ROUTING SECURITY WORKSHOP

Taipei, Taiwan
09-11 December 2019

1

Agenda

- Internet Routing Basics
- IPv6 Address Planning
- BGP Fundamentals
- BGP Attributes
- BGP Scaling
- BGP Policy Control
- BGP Operations and Security Practices
- Multihoming
- RPKI and ROV

2

Acknowledgement:

- Philip Smith
- Cisco Systems

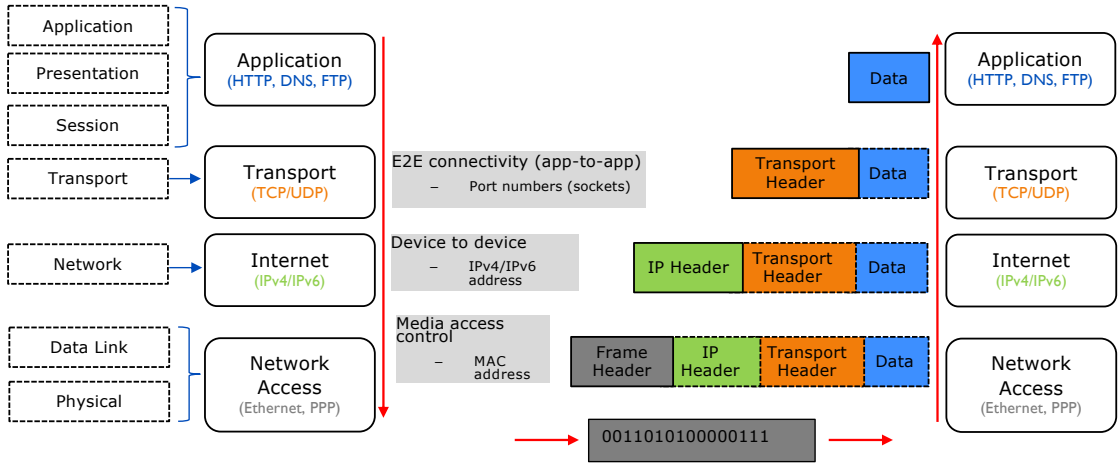
3

Module 1:

INTERNET ROUTING BASICS

4

Back to basics ☺



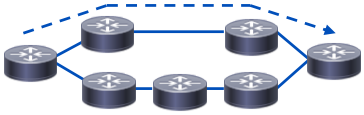
Addressing is the key**!**

Internet/Network Layer

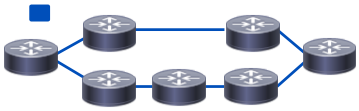
- Host to host communication across networks
  - Addressing
    - unique and hierarchical network-wide address
  - Routing
    - the best path to the destination
- Current protocols
  - IPv4 and IPv6

L3 Device/Router

- L3 device gets the packet one step closer
  - The next hop to reach the destination!



- Router
  - Exchanges network information
  - Finds the best path to a destination, and
  - Forwards the packet to the next hop (a step closer) to reach the destination



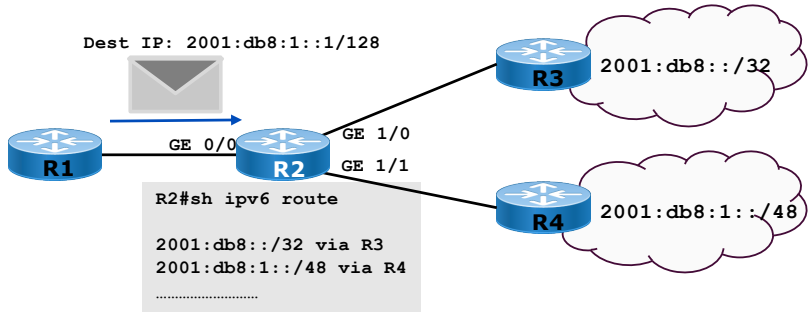
7

Best path lookup – Routing Decision

- Inspects the destination address of the packet
  - Network portion
- Looks up its routing table for a “best match”
  - Longest matching left-most bits
- If no match, checks for default route
  - If no default route, drop the packet!

8

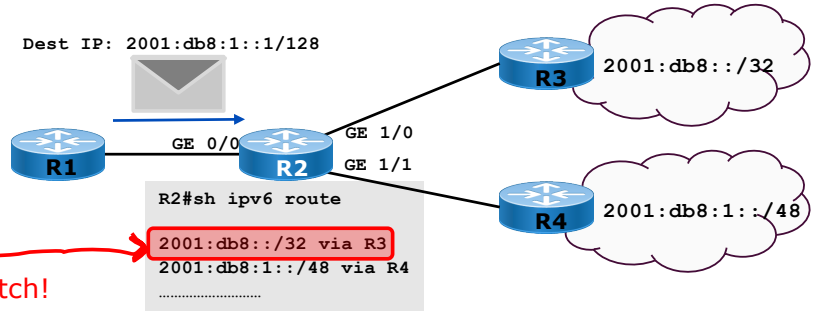
Best path (route) lookup



2001:db8::/32	0010000000000001:1101101110000000::
2001:db8:1::/48	0010000000000001:1101101110000000:0000000000000001::

9

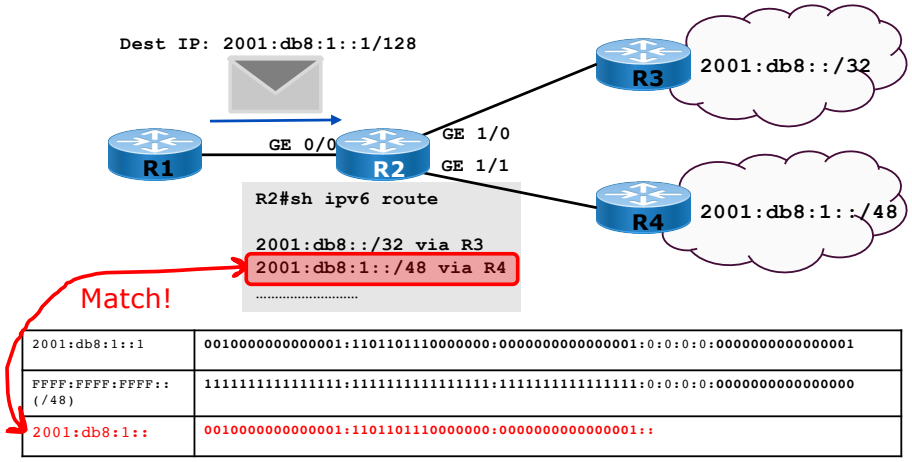
Best path (route) lookup



2001:db8:1::1	0010000000000001:1101101110000000:0000000000000001:0:0:0:0:0000000000000001	AND
FFFF:FFFF:: (/32)	1111111111111111:1111111111111111:0000000000000000:0:0:0:0:0000000000000000	
2001:db8::	0010000000000001:1101101110000000::	

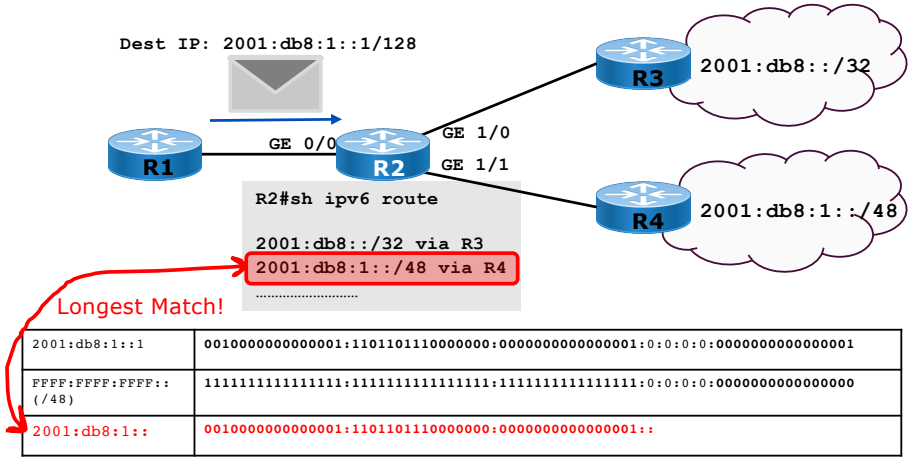
10

Best path (route) lookup



11

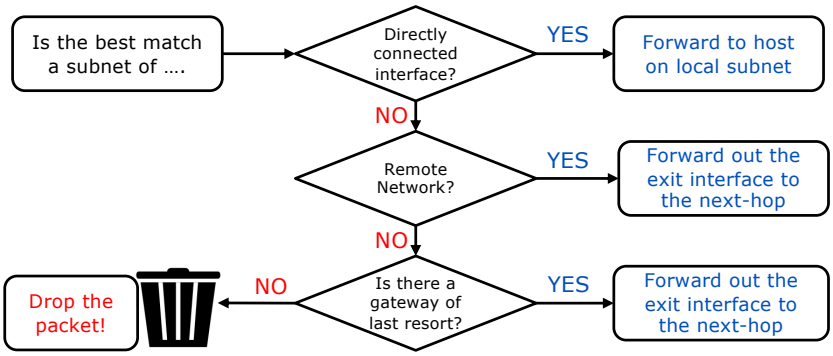
Best path (route) lookup



12

Packet Forwarding

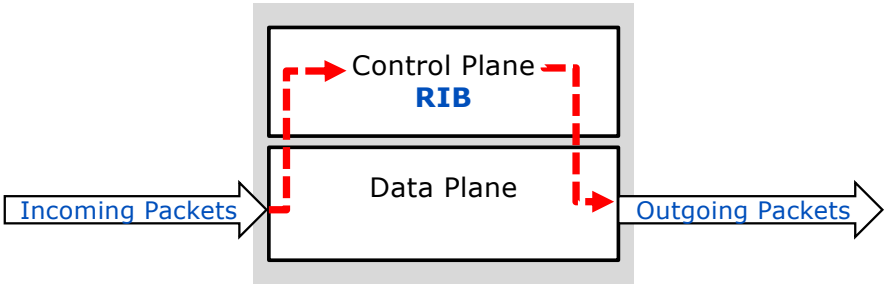
- If a best match is found, the router determines
  - the correct exit interface to reach the next-hop/destination



13

Process vs Hardware Switching

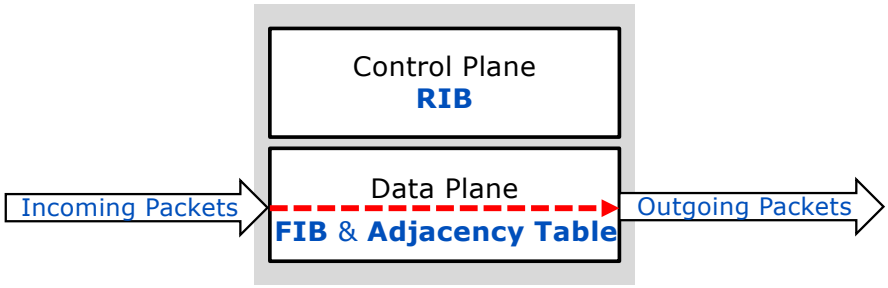
- Incoming packet forwarded to the control plane (CPU)
  - routing table (RIB) lookup, frame re-write (next-hop MAC), and forwarded to the exit interface



14

Process vs Hardware Switching

- Instead two hardware based tables
  - FIB derived from the RIB- all destinations and next-hops
  - Adjacency table from the ARP table- L2 header info for each next-hop in the FIB



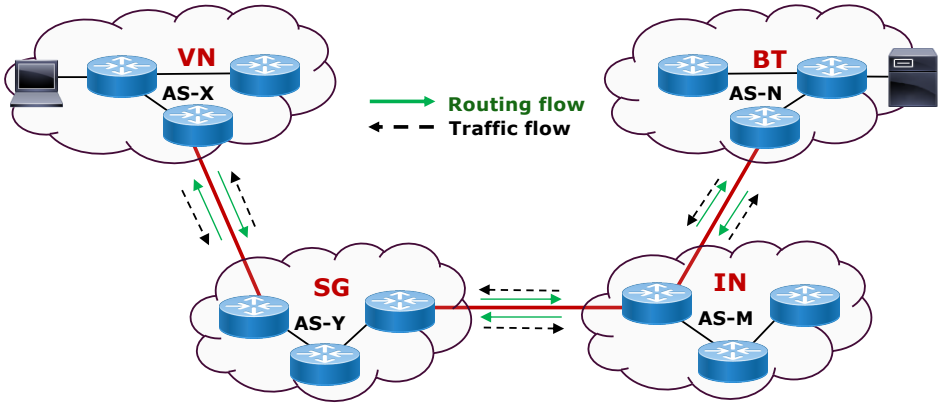
15

Internet Routing

- How does a user in India access a service hosted in the Taiwan?
  - The ISP in IN could directly connect to the ISP in TW
    - Neither scalable nor economical
  - Instead, the IN ISP shares its network information with its neighbor ISPs
  - The ISP in TW does the same with its own neighbors
  - Neighbor ISPs propagate the information to their neighbors, and so on…
    - Eventually, they both learn about each other's network!

16

Internet Routing

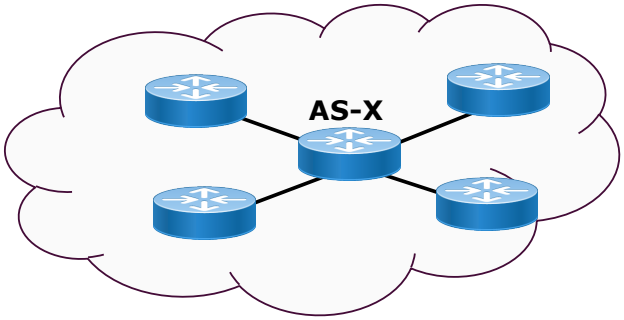


Exchange of network information – Routing
Networks (ASes) connected together – Internet

17

Autonomous System (AS)

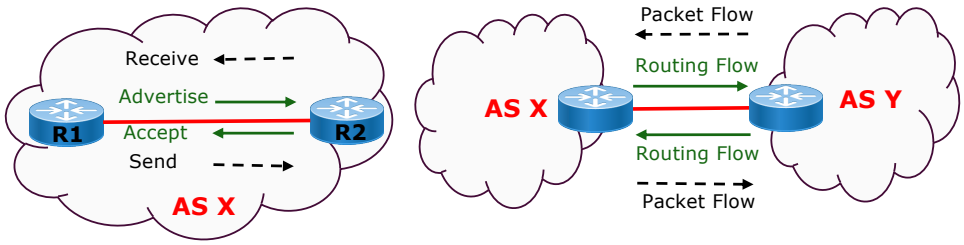
- A group of networks with the same routing policy (external)
  - Usually under single administrative control



18

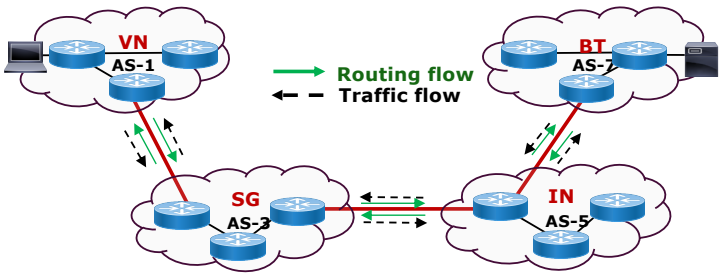
Routing Flow & Traffic Flow

- Traffic and network info always flow in opposite direction!
  - network info exchanged in both directions for bi-directional traffic flow



Routing & Traffic Flow: Internet

- For user (N1) in AS1 to send traffic to user (N7) in AS7:
  - AS7 must originate and announce N7 to AS5.
  - AS5 must accept N7 from AS7, and advertise to AS3.
  - AS3 must accept and forward N7 to AS1
  - AS1 must accept N7 from AS3



Routing Policy

- To manipulate/control traffic flow in/out of a network
  - manipulate inbound routing info to influence outgoing traffic
  - manipulate outbound routing info to influence incoming traffic

Routing Protocols

- How do routers exchange network information with each other?
  - Routing Protocols!
  - IGP & EGP

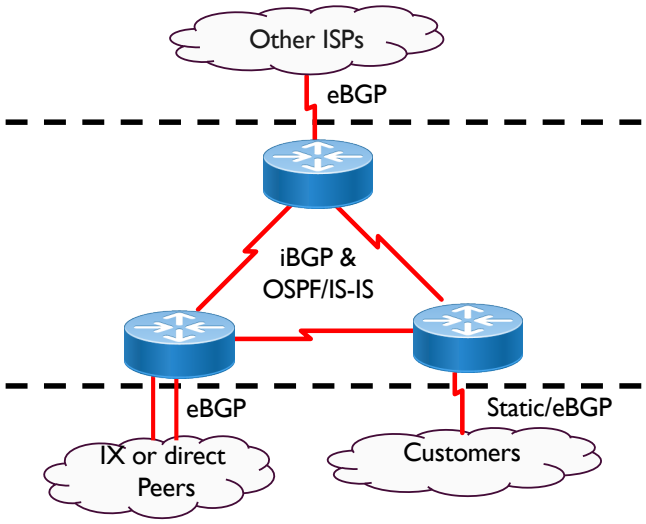
Interior Gateway Protocol (IGP)

- To exchange network info within an AS
  - Allows all routers within an AS to learn about each other
  - To carry infrastructure information (loopbacks & ptp)
    - No customer routes!
  - The design goal is scalability and fast convergence
    - Hence, minimise the number of prefixes carried in IGP!
- Two most widely used IGPs in operator networks
  - OSPF & IS-IS
    - Uses the SPF algorithm
    - Best path selection based on lowest cost/metric
    - Supports hierarchical routing – scalability!

Exterior Gateway Protocol (EGP - BGP)

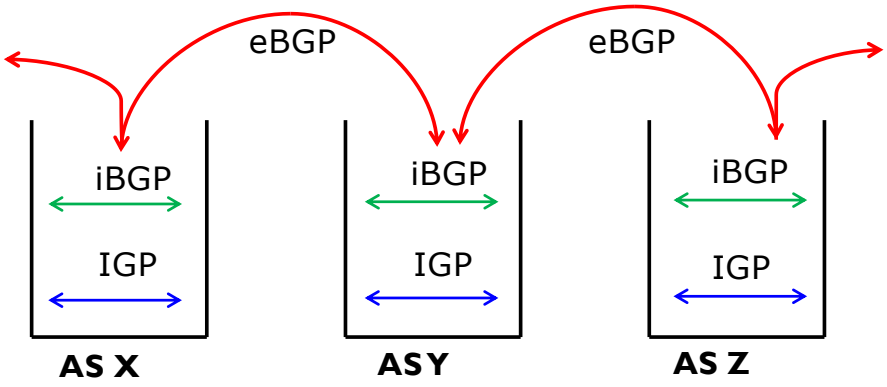
- To exchange network information between ASes
  - Implement routing policies (manipulate traffic path)
  - Define administrative boundary
- BGP is the de facto EGP!

Routing Protocols Hierarchy



25

How it all works



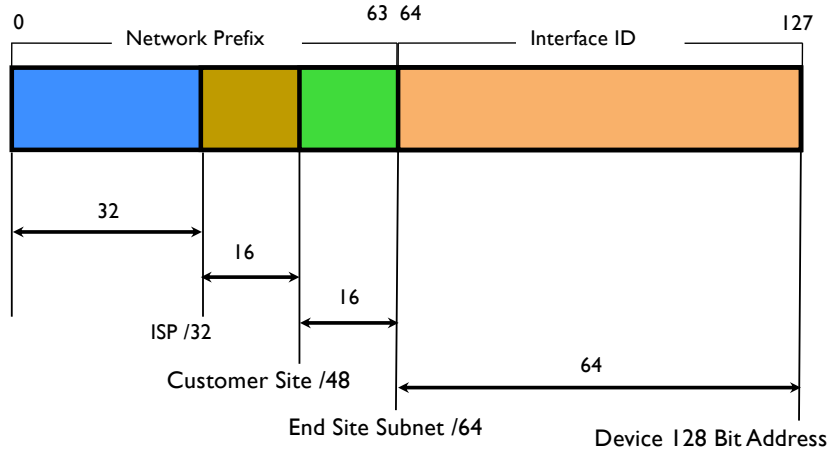
Barry Greene & Philip Smith "Cisco ISP Essentials"

26

27

28

IPv6 Address Recap



29

IPv6 Address Planning

- Network Operators allocated /32 by RIRs
- Global Routing prefix /48
  - /56 (ISPs to end site)
  - upstream could filter anything smaller
  - Consider the routing table size!

30

IPv6 Address Planning

- Future traffic engineering needs?
  - Contiguous assignment vs Split assignment
- Shift in thought:
  - IPv4: number of hosts ☹
  - IPv6: number of subnets!

31

IPv6 Address Plan: ISP Infra

- Loopbacks
- Point-to-Point links
- Internal Server LAN
  - also called NOC LAN
  - not seen from outside
- External Server LAN
  - Mail, DNS, etc

32

IPv6 Address Plan: ISP Infra

- Dedicate a /40 (or /48) for the backbone infra
  - Every infrastructure assignment from this block!
  - Carried by IGP (NOT iBGP)
- Loopbacks
  - Generally one /48 (/60 and /64 also common) for all loopbacks
  - /128 as loopback
- Point-to-Point links (Ex: /48 for all P2P links)
  - Assign /64 per link (RFC3177); RFC6164/6547 recommends /127
    - Reserve /64 per link but use /127

33

IPv6 Address Plan: ISP Infra

- Internal Server/NOC LAN
  - /60 (if different subnets within the NOC), or
  - /64
- External Server LAN
  - /64 (allows up to 2^64 services to be hosted)

34

IPv6 Addr Plan: Enterprise Customer

- RFC6177/BCP157:
  - Obsoletes RFC3177 specification, which recommended end-sites should get a /48 by default
  - Suggests that end-sites are still given more than a /64 to allow for future growth (multiple subnets)
    - Multiples of /64, or better still
    - /60, /56, or /52 (nibble boundaries)

35

IPv6 Addr Plan: Enterprise Customer

- Consider regional delegation
  - Aggregation in mind!
  - /40 per region?
- One /48 per customer
  - Could be transit customers or leased line customers
  - Could be given additional /48s as they grow (more than 65K subnets)
- We also see ISPs give:
  - /52 or /56 to mid-sized customers ()
  - /60 for very small customers
  - /64 to end sites NOT recommended!!

