deTector: a Topology-Aware Monitoring System for Data Center Networks

Yanghua Peng[1], Ji Yang[2], Chuan Wu[1], Chuanxiong Guo[3], Chengchen Hu[2], Zongpeng Li[4]

[1]The University of Hong Kong, [2]Xi'an Jiaotong University

[3]Microsoft Research, [4]University of Calgary

Data Center Network Monitoring

- Failures are the norm rather than exception
  - Typical first year for a new cluster (Jeff Dean, Google)
    - 8 network maintenances
    - 15 router reloads/failures
    - 26 rack failures/moves
    - Dozens of blips of DNS
    - 1000 individual machine failures
- SLA violation (99.999%)
  - Packet losses and latency spikes
  - Difficult to troubleshoot (up to days to fix the issues)

Data Center Network Monitoring

- Failures are the norm rather than exception
  - Typical first year for a new cluster (Jeff Dean, Google)
    - 8 network maintenances
    - 15 router reloads/failures
    - 26 rack failures/moves
    - Dozens of blips of
    - 1000 individual machine failures

A network monitoring system for rapid failure recovery

- SLA violation (99.999%)
  - Packet losses and latency spikes
  - Difficult to troubleshoot (up to days to fix the issues)

Challenges

- Clean failures
  - Easy to detect, e.g., server down.
- Gray failures
  - Not reported by the device (SNMP/CLI)
- Low-rate losses
  - Covered up by ECMP
- Transient failures
  - Difficult to play back and pinpoint

Challenges

- Clean failures
  - Easy to detect, e.g., server down.
- Gray failures
  - Not reported by the device (SNMP/CLI)
- Low-rate losses
  - Covered up by ECMP

Exhaustive detection

- Transient failures
  - Difficult to play back and pinpoint

Existing Solutions

- Existing systems
  - Passive: CLI/SNMP
  - Active: Pingmesh, NetNORAD

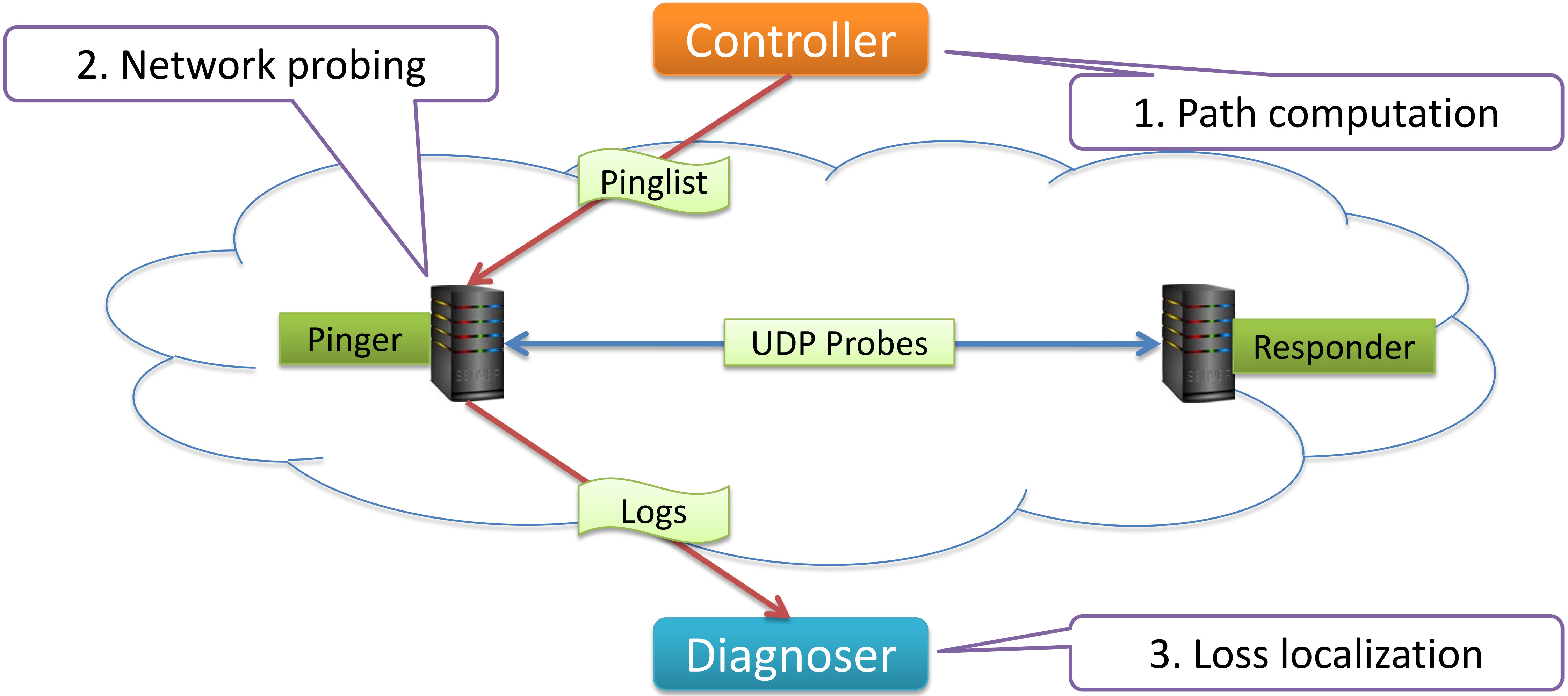
- Limitations
  - Fail to detect at least one type of losses
  - High overhead
  - Can not pinpoint failures without other tools (e.g., tracert)

Existing Solutions

- Existing systems
  - Passive: CLI/SNMP
  - Active: Pingmesh, NetNORAD
- Limitations
  - Fail to detect at least one type of losses
  - High overhead
  - Can not pinpoint failures without other tools (e.g., tracert)

Can we design a better network monitoring system by exploiting network topology?

deTector in One Slide



Phase I: Path Computation

Path Selection Problem

	Link 1	Link 2	Link 3
Path 1	1	1	0
Path 2	1	0	1
Path 3	0	0	1

- Given a routing matrix, select probing paths to send probes:
  - path number minimizing
  - α-coverage
  - β-identifiability

α-coverage

Each link is covered by at least α paths

	Link 1	Link 2	Link 3
Path 1	1	1	0
Path 2	1	0	1
Path 3	0	0	1

- Ensure even and enough path coverage of each link
- w[link]: track the number of paths through it.
- If w[link] > α, then the link has enough paths through it.

