Bicone Source Address Validation

Dan Li, **Lancheng Qin**, Li Chen, Libin Liu

July 23, 2024

Background

- Bicone SAV aims to enhance the SAV robustness by using blocklist filters generated based on BGP updates, ROAs, and ASPAs related to the provider cone
  - It helps avoid improper block (i.e., blocking legitimate traffic) in the case of limited propagation of prefixes in the customer cone[1,2,3]
    - Using an allowlist in this case may have improper block problems
  - It can work together with existing SAV solutions (e.g., EFP-uRPF and BAR-SAV) that generate allowlist filters
- Historical versions
  - draft-li-sidrops-bicone-sav-00, IETF 119 SIDROPS WG
  - draft-li-sidrops-bicone-sav-01, June 14, 2024
  - draft-li-sidrops-bicone-sav-02, IETF 120 SIDROPS WG

[1] Figure 4 in RFC 8704
[2] Figure 3 in draft-ietf-savnet-inter-domain-problem-statement-04
[3] Section 5 in draft-ietf-sidrops-bar-sav-03

Comments Received in IETF 119

- This document should consider dealing with the overlap between provider cone and customer cone [from Ben Maddison]
  - Response: We add a new Section 6.3 to analyze and address this issue
- Using both blocklist filter and allowlist filter on an interface takes up a lot of storage space [from Nan Geng and Ben Maddison] & The allowlist filter is not really used [from K. Sriram]
  - Response: The new version suggests to use one SAV filter on an interface. An AS can choose to use either an allowlist or a blocklist on interfaces facing a customer AS or lateral peer AS according to recommendations in the new Section 7
- The blocklist filter generated by Bicone SAV has improper admits [from K. Sriram]
  - Response: The blocklist has more improper admits than the allowlist but still maintains cone-level directionality. Therefore, it is recommended to use blocklist when using allowlist may have improper block problems. We add a new Section 4 to introduce the goals of Bicone SAV

