Prism: A Proxy Architecture for Datacenter Networks

Yutaro Hayakawa (Keio University)
Lars Eggert (NetApp)
Michio Honda (NEC Laboratories Europe)
Douglas Santry (NetApp)

Background

- TCP proxy plays important roles in modern cloud services

- Web cache
- Application level firewall
- Application level load balancer

Problem

- Aggregated bandwidth of backends are always constrained by proxy bandwidth

Problem

- Aggregated bandwidth of backends are always constrained by proxy bandwidth

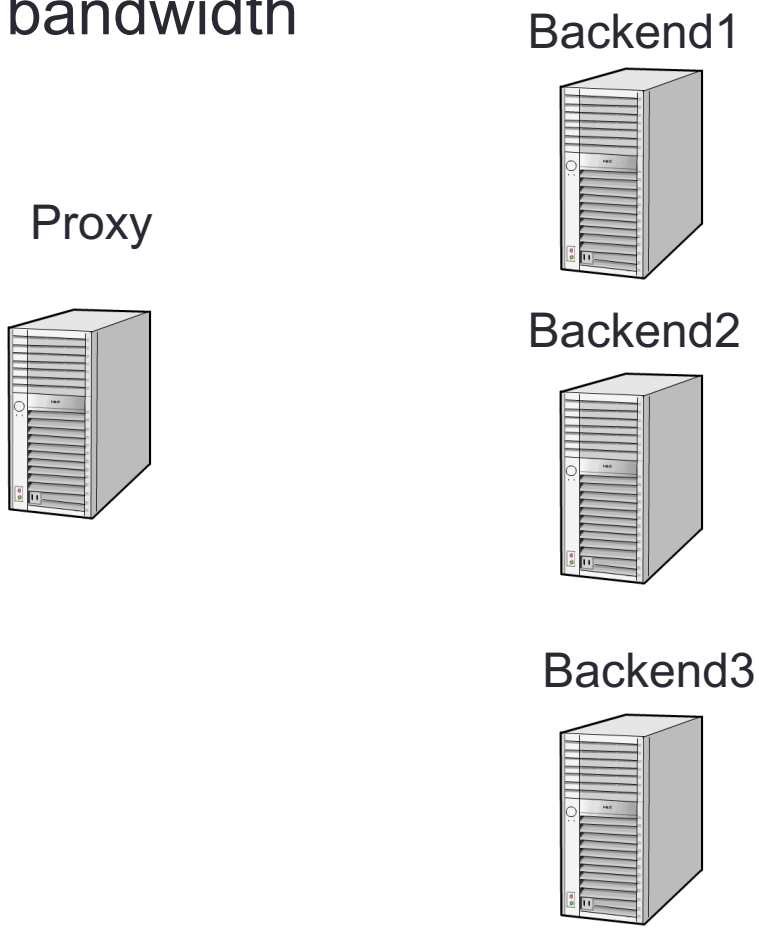
Backend1

Backend2

Backend3

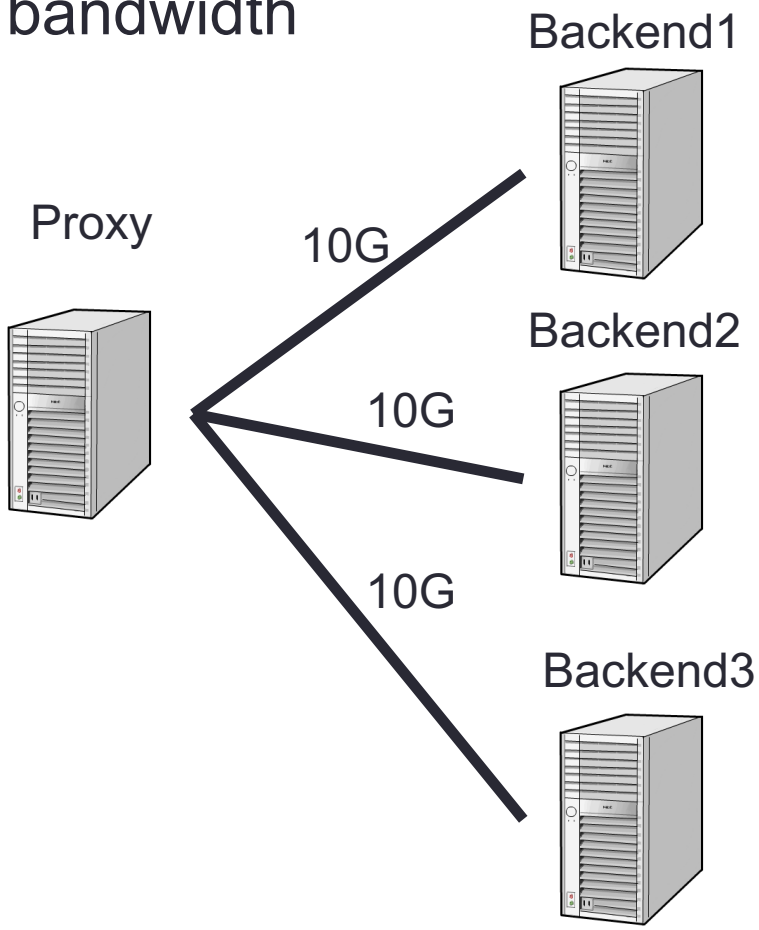
Problem

- Aggregated bandwidth of backends are always constrained by proxy bandwidth