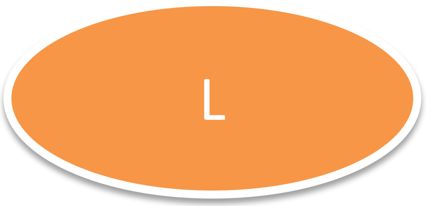
β-identifiability

	1,2	1	2
	Link 1	Link 2	Link 3
Path 1	1	1	0
Path 2	1	0	1
Path 3	0	0	1

- Any β failed links can be identified correctly
- Probe matrix: path1 + path2

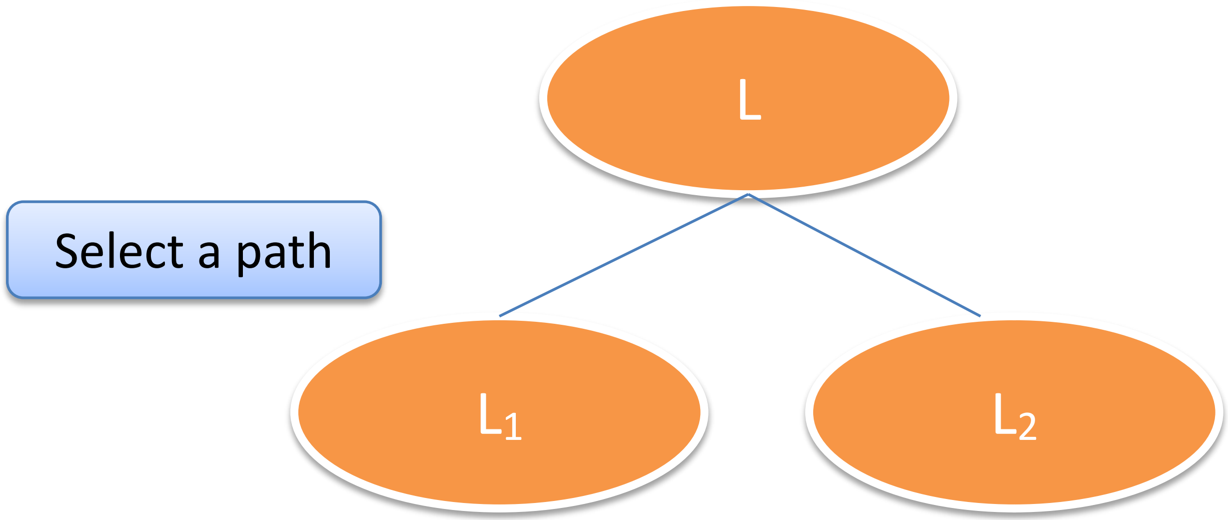
  1-identifiability but not 2-identifiability

Algorithm for 1-identifiability



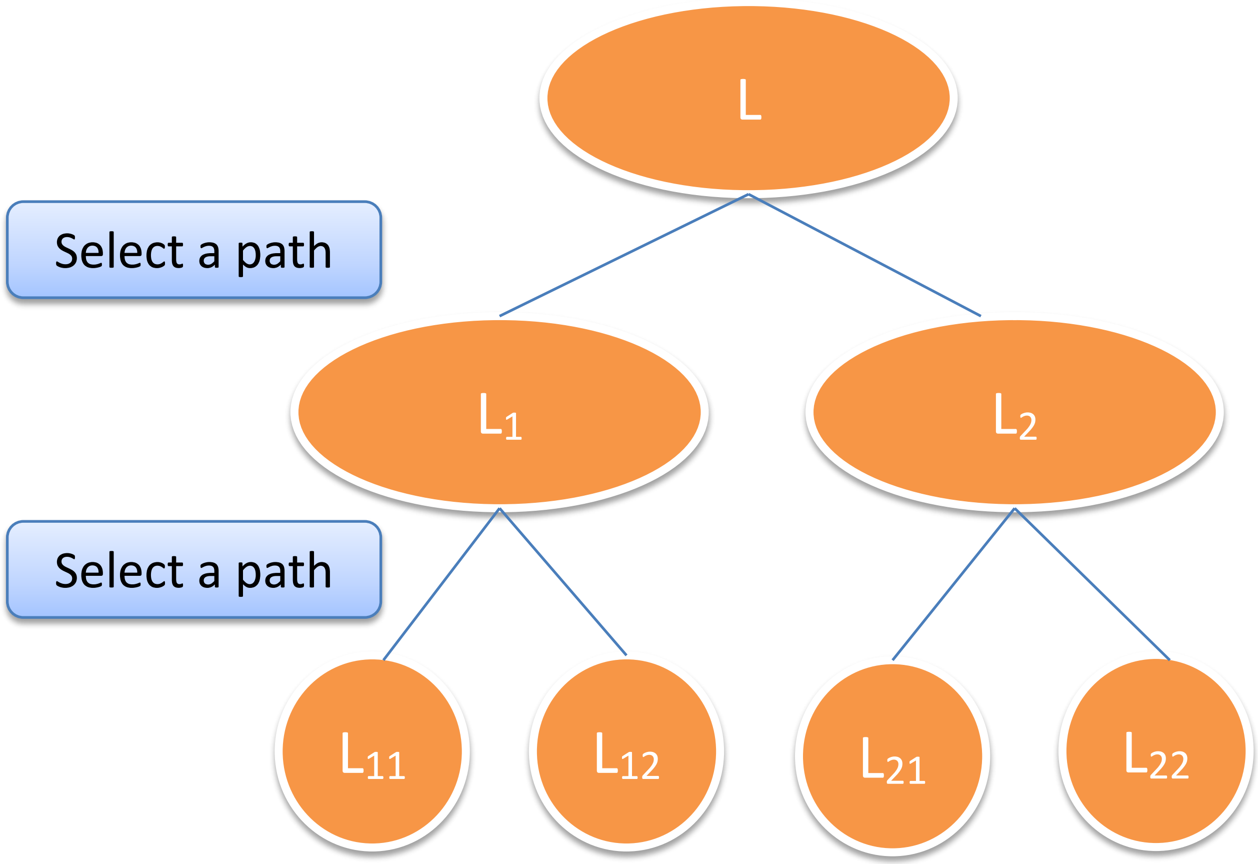
- Select the minimum number of paths so that each link has a different set of probe paths.
- Greedily select the path which splits the largest number of link sets in each iteration.