36

IPv6 Address Plan: Customer WAN links

- Dedicate a /48 block for customer WAN links
  - Helps to monitor customer links
  - Not to be mistaken with the trusted infra PtP block!
  - Actual addressing still the same:
    - Reserve /64 and use /127
- Carried in iBGP (not IGP)
  - Aggregated at the GW router or POP routers

37

IPv6 Address Plan: Broadband Customer

- Depends on your deployment
  - ND-RA for CPE WAN side
    - A /64 prefix on BRAS can still support 2^64 CPEs through SLAAC
  - DHCPv6-PD for CPE LAN side
    - A /48 pool on each BRAS (65k /64s can be delegated)
- Dedicate a /40 (or bigger) for Broadband network
  - /48s out of the /40 to each BRAS
  - Announced in iBGP by BRAS

38

IPv6 Address Plan: DC services

- DC infra blocks from your infra block
  - Loopbacks
  - PtP links
- dedicate /40 for Data Center (hosted) services
  - Depends on DC architecture
  - Dedicated VLAN/subnet per service?
    - /64 per VLAN/subnet (2^64 servers)
  - Dedicated subnet per customer (customer buys VMs/hosts services)?
    - /64 per customer or subnet (2^64 VMs)
  - Announced in iBGP (DC border router)

39

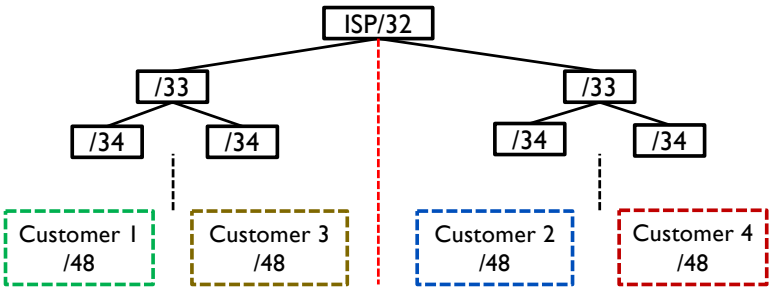
IPv6 Address Plan: Traffic Shaping

- Borrow from IPv4
  - sub-aggregates to shape traffic
  - Difficult with contiguous assignment
- Assign customer prefixes (that attract traffic) from both ends of address space
  - Infrastructure prefix do not attract traffic

40

IPv6 Address Plan: Traffic Shaping

- Customer prefixes assigned from each /33 sub-prefix
  - Similar to IPv4 sub-aggregates!
  - Allows us to balance incoming traffic



IPv6 Address Plan: Routing

- IGP to carry next-hop reachability information
  - Infrastructure blocks (PtPs, loopbacks)
  - Aggregation desirable in IGP
- Customer prefixes (Enterprise, broadband, DC customers/services)
  - Sub-aggregates for traffic shaping (mulithoming)
  - Consider regional delegation
  - iBGP carries all customer prefixes
    - Aggregation may interfere with traffic shaping
  - Aggregation necessary in eBGP (pull up routes)

IPv6 Address Planning: Example

- ISP has 2406:6400::/32 prefix
  - 16x /36s
  - easier to play at nibble boundaries

#	Prefix	Comments
1	2406:6400:0000::/36	First /33
2	2406:6400:1000::/36	
3	2406:6400:2000::/36	
4	2406:6400:3000::/36	
5	2406:6400:4000::/36	
6	2406:6400:5000::/36	
7	2406:6400:6000::/36	
8	2406:6400:7000::/36	
9	2406:6400:8000::/36	Second /33
10	2406:6400:9000::/36	
11	2406:6400:a000::/36	
12	2406:6400:b000::/36	
13	2406:6400:c000::/36	
14	2406:6400:d000::/36	
15	2406:6400:e000::/36	
16	2406:6400:f000::/36	

43

Example: High level plan

#	Prefix	Assignment	Comment
1	2406:6400:0000::/36	Infra + Cust	First /33
2	2406:6400:1000::/36	Customer	
3	2406:6400:2000::/36		
4	2406:6400:3000::/36		
5	2406:6400:4000::/36		
6	2406:6400:5000::/36		
7	2406:6400:6000::/36		
8	2406:6400:7000::/36		
9	2406:6400:8000::/36		Second /33
10	2406:6400:9000::/36		
11	2406:6400:a000::/36		
12	2406:6400:b000::/36		
13	2406:6400:c000::/36		
14	2406:6400:d000::/36		
15	2406:6400:e000::/36		
16	2406:6400:f000::/36		

44

Example: High Level

#	Prefix	Assignment	Comment
1	**2406:6400:0000::/36**	**Infra + Cust**	**First /33**
1	2406:6400:0000:0000::/40	Backbone Infra (PtP, Loopbacks)	
2	2406:6400:0100:0000::/40	Enterprise Customer Reg1	
3	2406:6400:0200:0000::/40	Broadband Region1	
4	2406:6400:0300:0000::/40	Future Customers	
5	2406:6400:0400:0000::/40		
6	2406:6400:0500:0000::/40		
7	2406:6400:0600:0000::/40		
:			
:			
:			
:			
16	2406:6400:0f00:0000::/40		

45

Example: High Level

#	Prefix	Assignment	Comments
9	**2406:6400:8000::/36**	**Customer**	**Second /33**
1	2406:6400:8000::/40	Broadband Region2	
2	2406:6400:8100::/40	Enterprise Customer Reg2	
3	2406:6400:8200::/40	Future Customers	
4	2406:6400:8300::/40		
5	2406:6400:8400::/40		
6	2406:6400:8500::/40		
7	2406:6400:8600::/40		
:			
:			
:			
:			
16	2406:6400:8f00::/40		

46

Example: Infrastructure

#	Prefix	Assignment	Comments
1	2406:6400:0000::/36	Infra + Cust	First /33
1	2406:6400:0000::/40	Backbone Infra	
1	2406:6400:0000::/48	Loopbacks	
2	2406:6400:0001::/48	Point-to-Point	
3	2406:6400:0002::/48	Future Infra use	
4	2406:6400:0003::/48		
5	2406:6400:0004::/48		
6	2406:6400:0005::/48		
:			
:			
:			
:			
256	2406:6400:00ff::/48		

47

Example: Customer

#	Prefix	Assignment	Comments
1	2406:6400:0000::/36	Infra + Cust	First /33
2	2406:6400:0100::/40	Enterprise Customer Reg1	
1	2406:6400:0100::/48	Customer WAN links	
2	2406:6400:0101::/48	Customer 1.1	
3	2406:6400:0102::/48	Customer 1.2	
4	2406:6400:0103::/48	Future Customers	
5	2406:6400:0104::/48		
6	2406:6400:0105::/48		
7	2406:6400:0106::/48		
8	2406:6400:0107::/48		
:			
:			
256	2406:6400:01ff::/48		

48

Example: Customer

#	Prefix	Assignment	Comments
1	2406:6400:0000::/36	Infra + Cust	First /33
3	2406:6400:0200::/40	Broadband Reg1	
1	2406:6400:0100::/48	BRAS 1	
2	2406:6400:0101::/48	BRAS 2	
3	2406:6400:0102::/48	BRAS 3	
4	2406:6400:0103::/48	Future Customers	
5	2406:6400:0104::/48		
6	2406:6400:0105::/48		
7	2406:6400:0106::/48		
8	2406:6400:0107::/48		
:			
:			
256	2406:6400:01ff::/48		

Example: Customer

#	Prefix	Assignment	Comments
9	2406:6400:8000::/36	Customer	Second /33
1	2406:6400:8000::/40	Broadband Reg2	
1	2406:6400:8000::/48	BRAS1	
2	2406:6400:8001::/48	BRAS2	
3	2406:6400:8002::/48	BRAS3	
4	2406:6400:8003::/48	Future BRAS	
5	2406:6400:8004::/48		
:			
:			
:			
256	2406:6400:80ff::/48		

Example: Customer

#	Prefix	Assignment	Comments
9	2406:6400:8000::/36	Customer	Second /33
2	2406:6400:8100::/40	Enterprise Customer Reg2	
1	2406:6400:8100::/48	Customer 2.1	
2	2406:6400:8101::/48	Customer 2.2	
3	2406:6400:8102::/48	Future Customers	
4	2406:6400:8103::/48		
5	2406:6400:8104::/48		
:			
:			
:			
256	2406:6400:81ff::/48		

51

52

Module 3:

BGP FUNDAMENTALS

53

Border Gateway Protocol - BGP

- Runs over TCP (port 179)
  - TCP connection required before BGP session
  - Need to be reachable!
- Path vector routing protocol
  - Best path selection based on path attributes
  - Route: destination and the attributes of the path to reach the destination
- Incremental BGP updates

54

Internal & External BGP

- eBGP used to:
  - Exchange networks/routes between ASes
    - Aggregates and sub-aggregates
  - Implement routing policies
    - To manipulate inbound and outbound traffic
- iBGP is used to:
  - Carry customer networks/prefixes
  - Internet routes (some or all) across the AS backbone

55

BGP Message Types

- Open:
  - After a TCP connection has been established between two BGP routers, an Open message is sent
    - Once the open message is confirmed (keepalive), the BGP session is established – become BGP peers/neighbors!
  - Contains:
    - Sender's ASN
    - BGP version
    - BGP router ID
    - Hold-time (3 x keepalive interval)

56

BGP Message Types

- Keepalive:
  - Exchanged initially to acknowledge Open messages
  - Exchanged periodically (60 secs) to maintain BGP session
    - Dataless packet
- Update:
  - BGP peers exchange network information through Update messages
    - One update for each path!
  - Contains:
    - Withdrawn routes – no more reachable
    - Path attributes – attributes for this path to reach the destinations specified by the NLRI
    - NLRI – list of networks reachable through this path <prefix, length>

57

BGP Message Types

- Notification:
  - Sent when an error condition is detected
  - The BGP session is torn down immediately!
  - Contains:
    - Error code
    - Error sub-code
    - Data related to error

58

BGP Neighbor States

A BGP router goes through six different states.

Idle

- The router is looking for a route to its neighbor

Connect

- BGP router moves from Idle to Connect state if it has found a route to its neighbor, and has started the TCP handshake
- If the TCP session successful, sends an Open message (and transitions to OpenSent)
- Else, move to Active state

Active

- A router transitions to Active state if the initial TCP connection was not successful (in Connect state)
- Restarts the TCP connection
- If successful, sends an Open message
- Else, falls back to Idle state

59

BGP Neighbor States

- A BGP router goes through six different states

OpenSent

- An Open message has been sent to the neighbor
- Waiting for Open message from neighbor
- If it receives an Open message and there are no mismatches (version, source addr same as TCP addr, ASN, router-ID, TTL, md5), sends KeepAlive, moves to OpenConfirm
- Else (if mismatches/errors), sent Notification and falls back to Idle

OpenConfirm

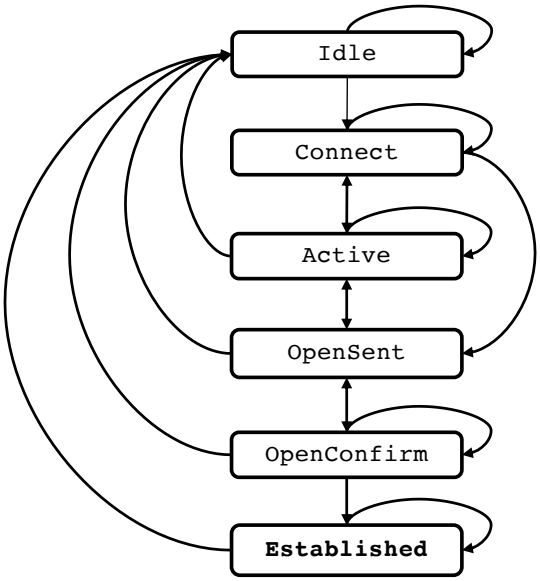
- waiting for the initial KeepAlive
- If received, transitions to Established
- If holdtimer expires or Notification received, moves to Idle

Established

- The BGP neighbor relationship (session) is established!
- Routing information can now be exchanged
- If holdtimer expires/error, moves back to Idle

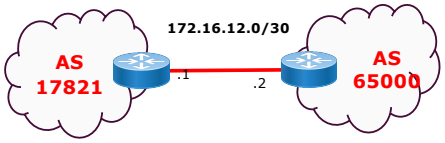
60

BGP State Machine



BGP Neighbor Relationship

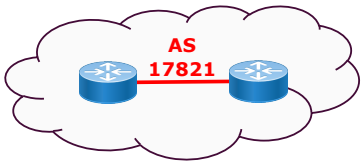
- eBGP neighbors/peers
  - BGP session established between routers in different ASes
  - Generally directly connected!
    - Session established using directly connected intf IP
    - Peering address must match the TCP session!
  - Else, we need a static route to reach the neighbor and change the eBGP TTL value (default 1)




```
router bgp 17821
 neighbor 172.16.12.2 remote-as 65000
!
 address-family ipv4
  neighbor 172.16.12.2 activate
 !
```

BGP Neighbor Relationship

- iBGP neighbors/peers
  - BGP session established between routers within the same AS
  - Does not need to be directly connected
    - IGP ensure reachability (TCP connection)
  - Generally using loopback addresses




```
router bgp 17821
 neighbor 10.10.10.2 remote-as 17821
!
```

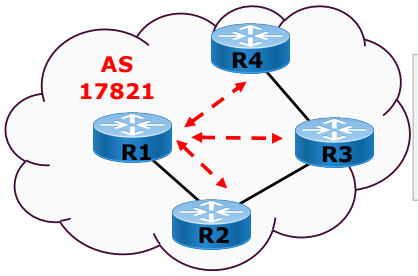

63

iBGP Operation

- iBGP routers must:
  - Originate directly connected routes
  - Carry routes learned from outside the AS to all routers within the AS
    - Fully-meshed instead of redistributing!
    - Advertise routes learned from eBGP peers to all iBGP peers!
  - To prevent routing loops (in a fully-meshed network)
    - iBGP routers are not allowed to advertise iBGP learned routes to other iBGP peers!

64

iBGP full-mesh




```
router bgp 17821
 neighbor 10.10.10.2 remote-as 17821
 neighbor 10.10.10.3 remote-as 17821
 neighbor 10.10.10.4 remote-as 17821
 !
```


65

Sourcing iBGP from Loopback

- By default, routers use the exit-interface address as the source address for locally originated packets (updates)
  - If the BGP TCP session was established using any other interface (loopbacks) addresses, the source address for BGP updates must match!
- The update-source loopback command achieves this

```
router bgp 17821
 neighbor 10.10.10.1 remote-as 17821
 neighbor 10.10.10.1 update-source loopback 0
!
```


66

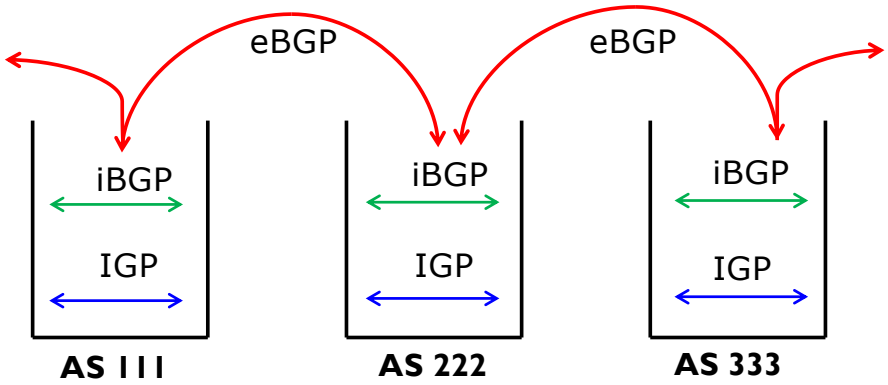
Advertising Networks in BGP

- The network statement
  - allows BGP to inject routes into BGP table and advertise to neighbors only if it already exists in the routing table!

```
router bgp 17821
 address-family ipv4 unicast
  network <prefix> mask <subnet-mask>
 address-family ipv6 unicast
  network <prefix/length>
```

- BGP "Synchronization Rule":
  - iBGP learned routes should not be installed in the routing table nor advertised to eBGP peers unless the route was learned through an IGP first!
    - Prevents black-hole routes!

How it all works?



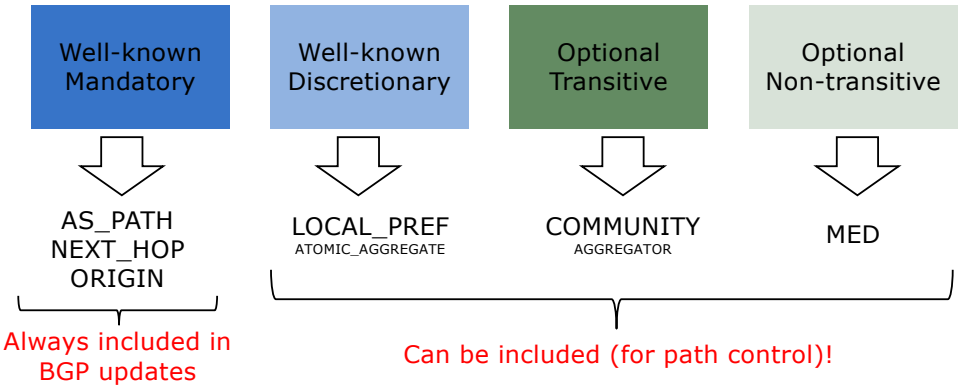
Barry Greene & Philip Smith "Cisco ISP Essentials"

69

70

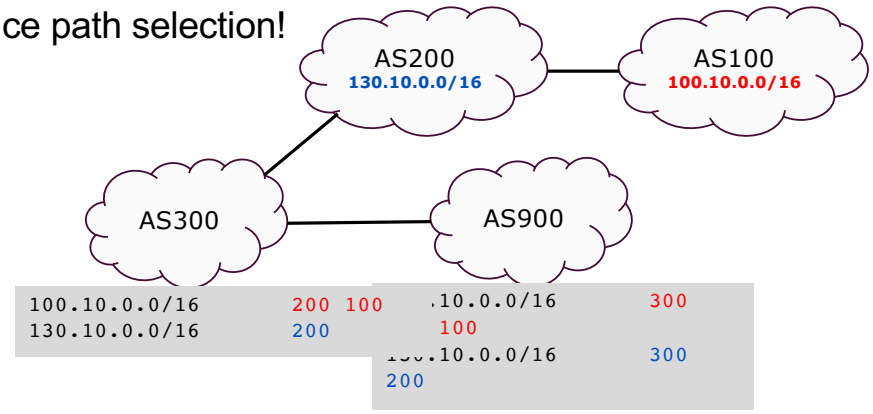
BGP Path Attributes

- Attributes describe the path to a network(s)/NLRI
  - Used to enforce routing policies for path control!



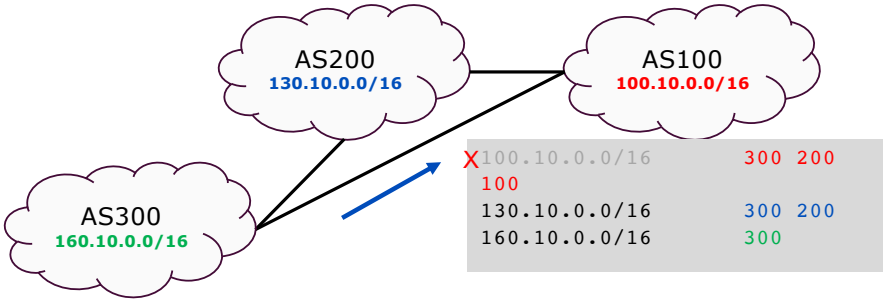
AS_PATH

- Indicates the list of ASes a route has passed through to reach the local AS
  - the list of ASes to reach a destination
  - can influence path selection!



AS_PATH

- Used to ensure a loop-free exchange of routing info between ASes
  - If own AS is seen in an update from an eBGP peer, loop is detected (Update is dropped)!

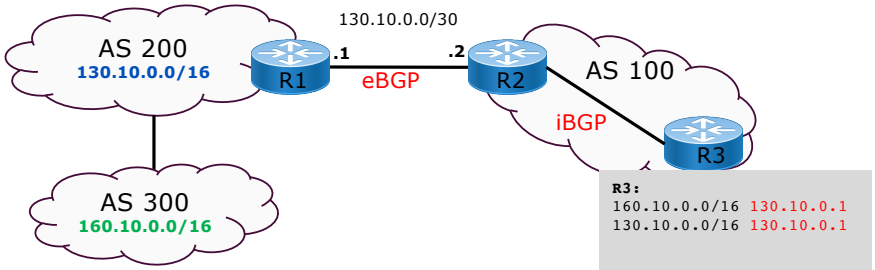


NEXT_HOP

- Indicates the next hop address to reach the destination
  - Source of the update packet!
- For eBGP
  - eBGP neighbor address (to reach the next AS)
- For iBGP
  - Generally the loopback address

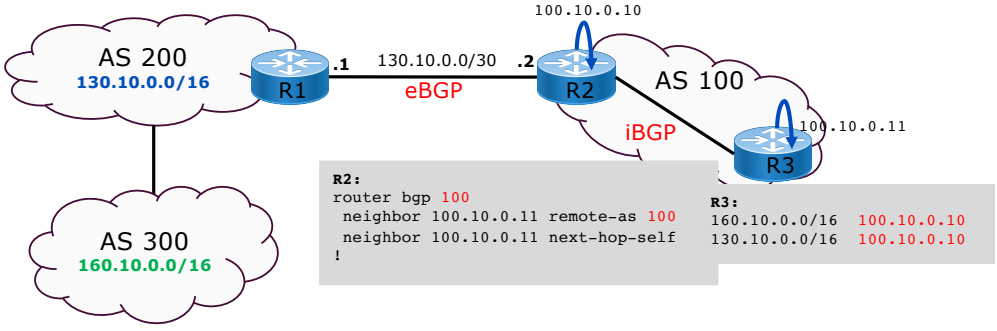
NEXT_HOP

- eBGP learned routes are advertised to iBGP peers without changing the next hop
  - Routers within the AS need to be able to reach the next hop (IGP or static)
  - Else, external routes not installed in the routing table!