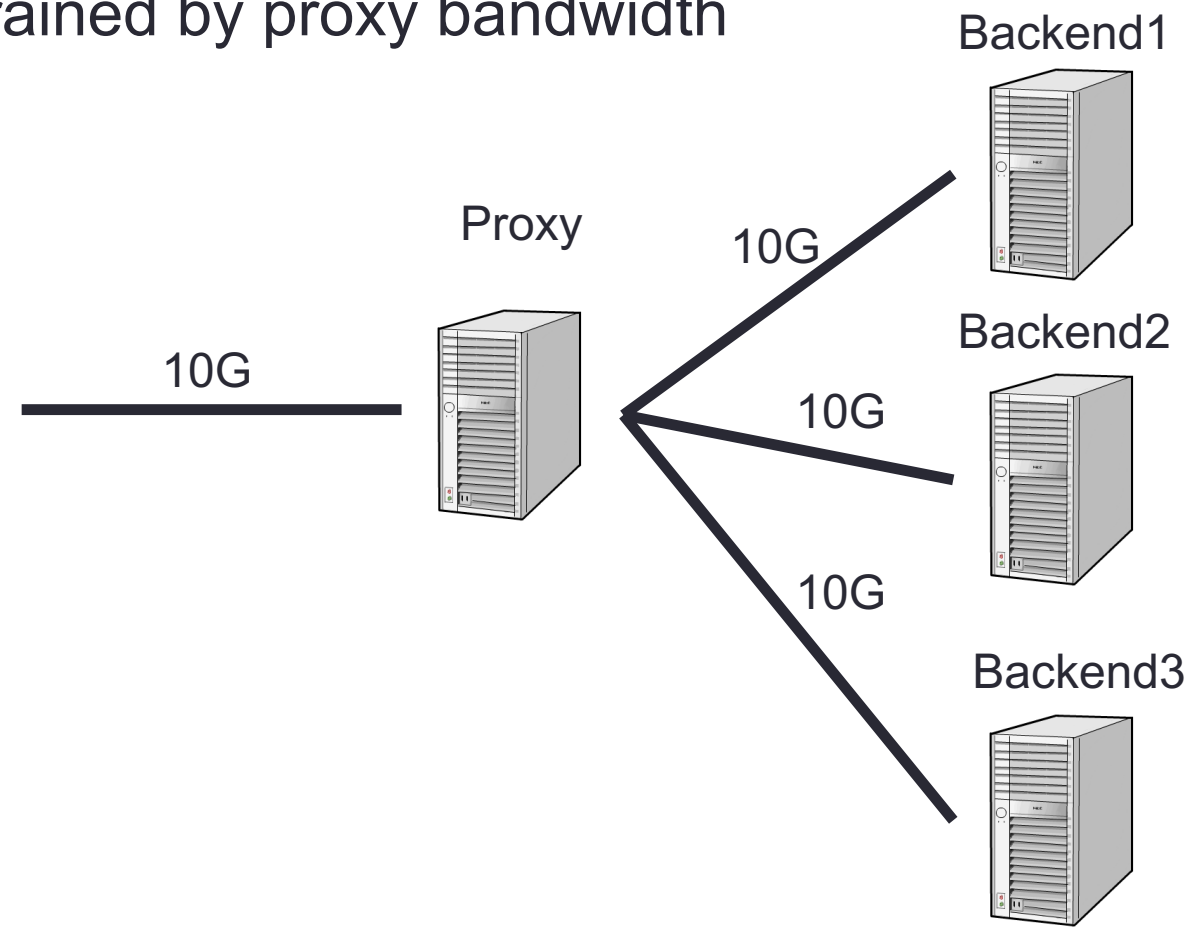
Problem

- Aggregated bandwidth of backends are always constrained by proxy bandwidth



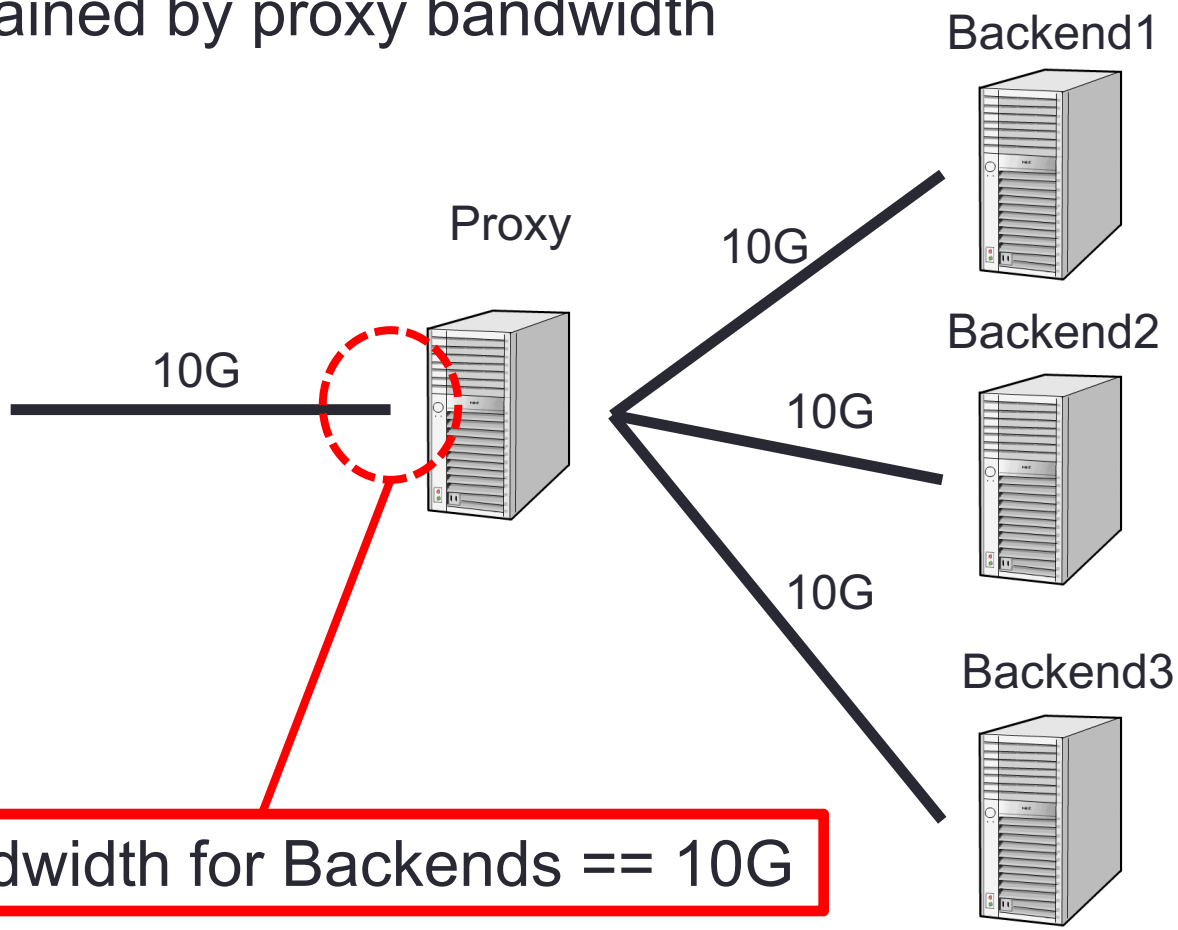
Problem

- Aggregated bandwidth of backends are always constrained by proxy bandwidth



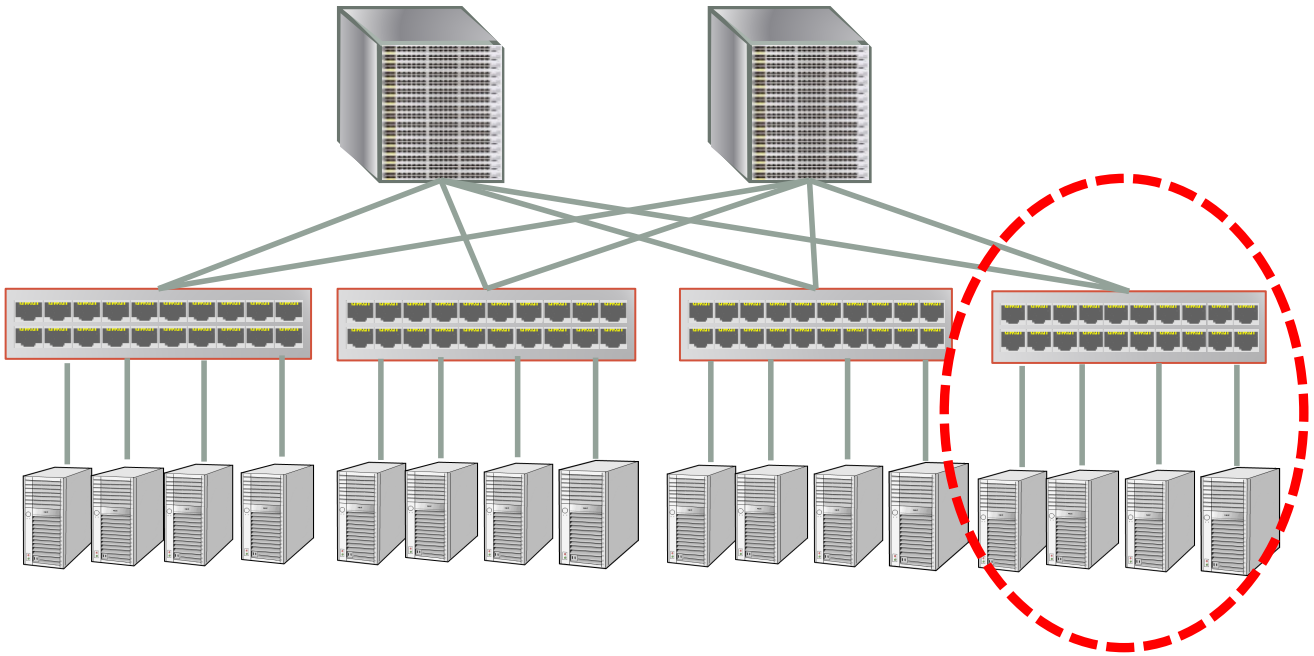
Problem

- Aggregated bandwidth of backends are always constrained by proxy bandwidth



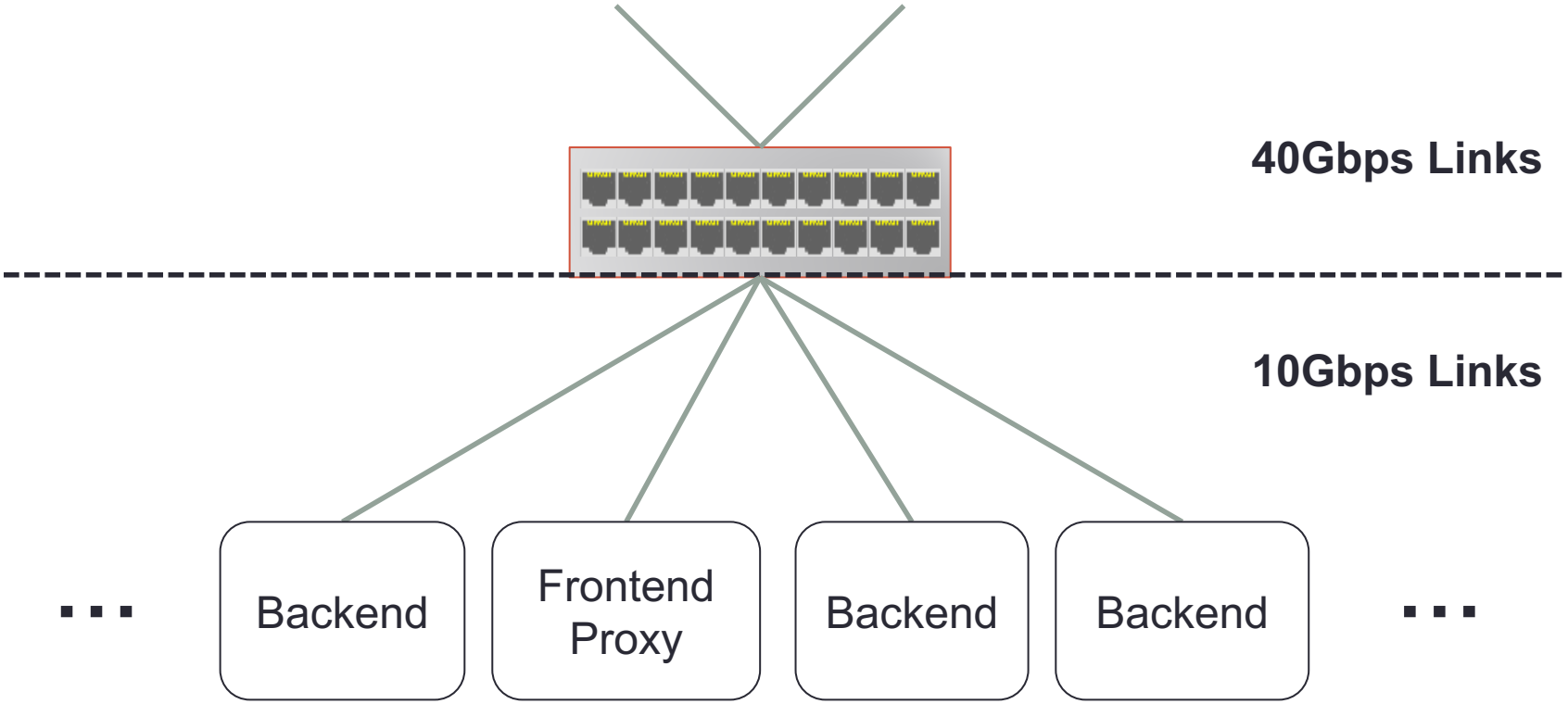
Why does it matter?

- This problem is significant especially at the edge of today's datacenter topology

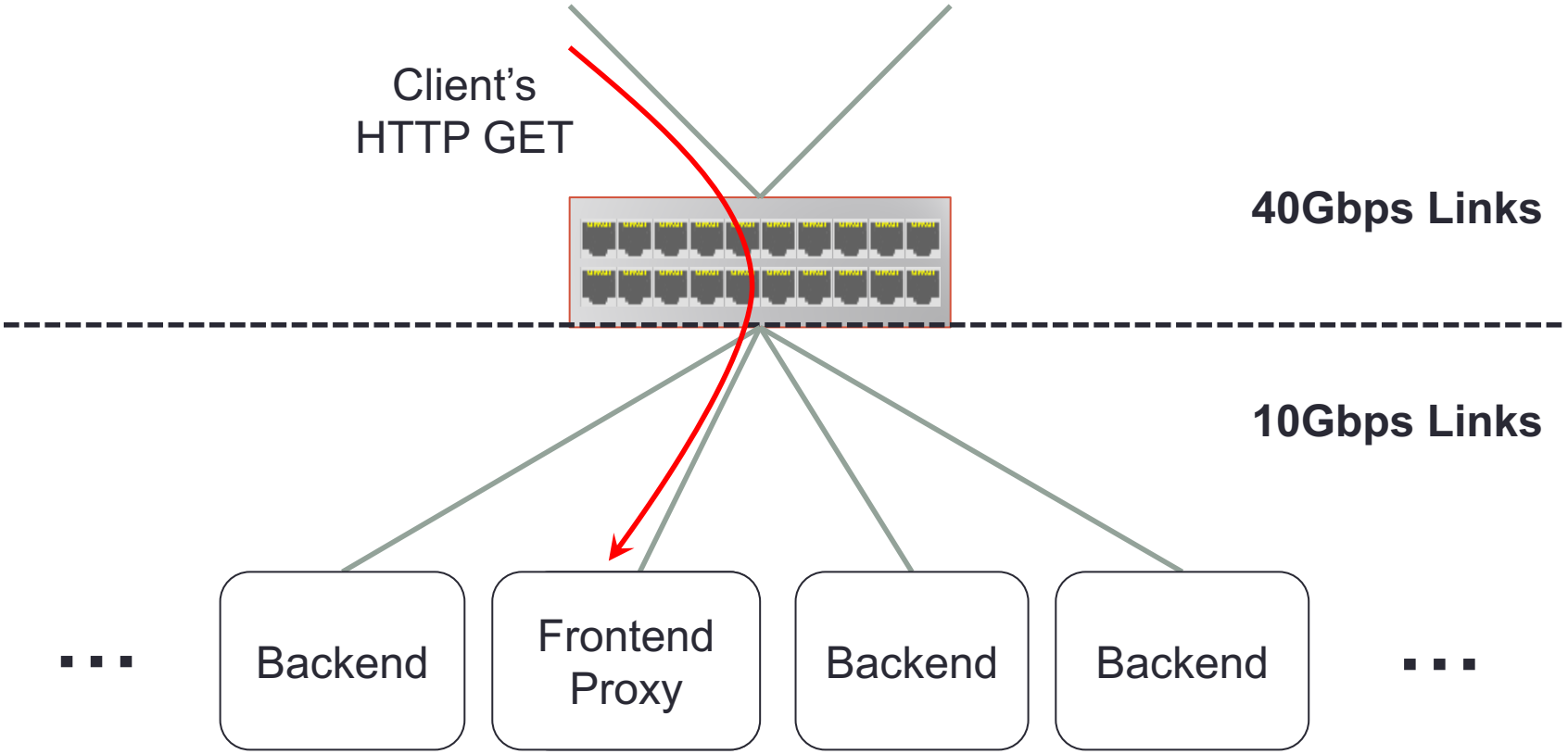


Example : HTTP/1.1

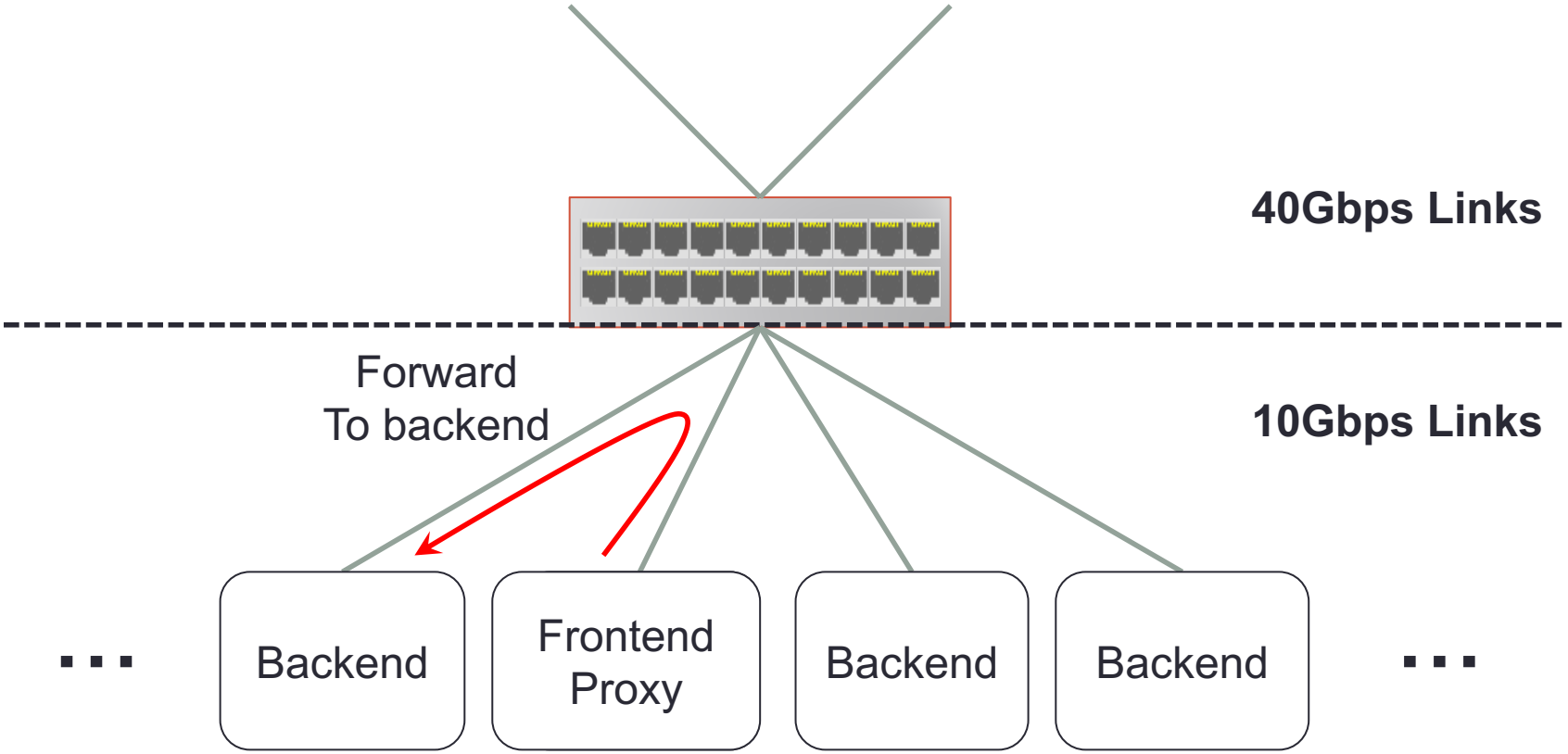
- Simple web system
  - One frontend proxy and multiple backend server



