# Non-interoperability Detection for Routing Protocol Implementations

## Motivation

<table>
<thead>
<tr>
<th>Non-interoperability</th>
<th>Prior Approaches</th>
<th>Black-Box Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Routing protocol standards are expressed in natural language which may be abstract or ambiguous. Different implementations of a routing protocol may embody different interpretations of the standard, leading to interoperability issues when used within/across routing domains.</td>
<td><strong>Standard</strong></td>
<td><strong>Model</strong></td>
</tr>
<tr>
<td>Forwards long AS path, Reboot</td>
<td>Does not consider actual implementations. [9]</td>
<td></td>
</tr>
<tr>
<td>Mikrotik Router</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cisco Router</td>
<td>Requires constructing a formal model that embodies the standard and does not elucidate differences between implementations. [4-7, 10]</td>
<td></td>
</tr>
<tr>
<td><strong>Implementation A</strong></td>
<td><strong>Implementation B</strong></td>
<td><strong>Compare</strong></td>
</tr>
<tr>
<td>Utilizes symbolic execution which requires access to implementations’ source code. [8]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Approach

### Basic Idea

We infer the correlation (i.e., **packet causal relationship**) between the sent (or received) packets to determine the set of **expected responses**.

**Naive Approach:** After a packet $A$ is sent (or received) by a router, if packet $B$ is the first packet received (or sent) by the same router, then we assume there is a causal relationship between the sending (or receiving) of $A$ and the receiving (or sending) of $B$.

### Problem

We want to compute packet causal relationships that are both **accurate** (reflected packets are indeed causally related) and **extensive** (consider and analyze different networks scenarios).

**High frequency packet exchange and small time gap** between packets often result in scenarios where a router receives multiple packets in chaotic order after sending a packet (or vice versa). This can lead to **incorrect inferences** of the packet causal relationships.

### Solution

1. Configure a **fixed delay ($T_{Delay}$)** on all network interfaces to exclude non-relevant packets from packet causal relationships.
   - Only consider packets after at least $2T_{Delay}$.
   - $T_{Delay}$ should be more than the variance in round trip time (RTT) and processing time and less than the re-transmit timeout.

2. Use diverse topologies to improve extensiveness.
   - **Linear** topologies with 2 or 5 routers and **mesh** topologies with 3 or 5 routers

## Evaluation

### Experimental Setup

To evaluate the effectiveness of the technique, we apply it to the **FRRouting** [2] and **BIRD** [1] implementations of **OSPF**.

We run these implementations in **Docker containers** connected by virtual links.

$T_{Delay}$ is added using the Pumba [3] chaos testing tool. We set $T_{Delay}$ to 900 ms which is higher than the variance in the RTT and lower than the re-transmit timeout in both of the implementations.

## Results

- Inferred causal relationships for packets **differențiated** by OSPF packet type, where **missing** relationships are represented with $\emptyset$.

- More specific packet causal relationships: whether the sending (or receiving) of **Link State Update (LSU)** or **Link State Acknowledgment (LSAck)** packets can trigger the sending (or receiving) of LSU or LSAck packets with greater Link State Advertisement sequence numbers (LS-SN).

We observe **clear discrepancies** between the implementations which are flagged as possible causes of non-interoperability.

## Future Work

- **Validate** our black-box inferences by examining the implementation source code.
- **Verify** whether (or what fraction of) our flagged potential causes of non-interoperabilities indeed lead to bugs through packet injection.
- **Scale** our system to consider more packet fields and other router features.

## References