# pHost

## Distributed Near-Optimal Datacenter Transport Over Commodity Network Fabric

**Peter X. Gao**, Akshay Narayan, Gautam Kumar

Rachit Agarwal, Sylvia Ratnasamy, Scott Shenker

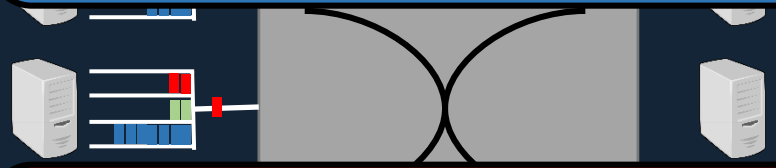
UC Berkeley/ICSI

# Datacenter Transport Protocol Designs


- Low Latency (FCT)
- High Throughput
- Deadline
- Fairness

# Performance and Flexibility

Close to optimal performance

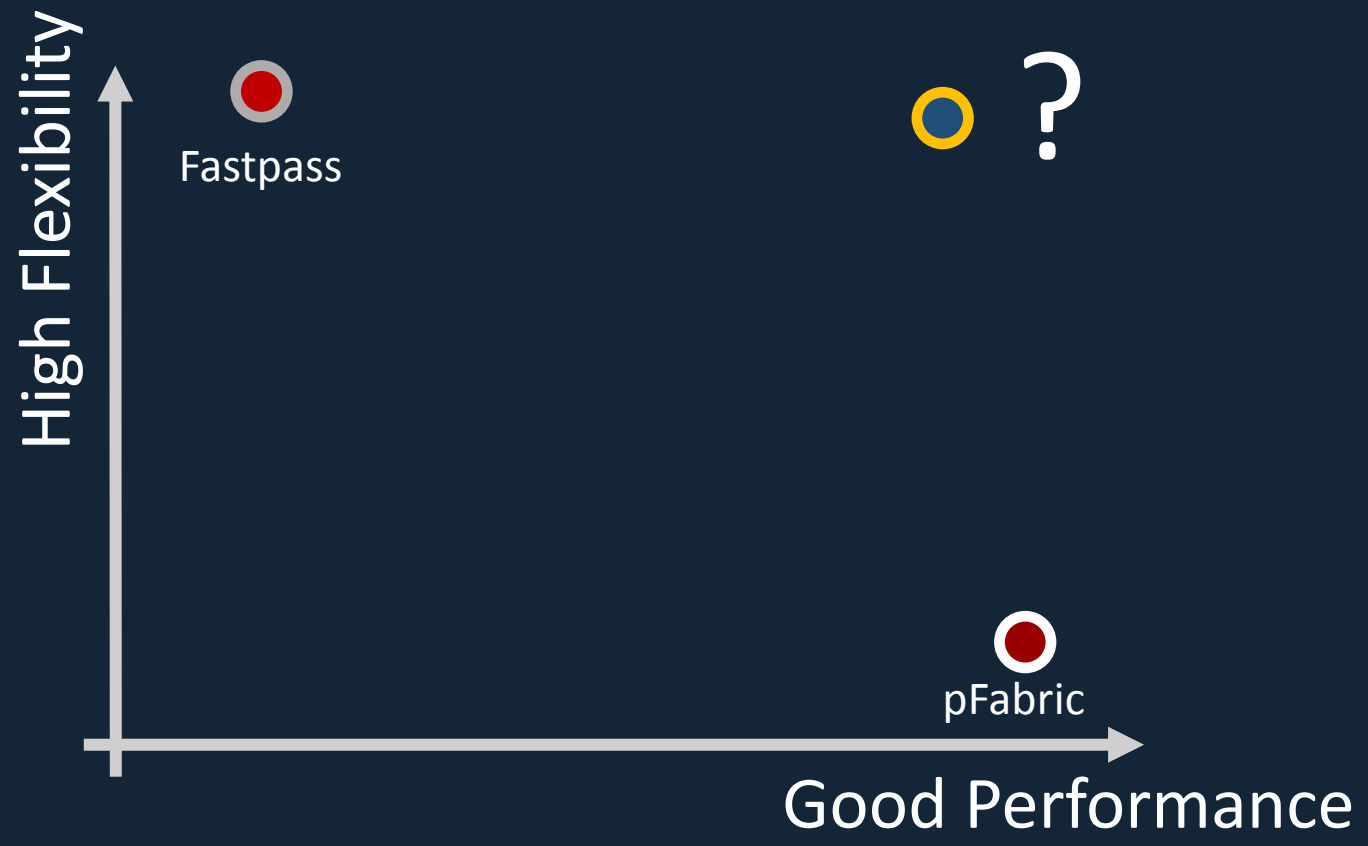


Embed policy inside network

Overhead on Scheduling Short flows


Highly flexible

# Can we get the best of both worlds?



# Design Intuition

Full Bisection Bandwidth

Packet Spraying

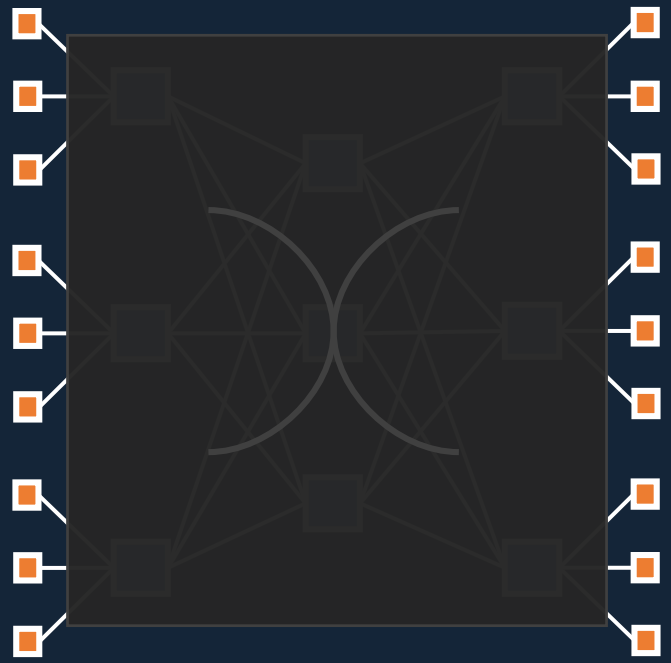
Congestion is at the edge

Sources

Destinations

Easy

Pull-based



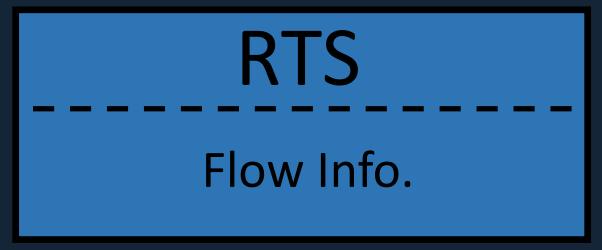
# pHost

- No specialized hardware
- No complex computation inside the fabric
- No centralized scheduler
- No explicit network feedback

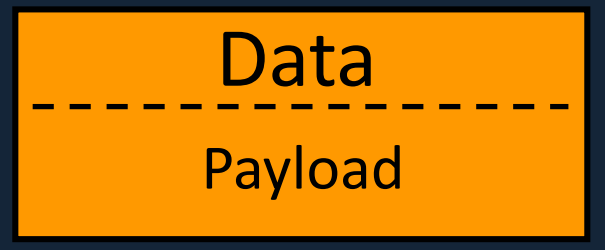
With surprisingly near optimal performance

# pHost Basics

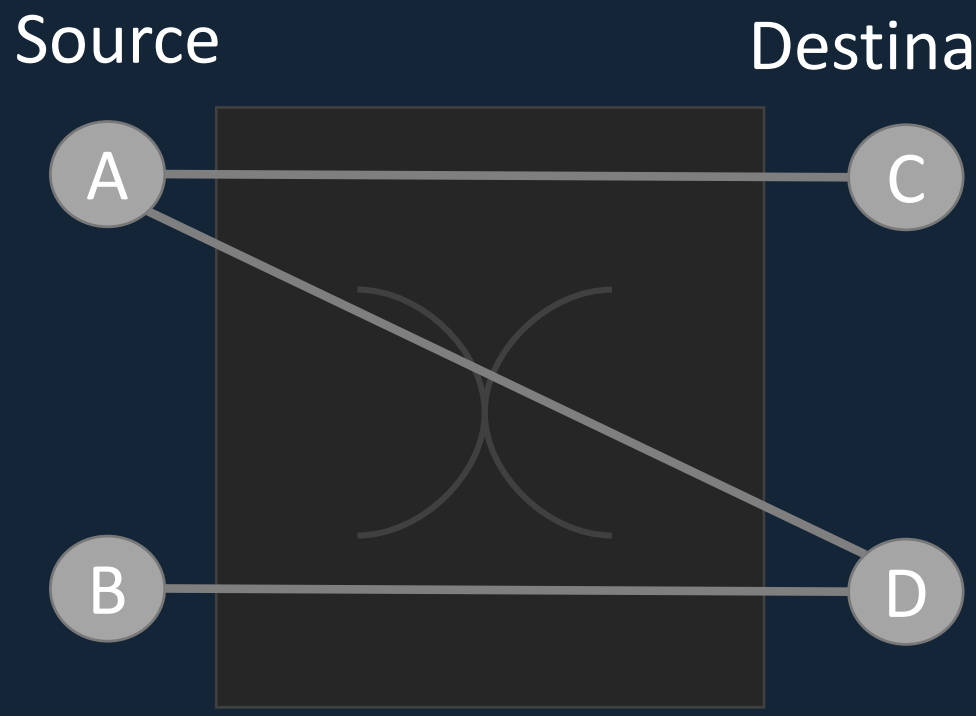
- Source sends RTS on flow arrival
- Destination issues Token every MTU transmission time
- Source sends Data packet when it has a token







