Path Validation Problem Statement and a Possible Proof-of-Transit Solution

draft-liu-path-validation-problem-statement-00

SECDISPATCH Meeting @ IETF 118, November 2023

Peter (Chunchi) Liu

Huawei

Contents

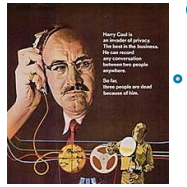
- Why do we care about Path Validation? What is Path Validation?
- Use Cases
- Our Proof-of-Transit solution based on **Vector Commitments**
- **Addressing feedbacks from past engagements**
- Call for collaboration

James bond is making a phone call from overseas
He really wants a secured line.

Is this line secure enough?

I have a top secret to tell you, but someone might be listening.
I would prefer a dedicated line!

Voice over IP network



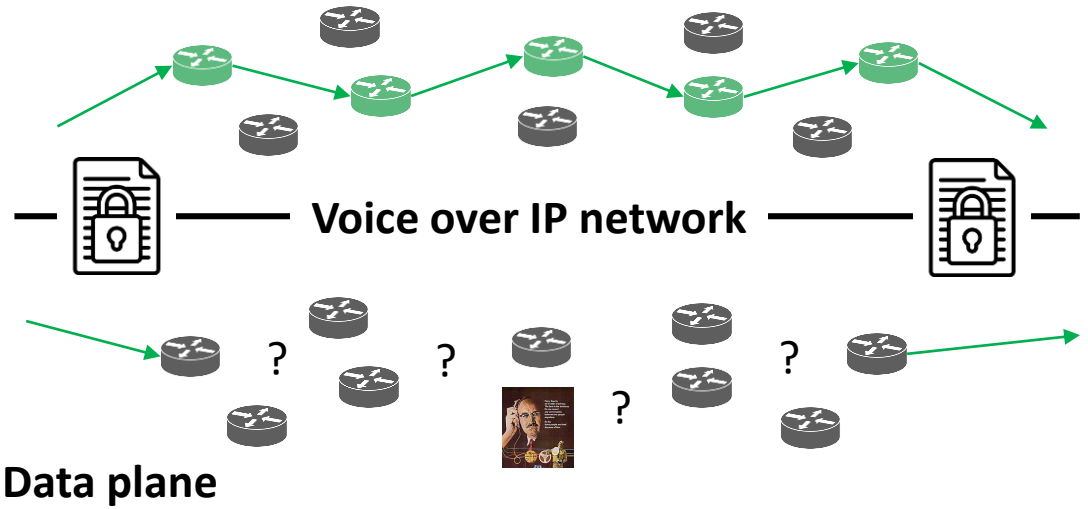
I can do some cryptanalysis...

Don't worry bond, all your traffic are *encrypted.*

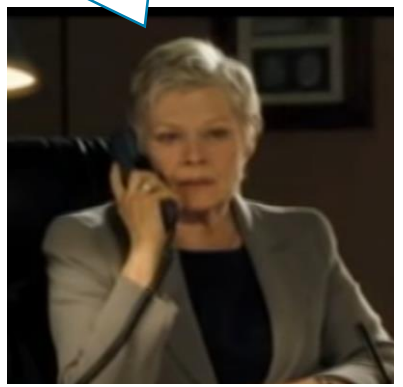
James bond is making a phone call from overseas
He really wants a secured line.

Great... But what do you mean ***should***?

- You need to validate your path, just like validating your claim.



- Here, I prepared a dedicated line for you
- Your connection ***should*** only transit on top of these trusted routers only.



- It means ***I can only plan these path*** in the control plane.
- But I don't know whether or not the planned path ***was actually taken*** in the data plane!

Why do we care about path validation? Because it helps routing security

Three-step recipe to secure routing

1. • Secure path selection and propagation
2. • Secure router execution
3. • Secure Proof of Transit

Reference correctness

Execution correctness

(indirectly implies) **(as-is)**

Final result correctness

Routing integrity **(as-is)**

gap

Forwarding integrity **(to-be)**

Common routing attacks

Not totally solving it, but it can be a step forward

- **Routing Hijack, Route Injection, Route Leak**
- **Denial of Service**
- **Router Misconfiguration (error)**

To **directly** fill this gap, we need **proof-of-transit** mechanisms.

What is path validation? What's its relationship with Proof-of-Transit?

- Path Validation:
  - **Old** Interpretation:
    - Validating the planned path is a trusted, authorized path.
    - **Control plane** path validation, **before** forwarding.
      - Mostly used in BGP context, validate AS-path.
  - **New** Interpretation:
    - Validating what paths a packet has actually traversed.
    - **Data plane** path validation, **after** forwarding.
      - Mostly used in research papers.

Disambiguates into → Proof of Transit

We believe path validation **scope** = **old** + **new** = Routing Integrity + Proof of Transit = Forwarding Integrity

- Path validation should include proof of transit.

