Introducing Fidelity into Network Emulation

Reproducible Research and Large-scale Experiment Orchestration

Houssam ElBouanani

Context

Fidelity

Refers to the similarity between the testing and production environments

Reproducibility

Refers to the ability for a research and its results to be reproduced

What tools do researchers use to conduct their experiments? How much fidelity and reproducibility do they allow?

Context

Option 1: open physical testbeds

- Realistic environments
- No universal interface
- Costly to deploy
- Only open to researchers
- Reproducibility not guaranteed

slices RI

EdgeNet

Options

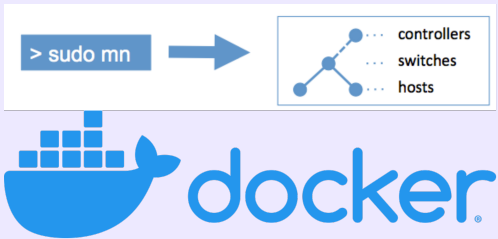
Option 2: network simulators

- Free
- Packageable and shareable results
- Applications must be adapted
- Fidelity not guaranteed

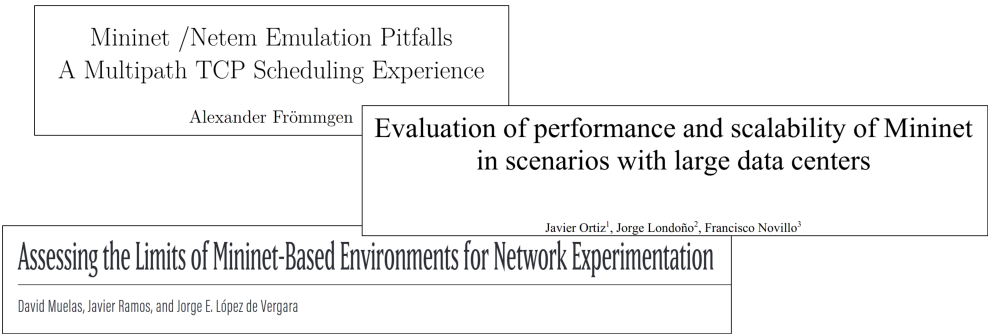
Options

Option 3: network emulators

- Free
- Packageable and shareable results
- No need for special adaptation
- Impacted by the host machine
- Fidelity not guaranteed
- Reproducibility not guaranteed



Problem


Mininet /Netem Emulation Pitfalls
A Multipath TCP Scheduling Experience
Alexander Frömmgen
Evaluation of performance and scalability of Mininet in scenarios with large data centers
Javier Ortiz[1], Jorge Londoño[2], Francisco Novillo[3]
Assessing the Limits of Mininet-Based Environments for Network Experimentation
David Muelas, Javier Ramos, and Jorge E. López de Vergara

State of the art

First solution: isolation and precision monitoring

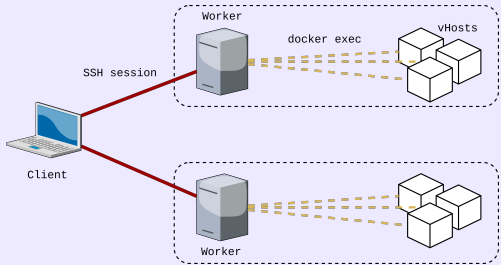
- Isolate nodes
- Enforce link capacity
- Check event timing
- Implement emulation invariants

The former principles are not sufficient and the latter are not operational.

State of the art

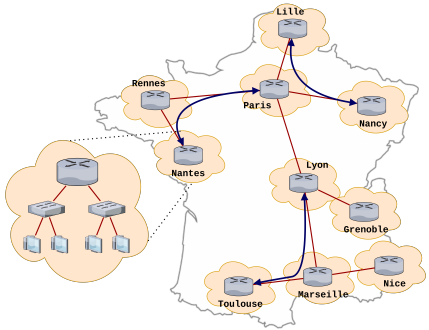
Second solution: distributed network emulation

- Distribute the emulated nodes over a cluster of machines
- Emulated network as overlay

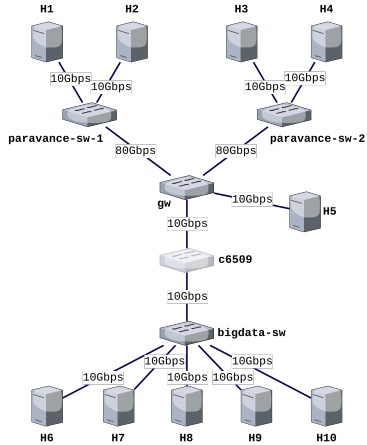


The underlay infrastructure adds a new point of failure to the emulation.

Example

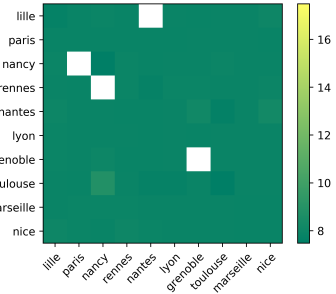


Emulated network

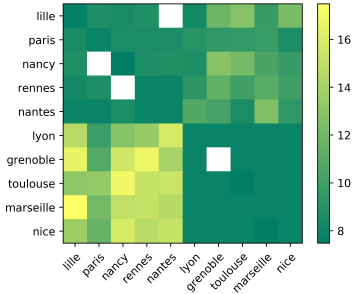


Infrastructure network

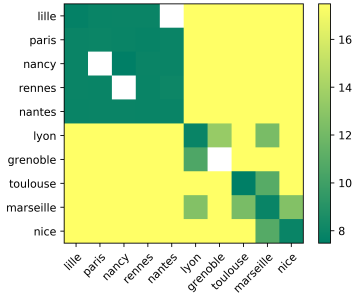
Example



Run #1: 7.90 ± 0.21 seconds.