# Pull-based End-host Scheduling




A distributed bipartite matching problem

NP-Hard

High Utilization at Sources

High Utilization at Destinations

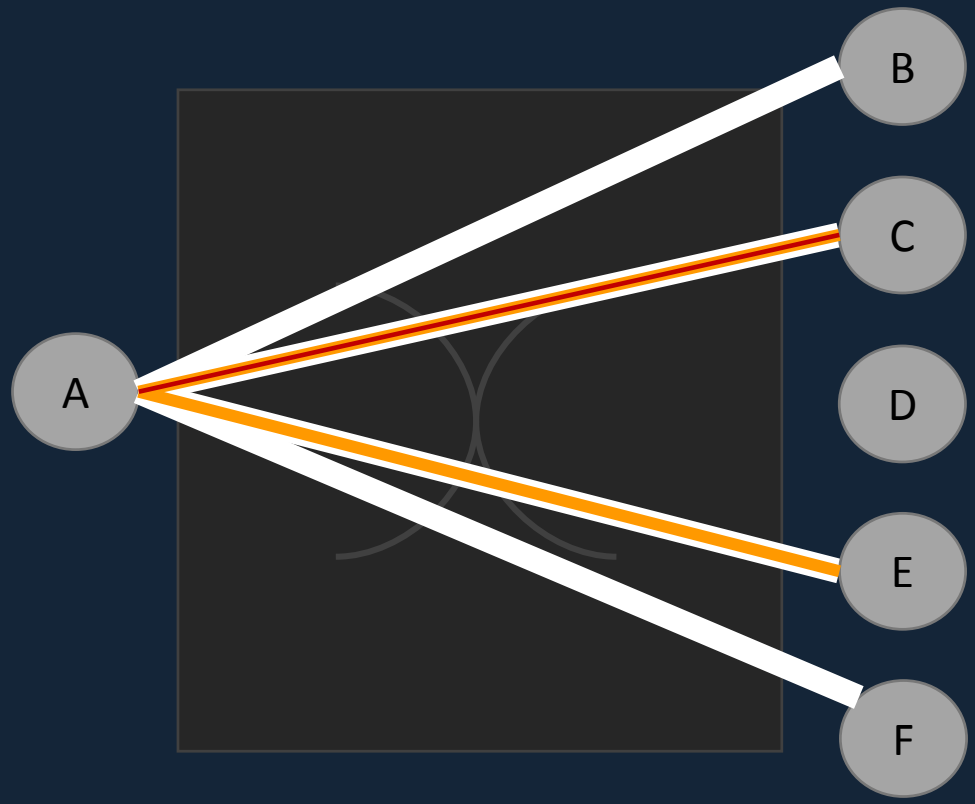
# Ensuring high utilization at the sources

Round Trip Time between RTS and first Token

Assign a BDP worth of Free Tokens initially

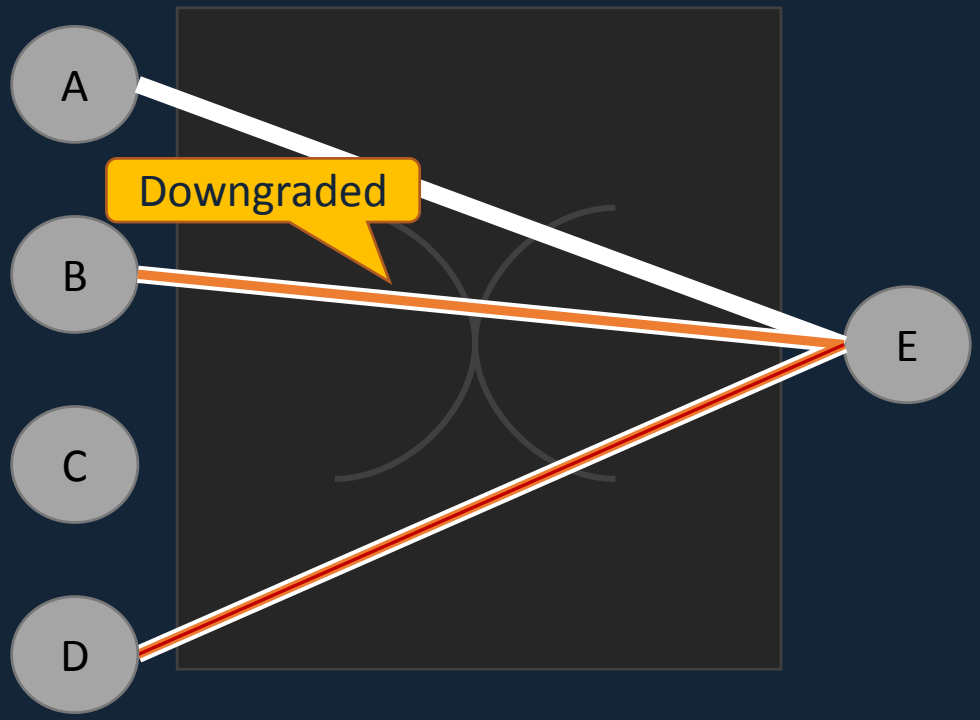
# Ensuring high utilization at the sources

- Allows sources to send multiple RTSes in parallel
- Sources may receive tokens from a subset of the flows
- Sources choose one of the flows depending on the scheduling policy



# Ensuring high utilization at the destination

- Tokens are expired if the source does not consume them
- If tokens issued to a flow expire, the flows are downgraded for a while



# Express Policy at Endhosts

- Destination side
  - Using token assignment
- Source side
  - Using token selection

Examples:

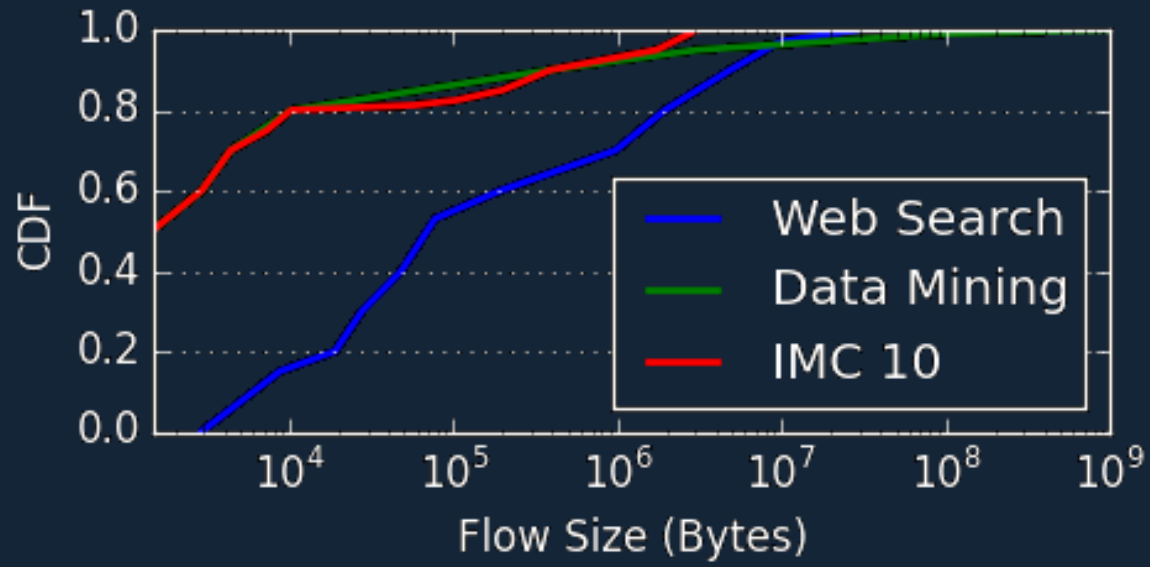
- SRPT
  - Shortest Remaining Flow Size
- Deadline
  - Earliest Deadline First
- Multi-tenant sharing
  - Round Robin / LRU

# Evaluation Setup

- Packet-level Simulator
- Topology
  - Two-tier multi-rooted tree
  - 9 Racks, 144 nodes
  - Cut-through switching
- Mean Slowdown

= mean$(FCT/OPT)$

- Workloads:



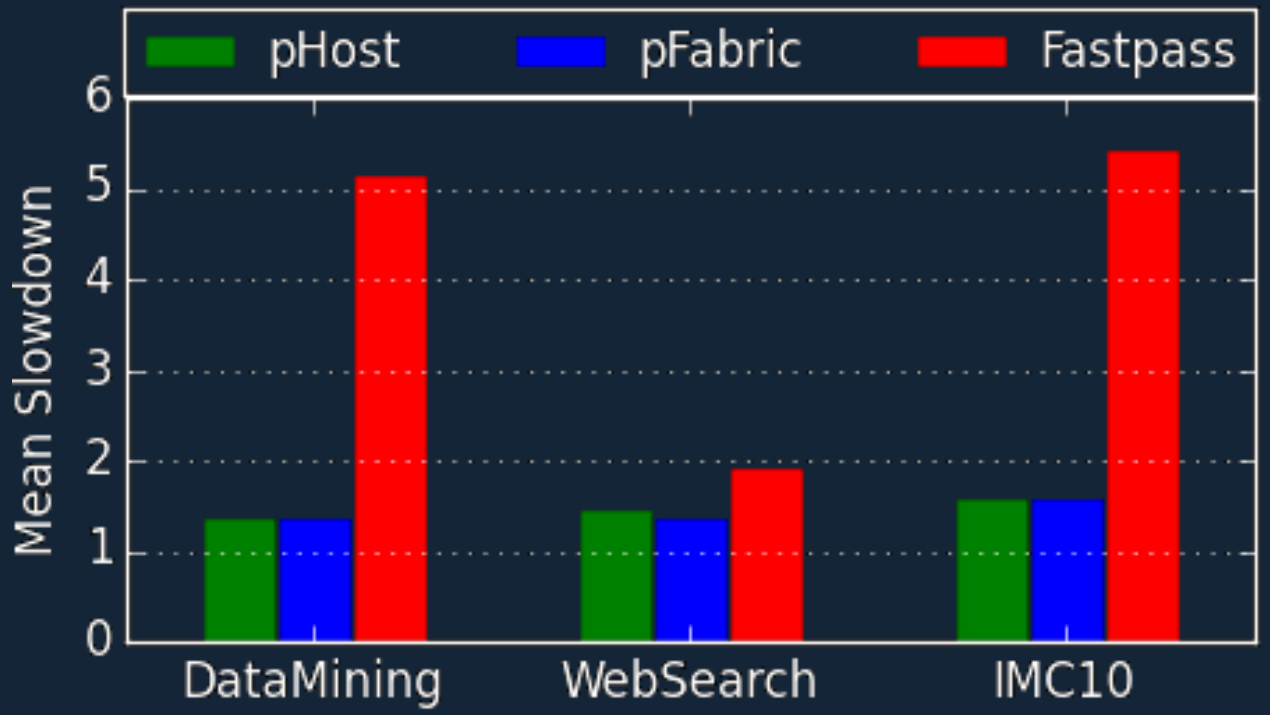
# Evaluation Outline

Can pHost match pFabric's performance?

Is pHost robust?

Is pHost flexible?

