KubeCon | CloudNativeCon

North America 2023

KubeCon | CloudNativeCon
North America 2023

Reliable RPCs over Hybrid Clouds, and the End-to-End Argument

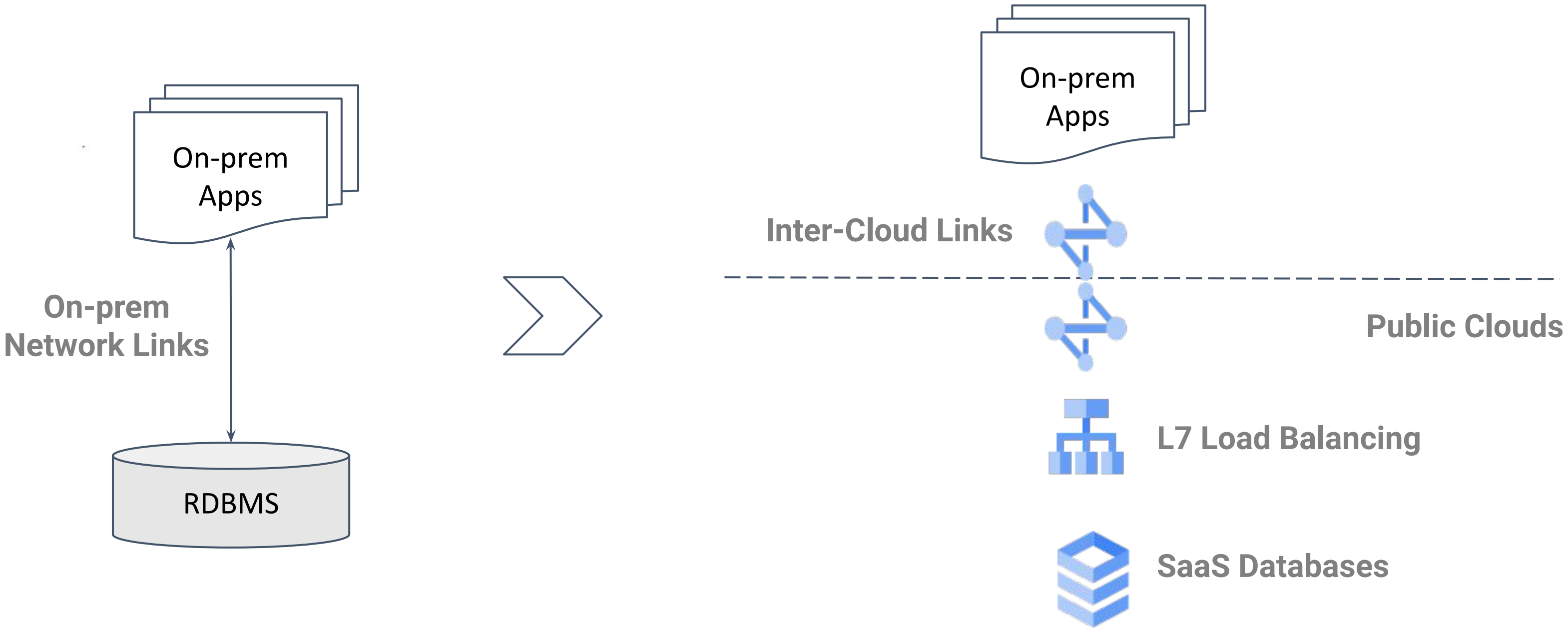
Wenbo Zhu, Vinod Lasrado

Google Inc.

Hybrid Clouds?

Background: On-prem storage/database services move to Clouds while applications stay on-prem.

