Source Address Validation Using BGP UPDATEs, ASPA, and ROA (BAR-SAV)

https://datatracker.ietf.org/doc/draft-ietf-sidrops-bar-sav/

K. Sriram

Authors: Kotikalapudi Sriram, Igor Lubashev, and Doug Montgomery

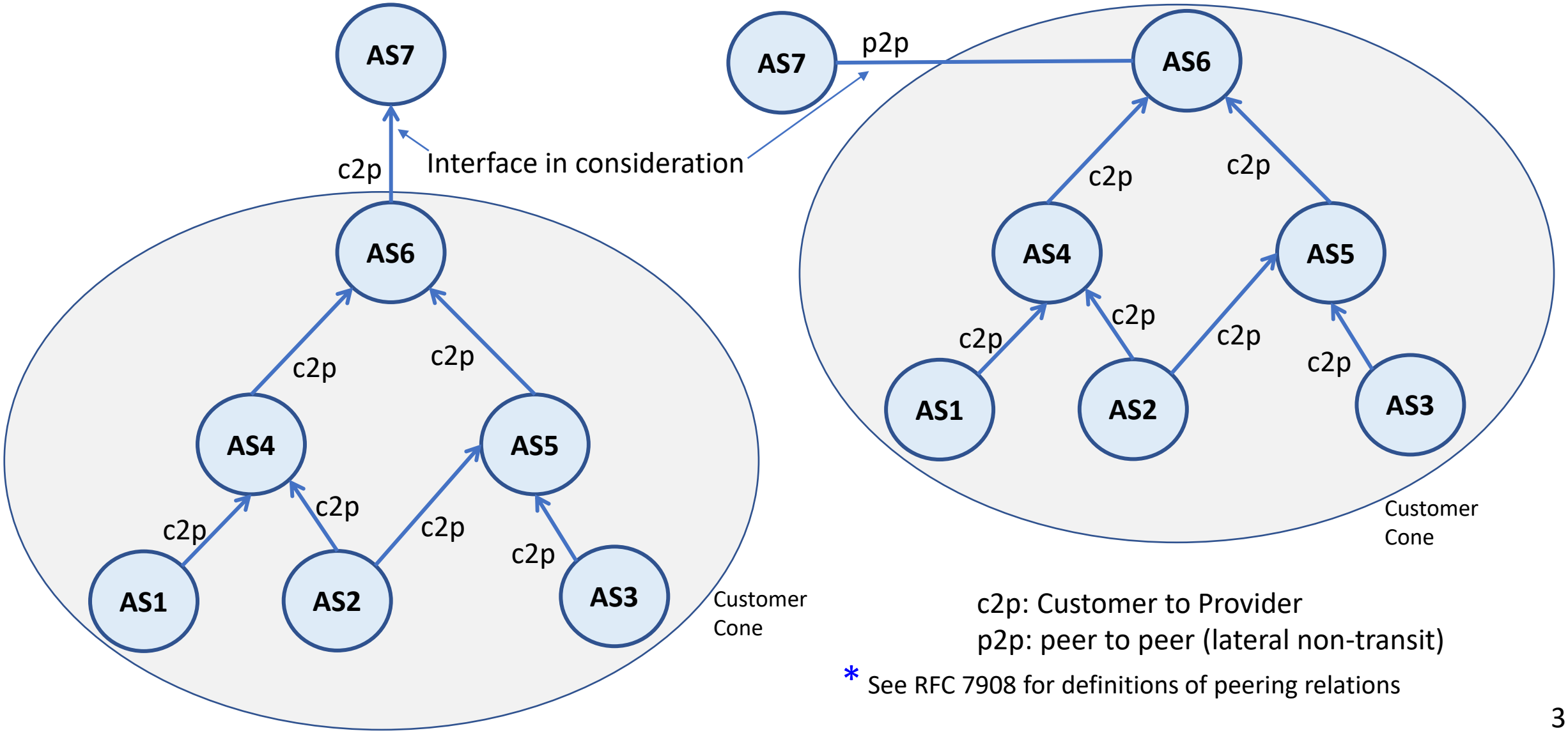
SAVNET Meeting, IETF 121
November 2024

Motivation and Summary

- Much interest seen in the community to improve Source Address Validation (SAV) techniques (e.g., RFC 8704, IETF SAVNET WG)
- There are attempts to further improve upon EFP-uRPF [RFC 8704]
- Proposed new BAR-SAV method makes complementary use of BGP UPDATEs, ASPAs, and ROAs
- BAR-SAV advances the technology for SAV filter design
  - ✓ Significantly improves the ability to detect hidden prefixes
  - ✓ Provides a solution to the CDN/Direct Server Return (DSR) problem
- No changes to protocol on the wire
- Offers immediate benefits to early adopters

Goal: Construct Permissible Ingress Prefix List for SAV (at AS7)

The methodology is the same for a Customer or Lateral (i.e., non-transit) Peer* Interface

