

# SecureMPV: Secure Multi-Path Verification

Sung Woo Jeon<sup>+</sup>, Tanya Banerjee<sup>\*</sup>, Wadia Ganim<sup>\*</sup>, Kulani Mahadewa<sup>\*</sup>, Utku Tefek<sup>\*</sup>, Ertem Esiner<sup>\*</sup>, Deming Chen<sup>+</sup>, Matthew Caesar<sup>+</sup>  
<sup>\*</sup>Illinois Advanced Research Center at Singapore Ltd., <sup>+</sup>University of Illinois at Urbana-Champaign

## Motivation

- **Modern networks are non-deterministic:** due to ECMP, failover, and elastic service pools, packets may traverse multiple valid paths.
- Existing path-verification systems assume a single-fixed path within path-aware network architectures.
- However, many enterprise and service-chain deployments lack such architectural support, leaving policy enforcement vulnerable to bypass by compromised in-network devices.

**Vision:** What if every packet could prove it followed an authorized policy execution, even under non-deterministic routing?

## Our Solution

We propose SecureMPV, an adversarially resilient path-set verification framework:

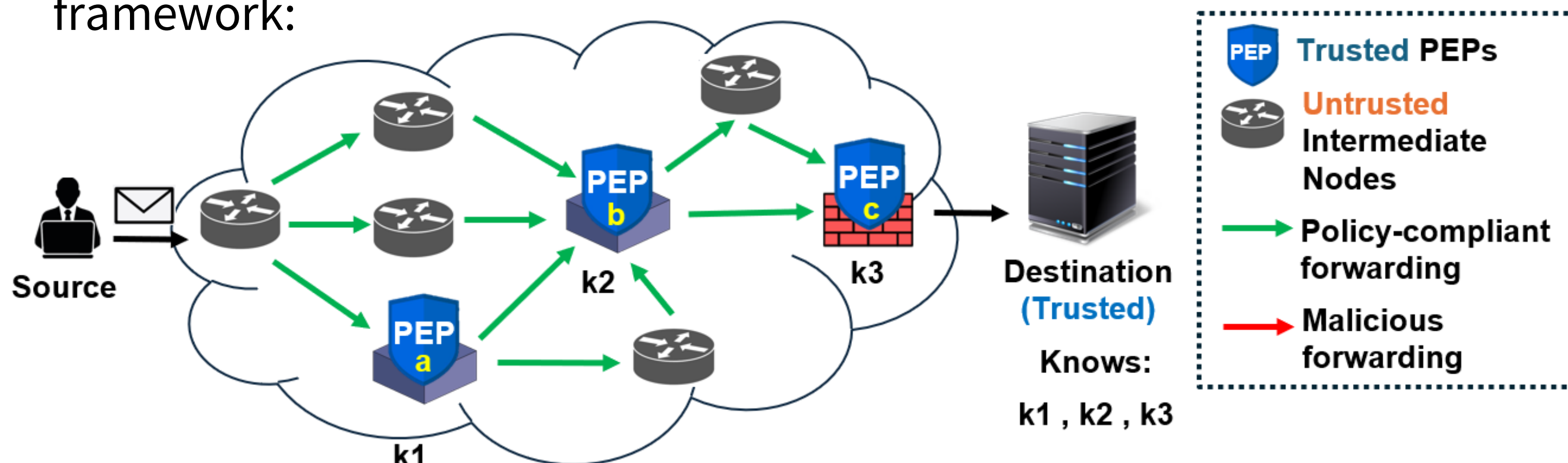


Fig. 1: Multiple-valid paths that match the policy in Fig. 2

### 1) Policy Specification:

Policies are written as star-free regular expressions. Then compiled into acyclic Deterministic Finite Automaton (DFA) representing all valid path traversals.