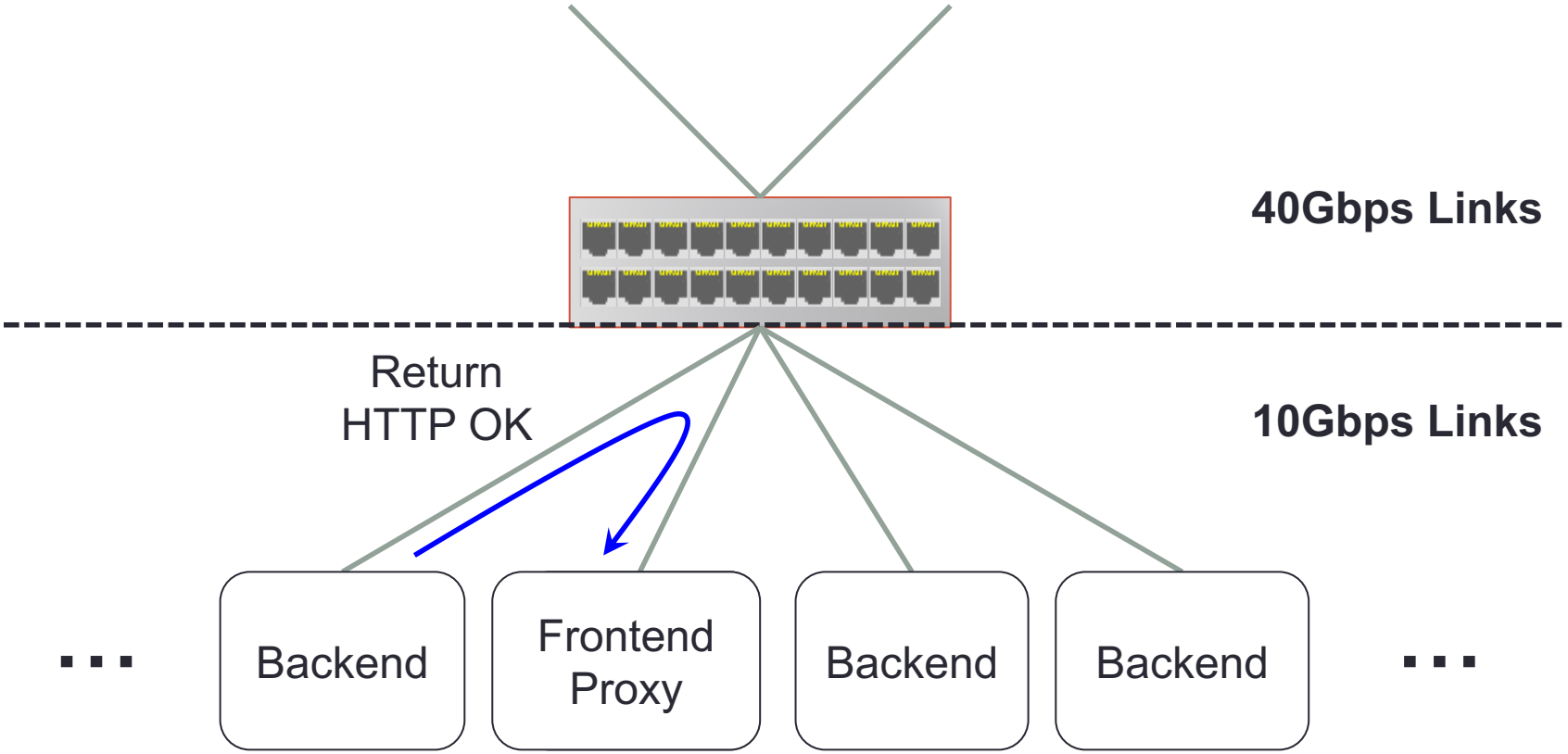
Example : HTTP/1.1



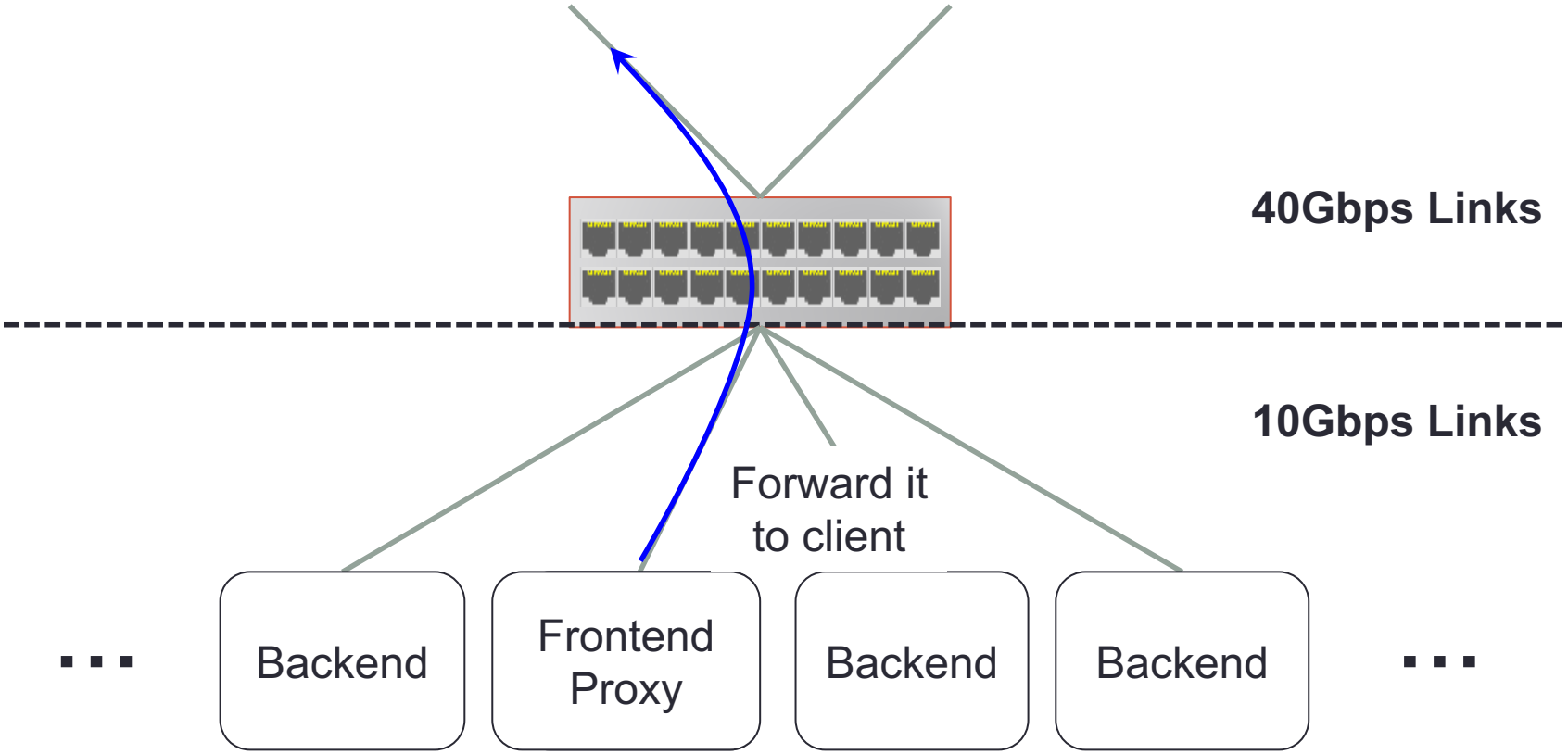
Example : HTTP/1.1



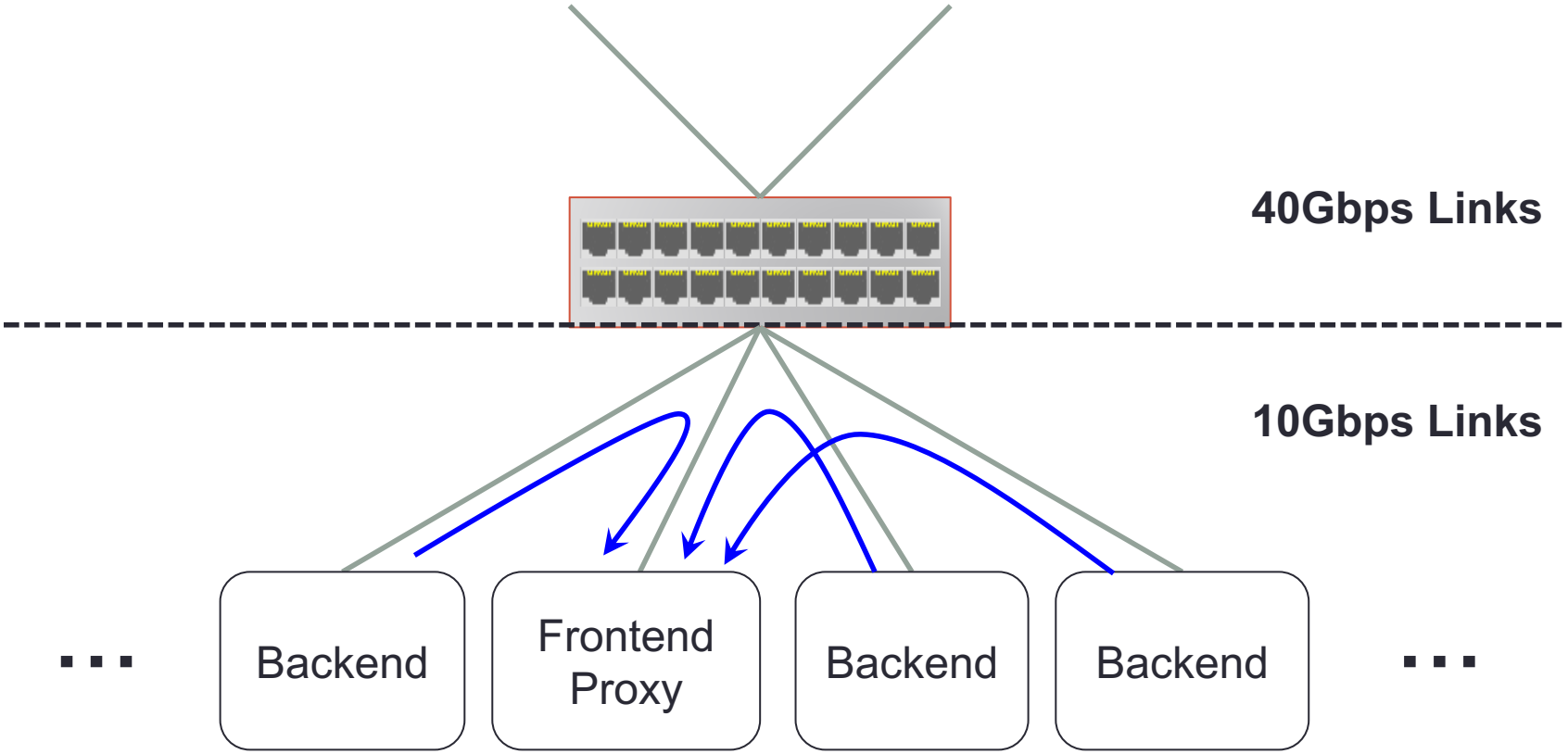
Example : HTTP/1.1



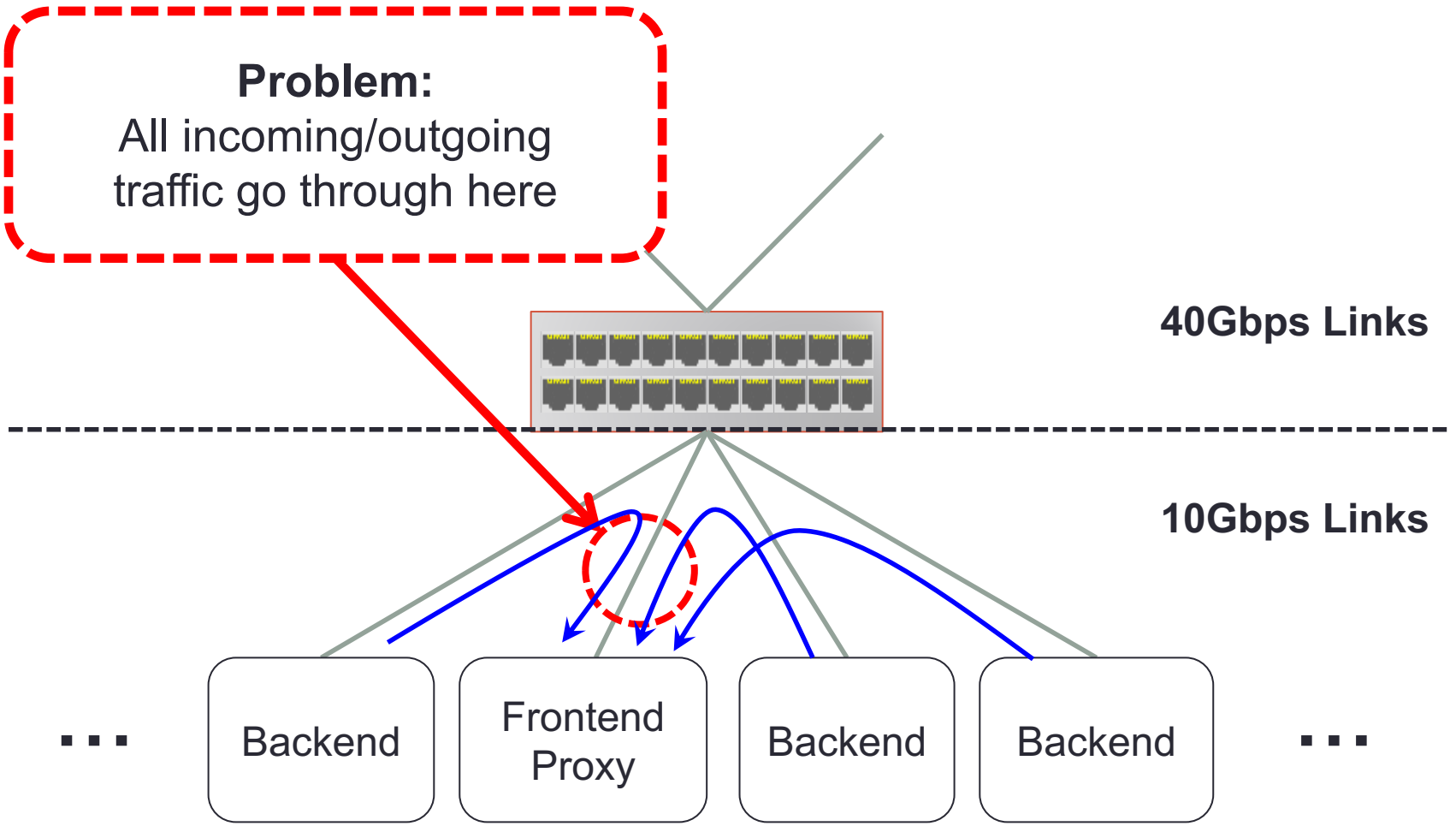
Example : HTTP/1.1



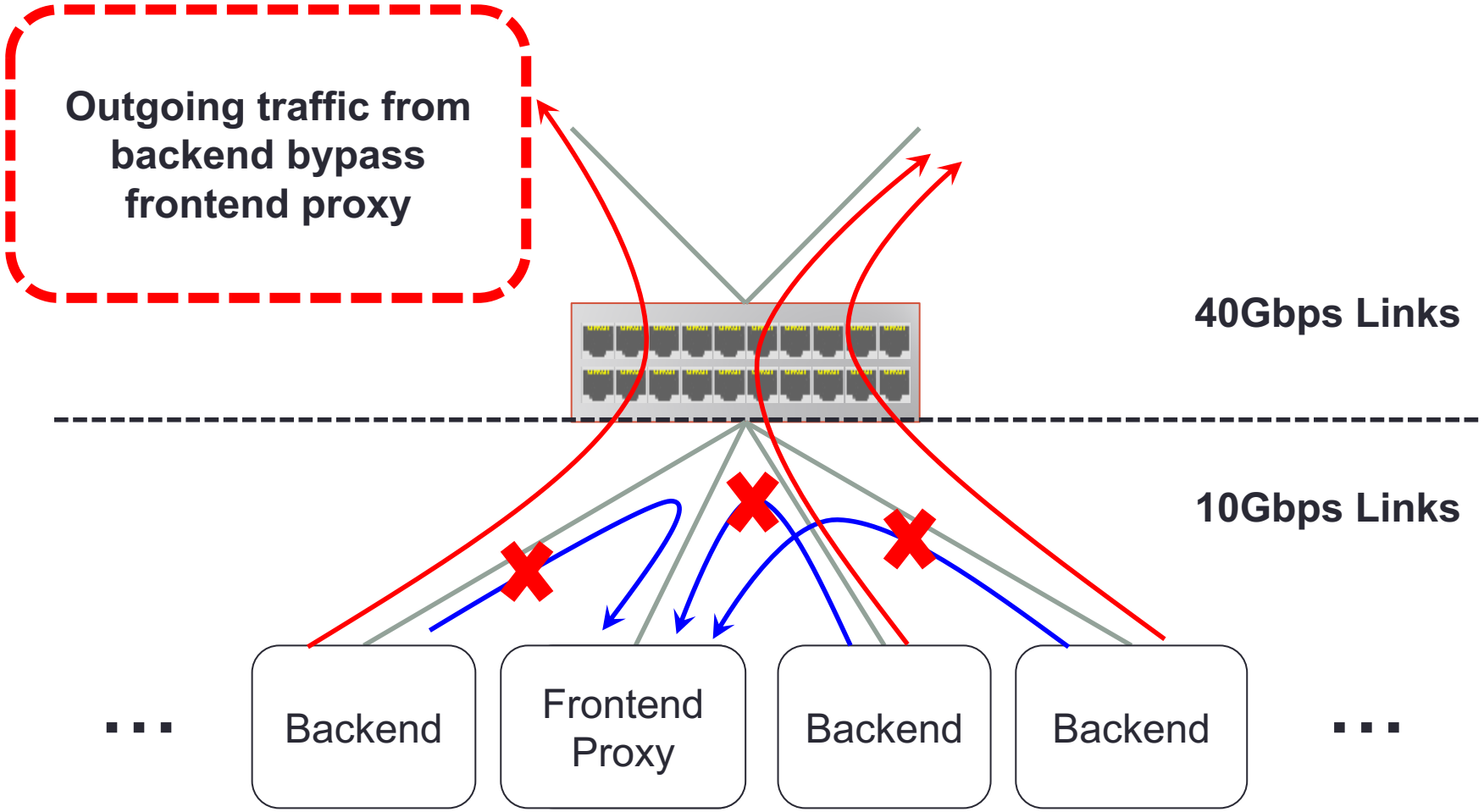
Example : HTTP/1.1



Example : HTTP/1.1

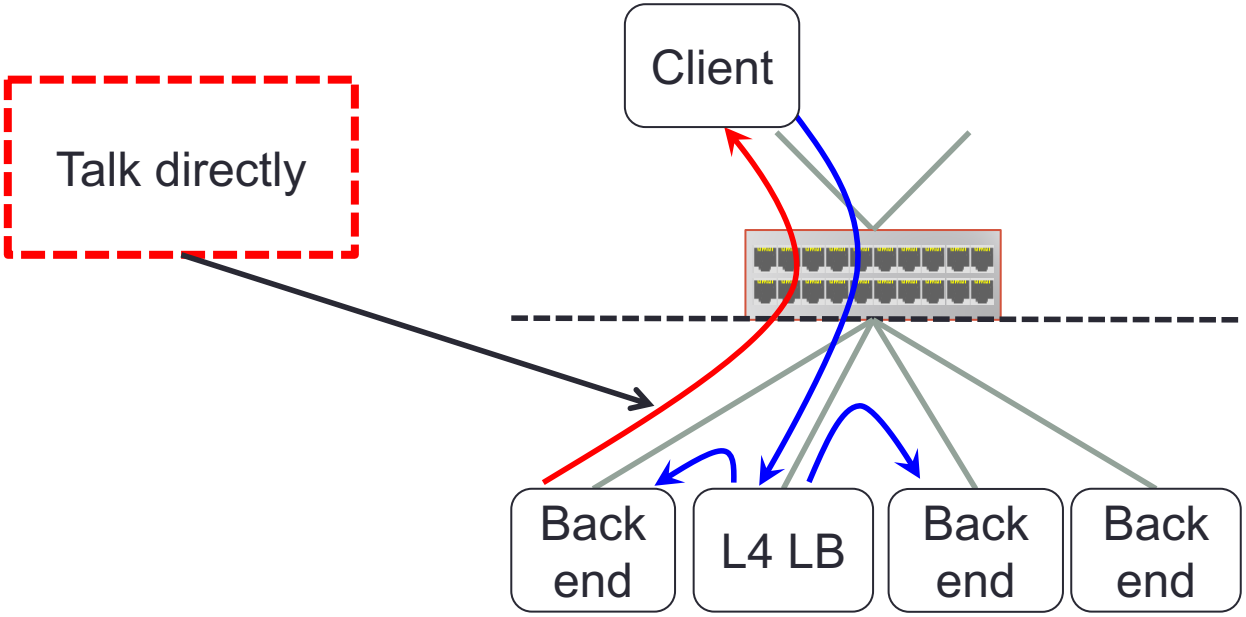


How can we solve this problem?



Related work

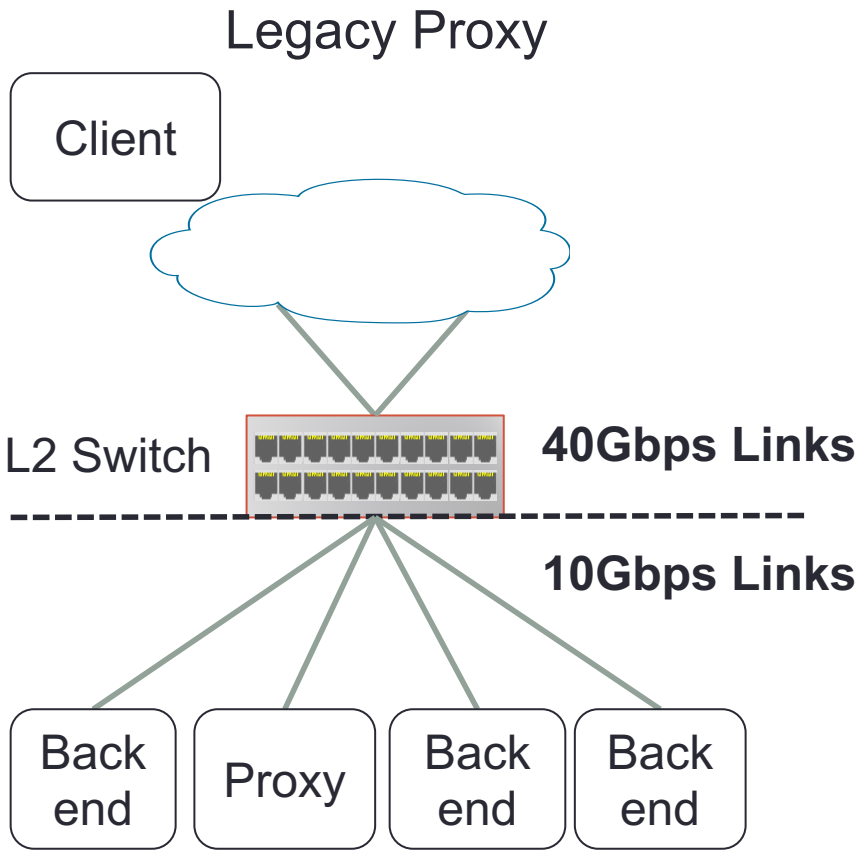
- L4 load balancer
  - Maglev[1], Ananta[2], Duet[3]



