium point is at a lower quantity if Dual Stack supply costs are passed on to customers**

# What about IPv4 Exhaustion?

- Does IPv4 address exhaustion change this picture?
- What are the economic implications of service providers adding NAT444 or NAT64 as a service offering ?
- Should we drive deeper NAT444 solutions ?

# The Supply Demand Schedule

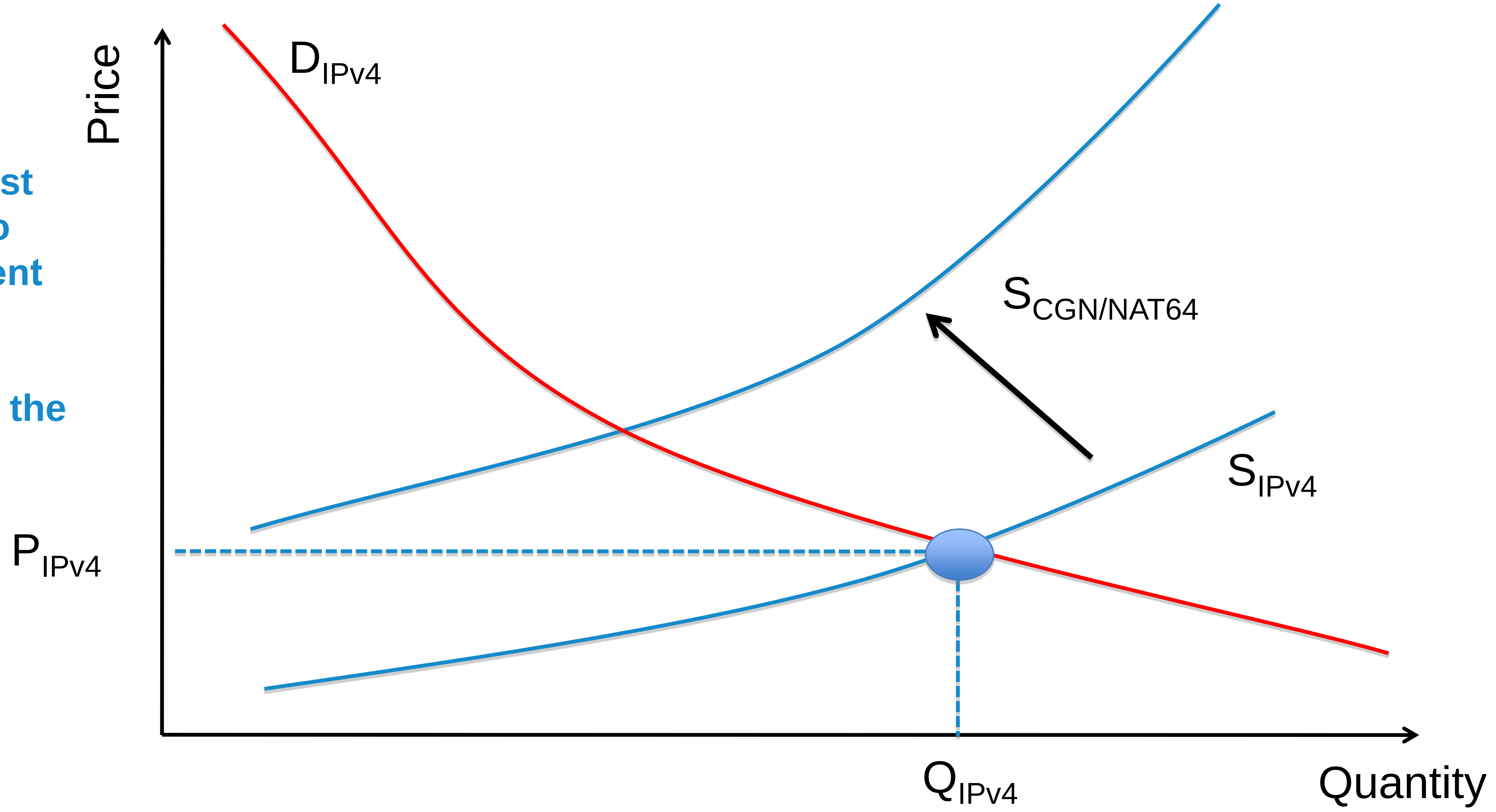
## Baseline - Existing IPv4



# The Supply Schedule Shift

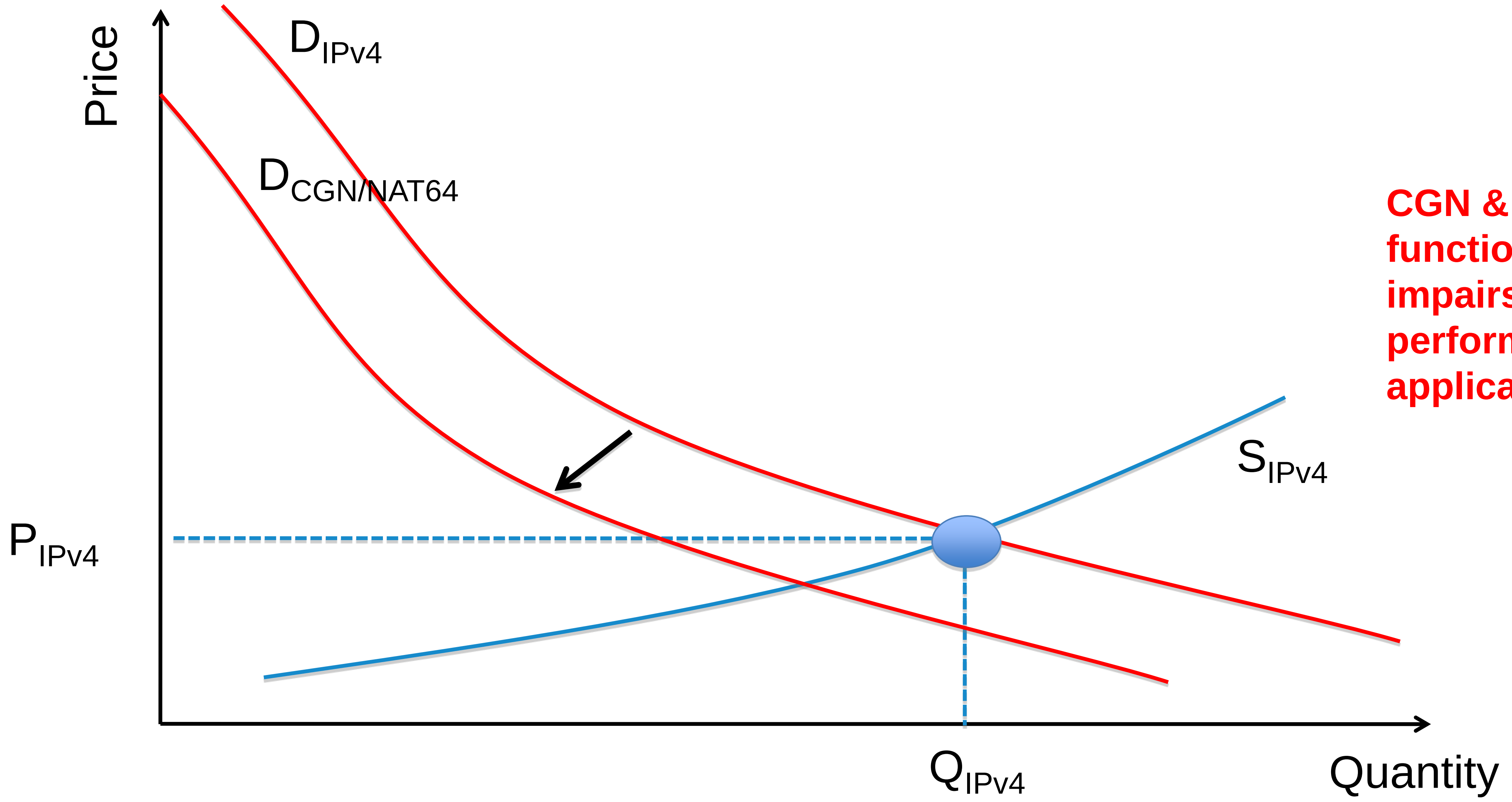
## Adding CGN / NAT64 Supply

**Supply side cost increase due to SP's requirement to deploy a CGN/NAT64 solution within the network's infrastructure**



# The Demand Schedule Shift

## Adding CGN / NAT64 Demand

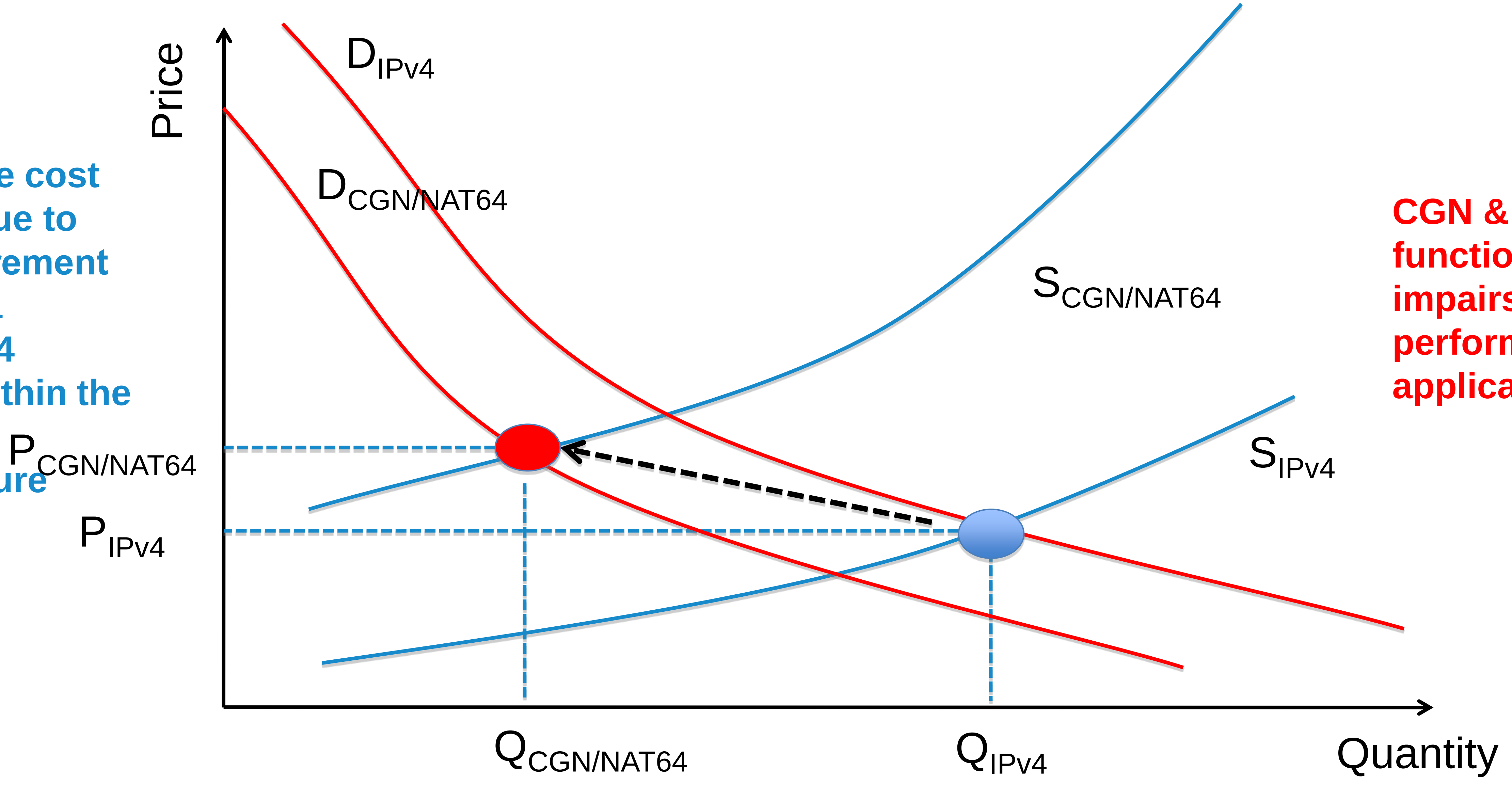



