
Export of Source Address Validation (SAV) Information in IPFIX

Presented by Qian CAO

Joint work by: Mingqing HUANG¹, Benoît CLAISE², Tianran ZHOU³

¹: Zhongguancun Laboratory, ²: Everything OPS, ³: Huawei

Draft Modification

□ The -01 version mainly includes the feedbacks from IANA.

- Adding the corresponding sections of this document as references for the new subregistries.
- Adding a statement reserving Value 2~255 for 'savRuleType' and 'savTargetType' subregistries in Sections 9.5 and 9.6.

9.5. IPFIX savRuleType (TBD1) Subregistry

* *Reference*: [I-D.ietf-savnet-general-sav-capabilities], Section 2 (Validation Modes)

* *Allocation Policy*: Expert Review [RFC8126]

* *Expert Guidance*: Experts should ensure that new values are consistent with the SAV architecture concepts defined in [I-D.ietf-savnet-general-sav-capabilities], particularly the validation modes and rule types (allowlist or blocklist).

Initial values:

Value	Name	Description
0	allowlist	The packet was validated against an allowlist
1	blocklist	The packet was validated against an blocklist

9.5. IPFIX savRuleType (TBD1) Subregistry

* *Reference*: [This document, savRuleType \(Section 4.2\)](#); [I-D.ietf-savnet-general-sav-capabilities], Section 2 (Validation Modes)

* *Allocation Policy*: Expert Review [RFC8126]

* *Expert Guidance*: Experts should ensure that new values are consistent with the SAV architecture concepts defined in [I-D.ietf-savnet-general-sav-capabilities], particularly the validation modes and rule types (allowlist or blocklist).

Initial values:

Value	Name	Description
0	allowlist	The packet was validated against an allowlist
1	blocklist	The packet was validated against an blocklist
2-255	unassigned	Reserved for future assignment

9.6. IPFIX savTargetType (TBD2) Subregistry

* *Reference*: [I-D.ietf-savnet-general-sav-capabilities], Section 2 (Validation Modes)

* *Allocation Policy*: Expert Review [RFC8126]

* *Expert Guidance*: Experts should consult [I-D.ietf-savnet-general-sav-capabilities] to ensure new values align with the defined target types (interface-based or prefix-based).

Initial values:

Value	Name	Description
0	interface-based	The rule is indexed by an interface
1	prefix-based	The rule is indexed by a prefix

9.6. IPFIX savTargetType (TBD2) Subregistry

* *Reference*: [This document, savRuleType \(Section 4.3\)](#); [I-D.ietf-savnet-general-sav-capabilities], Section 2 (Validation Modes)

* *Allocation Policy*: Expert Review [RFC8126]

* *Expert Guidance*: Experts should consult [I-D.ietf-savnet-general-sav-capabilities] to ensure new values align with the defined target types (interface-based or prefix-based).

Initial values:

