Orchestrating a brighter world

NEC

Cloud Federation

Peer Hasselmeyer, et al.
NEC Laboratories Europe

Acknowledgement – SSICLOPS

This presentation has received funding from the European Union's Horizon 2020 research and innovation programme 2014-2018 under grant agreement No. 644866.

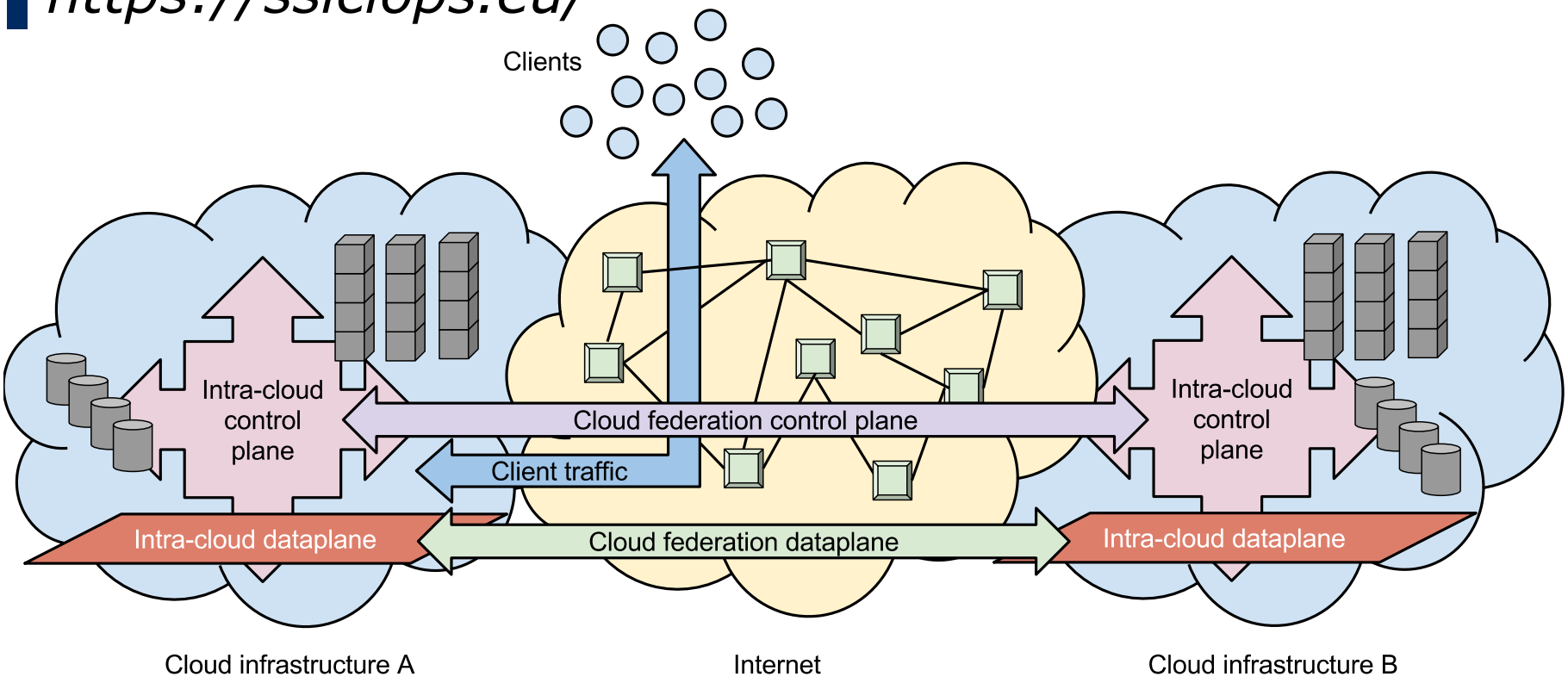
Disclaimer

This presentation reflects only the authors' views and the European Commission is not responsible for any use that may be made of the information it contains.

SSICLOPS H2020 Project

- "Scalable and Secure Infrastructures for Cloud Operations"
- Optimizing and securing packet transport in and across data centers
- *https://ssiclops.eu/*

SSICLOPS



Cloud Computing

- Trend towards migrating applications to the cloud
  - public vs. private
- Create VM
- Upload to Cloud
- Run VM
- Use VM
- → Perfect!

Cloud Computing – Issues

Single point of failure

- Application is hosted on remote server
- If cloud fails, application, data, ... becomes unavailable
- Same thing for the connection to the cloud

Latency

- Application is hosted remotely
- Farther away than hosted locally
- Depending on client location, latency will vary

Load bursts

- (Private) cloud infrastructure might not be able to handle large bursts in application usage

Can we do better?

Multi-Site Deployment – Solution?

Deploy application in multiple locations

- different private clouds
- different availability zones

Impact

- improves resilience against outages
- decreases latency
- increases resource pool

Drawback: Instances are separate islands

- No data sharing between instances
- Inconsistent data across locations
- No coordinated load-balancing across locations

Can we do better?

Multi-Site Deployment – Connectivity?