Main Updates Compared to Version-00

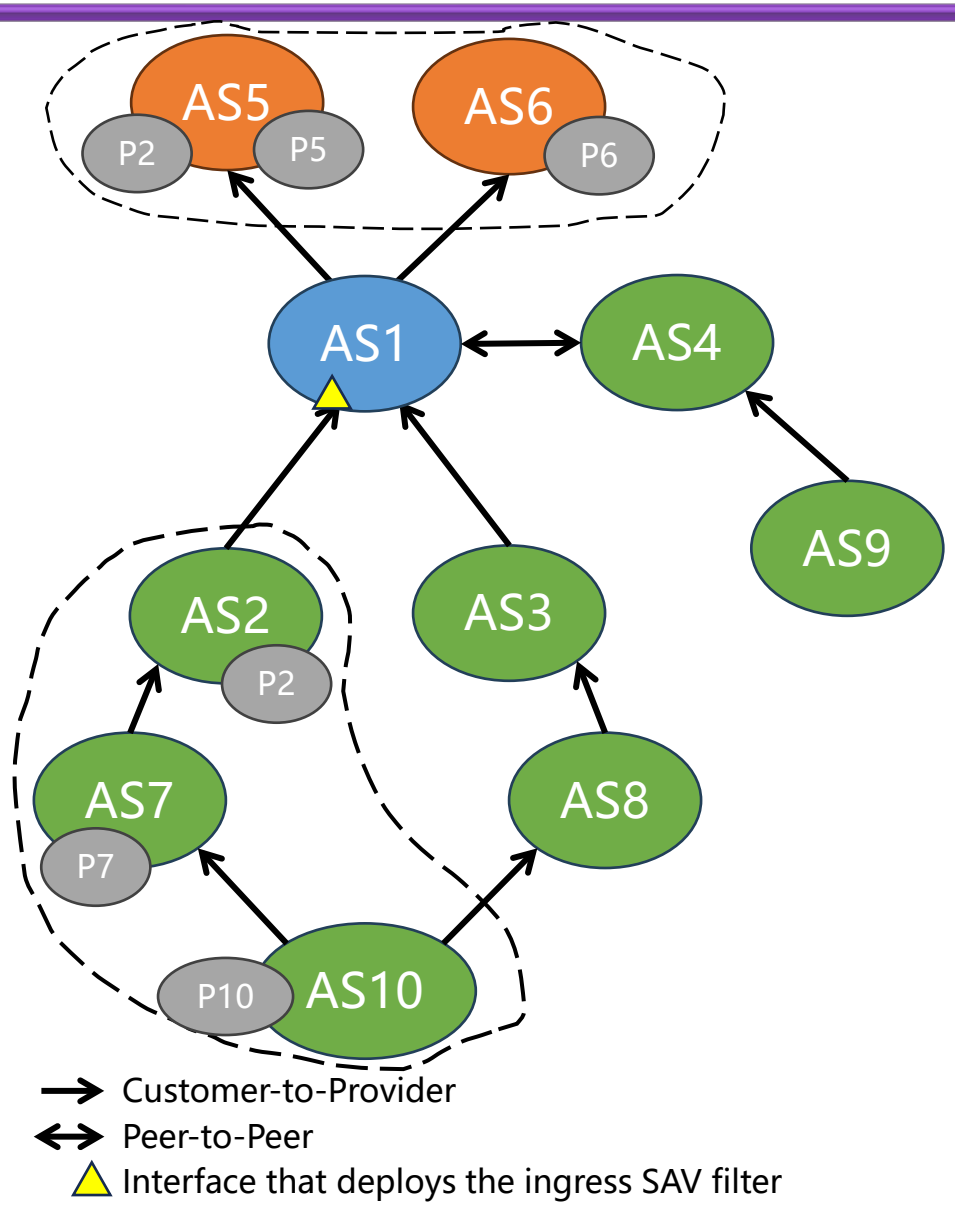
- Add a new Section 4 to introduce two goals of Bicone SAV
- Add a new Section 5 to introduce some existing methods that can generate allowlist SAV filters
- In Section 6.3, we describe how to deal with the overlap between provider cone and customer cone
- Add a new Section 7 to provide a summary of recommendations

Goals of Bicone SAV

Bicone SAV aims to perform more robust ingress SAV filtering at interfaces facing a customer AS or a lateral peer AS by flexibly using allowlist and blocklist filters

- Goal #1: Avoid improper block
  - If using an allowlist at an interface may have improper block problems, Bicone SAV recommends using a blocklist at this interface
    - The blocklist helps avoid improper block
- Goal #2: Maintain directionality
  - When using an allowlist, the SAV filter can block data packets using source addresses not belonging to the customer's or lateral peer's customer cone
  - When using a blocklist, the SAV filter can block data packets using source addresses belonging to the provider cone (i.e., cone-level directionality)
    - The blocklist has more improper admits than the allowlist but much less improper admits than loose uRPF
      - **The blocklist performs better in achieving Goal #1**
      - **The allowlist performs better in achieving Goal #2**

Prefixes in Allowlist & Blocklist



- The allowlist on AS1-AS2 interface must contain all prefixes belonging to the customer cone of AS2
  - If a prefix is limited propagated (and some ASes do not have ASPAs) in the customer cone, the allowlist may miss this prefix, leading to improper block
- The blocklist on AS1-AS2 interface should contain as many prefixes belonging to the provider cone of AS1 as possible
  - It must remove prefixes that also belong to the customer cone (e.g., prefix P2)
    - These prefixes can be identified by computing the overlap between blocklist and allowlist

Recommendations

Use an allowlist or a blocklist on different interfaces facing a customer AS or a lateral peer AS