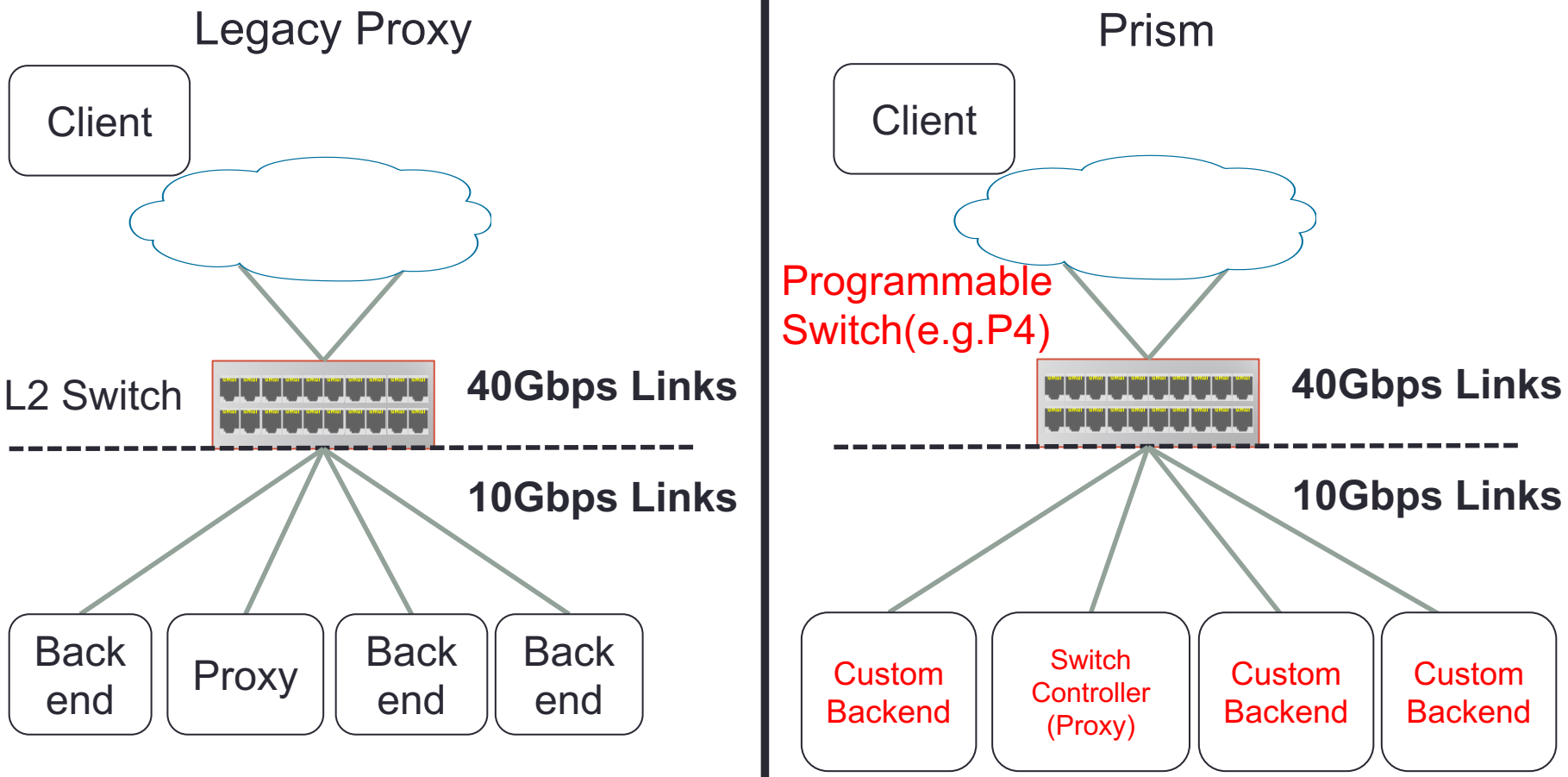
Balance TCP connect request

[1] D. E. Eisenbud et al. Maglev: A Fast and Reliable Software Network Load Balancer. *Proc. usenix nsdi*, 2016.
[2] P. Patel et al. Ananta: Cloud Scale Load Balancing. *Proc. acm sigcomm*, 2013.
[3] R. Gandhi et al. Duet: Cloud Scale Load Balancing with Hardware and Software. *Proc. acm sigcomm*, 2014.
[4] M. Alizadeh et al. CONGA: distributed congestion-aware load balancing for datacenters.
Proc. Acm sigcomm. Chicago, IL, USA, 2014, pp. 503–514.