- Application instances need to be connected to share load and data
- Could be done with WAN and public IP addresses
- Most applications expect being used on private networks, though
  - assumes trust and attack protection
  - no traffic encryption
  - no traffic filtering and firewalling between components
- Solution: tunnels, VPNs, ...
  - to securely extend private networks across locations
  - to avoid applications needing to be extended for public network environments

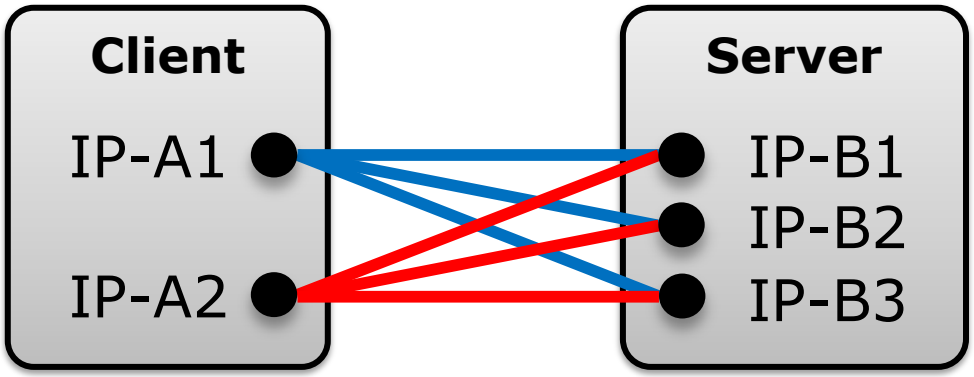
Federated Cloud Networks

- Deployment on multiple sites increases resilience
- Connecting the sites ensures data consistency
- But: connectivity between sites is still single point of failure
- Typically, data centers have multiple uplinks
- Fail-over mechanism in place
- Issues
  - Fail-over mechanism needs to be engineered specifically
  - No aggregation of bandwidth of uplinks
  - Maybe assigning flows to uplinks
    - on a flow basis, not individual packets
    - worst case: all (large) flows end up on same link
  - When private traffic is put into tunnels, tunnel will hide IP/port diversity (single source, single destination → single uplink)

Can we do better?

Digression: MPTCP

- MPTCP: MultiPath TCP
- Splits up TCP connections into multiple subflows
- Each subflow behaves like a separate TCP flow
- Client initiates subflows
  - by default in mesh-like fashion across all local and all remote IP addresses
- Server tells client about additional IP addresses



MPTCP Benefits for Cloud Interconnection

- Can make use of multiple paths
- Spreads packets to path on individual basis
- Automatically adjusts to current capacity of paths
  - including reduction to zero if path becomes unavailable
- Nicely coexists with regular TCP traffic
- Nicely coexists with existing middleboxes (routers, firewalls, DPIs, ...)

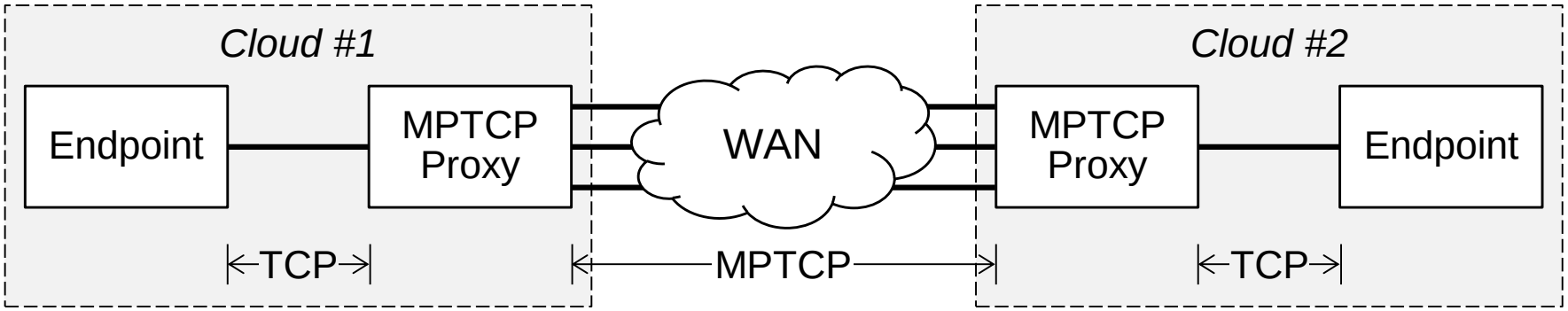
→ ***How to implement?***

MPTCP Use by Cloud Applications

- Extend applications with MPTCP and we're done?
- Endpoints (client/server) need MPTCP-capable kernel
  - not part of standard distributions
  - for Linux, kernel needs to be patched and compiled
  - difficult for already existing applications/VMs
- VMs in cloud do not know about multiple uplinks
  - typically have single virtual NIC connected to "the network"
  - only have single IP address

MPTCP Proxy

- Leave applications alone and introduce MPTCP at infrastructure level
  - like a tunnel, but with different capabilities
- Idea: add component which transforms TCP streams into (multiple) MPTCP ones
- Infrastructure knows about multiple uplinks



MPTCP Proxy: Direct Mapping

Take each TCP packet and transform into MPTCP one

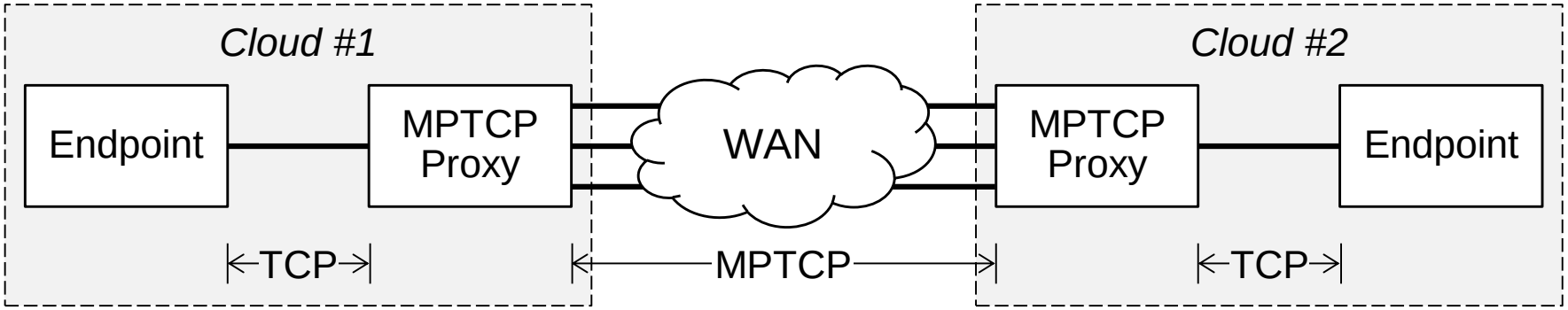
- add an MPTCP header option
- transform TCP header fields (IP addresses, TCP ports, sequence numbers)