- If the network operator ensures the generated allowlist covers all prefixes in a customer's or lateral peer's customer cone, it is recommended to use an allowlist filter
  - For example, when the customer cone is relatively small or ASes in the customer cone have registered ASPAs (if using BAR-SAV)
- If the network operator cannot ensure the generated allowlist covers all prefixes in a customer's or lateral peer's customer cone, it is recommended to use a blocklist filter by default to avoid potential improper block
  - For example, when the customer cone is large or some ASes in the customer cone have not registered ASPAs

Summary

- Bicone SAV provides an additional option for network operators to deploy ingress SAV filter on interfaces facing a customer AS or a lateral peer AS
  - Generating a blocklist SAV filter by using BGP updates, ROAs, and ASPAs related to the provider cone
- The blocklist helps avoid improper block and maintains cone-level directionality
  - Blocking data packets from a customer AS or a lateral peer AS using spoofed source addresses belonging to the provider cone
- Network operators can flexibly choose to use the allowlist or the blocklist on different interfaces according to their needs and actual situation
  - Blocklist and allowlist have advantages and disadvantages in different situations

Acknowledgement

- Thank Ben Maddison, K. Sriram, Nan Geng, Shengnan Yue, Siyuan Teng for their thoughtful comments

Next Step

- Collaboration is welcome
- Request WG adoption in SIDROPS

Thanks!