Run #2: 10.37 ± 3.14 seconds.



Run #3: 38.39 ± 25.80 seconds.

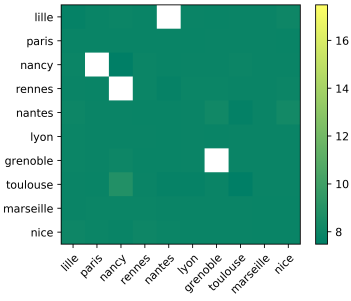
What happened? How to make the output accurate and reproducible?

Outline

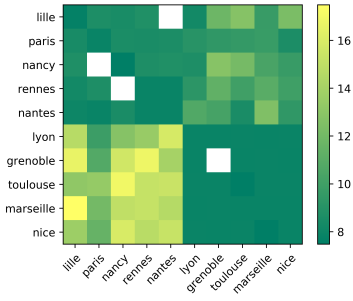
Thesis question

How can we best improve the limitations of network emulation to make it the reproducible and accurate research medium it was designed to be?

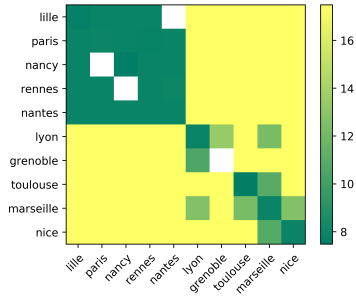
Problem



Run #1: 7.90 ± 0.21 seconds.



Run #2: 10.37 ± 3.14 seconds.



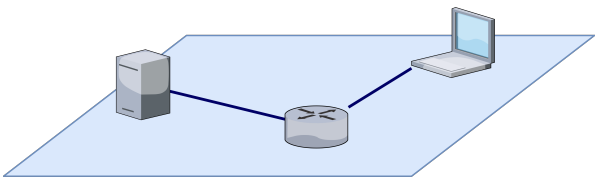
Run #3: 38.39 ± 25.80 seconds.

Is there a correct run? Which is it? How to systematically distinguish *bad* from *good* results?

Emulation fidelity

Emulation fidelity

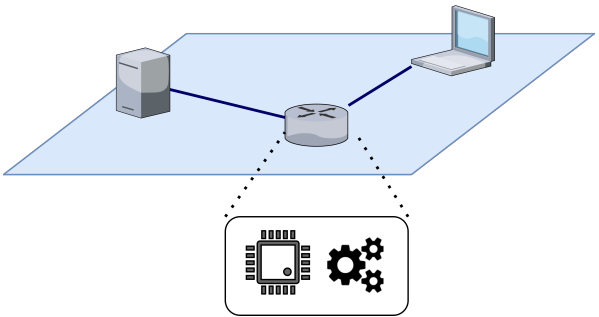
Measure of similarity in design, implementation, and operation between an emulated network and the real network it mimics.



Emulation fidelity

Noumenal fidelity

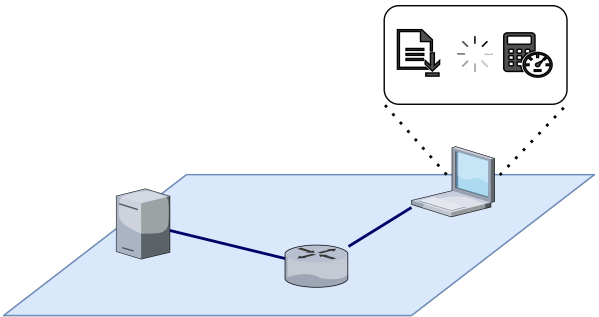
- Not operational



Emulation fidelity

High-level phenomenal fidelity

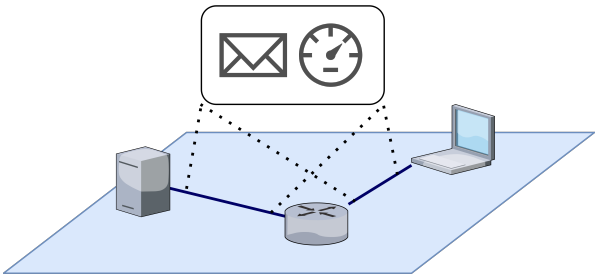
- Operational
- Scenario-dependent
- Analysis required



Emulation fidelity

Low-level phenomenal fidelity

- Operational
- Scenario-agnostic
- No analysis required



Low-level phenomenal emulation fidelity

Bandwidth

Link is accurately emulated if transmitted packets < bandwidth

Packet order

Link is accurately emulated if packets are transmitted in the correct order

Packet loss

Link is accurately emulated if no extra packets are lost

Packet delay

Link is accurately emulated if packets are correctly delayed

Low-level phenomenal emulation fidelity

Packet delay is the finest measure of link-level phenomenal fidelity

- Universal and scenario-agnostic
- Offers insights about other link-level phenomena
- Suitable for fine- and coarse-grained analysis
- Accurate and efficient measurement
- Impacted by network-affecting anomalies
- **Can be compared and evaluated against a reference model**:

$$d(P) = \begin{cases} \frac{|Q(P)|}{B} + \frac{|P|}{B} + \delta, & \text{if packet is not dropped} \\ \infty, & \text{otherwise} \end{cases}$$

Delay-based fidelity monitoring

Fidelity criterion

If throughout the duration of an emulated experiment the deviation of the measured delays from the expected delays of a sample set of packets is too large, then that emulation should be considered incorrect.