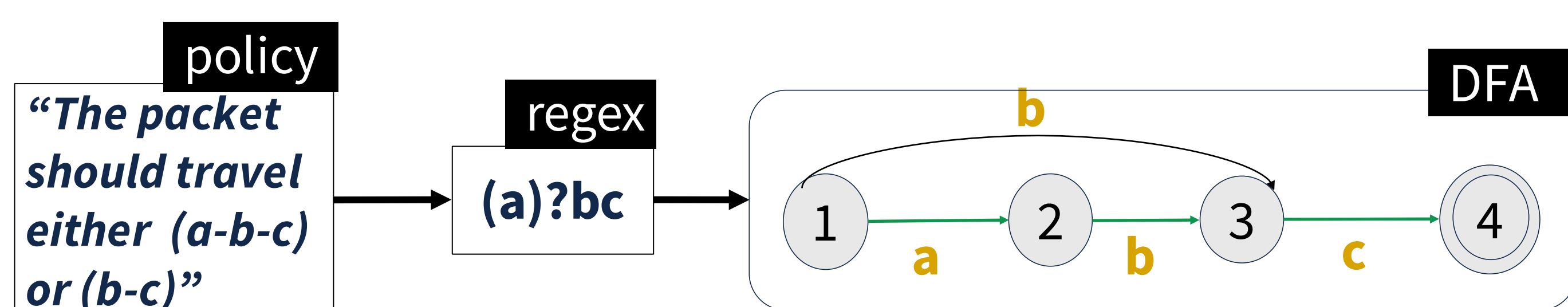


Fig. 2: An example of policy and its regex and DFA representations

### 2) Secure Policy Enforcement via Distributed State Machines:

Each packet carries its current DFA state and cryptographic evidence, along with a bit vector marking traversed PEPs to enable path reconstruction at the destination.

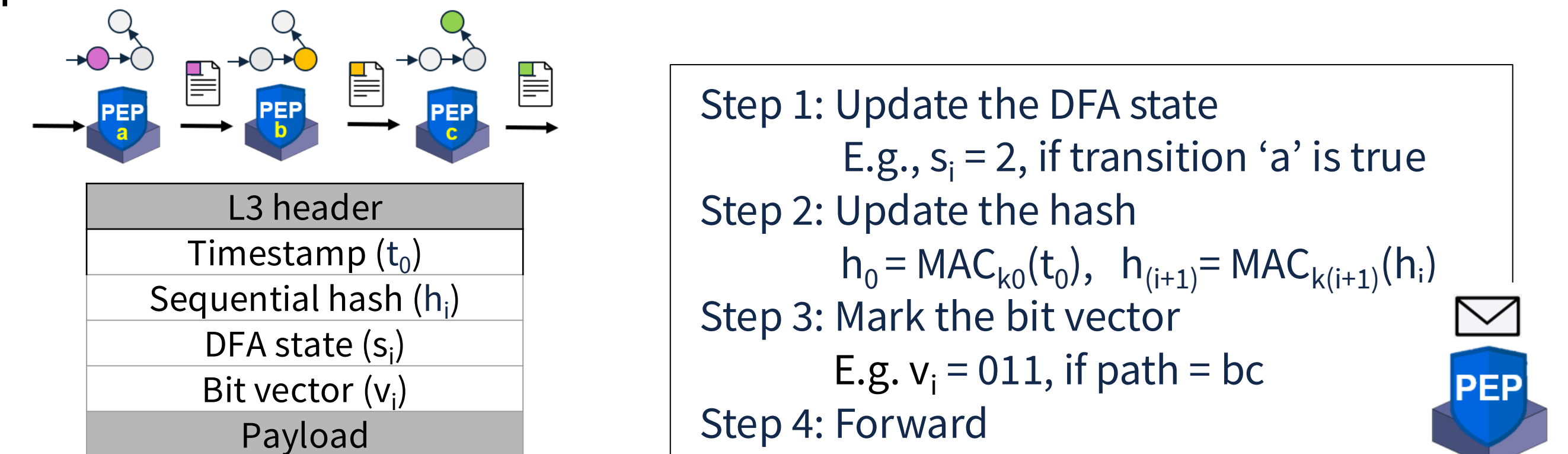


Fig. 3: Packet header format

Fig. 4: Packet Processing at a PEP

### 3) Path Verification at the Destination:

The destination validates that the packet's DFA state is accepting and verifies the traversal against the specified policy, ensuring no forgery or tampering occurred