Algorithm for 1-identifiability



- Select the minimum number of paths so that each link has a different set of probe paths.
- Greedily select the path which splits the largest number of link sets in each iteration.

Algorithm for 1-identifiability



- Select the minimum number of paths so that each link has a different set of probe paths.
- Greedily select the path which splits the largest number of link sets in each iteration.

1-identifiability => β-identifiability

	Link 1	Link 2	Link 3
Path 1	1	1	0
Path 2	1	0	1
Path 3	0	0	1

Physical links

- Extend routing matrix with virtual links.
- If a virtual link is down, we say its corresponding physical links have failed.

1-identifiability => β-identifiability

	Link 1	Link 2	Link 3	Link 12	Link 13	Link 23
Path 1	1	1	0	1	1	1
Path 2	1	0	1	1	1	1
Path 3	0	0	1	0	1	1

Physical links OR Virtual links

- Extend routing matrix with virtual links.
- If a virtual link is down, we say its corresponding physical links have failed.

Probe Matrix Construction (PMC) Algorithm

- Extend the routing matrix with virtual links
- Define a score for each path

$$score(path) = \sum_{link \in path} w[link] - \#\text{ of link sets on } path$$

Quantify coverage

Quantify identifiability

- Select a path with minimal score in each iteration
- Stop when achieving α-coverage and β-identifiability

PMC Algorithm

- Achieve 63% approximation ratio.
- Time complexity $O(n^2)$ where n is the number of paths.
- A Fattree(64) DCN has more than 2^{32} paths, running time > 24 hours

PMC Algorithm

- Achieve 63% approximation ratio.
- Time complexity $O(n^2)$ where n is the number of paths.
- A Fattree(64) DCN has more than 2^{32} paths, running time > 24 hours 

PMC Algorithm

- Achieve 63% approximation ratio.
- Time complexity $O(n^2)$ where n is the number of paths.
- A Fattree(64) DCN has more than 2^{32} paths, running time > 24 hours

Optimizations for speedup

Optimization I: Routing Matrix Decomposition

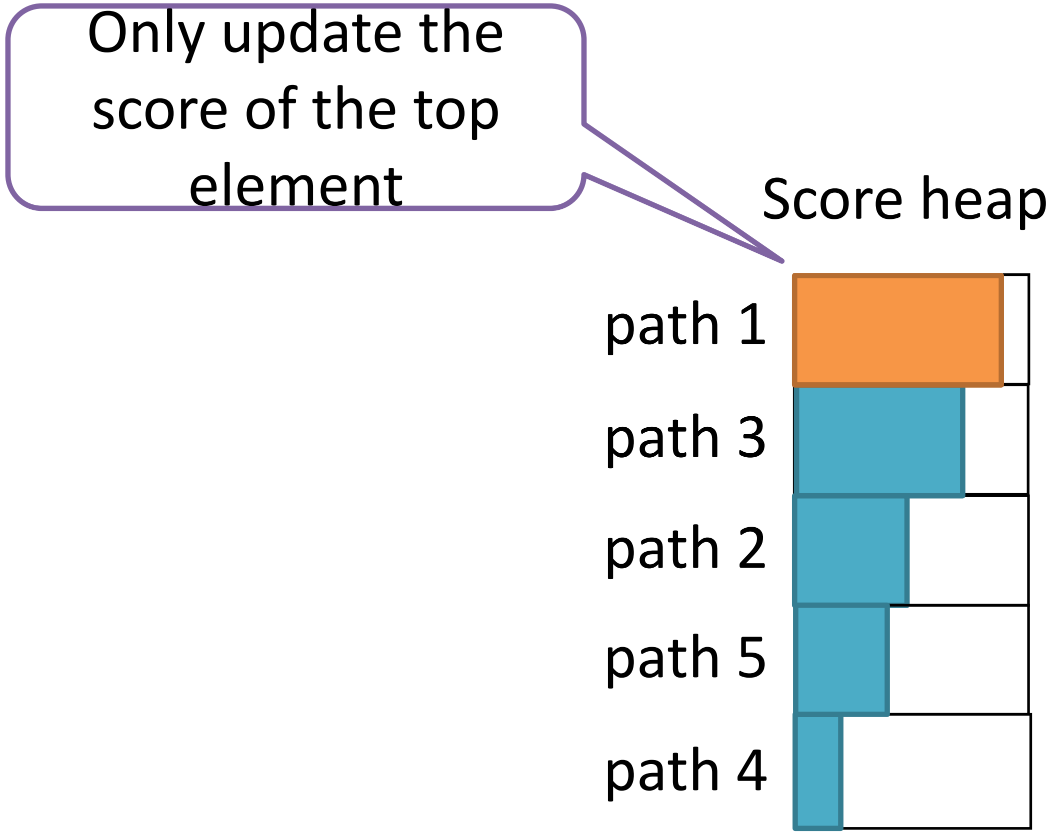
1	1	0	0	0
1	0	1	0	0
0	1	1	0	0
0	0	0	1	0
0	0	0	1	1