Backup Slides

SAV Solutions that Generate Allowlist or Blocklist

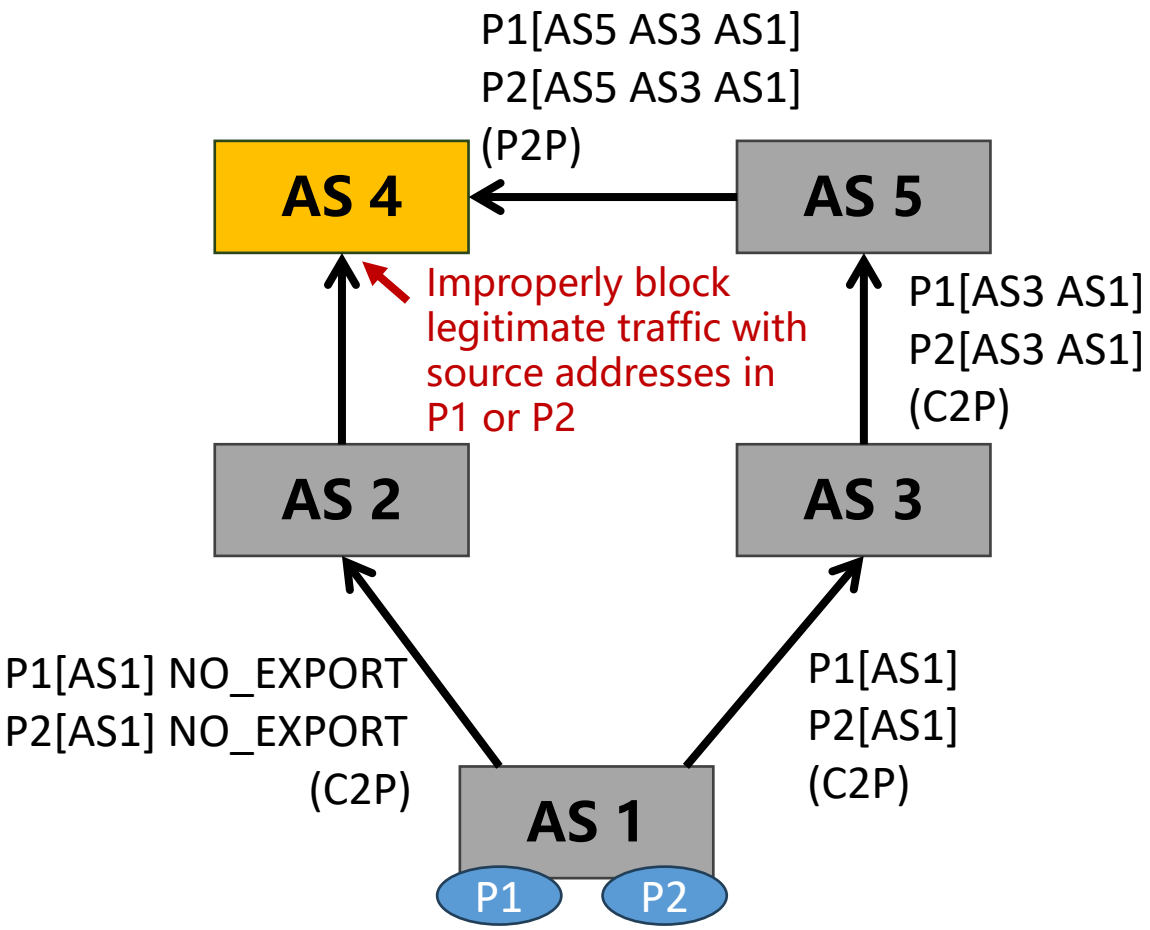
Allowlist-based SAV solutions

- EFP-uRPF [RFC8704] and BAR-SAV [draft-ietf-sidrops-bar-sav]
- Key idea
  - ◆Generate the allowlist by using BGP updates, ROAs, or/and ASPAs related to the **customer cone** of a customer AS or a lateral peer AS
  - ◆The allowlist contains prefixes belonging to that customer cone

Blocklist-based SAV solutions

- Bicone SAV (which is proposed in this document)
- Key idea
  - ◆Generate the blocklist by using BGP updates, ROAs, and ASPAs related to the **provider cone**
  - ◆The blocklist contains prefixes belonging to the provider cone