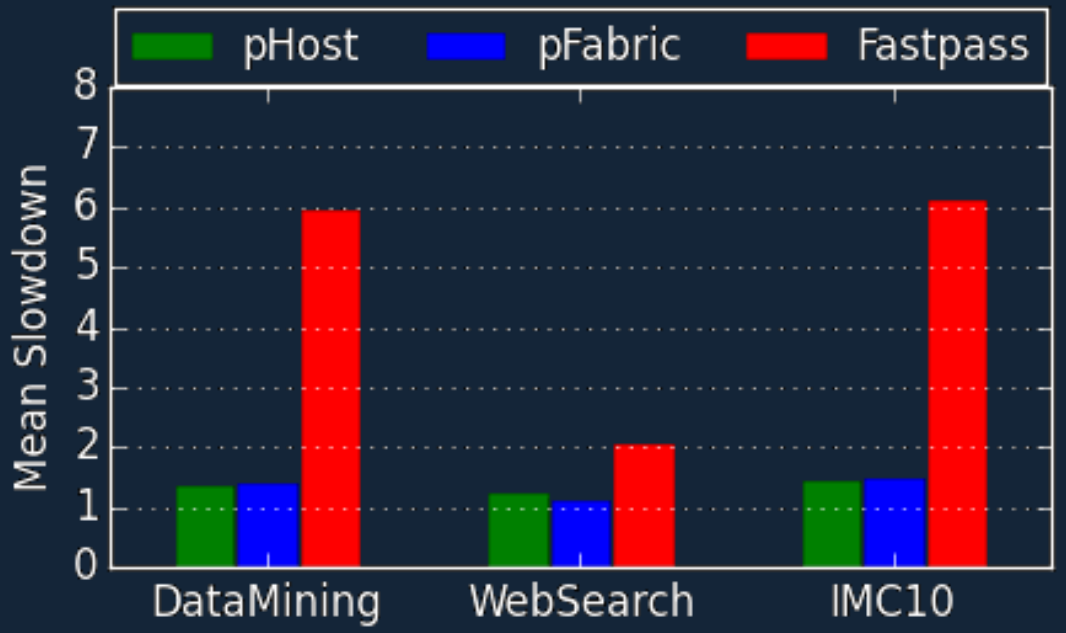
# Overall Mean Slowdown



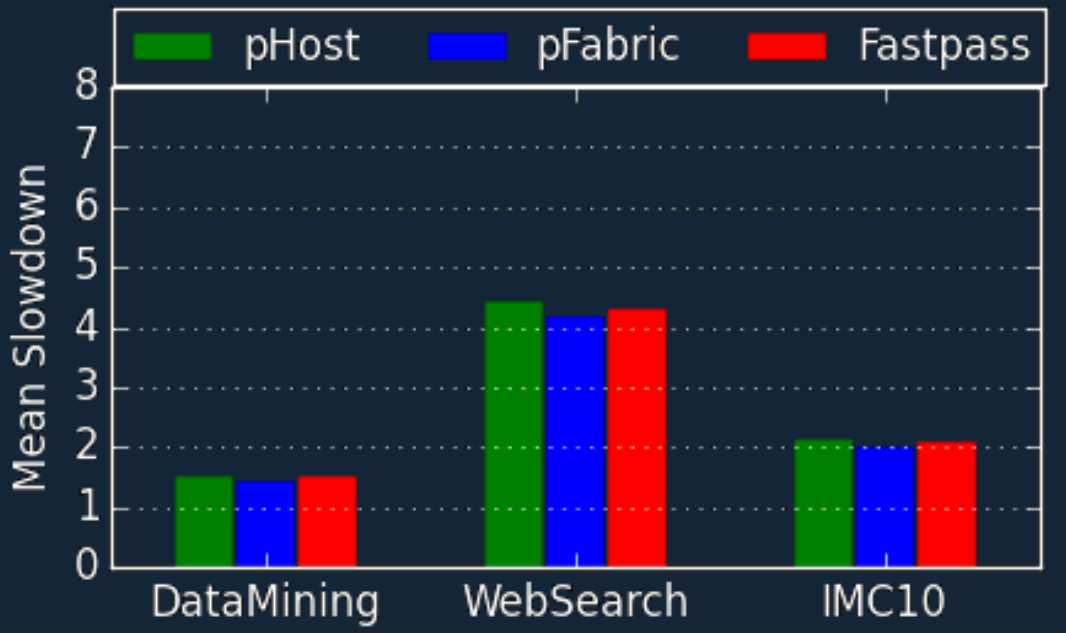

pHost matches pFabric and 4x better than Fastpass

pHost uses commodity switches

# Digging Deeper




Short Flow Slowdown

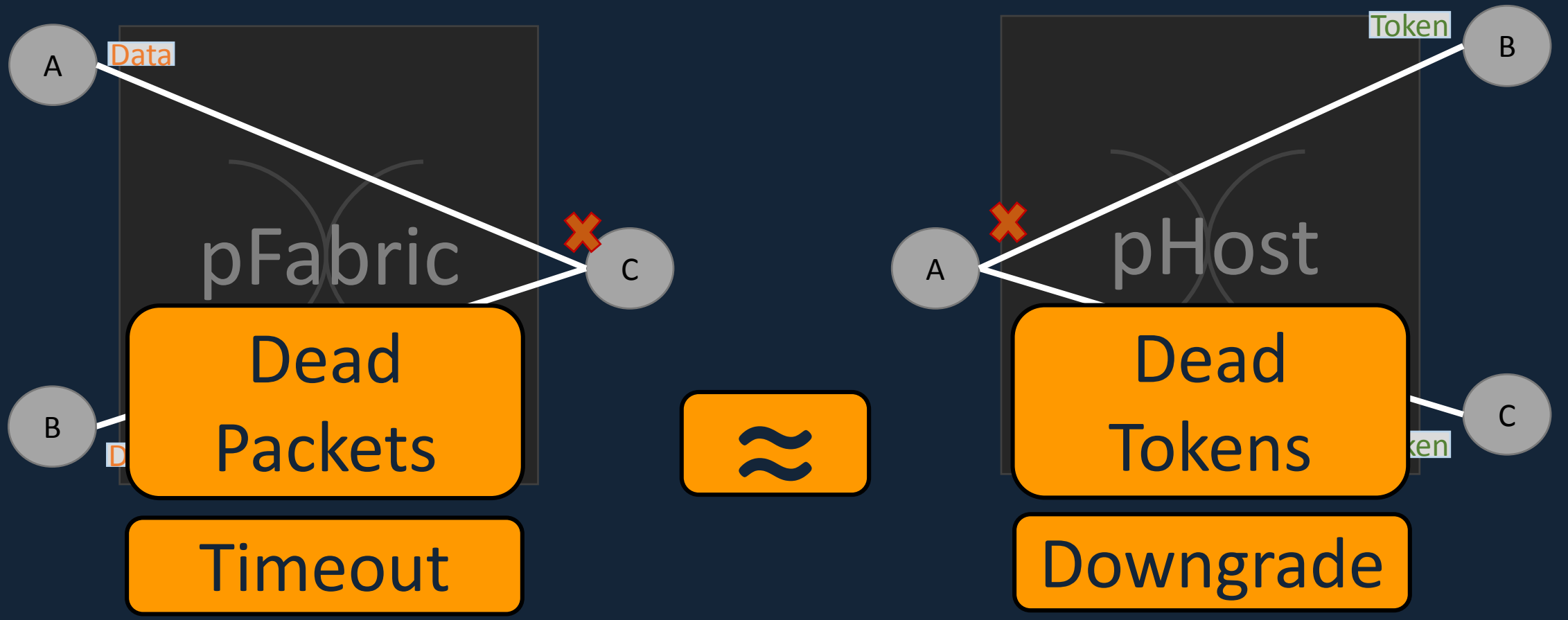



Long Flow Slowdown

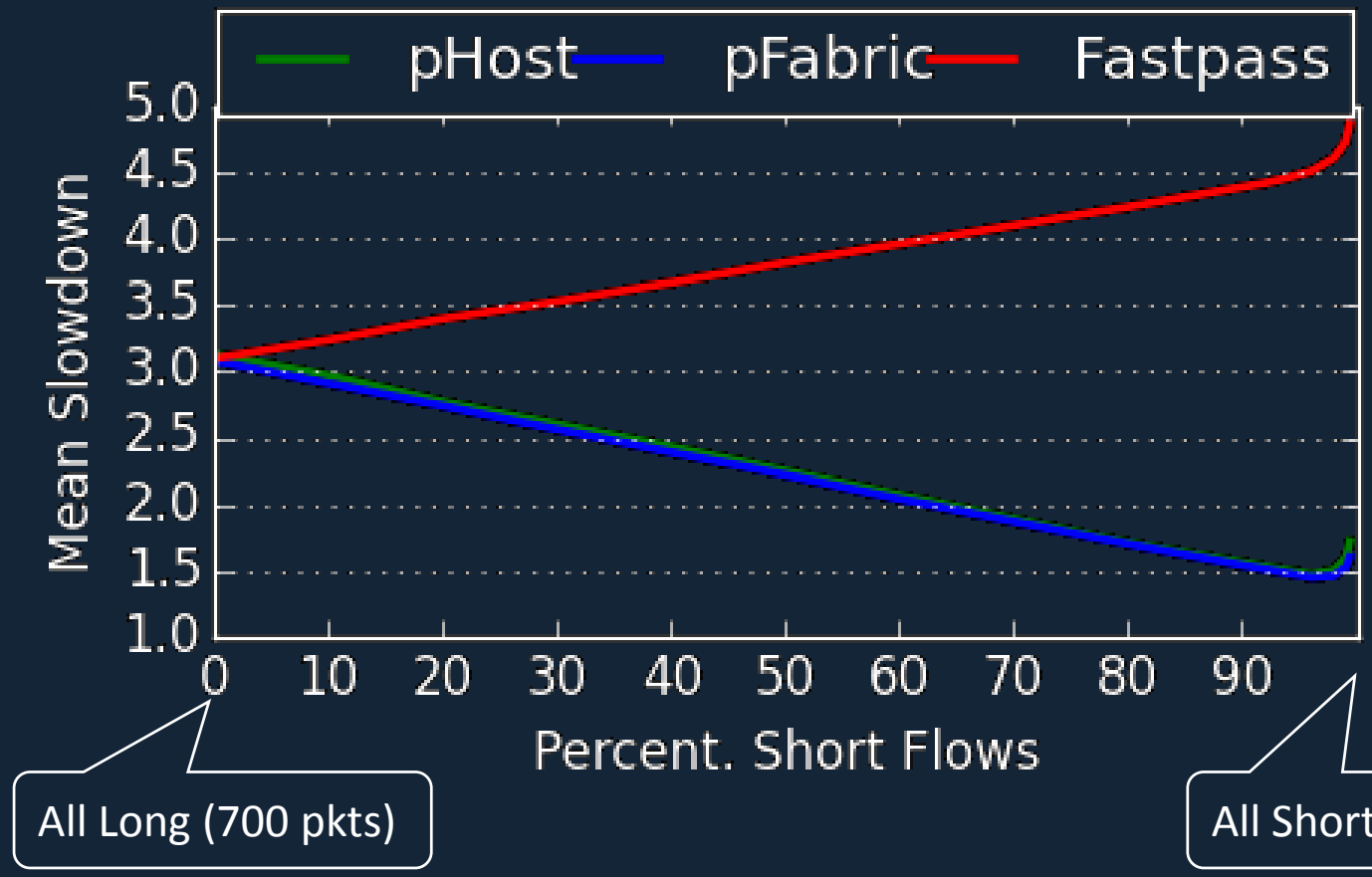
Overall performance dominated by short flow performance

# Digging Deeper

## How does pHost match pFabric's performance?

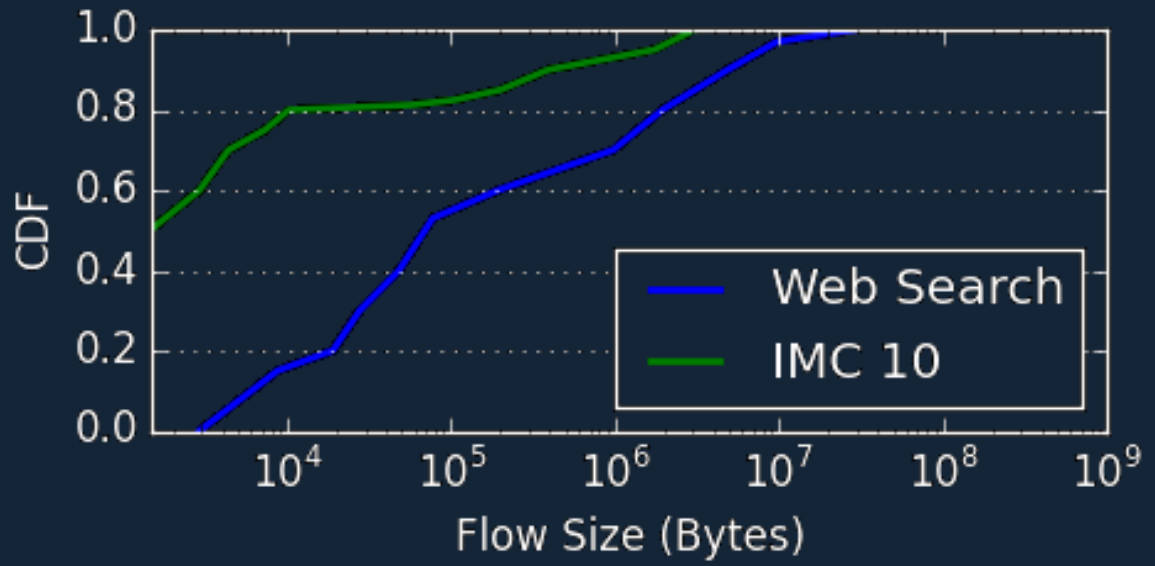


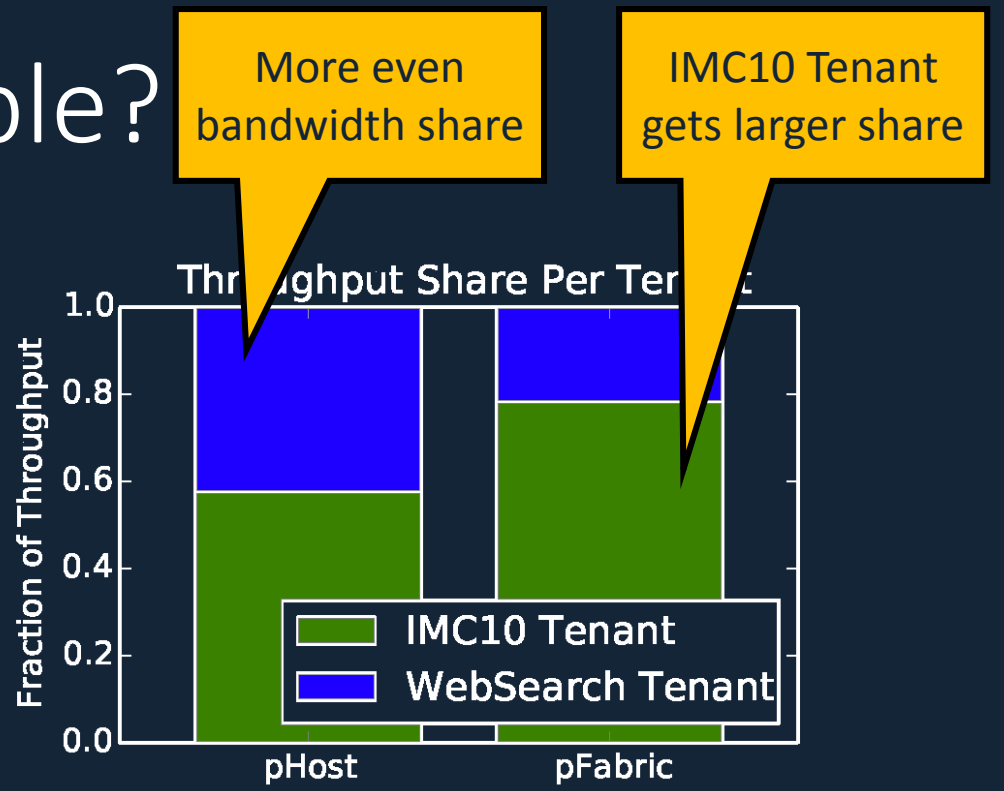
# Is pHost robust?