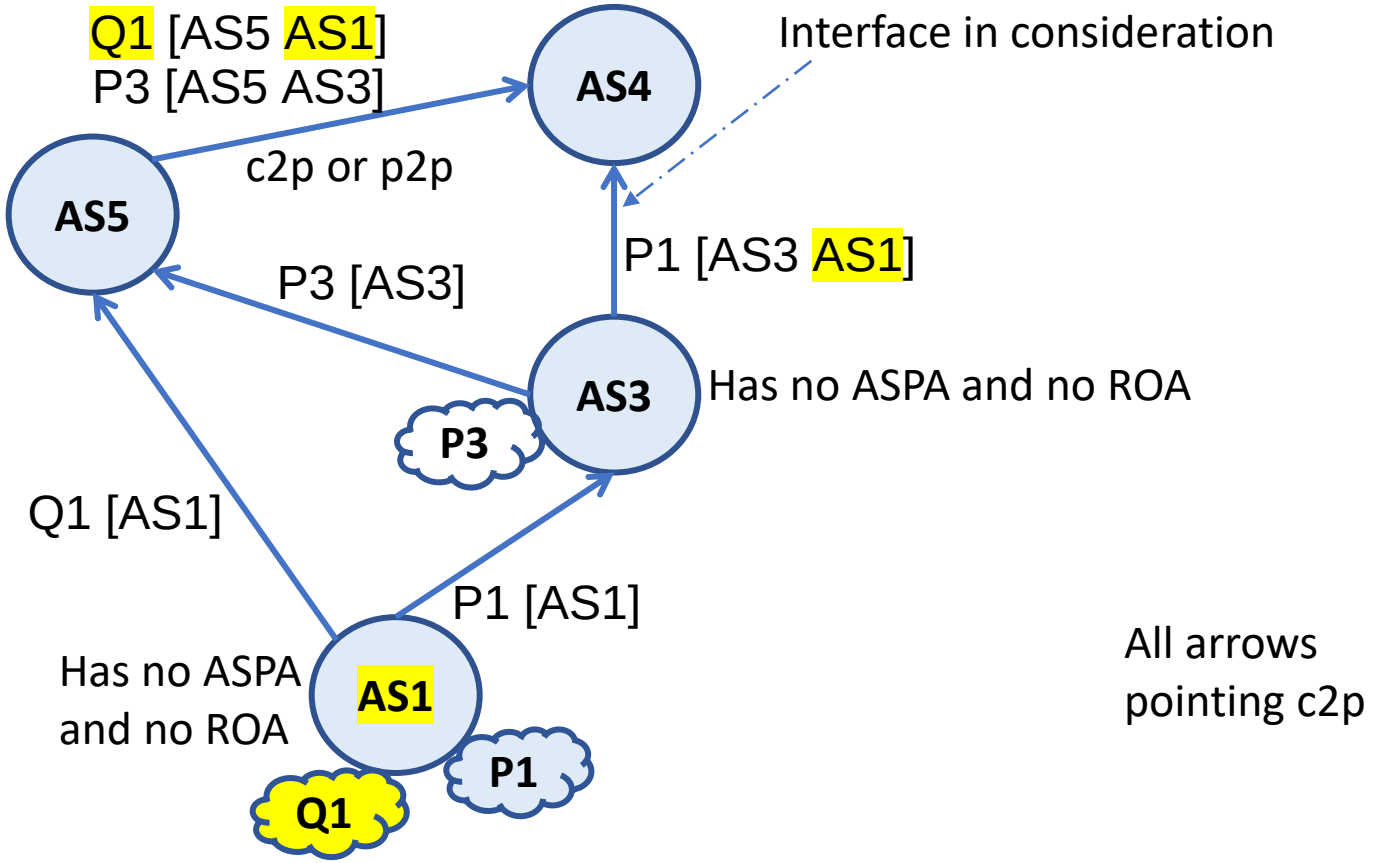


c2p: Customer to Provider
p2p: peer to peer (lateral non-transit)

* See RFC 7908 for definitions of peering relations

EFP-uRPF [RFC 8704] in a Nutshell

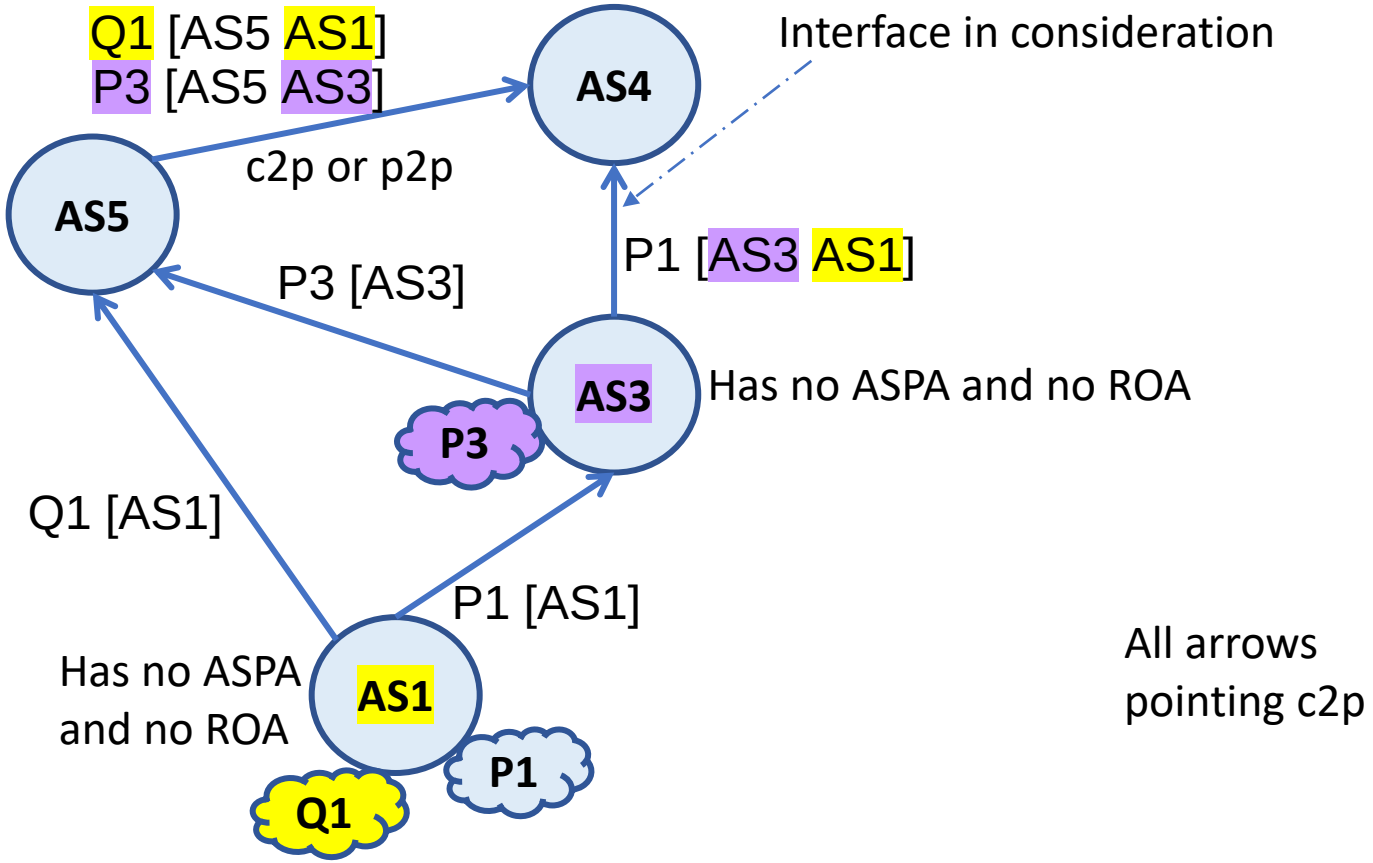
- Prefixes P1 and Q1 are captured by Alg. A of RFC 8704
- P3 is not captured
- EFP-uRPF considered only the origin AS in AS_PATHs
- Did not consider the ASes in the middle to capture more hidden prefixes