Delay-based fidelity monitoring

Caveat 1: *deviation*

- Many statistical metrics
  - Numeric Absolute Error: $NAE(P) = |\hat{d}(P) - d(P)|$
  - Percentage Absolute Error: $PAE(P) = \frac{|\hat{d}(P) - d(P)|}{d(P)}$
- Delay deviation is measured on each packet but the judgment is made on the whole emulation
  - Average delay deviation
  - Quantile delay deviation
  - Sliding window delay deviation

Delay-based fidelity monitoring

Caveat 2: fixing a threshold

- The calculated statistic is compared to a threshold value
  - Small (1 ms, 5%) for strict fidelity constraints
  - Large (10 ms, 100%) for coarse fidelity constraints
- The choice of a specific value is generally not important
  - Deviation tends to grow exponentially when problems arise

Delay-based fidelity monitoring

Caveat 3: packet sampling

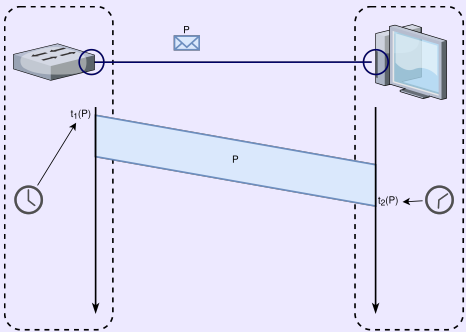
- Considering all packets is neither feasible nor necessary
  - 1500 bytes per packet × 10 links × 1 Gbps per link × 100 seconds ≈ 80 million packets
  - Does not impact performance but practically impossible to store and analyse
- Perform hash-based packet sampling
  - Uniform-random sampling (1/10, 1/100, etc.)
  - Select sampling based on packet size, queue length, link bandwidth, etc.

Delay-based fidelity monitoring

Caveat 4: passive delay measurement

First solution: clock synchronisation

- Machines need to be close
- NTP: long time to converge
- PTP: needs hardware acceleration



One-way passive delay measurement.

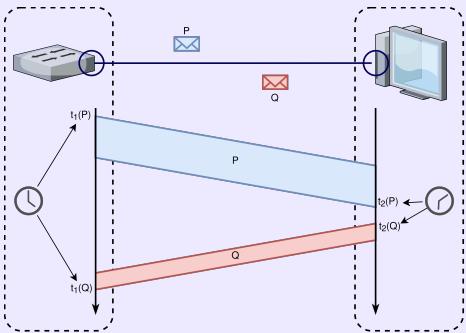
Delay-based fidelity monitoring

Caveat 4: passive delay measurement

Second solution: round-trip delay

- Eliminates clock offset
- For any pair of packets (P, Q)
- The measured RTD needs to be compared to the sum of the pair's individual expected delays:

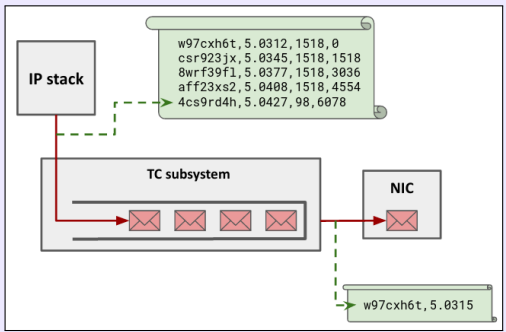
$$\begin{aligned} \hat{RTD}(P,Q) &= (t_1(Q) - t_1(P)) \\ &\quad - (t_2(Q) - t_2(P)), \\ RTD(P,Q) &= d(P) + d(Q) \end{aligned}$$



Round-trip delay measurement.

Implementation

Packet loggers

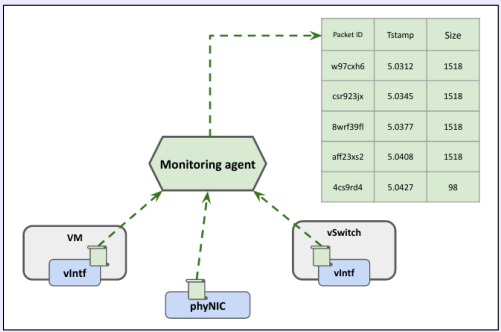


eBPF-based packet interception.

- Lightweight packet interception processes
- Written in eBPF
- Run at each emulated interface
- Log information about packet events

Implementation

Monitoring agents

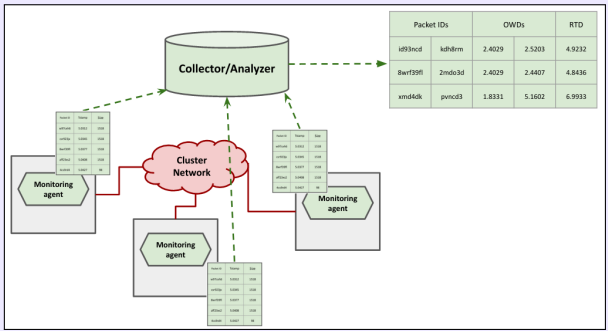


Compilation of packet logs.