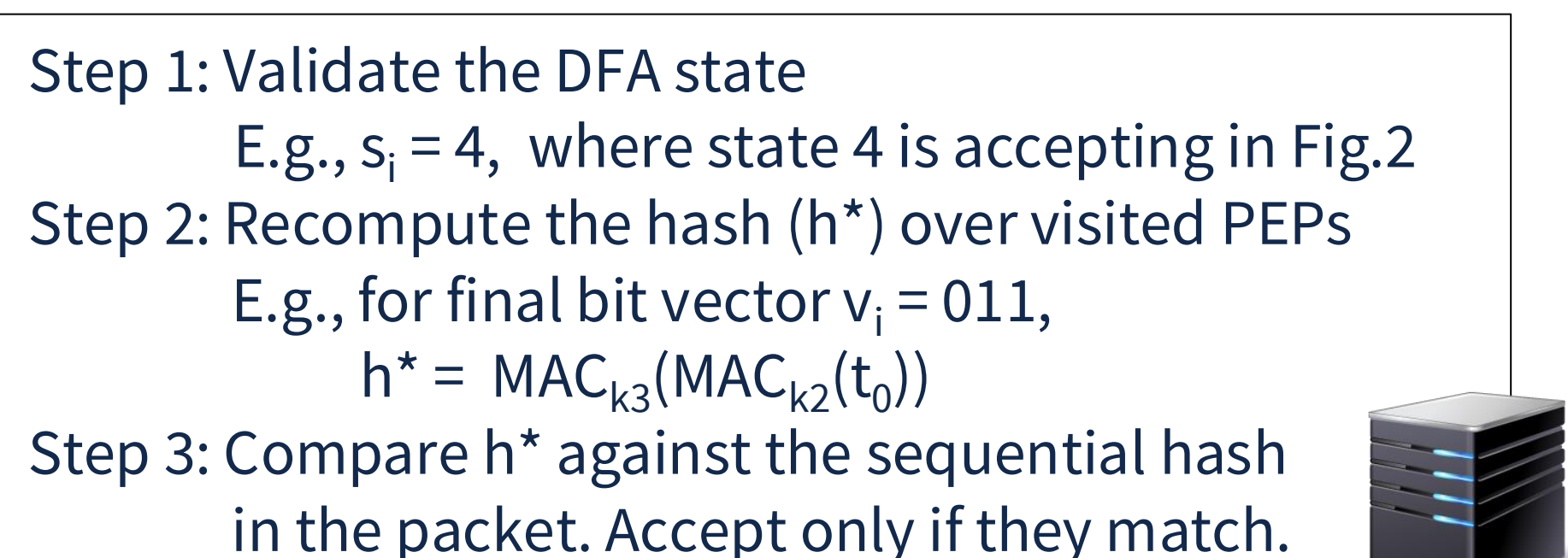


Fig. 5: Verification steps for path (b-c) in Fig. 2

## Objectives

Our goal is to **design a path verification system** that:

- Verifies compliance with an authorized set of service-chain executions, rather than a single fixed path
- Remains secure under strong adversaries, even if packets are injected, modified, or replayed
- Operates at line rate with low overhead

## Threat Model

- **Adversary Capabilities:** may inject, replay, drop, reorder, or modify packets and control non-enforcement network devices.
- **Trusted Entities:** Policy Enforcement Points (PEPs) and the destination are trusted and hold cryptographic secrets.
- **Untrusted Entities:** The source and intermediate forwarding devices are not trusted.

## Case Study

Below we demonstrate how SecureMPV can detect attempts to bypass required security functions (e.g., firewall, IDS)

**Bypass Security Checkpoints Detection:** If a packet skips a mandated PEP, even if an attacker attempts to forge a valid DFA state and a bit vector, they cannot correctly extend the hash chain without the skipped PEP's secret key. As a result, the recomputed hash chain at the destination will not match the packet's accumulated hash.