EFP-uRPF = Enhanced Feasible Path uRPF

Refined Version of Algorithm A of EFP-uRPF [RFC 8704] Incorporated into BAR-SAV

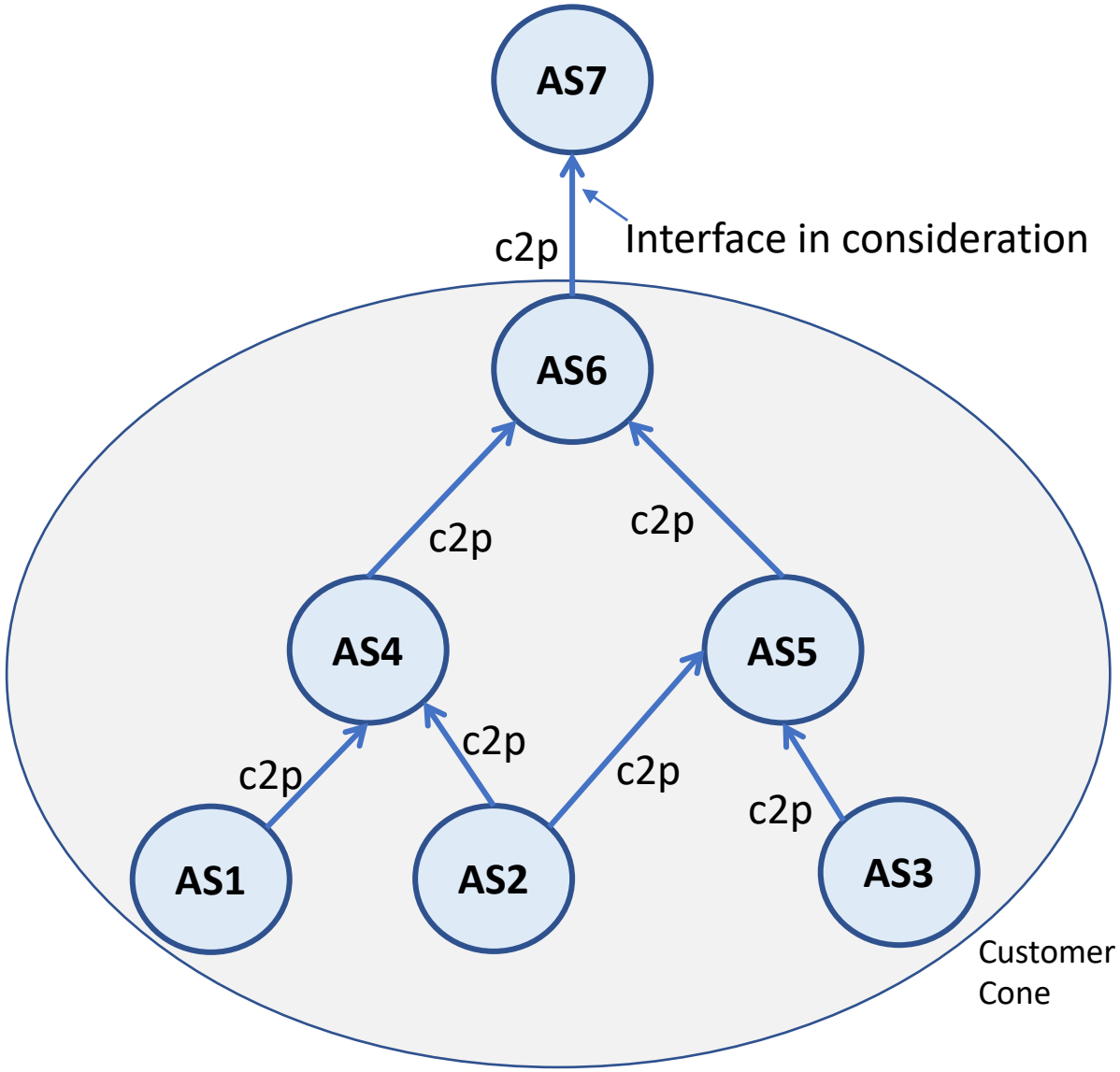
- BAR-SAV makes more efficient use of AS_PATHs
- Considers ASes in the middle of the path (e.g., AS3) as well, not just the origin ASes
- Unlike EFP-uRPF, BAR-SAV captures P3 also



- Much better detection of "Hidden" prefixes in multihoming scenarios by BAR-SAV

SAV Using ASPA, ROA, and BGP UPDATE (BAR-SAV)

Construction of Permissible Ingress Prefix List for SAV (at AS7)



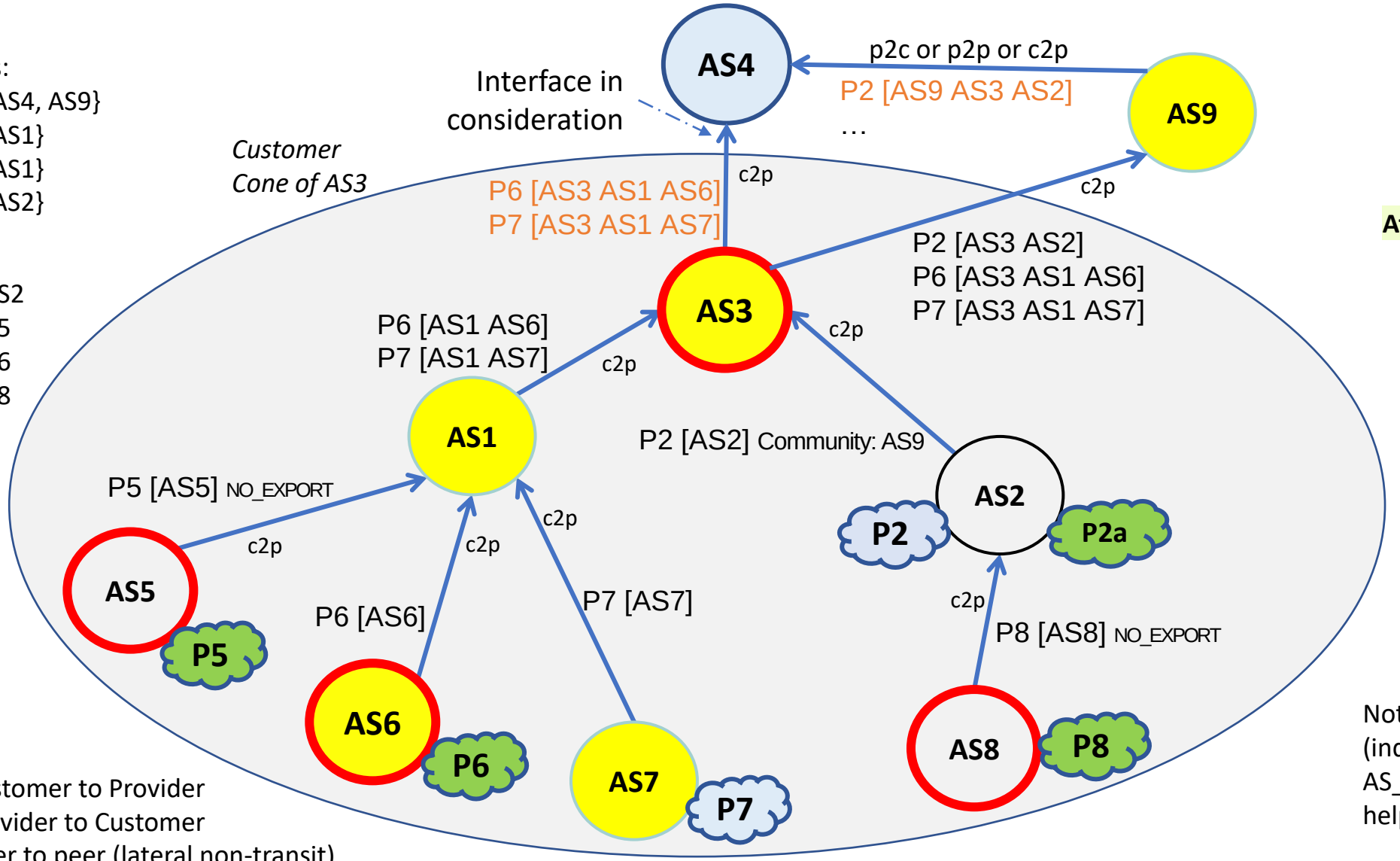
Applicable in the period when ASPA and ROA adoption is not ubiquitous

A. Obtain the set of ASNs in the Customer's customer cone (CC) using ASPAs and AS_PATHs
B. Gather all prefixes in ROAs associated with the ASNs found in Step A.
C. Gather all prefixes in BGP UPDATE messages with originating ASN among ASNs found in Step A.
D. Combine sets found in Steps B and C. Keep only the unique prefixes. This is the permissible prefix list for SAV for the interface in consideration.