- User-level processes
- Run on each machine of the cluster
- Collect log files from the packet loggers
- Compile packet information into tables

Implementation

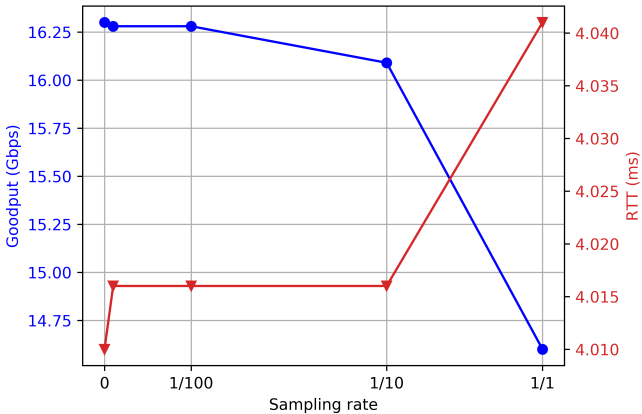
Collector/analyser



Centralised analysis and fidelity assessment.

- Logically unique component
- Collects packet information tables from the monitoring agents
- Compute measured and expected delays
- Assess fidelity according to user-defined strategy

Overhead



Impact of fidelity monitoring on performance.

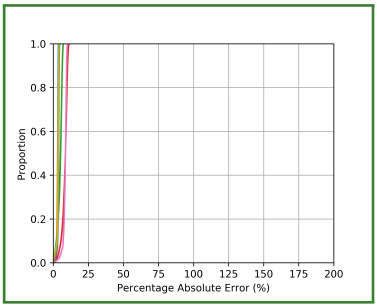
Latency

- 31 microseconds of added latency (to 4 ms)
- $RTT(s) = \frac{RTT(0)}{1-\alpha s}$

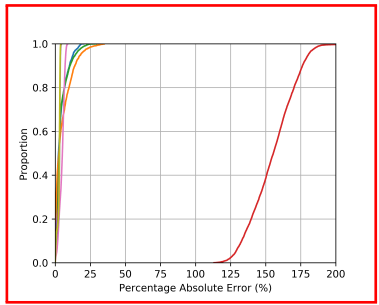
Throughput

- 1.7 Gbps of reduced throughput (from 16.3 Gbps)
- $\theta(s) = \frac{\theta(0)}{1+\beta s}$

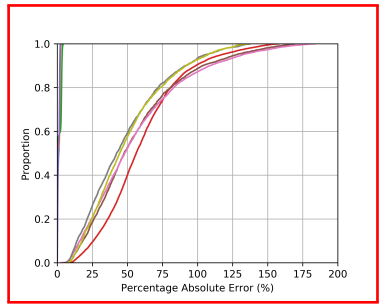
In practice



Run #1: 90th percentile of PAE: 9.42%.

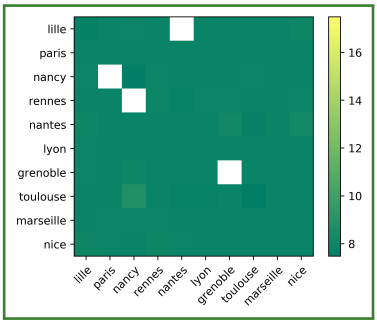


Run #2: 90th percentile of PAE: 156.70%.

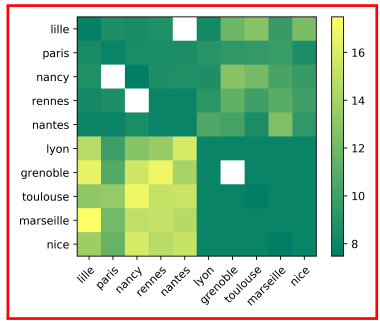


Run #3: 90th percentile of PAE: 84.74%.

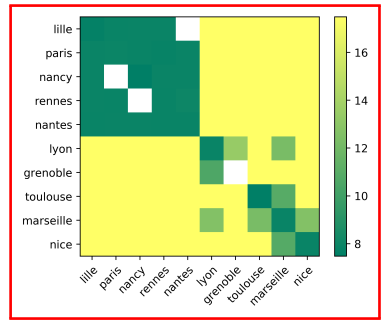
In practice



Run #1: 7.90 ± 0.21 seconds.



Run #2: 10.37 ± 3.14 seconds.



Run #3: 38.39 ± 25.80 seconds.

Takeaways

- Monitoring fidelity is necessary for ensuring quality results
- Phenomenal fidelity is an operational framework for assessing emulation accuracy
- Packet delay is a good metric for measuring fidelity

Contributions

Conference proceedings

- ElBouanani, et al., *Passive delay measurement for fidelity monitoring of distributed network emulation*, In Proceedings of IEEE 20th Mediterranean Communication and Computer Networking Conference (MedComNet 2021).
- ElBouanani, et al., *Delay-based fidelity monitoring of network emulation*, Testbeds for Advanced Systems Implementation and Research (TASIR workshop), IEEE 15th International Conference on Communication Systems and Networks (COMSNETS 2023).