**CGN & NAT64 reduces functionality and impairs the performance of some applications**

# The Supply Demand Schedule

## Adding CGN / NAT64 the new equilibrium

**Supply side cost increase due to SP's requirement to deploy a CGN/NAT64 solution within the network's infrastructure**



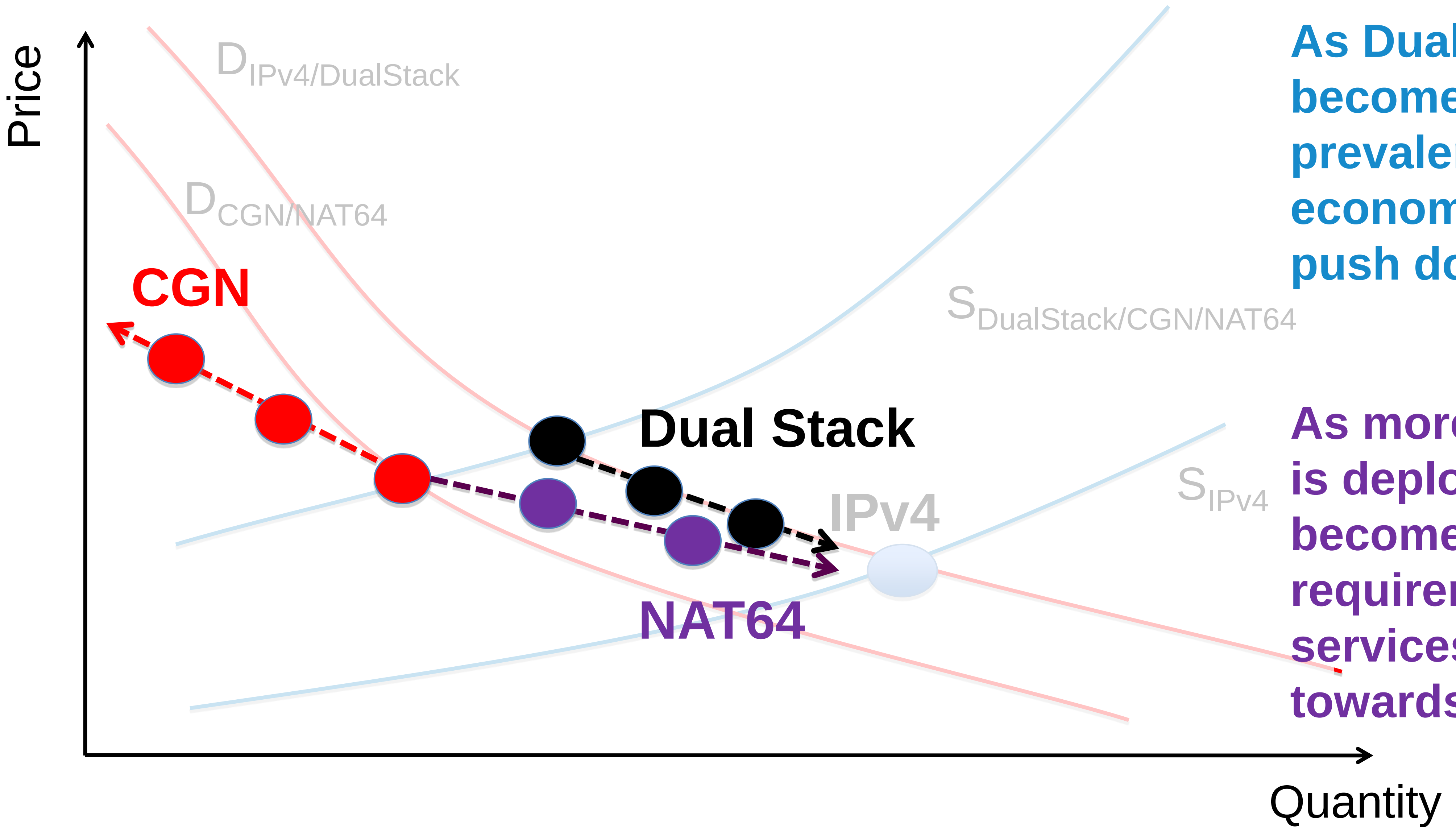

**CGN & NAT64 reduces functionality and impairs the performance of some applications**

- Equilibrium point of CGN & NAT64 represent higher cost and lower value for customers
- Even if costs are not passed on, cannot escape perceived change in service & issues