NEXT-HOP-SELF

- Override the eBGP next hop default behavior with next-hop-self command
  - Advertises itself as the next hop for external routes
    - Reachable through IGP



ORIGIN

- Indicates the origin of the route
  - IGP (i)
    - Interior to the originating AS (advertised with the network command)
  - EGP (e)
    - Generated by EGP (obsolete!)
  - Incomplete (?)
    - Route's origin is unknown (usually redistributed)

77

ORIGIN

```
R1# show ip bgp
BGP table version is 24, local router ID is 172.16.1.2
Status codes: s suppressed, d damped, h history, * valid, > b
internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop          Metric  LocPrf  Weight  Path
*> 192.208.10.0     192.208.10.5           0              0  300 i
*> 172.16.1.0       0.0.0.0                0          32768  i
<output omitted>
```

i = Route generated by the **network** command.

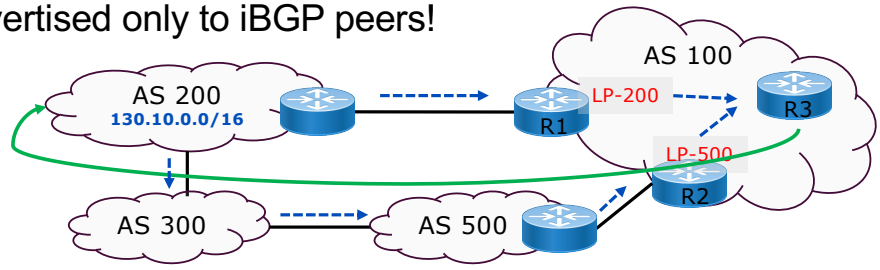
```
R1# show ip bgp
<output omitted>
   Network          Next Hop            Metric  LocPrf  Weight  Path
*> 10.1.1.0/24       0.0.0.0                0          32768  ?
*> 192.168.1.0/24    10.1.1.2              84          32768  ?
*> 192.168.2.0/24    10.1.1.2              74          32768  ?
<output omitted>
```

? = Route generated by unknown method (usually redistributed).

78

LOCAL_PREF

- Local preference tells routers within the AS (local) the preferred path to exit the AS
  - Path with highest local_pref wins
    - Outbound traffic!
- Local to the AS
  - Advertised only to iBGP peers!

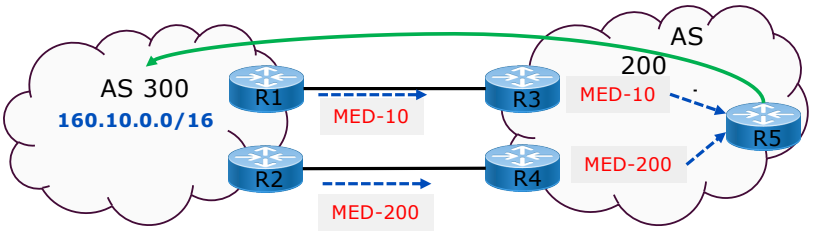


COMMUNITY

- Used to group prefixes (incoming/outgoing) and apply policies to the communities
  - A prefix can belong to more than one community
- Is (was?) a 32-bit integer
  - Represented as two 16-bit integers [ASN:number]
    - Works well for 2-byte ASN
- With 4-byte ASNs
  - Common to see [private-ASN:number]
  - RFC 8092 (BGP Large Communities): 96-bit integer
    - [32-bit ASN:32-bit:32-bit]

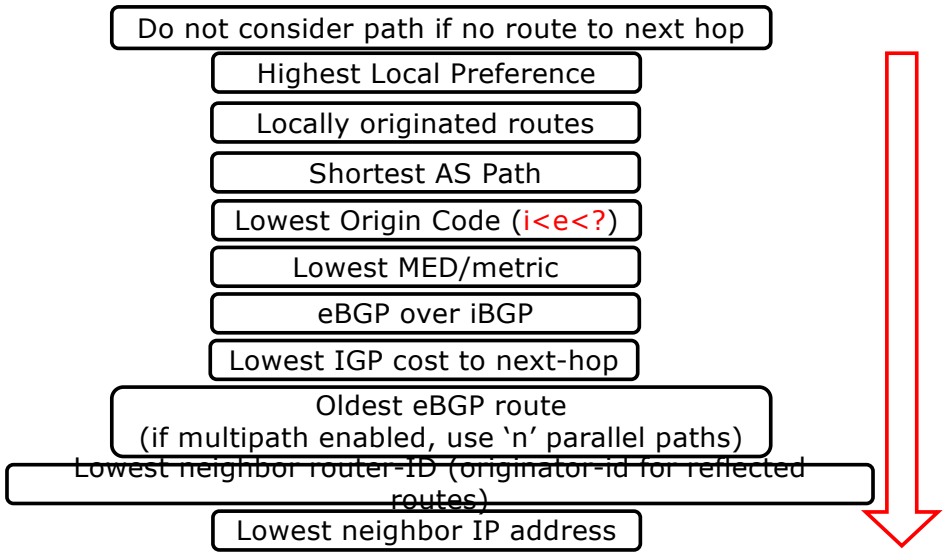
MED

- Multi-exit discriminator is inter-AS non-transitive
  - Indicates to neighbor AS about the preferred entry points into the local AS (incoming traffic)
- The path with lowest MED wins!



81

BGP Best Path Selection



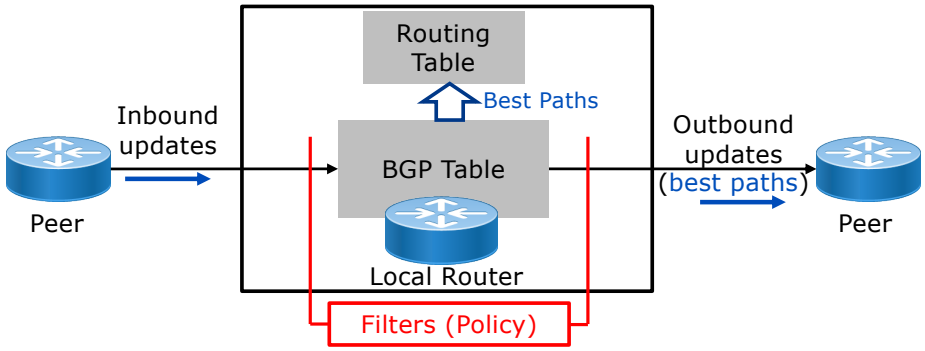
82

BGP Operation

- BGP learns routes from iBGP and eBGP peers
  - Selects best path based on the attributes
  - Installs best path in the routing table
  - Advertises the best paths to its other BGP peers
    - eBGP learned routes to iBGP peers
    - iBGP learned routes to eBGP peers

83

BGP Operation



84

BGP Tables

- Neighbor Table
  - List of all BGP neighbors
- BGP Table
  - List of routes learned from all BGP neighbors
  - (And locally originated routes!)
- Routing (Forwarding) Table
  - All best paths
    - selected based on attributes and whose next-hops are reachable!

85

86

Module 5:

BGP SCALING

87

BGP Peer Group

- Problem: number of BGP updates in a iBGP mesh
  - BGP updates generated for each neighbor individually
    - CPU wasted on repeat calculations
  - iBGP neighbors receive the same update
    - Contain same info
- Solution: Peer Groups
  - Group neighbors with the same outbound update policy
  - Updates are generated once per group

88

BGP Peer Group

- Still need to establish TCP sessions individually
- Useful when many neighbors have the same outbound policies
  - Runs through the outbound filters only once for the group (applied to all members)
- Members can have a different inbound policy!
- Simplifies configuration
  - Define the peer group
  - Add neighbors to the peer group
    - Still need to configure peering individually
  - Apply filters (outbound) to the group

89

Peer Group – Best Practices

- Always configure peer-groups for iBGP
  - Even if there are only a few iBGP peers
  - Easier to scale network in the future
- Consider using peer-groups for eBGP
  - Especially useful for multiple BGP customers using same AS (RFC 2270)
  - Also at IXPs where ISP policy is generally the same for each IX peer

90

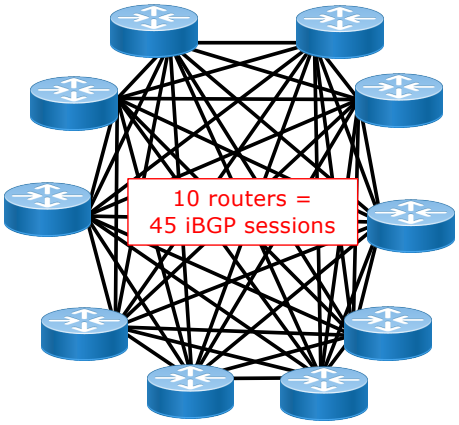
BGP Loop Prevention

- **eBGP**
  - AS-PATH attribute
  - If the local ASN is seen in a route received from a eBGP peer, a routing loop has occurred
    - Drop the route!
- **iBGP**
  - BGP router is not allowed to advertise iBGP learned routes to other iBGP peers within the AS
  - How do all iBGP routers learn about each other's networks?
    - iBGP full-mesh!

91

Scaling iBGP mesh

- Number of iBGP sessions
  - n(n-1)/2
    - 10 routers = 45 sessions
    - 100 routers = 4950 sessions
- Number of BGP updates
  - Every update needs to be sent to all iBGP peer



92

Solution

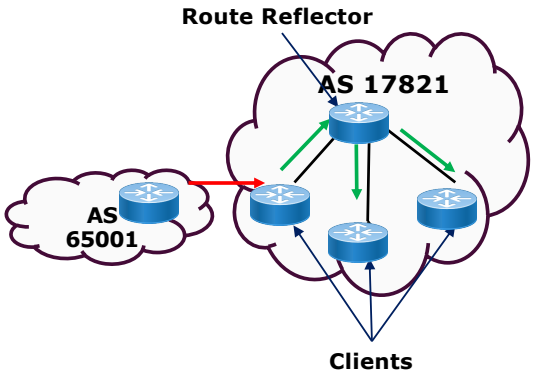
- Route Reflection (RFC4456)
  - RR client peers only with the RR
  - RR and its clients form a CLUSTER
  - Non-clients
  - An AS can have multiple clusters

RRs do not affect the actual traffic path;
only affects the path BGP messages take!

93

RR Operation

- When a RR receives an Update:
  - If from a client peer, the route is reflected (advertised) to other clients, and non-client peers.
  - If from a non-client, only reflected to client peers
  - If from a eBGP peer, reflected to both client and non-client peers.



Routing loops can happen with RRs!

94

Avoiding loops in RR

- Originator_ID attribute
  - The BGP router id of the originator, created by the RR
    - If you see your router id in the Originator_ID attribute, loop has occurred.
- Cluster_List attribute
  - In a cluster with a single RR, the Cluster ID is the router ID of the RR
  - If more than one RR in a cluster, a 4-byte Cluster ID configured
  - Cluster_List reflects the sequence of clusters a route has passed through
  - When a RR reflects routes (from clients) to non-clients, it appends the local Cluster ID to the Cluster_List
    - When a RR receives an update, if the local Cluster ID is seen in the list, a loop has occurred! (drops the update)

95

RR Design

- Divide the AS into multiple clusters
  - At least one RR and few clients in a cluster
  - Could have more than one RR in a cluster for redundancy
    - NOT recommended!
- Peering between clients in a cluster not necessary (but could be)
  - The RR reflects routes between them
- RRs in different clusters must be fully meshed with each other and with any iBGP router that's not a part of any cluster

96

RR example

- RR configuration

```
router bgp 17821
 <output-omitted>
 neighbor 2406:6400:2 remote-as 17821
 neighbor 2406:6400:2 route-reflector-client
 neighbor 2406:6400:3 remote-as 17821
 neighbor 2406:6400:3 route-reflector-client
 neighbor 2406:6400:4 remote-as 17821
 neighbor 2406:6400:4 route-reflector-client
<output-omitted>
```

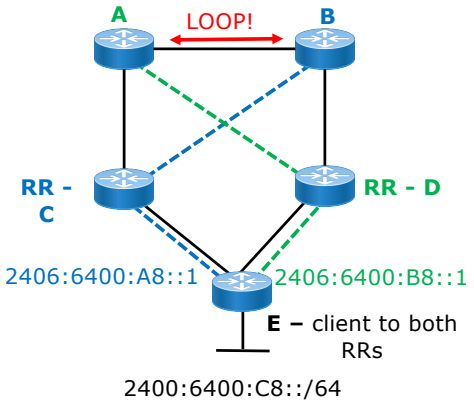
- RR Client config

```
router bgp 17821
 <output-omitted>
 neighbor 2406:6400:1 remote-as 17821
 <output-omitted>
```

Only peers with the RR

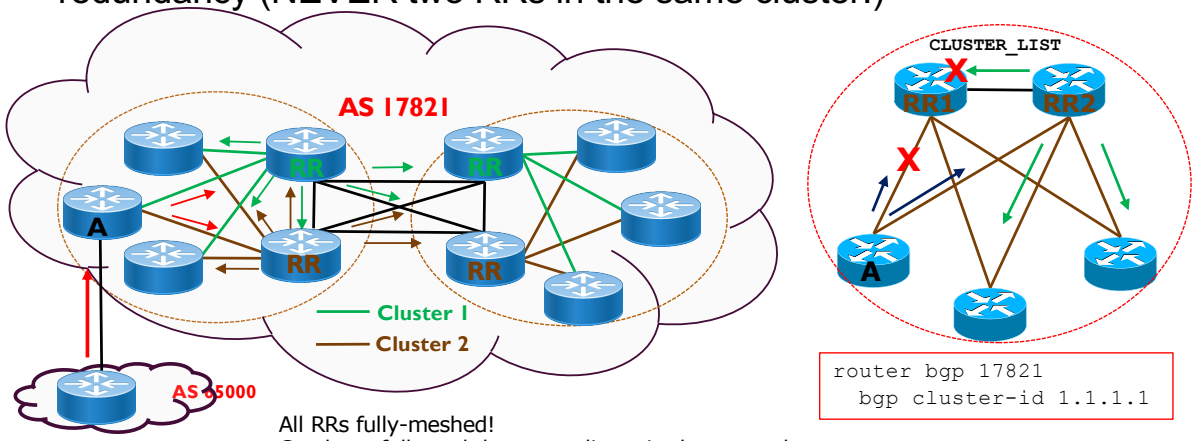
RR Selection

- Best practice is to follow the physical topology
  - Ensures traffic forwarding paths wont be affected
  - Prevents routing loops



RR Redundancy

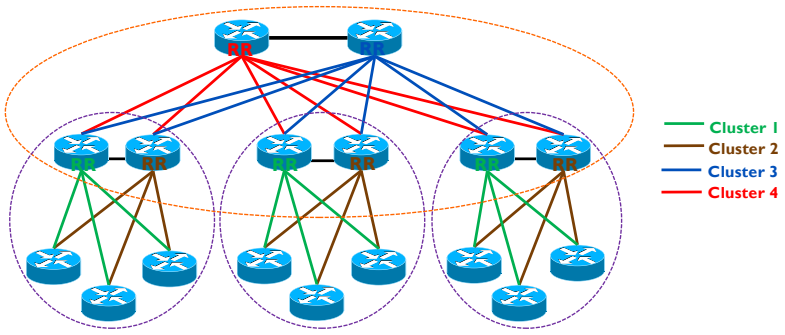
- Most ISP networks would overlay two clusters
  - Each client peers with RRs in different clusters (same POP) for redundancy (NEVER two RRs in the same cluster!)



All RRs fully-meshed!
Can have full-mesh between clients in the same cluster

RR Redundancy – Best Practice

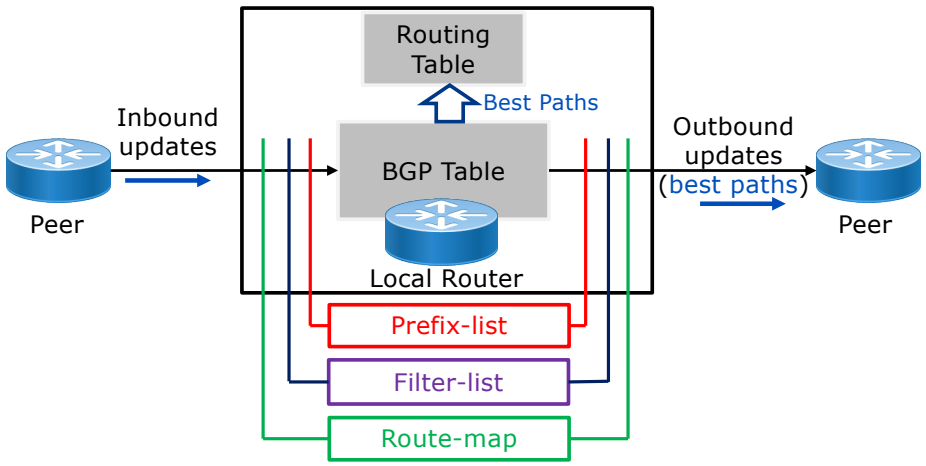
- A hierarchical RR design
  - RRs of some clusters are clients of other RRs in different clusters
  - Less iBGP sessions
  - Easier to manage and scale



101

102

Influence Path Selection – Policy Control



103

Policy Tools

- Prefix-list
  - To filter routes/prefixes
    - More granularity than as-path filters
- Filter-list
  - To filter based on AS-path
  - To apply AS-path ACLs
- Route-map
  - modify attributes based on condition matches

104

Path control - Attributes

- Inbound Traffic:
  - AS-Path, MED, Community
- Outbound Traffic:
  - Local Preference

105

Prefix List

```
ip prefix-list name/num [seq#] permit | deny
prefix/length [ge value][le value]
```

- Ex 1:
  - Allows any prefix with prefix length between 8 and 24
  - Implicit DENY at the end!

```
ip prefix-list TEST permit 0.0.0.0/0 ge 8 le 24
```

- Ex 2:
  - Permit the prefix 2400:6400::/32 up to /48
  - Implicit DENY at the end!

```
ipv6 prefix-list TEST-v6 permit 2001:6400::/32 le 48
```


106

Prefix List

- Ex 3:
  - Deny default route

```
ip prefix-list TEST deny 0.0.0.0/0
```

- Ex 4:
  - Deny IPv6 default routes

```
ipv6 prefix-list TEST-v6 deny ::/0
```


107

Prefix List

```
router bgp 17821
 network 100.100.0.0 mask 255.255.224.0
 neighbor 20.20.20.1 remote-as 20
 neighbor 20.20.20.1 prefix-list MY-PREFIX out
 neighbor 20.20.20.1 prefix-list PEER-PREFIX in
!
ip prefix-list MY-PREFIX permit 100.100.0.0/19
ip prefix-list MY-PREFIX deny 0.0.0.0/0 le 32
!
ip prefix-list PEER-PREFIX permit 200.200.0.0/16
ip prefix-list PEER-PREFIX deny 0.0.0.0/0 le 32
```


108

AS-path ACL

- AS-path access list use regular expressions

```
ip as-path access-list num [permit|deny] regex
```

.	Matches any one character
*	Matches any sequence of pattern before *
+	match at least one preceding expression
^	beginning with
$	ending with
_	matches start, end, space, comma, braces

AS-path ACL

- Example regular expressions:

^$	locally originated routes
_100$	originated by AS 100
_100_200_	passing through 100 and 200
^(_100)+$	originated by 100, multiple occurrence

- Example 1:

```
ip as-path access-list 10 permit ^100$
```

  - Allow any prefix originated and received from AS100
  - Implicit DENY at the end

- Use **filter-list** to apply AS-PATH access-lists

AS-path ACL

```
router bgp 17821
 network 100.100.0.0 mask 255.255.224.0
 neighbor 30.30.30.1 remote-as 30
 neighbor 30.30.30.1 filter-list 30 out
 neighbor 30.30.30.1 filter-list 40 in
!
ip as-path access-list 30 permit ^$
ip as-path access-list 40 permit ^30$
```

Route Map

- Default is permit
  - Implicit DENY at the end!

```
route-map name [permit | deny] [sequence]
```

```
route-map TEST permit 10
 match A B C
 match D
 set X
 set Y
```

If {(A or B or C) and D} match
Then {set X and Y}

```
route-map TEST permit 20
 match E
 set Z
```

Else
If E matches
Then set Z

```
route-map TEST permit 30
```

Else (for everything else)
Do/set nothing

Match (conditions) & Set (actions)

Command	Description
match community	BGP community tag
match as-path	AS-path access list
match ip address	Access list or prefix-list

Command	Description
set as-path <prepend>	Modify AS-path
set community	Apply BGP community tag
set metric	Modify MED
set local-preference	Modify local preference

Route Map

```
router bgp 17821
 neighbor 30.30.30.1 remote-as 30
 neighbor 30.30.30.1 route-map AS-OUT out
 neighbor 30.30.30.1 route-map LP-IN in
!
route-map AS-OUT permit 10
 set as-path prepend 17821 17821 17821
!
route-map LP-IN permit 10
 match as-path 1
 set local-preference 150
!
route-map LP-IN permit 20
!
ip as-path access-list 1 permit _30$
```

Route Map

- Setting and Matching communities:

```
router bgp 17821
 network 100.100.0.0 mask 255.255.224.0 route-map SET-AGG
 neighbor 20.20.20.1 remote-as 20
 neighbor 20.20.20.1 send-community
 neighbor 20.20.20.1 route-map TR-IN in
!
route-map SET-AGG permit 10
 set community 100:1000
!
route-map TR-IN permit 10
 match community 5
 set local-preference 150
!
route-map TR-IN permit 20
!
ip community-list 5 permit 20:3000
ip community-list 5 permit 20:4000
```


Applying Policy Filters

- Incoming/Outgoing updates are filtered through policies
  - BGP table does not contain routes rejected by policies
- Whenever there is a BGP policy change, we need to
  - Trigger an update to force in/outbound routes through the new filters (else only the ones already in BGP table)
  - either through a Hard Reset or a Soft Reset (Route Refresh)
- If the filter is applied to:
  - Outbound routes: need to resend its BGP table through the filter
  - Inbound routes: need its neighbors to resend their BGP tables

Hard Reset

- Hard reset of a BGP session
  - Tears down the TCP connection
  - Re-establish the TCP session
  - Resend the BGP table to neighbors affected by the reset
  - Relearn all routes from neighbors

```
clear ip bgp *
clear ip bgp <peer-address>
clear bgp ipv6 unicast *
clear bgp ipv6 unicast <peer-address>
```

  - Disrupts network connectivity
    - Same as a router reboot!