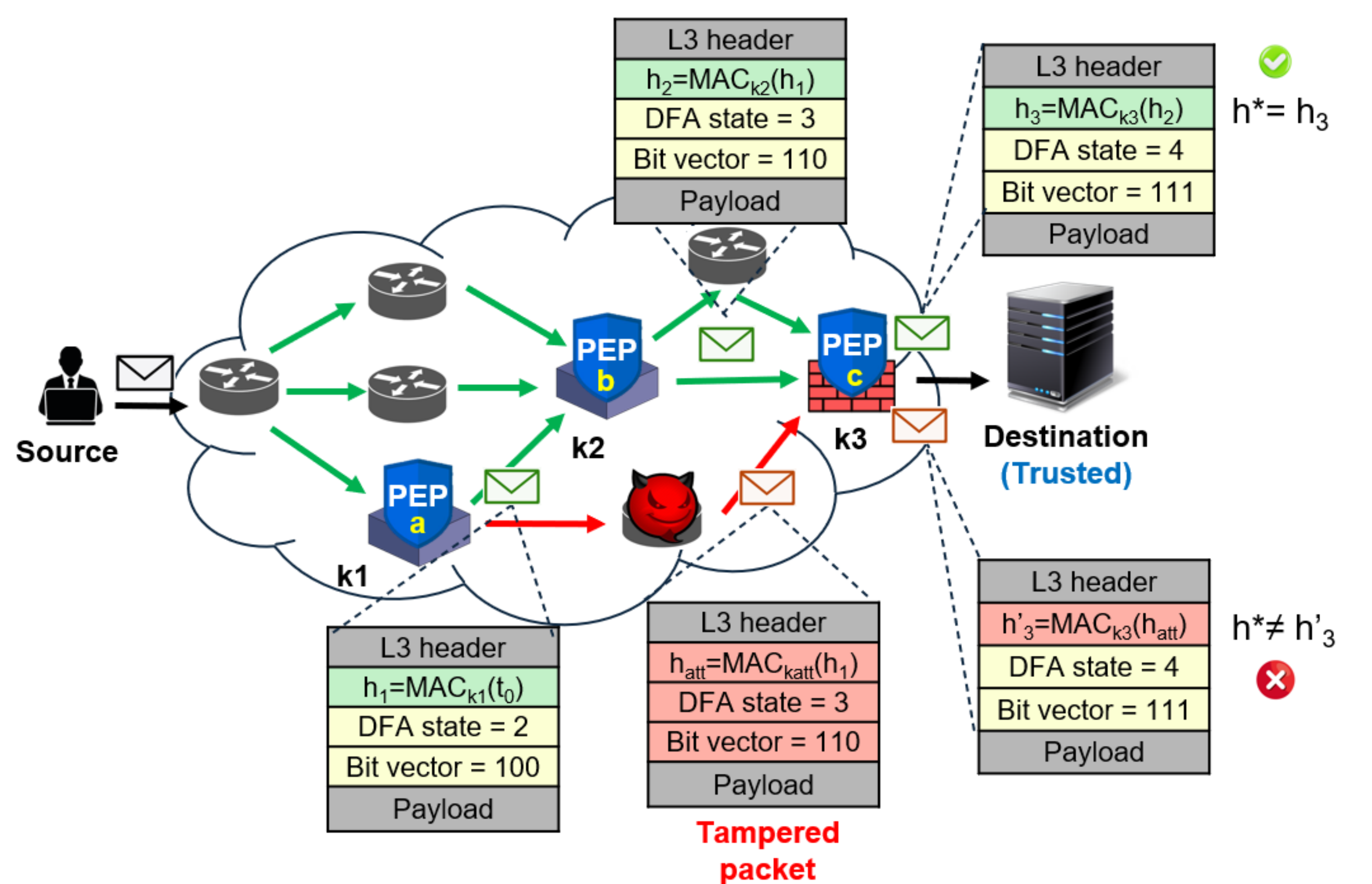


Fig. 6: Policy compliant vs non-compliant (attacker bypass PEP b) forwarding

## Experimental Evaluation

- We implemented our design on the Intel Tofino simulator and NVIDIA BlueField devices.
- On Tofino, hash chaining requires packet recirculation due to hardware limitations, reducing peak throughput.
- Our BlueField implementation leverages hardware cryptographic accelerators, eliminating recirculation overhead and enabling line-rate verification in principle. Our initial experiments show minimal ~40  $\mu\text{s}$  per-packet overhead at each PEP (sub-millisecond).