Advantages

- direct mapping of packets
- no need to touch data (e.g., split packets and reassemble data)

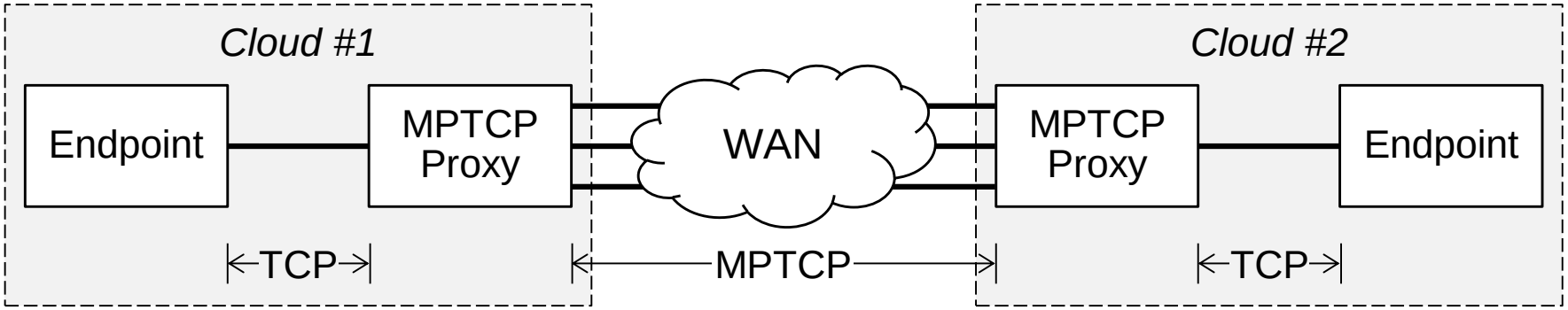
Disadvantages

- Packets get larger (added header option!)
  - Need to configure/tell endpoints to reduce packet size
- TCP behaviour for subflows needs to be implemented



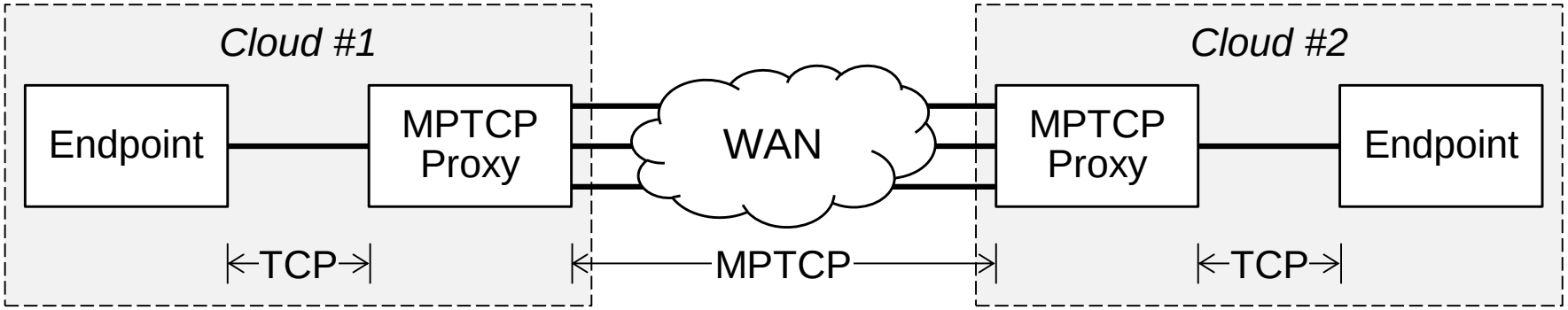
MPTCP Proxy: Decoupled Connections

- TCP and MPTCP connections terminated at both sides of the proxy
- No direct mapping/coupling of packets
  - works on TCP's stream abstraction
- Advantages
  - no need to explicitly fiddle around with individual packets
  - no need to re-implement TCP
- Disadvantages
  - buffering of data required at proxy
  - has impact on congestion control, latency ("buffer bloat")



MPTCP Proxy: Implementation

- Another advantage of decoupled connections: proxies that do this already exist!
- SOCKS is doing exactly this (w/o MPTCP)
- Stable proxies exist
- Implementation: run a SOCKS proxy on top of an MPTCP-capable kernel!
- Traffic needs to be steered to proxies, as applications do not know about them

