Intra-domain Source Address Validation (SAVNET) Architecture

draft-li-savnet-intra-domain-architecture-01

D. Li, J. Wu, M. Huang, L. Chen, N. Geng, L. Qin, F. Gao

March 2023
Main Updates

solution
draft-li-savnet-intra-domain-architecture-00

A preliminary solution to automatically generating accurate SAV rules through message interactions between routers

draft-li-savnet-intra-domain-architecture-01

An architecture collecting SAV information from multiple information sources for automated and accurate SAV rule generation

Many thanks to the related comments and suggestions from Alvaro, Igor, Aijun, Joel, Jared, Sriram, ...
SAV Information Source

- Required SAV information for SAV rule Generation
  - All source addresses/prefixes even not advertised by route
  - All real incoming directions and limited non-real incoming directions

- Information sources of SAV mechanisms
  - Configuration (e.g., ACL-based filtering)
  - Routing protocols (e.g., strict uRPF and loose uRPF)
  - SAV protocols (new) (e.g., IGP-extended SAV, BGP-extended SAV)
  - etc.

- SAV protocols
  - Newly defined information source representing the proposed protocols for advertising SAV-related information
  - It can provide complete or additional information (e.g., hidden prefixes or real path discovery) for accurate validation
Intra-domain SAVNET Architecture

- **Source Entity**
  - Advertise SAV-related information

- **Validation Entity**
  - Generate SAV rules and/or conduct validation

- A device can be both the source entity and the validation entity
Source Speaker and Validation Speaker

- **Source/Validation Speaker**
  - An abstracted speaker, which is a union of multiple “speakers” (also information sources)

1. **Configuration Interface**
   - CLI, YANG, FlowSpec, and any protocols for SAV

2. **Routing Protocol Speaker**
   - OSPF, IS-IS, BGP, etc.

3. **SAV protocol Speaker**
   - Can be an extension to the routing protocol speaker

Interact with other Speakers
Communication Channel and Messages

- **Channels**
  - Constructed between configuration interfaces, routing protocol speakers, or SAV protocol speakers

- **Messages**
  - SAV rule, e.g., `<prefix, interfaces, validity state actions>`
  - Route information, such as prefixes, links, or feasible paths
  - Forwarding information, e.g., preferred paths of prefixes
SAV Agent

- **SAV information Manager**
  - Store SAV-related information from the Validation Speaker
  - Raw data with many details

- **SAV Rule Generator**
  - SAV rule: <prefix, interfaces, validity states>
  - Rules stored in SAV Table (draft-huang-savnet-sav-table)

- **Rule conflicts due to multiple information sources**
  - Priorities can be set to these sources

**Example 1:**
- High priority source X: <P1, intf1>
- Low priority source Y: <P1, intf2>
- Output SAV rule: <P1, intf1>

**Example 2:**
- Same priority source X: <P1, intf1>
- Same priority source Y: <P1, intf2>
- Output SAV rule: <P1, [intf1, intf2]>
Connectivity Models

(a) Source Speaker → Validation Speaker → Source Speaker → Validation Speaker

(b) Source Speaker → Validation Speaker → Validation Speaker → Validation Speaker

(c) Source Speaker → Validation Speaker → Source & Validation Speaker → Validation Speaker
Conclusion

- Propose a high-level architecture design
  - Besides configuration and routing protocols, **SAV protocols** are included for improving validation
  - **Suitable to any route-based SAV mechanisms.** The architecture can be used for
    - Analyzing existing SAV mechanisms
    - Improving validation based on existing mechanisms
    - Designing new SAV mechanisms

Architecture

- ACL-based filtering
- **strict uRPF**
- **BGP-extended SAV**
- IS-IS-extended SAV
- **loose uRPF**
- OSPF-extended SAV
Next Steps

- Now a preliminary and incomplete architecture. Just a start.

- Any comments are welcome, and welcome to collaborating in contributing to the draft.

- According to the status of the problem statement draft, update the draft and provide the first complete design of the architecture.
Thanks!