Improper Block Problem of Solely Using An Allowlist

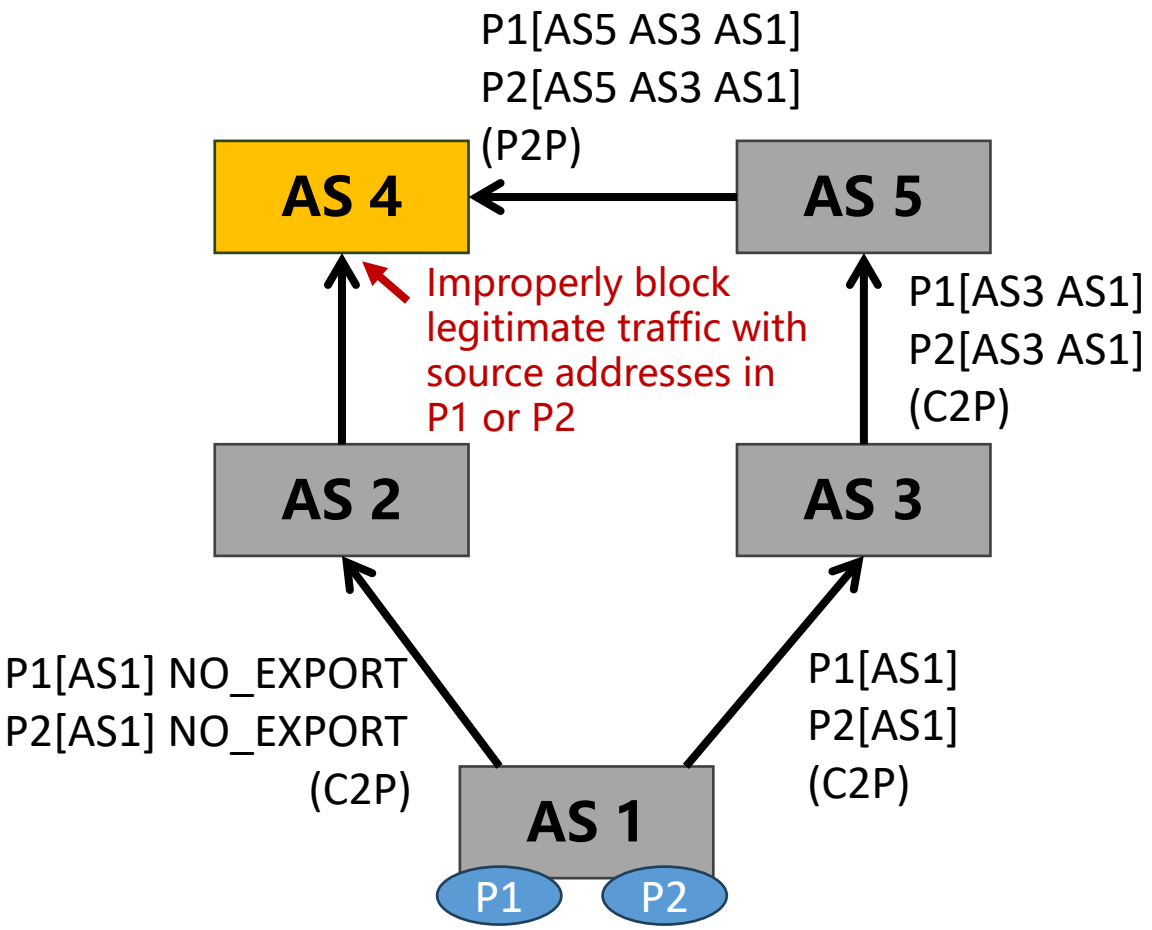


An example of limited propagation of prefixes in the customer cone

Ingress SAV filtering on AS4

- AS1 attaches NO_EXPORT to all prefixes announced to AS2
- AS4 never receives routes for P1 and P2 from its customer AS2
- EFP-uRPF Algorithm A and Algorithm B on AS4 have **improper block problems**
  - ◆Block legitimate data packets received on AS4-AS2 interface with source addresses in P1 or P2

Improper Block Problem of Solely Using An Allowlist



An example of limited propagation of prefixes in the customer cone

Ingress SAV filtering on AS4

- AS1 attaches NO_EXPORT to all prefixes announced to AS2
- AS4 never receives routes for P1 and P2 from its customer AS2
- More recent SAV solutions (e.g., BAR-SAV) additionally use ASPAs and ROAs
  - More robust SAV filtering than EFP-uRPF when the needed ASPAs and ROAs are available
  - **When some ASPAs and ROAs related to the customer cone are missing, improper block still exists**

Ingress SAV Blocklist Filter

Blocklist Generation Procedures

1. Create the set of all directly connected Provider ASNs. Call it AS-set Z(1).
2. Create the set of all unique AS_PATHs in Adj-RIBs-In of all interfaces facing Providers.