# The Schedule Shift Over Time

## CGN, NAT64 or Dual Stack?

As NAT compression becomes more intense the IPv4 CGN approach become decreasingly viable




As Dual Stack becomes more prevalent economies of scale push down costs

As more native IPv6 is deployed NAT64 becomes less of a requirement. NAT64 services trend towards Dual Stack

# Conclusion

- The market will go through a transitional phase before stability is reached
  - Should consumers pay more for the same, or a lesser service than they get today ?
- Dual Stack is the better long term option (This should be your goal)
- Translation is evil, though some are a bit more evil than others
  - Pick your translation technology carefully. Only apply it where you must.
- CGN/NAT444 is a short term fix that buys you the time to do an IPv6 deployment
  - Long term CGN/NAT444 deployments will only get worse and more expensive over time.
- NAT64 is a hybrid of both CGN/NAT444 and Dual Stack, though becomes better over time as more native IPv6 becomes available and there is less dependency on NAT64
  - NAT64 is painful now, though gets better and cheaper over time
- Irrespective if your choice the CGSE will help you get there ☺

# Summary

- CGv6 Overview
- Introduction to CGSE
- Configurations
- NAT64
  - 6rd
  - DS-Lite
- Deployment Options
- IPv6 Economics

# Q & A

# Complete Your Online Session Evaluation

## Give us your feedback and receive a Cisco Live 2013 Polo Shirt!

Complete your Overall Event Survey and 5 Session Evaluations.

- Directly from your mobile device on the Cisco Live Mobile App
- By visiting the Cisco Live Mobile Site www.ciscoliveaustralia.com/mobile
- Visit any Cisco Live Internet Station located throughout the venue

Polo Shirts can be collected in the World of Solutions on Friday 8 March 12:00pm-2:00pm

Cisco live! 365

Don't forget to activate your Cisco Live 365 account for access to all session material, communities, and on-demand and live activities throughout the year. Log into your Cisco Live portal and click the "Enter Cisco Live 365" button.
www.ciscoliveaustralia.com/portal/login.ww

Cisco live!

