Export of Source Address Validation (SAV) Information in IPFIX

Presented by CAO Qian

Joint work by: Mingqing HUANG[1], Benoît CLAISE[2], Tianran ZHOU[3]

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

The Problem - SAV Visibility Gap

SAV is fundamental defense against IP spoofing, but current implementations lack operational visibility.

- Critical questions need to be answered:
  - How many spoofed packets are detected and dropped?
  - From which interfaces do spoofed packets arrive?
  - Which specific SAV rules trigger enforcement?
  - Are SAV rules functioning correctly?

Solution Overview - IPFIX SAV Telemetry

Approach

- Extends IPFIX protocol for SAV-specific telemetry

Objectives

- Capturing validation context and outcomes
- Identification of validation modes and triggering rules
- Reporting policy actions on spoofed packets

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

SAV Validation Modes

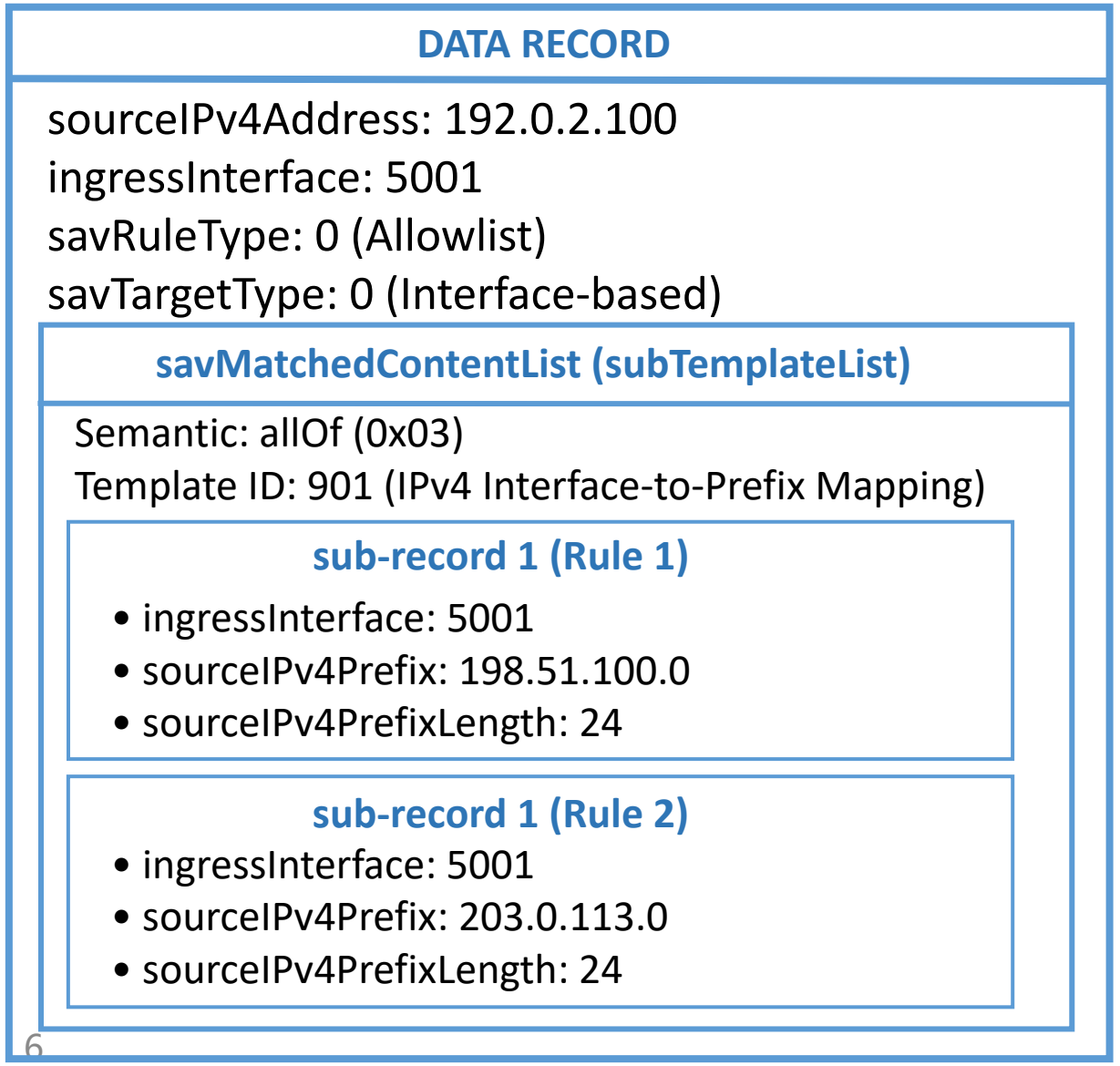
- Four Canonical Modes defined in the general SAV capabilities framework
  - Interface-based Validation
    - **Mode 1:** Prefix Allowlist: "On interface X, what prefixes are allowed?"
    - **Mode 2:** Prefix Blocklist: "On interface X, what prefixes are blocked?"
  - Prefix-based Validation
    - **Mode 3:** Interface Allowlist: "For prefix Y, which interfaces are allowed?"
    - **Mode 4:** Interface Blocklist: "For prefix Y, which interfaces are blocked?"