3. For each unique AS_PATH with N (N>1) ASNs, i.e., [ASN_{1}, ASN_{2}, ..., ASN_{i}, ASN_{i+1}, ..., ASN_{N}] where ASN_{i} is the ith ASN in AS_PATH and the first ASN (i.e., ASN_{1}) is a directly connected Provider ASN. If all unique AS_PATHs have been processed, go to Step 8.
4. Let i = N
5. Decrement i to i-1.
6. If ASN_{i} authorizes ASN_{i+1} as a Provider in ASN_{i}'s ASPA, ASNs from ASN_{1} to ASN_{i+1} (i.e., ASN_{1}, ASN_{2}, ..., ASN_{i}, and ASN_{i+1}) are included in AS-set Z(1) and go to Step 3.
7. If i == 1, go to Step 3. Else, go to Step 5.
8. Let k = 1.
9. Increment k to k+1.
10. Create AS-set Z(k) of ASNs that are not in AS-set Z(k-1) but are authorized as Providers in ASPAs of any ASN in AS-set Z(k-1).
11. If AS-set Z(k) is null, then set k_max = k-1 and go to Step 12. Else, form the union of AS-set Z(k) and AS-set Z(k-1) as AS-set Z(k) and go to Step 9.
12. Select all ROAs in which the authorized origin ASN is in AS-set Z(k_max). Form the union of the sets of prefixes in the selected ROAs. Call it Prefix-set P1.
13. Using the routes in Adj-RIBs-In of all interfaces facing Providers, create a set of prefixes originated by any ASN in AS-set Z(k_max). Call it Prefix-set P2.
14. Form the union of Prefix-set P1 and Prefix-set P2. Apply this union set as a blocklist on every interface facing a Customer or Lateral Peer.