Prism Architecture

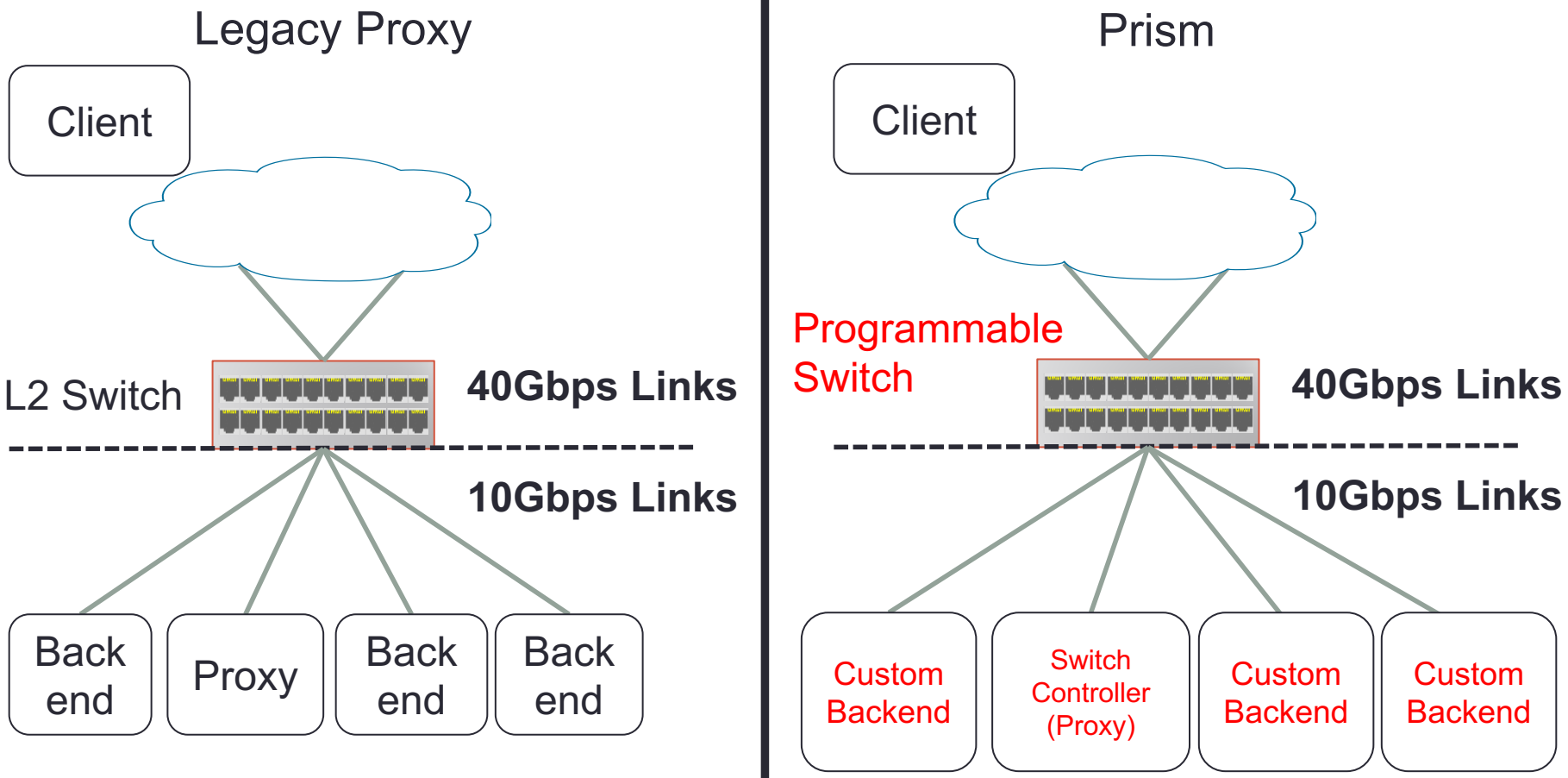


Prism Architecture



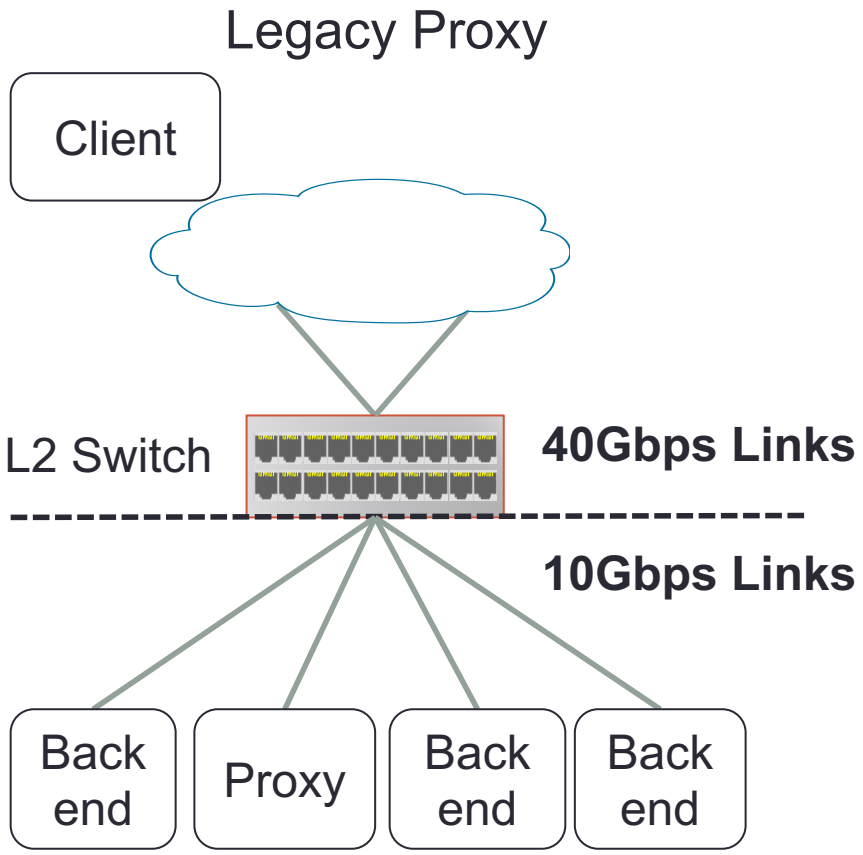
Prism Architecture

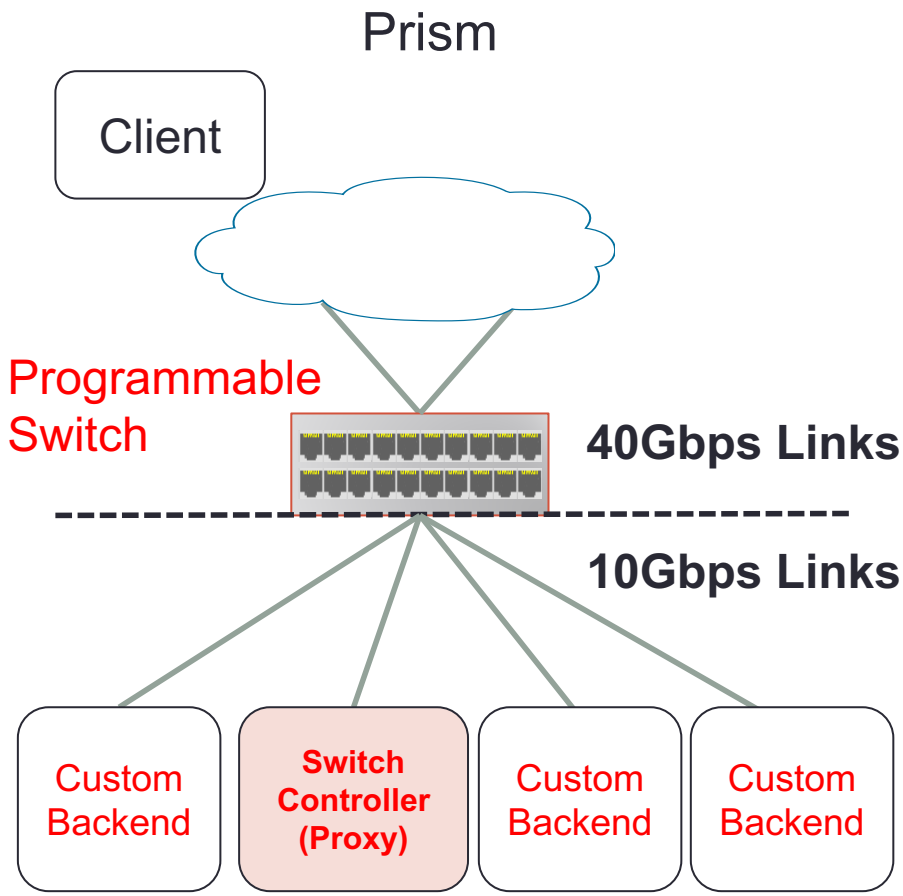
Basic idea: Offloading forwarding of application protocol payload to a switch



Prism Architecture

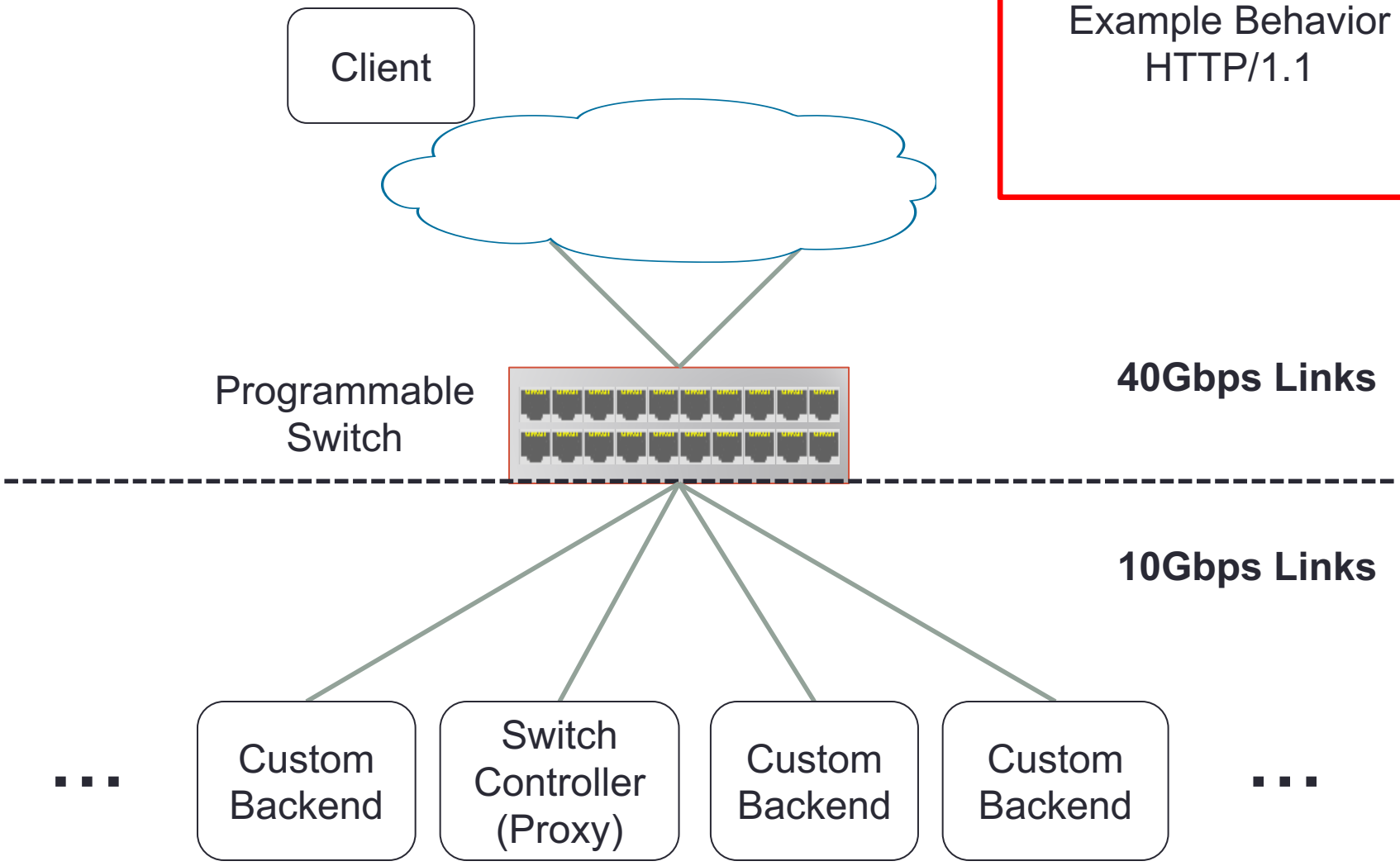
Basic idea: Offloading application protocol payload forwarding to a switch
=> the Switch Controller (Proxy) only handles header exchange





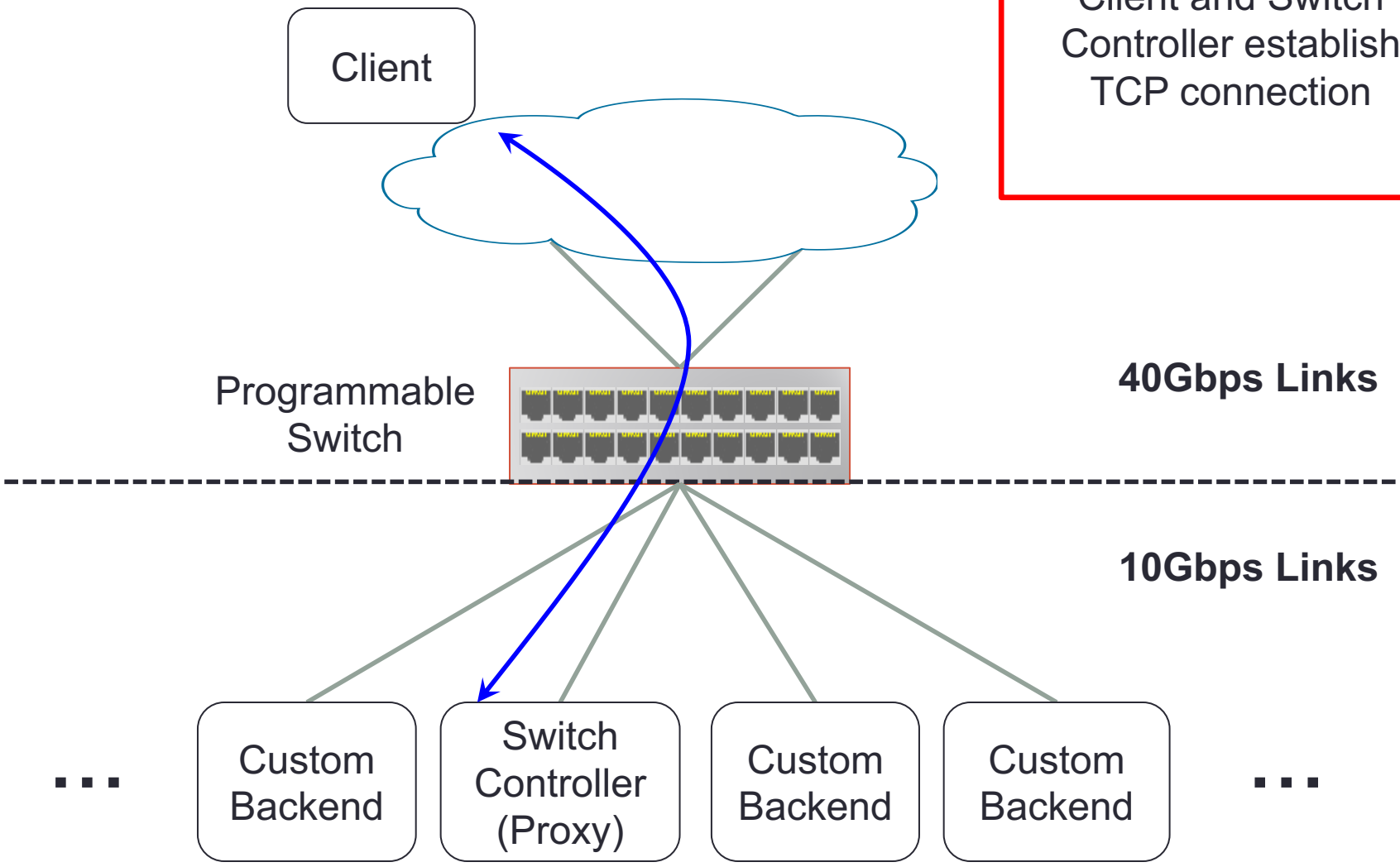
How does it work?

Example Behavior
HTTP/1.1



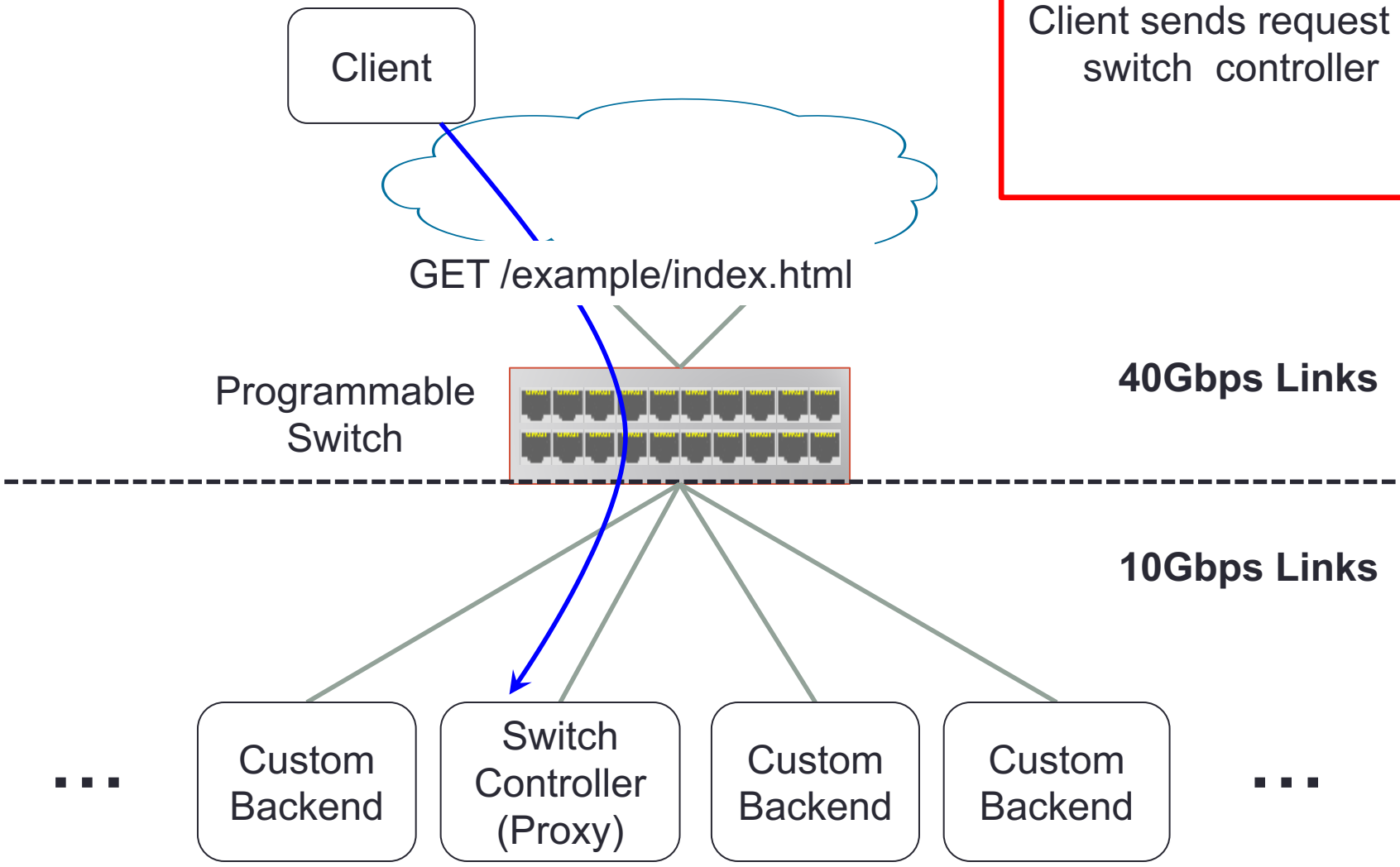
How does it work?