pHost is robust against a wide range of workloads

# Is pHost flexible?




pHost is more flexible

# More Results

| Metrics | Traffic Patterns | Sensitivity |
| --- | --- | --- |
| • Normalized FCT<br>• Tail Slowdown<br>• Deadline<br>• Throughput | • Random<br>• Incast<br>• Permutation | • Stability Analysis<br>• Varying Traffic Load<br>• Varying Switch Buffer Size<br>• Parameter Sensitivity |

Please read our paper for details

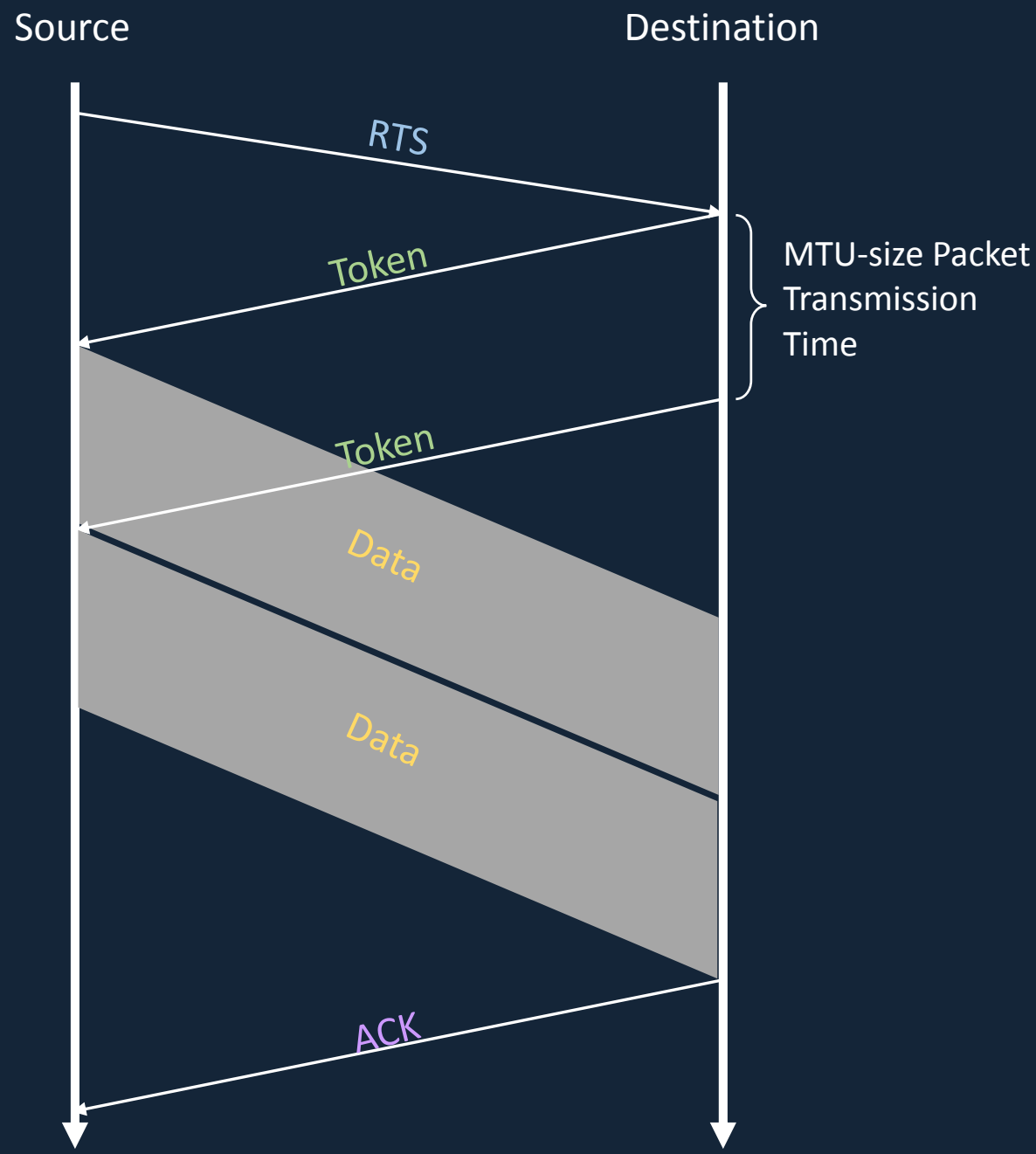
# pHost

- No specialized hardware
- No complex computation inside the fabric
- No centralized scheduler
- No explicit network feedback

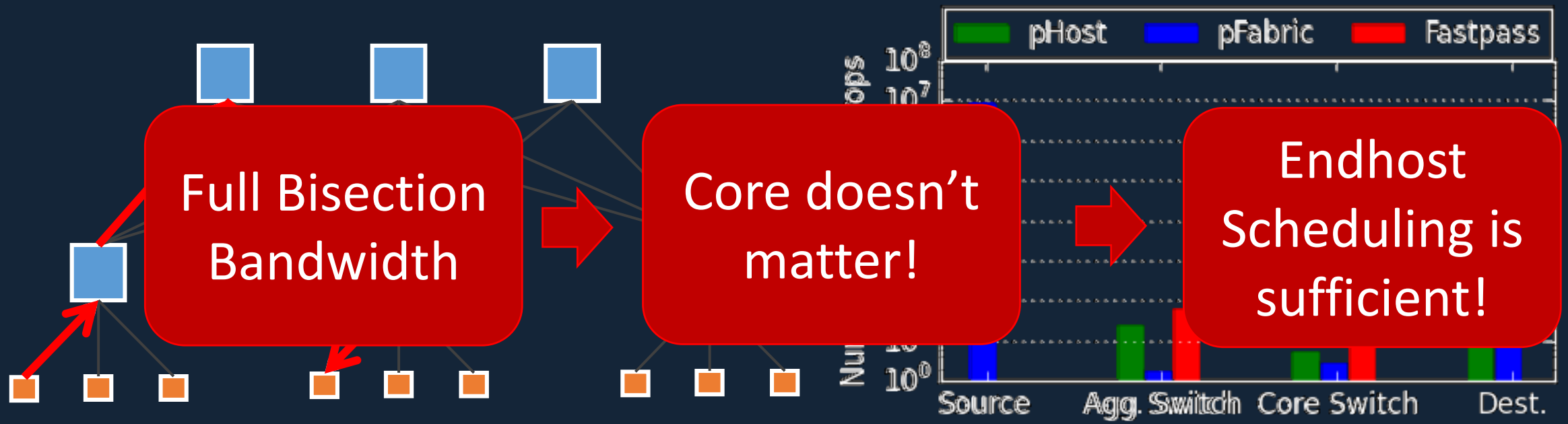
With ~~surprisingly~~ near optimal performance

# pHost Basics

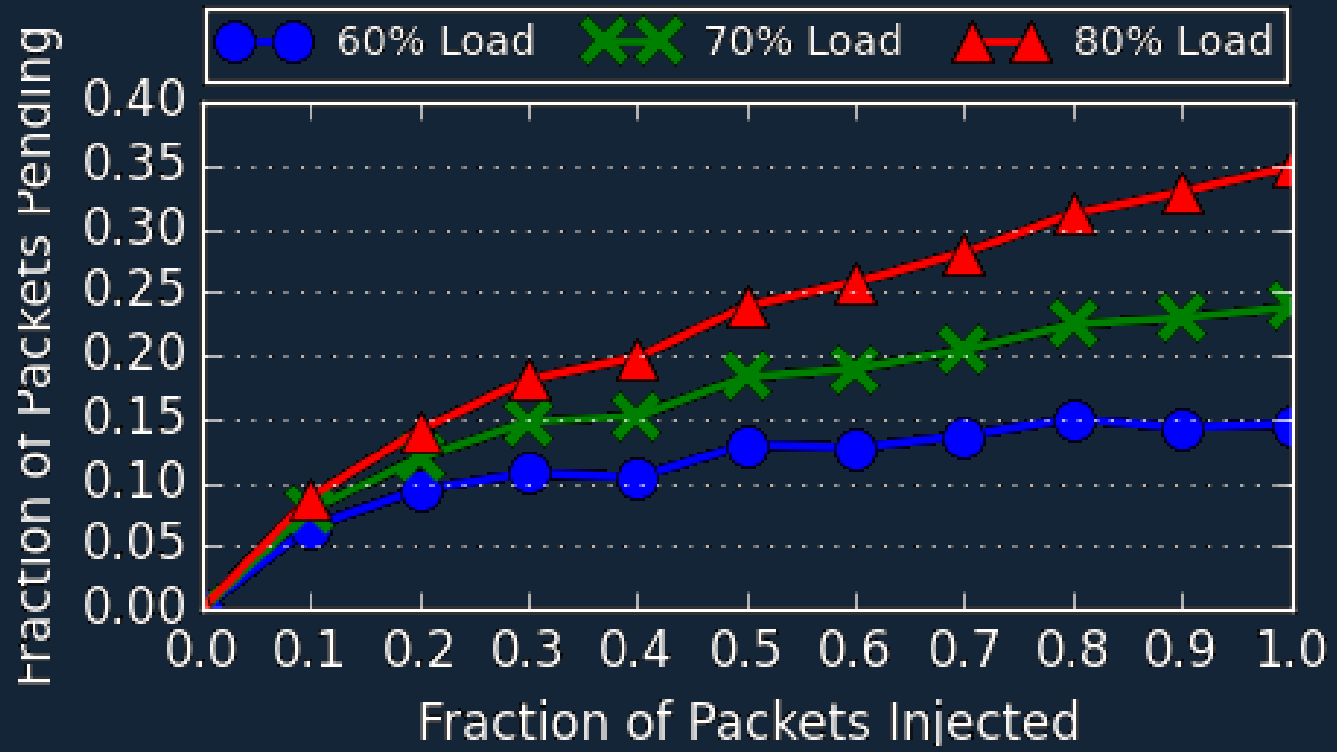
- Source sends an *RTS* on flow arrival
- The destinations issues a *Token* to the source every MTU-size packet transmission time
- The source can consume that token by sending a *Data* packet to the destination
- A token expires if the source does not use it quickly
- Finally, when the destination receives all the data packets, it sends an *ACK* to the source