- Use BGP UPDATE messages and ASPAs to identify as many as ASes in the provider cone
- Use BGP UPDATE messages and ROAs to identify prefixes belonging to ASes in the provider cone

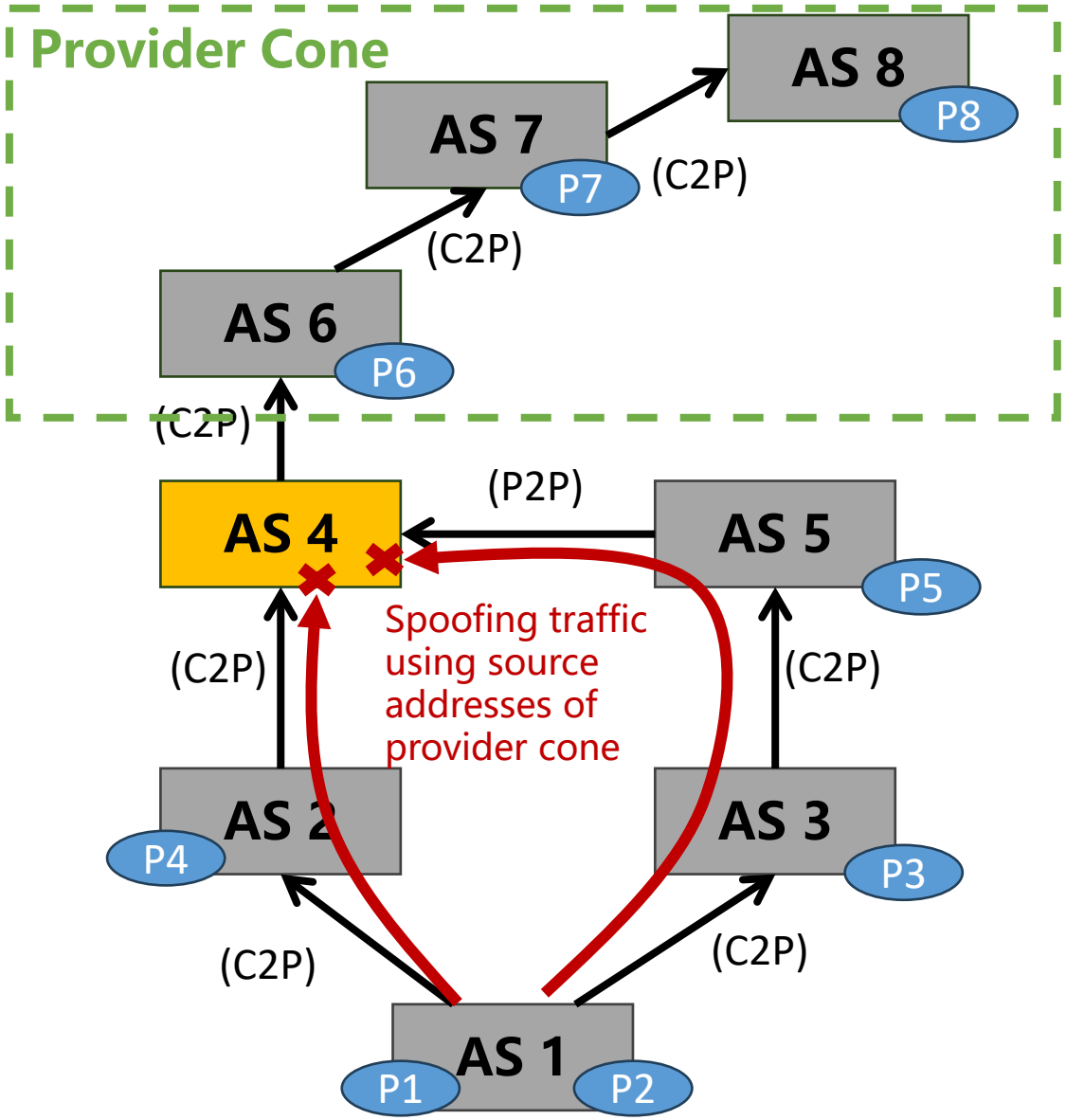
Ingress SAV Blocklist Filter

Blocklist Generation Procedures

1. Create the set of all directly connected Provider ASNs. Call it AS-set Z(1).
2. Create the set of all unique AS_PATHs in Adj-RIBs-In of all interfaces facing Providers.
3. For each unique AS_PATH with N (N>1) ASNs, i.e., [ASN_{1}, ASN_{2}, ..., ASN_{i}, ASN_{i+1}, ..., ASN_{N}] where ASN_{i} is the ith ASN in AS_PATH and the first ASN (i.e., ASN_{1}) is a directly connected Provider ASN. If all unique AS_PATHs have been processed, go to Step 8.
4. Let i = N
5. Decrement i to i-1.
6. If ASN_{i} authorizes ASN_{i+1} as a Provider in ASN_{i}'s ASPA, ASNs from ASN_{1} to ASN_{i+1} (i.e., ASN_{1}, ASN_{2}, ..., ASN_{i}, and ASN_{i+1}) are included in AS-set Z(1) and go to Step 3.
7. If i == 1, go to Step 3. Else, go to Step 5.
8. Let k = 1.
9. Increment k to k+1.
10. Create AS-set Z(k) of ASNs that are not in AS-set Z(k-1) but are authorized as Providers in ASPAs of any ASN in AS-set Z(k-1).
11. If AS-set Z(k) is null, then set k_max = k-1 and go to Step 12. Else, form the union of AS-set Z(k) and AS-set Z(k-1) as AS-set Z(k) and go to Step 9.
12. Select all ROAs in which the authorized origin ASN is in AS-set Z(k_max). Form the union of the sets of prefixes in the selected ROAs. Call it Prefix-set P1.
13. Using the routes in Adj-RIBs-In of all interfaces facing Providers, create a set of prefixes originated by any ASN in AS-set Z(k_max). Call it Prefix-set P2.
14. Form the union of Prefix-set P1 and Prefix-set P2. Apply this union set as a blocklist on every interface facing a Customer or Lateral Peer.

- SAV filtering
  - Block data packets received from customers and lateral peers with source addresses covered in the blocklist
- Immediate incremental benefits
  - If the generated blocklist does not include all prefixes in the provider cone
    - it can still block spoofing traffic while avoiding improper block when some ASes' ASPAs are unavailable