How BAR-SAV Works
Finding **All** ASes and Prefixes in Customer's (or Peer's) Customer Cone Using BGP Announcements (as seen at AS4), ASPA, and ROA

ASPAs:
AS3 {AS4, AS9}
AS5 {AS1}
AS6 {AS1}
AS8 {AS2}

ROAs:
P2a AS2
P5 AS5
P6 AS6
P8 AS8



Has ROA Pn

No ROA Pn

At AS4's interface with AS3:

Visible in ASPA and AS_PATH — ASn

Visible only in AS_PATH — ASn

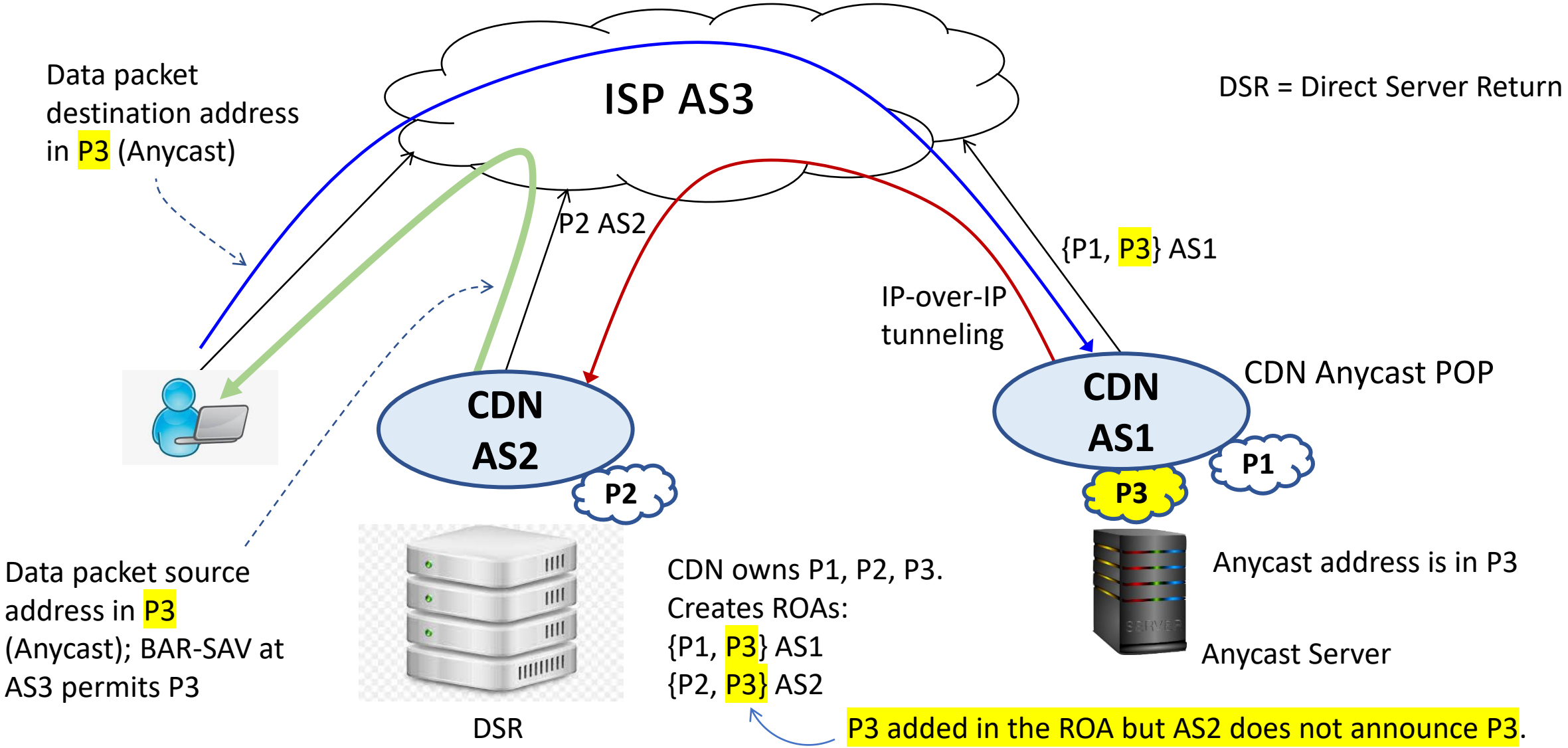
Visible only in ASPA — ASn

Not visible in ASPA (indirectly visible in AS_PATH; BAR-SAV helps find it) — ASn

c2p: Customer to Provider
p2c: Provider to Customer
p2p: peer to peer (lateral non-transit)