Journal papers

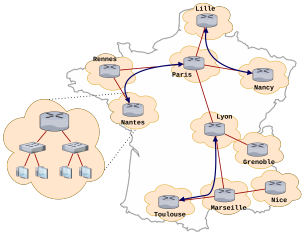
- ElBouanani, et al., *Passive delay measurement for fidelity monitoring of distributed network emulation*, Elsevier Computer Communications Journal (COMCOM), volume 195, pages 40-48.

Outline

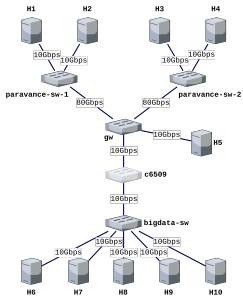
Thesis question

How can we best improve the limitations of network emulation to make it the reproducible and accurate research medium it was designed to be?

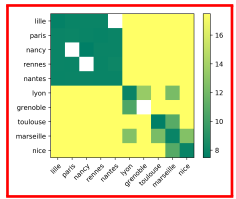
Problem

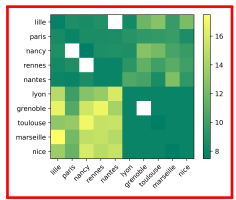


Emulated network topology.



Infrastructure network topology.





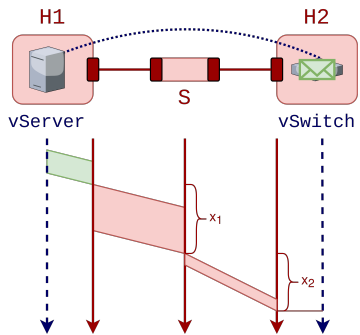
Why did these two runs produce inaccurate results?

Solution

Step 1	Step 2	Step 3
Measure overlay-(emulated network-level) delays	**Convert overlay measurements into underlay measurements**	Infer link failure (congestion/overloading) from delay measurements

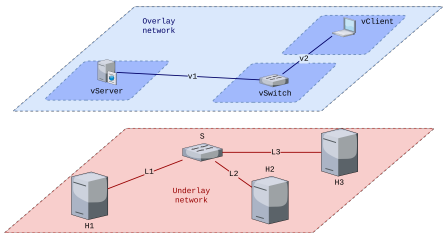
Delay tomography

- Packets sent and received on the overlay link cross two underlay links
- The delay added to their expected emulated delays is the physical delay
- Is it possible to extract physical underlay delays from the measured delay differences?
  - Individually: no



Delay tomography

- Packets sent and received on the overlay link cross two underlay links
- The delay added to their expected emulated delays is the physical delay
- Is it possible to extract physical underlay delays from the measured delay differences?
  - Individually: no
  - On large samples of packets crossing different underlay paths: yes (with statistical degree of confidence)



Delay tomography

In this example:

$$\begin{cases} x_1 + x_2 = \overline{\epsilon}(v_1) := b_1 \\ x_2 + x_3 = \overline{\epsilon}(v_2) := b_2 \end{cases}$$

In general, for collections (L_i) of underlay links and (v_j) of overlay links:

$$\begin{cases} \sum_{L_i} a_{i,j} \cdot x_i = b_j, \ \forall v_j \end{cases}$$

$$\mathbf{A} \cdot \mathbf{X} = \mathbf{b}$$

Obstacles

Problem 1: inaccuracy

- Inaccurate measurements give inaccurate inferences
  - May also make the problem unsolvable
- The proposed passive delay measurement framework is accurate by design
- Solution: clean the measurements beforehand
  - What is the lowest error value to be tolerated to make the problem solvable?

$$\begin{aligned} &\text{minimise}_{\mathbf{X},\varepsilon} && \|\varepsilon\|^2 \\ &\text{subject to} && \mathbf{A} \cdot \mathbf{X} = \mathbf{b} + \varepsilon \\ & && \mathbf{X} \geq 0\ . \end{aligned}$$

Obstacles

Problem 2: identifiability

- Once measurements are cleaned the problem may have infinite solutions:

$$\begin{cases} x_1 + x_2 = b_1 \\ x_2 + x_3 = b_2 \end{cases}$$

- Solution: add an objective function to reduce the set of solutions

Solution

Algorithm

1. Compute $\mathbf{b}$
2. Solve for ε^* in

$$\begin{aligned} \text{minimise}_{\mathbf{X},\varepsilon} \quad & \|\varepsilon\|^2 \\ \text{subject to} \quad & \mathbf{A}\cdot\mathbf{X} = \mathbf{b} + \varepsilon \\ & \mathbf{X} \geq 0\,. \end{aligned}$$

3. Solve for $\mathbf{X}^*$ in

$$\begin{aligned} \text{minimise}_{\mathbf{X}} \quad & f(\mathbf{X}) \\ \text{subject to} \quad & \mathbf{A}\cdot\mathbf{X} = \mathbf{b} + \varepsilon^* \\ & \mathbf{X} \geq 0\,. \end{aligned}$$

Solution

Heuristic 1: Occam's razor

- Assumption: emulation failure is caused by a small set of faulty links
- Algebraicly: minimise the number of overloaded links (or total delay), i.e.

$$f(x_1, ..., x_n) = \sum_i x_i$$

- Pros
  - Works well when the assumption holds

Solution

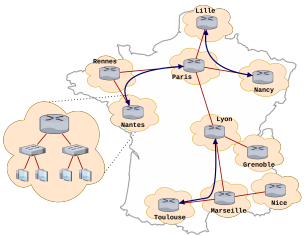
Heuristic 2: dynamic adaptive Occam's razor

- All underlay links should not be given equal probability of failure
- Use static information when available (link bandwidth, link type, etc.)
- Algebraicly: minimise a weighted sum of underlay delays, i.e.

