A Profile for ROV Deployment Transparency

draft-ling-sidrops-rov-tag-profile-01

Beijing Zhongguancun Lab, Tsinghua University, Capital Normal University

Sitong Ling, Ke Xu, Qi Li, Zhuotao Liu, Xiaoliang Wang

2026.03.19

Background: Security Vulnerability in Partial ROV Deployment

Current Deployment Reality

ROV deployment remains partial; partial deployment is a lasting reality

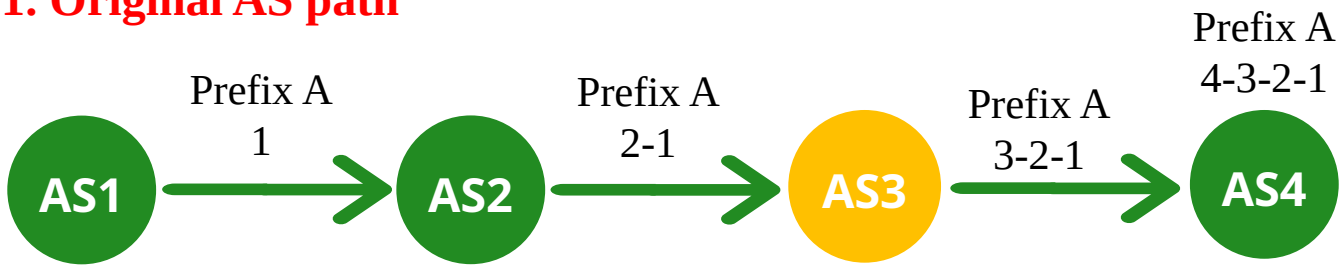
Security Vulnerability

When AS detects hijacked route from upstream AS → Upstream hasn't deployed ROV → Current path compromised → **Traffic may be hijacked**

Challenge

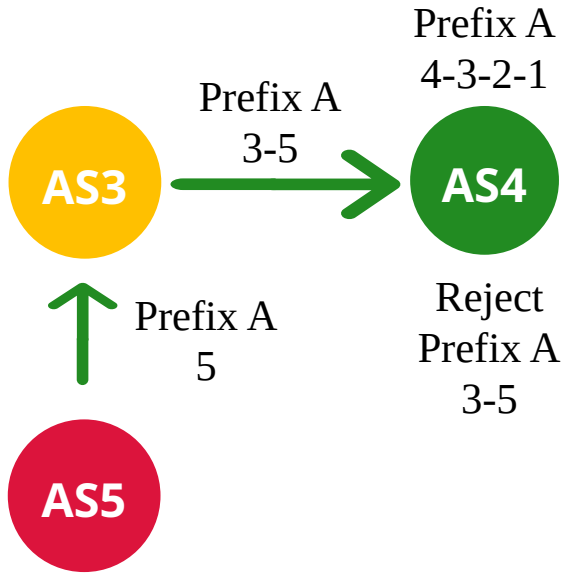
Cannot identify which alternative paths go through **ROV-deployed upstream ASes**

1. Original AS path



2. Source hijack

1. AS4 performs ROV and filters out the hijacked route (AS5 → AS4)
2. AS4 continues using the original path AS4 → AS3 → AS2 → AS1
3. AS3, however, forwards traffic toward the fake origin AS5
4. Traffic is hijacked even though AS4 has deployed ROV



Solution & Mechanism

Solution Overview

- ROV_TAG: a new RPKI object declaring ROV deployment status
- Core field: `rovDeployed` (BOOLEAN, must be TRUE); structure: version, asID, rovDeployed
- Validation follows RFC6488 template; target: Transit ASes (Stub AS not recommended)