Core Information Elements (IE)

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

Key Design - Rule Context Capture

- These IEs work together to provide detailed context. `savRuleType`, `savTargetType`, and `savPolicyAction` describe the **what** and **how**, the `savMatchedContentList` explains the **why**.
- Capturing the specific SAV rules: The `savMatchedContentList` IE, encoded as a `subTemplateList` [RFC6313], acts as a container for structured SAV rule data.



* *Example: An IPv4 allowlist non-match event. The list contains all SAV rules in the allowlist that were checked, showing the packet matched none (allOf).*

Key Design - Sub-Templates for SAV Rules

- `savMatchedContentList` Element
  - Each entry in the `savMatchedContentList` is a complete SAV rule.
  - Four sub-templates are defined to cover IPv4/6 address families and validation modes.

Sub-Template ID (exporter 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
904	IPv4 Prefix-to-Interface (Modes 3 & 4)	sourceIPv6Prefix ingressInterface sourceIPv6PrefixLength

Use Cases

- SAV Enforcement Monitoring:
  - Quantify spoofing attempts per rule type
  - Track SAV effectiveness over time
- Rule-Level Attribution and Troubleshooting:
  - Identify exact SAV rule configurations
  - Audit rule correctness and coverage
- Forensic Analysis and Compliance:
  - Correlate SAV Data Records with other data sources for incident investigation and compliance reporting.

Operational Considerations

- For Exporters:
  - encode the subTemplateList structure and ensure semantic
  - define the sub-templates used within the subTemplateList before exporting data
- For Collectors:
  - parsing the `subTemplateList` and understanding the context-dependent semantics
- IPv4/IPv6 support:
  - separate, dedicated sub-templates

Example - Scenario Setup

Two Observed SAV Validation Events

Event ID	Source Address	Ingress Interface	Rule Type	Target Type	Policy Action
1	192.0.2.100	5001	Allowlist	Interface	Discard
2	2001:db8::1	5001	Blocklist	Prefix	Discard