Share no links and paths

1	1	0
1	0	1
0	1	1

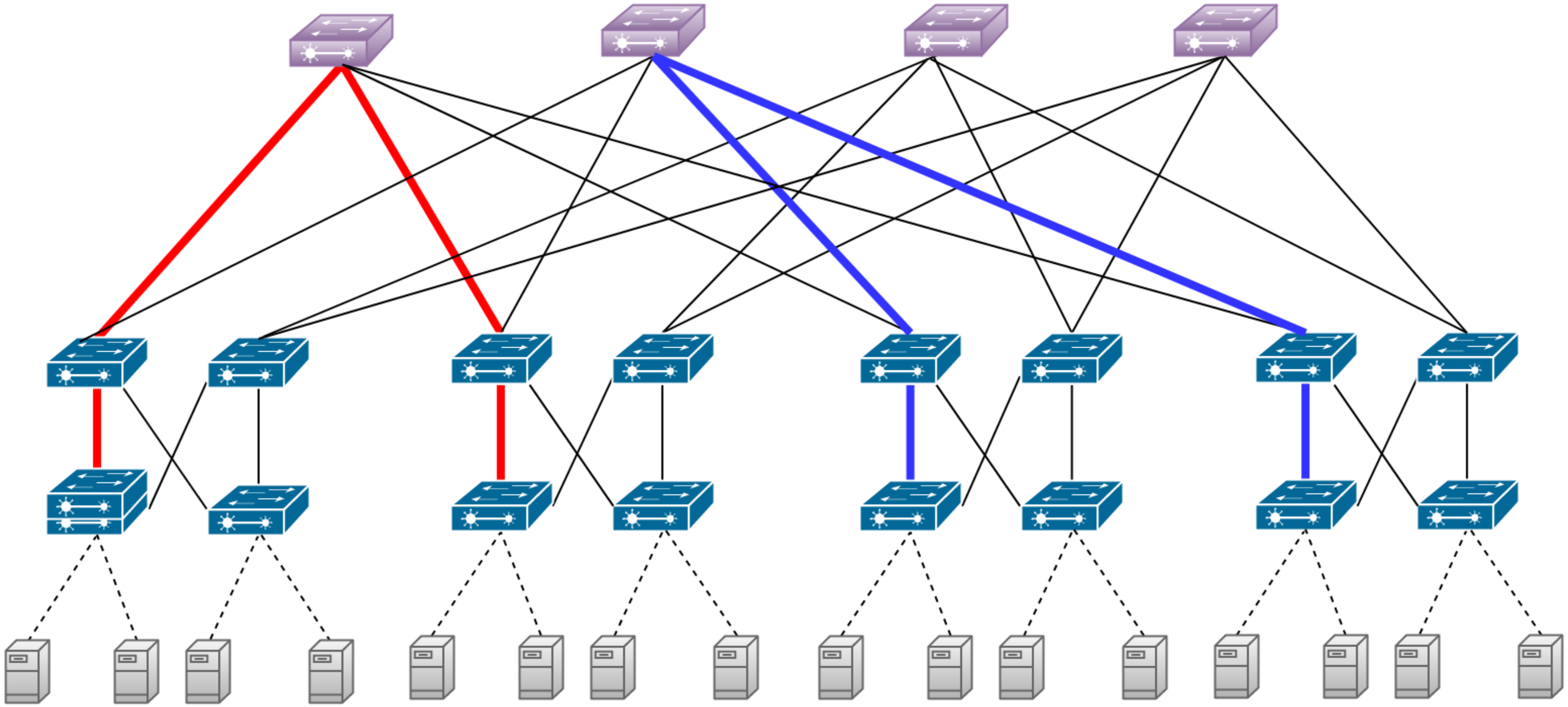
1	0
1	1

Optimization II: Lazy Update



- Defer the score update of a path as much as possible until we have to.
- Correctness guaranteed by the submodularity of the objective function.

Optimization III: Symmetry Reduction



Most DCN topologies are symmetric!

PMC Algorithm Results of Fattree

- Running time on one Xeon E5-2620 CPU

From days to seconds

DCNs	# of nodes	# of links	# of original paths	Strawman	Decomposition	Lazy update	Symmetry reduction
Fattree(12)	612	1296	184,032	231.458	5.216	0.506	0.126
Fattree(24)	4,176	10,368	11,902,464	> 24h	1381.226	23.254	0.280
Fattree(72)	99,792	279,936	8,703,770,112	> 24h	> 24h	> 24h	17.054

PMC Algorithm Results of Fattree

- Running time on one Xeon E5-2620 CPU

From days to seconds

DCNs	# of nodes	# of links	# of original paths	Strawman	Decomposition	Lazy update	Symmetry reduction
Fattree(12)	612	1296	184,032	231.458	5.216	0.506	0.126
Fattree(24)	4,176	10,368	11,902,464	$>$ 24h	1381.226	23.254	0.280
Fattree(72)	99,792	279,936	8,703,770,112	$>$ 24h	$>$ 24h	$>$ 24h	17.054

- The number of probe paths

The optimal needs at least 52428 paths

DCNs	# of original paths	# of selected paths w (α, β)		
		(1, 0)	(1, 1)	(3, 2)
Fattree(32)	66,977,792	4,096	7,680	12,288
Fattree(64)	4,292,870,144	32768	61,440	98,304

Phase II: Network Probing

Network Probing

- UDP probes: varying packet length, DSCP, source port
- Source routing: IP-in-IP encapsulation and decapsulation
- Responders: simply echo probes back

Phase III: Loss Localization

Loss Localization Problem

	Link 1	Link 2	Link 3	Loss measurements
Path 1	1	1	0	4
Path 2	1	0	1	1
Path 3	0	0	1	3

- Given the probe matrix and loss measurements, select the least number of links to explain the observation.
- NP-hard

Packet Loss Localization (PLL) Algorithm

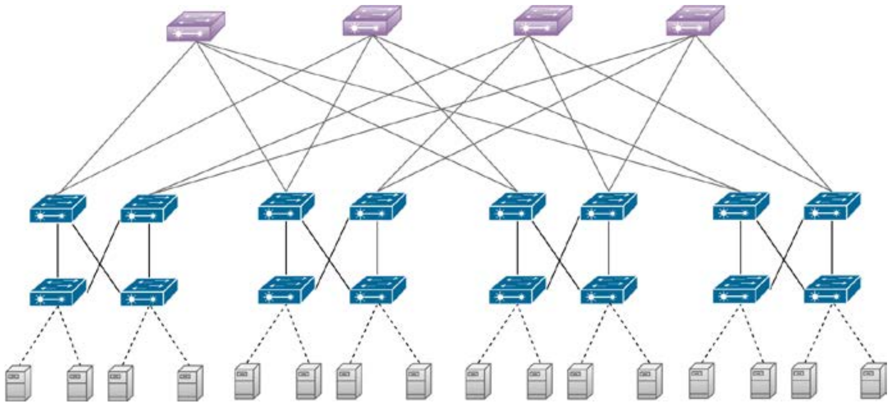
- In each iteration we select a link that can explain the largest number of probe losses until all are explained
  – If a link lies in the packet path, then the link can explain the loss.