Client and Switch Controller establish TCP connection

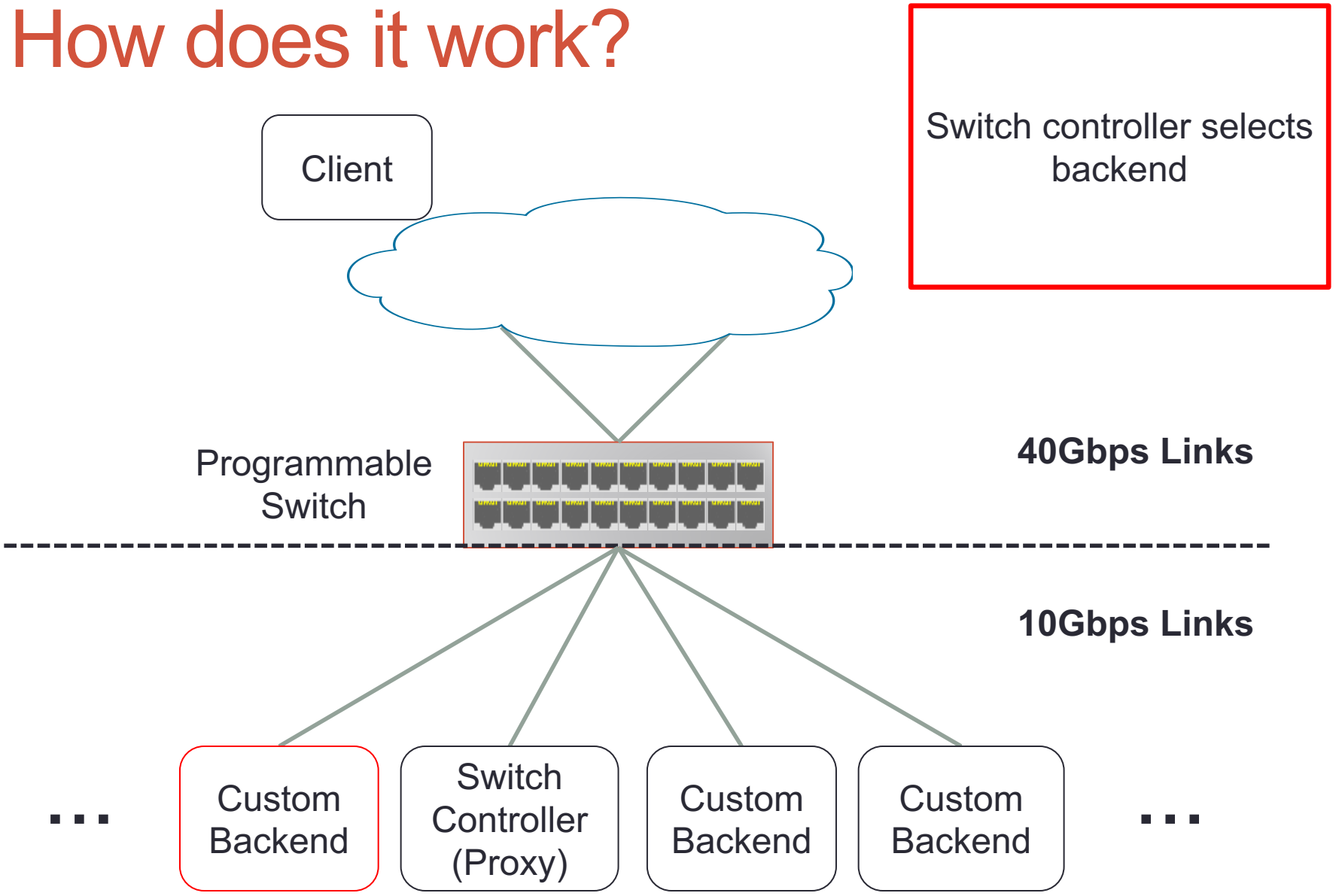


How does it work?

Client sends request to switch controller

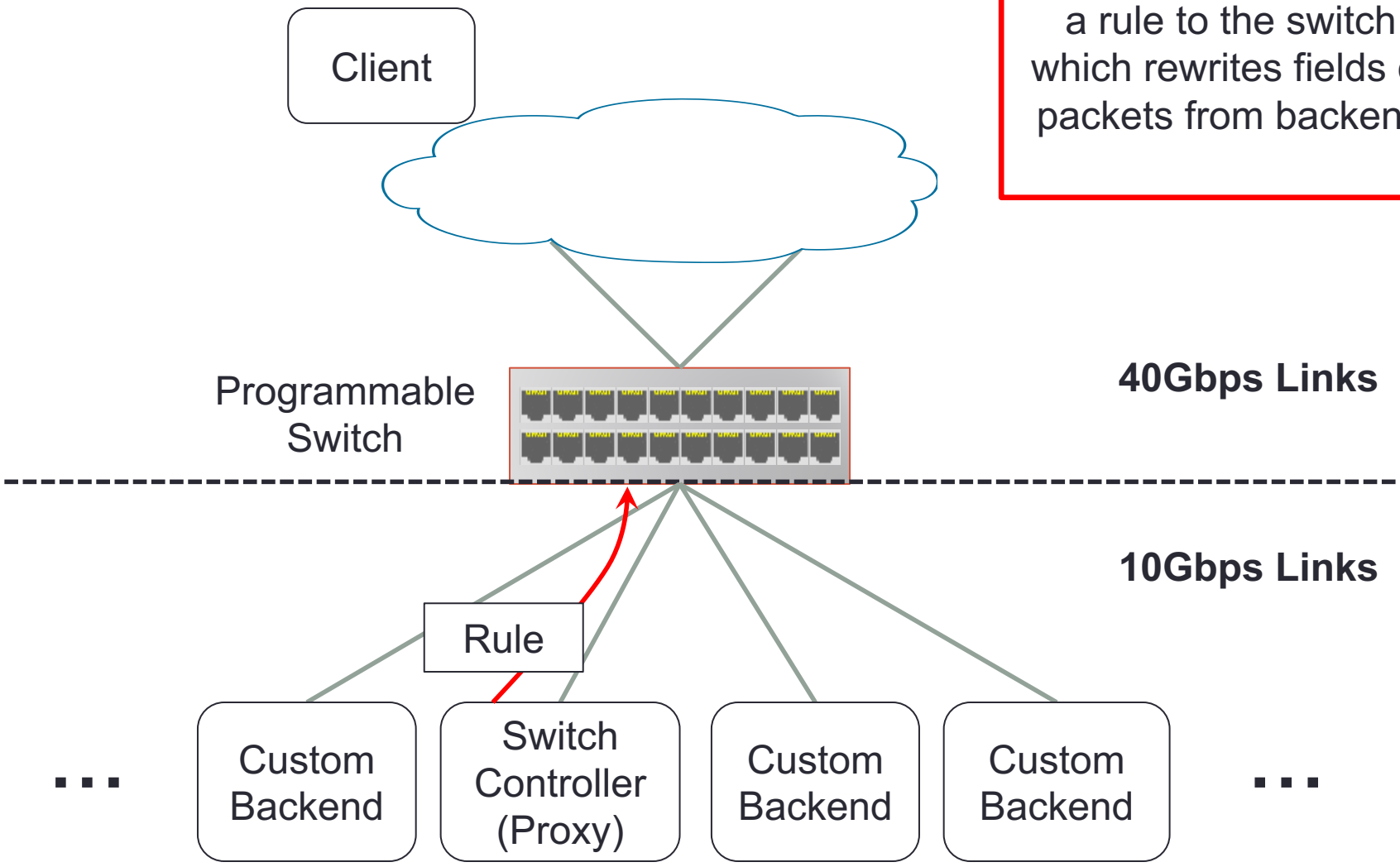


How does it work?



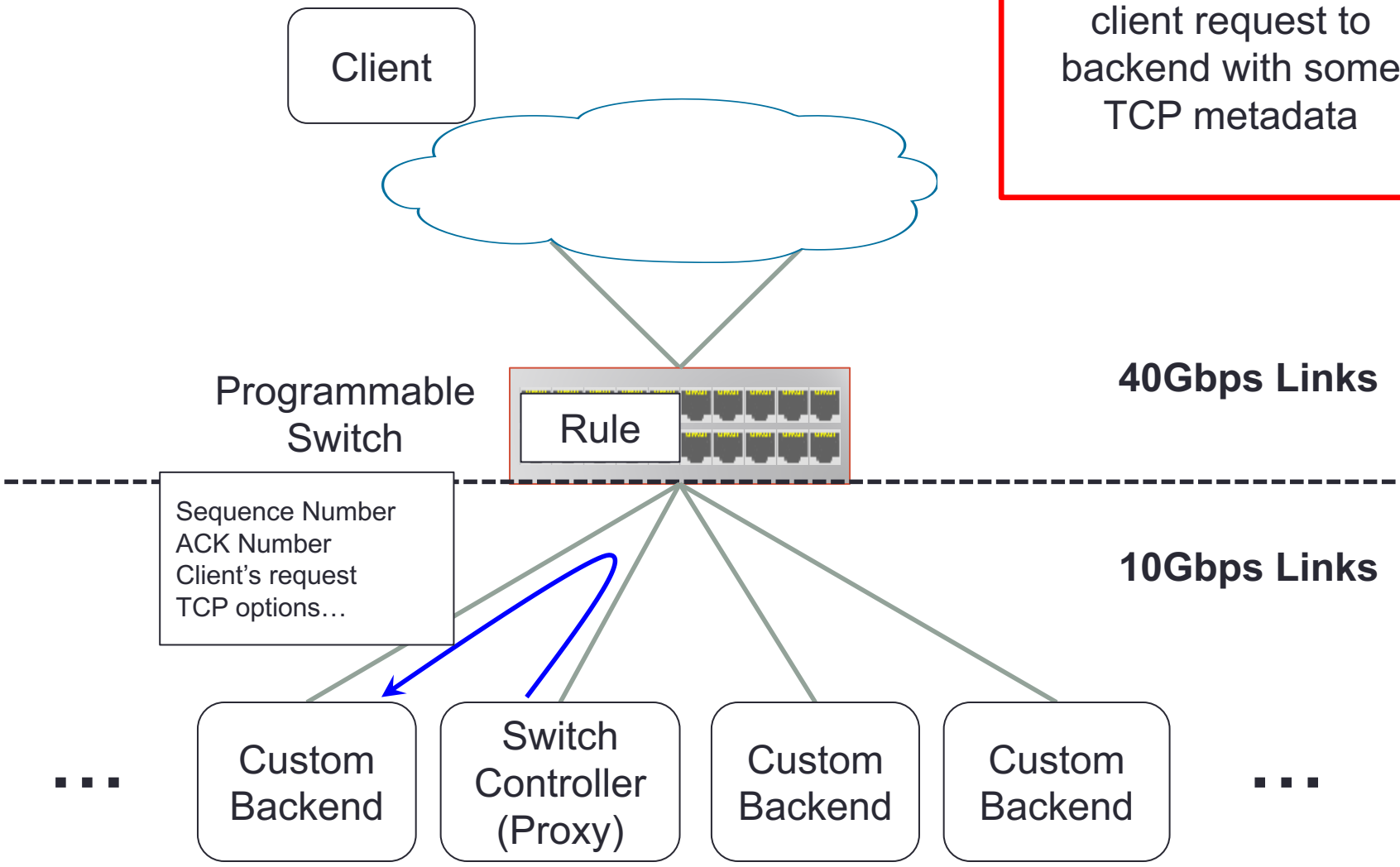
How does it work?

Switch controller injects a rule to the switch which rewrites fields of packets from backend



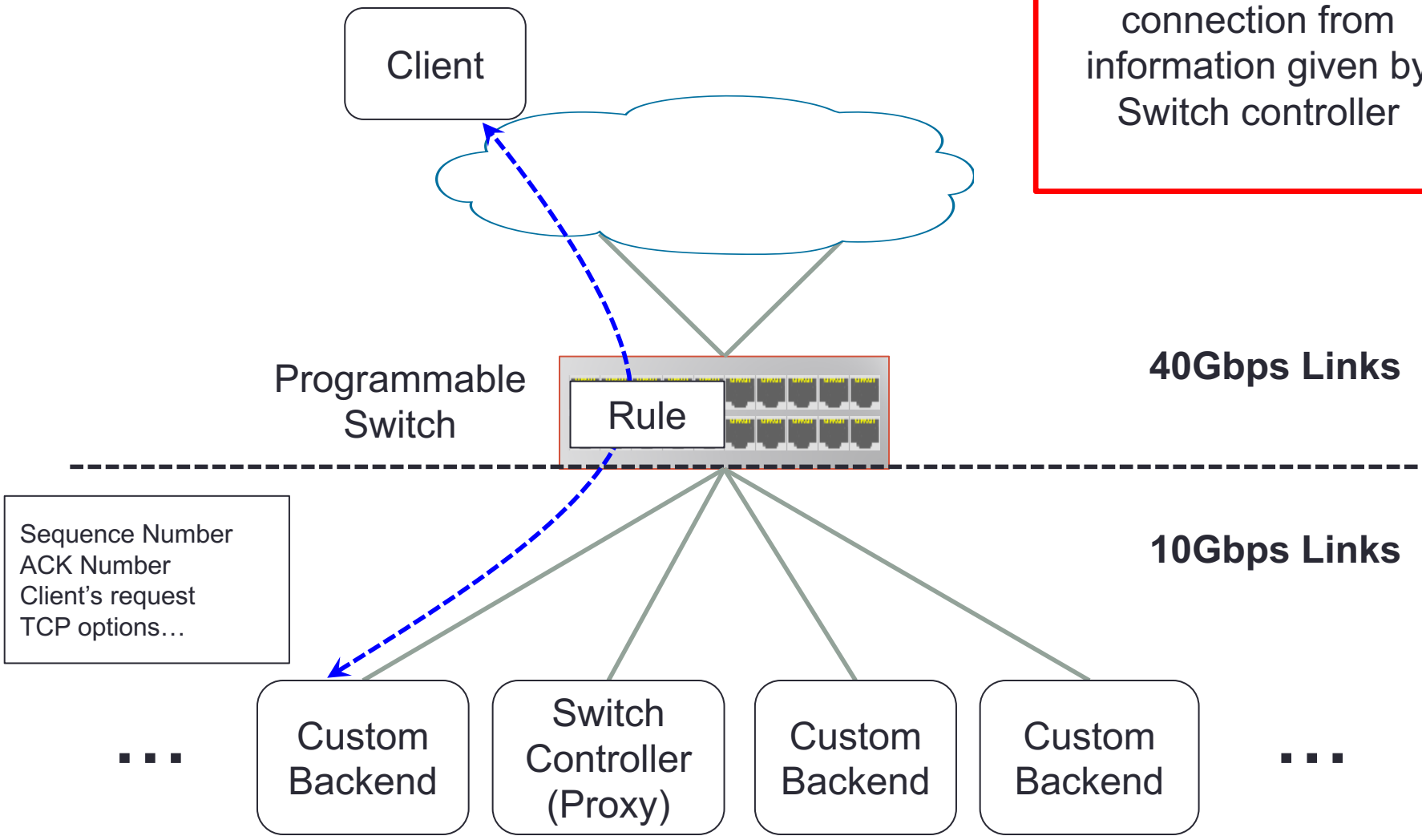
How does it work?