Route Refresh

- Route refresh (soft reset) does not tear down BGP session
  - Not disruptive!
- No manual configuration required
  - Negotiated automatically
  - Peering routers need to support route refresh

```
sh ip bgp neighbor <peer-address>
sh ip bgp all neighbors
sh bgp ipv6 unicast neihgbors
```

```
Neighbor capabilities:
    Route refresh: advertised and received
```

Route Refresh

```
clear ip bgp <peer-address> [soft] in
clear bgp ipv6 unicast <peer-address> [soft] in
```

– Tells neighbor to resend its BGP table

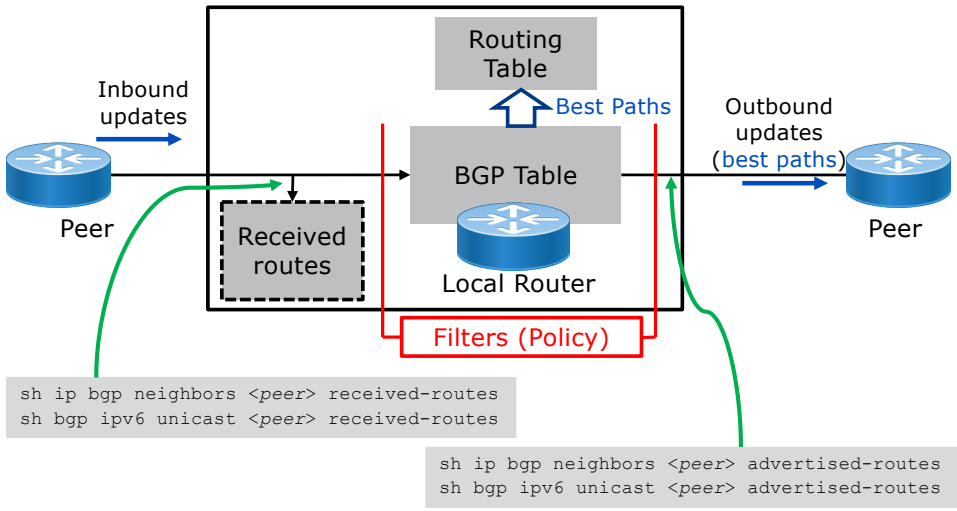
```
clear ip bgp <peer-address> [soft] out
clear bgp ipv6 unicast <peer-address> [soft] out
```

– Resends full BGP table to its neighbor

Soft Reconfiguration

- All current routers should support route refresh
  – If the router (local or peer) does not have route refresh capability, use soft-reconfiguration
- With soft-reconfiguration, the router stores a copy of the received routes in addition to the BGP table (allowed by policy filters)
  – Thus, requires additional memory!

Soft Reconfiguration



121

Soft Reconfiguration

– Whenever there is change in policy

```
router bgp 17821
 neighbor 1.1.1.1 remote-as 100
 neighbor 1.1.1.1 route-map SET-LPREF in
 neighbor 1.1.1.1 soft-reconfiguration
inbound
```

- If “in”, runs the stored received routes through the new filter
- If ”out”, sends the BGP table through the filters
- Does not tear down the BGP session!

– Route refresh capability cannot be used if soft-reconfig is used!

```
clear ip bgp 1.1.1.1 soft [in|out]
```


122

Questions
APNIC
123

123

Module 7:
BGP OPERATIONS AND SECURITY PRACTICES
APNIC
124

124

Protecting BGP sessions

- MD5 authentication (RFC2385)
  - To protect the BGP TCP session between peers
    - generates a keyed hash (16-byte) - using the TCP segment and the password

```
router bgp 17821
 neighbor 30.30.30.1 password <key-value>
```

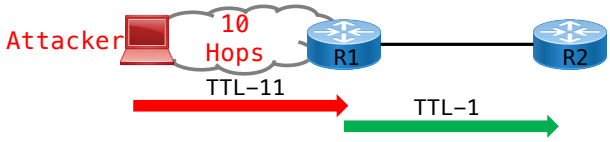
    - But susceptible to collision attacks

- TCP Authentication Option (RFC5925)
  - Obsoletes RFC2385, but no known implementations… please correct ??
    - Key chains (allows moving from one key to another for the same connection), and
    - Support for stronger hash functions

125

Protecting BGP sessions

- GTSM – BGP TTL Security (RFC5082)
  - eBGP has a default TTL of 1
    - Which requires eBGP peers to be directly connected
    - To use any interface other than the directly connected, we generally use ebgp-mulithop
  - A remote attacker could send spoofed packets with adjusted TTL values to make it seem like its directly connected



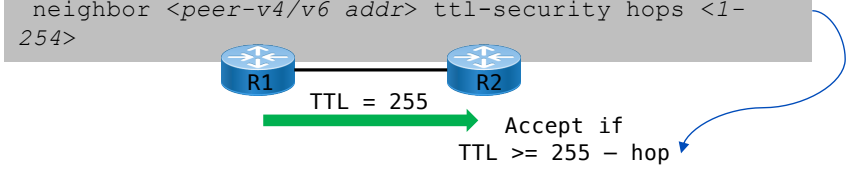
    - This way, R2 could be DoSed (will accept the initial TCP SYN) or TCP resets

126

Protecting BGP sessions

- With GTSM, the TTL between directly connected eBGP peers set to the maximum – 255
  - Only incoming packets with a TTL value equal to or greater than the locally configured value is accepted,
  - Else packet is silently discarded and no ICMP messages generated

```
router bgp 17821
 neighbor <peer-v4/v6 addr> ttl-security hops <1-254>
```

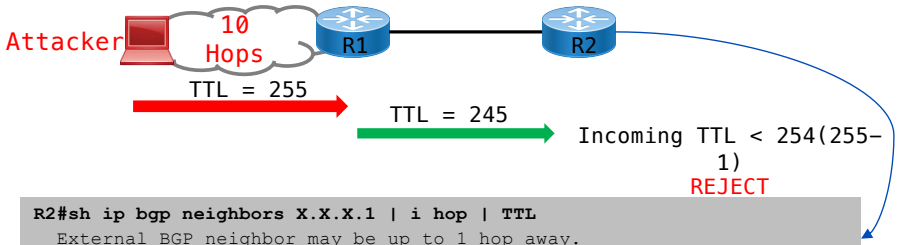


- Needs to be same on both peers!

Protecting BGP sessions

- GTSM – BGP TTL Security (RFC5082)
  - Ex: If R2's config was

```
router bgp 17821
 neighbor X.X.X.1 ttl-security hops 1
```




```
R2#sh ip bgp neighbors X.X.X.1 | i hop | TTL
  External BGP neighbor may be up to 1 hop away.
Connection is ECN Disabled, Mininum incoming TTL 254, Outgoing TTL 255
```

Prefix Filters - Outbound

- To Customers speaking BGP
  - Probably only a default route, or
  - WHOLE Internet feed except default and bogons (special use addresses – rfc6890, rfc1918, unassigned blocks)
- To Peers (other ISPs with whom you peer)
  - Send your prefixes + your downstream customer, or
  - what you agreed to send
- To Upstream/Transit provider
  - Send your prefixes + your downstream customers

Prefix Filters - Inbound

- Customers speaking BGP
  - Only accept their prefixes
    - Verify that the prefixes were Assigned or Allocated, and
    - Make sure the prefix length does not exceed /24 (IPv4) and /48 (IPv6)

```
whois -h jwhois.apnic.net 61.45.248.0/21

inetnum:        61.45.248.0 - 61.45.255.255
netname:        APNICTRAINING-AP
descr:          APNIC TRAINING UNIT
descr:          6 Cordelia St. South Brisbane, QLD
country:        AU
org:            ORG-ATU1-AP
admin-c:        AINT1-AP
tech-c:         AINT1-AP
status:         ALLOCATED PORTABLE
mnt-by:         APNIC-HM
mnt-irt:        IRT-ABCINTERNET-SG
last-modified:  2017-08-29T23:00:41Z
source:         APNIC
```

Means it was delegated to an entity

Means it was allocated to the customer, and they can announce the prefix

Prefix Filters - Inbound

- Customers speaking BGP
  - Ex: for a customer with the block 61.45.248.0/21 block

```
router bgp 17821
 address-family ipv4
  neighbor X.X.X.1 prefix-list CUST-PREFIX in
!
ip prefix-list CUST-PREFIX permit 61.45.248.0/21 le 24
ip prefix-list CUST-PREFIX deny 0.0.0.0/0 le 32
!
```


131

Prefix Filters - Inbound

- From Peers
  - Other ISPs/operators with whom you have agreed to exchange routes
  - Only accept their prefixes and their downstream customers (or what was agreed)
    - Verify they have the authority to route those prefixes (and their customers)
    - Prefix length should not exceed /24 (IPv4) and /48 (IPv6)
    - Can use RPSL tools like bgpq3 - https://github.com/snar/bgpq3

```
bgpq3 -6Al PEERv6-IN AS17660
no ipv6 prefix-list PEERv6-IN
ipv6 prefix-list PEERv6-IN permit 2405:d000::/32
ipv6 prefix-list PEERv6-IN permit 2405:d000:7000::/36
```


132

Prefix Filters - Inbound

- From Peers
  - Ex: if a peer has 2001:dc0::/32 and 203.119.96.0/20 prefixes

```
router bgp 17821
 address-family ipv4
  neighbor X.X.X.1 prefix-list PEERv4-IN in
 address-family ipv6
  neighbor X:X:X::1 prefix-list PEERv6-IN in
!
ip prefix-list PEERv4-IN permit 203.119.96.0/20 le 24
ip prefix-list PEERv4-IN deny 0.0.0.0/0 le 32
!
ipv6 prefix-list PEERv6-IN permit 2001:dc0::/32 le 48
ipv6 prefix-list PEERv6-IN deny ::/0 le 128
!
```


133

Prefix Filters - Inbound

- From Upstream (Transit Provider)
  - Could just be a default route

```
router bgp 17821
 address-family ipv4
  neighbor X.X.X.1 prefix-list DEF-IN in
 address-family ipv6
  neighbor X:X:X::1 prefix-list DEFv6-IN in

!
ip prefix-list DEF-IN permit 0.0.0.0/0
!
ipv6 prefix-list DEFv6-IN permit ::/0
```


134

Prefix Filters - Inbound

- From Upstream (Transit Provider)
  - the WHOLE Internet feed
  - Do not accept your own prefixes
  - Do not accept bogons
    - special use addresses – rfc6890, rfc1918, and unassigned blocks
  - Do not accept prefix lengths longer than /24 (IPv4) and /48 (IPv6)

Prefix Filters - Inbound

- From Upstream (Transit Provider) – IPv4

```
router bgp 17821
 address-family ipv4
  neighbor X.X.X.1 prefix-list TRANSITv4-IN in
!
ip prefix-list TRANSITv4-IN deny 0.0.0.0/0                  ! Default
ip prefix-list TRANSITv4-IN deny 0.0.0.0/8 le 32            ! Network Zero
ip prefix-list TRANSITv4-IN deny 10.0.0.0/8 le 32           ! RFC1918
ip prefix-list TRANSITv4-IN deny 100.64.0.0/10 le 32        ! RFC6598 shared
address
ip prefix-list TRANSITv4-IN deny <your prefix>/X le 32      ! Your address space
ip prefix-list TRANSITv4-IN deny 127.0.0.0/8 le 32          ! Loopback
ip prefix-list TRANSITv4-IN deny 169.254.0.0/16 le 32       ! APIPA
ip prefix-list TRANSITv4-IN deny 172.16.0.0/12 le 32        ! RFC1918
ip prefix-list TRANSITv4-IN deny 192.0.0.0/24 le 32         ! IETF Protocol
ip prefix-list TRANSITv4-IN deny 192.0.2.0/24 le 32         ! TEST1
ip prefix-list TRANSITv4-IN deny 192.168.0.0/16 le 32       ! RFC1918
ip prefix-list TRANSITv4-IN deny 198.18.0.0/15 le 32        ! Benchmarking
ip prefix-list TRANSITv4-IN deny 198.51.100.0/24 le 32      ! TEST2
ip prefix-list TRANSITv4-IN deny 203.0.113.0/24 le 32       ! TEST3
ip prefix-list TRANSITv4-IN deny 224.0.0.0/4 le 32          ! Multicast
ip prefix-list TRANSITv4-IN deny 240.0.0.0/4 le 32          ! Future Use
ip prefix-list TRANSITv4-IN deny 0.0.0.0/0 ge 25            ! Prefixes longer than
/24
ip prefix-list TRANSITv4-IN permit 0.0.0.0/0 le 32
```

Prefix Filters - Inbound

- From Upstream (Transit Provider) – IPv6

```
router bgp 17821
 address-family ipv6
  neighbor X:X:X::1 prefix-list TRANSITv6-IN in
!
ipv6 prefix-list TRANSITv6-IN deny 2001::/32 le 128                  ! Teredo
subnets
ipv6 prefix-list TRANSITv6-IN deny 2001:db8::/32 le 128              !
Documentation
ipv6 prefix-list TRANSITv6-IN deny 2002::/16 le 128                  ! 6to4
subnets
ipv6 prefix-list TRANSITv6-IN deny <your::/32> le 128                ! Your
prefix
ipv6 prefix-list TRANSITv6-IN deny 3ffe::/16 le 128                  ! Old 6bone
ipv6 prefix-list TRANSITv6-IN deny fc00::/7 le 128                   ! ULA
ipv6 prefix-list TRANSITv6-IN deny fe00::/9 le 128                   ! Reserved
IETF
ipv6 prefix-list TRANSITv6-IN deny fe80::/10 le 128                  ! Link-local
ipv6 prefix-list TRANSITv6-IN deny fec0::/10 le 128                  ! Reserved
IETF
ipv6 prefix-list TRANSITv6-IN deny ff00::/8 le 128                   ! Multicast
ipv6 prefix-list TRANSITv6-IN permit 2000::/3 le 48                  ! Global
Unicast
ipv6 prefix-list TRANSITv6-IN deny ::/0 le 128
```


137

Aside - Bogons

- Not all IP (v4 and v6) are allocated by IANA
- Addresses that should not be seen on the Internet are called “Bogons” (also called “Martians”)
  - RFC6890+RFC1918s + Reserved space
- IANA publishes list of number resources that have been allocated/assigned to RIRs/end-users
  - https://www.iana.org/assignments/ipv6-unicast-address-assignments/ipv6-unicast-address-assignments.xhtml
  - https://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml

138

Bogons

- Commonly found as source addresses of DDoS packets
- We should have ingress and egress filters for bogon routes
  - Should not route them nor accept them from peers
- We could manually craft prefix filters based on the bogon list from IANA
  - But bogon list is dynamic
  - New allocations made out of reserved blocks frequently

Bogon Route Server Project

- In comes the Bogon Route Server project by Team Cymru
  - Provides dynamic bogons information using eBGP multihop sessions
  - Traditional bogons (AS65333)
    - martians plus prefixes not allocated by IANA
  - Full-bogons (AS65332)
    - above plus prefixes allocated to RIRs but not yet assigned to ISPs/end-users by RIRs
- For details:
  - http://www.team-cymru.org/bogon-reference-bgp.html

Peering- Bogon Route Servers

- To peer with bogon route servers
  - Write to bogonrs@cymru.com
- You should provide:
  - Your ASN
  - Which bogons you wish to receive
  - Your peering addresses
  - MD5 for BGP?
  - PGP public key (optional)
- It is recommended to have at least 2 (two) peering sessions for redundancy

Maximum Prefix Limit

- It is RECOMMENDED to set the maximum number of prefixes accepted from a peer
  - Prevent memory exhaustion,
  - Prevent impacts of route leaks (where more specifics of big covering prefix gets leaked)

```
router bgp 17821
 address-family ip[v4|v6]
  neighbor <peer-addr|group> maximum-prefix <max-value> [threshold][restart N] [warning-only]
```

  - max-value: the max prefix limit
  - threshold: threshold in % of max limit (by default 75%) to generate warning msg
  - restart: restart BGP connection after N minutes
  - warning-only: generate a warning msg when limit is exceeded instead of terminating the BGP session

Maximum Prefix Limit

- Setting the max-prefix limit:
  - Allow future growth
    - Ex: If a peer has a /12 block, allowing upto /24s
    - The number of /24s = 4096 ~ max limit
  - IXPs generally publish the max IPv4/v6 prefixes announced by their route servers (RS)
    - Set your max limit accordingly
    - Ex: @HKIX - max prefix limit ~ [(monthly 2-hr avg)/0.7]
  - Examples:
    - Drop the peering if more than 1000 prefixes received

```
neighbor X.X.X.1 maximum-prefix 1000
```

    - Log a warning when it receives more than 1000 prefixes

```
neighbor X.X.X.1 maximum-prefix 1000 warning-only
```

    - Logs a warning at 900 prefixes, and drops if more than 1000 received

```
neighbor X.X.X.1 maximum-prefix 1000 90
```


143

AS-PATH Filtering

- Do NOT accept/announce prefixes with private ASNs
  - Unless you have customers using private ASNs, or
  - Special arrangements like Cymru Bogon RS

```
router bgp 17821
 address-family ip[v4|v6]
  neighbor <peer-addr|group> remove-private-as [all [replace-as]
```

- Enforce the first ASN in the AS_PATH attribute to be the peer's ASN
  - except if peering with a RS @IXPs

```
router bgp 17821
 bgp enforce-first-as !by default in IOS
```


144

AS-PATH Filtering

- Limit the AS-PATH length
  - Unusually long AS-PATHs would cause
    - Memory exhaustion, or
    - Issues like below which caused massive global routing updates (~107K per second) for an hour:
    - https://dyn.com/blog/the-flap-heard-around-the-world/
    - https://dyn.com/blog/longer-is-not-better/

```
bgp-prepend (integer:0-16)
>> bgp-prepend 47868 ~ 47868 mod 256 ~ 252
```

- Some weird announcements:

```
N*>  45.162.216.0/24   4608 24130 7545 6939 263311 268528 268528 268528 268528 268528 268528
268528 268528 268528 268528 268528 268528 268528 268528 268528 268528 i

N*>  45.162.217.0/24   4608 24130 7545 6939 263311 268528 268528 268528 268528 268528 268528
268528 268528 268528 268528 268528 268528 268528 268528 268528 268528
```

```
0x7F48C876B370  4608 24130 7545 6939 28186 264437 264437 264437 264437 264437 264437 264437
264437 264437 264437 264437 264437 264437 264437 264437 264437 264437 264952 i
```


AS-PATH Filtering

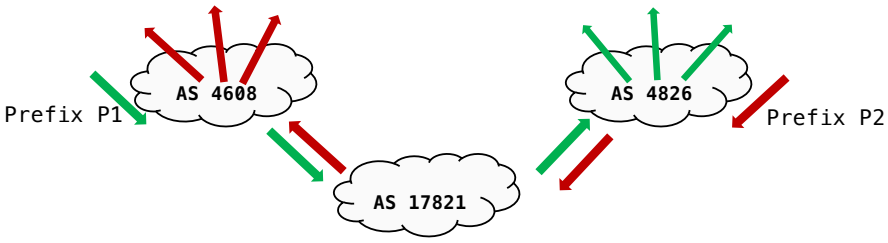
- Limit the AS-PATH length
  - The internet would be around 5-10 ASes deep on average
  - Longest AS-PATHs could be ~ 30 ASNs
  - Consider limiting the AS-PATH length for prefixes you accept

```
router bgp 17821
 bgp maxas-limit <1-254>
```

RFC 8212 – BGP default reject

- On many platforms, BGP is implicitly permissive!
  - AS-to-AS leaks common during maintenance

```
router bgp 17821
 !
 neighbor X.X.X.1 remote-as 4608
 neighbor X.X.X.1 description eBGP with Upstream-1
 !
 neighbor A.A.A.1 remote-as 4826
 neighbor A.A.A.1 description eBGP with Upstream-2
```



RFC 8212 – BGP default reject

- RFC8212- eBGP route propagation without policies
  - Neither accept nor announce routes from/to eBGP peers without explicit policy (import/export) configurations
    - implicit deny-all associated with eBGP sessions!
  - Changes to RFC4271:
    - Decision Process: Routes contained in an Adj-RIB-In associated with an EBGP peer SHALL NOT be considered eligible in the Decision Process if no explicit Import Policy has been applied.
    - Route Dissemination: Routes SHALL NOT be added to an Adj-RIB-Out associated with an EBGP peer if no explicit Export Policy has been applied.

RFC 8212 – BGP default reject

- But very few known implementations:
  - IOS-XR (all versions)
  - BIRD (2.0.1 and higher)
  - OpenBGPD (6.4 and higher)
  - Nokia SR OS (19.5.R1 and above)
    - https://github.com/bgp/RFC8212

- For those OSes that don't support RFC8212 yet
  - Shut the BGP session with the peer (group) during configuration
  - Define and apply explicit export and import policies to the eBGP peer(s)
  - Then no shut the BGP session
  - Talk to your vendors and force them to support RFC8212!