Content Delivery Network (CDN) Application

Example of how the BAR-SAV method solves the DSR blocking problem



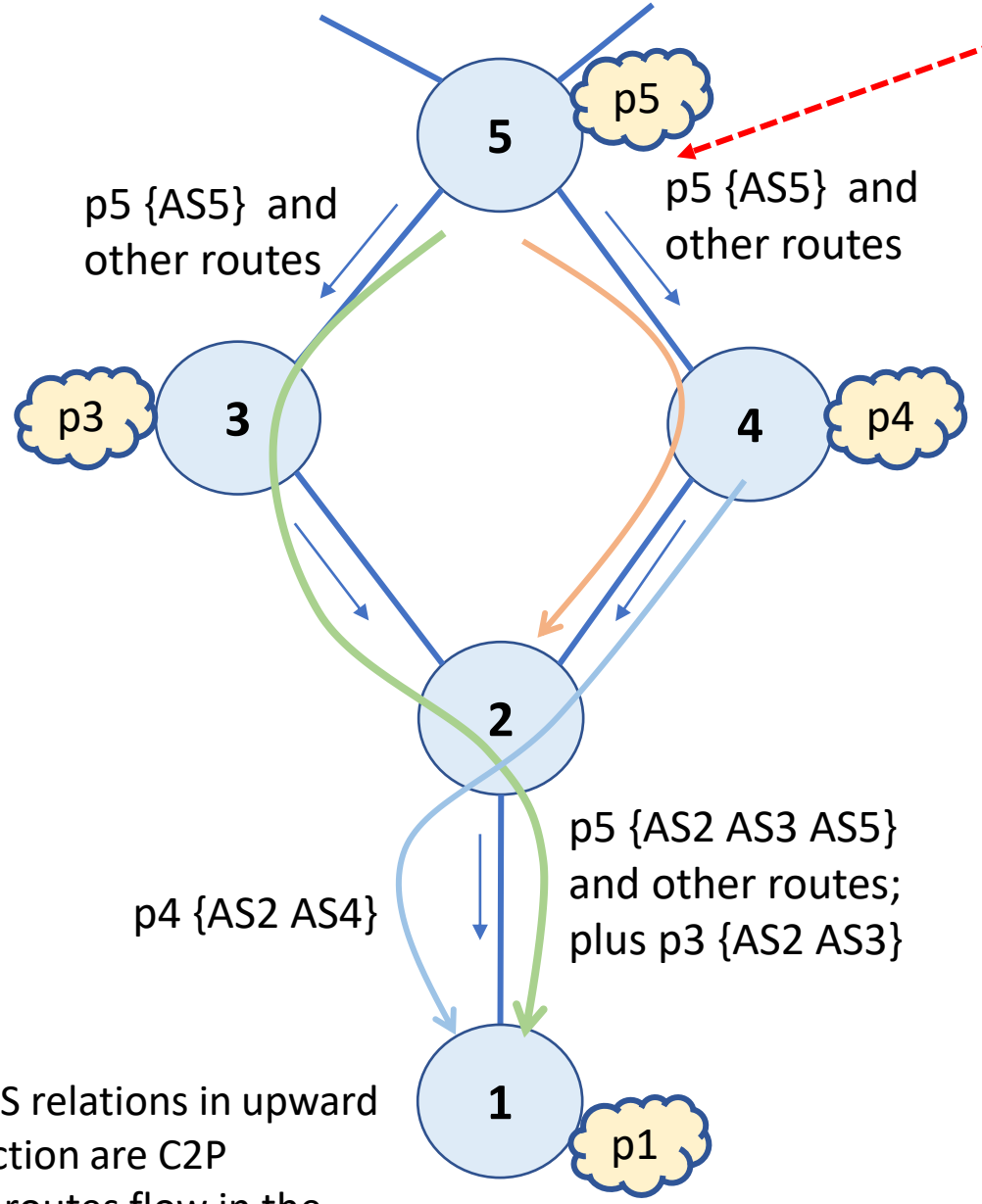
NO_EXPORT and DSR Scenarios

- BAR-SAV utilizes ASPA to uncover hidden NO_EXPORT prefixes
- It utilizes ROA to uncover hidden DSR (Direct Server Return) prefixes
- Strongly recommend network operators to
  - ➢ Register ASPA in case of origination of routes with NO_EXPORT
    - And/or advertise a covering less-specific prefix also without NO_EXPORT to the same providers
  - ➢ Register ROA in case the unannounced DSR anycast prefix scenario applies to their AS

Best Practices Considering RPKI Propagation Delay

- ROA is newly registered for a prefix already announced
  - ROV status change from Unknown to Valid has no impact on BAR-SAV
- ROA is newly registered for a newly acquired prefix
  - Prefix announced before the ROA propagates through RPKI system
  - ROV status change from Unknown to Valid has no impact on BAR-SAV
- Existing prefix with ROA is split and more specific prefixes under it are announced
  - New ROA is propagated
  - BCP recommendation to operator is to use make-before-break principle
  - E.g., withdraw the less-specific prefix and old ROA only after the new ROA has propagated
- AS has a new transit provider and ASPA is updated
  - Again, use the make-before-break principle
  - E.g., continue to announce routes via old transit provider until new ASPA has propagated

Fork and Merge Scenario: BAR-SAV works fine



- All AS relations in upward direction are C2P
- BGP routes flow in the direction of arrows

Issue with SAV-specific data proposal

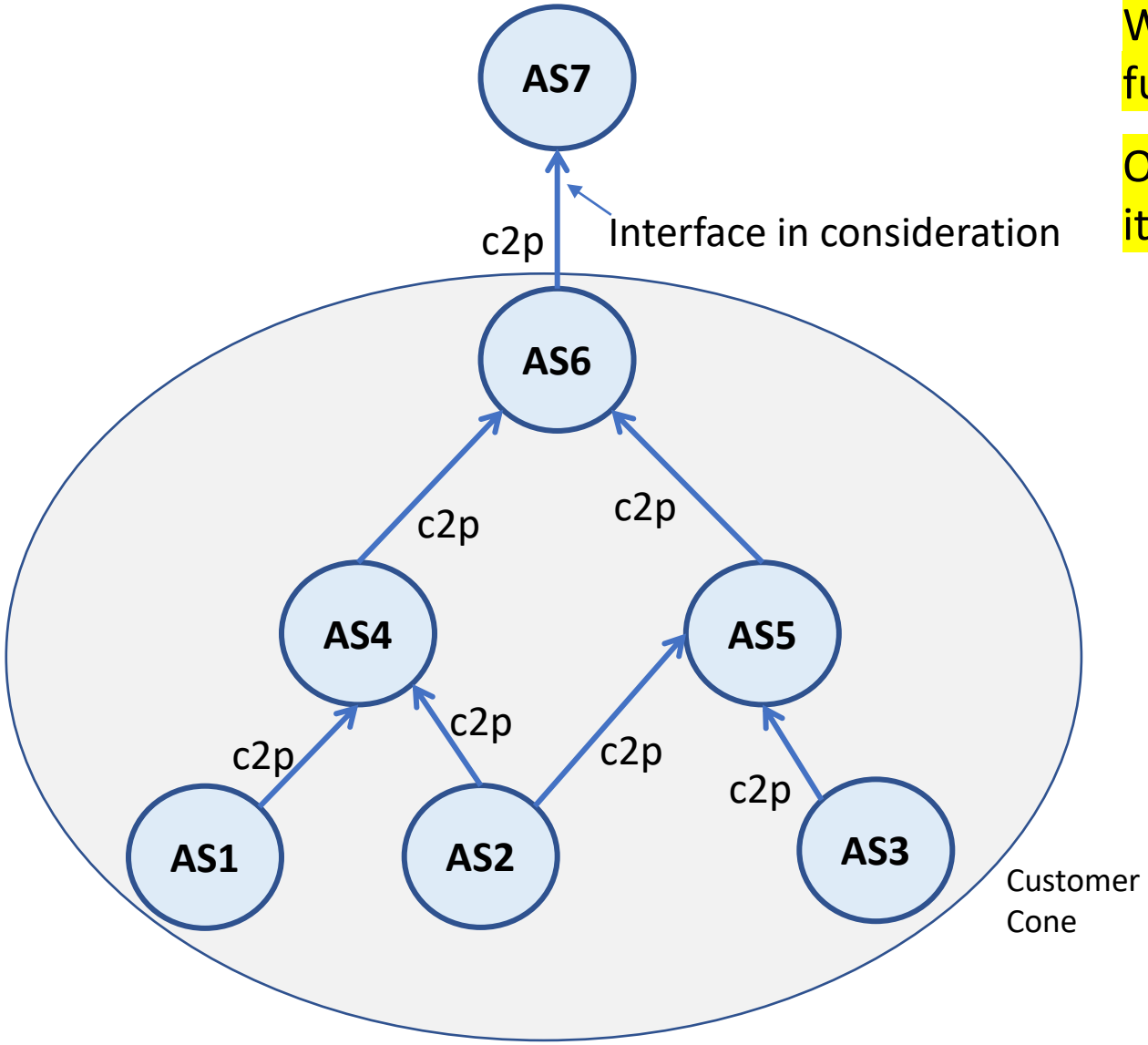
- Consider AS5's SAV table towards AS4
- AS1 and AS5 are participating in SAV-specific messaging
- AS5 sends the same routes to AS3 and AS4
- AS2 prioritizes routes via AS3 over routes via AS4 in best path selection
- So, the hop AS5 to AS4 is not present in any route received at AS1 from AS2
- AS1's SAV-specific messaging informs AS5 to expect SA in p1 on the interface with AS3
- A connectivity loss between AS2 and AS3, or a policy change at AS2, causes AS2 to prioritize AS4 over AS3 in best path selection
- All packets with SA in p1 will be improperly blocked at AS5 on the interface with AS4, until a BGP UPDATE from AS2 is received by AS1, and the corresponding SAV-specific messaging from AS1 is received by AS5
- *In fact, the SAV-specific messaging from AS1 may itself be blocked at AS5 on the interface with AS4*

Thank you

Backup slides

SAV Using Only ASPA and ROA (Procedure X)

Construction of Permissible Ingress Prefix List for SAV (at AS7)



When ASPA and ROA adoption is ubiquitous (in the future)

Or an ISP may use Procedure X on customer interfaces if it requires all its customers to register ROAs and ASPAs

A. Obtain the set of ASNs in the Customer's customer cone (CC) using ASPAs
B. Gather all prefixes in ROAs associated with the ASNs found in Step A. Keep only the unique prefixes.
C. The set computed in Step B is the permissible prefix list for SAV for the interface in consideration.