- Two simple improvements
  – Matrix decomposition to speedup computation
  – Use threshold to filter false positives
    - The ratio of # of lossy paths over # of all probe paths through the link

Experiment

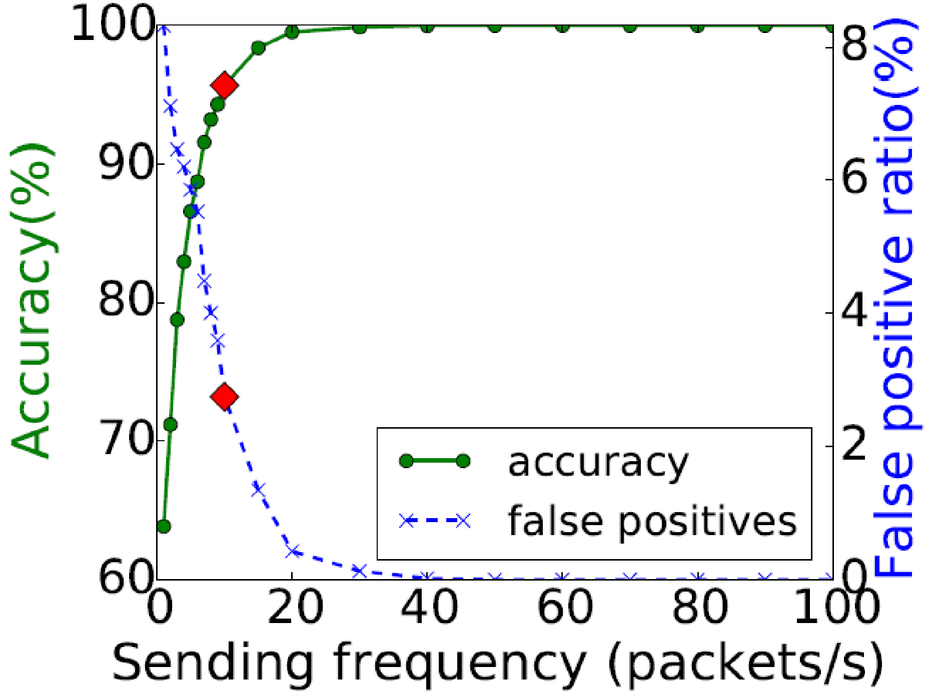
- A 4-ary Fattree testbed with 20 SDN switches
- Install OpenFlow rules to emulate losses caused by various failures
  - Full packet loss: link down etc.
  - Deterministic partial loss: packet blackhole etc.
  - Random partial loss: bit flips etc.
- Performance metric
  - Accuracy
  - False positive ratio