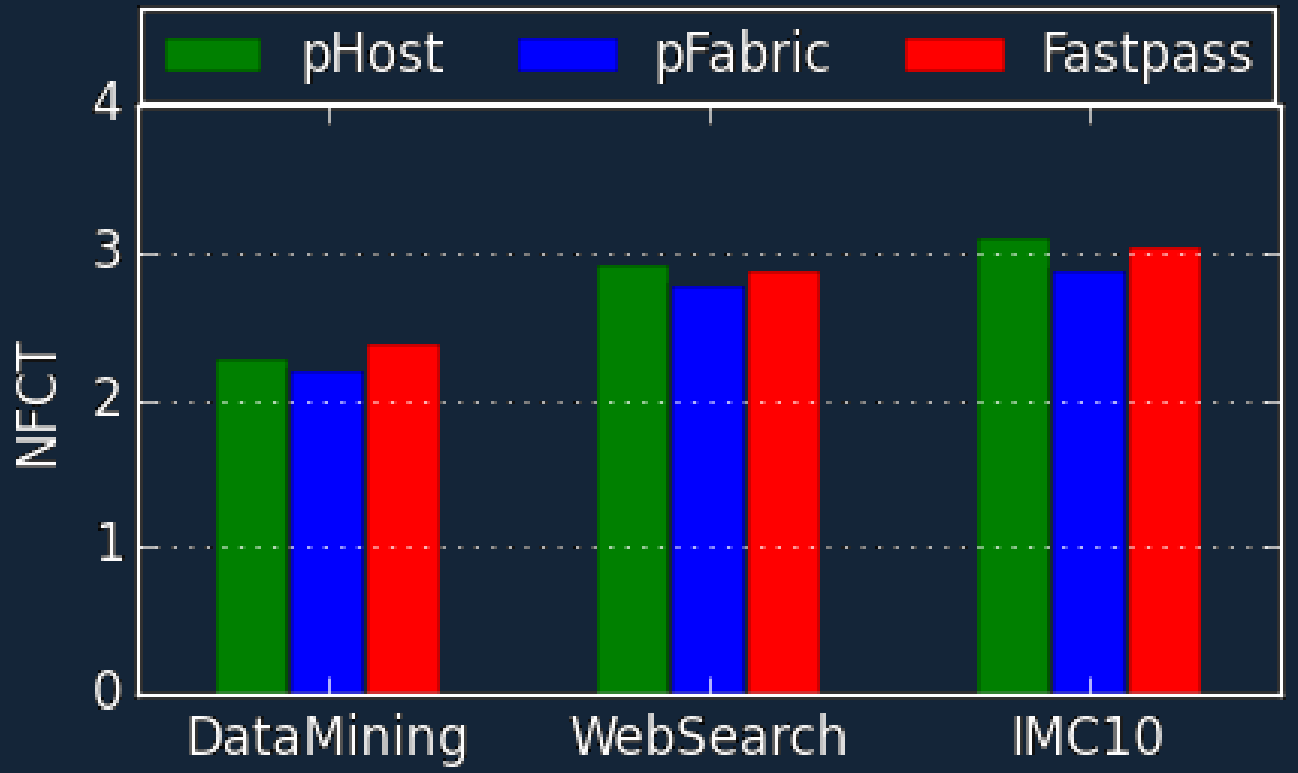
# Digging Deeper:
## How pHost matches pFabric performance?



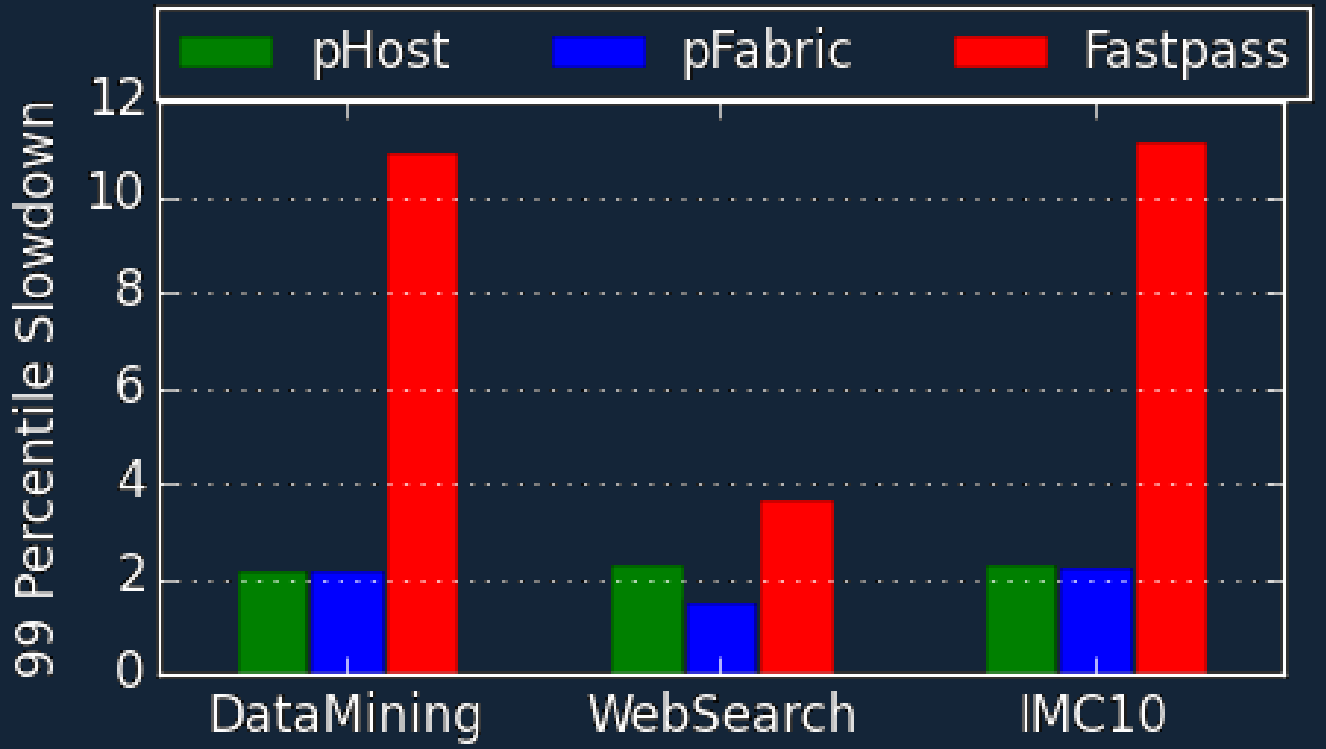
# Stability


60% Load
70% Load
80% Load
Fraction of Packets Pending
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Fraction of Packets Injected