149

Traffic Filters

- BCP38 (RFC2827)
  - Since 1998!
  - https://tools.ietf.org/html/bcp38

- Only allow traffic with valid source addresses to
  - Leave your network
    - Only packets with source address from your own address space
  - To enter/transit your network
    - Only source addresses from downstream customer address space

150

uRPF – Unicast Reverse Path

- Unicast Reverse Path Forwarding (uRPF)
  - Router verifies if the source address of incoming packets is in the Forwarding table and also checks in the incoming interface
    - Drop if not!
  - Recommended on customer facing interfaces

```
(config-if)#ipv6 verify unicast source reachable-via {rx | any}
```


151

uRPF – Unicast Reverse Path

- Modes of Operation (IOS):
  - Strict: verifies both source address and incoming interface with entries in the forwarding table
  - Loose: verifies existence of route to source address

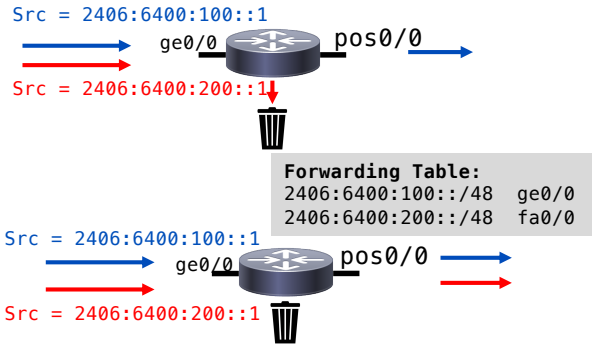


Image source: "Cisco ISP Essentials", Barry Greene & Philip Smith 2002

152

MANRS

- Mutually Agreed Norms of Routing Security
  - An ISOC led initiative to implement industry best practices to ensure security of routing system
  - https://www.manrs.org/
    - Inbound/outbound filtering – prefix/as-path
    - Source address validation – BCP38
    - Coordination – correct & up to date contacts
    - Validation – ROAs/IRR objects

153

154

Module 8

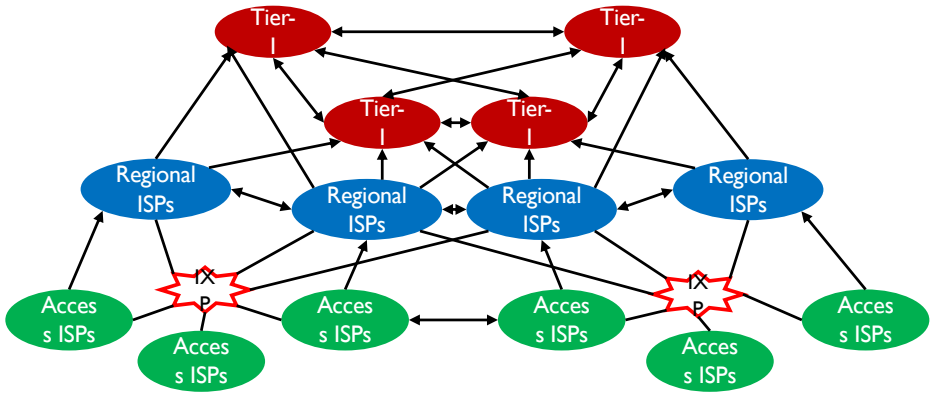
MULTIHOMING

APNIC

155

155

ISP Hierarchy



Philip Smith "Introduction to Internet"

APNIC

156

156

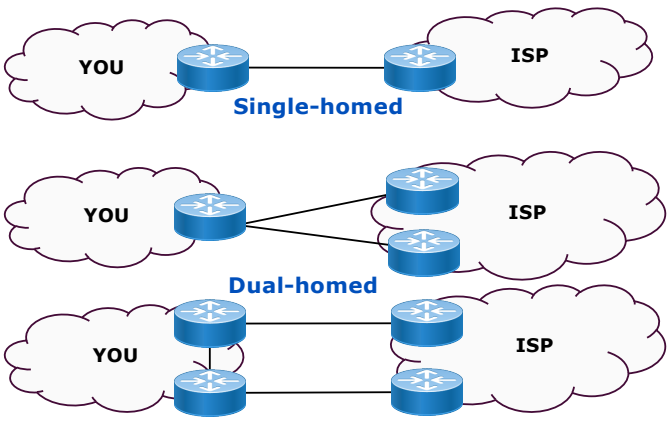
Exchanging Routes

- Pay someone to advertise your networks
  - TRANSIT
  - Make sure they have good onward peering/transit!
- Interconnect with as other ASes to exchange locally originated routes and traffic
  - PEERING
  - Private Peering
    - Between two ASes
  - Public Peering
    - at an IXP (domestic/global)

157

Achieving Redundancy

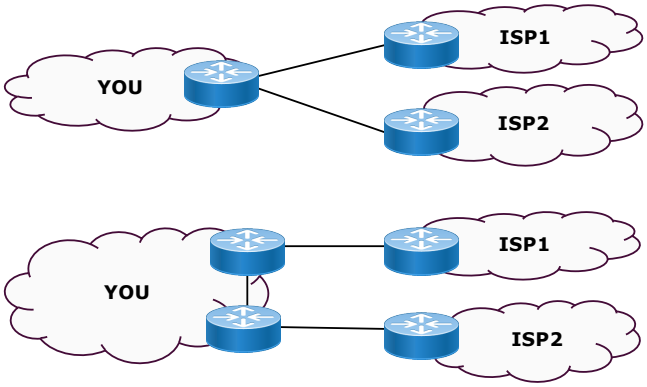
- More than one path to the same ISP
  - Dual-homed



158

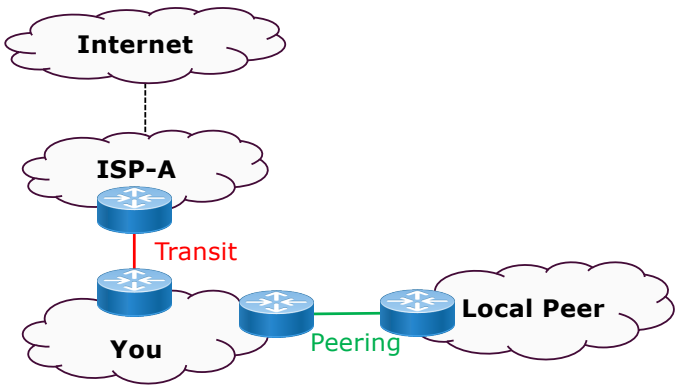
Achieving Redundancy – Multihoming

- More than one upstream ISP
  - Multi-homed



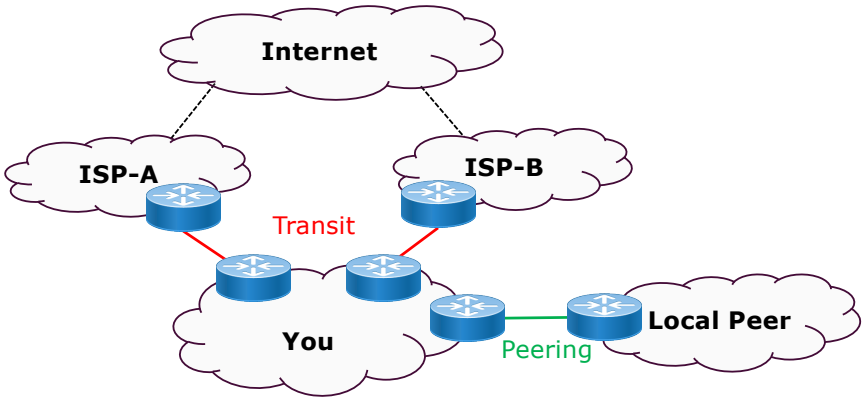
Multihoming

- One upstream and local peering



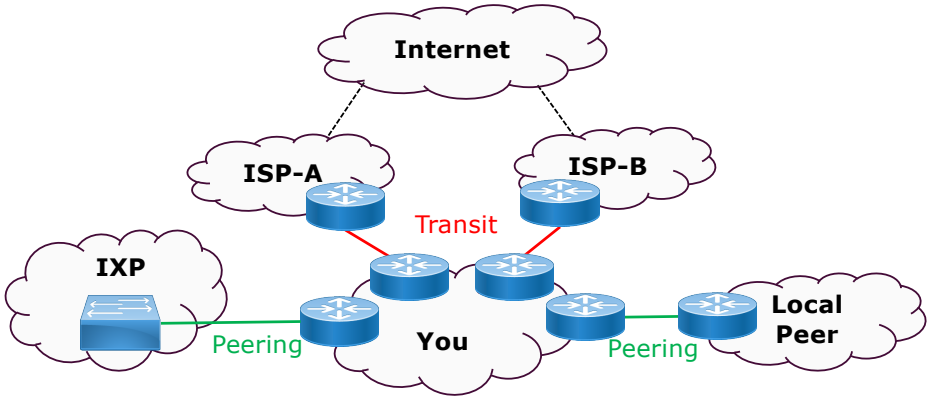
Multihoming

- More than one upstream ISP and local peering



Multihoming

- More than one upstream ISP with local and public peering

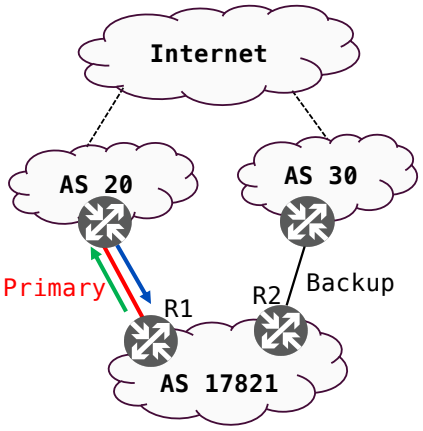


Recap: Path control Attributes

- Inbound Traffic:
  - AS-PATH, MED, Community (sub-aggregates/more specifics)
- Outbound Traffic:
  - Local Preference

Two Upstream – One backup

- We want:
  - Both incoming and outgoing traffic via AS20 (R1)
  - AS30 (R2) path to be used only if the link to AS20 fails
- AS-PATH to control incoming traffic
  - Prepend outbound on R2
- LOCAL-PREF for outgoing traffic
  - Higher LP for inbound routes on R1



Two Upstream – One backup

- Always announce the aggregate on both links!
- R1 (main link) config:

```
router bgp 17821
 network 61.45.248.0 mask 255.255.248.0
 neighbor 20.20.20.1 remote-as 20
 neighbor 20.20.20.1 prefix-list AGGR out
 neighbor 20.20.20.1 prefix-list DEF in
!
ip prefix-list AGGR permit 61.45.248.0/21
ip prefix-list DEF permit 0.0.0.0/0
!
ip route 61.45.248.0 255.255.248.0 null0
```

Advertise aggregate

Prefix-list applied to outbound routes

Prefix-list applied to inbound routes

Define the prefix-lists

Aggregate should exist in the routing table (pull-up route)

Two Upstream – One backup

- R2 (backup) config:

```
router bgp 17821
 network 61.45.248.0 mask 255.255.248.0
 neighbor 30.30.30.1 remote-as 30
 neighbor 30.30.30.1 prefix-list AGGR out
 neighbor 30.30.30.1 route-map BACKUP-OUT out
 neighbor 30.30.30.1 prefix-list DEF in
 neighbor 30.30.30.1 route-map BACKUP-IN in
!
ip prefix-list AGGR permit 61.45.248.0/21
ip prefix-list DEF permit 0.0.0.0/0
!
ip route 61.45.248.0 255.255.248.0 null0
!
route-map BACKUP-OUT permit 10
 set as-path prepend 17821 17821 17821
!
route-map BACKUP-IN permit 10
 set local-preference 80
```

Advertise aggregate in BGP

Route-map applied to outbound routes

Route-map applied to inbound routes

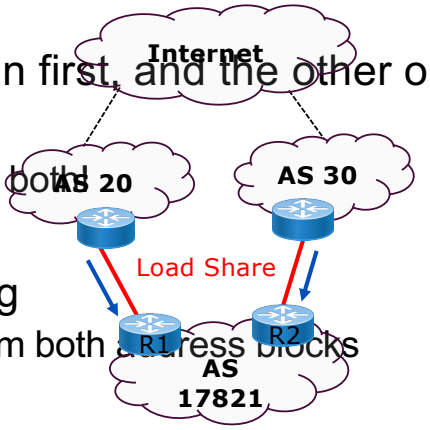
Define the prefix-lists

BACKUP-OUT prepends the AS-PATH for all outbound BGP updates

BACKUP-IN sets lower local pref for all inbound BGP updates

Two Upstream – Load Sharing (Inbound Traffic)

- Load share incoming and outgoing traffic on the two links
- Announce one sub-aggregate on first, and the other on the second link
  - Always announce the aggregate on both
- Requires good address planning
  - Customers need to be assigned from both address blocks



Two Upstream – Load Sharing (Inbound Traffic)

- R1 config:

```
router bgp 17821
network 100.100.0.0 mask 255.255.224.0
network 100.100.0.0 mask 255.255.240.0
neighbor 20.20.20.1 remote-as 20
neighbor 20.20.20.1 prefix-list SUB-A out

!
ip prefix-list SUB-A permit
100.100.0.0/19
ip prefix-list SUB-A permit
100.100.0.0/20

!
ip route 100.100.0.0 255.255.224.0 null0
ip route 100.100.0.0 255.255.240.0 null0
```

Advertise both aggregate and first sub-prefix in BGP

Advertise sub-aggregate along with the aggregate

Sub-aggregate should exist in the routing table (pull-up route)

Two Upstream – Load Sharing (Inbound Traffic)

- R2 config:

```
router bgp 17821
network 100.100.0.0 mask 255.255.224.0
network 100.100.16.0 mask 255.255.240.0
neighbor 30.30.30.1 remote-as 30
neighbor 30.30.30.1 prefix-list SUB-B out

!
ip prefix-list SUB-B permit 100.100.0.0/19
ip prefix-list SUB-B permit
100.100.16.0/20

!
ip route 100.100.0.0 255.255.224.0 null0
ip route 100.100.16.0 255.255.240.0 null0
```

Advertise both aggregate and second sub-prefix in BGP

Advertise sub-aggregate along with the aggregate

Sub-aggregate should exist in the routing table (pull-up route)

Load Sharing – Outbound (Full)

- What about outbound traffic load balancing?

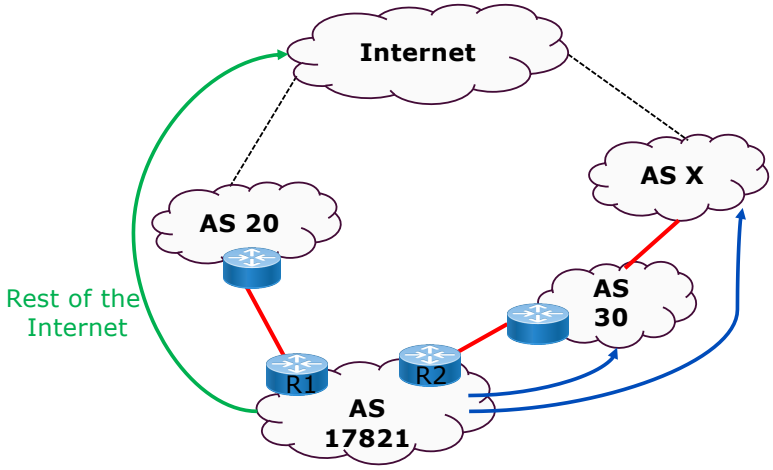
- Case I: Full Internet routes (more memory/CPU)
  - Accept full route from both (AS20 and AS30)
  - For routes from AS30 (R2)
    - Higher local pref prefixes originated by AS30 and its immediate neighbors (one AS hop away) – traffic goes via AS30
    - Lower local pref all other routes (lower than 100) – traffic to these goes via AS20
  - For routes learned from AS20 (R1)
    - Use the default local pref

Load Sharing – Outbound (Partial)

- Partial Routes – less HW resources!
- Case II: Partial Internet routes
  - Accept default from AS20
  - Default and full from AS30 (well-connected than AS20)
  - filter to only accept prefixes originated by AS30 and its neighbor ASes (AS-Path ACLs)
    - Higher pref those routes
    - Low pref the default route
    - so that traffic to these goes via AS20
  - Traffic to rest of Internet via AS 20

171

Load Sharing – Outbound (Partial)



172

Load Sharing – Outbound (Partial)

- R1 configuration:

```
router bgp 17821
neighbor 20.20.20.1 remote-as 20
neighbor 20.20.20.1 prefix-list DEF in
!
ip prefix-list DEF permit 0.0.0.0/0
!
```


Load Sharing – Outbound (Partial)

- R2 config:

```
router bgp 17821
neighbor 30.30.30.1 remote-as 30
neighbor 30.30.30.1 filter-list 30 in
neighbor 30.30.30.1 prefix-list ALL in
neighbor 30.30.30.1 route-map DEF-LOW in
!
ip prefix-list DEF permit 0.0.0.0/0
!
ip prefix-list ALL deny <bogons-rfc1918>
ip prefix-list ALL permit 0.0.0.0/0 le 32
!
ip as-path access-list 30 permit ^(30_)+$
ip as-path access-list 30 permit ^(30_)+_[0-9]+$
!
route-map DEF-LOW permit 10
 match ip address prefix-list DEF
 set local-preference 90
route-map DEF-LOW permit 20
```

Filter inbound routes with AS-PATH ACL using filter-list

Accept full internet feed except bogon routes and RFC 1918 routes

Purely for redundancy (if path via AS 20 fails)

Accept routes local to and received from AS30 (AS-path prepend included)

Received from AS30 but AS-PATH length of two (its neighbor ASes)

Low-pref default route

Using Communities

- Community attribute provides greater flexibility for traffic shaping than prefix-list
  - Simplifies BGP configuration
  - Greater policy control
- Not sent by default to BGP peers
  - explicitly send (neighbor x.x.x.x send-community)
- Can carry policy information
  - Example:
    - ASN:80 (set local-pref 80)
    - ASN:1 (set as-path prepend ASN)
    - ASN:888 (set ip next-hop 192.0.2.1 – Cymru bogons)

175

Route Server Policy

0:(IXP-AS)	Do not announce prefixes to all peers
0:(PEER-AS)	Do not announce prefixes to certain peer
(IX-AS):(PEER-AS)	Advertise to a certain peer
(IX-AS):(IX-AS)	Advertise prefixes to all peers

176

Setting Communities

```
router bgp 17821
neighbor 20.20.20.1 remote-as 20
neighbor 20.20.20.1 send-community
!
address-family ipv4 unicast
 network 100.100.0.0 mask 255.255.224.0 route-map SET-COMM-AGG
 network 100.100.0.0 mask 255.255.248.0 route-map SET-COMM-4G
 network 100.100.8.0 mask 255.255.248.0 route-map SET-COMM-BB
 network 100.100.16.0 mask 255.255.248.0 route-map SET-COMM-ENT
 network 100.100.24.0 mask 255.255.248.0 route-map SET-COMM-CORP
!
ip route 100.100.0.0 255.255.224.0 null0
ip route 100.100.0.0 255.255.248.0 null0 254
ip route 100.100.8.0 255.255.248.0 null0 254
ip route 100.100.16.0 255.255.248.0 null0 254
ip route 100.100.24.0 255.255.248.0 null0 254
!
```


177

Setting Communities

```
!
route-map SET-COMM-AGG permit 10
 set community 100:1000
!
route-map SET-COMM-4G permit 10
 set community 100:1101
!
route-map SET-COMM-BB permit 10
 set community 100:1102
!
route-map SET-COMM-ENT permit 10
 set community 100:1103
!
route-map SET-COMM-CORP permit 10
 set community 100:1104
!
```


178

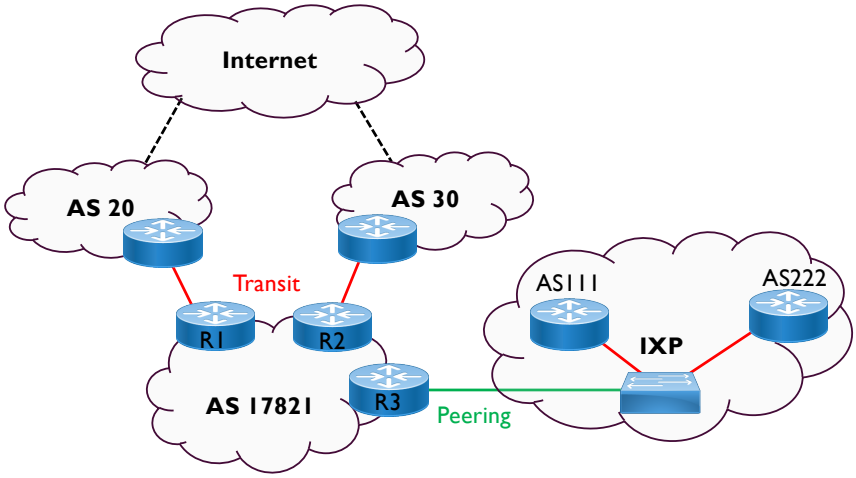
Grouping Communities

- We can group communities together using community-list:

```
!
ip community-list 20 permit 100:1000
ip community-list 21 permit 100:1101
ip community-list 22 permit 100:1102
ip community-list 23 permit 100:1103
ip community-list 24 permit 100:1104
!
```


179

Two Upstream and IXP – using Communities



180

Two Upstream and IXP – IX Router

- R3 (IXP) configuration:
  - both incoming and outgoing traffic, IXP should be the preferred path!

```
router bgp 17821
 neighbor IX-PEERS peer-group
 neighbor 12.12.12.111 remote-as 111
 neighbor 12.12.12.111 peer-group IX-PEERS
 neighbor 12.12.12.222 remote-as 222
 neighbor 12.12.12.222 peer-group IX-PEERS
!
 address-family ipv4
  neighbor IX-PEERS send-community
  neighbor IX-PEERS remove-private-as
  neighbor IX-PEERS route-map IX-IN in
  neighbor IX-PEERS route-map IX-OUT out
```

Define peer-groups for all IX peers

Add neighbors to the peer group

Define common policies applied to all neighbors on the peer-group
- Send communities
- Remove private ASNs

Apply inbound and outbound routing policies

181

Two Upstream and IXP – IX Router

- R3 (IXP) configuration (contd..):

```
!
ip community-list 20 permit 100:1000
ip community-list 21 permit 100:1101
ip community-list 22 permit 100:1102
ip community-list 23 permit 100:1103
ip community-list 24 permit 100:1104
!
route-map IX-IN permit 10
 set local-preference 250
 set community 100:1212 add !(IX ASN)
!
route-map IX-OUT permit 10
 match community 20 21 22 23 24
 set metric 10
!
```

Define the communities

High local-pref for routes received from IX peers (outbound traffic via IX)

Define a community for all routes learned via IXP

Send all our prefixes (aggregates and sub-aggregates)

Set lower MED for all routes sent to IX peers (inbound traffic via IX)

182

Two Upstream and IXP – Transit Router

- For Transit/Upstream:
  - Tier-1 ISPs (or ISPs who are run properly) use communities to group their regional prefixes
  - Filter based on those to shape outbound traffic to Internet!
    - Ex: receive US routes from one ISP, and Europe routes from the other
  - Example:
    - NTT US – 2914:3000
    - NTT Europe – 2914:3200
    - NTT Asia – 2914:3400
    - NTT South America – 2914:3600

183

Two Upstream and IXP – Transit Router

- For Inbound traffic:
  - We can use our sub-prefixes to balance incoming traffic
  - Ex: Advertise half of our routes to one, and the other half to the other
    - keep playing until we reach symmetry!
  - But remember to announce the aggregate to both (REDUNDANCY!)