ONetSwitch

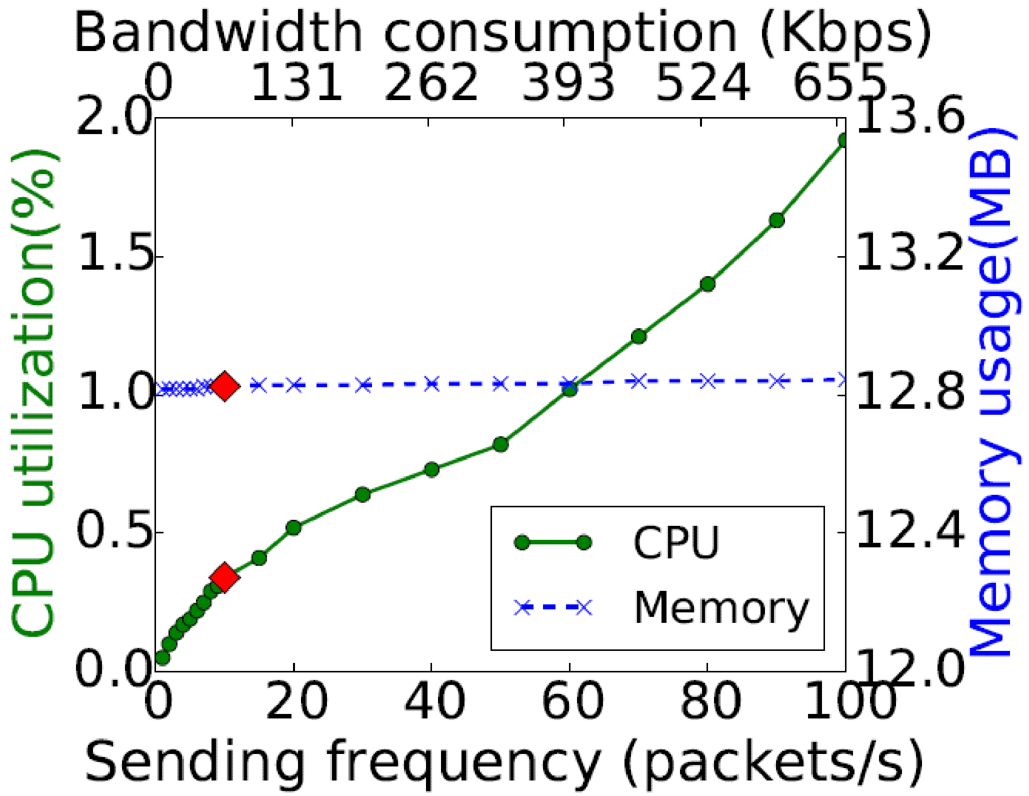


Experiment

- Sensitivity test of sending frequency



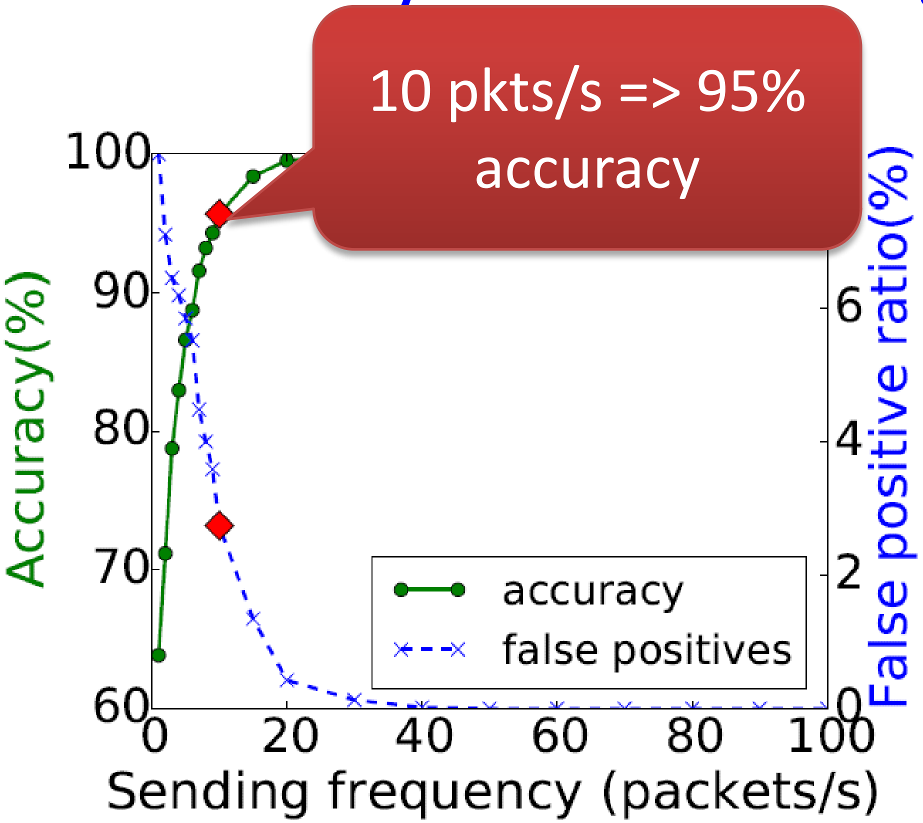
Accuracy and false positives



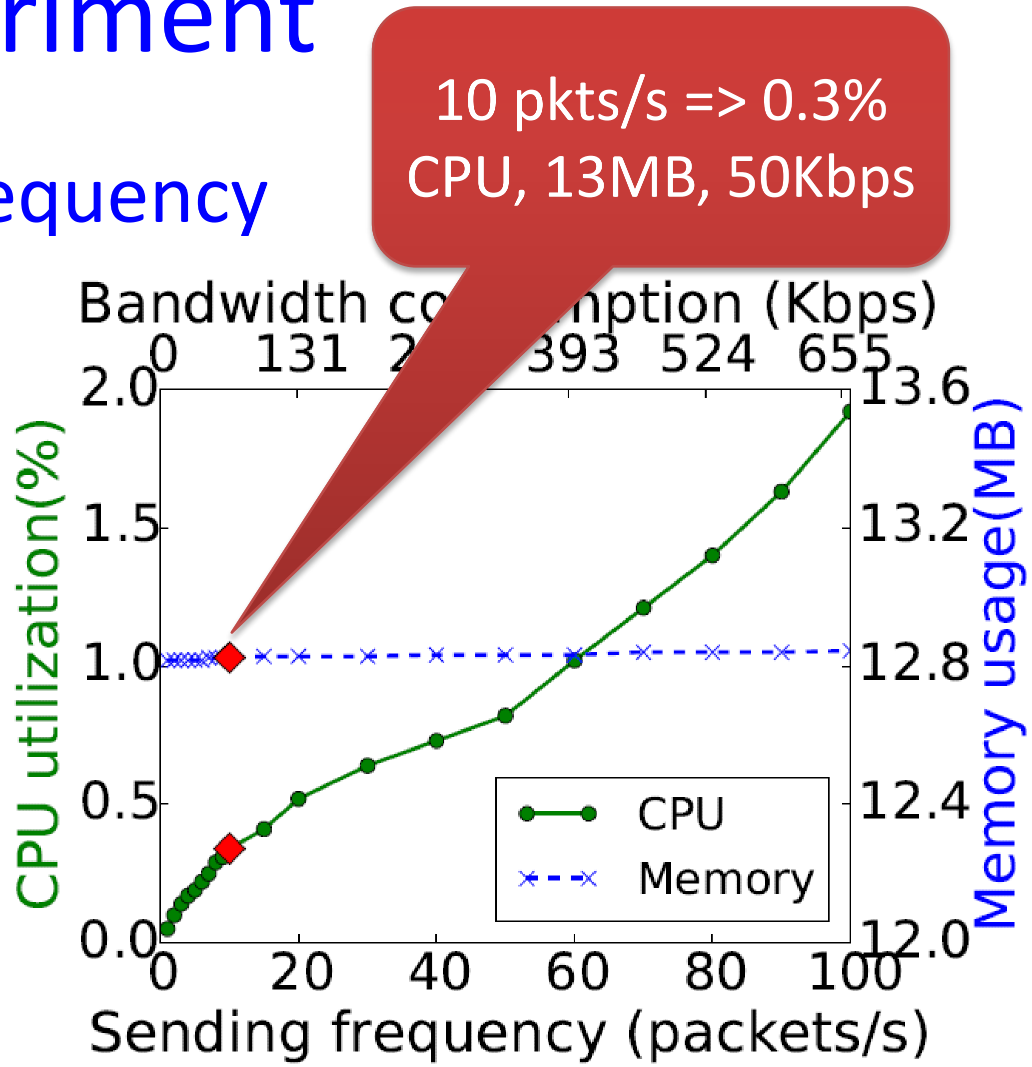
Overhead of pingers

Experiment

