Non-interoperability Detection for Routing Protocol Implementations

Xi Jiang, Aaron Gember-Jacobson (Colgate University)

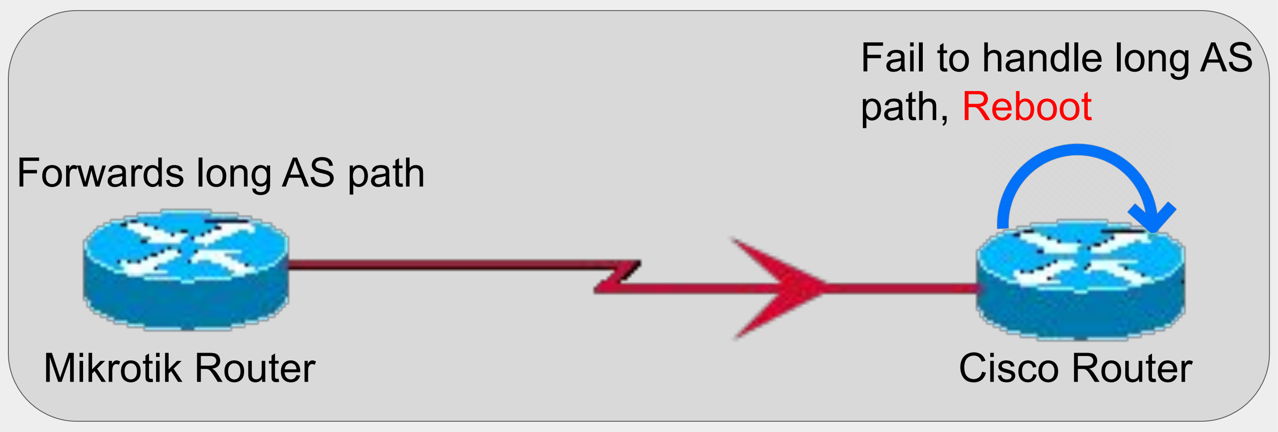
Motivation

Non-interoperability

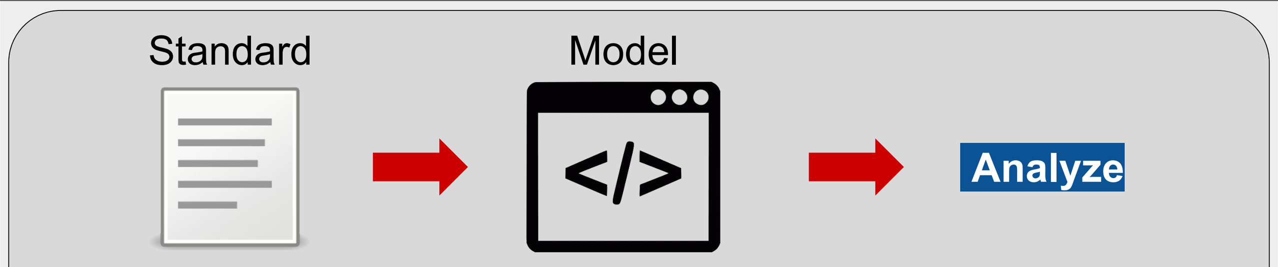
Routing protocol standards are expressed in natural language which may be ***abstract*** or ***ambiguous***.

Different ***implementations*** of a routing protocol may embody ***different interpretations*** of the standard, leading to interoperability issues when used within/across routing domains.

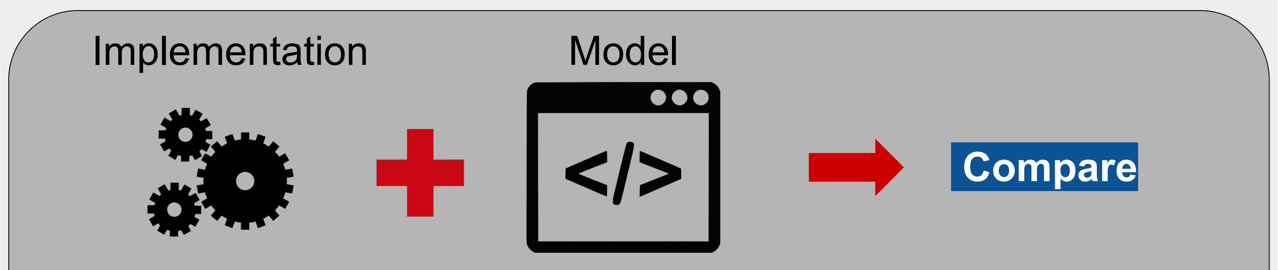
Example: 2009 Supronet Incident [11]