184

Two Upstream and IXP – TR1

- R1 configuration:
  - Let us assume NTT (AS2914) as transit here

```
router bgp 17821
 neighbor 29.29.29.1 remote-as 2914
 neighbor 29.29.29.1 description eBGP with
NTT
!
 address-family ipv4
  neighbor 29.29.29.1 send-community
  neighbor 29.29.29.1 route-map NTT-IN in
  neighbor 29.29.29.1 route-map NTT-OUT out
!
! We want Asia, US and SA routes
ip community-list 1 permit 2914:3000 !US
ip community-list 1 permit 2914:3400 !AS
ip community-list 1 permit 2914:3600 !SA
ip community-list 2 permit 2914:3200 !EU
```

- Send communities
- Apply inbound and outbound routing policies

Define communities for NTT global routes
- In this example, we will source US and Asia routes from NTT

185

Two Upstream and IXP – TR1

- R1 configuration (contd..):

```
!
route-map NTT-IN permit 10
 match community 1
 set local-preference 210
route-map NTT-IN permit 20
 match community 2
 set local-preference 50
route-map NTT-IN permit 40
!
route-map NTT-OUT permit 10
 match community 20
 match community 21
 match community 22
!
```

Route-map to influence outbound traffic
- Set higher local-pref for US, Asia, and SA routes (outbound traffic)
- Still lower than IX!

Lower local-pref for EU routes (will prefer the second ISP, but available if that link fails)

Route-map to influence inbound traffic
- Send our aggregate (in case ISP2 fails)
- And half of our sub-prefixes

186

Two Upstream and IXP – TR2

- R2 configuration:
  - Let us assume Zayo (AS6461) as transit here

```
router bgp 17821
 neighbor 64.64.64.1 remote-as 6461
 neighbor 64.64.64.1 description eBGP with
Zayo
!
 address-family ipv4
  neighbor 64.64.64.1 send-community
  neighbor 64.64.64.1 route-map ZAYO-IN in
  neighbor 64.64.64.1 route-map ZAYO-OUT out
!
! Zayo Europe routes
ip community-list 3 permit 6461:5996
ip community-list 3 permit 6461:5998
ip community-list 3 permit 6461:5999
! Zayo Global routes
ip community-list 4 permit 6461:5997
```

- Send communities
- Apply inbound and outbound routing policies

Define communities for Zayo global routes
- In this example, we will source EU routes from Zayo

187

Two Upstream and IXP – TR2

- R2 configuration (contd..):

```
!
route-map ZAYO-IN permit 10
 match community 3
 set local-preference 210
route-map ZAYO-IN permit 20
 match community 4
 set local-preference 50
route-map ZAYO-IN permit 40
!
route-map ZAYO-OUT permit 10
 match community 20
 match community 23
 match community 24
!
```

Route-map to influence outbound traffic
- Set higher local-pref for EU routes (outbound traffic)
- Still lower than IX!

Lower local-pref for global routes (NTT is preferred, but will work if that link fails)

Route-map to influence inbound traffic
- Send our aggregate (in case ISP1 fails), and
- other second-half of our sub-prefixes

188

189

Module 9.1

SECURE INTERNET ROUTING (RPKI & ROV)

APNIC

190

Recent - Fat-finger/Hijacks/Leaks

- BGP Optimizers impact Internet – June 2019
  - Most CF (AS13335) hosted sites were not reachable during the leak
    - About 15% of their global traffic!!
    - ~ 120mins

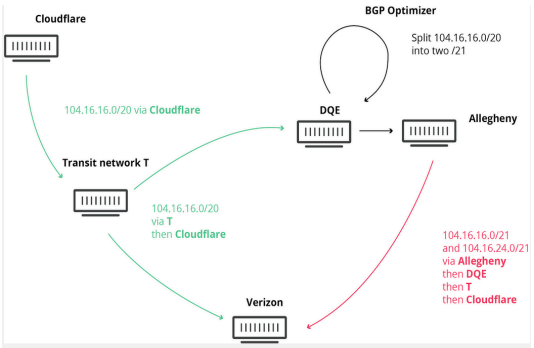
On Mon, Jun 24, 2019 at 3:57 AM [redacted] wrote:
Hello are there any issues with CloudFlare services now?

https://twitter.com/atoonk/status/1143143943531454464/photo/1

191

Recent - Fat-finger/Hijacks/Leaks

- BGP Optimizers impact Internet (contd…)
  - How and What happened?



https://blog.cloudflare.com/how-verizon-and-a-bgp-optimizer-knocked-large-parts-of-the-internet-offline-today/amp/

BGP Optimizers (Was: Validating possible BGP MITM attack)

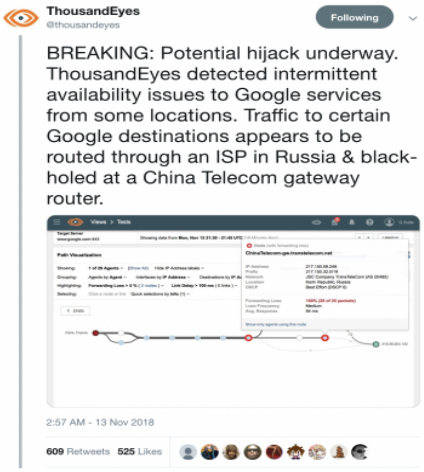
From: Job Snijders <job () ntt net>
Date: Thu, 31 Aug 2017 22:06:49 +0200

```
Dear all,

disclaimer:

    [ The following is targetted at the context where a BGP optimizer
    generates BGP announcement that are ordinarily not seen in the
    Default-Free Zone. The OP indicated they announce a /23, and were
    unpleasantly surprised to see two unauthorized announcements for /24
    more-specifics pop up in their alerting system. No permission was
    granted to create and announce these more-specifics. The AS_PATH
    for those /24 announcements was entirely fabricated. Original thread
    https://mailman.nanog.org/pipermail/nanog/2017-August/092124.html ]

On Thu, Aug 31, 2017 at 11:13:18AM -0700, Andy Litzinger wrote:
  Presuming it was a route optimizer and the issue was ongoing, what
  would be the suggested course of action?

I strongly recommend to turn off those BGP optimizers, glue the ports
shut, burn the hardware, and salt the grounds on which the BGP optimizer
sales people walked.
```


192

Recent - Fat-finger/Hijacks/Leaks

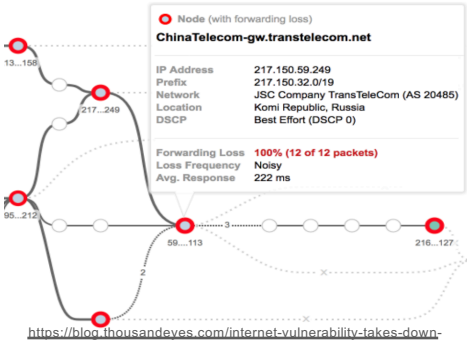
- Google prefix leaks – Nov 2018
  - Google services (G-Suite, Google search and Google analytics) affected by the leak
    - Traffic dropped at AS4809 (China Telecom)
    - ~ 74mins



193

Recent - Fat-finger/Hijacks/Leaks

- Google prefix leaks (contd…)
  - How did it happen?
    - AS37282 (MainOne) leaked Google prefixes to AS4809 (CT) at IXPN, who leaked it to other transit providers like AS20485 (TransTelecom)



https://blog.thousandeyes.com/internet-vulnerability-takes-down-google/

194

Recent - Fat-finger/Hijacks/Leaks

- Amazon (AS16509) Route53 hijack – April2018
  - AS10279 (eNET) originated more specifics (/24s) of Amazon Route53's prefix (205.251.192.0/21)

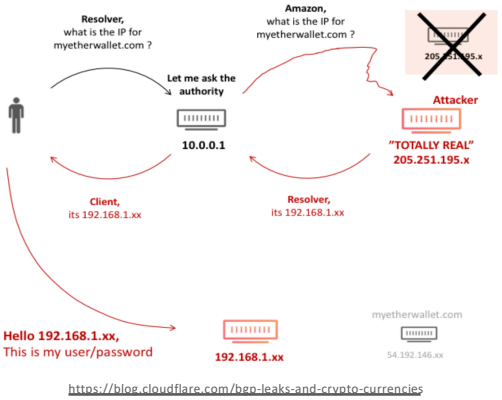
    205.251.192.0/24 ……. 205.251.199.0/24

    https://ip-ranges.amazonaws.com/ip-ranges.json
  - Its peers, like AS6939 (HE), shared these routes with 100s of their own peers…
  - The motive?
    - During the period, DNS servers in the hijacked range only responded to queries for myetherwallet.com
    - Responded with addresses associated with AS41995/AS48693

195

Recent - Fat-finger/Hijacks/Leaks

- Route53 hijack (contd…)
  - Resolvers querying any Route53 managed names, would ask the authoritative servers controlled through the BGP hijack
    - *Possibly, used an automated cert issuer to get a cert for* myetherwallet.com
  - use _*THEIR*_ crypto to end-users to see everything (including passwords)



196

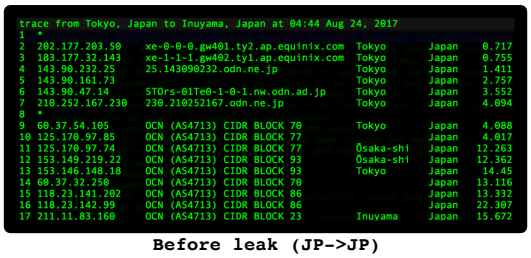
Recent - Fat-finger/Hijacks/Leaks

- Bharti (AS9498) originates 103.0.0.0/10 - Dec 2017
  - ~ 2 days
  - No damage done – more than 8K specific routes!

- Google brings down Internet in Japan - Aug 2017
  - ~ 24 hours)
  - Google (AS15169) leaked >130K prefixes to Verizon (AS701) – in Chicago
    - Normally ~ 50 prefixes
    - ~25K of those were NTT OCN's (AS4713) more specifics
    - which was leaked onwards to KDDI and IIJ (and accepted)
  - Everyone who received the leaked more specifics, preferred the Verizon-Google path to reach NTT OCN!

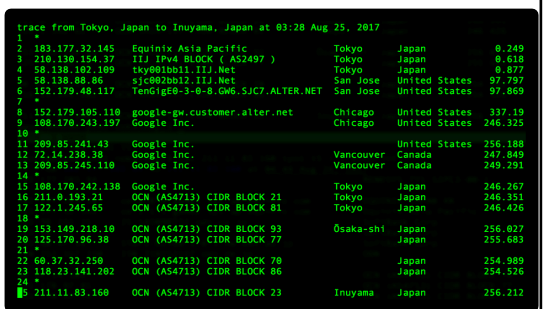
Recent - Fat-finger/Hijacks/Leaks

- Google leak (contd…)

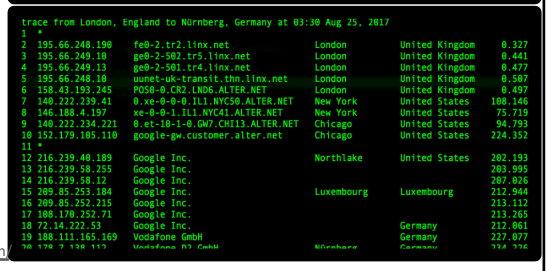


Before leak (JP->JP)

After leak (JP->JP)



After leak (EU->EU)



https://dyn.com/blog/large-bgp-leak-by-google-disrupts-internet-in-japan/

Fat-finger/Hijacks/Leaks

- YouTube (AS36561) Incident - Feb 2008
  - ~ 2 hours
  - AS17557 (PT) announced 208.65.153.0/24 (208.65.152.0/22)
    - Propagated by AS3491 (PCCW)

199

Why do we keep seeing these?

- Because NO ONE is in charge?
  - No single authority model for the Internet
  - No reference point for what's right in routing

200

Why do we keep seeing these?

- Routing works by RUMOUR
  - Tell what you know to your neighbors, and Learn what your neighbors know
  - Assume everyone is correct (and honest)
    - Is the originating network the rightful owner?

201

Why do we keep seeing these?

- Routing is VARIABLE
  - The view of the network depends on where you are
    - Different routing outcomes at different locations
  - ~ no reference view to compare the local view ☹

202

Why do we keep seeing these?

- Routing works in REVERSE
  - Outbound advertisement affects inbound traffic
  - Inbound (Accepted) advertisement influence outbound traffic

Why do we keep seeing these?

- As always, there is no E-bit (evil!)
  - A bad routing update does not identify itself as BAD
  - All we can do is identify GOOD updates
  - But how do we identify what is GOOD???

Why should we worry?

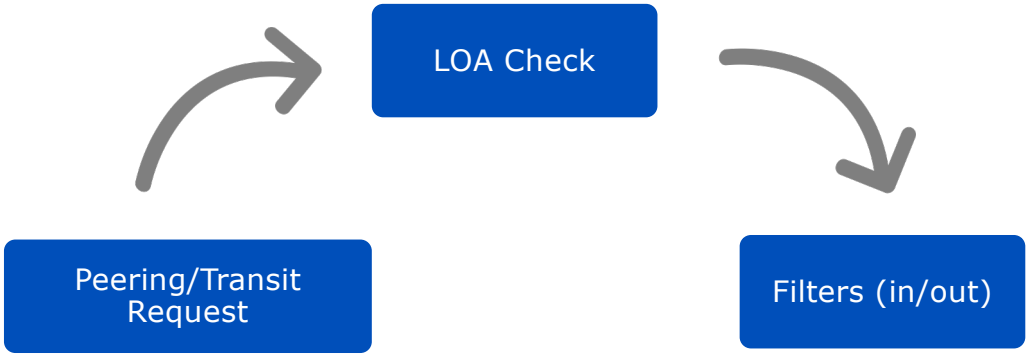
- Because it's just so easy to do bad in routing!

By Source (WP:NFCC#4), Fair use, https://en.wikipedia.org/w/index.php?curid=42515224

How do we address these?

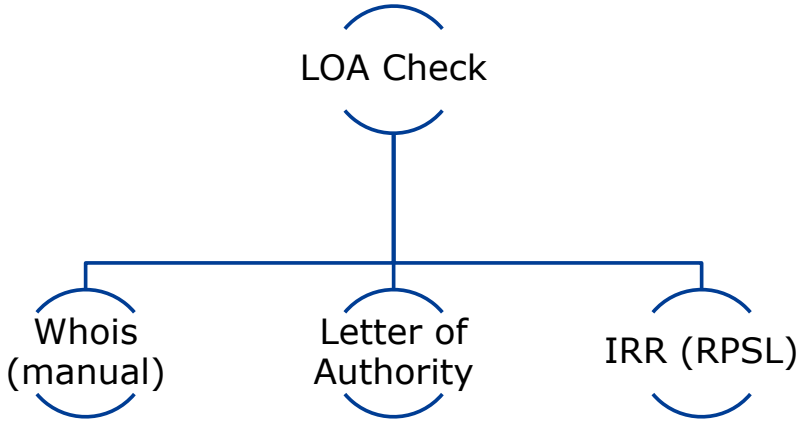
- Filtering!
  - Filters with your peers, upstream(s) and customers
    - Prefix filters
    - Prefix limit
    - AS-PATH filters
    - AS-PATH limit
    - RFC 8212 – BGP default reject or something similar

Current practice



207

Tools & Techniques



208

Tools & Techniques

- Look up whois
  - verify holder of a resource

```
role:           APNIC Training
address:        6 Cordelia Street
address:        South Brisbane
address:        QLD 4101
country:        AU
phone:          +61 7 3858 3100
fax-no:         +61 7 3858 3199
e-mail:         training@apnic.net
admin-c:        JW3997-AP
tech-c:         JW3997-AP
nic-hdl:        AT480-AP
mnt-by:         MAINT-AU-APNICTRAINING
last-modified:  2017-08-22T04:59:14Z
source:         APNIC

% Information related to '202.125.96.0/24AS131107'

route:          202.125.96.0/24
descr:          Prefix for APNICTRAINING LAB DC
origin:         AS131107
mnt-by:         MAINT-AU-APNICTRAINING
country:        AU
last-modified:  2016-06-16T23:23:00Z
source:         APNIC
```

```
tashi@tashi ~> whois -h whois.apnic.net 202.125.96.0
% [whois.apnic.net]
% Whois data copyright terms    http://www.apnic.net/db/dbcopyright.html

% Information related to '202.125.96.0 - 202.125.96.255'

% Abuse contact for '202.125.96.0 - 202.125.96.255' is 'training@apnic.net'

inetnum:        202.125.96.0 - 202.125.96.255
netname:        APNICTRAINING-AP
descr:          Prefix for APNICTRAINING LAB DC
country:        AU
admin-c:        AT480-AP
tech-c:         AT480-AP
status:         ALLOCATED NON-PORTABLE
mnt-by:         MAINT-AU-APNICTRAINING
mnt-irt:        IRT-APNICTRAINING-AU
last-modified:  2016-06-17T00:17:28Z
source:         APNIC

irt:            IRT-APNICTRAINING-AU
address:        6 Cordelia Street
address:        South Brisbane
address:        QLD 4101
e-mail:         training@apnic.net
abuse-mailbox:  training@apnic.net
admin-c:        AT480-AP
tech-c:         AT480-AP
auth:           # Filtered
mnt-by:         MAINT-AU-APNICTRAINING
last-modified:  2013-10-31T11:01:10Z
source:         APNIC
```


209

Tools & Techniques

- Ask for a Letter of Authority
  - Absolve from any liabilities

APNIC

Asia Pacific Network Information Centre
APNIC Pty Ltd
ABN: 42 081 528 010
6 Cordelia Street
PO Box 3646
South Brisbane
QLD 4101 AUSTRALIA
URL www.apnic.net
Enquiries helpdesk@apnic.net
Accounts billing@apnic.net
Phone +61 7 3858 3100
Fax +61 7 3858 3199

31/03/2018

Letter of Authorization

To whom it may concern,

APNIC Training (AS45192) runs a lab network to reproduce technical problems faced by members to help troubleshoot specific issues.

This letter serves as an authorization for APNIC Infra (AS4608) to advertise the following address blocks:

202.125.96.0/24

As a representative of APNIC Training team, that is the owner of the subnet and ASN, I hereby declare that I am authorized to sign this LOA.

Tashi Phuntsho
Training Delivery Manager

Email: tashi@apnic.net
Phone: +61 7 3858 3114

210

Tools & Techniques

- Look up (or ask to enter) details in internet routing registries (IRR)
  - describes route origination and inter-AS routing policies

```
tashi@tashi ~> whois -h whois.radb.net 61.45.248.0/24
route:          61.45.248.0/24
descr:          APNICTRAINING-DC
origin:         AS135533
mnt-by:         MAINT-AS4826
changed:        noc@vocus.com.au 20160702
source:         RADB

route:          61.45.248.0/24
descr:          Prefix for APNICTRAINING LAB - AS135533
origin:         AS135533
mnt-by:         MAINT-AU-APNICTRAININGLAB
country:        AU
last-modified:  2017-10-19T01:36:37Z
source:         APNIC
```

```
tashi@tashi ~> whois -h whois.radb.net AS17660
aut-num:        AS17660
as-name:        BT-Bhutan
descr:          Divinetworks for BT
admin-c:        DUMY-RIPE
tech-c:         DUMY-RIPE
status:         OTHER
mnt-by:         YP67641-MNT
mnt-by:         ES6436-RIPE
created:        2012-11-29T10:31:33Z
last-modified:  2018-09-04T15:26:24Z
source:         RIPE-NONAUTH
remarks:        ****************************
remarks:        * THIS OBJECT IS MODIFIED
remarks:        * Please note that all data that is generally regarded as personal
remarks:        * data has been removed from this object.
remarks:        * To view the original object, please query the RIPE Database at:
remarks:        * http://www.ripe.net/whois
remarks:        ****************************

aut-num:        AS17660
as-name:        DRUKNET-AS
descr:          DrukNet ISP
descr:          Bhutan Telecom
descr:          Thimphu
country:        BT
org:            ORG-BTL2-AP
import:         from AS6461   action pref=100;     accept ANY
export:         to AS6461     announce AS-DRUKNET-TRANSIT
import:         from AS2914   action pref=150;     accept ANY
export:         to AS2914     announce AS-DRUKNET-TRANSIT
import:         from AS6453   action pref=100;     accept ANY
export:         to AS6453     announce AS-DRUKNET-TRANSIT
```


Tools & Techniques

- IRR
  - Helps auto generate network (prefix/as-path) filters using RPSL tools
    - Filter out route advertisements not described in the registry

```
tashi@tashi ~> bgpq3 -Al PEER-v4IN AS17660
no ip prefix-list PEER-v4IN
ip prefix-list PEER-v4IN permit 45.64.248.0/22
ip prefix-list PEER-v4IN permit 103.7.252.0/22
ip prefix-list PEER-v4IN permit 103.7.254.0/23
ip prefix-list PEER-v4IN permit 103.245.240.0/22
ip prefix-list PEER-v4IN permit 103.245.242.0/23
ip prefix-list PEER-v4IN permit 119.2.96.0/19
ip prefix-list PEER-v4IN permit 119.2.96.0/20
ip prefix-list PEER-v4IN permit 202.89.24.0/21
ip prefix-list PEER-v4IN permit 202.144.128.0/19
ip prefix-list PEER-v4IN permit 202.144.128.0/23
ip prefix-list PEER-v4IN permit 202.144.144.0/20
ip prefix-list PEER-v4IN permit 202.144.148.0/22
tashi@tashi ~> bgpq3 -6Al PEER-v6IN AS17660
no ipv6 prefix-list PEER-v6IN
ipv6 prefix-list PEER-v6IN permit 2405:d000::/32
ipv6 prefix-list PEER-v6IN permit 2405:d000:7000::/36
```

```
tashi@tashi ~> bgpq3 -Abl PEER-v4IN AS17660
PEER-v4IN = [
    45.64.248.0/22,
    103.7.252.0/22,
    103.7.254.0/23,
    103.245.240.0/22,
    103.245.242.0/23,
    119.2.96.0/19,
    119.2.96.0/20,
    202.89.24.0/21,
    202.144.128.0/19,
    202.144.128.0/23,
    202.144.144.0/20,
    202.144.148.0/22
];
tashi@tashi ~> bgpq3 -6Abl PEER-v6IN AS17660
PEER-v6IN = [
    2405:d000::/32,
    2405:d000:7000::/36
];
```

```
tashi@tashi ~> bgpq3 -f 38195 -lSUPERLOOP-IN AS-SUPERLOOP
no ip as-path access-list SUPERLOOP-IN
ip as-path access-list SUPERLOOP-IN permit ^38195(_38195)*$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(681|4647|4749|4785)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(4846|4858|7477|7578)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(7585|7604|7628|7631)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(7699|9290|9297|9336)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(9499|9544|9549|10143)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(10145|11031|12041|15133)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(15967|17462|17498|17766)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(17829|17907|17991|18000)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(18110|18201|18292|23156)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(23456|23677|23858|23935)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(24007|24065|24093|24129)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(24231|24233|24238|24341)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(24459|27232|30215|30762)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(36351|37993|38263|38269)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(38451|38534|38549|38570)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(38595|38716|38719|38790)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(38809|38830|38858|42909)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(44239|45158|45267|45278)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(45570|45577|45638|45671)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(45844|46571|55411|55419)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(55455|55506|55575|55707)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(55752|55766|55803|55845)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(55884|55931|55954|56037)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(56098|56135|56178|56225)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(56271|56287|58422|58443)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(58511|58606|58634|58676)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(58712|58739|58750|58868)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(58914|59256|59330|59339)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(59356|60592|60758|63926)$
ip as-path access-list SUPERLOOP-IN permit ^38195(_[0-9]+)*_(63937|63956)$
```

Tools & Techniques

- Problem(s) with IRR
  - No single authority model
    - How do I know if a RR entry is genuine and correct?
    - How do I differentiate between a current and a lapsed entry?
  - Many RRs
    - If two RRs contain conflicting data, which one do I trust and use?
  - Incomplete data - Not all resources are registered in an IRR
    - If a route is not in a RR, is the route invalid or is the RR just missing data?
  - Scaling
    - How do I apply IRR filters to upstream(s)?