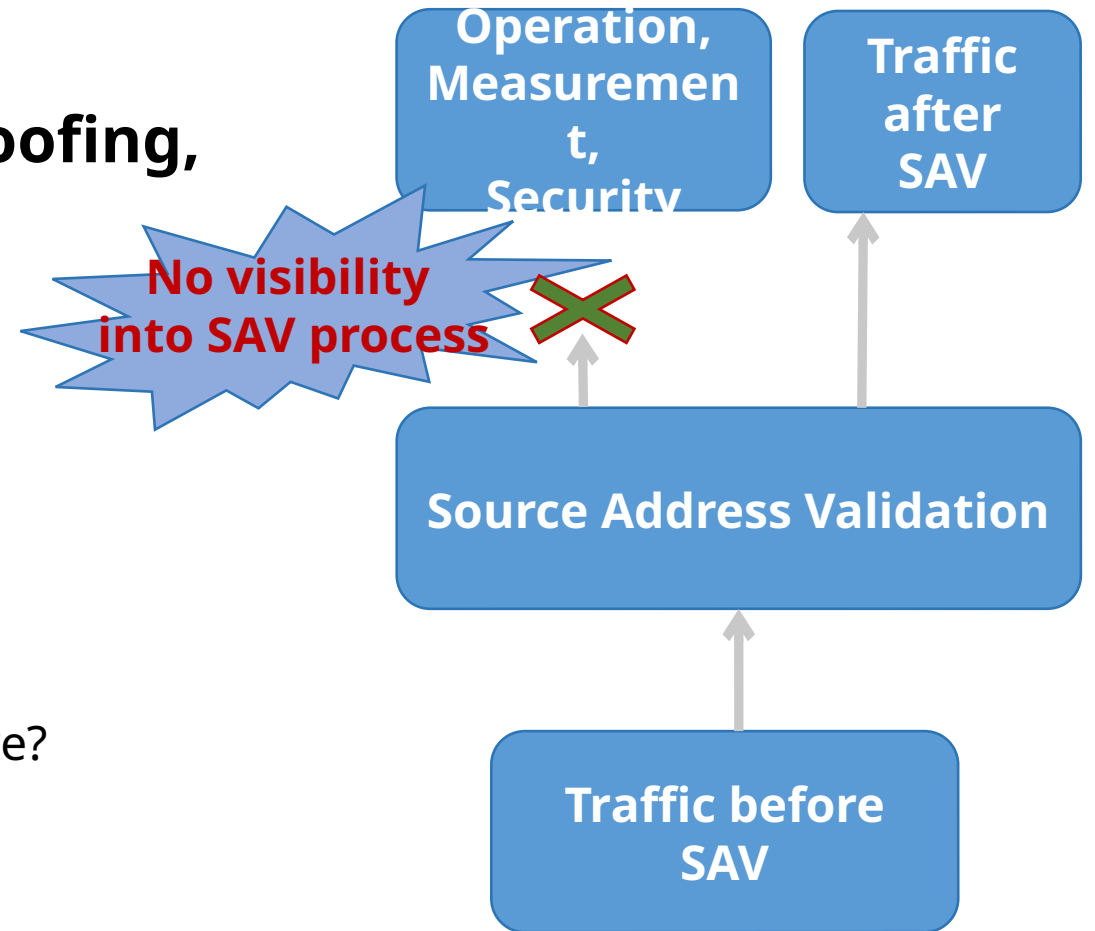
Value	Name	Description
0	interface-based	The rule is indexed by an interface
1	prefix-based	The rule is indexed by a prefix
2-255	unassigned	Reserved for future assignment

Gap: SAV Needs Visibility

□ Source Address Validation (SAV) is fundamental defense against IP spoofing, but current implementations **lack operational visibility**.

- Missing Piece:

- How many spoofed packets are dropped?
- From which interfaces do spoofed packets arrive?
- Which specific SAV rules trigger enforcement?
- Are SAV rules functioning correctly?



[What happens in the data plane]

Background: SAV Validation Modes

- Four Canonical Modes defined in the general SAV capabilities framework

	Interface-based	Prefix-based
Allowlist	Mode 1: "On interface X, what prefixes are allowed?"	Mode 3: "For prefix Y, which interfaces are allowed?"
Blocklist	Mode 2: "On interface X, what prefixes are blocked?"	Mode 4: "For prefix Y, which interfaces are blocked?"