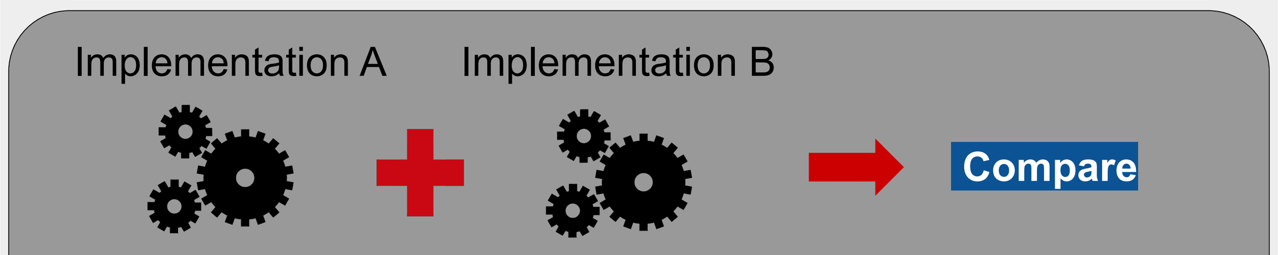
Prior Approaches



Does not consider ***actual implementations***. [9]



Requires ***constructing a formal model*** that embodies the standard and does not elucidate differences between implementations. [4-7, 10]



Utilizes symbolic execution which ***requires*** access to implementations' ***source code***. [8]

Black-Box Approach

We present a ***black-box*** technique for detecting interoperability issues between routing protocol implementations ***based on the packets routers send and receive***.

✓ Avoids the need to translate a protocol standard's natural language into a formal model.

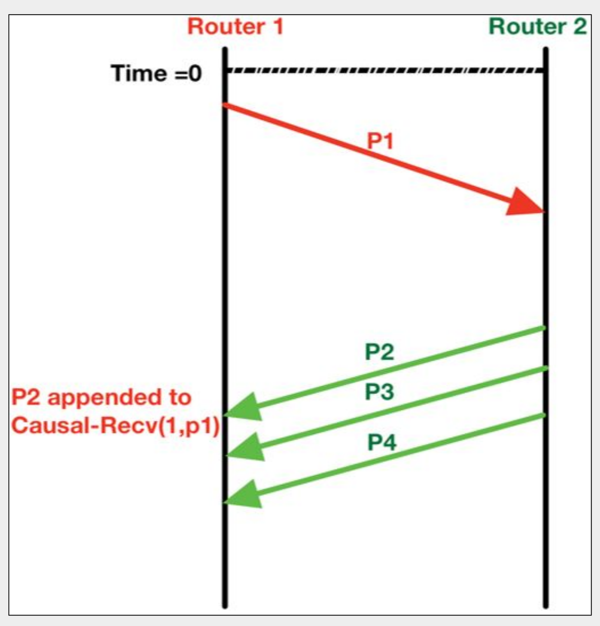
✓ Does not require access to implementations' source code, which enables our technique to be applied to commercial protocol implementations.

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.

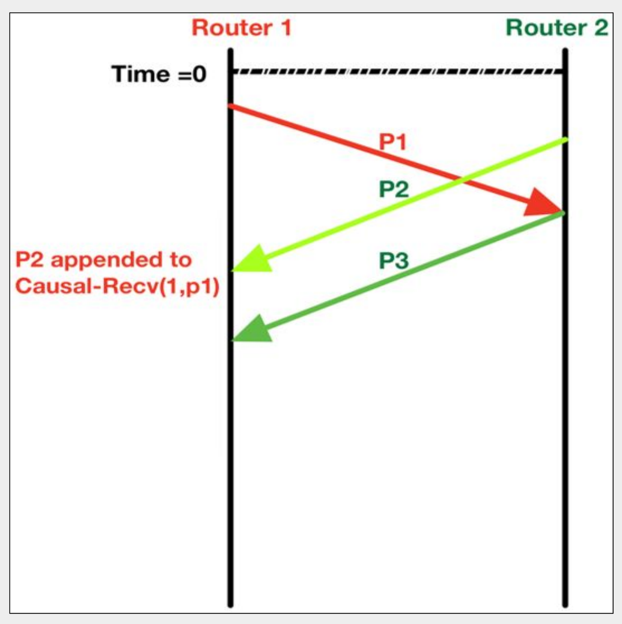


Correct inference

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).