But there will be…

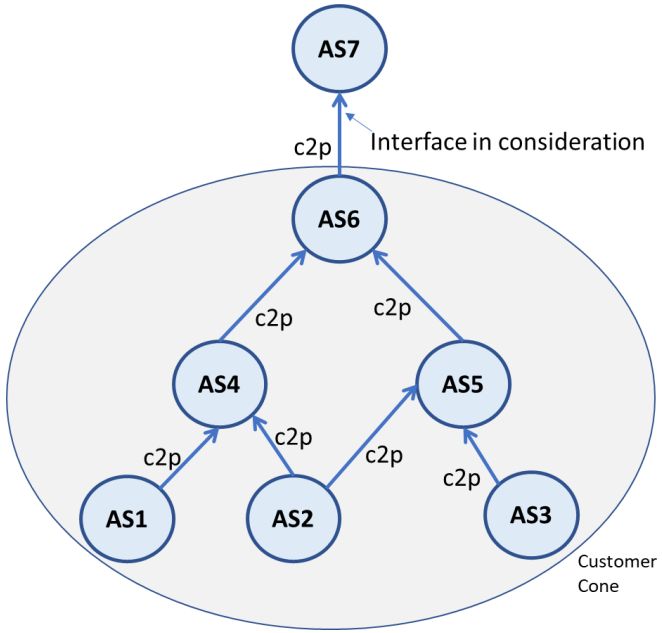
Partial deployment of ROAs and ASPAs for some time

- During that period…
  - ✓ BAR-SAV compensates
  - ✓ Makes complementary use of BGP UPDATEs, ASPA, and ROA
    - Incorporates a refined version of EFP-uRPF*

* Enhanced Feasible Path uRPF (EFP-uRPF) [RFC 8704]

A Note on Customer Cone Computation

- One should *not* compute a customer cone by separately processing ASPA data and AS_PATH data and then merging the two sets of ASes at the end. Doing so is likely to miss ASes from the customer cone.



- Instead, both ASPAs and AS_PATHs should be used to iteratively expand the discovered customer cone. When new ASes are discovered, both ASPA and AS_PATH data should be used to discover customers of those ASes. This process is repeated for newly discovered customer ASes until there are no new ASes to be found.

The next 3 slides illustrate the details of how BAR-SAV works

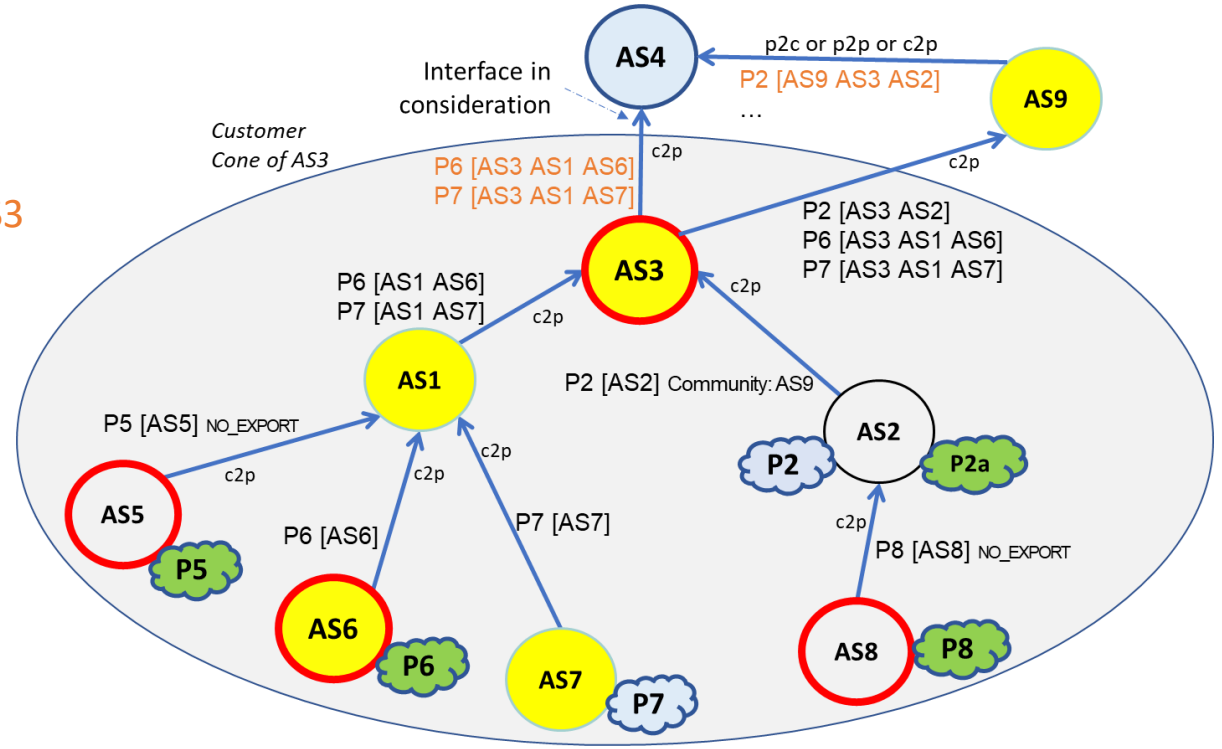
Finding All ASes in the CC using BGP AS_PATH and ASPA

INPUTS

ASPAs:
AS3 {AS4, AS9}
AS5 {AS1}
AS6 {AS1}
AS8 {AS2}

ROAs:
P2a AS2
P5 AS5
P6 AS6
P8 AS8

BGP UPDATE AS_PATHs:
Interface in Consideration: AS3
P6 [AS3 AS1 AS6]
P7 [AS3 AS1 AS7]
Other Interfaces:
P2 [AS9 AS3 AS2]



OUTPUT

Iteration	Customer Cone	New ASes from ASPA	New ASes from AS_PATH
1	AS3	None	P6 [AS3 AS1 AS6] → AS1 P7 [AS3 AS1 AS7] → AS1 P2 [AS9 AS3 AS2] → AS2
2	AS3, AS1, AS2	AS5 {AS1} → AS5 AS6 {AS1} → AS6 AS8 {AS2} → AS8	P6 [AS3 AS1 AS6] → AS6 P7 [AS3 AS1 AS7] → AS7
3	AS3, AS1, AS2, AS5, AS6, AS8, AS7	None	None