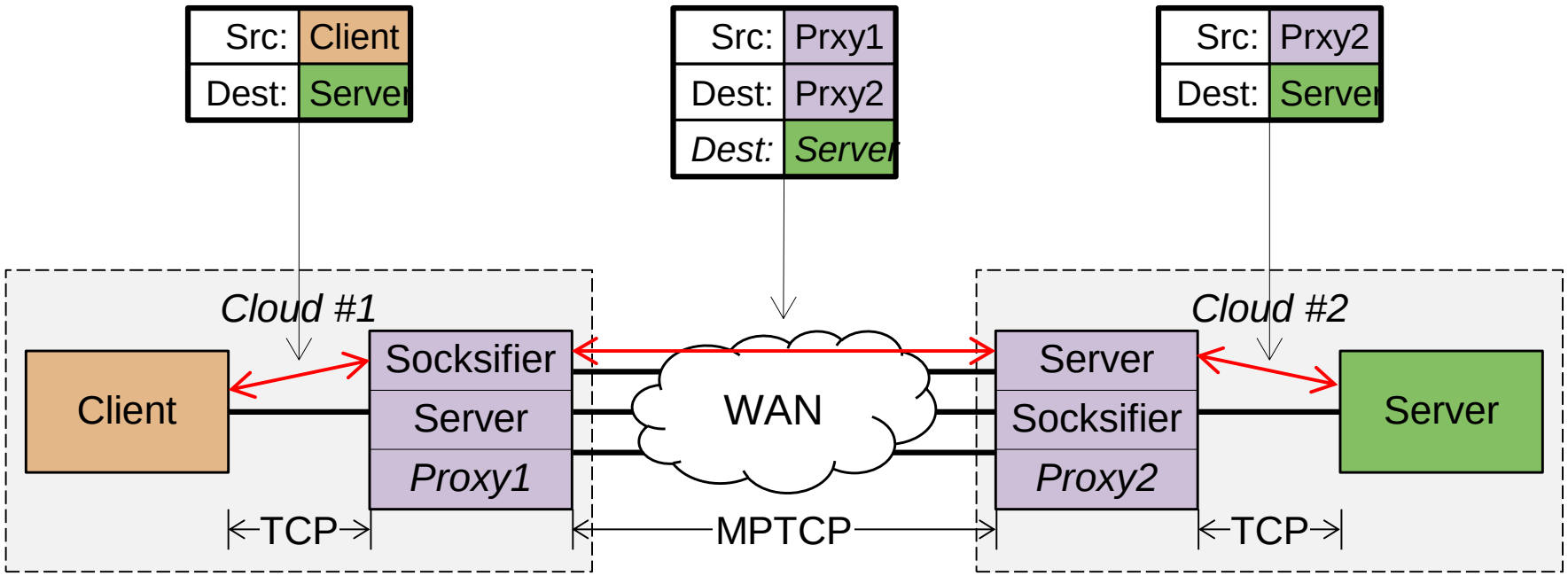


MPTCP Proxy: Peer-to-Peer

- SOCKS is designed for client-server operation
- For transparent operation, two components are needed: Socksifier, SOCKS server
- Still, only client-server
- Cloud federation is a peer-to-peer application
  - connections can be initiated from both sides
- Double instantiation of Socksifier – SOCKS server in opposite directions work
  - traffic needs to be isolated and steered appropriately, depending on direction of connection establishment

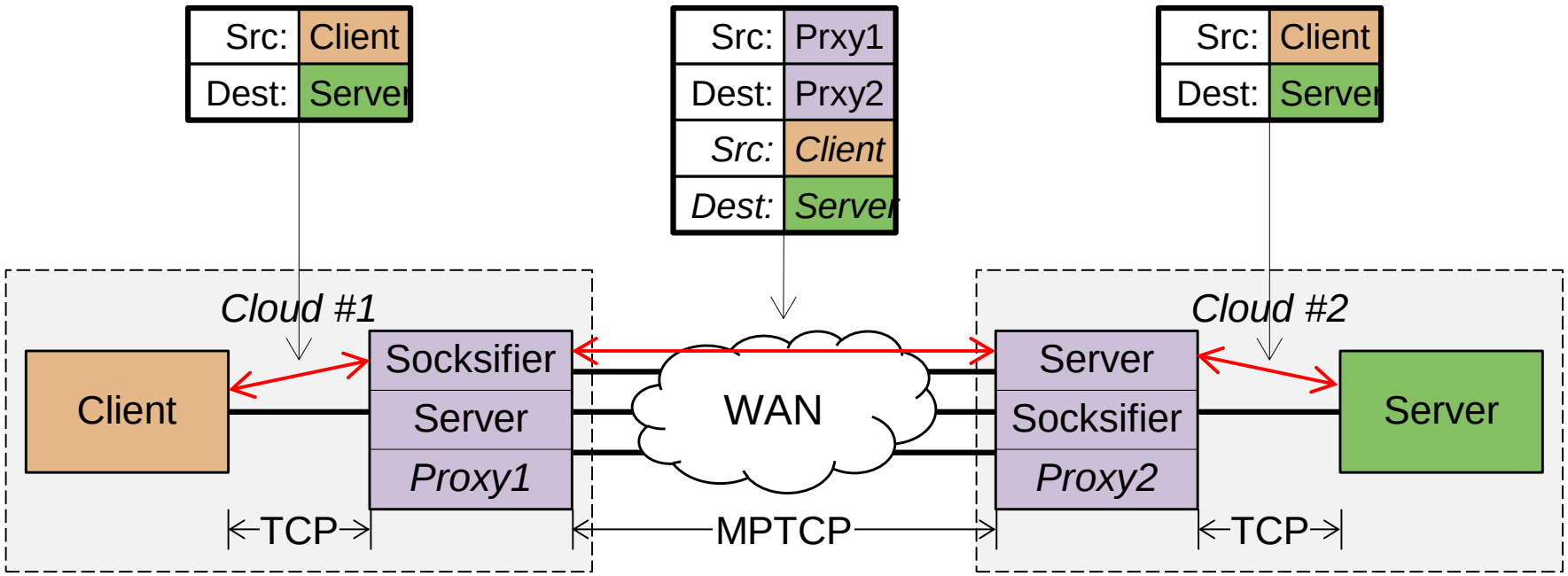
MPTCP Proxy: IP Addresses

- Proxy uses its own address for establishing connections to destination
- Might affect access restrictions / content selection on server