We are discussing the path validation problem, but proposing just a proof-of-transit solution.

Existing Proof-of-Transit-like works

1. Telemetry: IOAM/IFIT/Path tracing

2. Proof of transit

3. Path reconstruction
   - Reconstruct forwarding path by collecting router forwarding configuration data

The common blocker is a general Proof of Transit solution!

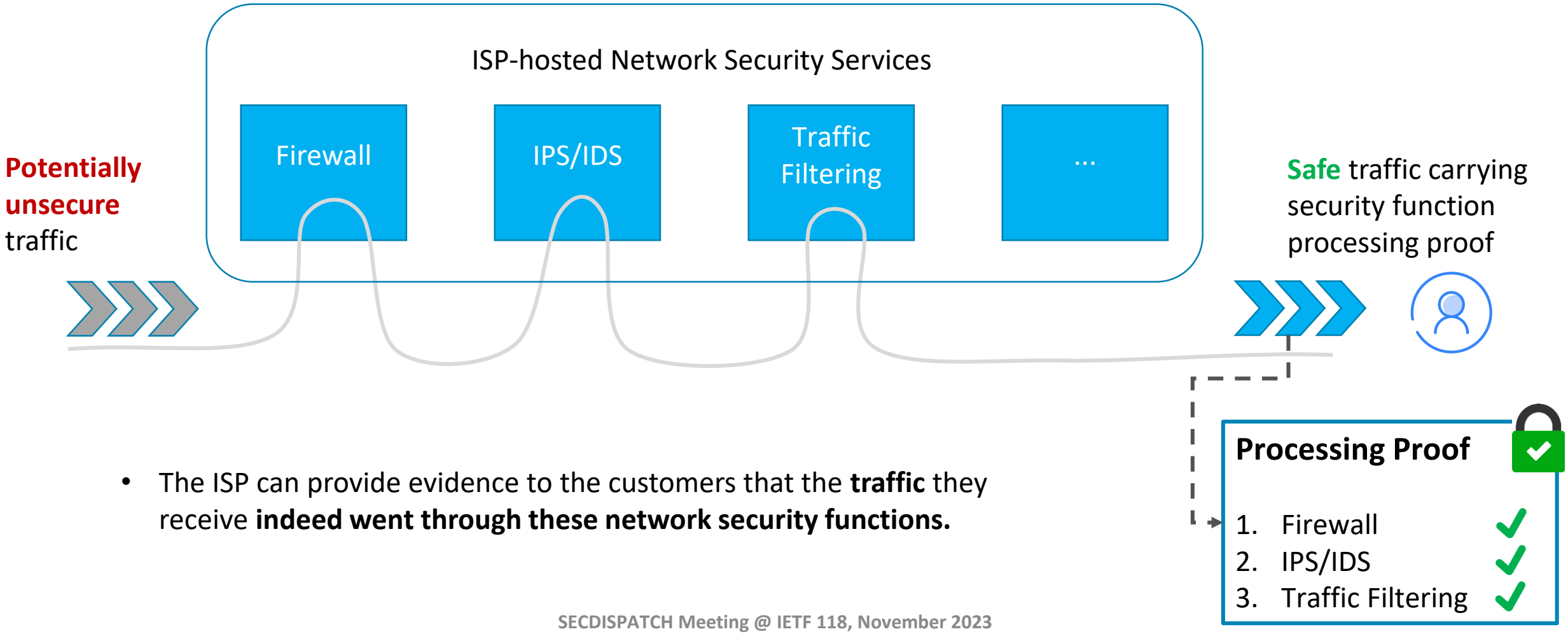
The Good	The Bad
• Allow **underlay** network data telemetry	• **Not applicable to virtual paths** composed of network functions • **Need secure POT** as building block
• **Works for both** for **virtual paths** like SFC and **underlay path**.	• Could pose computational and packet **overhead.** • Inability to perceive **stealth nodes**.
• No data plane modification	• **Indirect way** of verifying forwarding outcome, **inferior than secure POT.** • Need admin access to all routers • No drafts, just research papers

Generalizing the concept of “path”

- Underlay path consists of physical devices
- Virtual path consists of virtual functions

- Complies with [RFC9473], *Vocabulary of Path Properties*, a product of panrg

Core Use Case: Proof of SFC processing



- The ISP can provide evidence to the customers that the **traffic** they receive **indeed went through these network security functions.**