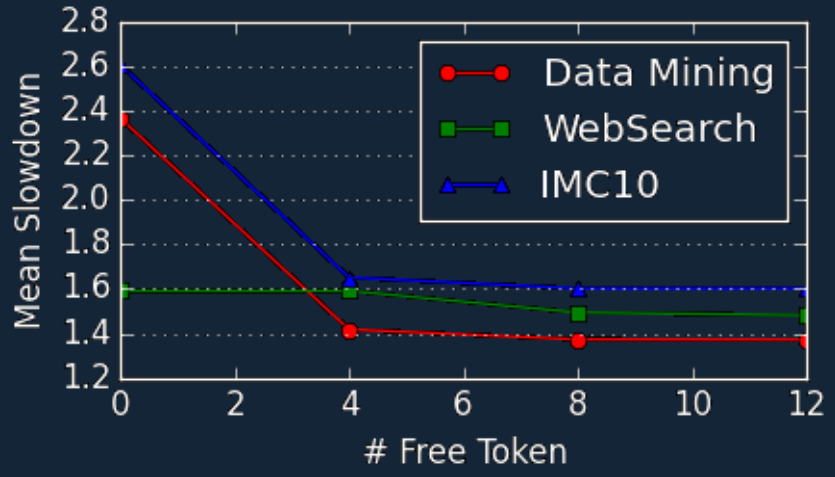
# NFCT

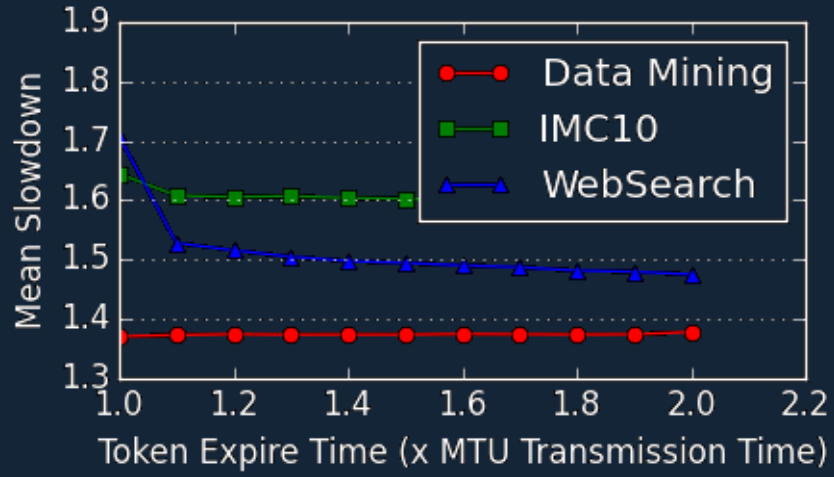

pHost
pFabric
Fastpass
NFCT
4
3
2
1
0
DataMining
WebSearch
IMC10

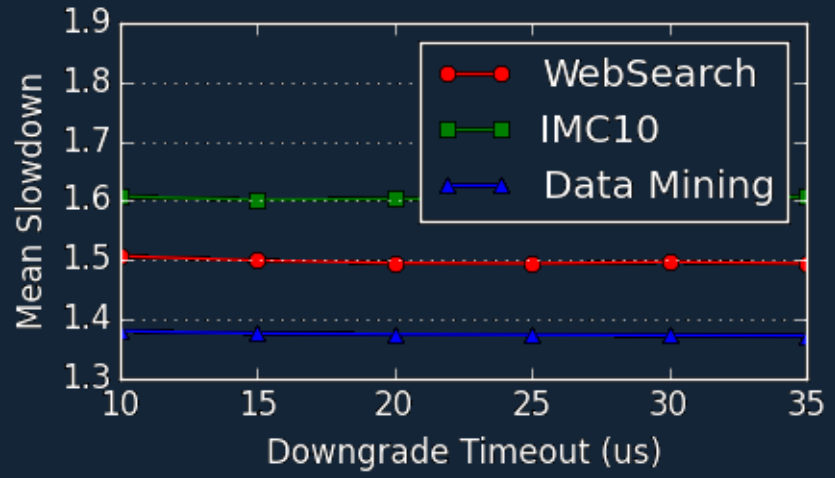
# 99%-ile slowdown for short flows

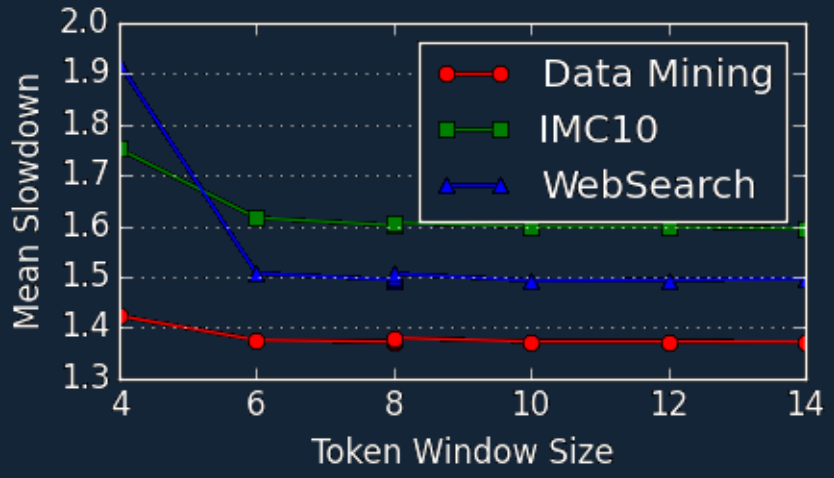


# Parameter Sensitivity Analysis



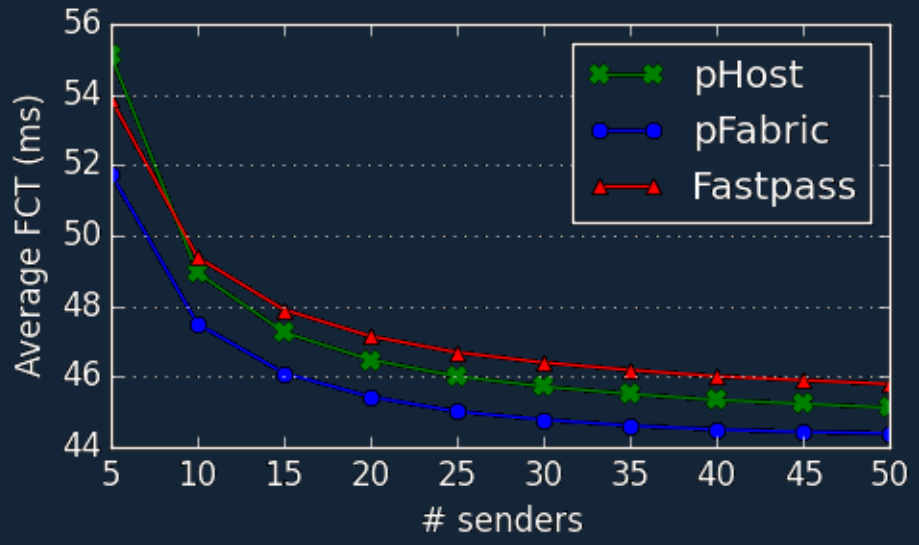



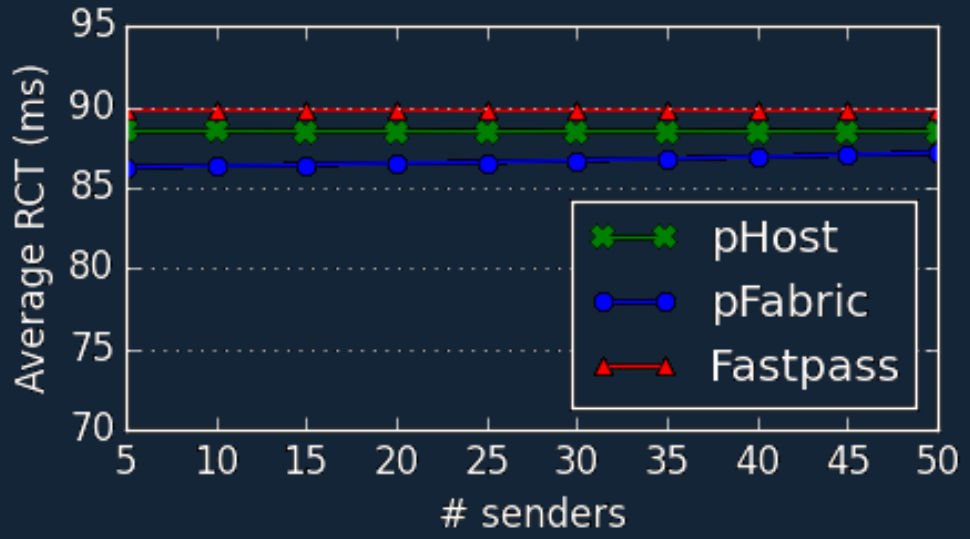



# Incast Traffic

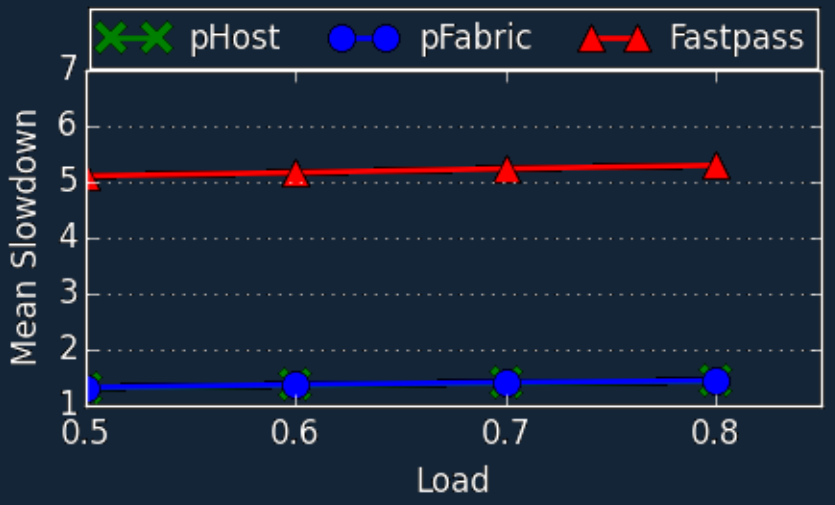
- Each Request: N sources send 100MB data to one receiver
- 10000 requests