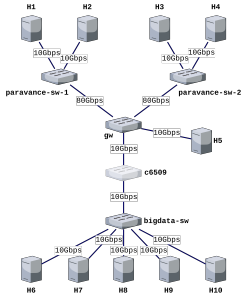
$$f(x_1, ..., x_n) = \sum_i \alpha_i x_i, \text{ (e.g., } \alpha_i := -\log \mathbb{P}[L_i \text{ is faulty}])$$

- Pros
  - Can discriminate between otherwise undistinguishable links

Evaluation



Emulated network topology.



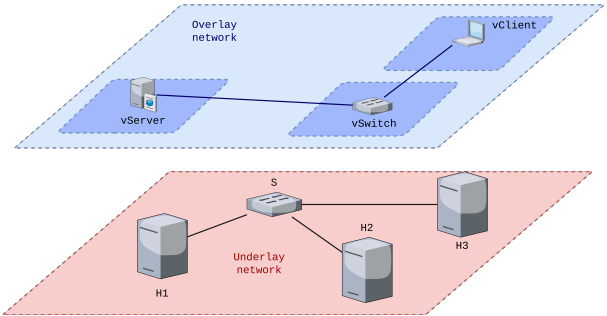
Infrastructure network topology.

AS	Host
Lille	H1
Nancy	H2
Rennes	H3
Nantes	H4
Paris	H5
Lyon	H6
Grenoble	H7
Toulouse	H8
Marseille	H9
Nice	H10

Evaluation

Simulation flow

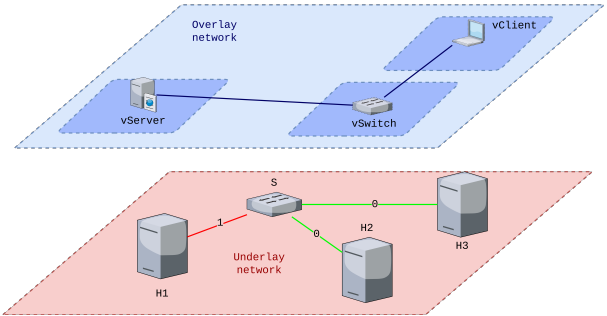
For each link overload combination $\mathbf{u}$,



Evaluation

Simulation flow

For each link overload combination $\mathbf{u}$,

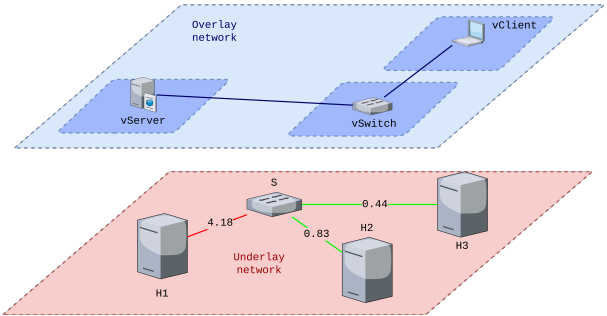


Evaluation

Simulation flow

For each link overload combination $\mathbf{u}$,

1. Simulate underlay delays $\mathbf{X}$ ($x_i > 1$ ms iff $u_i = 1$)

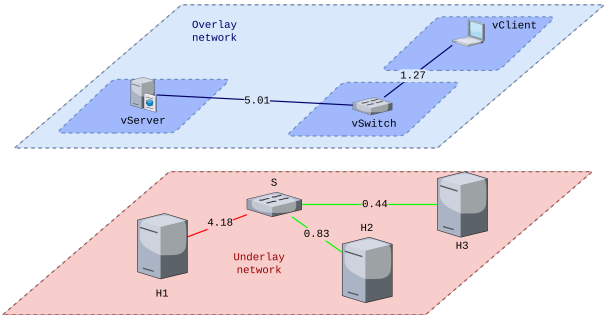


Evaluation

Simulation flow

For each link overload combination $\mathbf{u}$,

1. Simulate underlay delays $\mathbf{X}$ ($x_i > 1$ ms iif $u_i = 1$)
2. Compute overlay delays $\mathbf{b}$

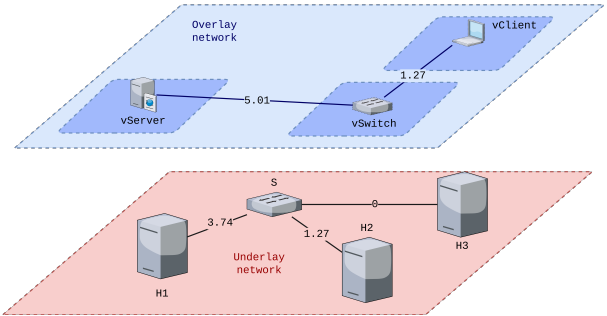


Evaluation

Simulation flow

For each link overload combination $\mathbf{u}$,

1. Simulate underlay delays $\mathbf{X}$ ($x_i > 1$ ms iif $u_i = 1$)
2. Compute overlay delays $\mathbf{b}$
3. Estimate underlay delays $\widehat{\mathbf{X}}$ using the algorithm