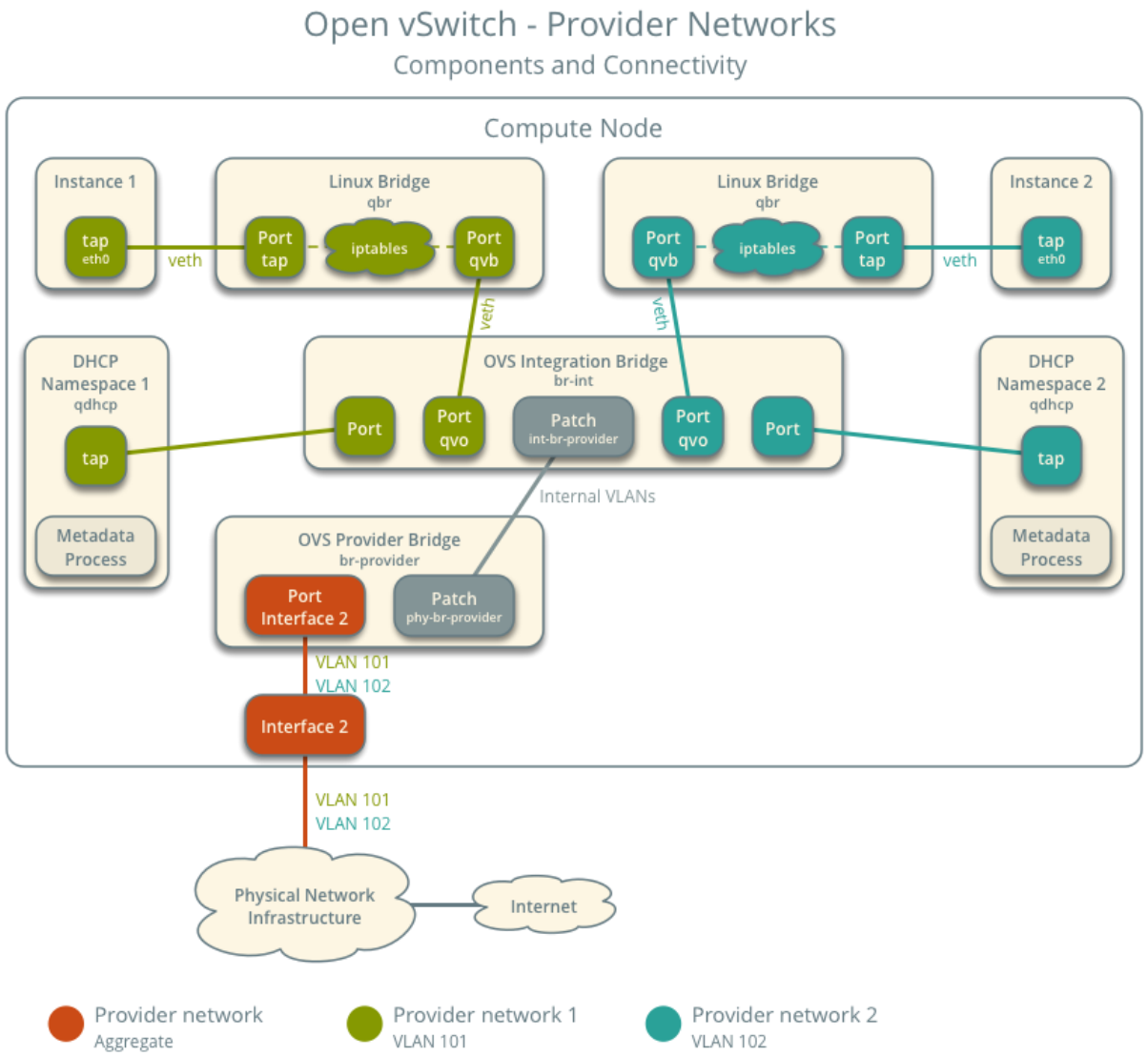


MPTCP Proxy: IP Address Fixing

- Extended SOCKS protocol to also communicate Client address
- Rewrite source address at destination proxy