- Sensitivity test of sending frequency



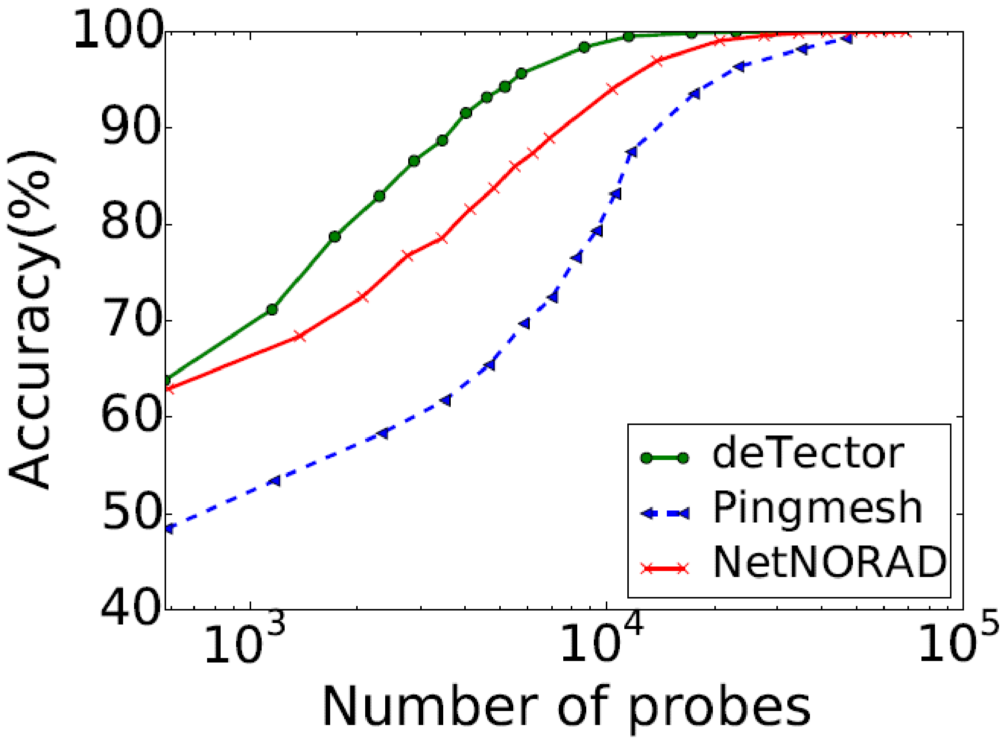
Accuracy and false positives

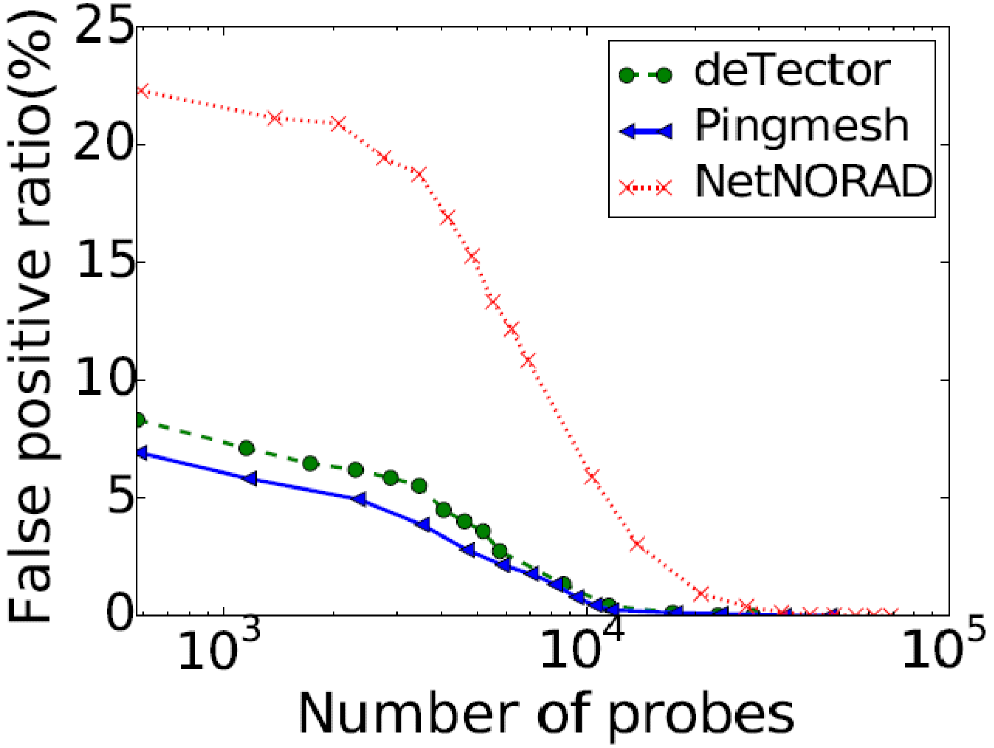


Overhead of pingers

Experiment

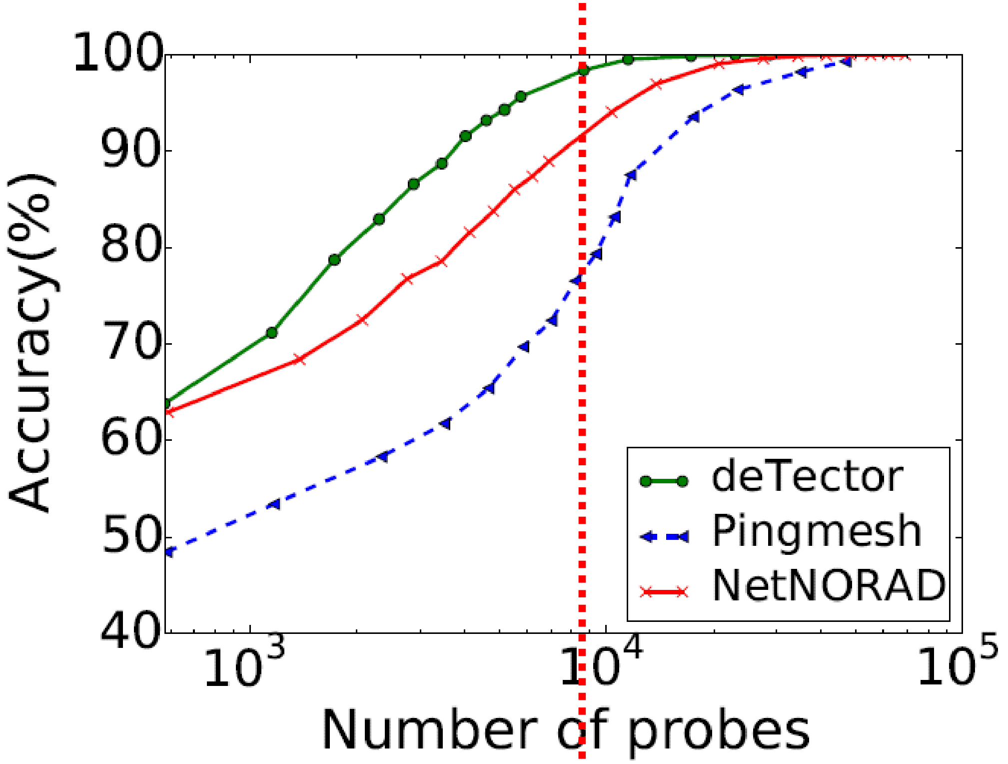
- Accuracy and false positives of three monitoring systems