213

Tools & Techniques

- Automating network filters (IRR filters) - Caution
  - IRR filters only as good as the correctness of the IRR entries
    - Might require manual overrides and offline verification of resource holders
    - Good idea to use specific sources (-S in bgpq3, -s in rtconfig) when generating filters, assuming mirrors are up to date

214

Back to basics – identify GOOD

- Could we use a digital signature to convey the "authority to use"?
  - Using a private key to sign the authority, and
  - the public key to validate the authority
- The idea being:
  - If the holder of the resource has the private key, it can sign/authorize the use of the resource

How about trust?

- How do we build a chain of trust in this framework??
  - Follow the resource allocation/delegation hierarchy

```
IANA → RIRs → NIRs/LIRs → End Holders
          |
          v
     End Holders
```

- To describe the address allocation using digital certificates

RPKI Chain of Trust

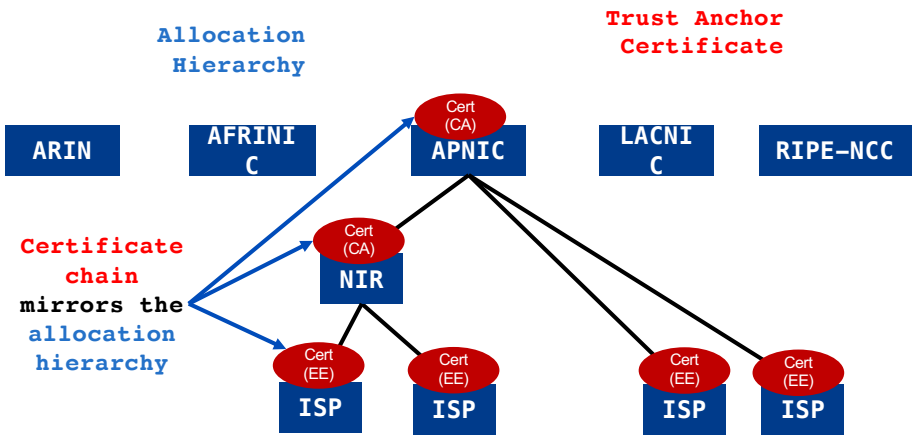


Image 4

217

RPKI Chain of Trust

- RIRs hold a self-signed root certificate for all the resources they have in the registry
  - they are the Trust Anchor for the system
- The root certificate signs the resource certificates for end-holder allocations
  - binds the resources to the end-holders public key
- Any attestations signed by the end-holder's private key, can now be validated up the chain of trust

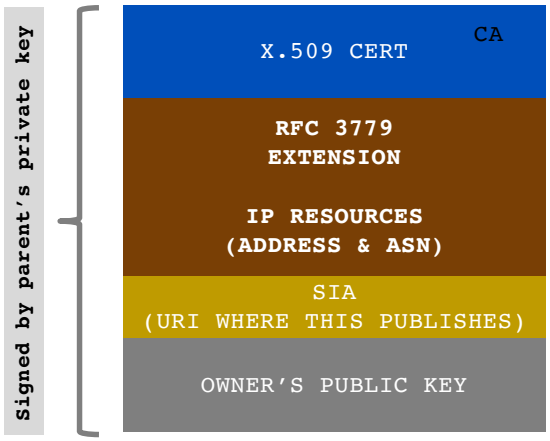
218

X.509 Certificates recap (RFC5280)

- Associates a public key with an individual or an organization

VERSION	Version of X.509
SERIAL NUMBER	Uniquely identifies the certificate
SIGNATURE ALGORITHM	Algorithms used by the CA to sign the cert
ISSUER NAME	Id of the CA (that issued the cert)
VALIDITY PERIOD	Cert validity
SUBJECT NAME	Entity associated with the public key
SUBJECT PUBLIC KEY	Owner's public key
EXTENSIONS (ISSUER KEY ID)	Identify the pub key of issuer of the cert
EXTENSIONS (SUBJECT KEY ID)	Extra info (owner of the cert)
EXTENSIONS (CRL)	Extensions (CRL)
CA DIGITAL SIGNATURE	Certifies the binding between the pub key & subject of the cert

RPKI profile ~ Resource Certificates



- RFC 3779 extensions – binds a list of resources (IPv4/v6,ASN) to the subject of the certificate (private key holder)
- SIA (subject information access) contains a URI that identifies the publication point of the objects signed by the subject of the cert.

Resource Certificates

- When an address holder A (*IRs) allocates resources (IP address/ASN) to B (end holders)
  - A issues a public-key/resource certificat that binds the allocated address with B's public key, all signed by A's (CA) private key
  - The resource certificate proves the holder of the private key (B) is the legitimate holder of the number resource!

Route Origin Authorization (ROA)

- The resource holder (B) can now sign authorities using its private key, which can be validated by any third party against the TA
- For routing, the address holder can authorize a network (ASN) to originate a route into the BGP routing system, and sign this permission with its private key (ROA)

Route Origin Authorization (ROA)

- Digitally signed object
  - list of prefixes and the nominated ASN
  - can be verified cryptographically

Prefix	203.176.32.0/19
Max-length	/24
Origin ASN	AS17821

- ** *Multiple ROAs can exist for the same prefix*

223

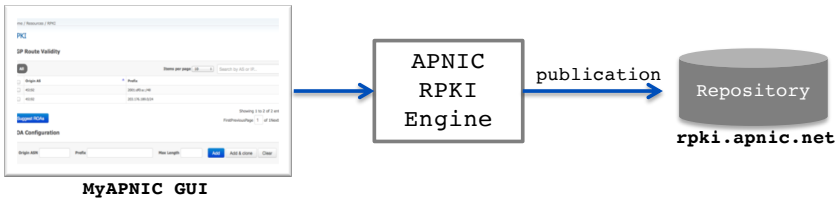
What can RPKI do?

- Authoritatively proof:
  - Who is the legitimate owner of an address, and
  - Identify which ASNs have the permission from the holder to originate the address
- Hence, can help:
  - prevent route hijacks/mis-origination/misconfiguration

224

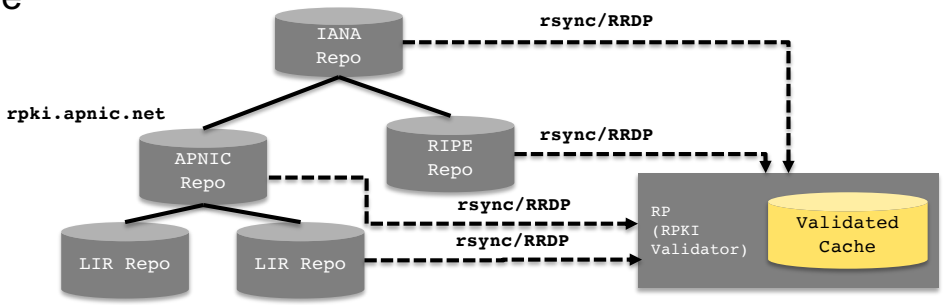
RPKI Components

- Issuing Party – Internet Registries (*IRs)
  - Certificate Authority (CA) that issues resource certificates to end-holders
  - Publishes the objects (ROAs) signed by the resource certificate holders



RPKI Components

- Relying Party (RP)
  - RPKI Validator tool that gathers data (ROA) from the distributed RPKI repositories
  - Validates each entry's signature against the TA to build a "Validated cache"

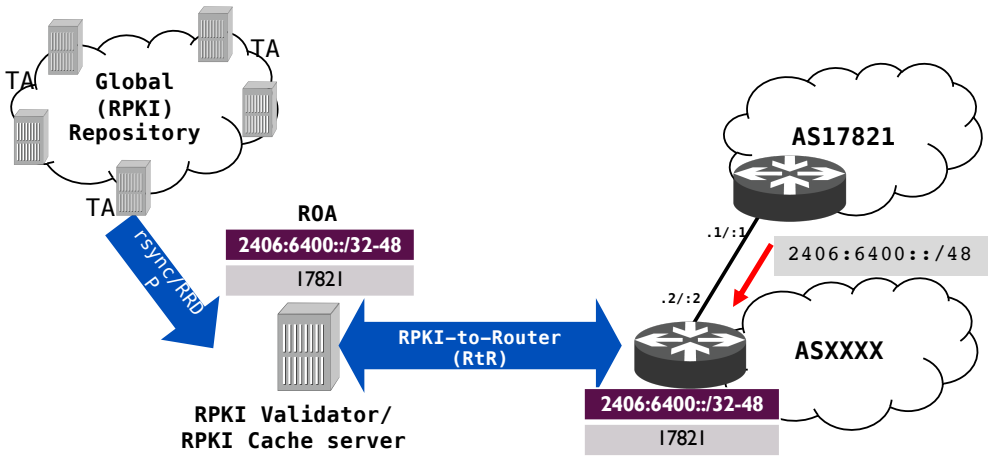


RPKI Service Models

- Hosted model:
  - The RIR (APNIC) runs the CA functions on members' behalf
    - Manage keys, repo, etc.
    - Generate certificates for resource delegations
- Delegated model:
  - Member becomes the CA (delegated by the parent CA) and operates the full RPKI system
    - JPNIC, TWNIC, CNNIC (IDNIC in progress)

227

Route Origin Validation (ROV)



228

Route Origin Validation

- Router fetches ROA information from the validated RPKI cache
  - Crypto stripped by the validator
- BGP checks each received BGP update against the ROA information and labels them

Validation States

- Valid
  - the prefix and AS pair found in the database.
- Invalid
  - prefix is found, but origin AS is wrong, OR
  - the prefix length is longer than the maximum length
- Not Found/Unknown
  - No valid ROA found
    - Neither valid nor invalid (perhaps not created)

Validation States

ROA

ASN	Prefix	Max Length
65420	10.0.0.0/16	18

BGP Routes

ASN	Prefix	RPKI State
65420	10.0.0.0/16	VALID
65420	10.0.128.0/17	VALID
65421	10.0.0.0/16	INVALID
65420	10.0.10.0/24	INVALID
65430	10.0.0.0/8	NOT FOUND

Possible actions - RPKI states

- Do Nothing (observe & learn)
- Tag with BGP communities
  - If you have downstream customers or run a route server (IXP)
    - Let them decide
  - Ex:
    - Valid (ASN:65XX1)
    - Not Found (ASN:65XX2)
    - Invalid (ASN:65XX3)
- Modify preference values
  - RFC7115 (High, Low, Lowest)
- Drop Invalids
  - ~6K IPv4 routes (might want to check your top flows)
    - https://rpki-monitor.antd.nist.gov/index.php?p=3&s=0

ROV – Industry trends

- AT&T (AS7018) drops Invalids!
  - 11 Feb 2019

AT&T/as7018 now drops invalid prefixes from peers

Jay Borkenhagen jayb at braeburn.org
Mon Feb 11 14:53:45 UTC 2019

- Previous message (by thread): BGP topological vs centralized route reflector
- Next message (by thread): AT&T/as7018 now drops invalid prefixes from peers
- **Messages sorted by:** [date] [thread] [subject] [author]

FYI:

The AT&T/as7018 network is now dropping all RPKI-invalid route announcements that we receive from our peers.

We continue to accept invalid route announcements from our customers, at least for now. We are communicating with our customers whose invalid announcements we are propagating, informing them that these routes will be accepted by fewer and fewer networks over time.

Thanks to those of you who are publishing ROAs in the RPKI. We would also like to encourage other networks to join us in taking this step to improve the quality of routing information in the Internet.

Thanks!

Jay B.

ROV – Industry trends

- Workonline Comms (AS37271) & SEACOM (AS37100) drops Invalids!
  - 1 and 5 April 2019 (does not use ARIN's TAL)

[apops] RPKI ROV & Dropping of Invalids - Africa

- ***To***: apops@apops.net
- ***Subject***: [apops] RPKI ROV & Dropping of Invalids - Africa
- ***From***: Mark Tinka <mark.tinka@seacom.mu>
- ***Date***: Tue, 9 Apr 2019 14:05:03 +0200

Hello all.

In November 2018 during the ZAPF (South Africa Peering Forum) meeting in Cape Town, 3 major ISP's in Africa announced that they would enable RPKI's ROV (Route Origin Validation) and the dropping of Invalid routes as part of an effort to clean up the BGP Internet, on the 1st April, 2019.

On the 1st of April, Workonline Communications (AS37271) enabled ROV and the dropping of Invalid routes. This applies to all eBGP sessions for IPv4 and IPv6.

On the 5th of April, SEACOM (AS37100) enabled ROV and the dropping of Invalid routes. This applies to all eBGP sessions with public peers, private peers and transit providers, both for IPv4 and IPv6. eBGP sessions toward downstream customers will follow in 3 months from now.

We are still standing by for the 3rd ISP to complete their implementation, and we are certain they will communicate with the community accordingly.

Please note that for the legal reasons previously discussed on various fora, neither Workonline Communications nor SEACOM are utilising the ARIN TAL. As a result, any routes covered only by a ROA issued under the ARIN TAL will fall back to a status of Not Found. Unfortunately, this means that ARIN members will not see any improved routing security for their prefixes on our networks until this is resolved. We will each re-evaluate this decision if and when ARIN's policy changes. We are hopeful that this will happen sooner rather than later.

If you interconnect with either of us and may be experiencing any routing issues potentially related to this new policy, please feel free to reach out to:

- noc@workonline.africa
- peering@seacom.mu

Workonline Communications and SEACOM hope that this move encourages the rest of the ISP community around the world to ramp up their deployment of RPKI ROV and dropping of Invalid routes, as we appreciate the work that AT&T have carried out in the same vein.

In the mean time, we are happy to answer any questions you may have about our deployments. Thanks.

Mark Tinka (SEACOM) & Ben Maddison (Workonline Communications).

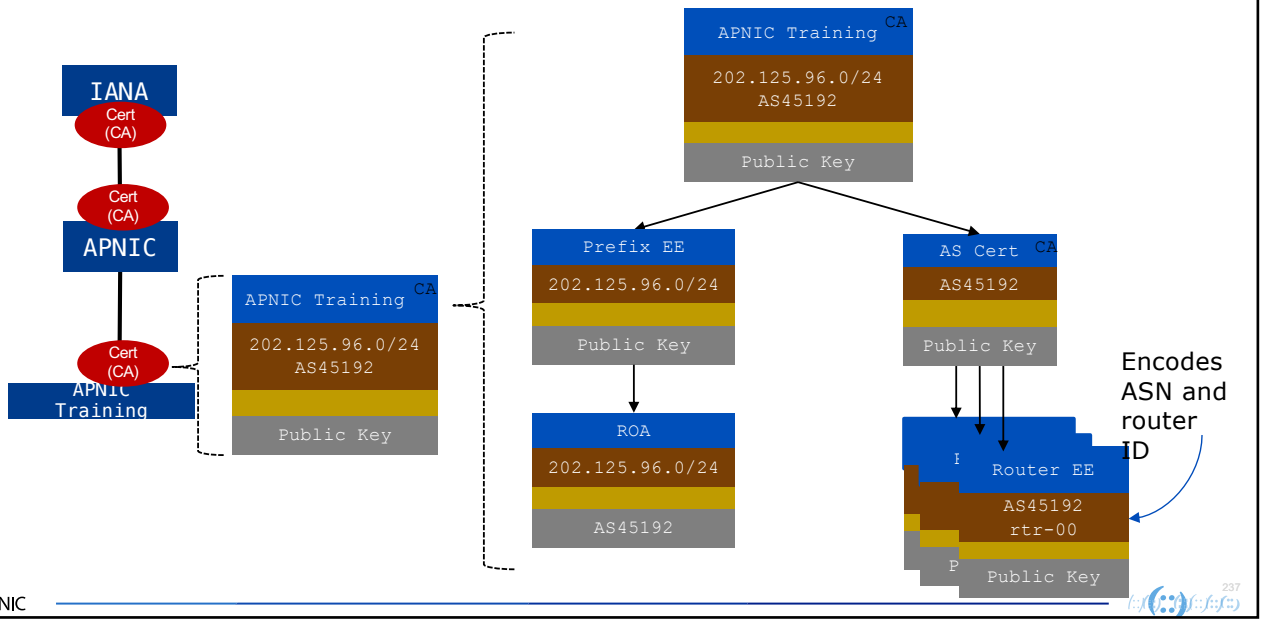
ROV – Industry trends

- MMIX & MyREN are dropping Invalids!
  - Since May and July this year ☺

Are ROAs enough?

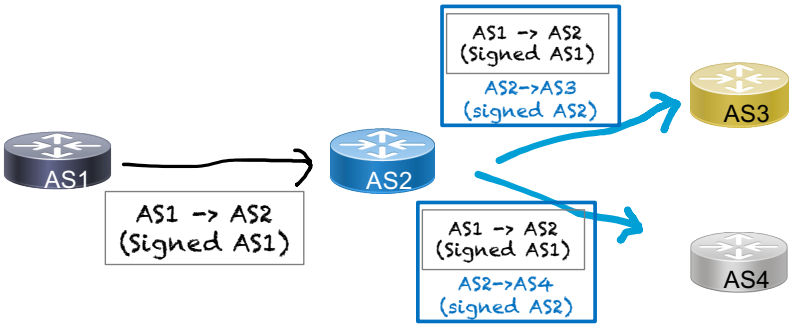
- What if I forge the origin AS in the AS path?
  - Would be accepted as “good” – pass origin validation!
- Which means, we need to secure the AS path as well
  - Need AS path validation (per-prefix)
- We can use RPKI certificates for this

AS keys (per-router keys)



AS path validation - BGPsec

- AS1 router crypto signs the message to AS2
- AS2 router signs the message to AS3 and AS4, encapsulating AS1's message



- A BGPsec speaker validates the received update by checking:
  - If there is a ROA that describes the prefix and origin AS
  - If the received AS path can be validated as a chain of signatures (for each AS in the AS path) using the AS keys

So why is AS path validation NOT happening?

- Cannot have partial adoption
  - Cannot jump across non-participating networks
- More HW resources
  - CPU - high crypto overhead to validate signatures, and
  - Memory
    - Updates in BGPsec would be per prefix
    - New attributes carrying signatures and certs/key IDs for every AS in the AS path
- No clarity on how to distribute the collection of certificates required to validate the signatures
- Given so much overhead, can it prevent more than route hijacks?
  - Route leaks?