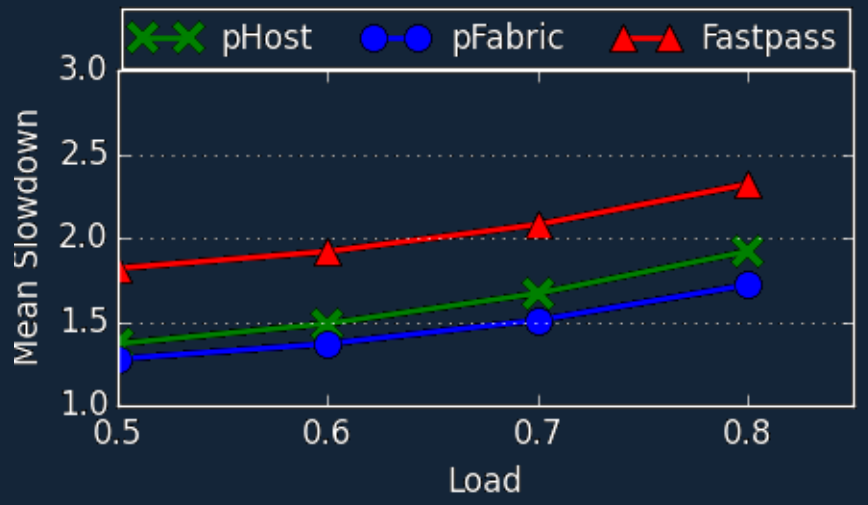




The performance difference between all three protocols are less than 4% and 7% in terms of FCT and RCT

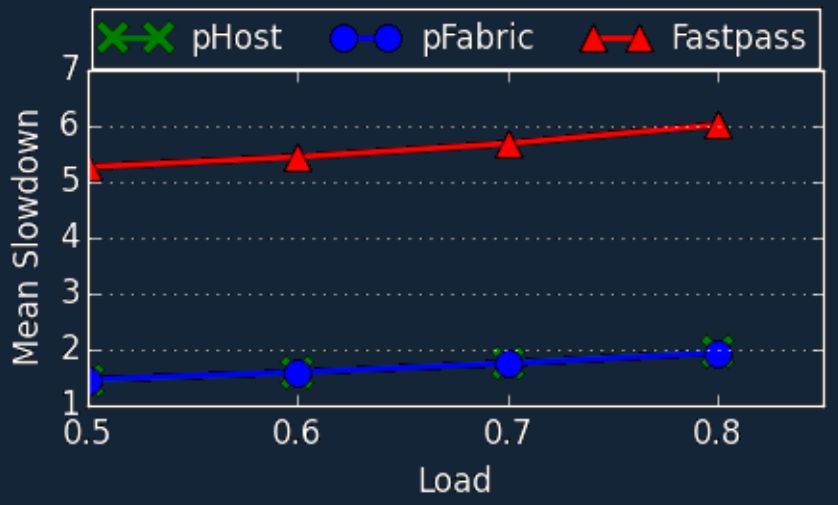
# Varying load




Data Mining




Web Search

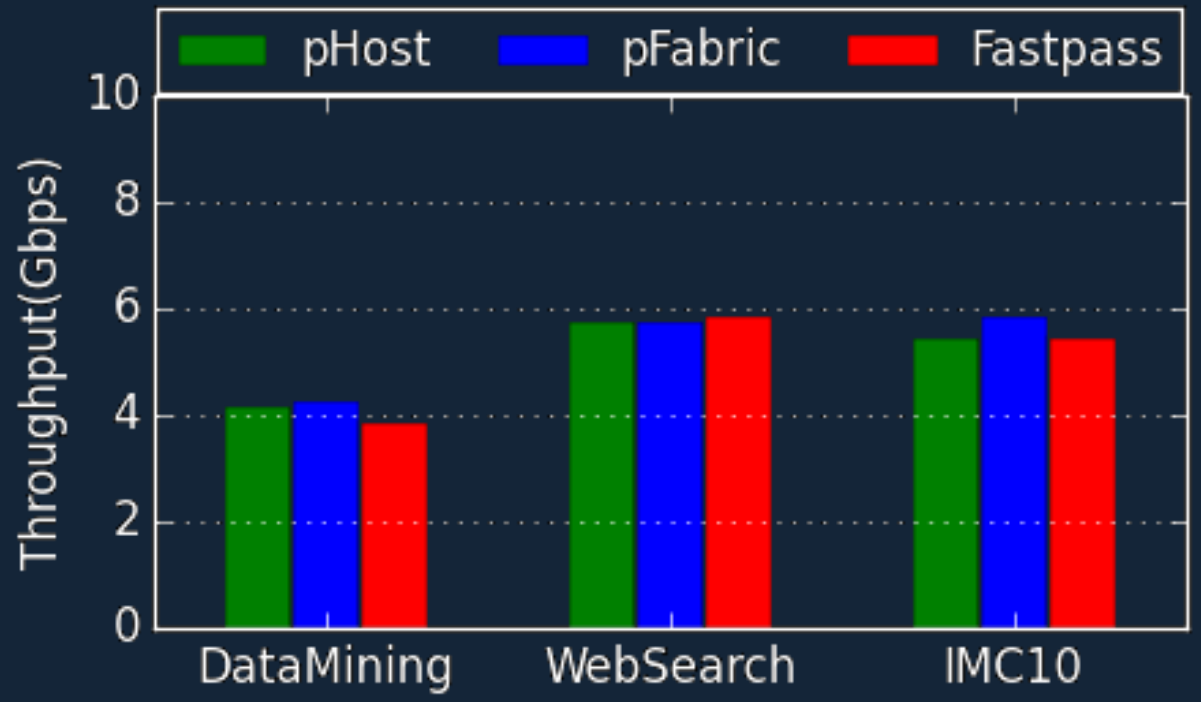



IMC 10

There is no relative performance change with varying load

# Throughput

- Load: Average byte rate at sender
- Throughput:  Average byte rate at receiver

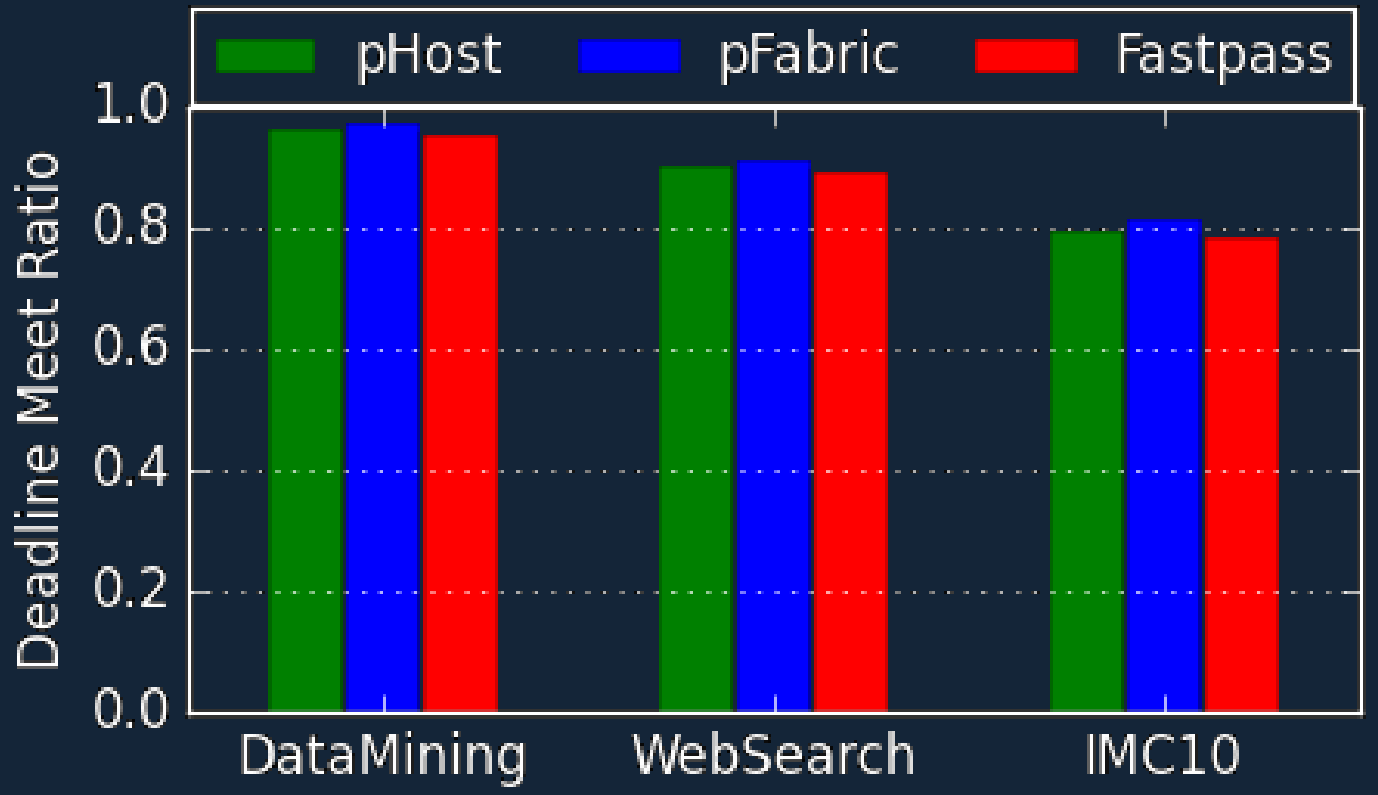



Similar performance for all flows (even for Fastpass)

Because throughput of more related to long flow performance

# Deadline

- Exponential deadline distribution with mean 1000us
- At least 1.25x optimal FCT




All protocols have similar performance