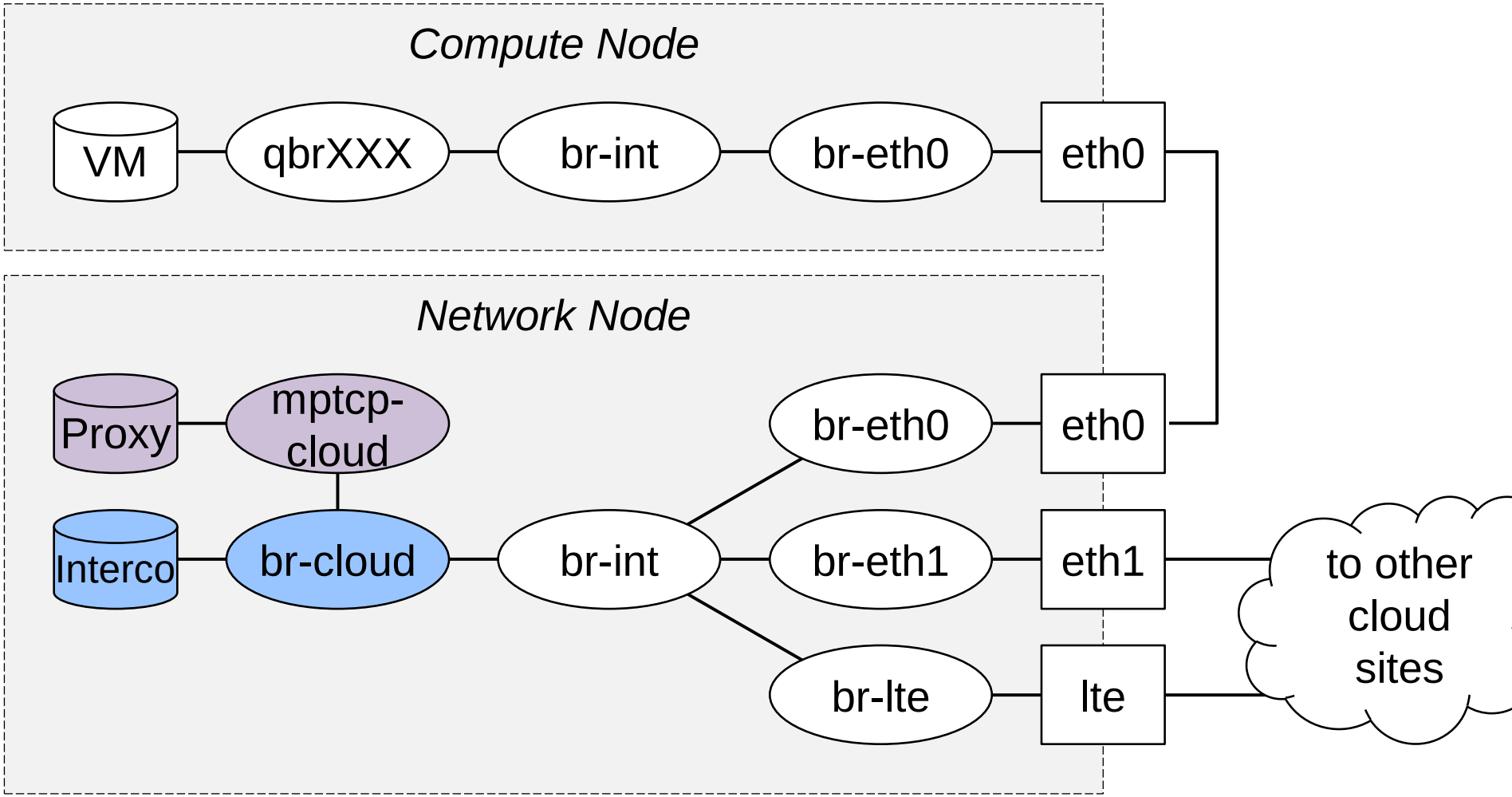


Digression: OpenStack Networking

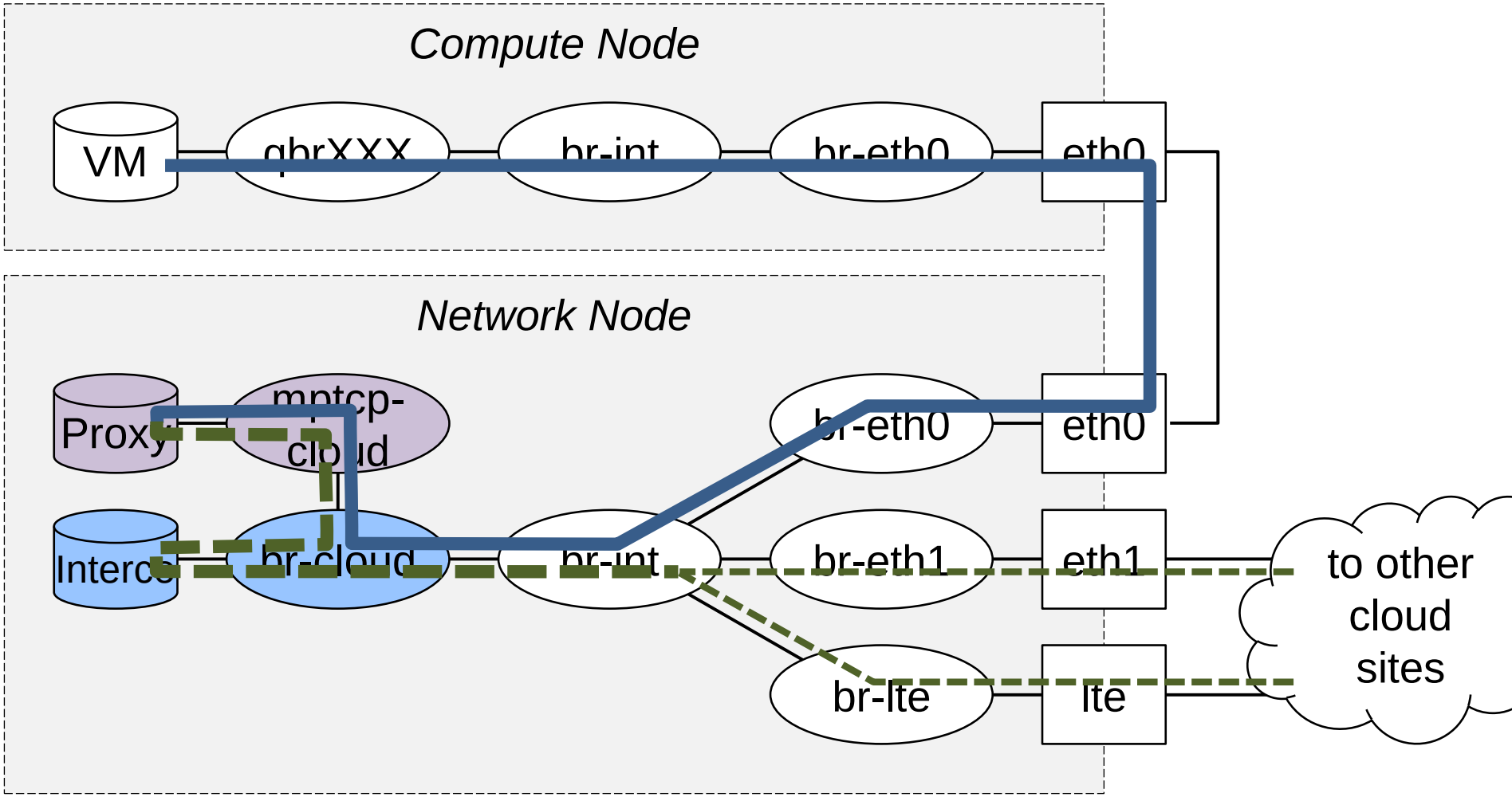


Source: https://docs.openstack.org/ocata/networking-guide/deploy-ovs-provider.html