Finding All Prefixes in the CC using BGP Routes and ROA

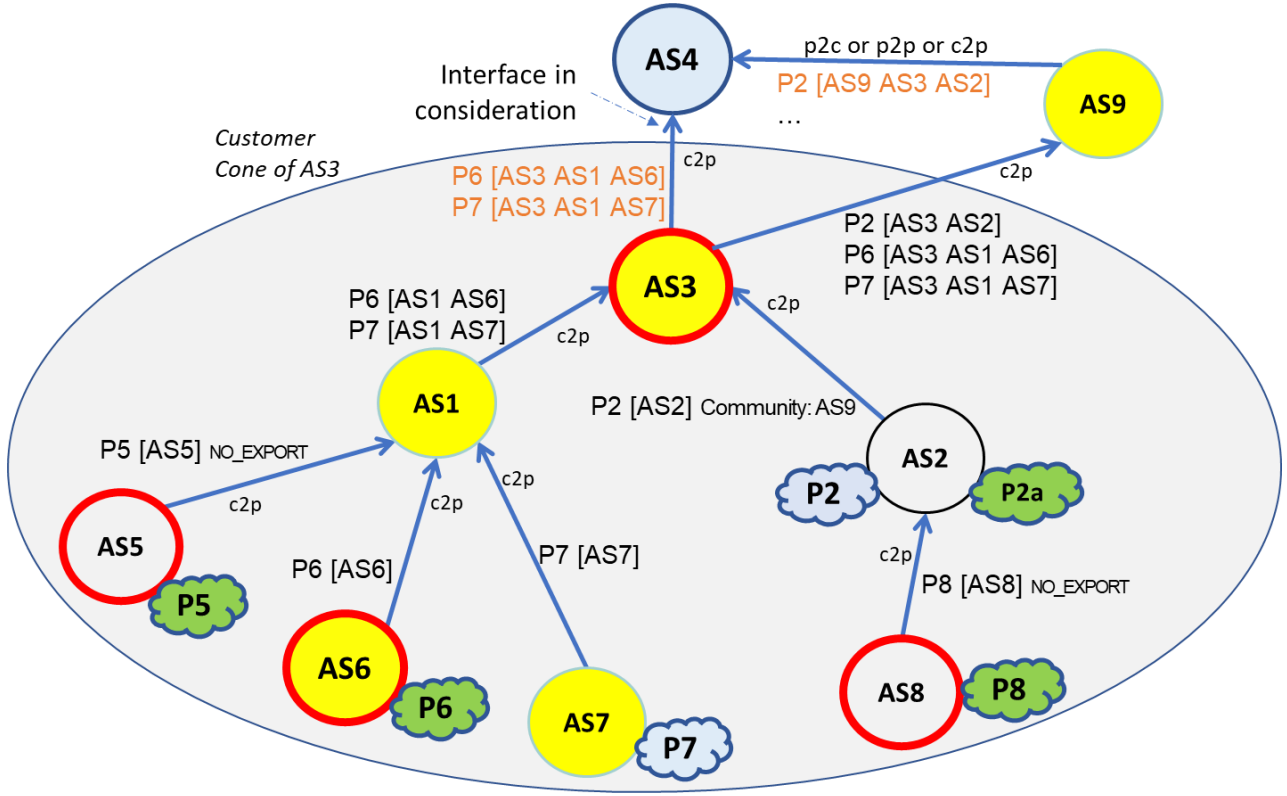
INPUTS

ASPAs:
AS3 {AS4}
AS3 {AS9}
AS5 {AS1}
AS6 {AS1}
AS8 {AS2}

ROAs:
P2a AS2
P5 AS5
P6 AS6
P8 AS8

BGP UPDATE AS_PATHs:
Interface in Consideration: AS3
P6 [AS3 AS1 AS6]
P7 [AS3 AS1 AS7]
Other Interfaces:
P2 [AS9 AS3 AS2]

Customer Cone
AS1, AS2, AS3, AS5, AS6, AS7, AS8



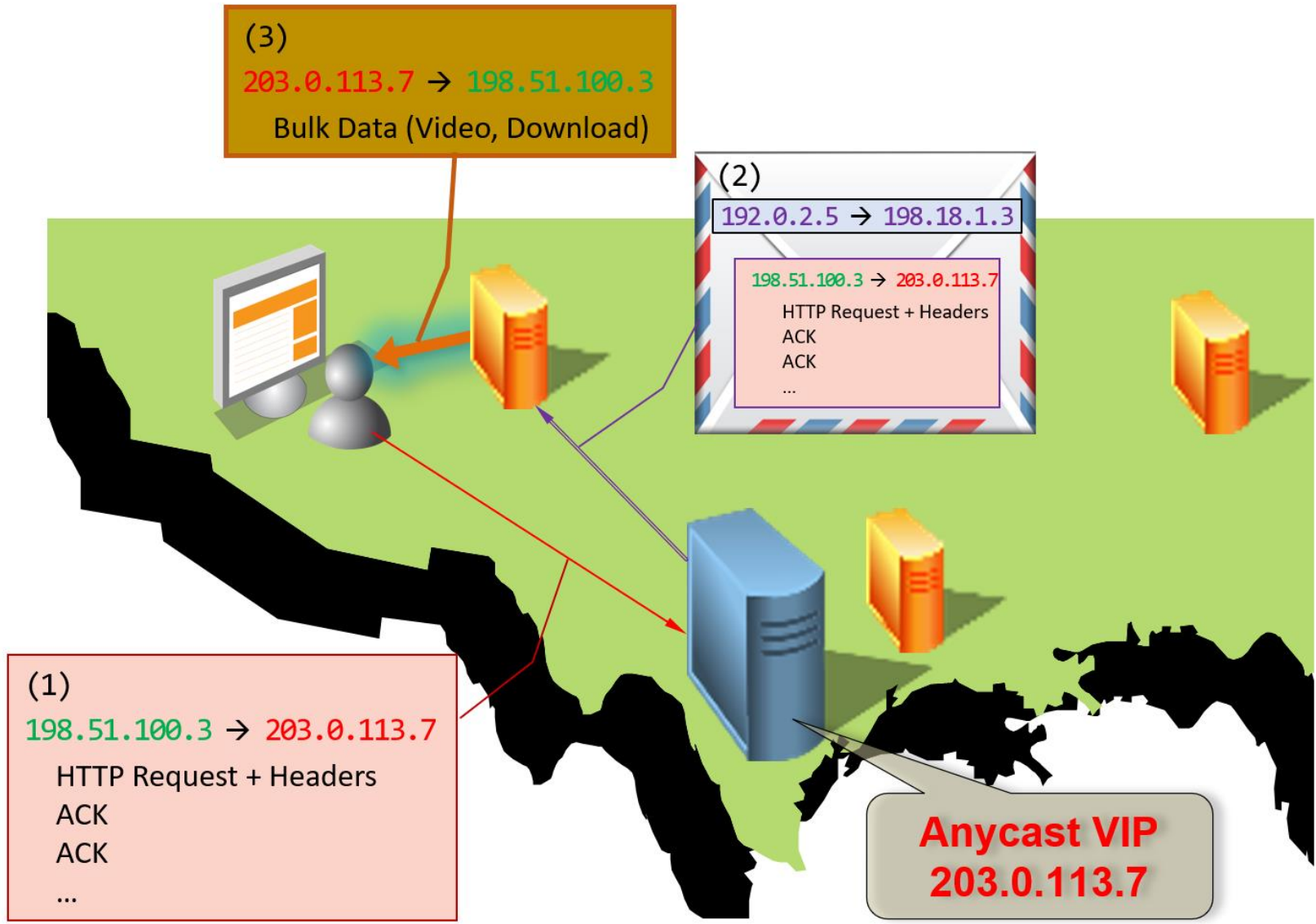
OUTPUT

ASN	Prefixes from ROA	Prefixes from BGP
AS1		
AS2	(P2a AS2) → P2a	P2 [AS9 AS3 AS2] → P2
AS3		
AS5	(P5 AS5) → P5	
AS6	(P6 AS6) → P6	P6 [AS3 AS1 AS6] → P6
AS7		P7 [AS3 AS1 AS7] → P7
AS8	(P8 AS8) → P8	