Event 1: IPv4 Allowlist Non-Match

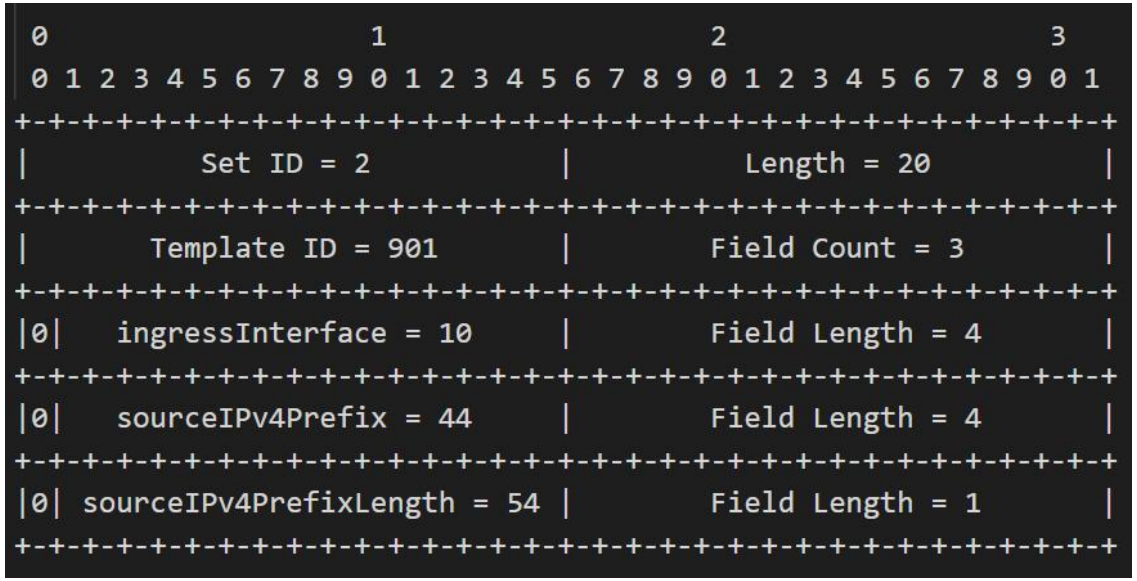
- Source IP 192.0.2.100 not in any allowed prefixes for interface 5001
- Policy: rate limit

Event 2: IPv6 Blocklist Match

- Interface 5001 explicitly blocked for prefix 2001:db8::/32
- Policy: discard packet

Example - Sub-Templates for Rule Content

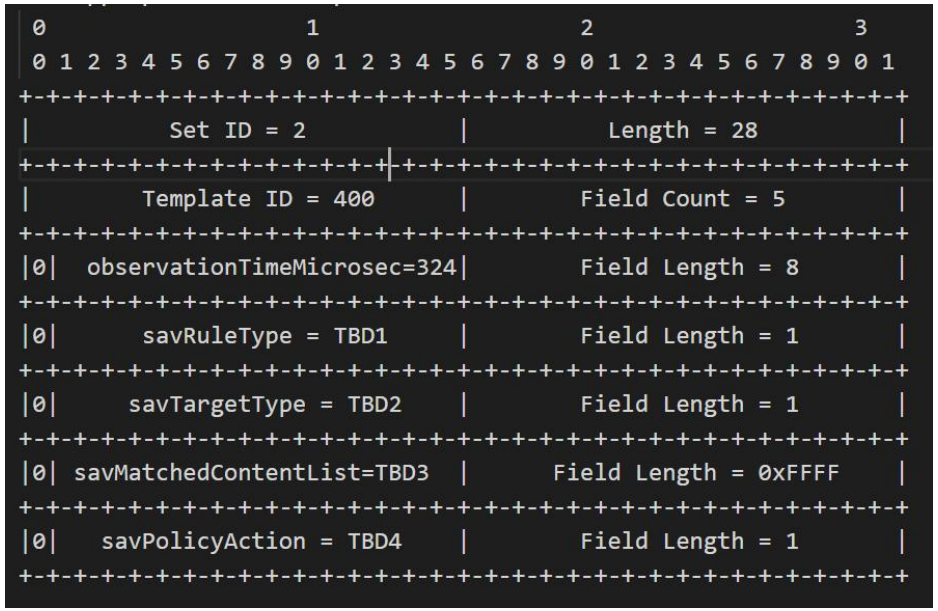
◆Sub-Template 901: IPv4 Interface-to-Prefix Mapping Rules