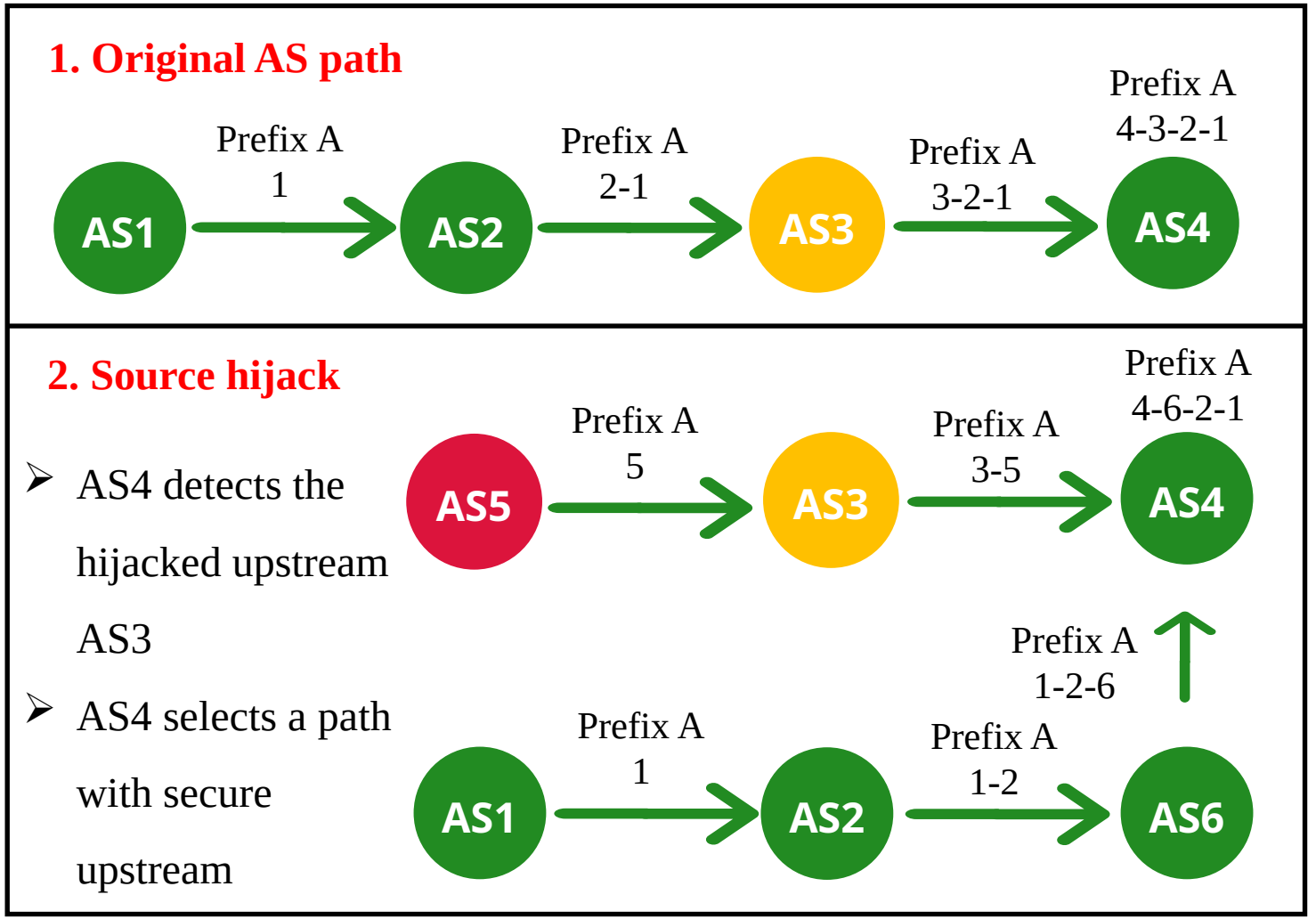
Defensive Mechanism (Triggered only when a security problem is detected)

1. AS performs ROV validation → detects hijacked route
2. Identifies immediate upstream AS that propagated it and recognizes current path also goes through that upstream
3. If it continues on current path, traffic will be hijacked
4. Uses ROV_TAG to find alternative paths whose immediate upstream has declared ROV deployment
5. May prefer these safer alternative paths to avoid the compromised upstream

Key: does not change normal routing decisions

Example Scenario

- Without ROV_TAG: AS4 cannot tell which alternative paths go through ROV-deployed upstreams
- With ROV_TAG: AS4 fetches ROV_TAG objects, learns which of its upstreams (AS6, …) have declared ROV deployment
- AS4 can prefer paths whose immediate upstream has ROV_TAG with rovDeployed=TRUE, avoiding the compromised upstream AS3



ROV_TAG provides the missing visibility for path selection when a security problem is detected

Technical Details & Limitations

Technical Details

- ASN.1: `ROVDeploymentAttestation ::= SEQUENCE { version [0] INTEGER DEFAULT 0, asID ASID, rovDeployed BOOLEAN }`
- Validation follows RFC6488 template; storage: efficient lookup (e.g., hash tables)
- Target: Transit ASes (Stub AS registration not recommended — they typically have limited path choice)

Limitations & Mitigations

- Self-asserted: no cryptographic verification that an AS actually performs ROV

Mitigations:

- Defensive use: triggered only when the security problem is detected; reduces abuse incentive
- Public reputation: false claims carry reputation risk in routing community
- Local violation records: optional mechanism to record inconsistent behavior, prefer ASes with no inconsistency record when selecting paths

Implementation & Deployment

How it works in practice

- Registration: Transit ASes that have deployed ROV create and publish ROV_TAG objects in RPKI (same distribution as ROA, ASPA)
- Consumption: RP fetches and validates ROV_TAG objects; delivers to routers via RPKI-to-router (RTR)
- Trigger: Only when hijacked route is detected from immediate upstream — then consult ROV_TAG to identify safer alternative paths
- Scope: Complements ROV; does not replace it; addresses the partial-deployment gap

Why this approach

- Lightweight: single boolean field; no new protocol
- Optional: ASes can adopt incrementally; no change to normal routing when no security problem

Next Steps & Q&A

Status： draft-01 submitted; seeking feedback and discussion

Optional Future Direction: fine-grained control

- Current: single `rovDeployed` per AS (coarse-grained)
- Potential extension: per-neighbor declaration — which neighbors' routes trigger ROV (e.g., "ROV performed for routes from AS X, Y; not for routes from AS Z")
- Enables more precise path selection when an AS performs ROV only for routes from certain upstreams

Questions welcome!