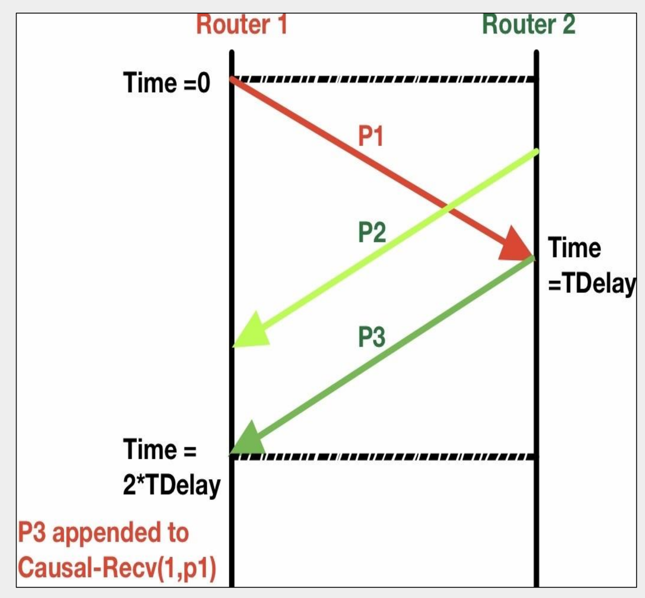
High frequency packet exchange and ***small time gap*** between packets often result in scenarios where a router receives ***multiple packets in chaotic order*** after sending a packet (or vice versa). This can lead to ***incorrect inferences*** of the packet causal relationships.



Incorrect inference

Solution

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



Correct inference with TDelay

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.

TDelay is added using the Pumba [3] chaos testing tool. We set TDelay 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

	FRR					BIRD				
	Snd(1)	Snd(2)	Snd(3)	Snd(4)	Snd(5)	Snd(1)	Snd(2)	Snd(3)	Snd(4)	Snd(5)
Rcv(1)	✓	✓	✓	✓	✓	✓	✓	∅	✓	✓
Rcv(2)	✓	✓	✓	∅	∅	✓	✓	✓	∅	∅
Rcv(3)	∅	∅	∅	✓	∅	∅	✓	✓	∅	∅
Rcv(4)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Rcv(5)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

⇦ Inferred causal relationships for packets ***differentiated*** by OSPF packet ***type***, where ***missing*** relationships are represented with ∅

	FRR		BIRD	
	Snd(LSU)	Snd(LSAck)	Snd(LSU)	Snd(LSAck)
Rcv(LSU) with greater LS-SN in LSA	✓	✓	✓	✓
Rcv(LSAck) with greater LS-SN in LSA	∅	∅	✓	∅

⇧ 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

[1] The BIRD Internet Routing Daemon Project. https://bird.network.cz.
[2] FRRouting Protocols. https://frrouting.org.
[3] Pumba. https://github.com/alexei-led/pumba/.
[4] Silva Alexandra. 2021. Prognosis: Black-Box Analysis of Network Protocol Implementations.
[5] Kenneth L. McMillan and Lenore D. Zuck. 2019. Formal specification and testing of QUIC. In SIGCOMM.
[6] Madanlal Musuvathi and Dawson R. Engler. 2004. Model checking large network protocol implementations. In NSDI.
[7] Madanlal Musuvathi, David Y. W. Park, Andy Chou, Dawson R. Engler, and David L. Dill. 2003. CMC: a pragmatic approach to model checking real code. ACM SIGOPS Operating Systems Review 36, SI.
[8] Luis Pedrosa, Ari Fogel, Nupur Kothari, Ramesh Govindan, Ratul Mahajan, and Todd Millstein. 2015. Analyzing protocol implementations for interoperability. In 12th USENIX Symposium on Networked Systems Design and Implementation (NSDI).
[9] Adi Sosnovich, Orna Grumberg, and Gabi Nakibly. 2013. Finding Security Vulnerabilities in a Network Protocol Using Parameterized Systems. In 25th International Conference on Computer Aided Verification (CAV).
[10] Adi Sosnovich, Orna Grumberg, and Gabi Nakibly. 2017. Formal Black-Box Analysis of Routing Protocol Implementations. CoRR abs/1709.08096 (arXiv:1709.08096)
[11] Earl Zmijewski. Reckless Driving on the Internet. https://blogs.oracle.com/internetintelligence/reckless-driving-on-the-internet.