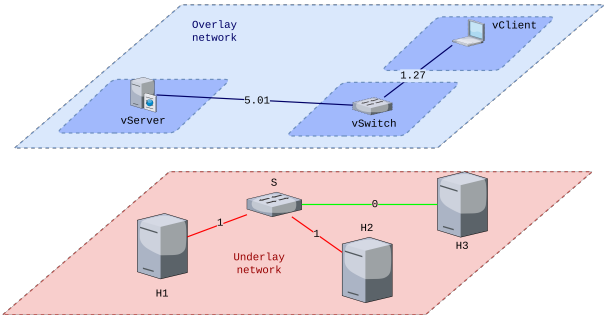


Evaluation

Simulation flow

For each link overload combination $\mathbf{u}$,

1. Simulate underlay delays $\mathbf{X}$ ($x_i > 1$ ms iff $u_i = 1$)
2. Compute overlay delays $\mathbf{b}$
3. Estimate underlay delays $\widehat{\mathbf{X}}$ using the algorithm
4. Infer which links are overloaded $\widehat{\mathbf{u}}$



Evaluation

Simulation flow

For each link overload combination $\mathbf{u}$,

1. Simulate underlay delays $\mathbf{X}$ ($x_i > 1$ ms iif $u_i = 1$)
2. Compute overlay delays $\mathbf{b}$
3. Estimate underlay delays $\widehat{\mathbf{X}}$ using the algorithm
4. Infer which links are overloaded $\widehat{\mathbf{u}}$
5. Compare estimation $\widehat{\mathbf{u}}$ to ground truth $\mathbf{u}$

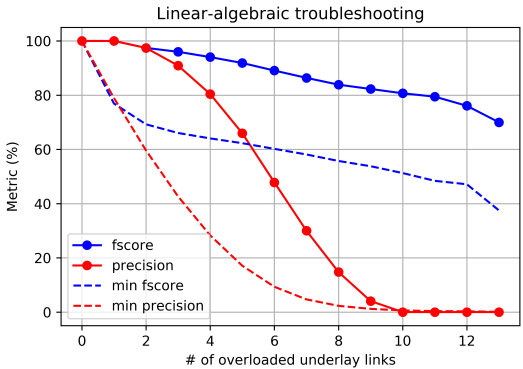
Simulation parameters

- 13 underlay links
- 9 overlay links

Performance indicators

- Precision
- F_1-score

Evaluation



Emulation results.