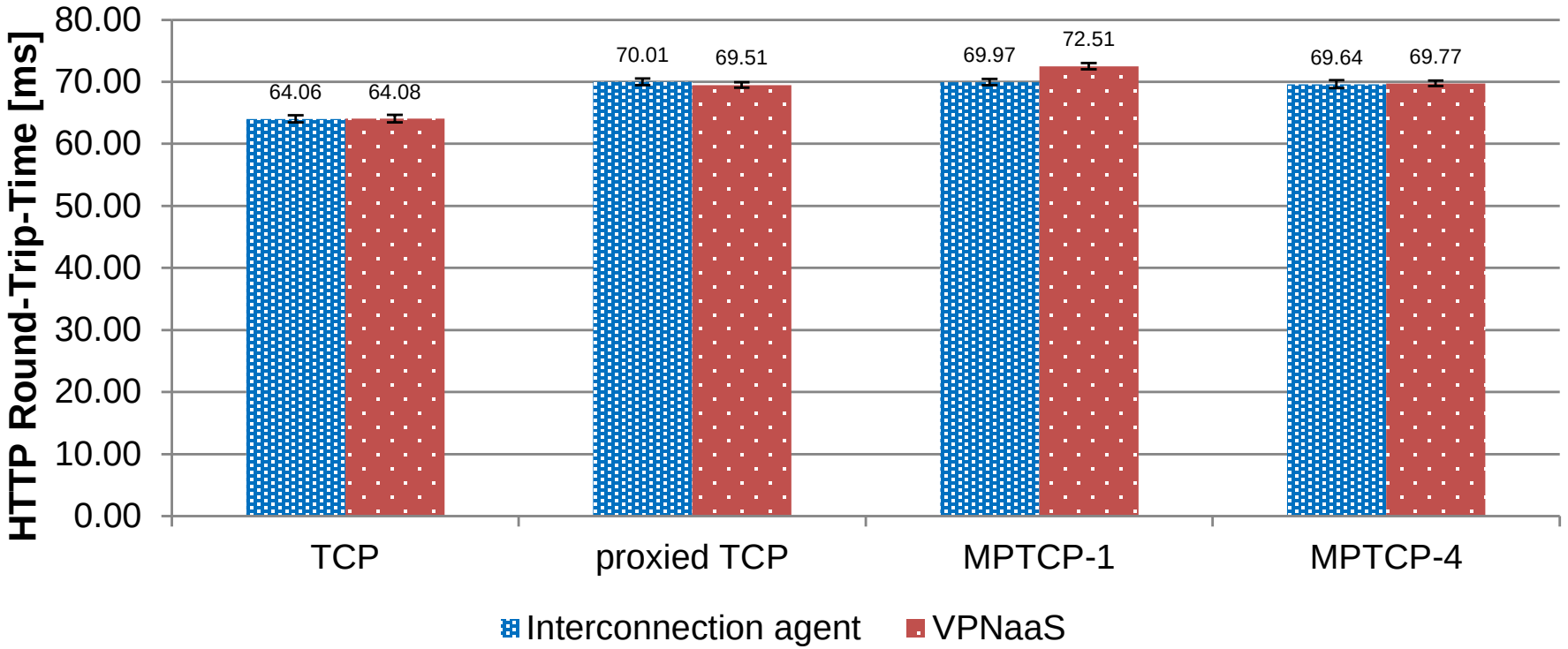
MPTCP Proxy & OpenStack: Components



MPTCP Proxy & OpenStack: Traffic Flow

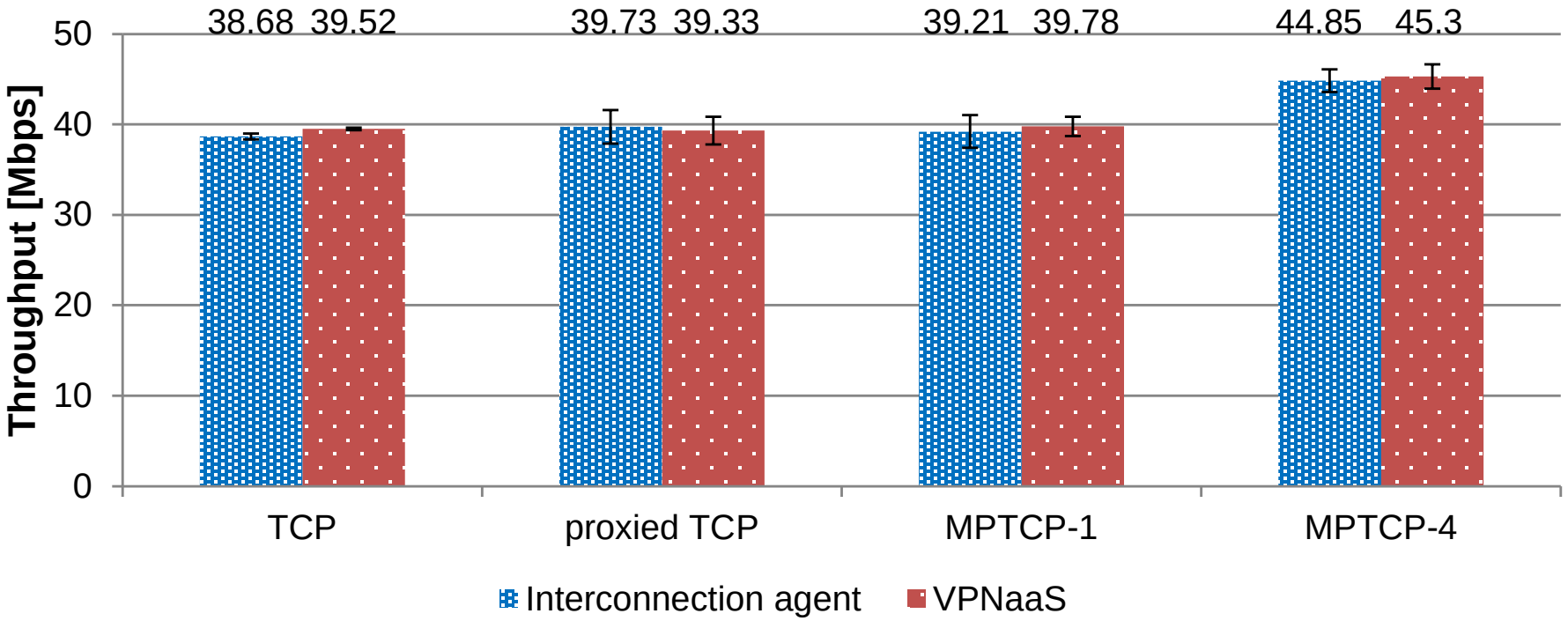


Cloud Federation Results: Latency



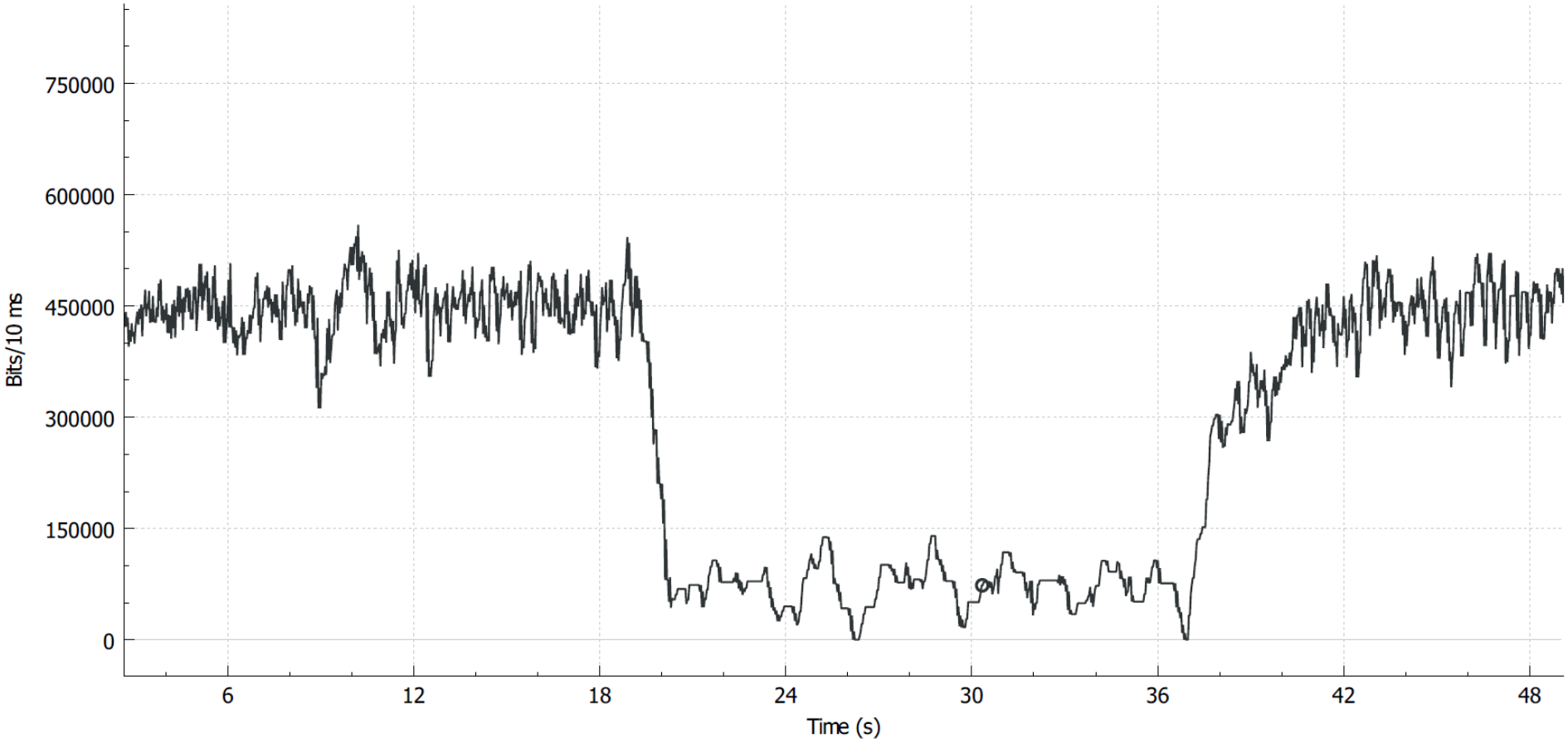
Connection between Heidelberg, Germany and Helsinki, Finland

Cloud Federation Results: Throughput



Connection between Heidelberg, Germany and Helsinki, Finland

Cloud Federation Results: Link Down/Up



Summary

- **Cloud federation** can **improve resilience** and **latency**
  - independent failure
  - multiple locations
- **Multipath** interconnection further **improves resilience** and **throughput** for inter-cloud traffic
  - independence of network paths
- **MPTCP** is a **viable technology** for interconnection
- MPTCP proxy makes **benefits** of multiple paths **available** to all cloud applications **transparently**
  - no changes to applications needed


Orchestrating a brighter world
NEC