Use Cases

Benefitting Techniques	Value-add of path validation	Related Drafts
• Service Function Chaining/WIMSE	• **Proof of Virtual Function Processing** • Proof of API/Microservices/Container Processing	draft-ietf-sfc-proof-of-transit
• Segment Routing/MPLS	**More accurate path tracing/logging/marking** • Transitive transit proof	RFC 9343, draft-filsfils-spring-path-tracing draft-filsfils-spring-path-tracing-srmpls
• IOAM/IFIT	**More accurate telemetry** • In-situ or individual packet	RFC 9197, RFC9378, RFC9452 draft-song-opsawg-ifit-framework
• Ingress Filtering	• **Augment uRPF check** by executing an actual path backward traversal, not just a FIB lookup, reduce false negative rate. • Or filter packets by checking the transit proof it carries.	RFC3704, RFC8704, RFC5635 draft-xu-ipsecme-risav
• Policy-based Routing	**Compliance check**: is policy correctly enforced at every router?	RFC1104, RFC9067
• Multipath (ECMP/TE)	Know which path it actually took among all valid paths, when one path went bad, **quickly locate the problematic path** and switch to other paths.	RFC6754
• ALTO	**Add a trust metric** using the result of path validation to ALTO path selection (Trust-enhanced networking).	RFC7286
• RATS	Use path validation result to **verify and attest a path**, instead of attesting just one device.	RFC9334

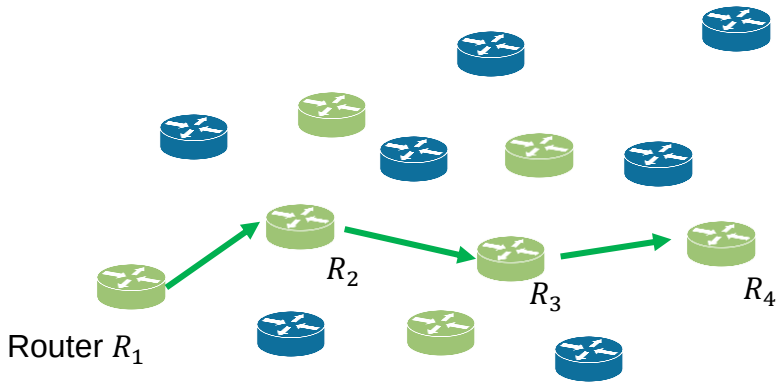
What are the common blockers?

- Proof of Transit mechanism!

A Graphical Overview of the VC-based Proof-of-Transit Solution

✓ Security

- **Position-binding property**: Transit proof P_i successfully passes verification *iff* it was created by the **right node** n_i at the **right position** i **as previously committed**

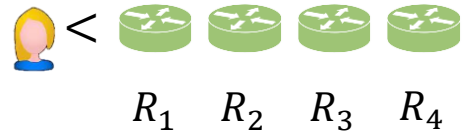


✓ Advantages

- **Efficient:** Proof creation and verification takes **O(1) time**
- **Succinct:** Transit proof and commitment is **O(1) size**
- **Batch-proof** friendly (same efficiency)

Stage 1: Compute Reference Value

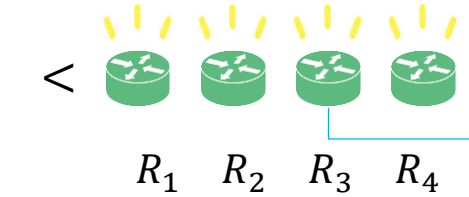
Network Controller



Output 1 **Commitment** → C

- **Controller** selects a path
- Computes a **commitment**

Stage 2: Generate Transit Proof



Output 2 **Transit Proof** → P_i

- **Router** R_i forwards data
- Computes his **transit proof** P_i

Stage 3: Verification

$$Verify(C, P_i) =^{?} 1$$

- **Observer** verifies P_i against C to check if it was the correct router in correct position.

Addressing Feedback from Prior Engagement

	IETF 117, OPSEC, Qs	As
1	How much is the **computing cost** in the forwarding plane?	1-2ms of computing transit proof per router, 24 Bytes of cost per packet. Demoed in the hackathon.
2	You should have control over the exit router to prevent sending the packet to a malicious router before a forwarding mistake was done.	This is right. We contracted the scope our proposed solution to a POT mechanism, in which case this problem will be out of scope.
3	You can design a Proof of Transit solution, but **this is not a Proof of NON Transit solution.** So your core value proposition should not be "preventing traffic to be diverted".	Yes, what we do is proof of transit, not proof of NON transit. The core use case would be inclusion proofs like SFC proof, not non-inclusion proofs. Also, large quantity sampling of POT can be a probabilistic alternative to PONT.
4	How do you prevent transparent tunnel and data out-of-band copy attacks that may cause traffic theft and diversion?	• Same as above. • But combination of POT and trusted routers (attested or proprietary routers) may be secure enough.

What are the out-of-scopes?

- **Illegal data copy:** Data obtained by a router is illegally copied by its owner and sent elsewhere.
  - Data is intangible in nature. This is a data watermark problem.