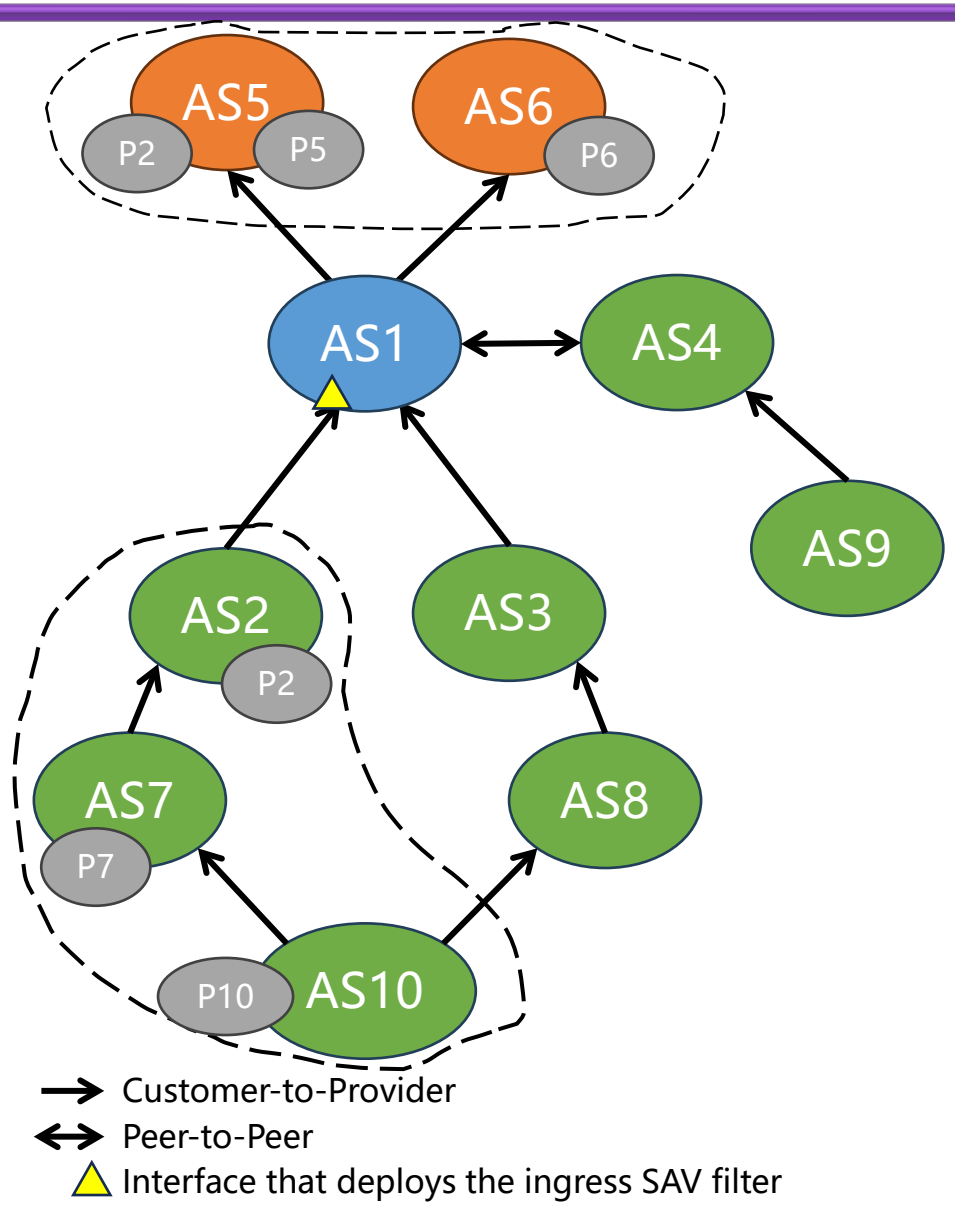
Example: Ingress SAV Blocklist Filter



Ingress SAV filtering on AS4

- Assume the provider cone of AS4 includes AS6, AS7, and AS8
- If AS4 identifies all prefixes (i.e., P6, P7, and P8) in the provider cone
  - ◆Block data packets received from AS2 and AS5 with source addresses in P6, P7 and P8
- If AS4 only identifies partial prefixes (e.g., P6 and P7) in the provider cone
  - ◆Block data packets received from AS2 and AS5 with source addresses in P6 and P7
  - ◆Avoid improper block

Use An Allowlist or A Blocklist?



- If the allowlist on AS1-AS2 interface can cover all prefixes belonging to the customer cone of AS2
  - The allowlist has no improper block and no improper admit
  - The blocklist has no improper block but has improper admits
  - Therefore, the allowlist is preferred in this case
- If the allowlist on AS1-AS2 interface fails to cover all prefixes belonging to the customer cone of AS2 (due to limited propagation of prefixes and lack of ASPAs)
  - The allowlist has improper block but has no improper admit
  - The blocklist has no improper block but has improper admits
  - Therefore, the blocklist is preferred in this case