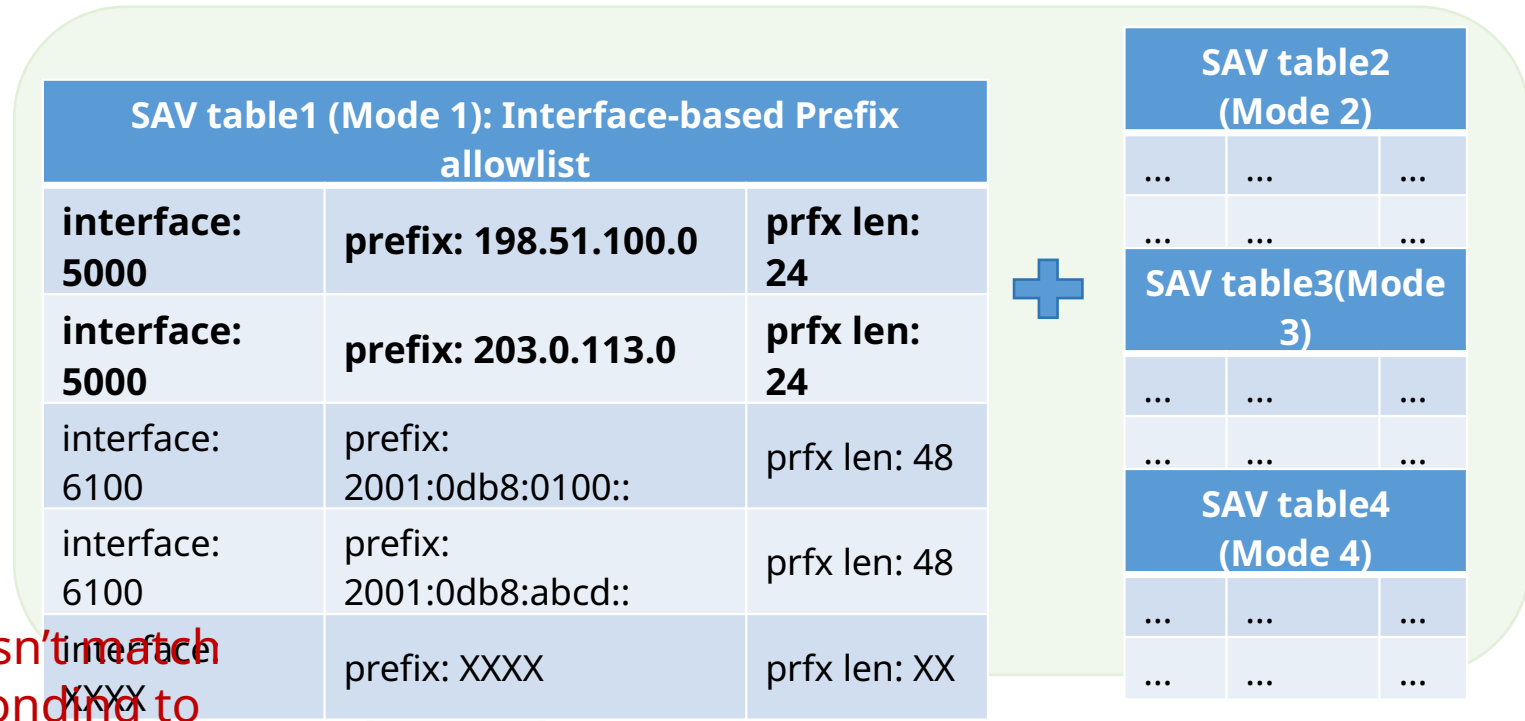
- Design Principles**

Conceptual Alignment with the SAV concepts and validation modes [*draft-ietf-savnet-general-sav-capabilities*]

Semantic Correlation to the SAV YANG data model

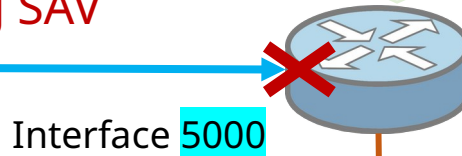
Background: SAV Enforcement

	Interface-based	Prefix-based
target type ↓ rule type		
Allowlist	Mode 1: "On interface X, what prefixes are allowed?"	Mode 3: "For prefix Y, which interfaces are allowed?"
Blocklist	Mode 2: "On interface X, what prefixes are blocked?"	Mode 4: "For prefix Y, which interfaces are blocked?"



Source IP address doesn't match with any prefix corresponding to interface 5000, triggering SAV enforcement.

Source IP Address:
203.8.113.8



export information of SAV enforcement via IPFIX

- What SAV mode was triggered?
- Why was it triggered - what SAV rules were matched?
- How were the spoofed packets handled?

Solution: IPFIX SAV Telemetry

- Design new Information Elements for SAV and export SAV information in IPFIX

IE Name	Abstract Data Type	Description
savRuleType	unsigned8	Identifies the rule as an Allowlist or Blocklist. <i>"What kind of rule was triggered?"</i>
savTargetType	unsigned8	Specifies if the rule is Interface-based or Prefix-based. <i>"Was the lookup key an Interface or a Prefix?"</i>
savMatchedContentList	subTemplate List	Carries the content of the SAV rules relevant to the decision. <i>"Which specific rule(s) were evaluated?"</i>
savPolicyAction	unsigned8	Reports the action applied to spoofed packets. <i>"What was done to the packet?"</i>

What SAV mode was triggered?

Why was it triggered - what SAV rules were matched?

How were the spoofed packets handled?

Key Design: Sub-Templates for SAV Rules

□ `savMatchedContentList` Element

◆ A structured data type `subTemplateList` ([RFC6313]), with each entry conforming to a sub-template.