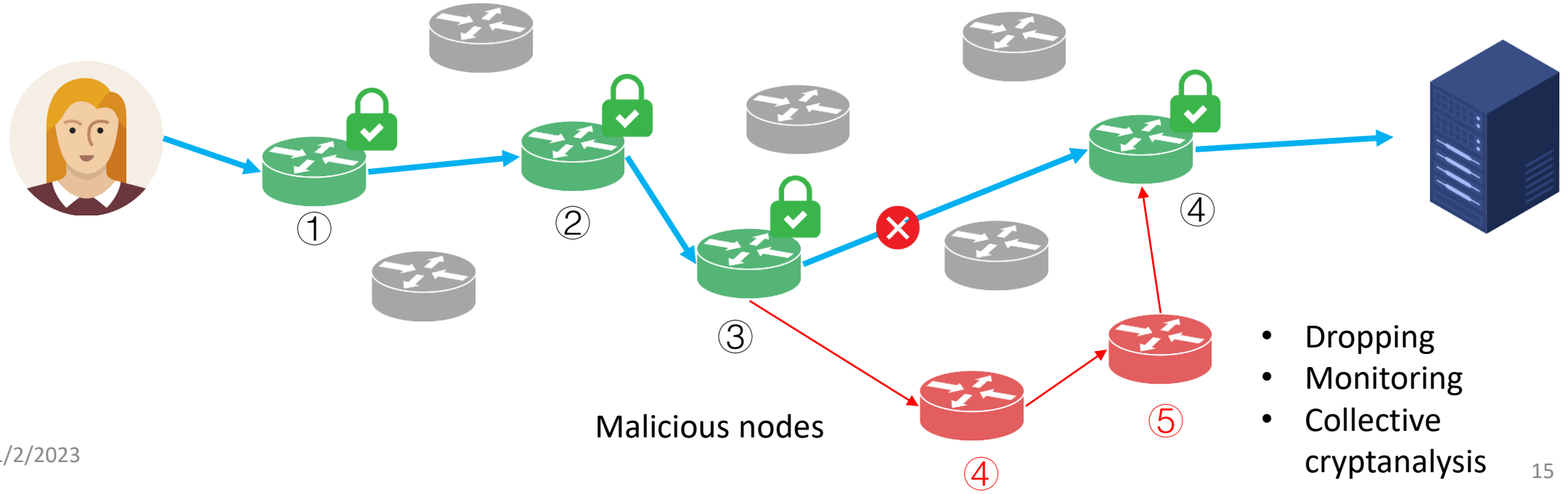
- **Stealth nodes:** inferior nodes not perceivable in the current layer
  - Layered design of Internet purposely make inferior nodes not perceivable. It does not make sense and violates layered design principle trying to perceive stealth nodes. To the very least, it is a different problem.
  - Stealth nodes in most of the times are not significant security threats. They are just either old or computationally weak.
  - One step ahead is better than no progress.

Addressing Feedback from Prior Engagement

	IETF 117, OPSEC, Qs	As
5	This is not compatible to the current stateless destination-based forwarding model of the Internet.	The best starting use case might be keeping it in the limited intradomain, source based routing (e.g. segment routing), where we have some control over the router instead of absolutely none.
6	Then, how will it benefit the current Internet where we don't have control over?	• If intradomain proof of transit is done, we can connect the transit proofs created in each limited domain to form a interdomain transit proof. • It can also be used to improve ingress filtering.
7	A potential use case will be keeping some cryptographically weak traffic inside of a controlled jurisdiction.	This can be developed as a data-sovereignty or geofencing use case, where sensitive traffic is contained within a controlled domain, for legal or business compliance purposes.
8	SFC processing proof could be a social problem rather than a technical problem if you don't believe them.	A cryptographically unforgeable proof may serve as an audit evidence for third party auditors or supervisors.
9	What is the processing throughput? Will you implement this feature in ASIC?	Stress test is under development and ASIC implementation might be expensive to invest. A virtual device that runs in a cloud container might be easier to implement.

The controversial use case

- Alice is having a **confidential** business video meeting or VOIP call.
- She doesn't want any data of this connection be detoured and monitored.
  - With POT and prop routers in a limited domain, we can provide VPN service proofs. But this may not work in an interdomain case where there's no control over routers.



Looking for collaboration

- We look for collaborators together to:
  - ~~POC is done, demoed in 118 hackathon~~
  - SPEC document is writing
  - Joint research

- Come to our side meeting! **Tuesday, 6:30PM – 8:00PM**, Room Karlin 4

Path Validation Problem Statement and a Possible Proof-of-Transit Solution

Thank you! Questions?

draft-liu-path-validation-problem-statement-00

SECDISPATCH Meeting @ IETF 118, November 2023

Peter (Chunchi) Liu

liuchunchi@huawei.com