Switch controller sends client request to backend with some TCP metadata



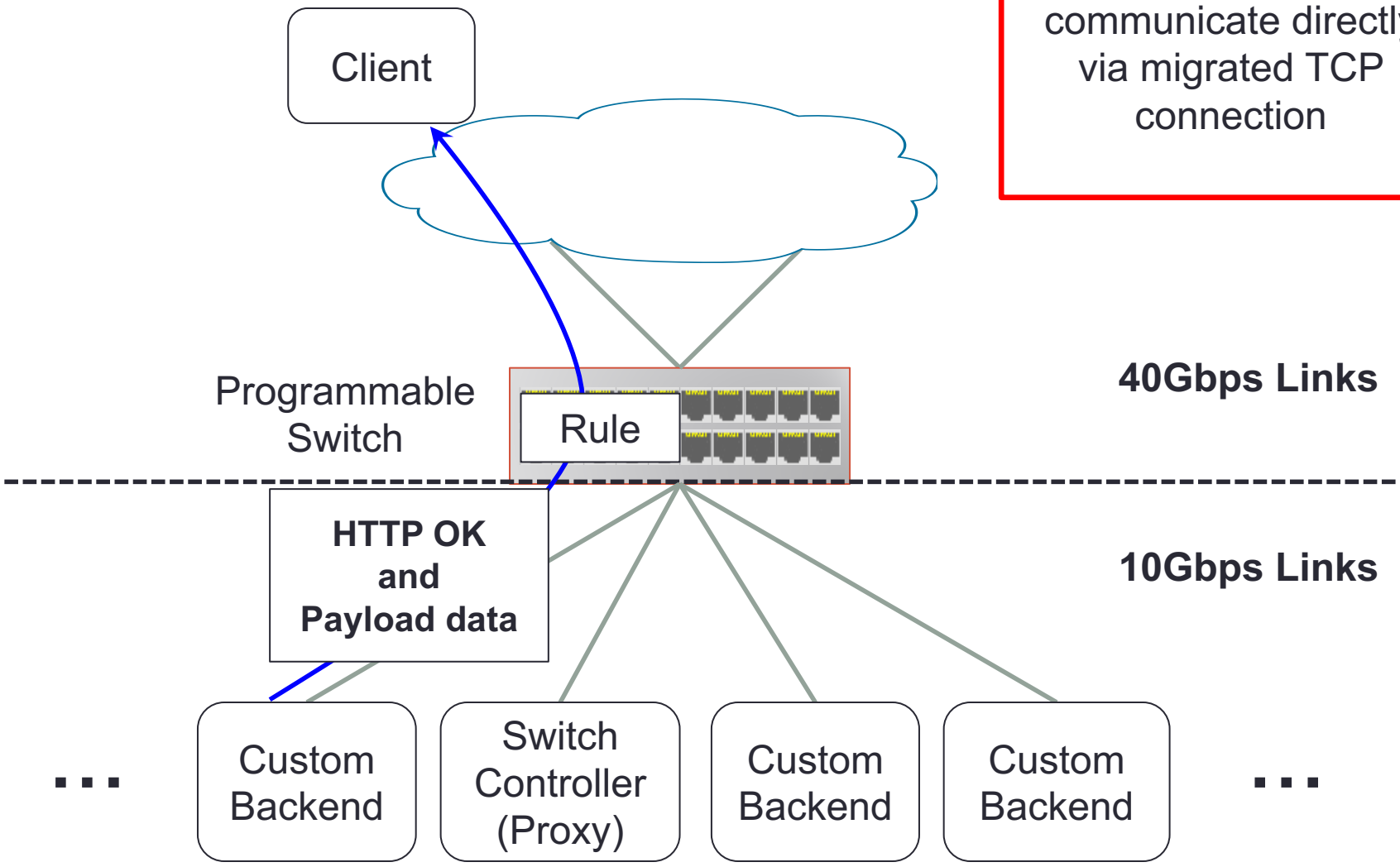
How does it work?

Backend migrate TCP connection from information given by Switch controller



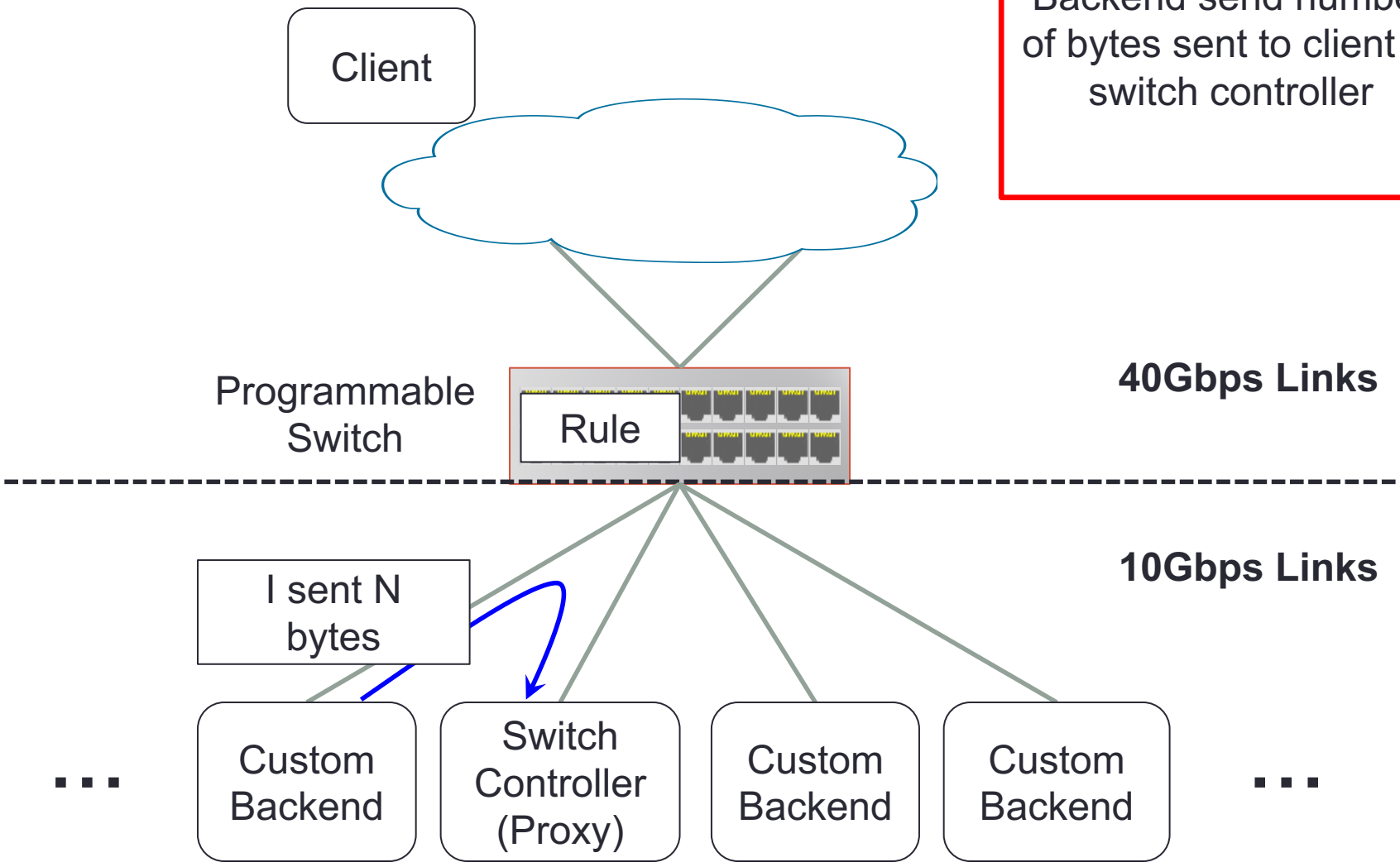
How does it work?

Backend and client communicate directly via migrated TCP connection

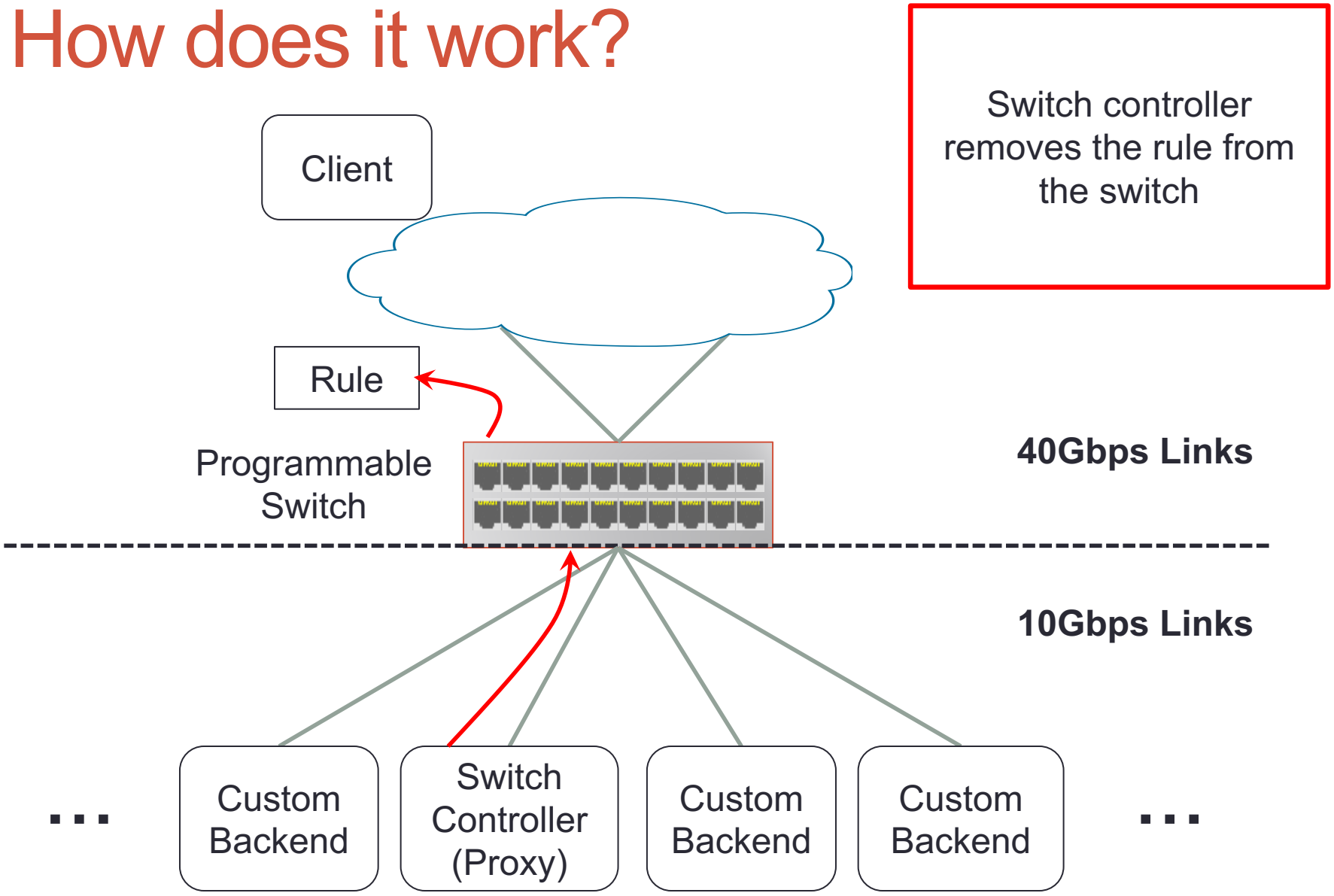


How does it work?

Backend send number of bytes sent to client to switch controller



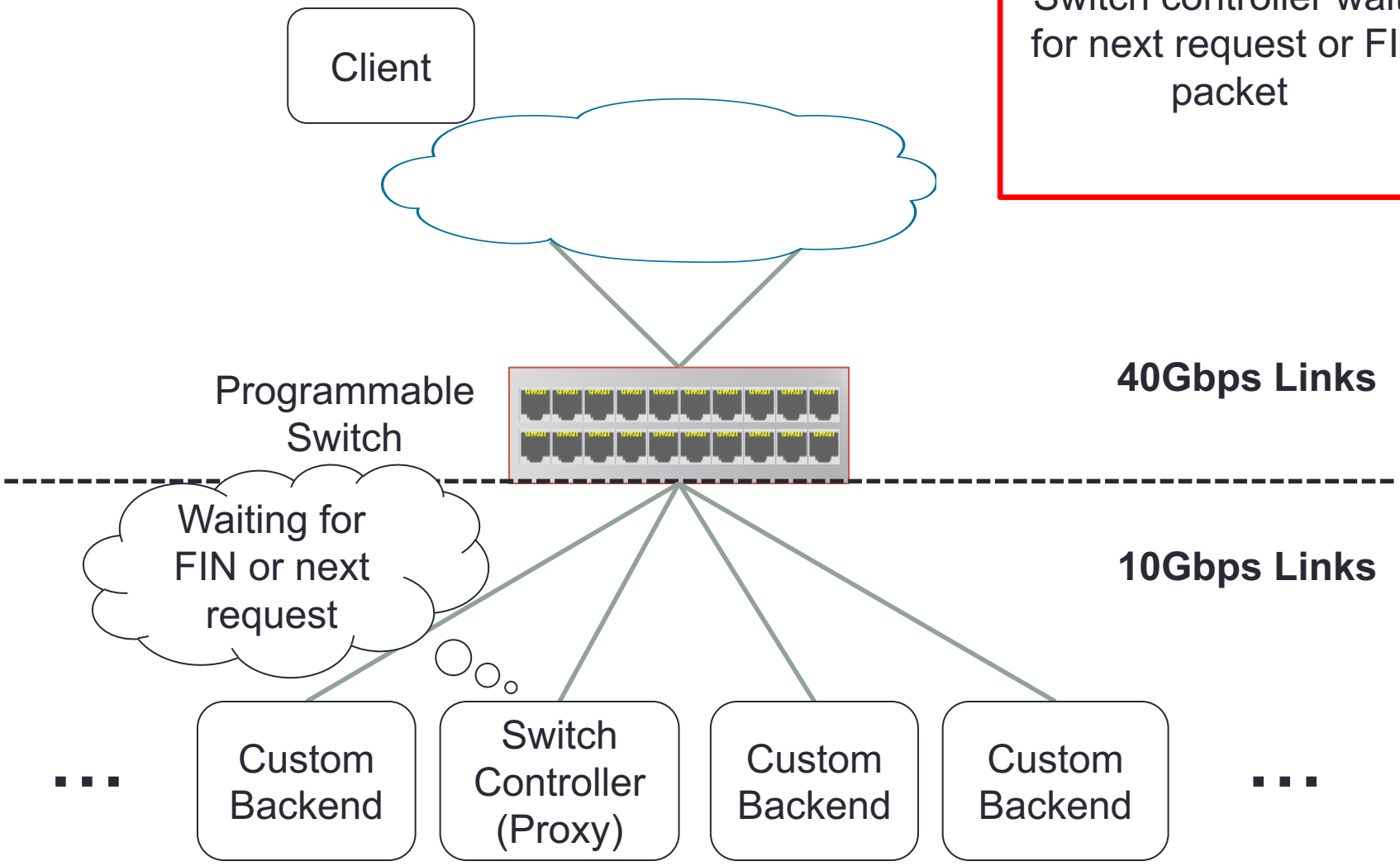
How does it work?