◆ Each entry in the `savMatchedContentList` is a complete SAV rule.

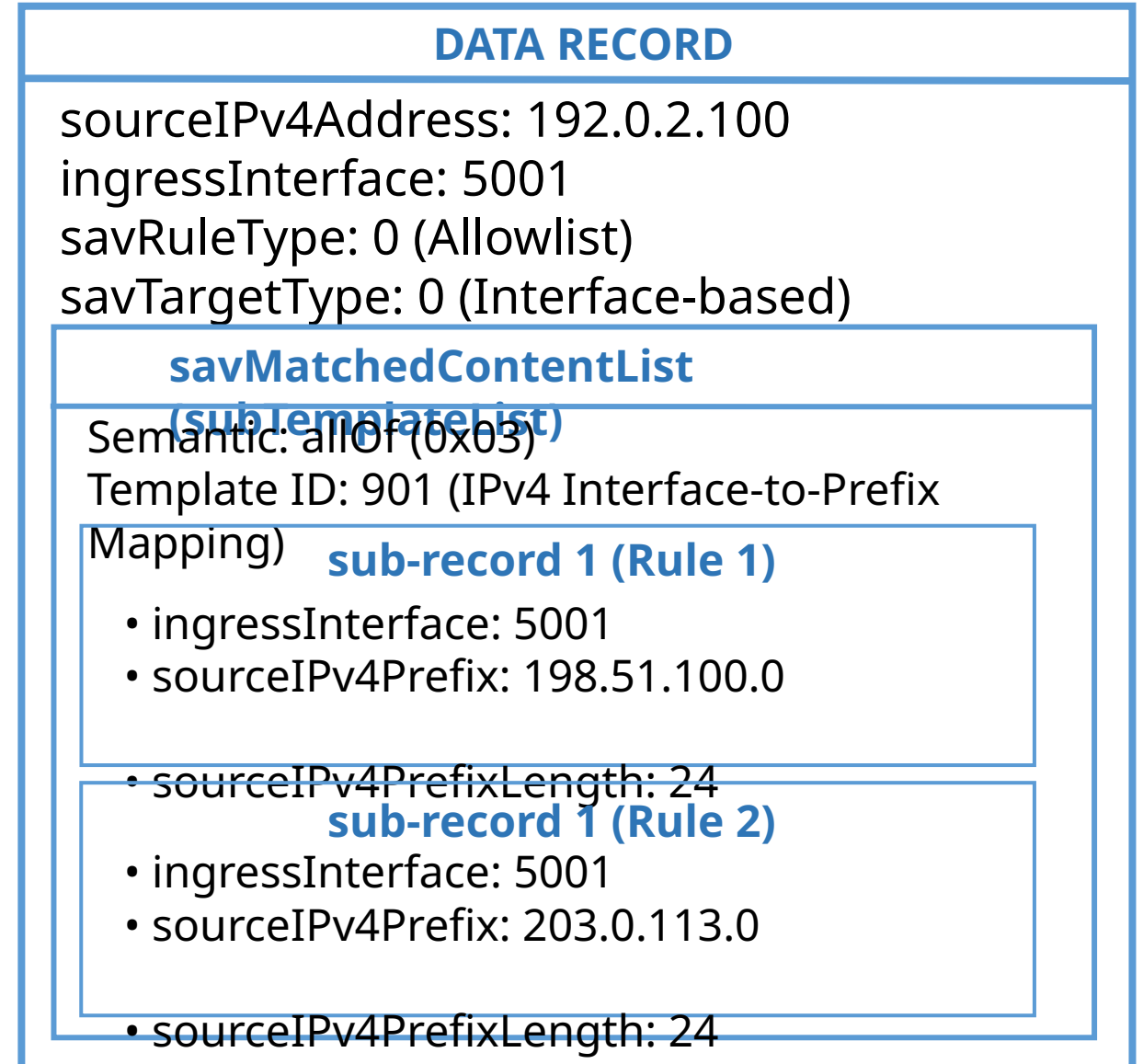
Sub-Template ID (user assign)	Usage	Key Fields (Rule Configuration)
901	IPv4 Interface-to-Prefix (Modes 1 & 2)	ingressInterface sourceIPv4Prefix sourceIPv4PrefixLength
902	IPv6 Interface-to-Prefix (Modes 1 & 2)	ingressInterface sourceIPv6Prefix sourceIPv6PrefixLength
903	IPv4 Prefix-to-Interface (Modes 3 & 4)	sourceIPv4Prefix ingressInterface sourceIPv4PrefixLength
	IPv4 Prefix-to-	sourceIPv6Prefix

4 sub-templates are defined to cover IPv4/6 address families and validation modes.