# Additional Slides

## Netflow v9 Templates

# NAT44

## Template 256: Add event

| Field ID | Attribute | Value |
|---|---|---|
| 234 | Incoming VRF ID | 32 bit ID |
| 235 | Outgoing VRF ID | 32 bit ID |
| 8 | Source IP Address | IPv4 Address |
| 225 | Translated Source IP Address | IPv4 Address |
| 7 | Source Port | 16 bit port |
| 227 | Translated Source Port | 16 bit port |
| 4 | L4 Protocol | 8bit value |
| | Total size | 21 Bytes |

68 records per 1500 export packet

# NAT44

## Template 257: Delete event

| Field ID | Attribute | Value |
| --- | --- | --- |
| 234 | Incoming VRF ID | 32 bit ID |
| 8 | Source IP Address | IPv4 Address |
| 7 | Source Port | 16 bit port |
| 4 | L4 Protocol | 8bit value |
| | Total size | 11 Bytes |

# NAT44

## Template 265: Add event w/ Bulk Port Allocation

| Field ID | Attribute | Value |
|---|---|---|
| 234 | Incoming VRF ID | 32 bit ID |
| 235 | Outgoing VRF ID | 32 bit ID |
| 8 | Source IP Address | IPv4 Address |
| 225 | Translated Source IP Address | IPv4 Address |
| 295 | Translated Source Port - Start | 16 bit port |
| 296 | Translated Source Port - Stop | 16 bit port |
| | Total size | 20 Bytes |

In 4.3.0: Field 295 and 296 will be replaced by 361 and 362 respectively

# NAT44

## Template 266: Delete event w/ Bulk Port Allocation

| Field ID | Attribute | Value |
|---|---|---|
| 234 | Incoming VRF ID | 32 bit ID |
| 8 | Source IP Address | IPv4 Address |
| 295 | Translated Source Port - Start | 16 bit port |
| | Total size | 10 Bytes |

In 4.3.0: Field 295 will be replaced by 361

# NAT44

## Template 271: Add w/ Destination Based Logging

| Field ID | Attribute | Value |
|---|---|---|
| 234 | Incoming VRF ID | 32 bit ID |
| 235 | Outgoing VRF ID | 32 bit ID |
| 8 | Source IP Address | IPv4 Address |
| 225 | Translated Source IP Address | IPv4 Address |
| 7 | Source Port | 16 bit port |
| 227 | Translated Source Port | 16 bit port |
| 12 | Destination Address | 4 bytes |
| 11 | Destination Port | 2 bytes |
| 4 | L4 Protocol | 8bit value |
| | Total size | 27 Bytes |

# NAT44

## Template 272: Delete w/ Destination Based Logging

| Field ID | Attribute | Value |
|---|---|---|
| 234 | Incoming VRF ID | 32 bit ID |
| 8 | Source IP Address | IPv4 Address |
| 7 | Source Port | 16 bit port |
| 12 | Destination Address | 4 bytes |
| 11 | Destination Port | 2 bytes |
| 4 | L4 Protocol | 8bit value |
| | Total size | 17 Bytes |

# NAT64

## Template 258: Add event

| Field ID | Attribute | Value |
|---|---|---|
| 27 | Source IPv6 Address | IPv6 Address |
| 225 | Translated Source IP Address | IPv4 Address |
| 7 | Source Port | 16 bit port |
| 227 | Translated Source Port | 16 bit port |
| 4 | L4 Protocol | 8bit value |
| | Total size | 25 Bytes |

# NAT64

## Template 259: Delete event

| Field ID | Attribute | Value |
|---|---|---|
| 27 | Source IPv6 Address | IPv6 Address |
| 7 | Source Port | 16 bit port |
| 4 | L4 Protocol | 8bit value |
| | Total size | 19 Bytes |

# NAT64

## Template 260: Add w/ Destination Based Logging

| Field ID | Attribute | Value |
|---|---|---|
| 27 | Source IPv6 Address | IPv6 Address |
| 225 | Translated Source IP Address | IPv4 Address |
| 28 | Destination IPv6 Address | IPv6 Address |
| 226 | Translated IPv4 Address | IPv4 Address |
| 7 | Source Port | 16 bit port |
| 227 | Translated Source Port | 16 bit port |
| 11 | Destination Port | 2 bytes |
| 4 | L4 Protocol | 8bit value |
| | Total size | 47 Bytes |




Cisco live!