Switch controller removes the rule from the switch

How does it work?

Switch controller waits for next request or FIN packet



Challenge

- Why do we need to use programmable switch like P4?
- How can we stop pipelined HTTP request from client during TCP connection is migrated in backend?
- What happen when the client resets the connection?
- More details are in paper!

Evaluation

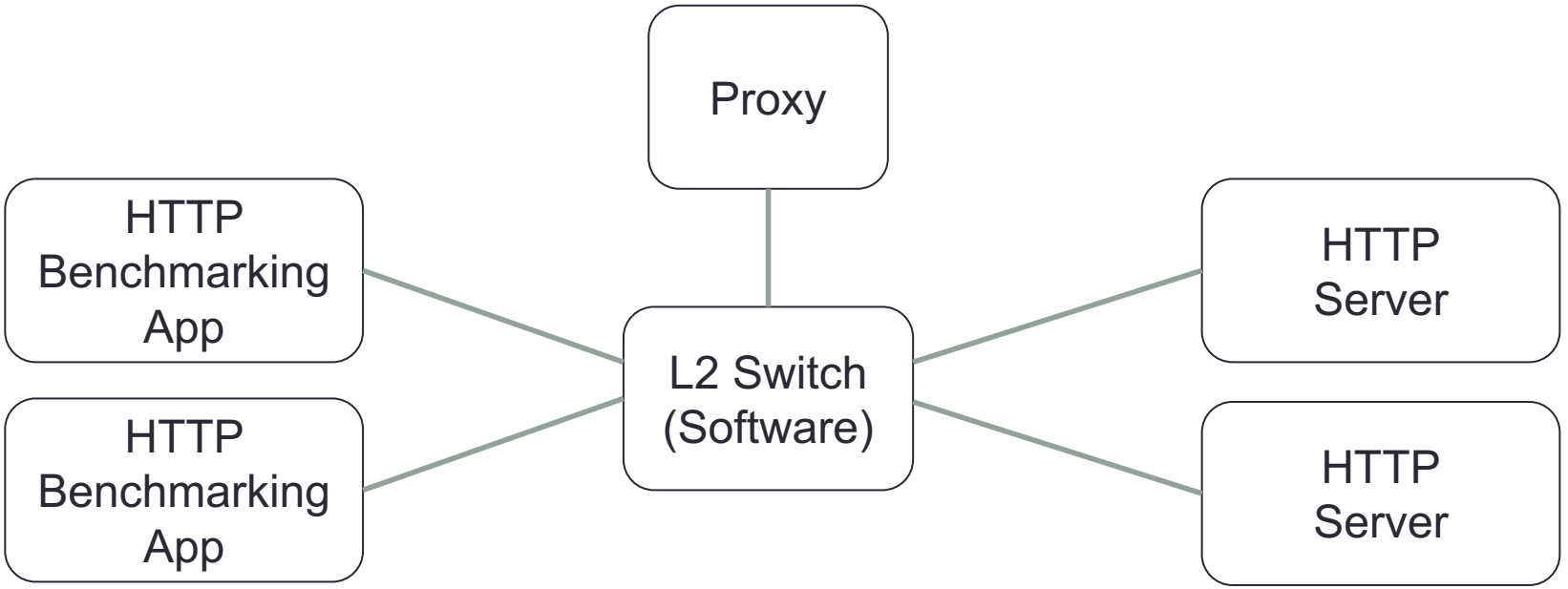
- We setup two experiments

1. End-to-End throughput comparison between legacy proxy and Prism for HTTP/1.1

2. Packet transformation performance measurement for Prism Switch

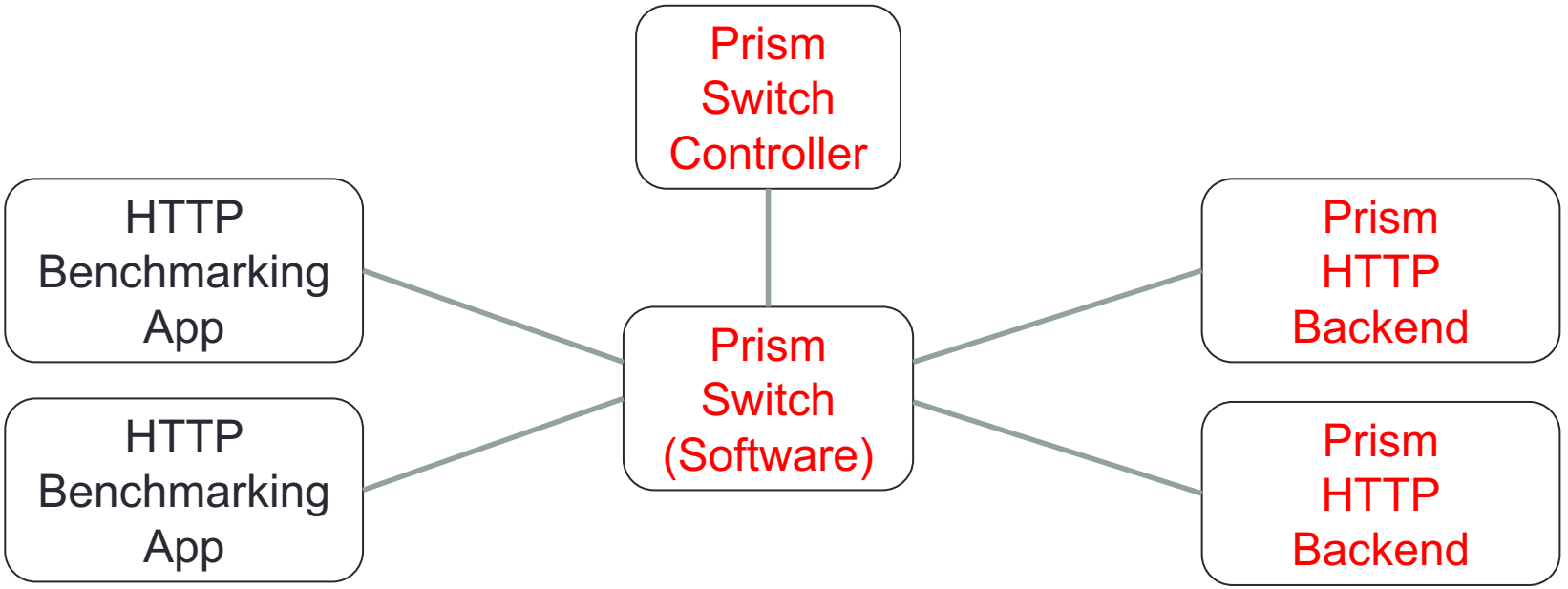
Evaluation: End-to-End Throughput

- 2 clients and 2 servers



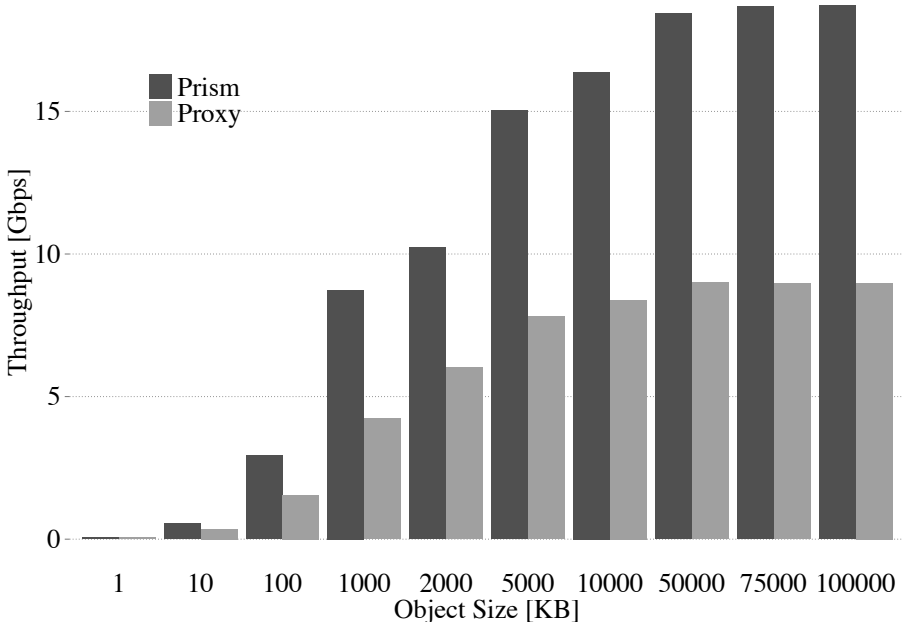
Evaluation: End-to-End Throughput

- 2 clients and 2 servers



Evaluation: End-to-End Throughput

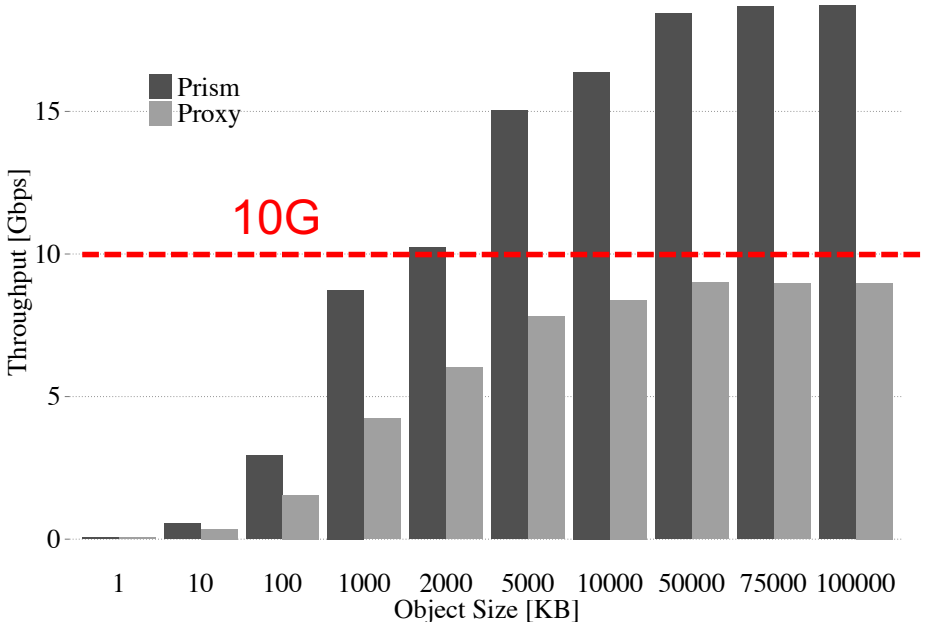
- Total throughput of proxy's clients are limited to about 9Gbps in maximum
- Total throughput of Prism's clients exceeds 10Gbps in 2MB data transfer
- Finally reached to about 18.7Gbps in maximum



Prism can use bandwidth of uplinks which proxy's clients can't use

Evaluation: End-to-End Throughput

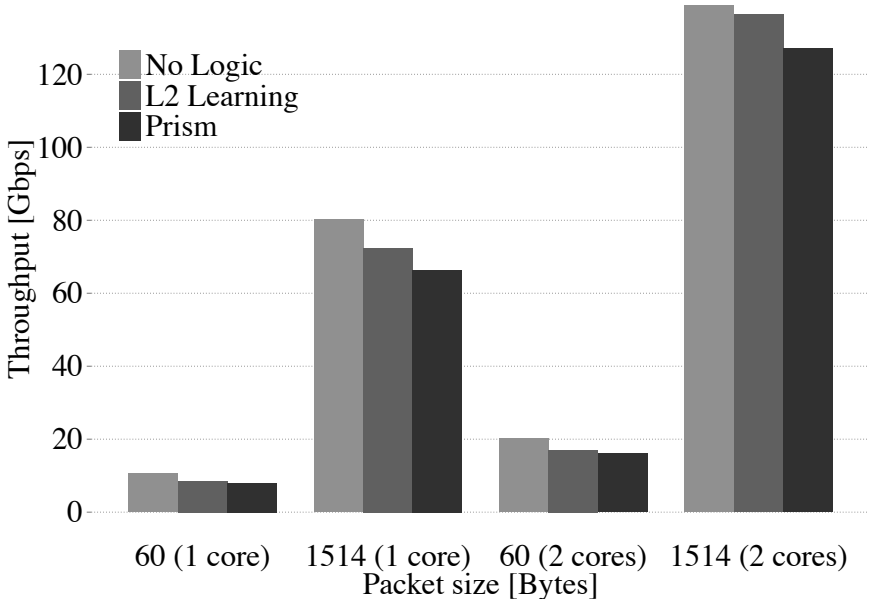
- Total throughput of proxy's clients are limited to about 9Gbps in maximum
- Total throughput of Prism's clients exceeds 10Gbps in 2MB data transfer
- Finally reached to about 18.7Gbps in maximum

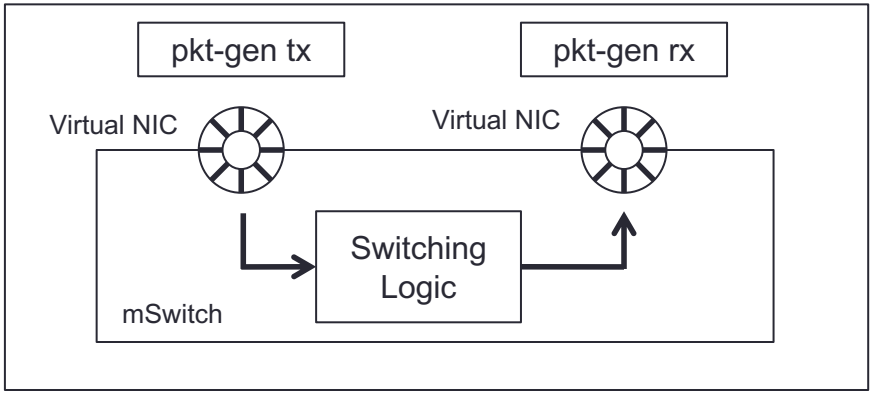


Prism can use bandwidth of uplinks which proxy's clients can't use

Evaluation: Prism Switch Performance

- Measure performance of the Prism switch packet transformation
- Prism switch achieves
  - 63Gbps in 1514byte packet using 1 core
  - 127Gbps in 2 cores





The Prism switch can achieve high throughput even if it is implemented as software

Conclusion

- Legacy proxy architecture can't utilize full bandwidth of switch uplinks in datacenters
- To solve this problem, we designed and implemented Prism which connects clients and backend servers directly during payload transaction of application protocol
- Prism improves bandwidth utilization

Question?