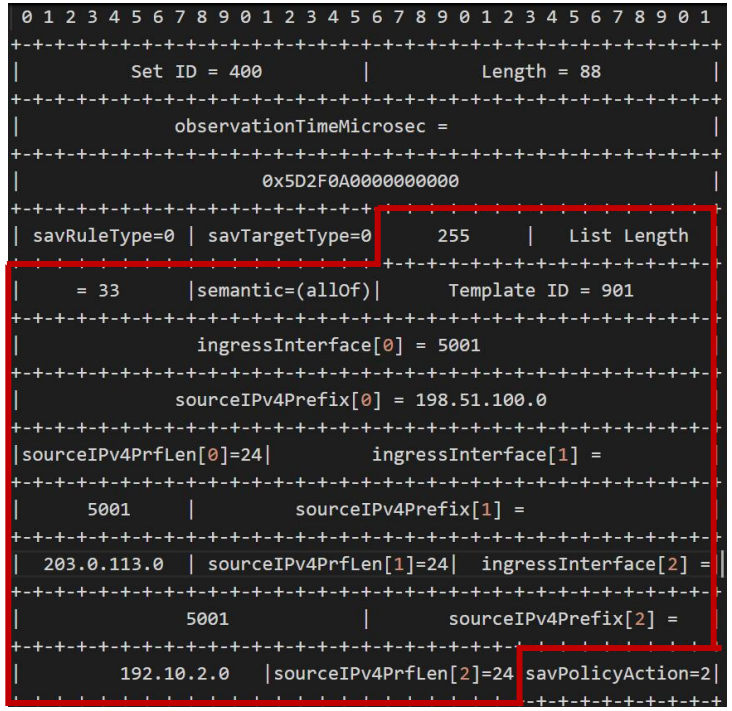
- Four different Sub-Templates (901-904) for:
  - IPv4 Interface-to-Prefix Mapping Rules
  - IPv6 Interface-to-Prefix Mapping Rules
  - IPv4 Prefix-to-Interface Mapping Rules
  - IPv6 Prefix-to-Interface Mapping Rules

Example - Data Export Details

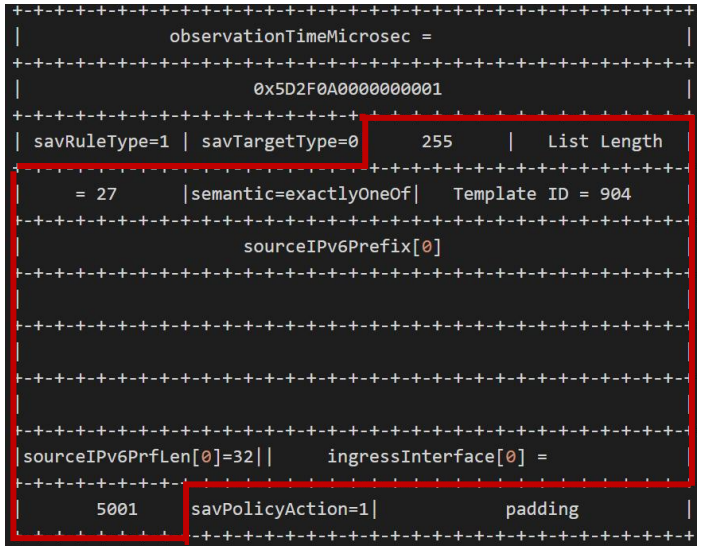
- Template Example (ID 400)



- Data Set Example



Event 1 Data Record: IPv4 Allowlist Non-Match
(Sub-template list with 3 SAV rules, highlighted in red)



Event 2 Data Record: IPv6 Blocklist Match
(Sub-template list with 1 SAV rule, highlighted in red)

Conclusion & Open Issues

Technical Summary:

- Fills critical operational **visibility** gap in SAV
- **Vendor-neutral implementation-independent** approach based on IPFIX
- Provides comprehensive **context for SAV** enforcement
- Enables effective monitoring, troubleshooting and threat analysis

Effective Impact and Potential Contributions

- Operational Verification: Confirm SAV is actively and correctly enforcing policies
- Root-Cause Analysis: Determine why specific packets were deemed invalid
- Threat Intelligence: Quantify spoofing attempts and analyzing attack patterns