SAV Prefixes
P2, P2a, P5, P6, P7, P8

Anycast/Edge Hybrid – Direct Server Return

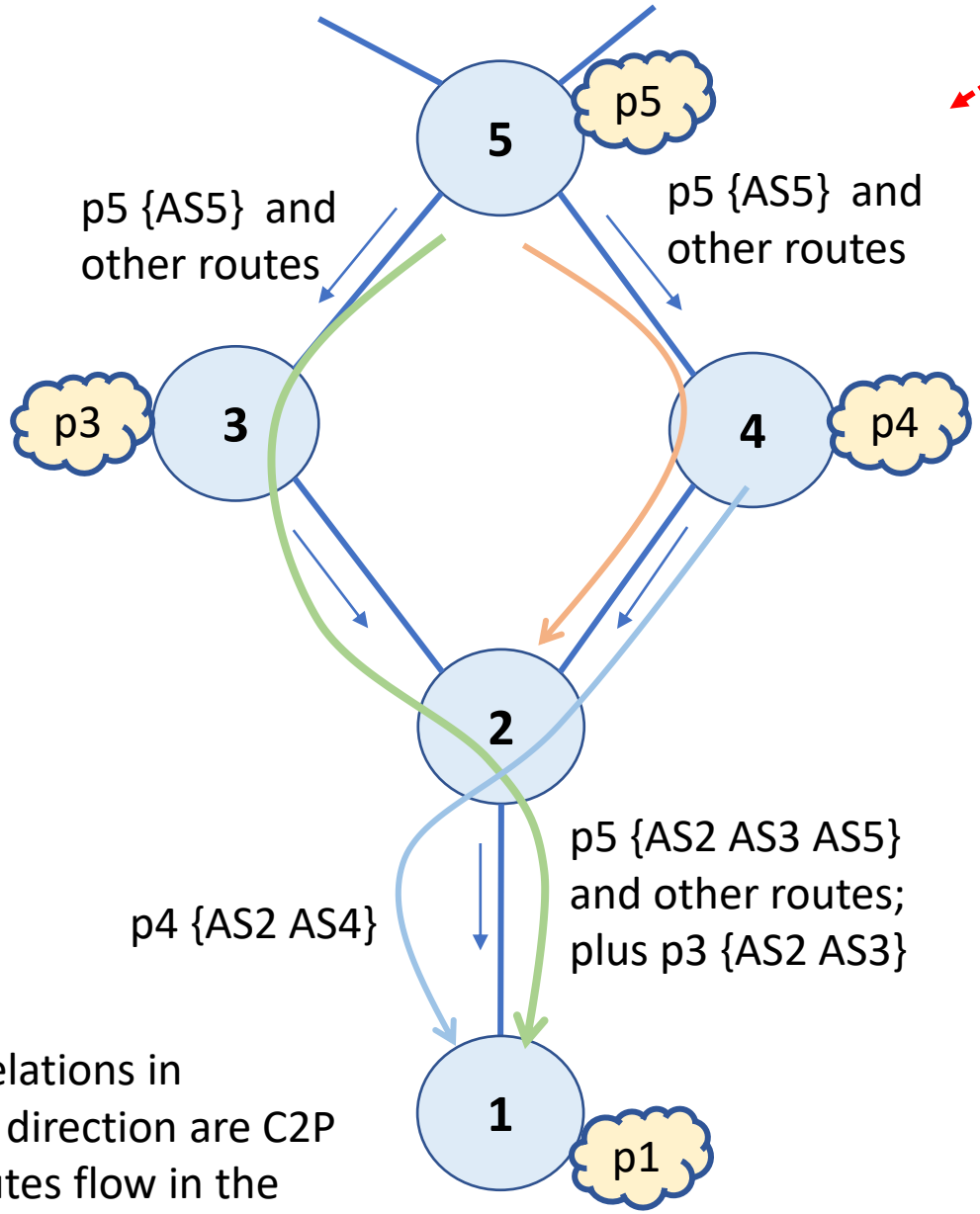
1. Anycast POPs lookup “best” edge POP for each new connection (using the actual user IP)
2. Anycast POPs tunnel packets to edge POPs
3. Edge servers send data to users directly – Direct Server Return (DSR)



Detailed Description of the BAR-SAV Procedure

1. Let the Customer or Lateral Peer ASN be denoted as AS-k.
2. Let i = 1. Initialize: AS-set Z(1) = {AS-k}.
3. Increment i to i+1.
4. Create AS-set A(i) of all ASNs whose ASPA data declares at least one ASN in AS-set Z(i-1) as a Provider.
5. Create AS-set B(i) of all customer ASNs each of which is a customer of at least one ASN in AS-set Z(i-1) according to unique AS_PATHs in Adj-RIBs-In of all interfaces at the BGP speaker computing the SAV filter.
6. Form the union of AS-sets A(i) and B(i) and call it AS-set C. From AS-set C, remove any ASNs that are present in Z(j), for j=1 to j=(i-1). Call the resulting set Z(i).
7. If AS-set Z(i) is null, then set i_max = i - 1 and go to Step 8. Else, go to Step 3.
8. Form the union of the AS-sets, Z(i), i = 1, 2, ..., i_max, and name this union as AS-set D.
9. Select all ROAs in which the authorized origin ASN is in AS-set D. Form the union of the sets of prefixes listed in the selected ROAs. Name this union set of prefixes as Prefix-set P1.
10. Using the routes in Adj-RIBs-In of all interfaces, create a list of all prefixes originated by any ASN in AS-set D. Name this set of prefixes as Prefix-set P2.
11. Form the union of Prefix-sets P1 and P2. Apply this union set as the list of permissible prefixes for SAV.

Fork and Merge Scenario: BAR-SAV works fine



- All AS relations in upward direction are C2P
- BGP routes flow in the direction of arrows

Issue with SAV-specific data proposal

- Consider AS5's SAV table towards AS4
- AS1 and AS5 are participating in SAV-specific messaging
- AS5 sends the same routes to AS3 and AS4
- AS2 prioritizes routes via AS3 over routes via AS4 in best path selection
- So, the hop AS5 to AS4 is not present in any route received at AS1 from AS2
- Thus, AS5 to AS4 interface is hidden to AS1
- AS1's SAV-specific messaging is unable to inform AS5 to expect SA in p1 on the interface with AS4
- AS1 has no knowledge about how AS2 chooses to route data packets
- AS2 may route data packets from AS1 with SA in p1 towards AS5 via AS4
- These packets will be improperly blocked at AS5 on the interface with AS4