239

RPKI Further Reading

- X.509 PKI Certificates

Extensions for IP Addresses and ASNs

Resource Public Key Infrastructure

240

Acknowledgement

- Geoff Huston, APNIC
- Randy Bush, IIJ Labs/Arrcus

241

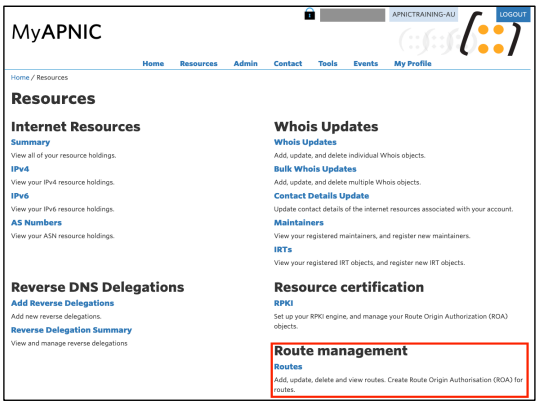
Module 9.2

IMPLEMENTING RPKI

242

Create & publish your ROA

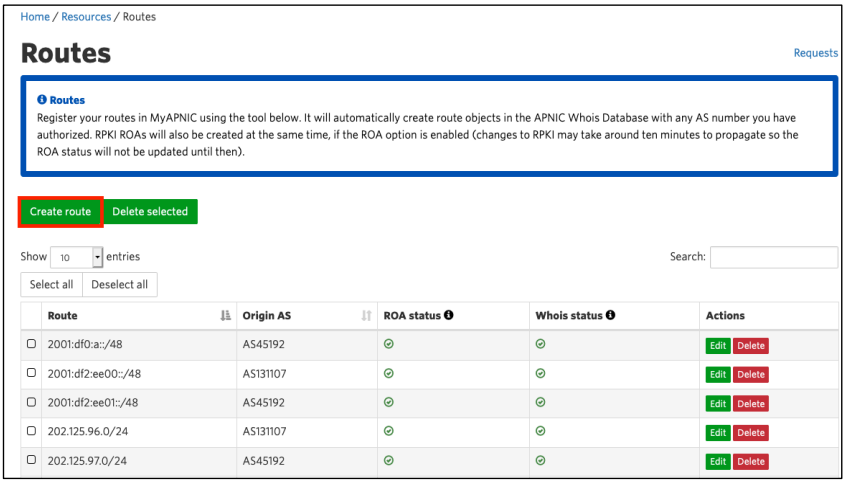
- Login to MyANIC
  - Go to Resources → Route Management (see image below)



https://www.apnic.net/wp-content/uploads/2017/12/ROUTE_MANAGEMENT_GUIDE.pdf

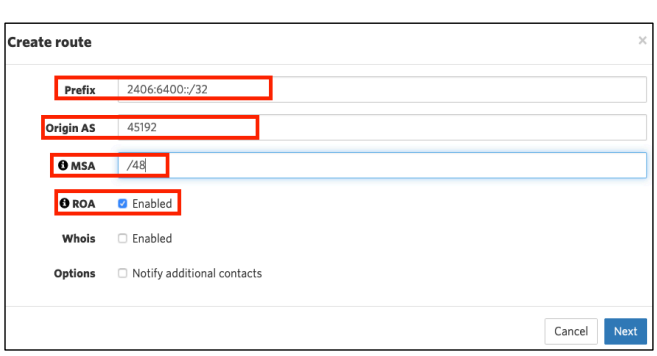
Create (publish) your ROA

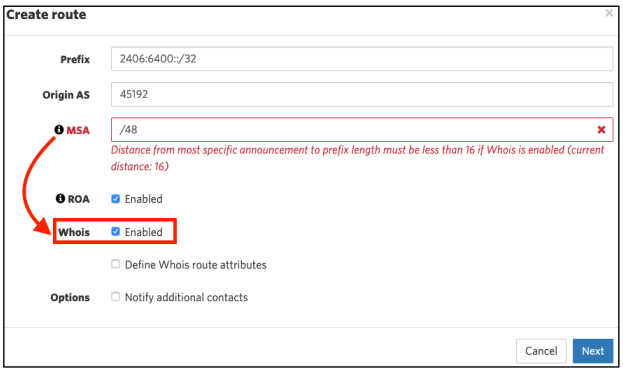
- Select Create route (as shown below)



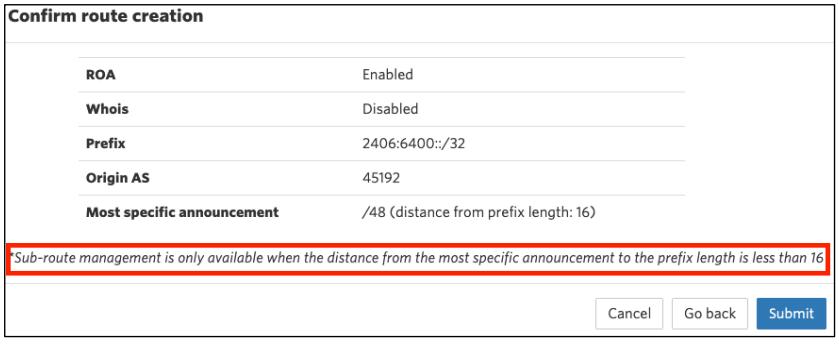
Create (publish) your ROA

- Example for IPv6 below



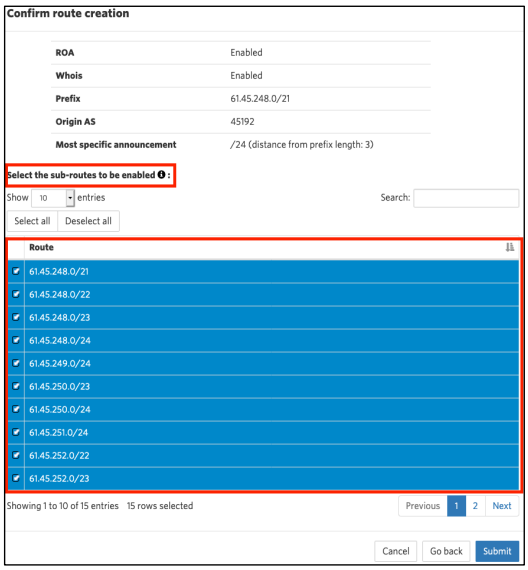


Create (publish) your ROA



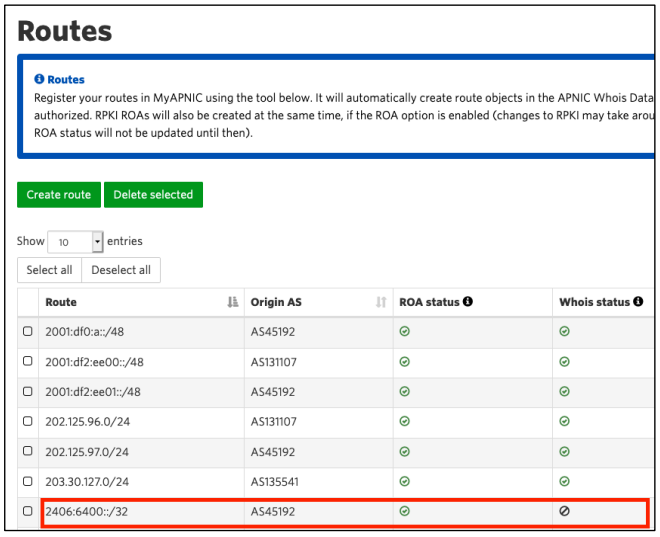
Create (publish) your ROA

- Example for IPv4



Create (publish) your ROA

- Your ROAs are ready!



Check your ROA

http://nong.rand.apnic.net:8080/roas

Validated ROAs

Validated ROAs from **APNIC RPKI Root**, **ARIN**, **AfriNIC RPKI Root**, **LACNIC RPKI Root**, **RIPE NCC RPKI Root**.

Show 10 entries

Search: 61.45.248.0

ASN	Prefix	Maximum Length	Trust Anchor
135533	61.45.248.0/24	24	APNIC RPKI Root

First Previous 1 Next Last

Showing 1 to 1 of 1 entries (filtered from 83,128 total entries)

249

Check your ROA

```
# whois -h rr.ntt.net 2001:df2:ee00::/48

route6:     2001:df2:ee00::/48
descr:      RPKI ROA for 2001:df2:ee00::/48
remarks:    This route object represents routing data retrieved from the RPKI
remarks:    The original data can be found here: https://rpki.gin.ntt.net/r/AS131107/2001:df2:ee00::/48
remarks:    This route object is the result of an automated RPKI-to-IRR conversion process.
remarks:    maxLength 48
origin:     AS131107
mnt-by:     MAINT-JOB
changed:    job@ntt.net 20180802
source:     RPKI  # Trust Anchor: APNIC RPKI Root
```


250

Check your ROA

```
# whois -h whois.bgpmon.net 2001:df2:ee00::/48

Prefix:                 2001:df2:ee00::/48
Prefix description:     APNICTRAINING-DC
Country code:           AU
Origin AS:                      131107
Origin AS Name:         APNICTRAINING LAB DC
RPKI status:            ROA validation successful
First seen:             2016-06-30
Last seen:              2018-01-21
Seen by #peers:         97
```

```
# whois -h whois.bgpmon.net "--roa 131107 2001:df2:ee00::/48"

------------------------
ROA Details
------------------------
Origin ASN:       AS131107
Not valid Before: 2016-09-07 02:10:04
Not valid After:  2020-07-30 00:00:00  Expires in 2y190d9h34m23.2000000029802s
Trust Anchor:     rpki.apnic.net
Prefixes:         2001:df2:ee00::/48 (max length /48) 202.125.96.0/24 (max length /24)
```


Check your ROA

https://bgp.he.net/

Announced By		
Origin AS	**Announcement**	**Description**
AS131107	2001:df2:ee00::/48	testing

Deploy RPKI Validator

- Many options:
  - RIPE RPKI Validator

```
https://www.ripe.net/manage-ips-and-asns/resource-management/certification/tools-and-resources
```

  - Dragon Research Labs RPKI Toolkit

```
https://github.com/dragonresearch/rpki.net
```

  - Routinator

```
https://github.com/NLnetLabs/routinator
```

  - OctoRPKI & GoRTR (Cloudflare's RPKI toolkit)

```
https://github.com/cloudflare/cfrpki
```


Configuration (IOS)

- Establishing session with the validator

```
router bgp 131107
 bgp rpki server tcp <validator-IP> port <323/8282/3323> refresh 120
```

- Note:
  - Cisco IOS by default does not include invalid routes for best path selection!
  - If you don't want to drop invalids, we need explicitly tell BGP (under respective address families)

```
bgp bestpath prefix-validate allow-invalid
```

Configuration (IOS)

- Policies based on validation:

```
route-map ROUTE-VALIDATION permit 10
 match rpki valid
 set local-preference 110
!
route-map ROUTE-VALIDATION permit 20
 match rpki not-found
 set local-preference 100
!
route-map ROUTE-VALIDATION permit 10
 match rpki invalid
 set local-preference 90
!
```


255

Configuration (IOS)

- Apply the route-map to inbound updates

```
router bgp 131107
!---output omitted-----!
 address-family ipv4
  bgp bestpath prefix-validate allow-invalid
  neighbor X.X.X.169 activate
  neighbor X.X.X.169 route-map ROUTE-VALIDATION in
 exit-address-family
 !
 address-family ipv6
  bgp bestpath prefix-validate allow-invalid
  neighbor X6:X6:X6:X6::151 activate
  neighbor X6:X6:X6:X6::151 route-map ROUTE-VALIDATION in
 exit-address-family
!
```


256

Configuration (JunOS)

- Establishing session with the validator

```
routing-options {
   autonomous-system 131107;
   validation {
       group rpki-validator {
           session <validator-IP> {
               refresh-time 120;
               port <323/3323/8282>;
               local-address X.X.X.253;
           }
       }
   }
}
```


257

Configuration (JunOS)

- Define policies based on the validation states

```
policy-options {
   policy-statement ROUTE-VALIDATION {
       term valid {
           from {
               protocol bgp;
               validation-database valid;
           }
           then {
               local-preference 110;
               validation-state valid;
               accept;
           }
       }
       term invalid {
           from {
               protocol bgp;
               validation-database invalid;
           }
           then {
               local-preference 90;
               validation-state invalid;
               accept;
           }
       }
```

```
       term unknown {
           from {
               protocol bgp;
               validation-database unknown;
           }
           then {
               local-preference 100;
               validation-state unknown;
               accept;
           }
       }
   }
}
```


258

Router Configuration (JunOS)

- Apply the policy to inbound updates

```
protocols {
   bgp {
       group external-peers {
           #output-ommitted
           neighbor X.X.X.1 {
               import ROUTE-VALIDATION;
               family inet {
                   unicast;
               }
           }
       }
       group external-peers-v6 {
           #output-ommitted
           neighbor X6:X6:X6:X6::1 {
               import ROUTE-VALIDATION;
               family inet6 {
                   unicast;
               }
           }
       }
   }
}
```


RPKI Verification (IOS)

- IOS has only

```
#sh bgp ipv6 unicast rpki ?
  servers Display RPKI cache server information
  table   Display RPKI table entries


#sh bgp ipv4 unicast rpki ?
  servers Display RPKI cache server information
  table   Display RPKI table entries
```

RPKI Verification (IOS)

- Check the RTR session

```
#sh bgp ipv4 unicast rpki servers

BGP SOVC neighbor is X.X.X.47/323 connected to port 323
Flags 64, Refresh time is 120, Serial number is 1516477445, Session ID is 8871
InQ has 0 messages, OutQ has 0 messages, formatted msg 7826
Session IO flags 3, Session flags 4008
 Neighbor Statistics:
 Prefixes 45661
 Connection attempts: 1
 Connection failures: 0
 Errors sent: 0
 Errors received: 0

Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled, Mininum incoming TTL 0, Outgoing TTL 255
Local host: X.X.X.225, Local port: 29831
Foreign host: X.X.X.47, Foreign port: 323
```


RPKI Verification (IOS)

- Check the RPKI cache

```
#sh bgp ipv4 unicast rpki table
37868 BGP sovc network entries using 6058880 bytes of memory
39655 BGP sovc record entries using 1268960 bytes of memory

Network            Maxlen Origin-AS Source Neighbor
1.9.0.0/16         24     4788      0      202.125.96.47/323
1.9.12.0/24        24     65037     0      202.125.96.47/323
1.9.21.0/24        24     24514     0      202.125.96.47/323
1.9.23.0/24        24     65120     0      202.125.96.47/323
```

```
#sh bgp ipv6 unicast rpki table
5309 BGP sovc network entries using 976856 bytes of memory
6006 BGP sovc record entries using 192192 bytes of memory

Network             Maxlen Origin-AS Source Neighbor
2001:200::/32       32     2500      0      202.125.96.47/323
2001:200:136::/48   48     9367      0      202.125.96.47/323
2001:200:900::/40   40     7660      0      202.125.96.47/323
2001:200:8000::/35  35     4690      0      202.125.96.47/323
```

Check routes (IOS)

```
#sh bgp ipv4 unicast 202.144.128.0/19
BGP routing table entry for 202.144.128.0/19, version 3814371
Paths: (1 available, best #1, table default)
 Advertised to update-groups:
    2
 Refresh Epoch 15
 4826 17660
   49.255.232.169 from 49.255.232.169 (114.31.194.12)
     Origin IGP, metric 0, localpref 110, valid, external, best
     Community: 4826:5101 4826:6570 4826:51011 24115:17660
     path 7F50C7CD98C8 RPKI State valid
     rx pathid: 0, tx pathid: 0x0
```

```
#sh bgp ipv6 unicast 2402:7800::/32
BGP routing table entry for 2402:7800::/32, version 1157916
Paths: (1 available, best #1, table default)
 Advertised to update-groups:
    2
 Refresh Epoch 15
 4826
   2402:7800:10:2::151 from 2402:7800:10:2::151 (114.31.194.12)
     Origin IGP, metric 0, localpref 100, valid, external, best
     Community: 4826:1000 4826:2050 4826:2110 4826:2540 4826:2900 4826:5203
     path 7F50B266CBD8 RPKI State not found
     rx pathid: 0, tx pathid: 0x0
```


263

RPKI Verification (JunOS)

- Check the RPKI cache

```
>show validation session
Session                                 State Flaps   Uptime #IPv4/IPv6 records
X.X.X.46                                    Up      75 09:20:59 40894/6747


>show validation session 202.125.96.46
Session                                 State Flaps   Uptime #IPv4/IPv6 records
X.X.X.46                                    Up       75 09:21:18 40894/6747
```


264

RPKI Verification (JunOS)

- Check the RPKI cache

```
>show validation database
RV database for instance master

Prefix              Origin-AS          Session               State  Mismatch
1.9.0.0/16-24            4788 202.125.96.46             valid
1.9.12.0/24-24          65037 202.125.96.46             valid
1.9.21.0/24-24          24514 202.125.96.46             valid
1.9.23.0/24-24          65120 202.125.96.46             valid

----------
2001:200::/32-32         2500 202.125.96.46             valid
2001:200:136::/48-48     9367 202.125.96.46             valid
2001:200:900::/40-40     7660 202.125.96.46             valid
2001:200:8000::/35-35    4690 202.125.96.46             valid
2001:200:c000::/35-35   23634 202.125.96.46             valid
2001:200:e000::/35-35    7660 202.125.96.46             valid
```

Would have been nice if they had per AF!

265

RPKI Verification (JunOS)

- Can filter per origin ASN

```
>show validation database origin-autonomous-system 45192
RV database for instance master


Prefix              Origin-AS          Session                State          Mismatch
202.125.97.0/24-24      45192          202.125.96.46           valid
203.176.189.0/24-24     45192          202.125.96.46           valid
2001:df2:ee01::/48-48   45192          202.125.96.46           valid


 IPv4 records: 2
 IPv6 records: 1
```

IOS should have something similar!

266

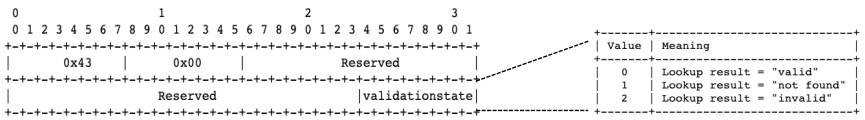
Check routes (JunOS)

```
>show route protocol bgp 202.144.128.0

inet.0: 693024 destinations, 693024 routes (693022 active, 0 holddown, 2
hidden)
+ = Active Route, - = Last Active, * = Both

202.144.128.0/20 *[BGP/170] 1w4d 21:03:04, MED 0, localpref 110, from
202.125.96.254
                      AS path: 4826 17660 I, validation-state: valid
                    >to 202.125.96.225 via ge-1/1/0.0
```

```
>show route protocol bgp 2001:201::/32

inet6.0: 93909 destinations, 93910 routes (93909 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

2001:201::/32     *[BGP/170] 21:18:14, MED 0, localpref 100, from
2001:df2:ee00::1
                      AS path: 65332 I, validation-state: unknown
                    >to fe80::dab1:90ff:fedc:fd07 via ge-1/1/0.0
```


Propagating RPKI states to iBGP peers

- To avoid every BGP speaker having an RTR session, and
- All BGP speakers have consistent information
  - Relies on extended BGP communities (RFC8097)



- Sender (one that has RTR session) attaches the extended community to Updates, and receiver derives the validation states from it
- Must be enabled on both sender and receiver!

Propagating RPKI states (IOS)

- Sender (one with RTR session)

```
router bgp 131107
 bgp rpki server tcp <validator-IP> port <323/8282/3323> refresh 120
!---output omitted-----!
 address-family ipv4
  neighbor X.X.X.X activate
  neighbor X.X.X.X send-community both
  neighbor X.X.X.X announce rpki state
 exit-address-family
 !
 address-family ipv6
  neighbor X6:X6:X6:X6::X6 activate
  neighbor X6:X6:X6:X6::X6 send-community both
  neighbor X6:X6:X6:X6::X6 announce rpki state
 exit-address-family
!
```


Propagating RPKI states (IOS)

- Receiver (iBGP peer)

```
router bgp 131107
!---output omitted-----!
 address-family ipv4
  neighbor Y.Y.Y.Y activate
  neighbor Y.Y.Y.Y send-community both
  neighbor Y.Y.Y.Y announce rpki state
 exit-address-family
 !
 address-family ipv6
  neighbor Y6:Y6:Y6:Y6::Y6 activate
  neighbor Y6:Y6:Y6:Y6::Y6 send-community both
  neighbor Y6:Y6:Y6:Y6::Y6 announce rpki state
 exit-address-family
!
```

  - If announce rpki state is not configured for the neighbor, all prefixes received from the iBGP neighbor will be marked VALID!

Propagating RPKI states (JunOS)

- Sender (one with RTR session)

```
   policy-statement ROUTE-VALIDATION {
       term valid {
           from {
               protocol bgp;
               validation-database valid;
           }
           then {
               local-preference 110;
               validation-state valid;
                   community add origin-validation-state-
valid;
               accept;
           }
       }
       term invalid {
           from {
               protocol bgp;
               validation-database invalid;
           }
           then {
               local-preference 90;
               validation-state invalid;
                   community add origin-validation-state-
invalid;
               accept;
           }
       }
```

```
term unknown {
           from {
               protocol bgp;
               validation-database unknown;
           }
           then {
               local-preference 100;
               validation-state unknown;
                   community add origin-validation-state-
unknown;
               accept;
           }
       }
   }
}
```


271

Propagating RPKI states (JunOS)

- Receiver (iBGP peer)

```
policy-statement ROUTE-VALIDATION-1 {
    term valid {
        from community origin-validation-state-valid;
        then validation-state valid;
    }
    term invalid {
        from community origin-validation-state-invalid;
        then validation-state invalid;
    }
    term unknown {
        from community origin-validation-state-unknown;
        then validation-state unknown;
    }

}
```


272

Propagating RPKI states – potential issues

- IOS as BR, propagating states to JunOS iBGP peers
  - unknown iana 4300
  - Hack:
    - Either act on the states at the border, or
    - Match and tag them with custom communities before propagating

Configuration - Reference Link

- Cisco
  - https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_bgp/configuration/xe-3s/irg-xe-3s-book/irg-origin-as.pdf
- Juniper
  - https://www.juniper.net/documentation/en_US/junos/topics/topic-map/bgp-origin-as-validation.html
- RIPE:
  - https://www.ripe.net/manage-ips-and-asns/resource-management/certification/router-configuration

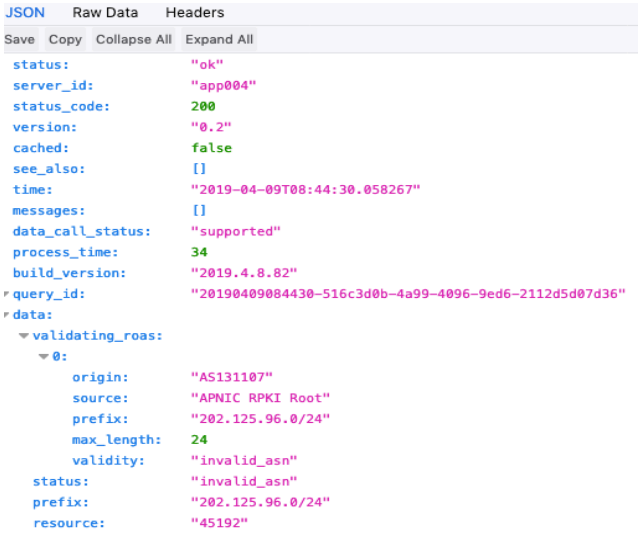
Operational Caveats

- When RTR session goes down, the validation state changes to Not Found for all routes after a while
  - Invalid => Not Found
  - we need at least two RTR sessions and/or need careful filtering policies
- During a router reload, do we receive ROAs first or BGP updates first?
  - If BGP update is faster than ROA, will propagate even invalid routes to its iBGP peers

Useful tools

- RIPEstat – prefix/ASN

https://stat.ripe.net/data/rpki-validation/data.json?resource=45192&prefix=202.125.96.0/24



https://www.apnic.net/community/security/resource-certification/#routing

277

278

Thank You!

END OF SESSION