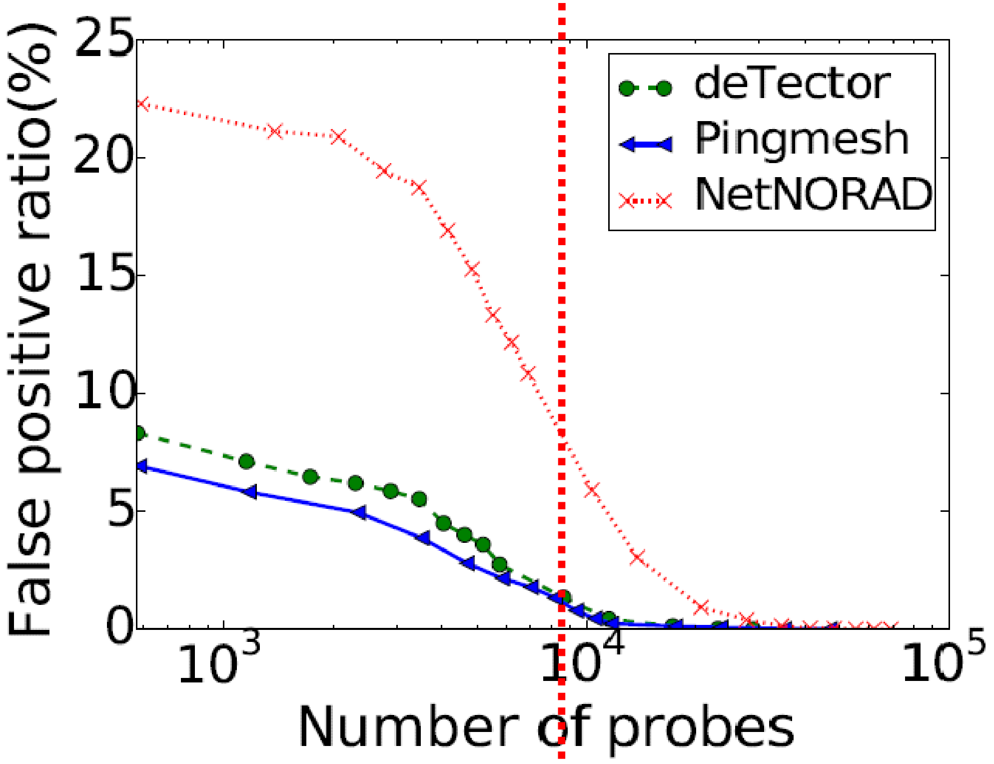




Experiment

- Accuracy and false positives of three monitoring systems





Same # of probes, 98%, 89%, 78% accuracy respectively

Same # of probes, 1%, 8%, 1% false positives respectively

Simulation

- Accuracy in a 18-radix Fattree, with probe matrices of different levels of α-coverage and β-identifiability

(α, β)	# of paths	Accuracy (%) with # of failed links				
		1	5	10	20	50
$(1, 0)$	729	30.56	30.87	30.30	30.26	29.19
$(2, 0)$	1485	58.43	57.43	57.08	56.81	57.11
$(3, 0)$	2187	68.22	70.61	69.89	70.40	70.14
$(1, 1)$	1269	94.74	93.37	94.21	93.43	90.29
$(1, 2)$	1512	99.26	99.06	99.02	98.77	95.92
$(1, 3)$	2349	99.63	99.63	99.67	99.62	98.07

Simulation

- Accuracy in a 18-radix Fattree, with probe m[illegible] different levels of α-coverage and β-identifiabil[illegible]

Identifiability is more effective than coverage for failure localization.

(α, β)	# of paths	Accuracy (%) with # of faile[illegible]nks				
		1	5	10	2[illegible]	50
(1, 0)	729	30.56	30.87	30.30	[illegible].26	29.19
(2, 0)	1485	58.43	57.43	57.08	56.81	57.11
(3, 0)	2187	68.22	70.61	69.89	70.40	70.14
(1, 1)	1269	94.74	93.37	94.21	93.43	90.29
(1, 2)	1512	99.26	99.06	99.02	98.77	95.92
(1, 3)	2349	99.63	99.63	99.67	99.62	98.07

Simulation

- Accuracy in a 18-radix Fattree, with probe matrices of different levels of α-coverage and β-identifiability

(α, β)	# of paths	Accuracy (%) with # of failed links				
		1	5	10	20	50
(1, 0)	729	30.56	30.87	30.30	30.26	29.19
(2, 0)	1485	58.43	57.43	57.08	56.81	57.11
(3, 0)	2187	68.22	70.61	69.89	70.40	70.14
(1, 1)	1269	94.74	93.37	94.21	93.43	90.29
(1, 2)	1512	99.26	99.06	99.02	98.77	95.92
(1, 3)	2349	99.63	99.63	99.67	99.62	98.07

2-identifiability is enough.

Summary

- The core of deTector is a carefully designed probe matrix, enabling fast and accurate loss detection and localization.
- deTector is practically deployable.
- Discussions
  - Packet entropy: limited destination IP addresses
  - Loss diagnosis: do not know why packets are dropped
  - Beyond deTector: apply probe matrix to optimize in-band techniques

Thanks