# NAT64

## Template 261: Delete w/ Destination Based Logging

| Field ID | Attribute | Value |
|---|---|---|
| 27 | Source IPv6 Address | IPv6 Address |
| 28 | Destination IPv6 Address | IPv6 Address |
| 7 | Source Port | 16 bit port |
| 11 | Destination Port | 2 bytes |
| 4 | L4 Protocol | 8bit value |
| | Total size | 37 Bytes |

# DS Lite

## Template 267: Add event

| Field ID | Attribute | Value |
|---|---|---|
| 234 | Ingress VRF ID | 32 bit ID |
| 235 | Egress VRF ID | 32 bit ID |
| 8 | Inside IPv4 Address | IPv4 Address |
| 27 | Inside IPv6 Address | IPv6 Address |
| 225 | Translated IPv4 Address | IPv4 Address |
| 7 | Inside Source Port | 16 bit port |
| 227 | Translated Source Port | 16 bit port |
| 4 | L4 Protocol | 8bit value |
| | Total size | 33 Bytes |

# DS Lite

## Template 270: Delete event

| Field ID | Attribute | Value |
|---|---|---|
| 234 | Ingress VRF ID | 32 bit ID |
| 8 | Inside IPv4 Address | IPv4 Address |
| 27 | Inside IPv6 Address | IPv6 Address |
| 7 | Inside Source Port | 16 bit port |
| 4 | L4 Protocol | 8bit value |
| | Total size | 27 Bytes |

# DS Lite

## Template 269: Add w/ Bulk Port Allocation

| Field ID | Attribute | Value |
|---|---|---|
| 234 | Ingress VRF ID | 32 bit ID |
| 235 | Egress VRF ID | 32 bit ID |
| 8 | Inside IPv4 Address | IPv4 Address |
| 27 | Inside IPv6 Address | IPv6 Address |
| 225 | Translated IPv4 Address | IPv4 Address |
| 7 | Inside Source Port | 16 bit port |
| 295 | Translated Source Port - Start | 16 bit port |
| 296 | Translated Source Port - Stop | 16 bit port |
| | Total size | 38 Bytes |

In 4.3.0: Field 295 and 296 will be replaced by 361 and 362 respectively

# DS Lite

## Template 271: Delete event w/ Bulk Port Allocation

| Field ID | Attribute | Value |
| --- | --- | --- |
| 234 | Ingress VRF ID | 32 bit ID |
| 8 | Inside IPv4 Address | IPv4 Address |
| 27 | Inside IPv6 Address | IPv6 Address |
| 295 | Translated Source Port - Start | 16 bit port |
| | Total size | 26 Bytes |

In 4.3.0: Field 295 and 296 will be replaced by 361 and 362 respectively

# DS Lite

## Template 273: Add w/ Destination Based Logging

| Field ID | Attribute | Value |
|---|---|---|
| 234 | Ingress VRF ID | 32 bit ID |
| 235 | Egress VRF ID | 32 bit ID |
| 8 | Inside IPv4 Address | IPv4 Address |
| 27 | Inside IPv6 Address | IPv6 Address |
| 225 | Translated IPv4 Address | IPv4 Address |
| 7 | Inside Source Port | 16 bit port |
| 227 | postNAPTSourceTransportPort | 16 bit port |
| 12 | Destination Address | 4 bytes |
| 11 | Destination Port | 2 bytes |
| 4 | L4 Protocol | 8bit value |
| | Total size | 43 Bytes |

# DS Lite

## Template 274: Delete event w/ Destination Based Logging

| Field ID | Attribute | Value |
|---|---|---|
| 234 | Ingress VRF ID | 32 bit ID |
| 8 | Inside IPv4 Address | IPv4 Address |
| 27 | Inside IPv6 Address | IPv6 Address |
| 7 | Inside Source Port | 16 bit port |
| 4 | L4 Protocol | 8bit value |
| | Total size | 27 Bytes |