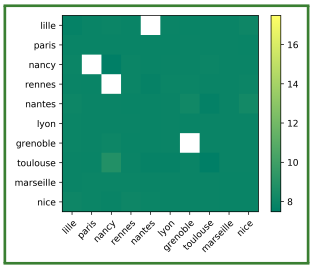
Minimum

- Average precision: 10.77%
- Average F_1-score: 59.05%

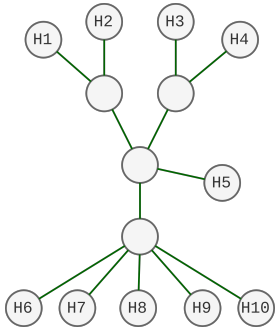
Heuristic

- Average precision: 40.64%
- Average F_1-score: 87.91%

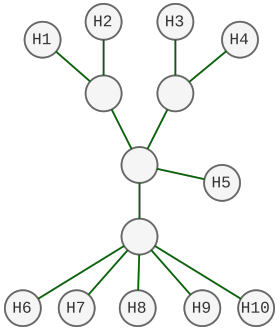
In practice



Run #1

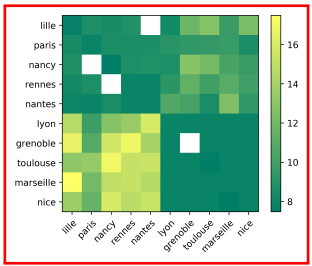


Ground-truth

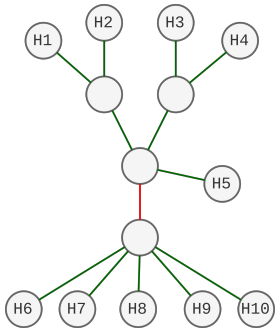


Estimation

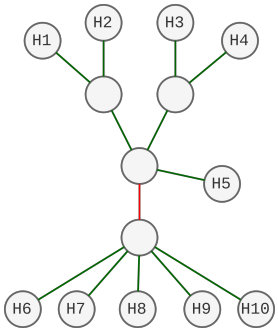
In practice



Run #2

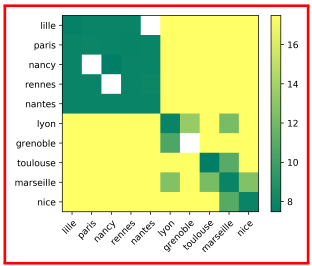


Ground-truth

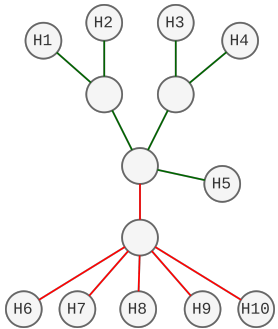


Estimation

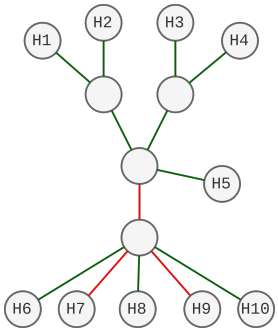
In practice



Run #3

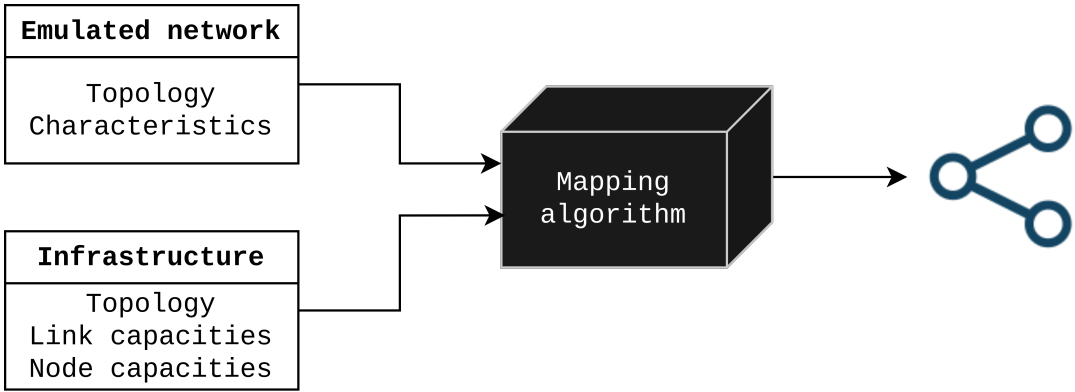


Ground-truth

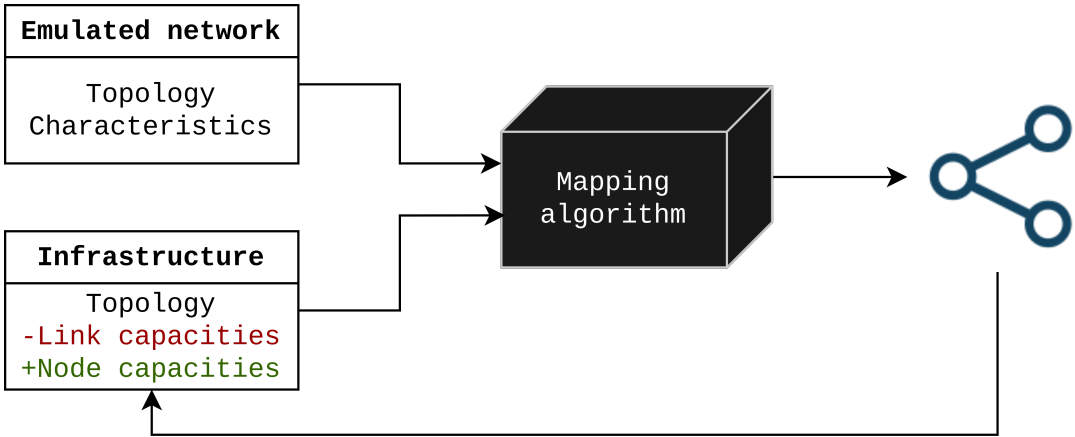


Estimation

Remapping


Emulated network
Topology
Characteristics
Infrastructure
Topology
Link capacities
Node capacities
Mapping
algorithm

Remapping


Emulated network
Topology
Characteristics
Infrastructure
Topology
-Link capacities
+Node capacities
Mapping
algorithm

Takeaways

- Troubleshooting emulation failures is necessary to ensure reproducibility
- Some failures can be troubleshot using overlay-level delay measurements
- General assumptions and/or prior knowledge about the network structure are necessary for good root-cause analysis
- Remap the emulated network onto the infrastructure to avoid problematic links

Contributions

Demo papers

- ElBouanani, et al., *Fidelity-aware distributed network emulation*, In proceedings of IEEE Conference on Standards for Communications and Networking (CSCN 2022).

Conference proceedings

- ElBouanani, et al., *Troubleshooting distributed network emulation*, In Proceedings of 23th IEEE Conference on Innovation in Clouds, Internet, and Networks (ICIN 2023).

Journal papers

- ElBouanani, et al., *Troubleshooting distributed network emulation*, **to be submitted (fast track) to** Springer Annals of Telecommunications.
- ElBouanani, et al., *Fidelity-aware large-scale distributed network emulation*, **to be submitted to** Elsevier Computer Networks (COMNET).

Conclusion

Summary

- Distributed network emulation is an efficient approach to network research and engineering
- It suffers from issues of accuracy that limit reproducibility
  - Lack of realism can be mitigated through fidelity monitoring
  - Lack of reproducibility can be mitigated through infrastructure network troubleshooting and intelligent remapping
- We proposed **Hifinet: a large-scale distributed network emulator powered with the fidelity monitoring and troubleshooting framework**

Limitations

- Physical limits on time measurement
- Compromises between efficiency and accuracy
- Choice of parameters
- Identifiability of network structures

The End