Example

An IPv4 prefix allowlist non-match event:

The list contains 'allOf'the SAV rules in the allowlist that were checked. The source prefix of the flow matched none of the prefix listed for that interface.



Hackathon Demo

```
--- Data Record 4 --- fields: 10, tmpl: 500 (0x01f4) --- Template ID for Template A
[Attribute] flowStartMilliseconds: 2024-12-26 00:00:00.768
[Attribute] flowEndMilliseconds: 2024-12-26 00:00:01.533
[Attribute] packetDeltaCount: 256
[Attribute] octetDeltaCount: 0
[Flow Key] ingressInterface: 5102
[Flow Key] sourceIPv4Prefix: 192.0.2.0
[Flow Key] savRuleType: [1] 0x00 → Allowlist mode
[Flow Key] savTargetType: [1] 0x01 → prefix-based mode
[Attribute] savPolicyAction: [1] 0x01 mode
[Flow Key] savMatchedContent: count: 3 {allof} tmpl: 902 (0x0386)
+--- STL Record 1 --- fields: 3, tmpl: 902 (0x0386) ---
+ sourceIPv4Prefix (44) : 192.0.2.0
+ sourceIPv4PrefixLength (9) : 24
+ ingressInterface (10) : 5002
+--- STL Record 2 --- fields: 3, tmpl: 902 (0x0386) ---
+ sourceIPv4Prefix (44) : 192.0.2.0
+ sourceIPv4PrefixLength (9) : 24
+ ingressInterface (10) : 5003
+--- STL Record 3 --- fields: 3, tmpl: 902 (0x0386) ---
+ sourceIPv4Prefix (44) : 192.0.2.0
+ sourceIPv4PrefixLength (9) : 24
+ ingressInterface (10) : 5004
```

ingressInterface does not match with any Allowlist rules

```
--- Data Record 15 --- fields: 12, tmpl: 502 (0x01f6) --- Template ID for Template B
[Flow Key] sourceIPv4Address: 192.0.2.40
[Flow Key] destinationIPv4Address: 198.18.0.1
[Flow Key] sourceTransportPort: 20459
[Flow Key] destinationTransportPort: 10927
[Flow Key] ingressInterface: 5102
[Attribute] packetDeltaCount: 52
[Attribute] flowStartMilliseconds: 2024-12-26 00:00:00.768
[Flow Key] protocolIdentifier: 6
[Attribute] savRuleType: [1] 0x00
[Attribute] savTargetType: [1] 0x01
[Attribute] savPolicyAction: [1] 0x01
[Flow Key] savMatchedContent: count: 3 {allof} tmpl: 902 (0x0386)
+--- STL Record 1 --- fields: 3, tmpl: 902 (0x0386) ---
+ sourceIPv4Prefix (44) : 192.0.2.0
+ sourceIPv4PrefixLength (9) : 24
+ ingressInterface (10) : 5002
+--- STL Record 2 --- fields: 3, tmpl: 902 (0x0386) ---
+ sourceIPv4Prefix (44) : 192.0.2.0
+ sourceIPv4PrefixLength (9) : 24
+ ingressInterface (10) : 5003
+--- STL Record 3 --- fields: 3, tmpl: 902 (0x0386) ---
+ sourceIPv4Prefix (44) : 192.0.2.0
+ sourceIPv4PrefixLength (9) : 24
+ ingressInterface (10) : 5004
```

These three SAV rules were 'all of' the SAV rules in the allowlist that were involved in the SAV validation, and none of them was matched, indicating spoofed packets.

Conclusion

□ Making SAV information visible with IPFIX

- ◆ Four new IPFIX Information Elements for SAV enforcement
- ◆ 'SubTemplateList' structured data type for SAV rules
- ◆ Hackathon Demo
- ◆ More understanding for effective monitoring, troubleshooting and threat analysis