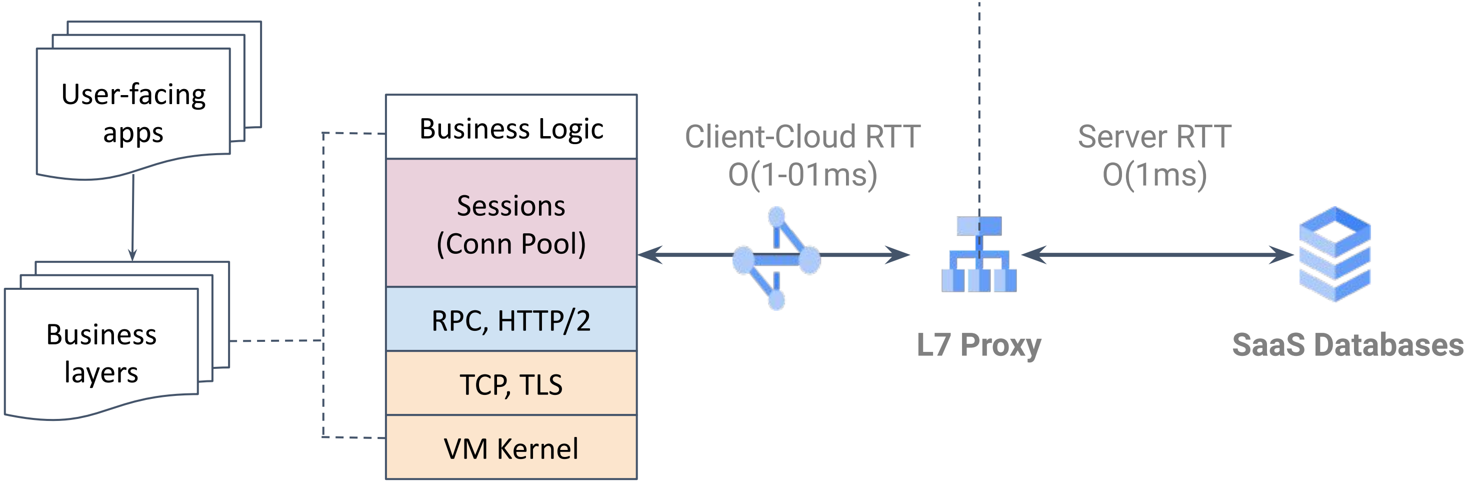
Challenges: applications are not yet adapted to the remote SaaS stack, esp. their failure modes and latencies.



Case Study

A generalized version

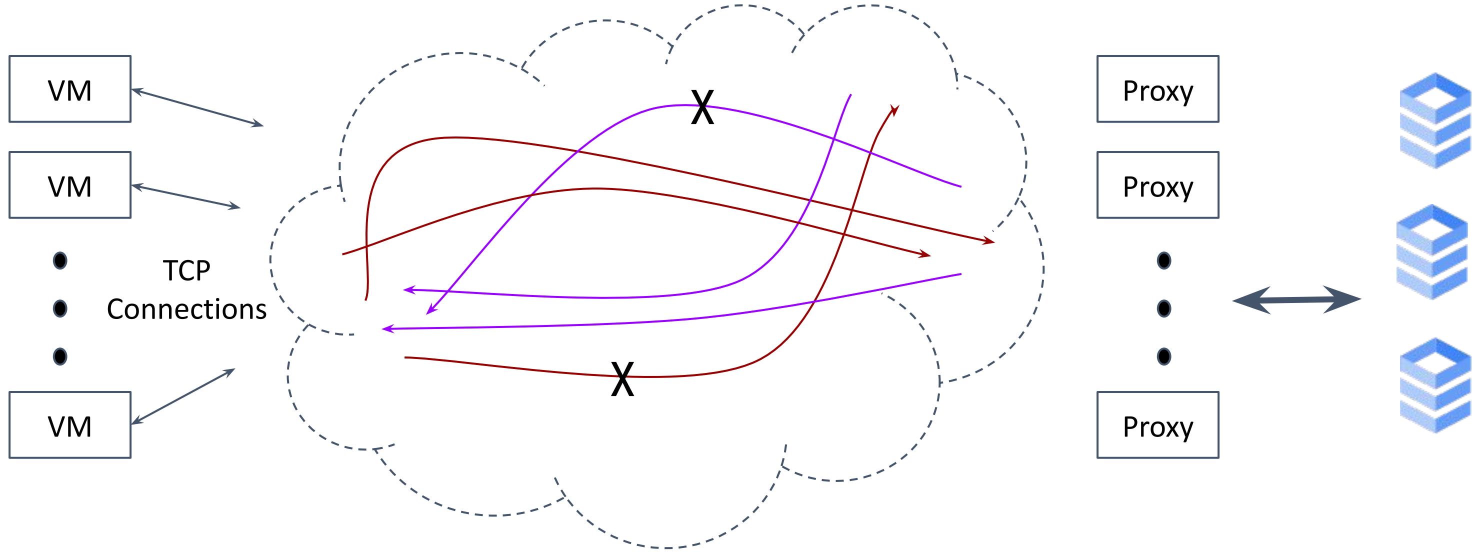
- RDMBS style transactional ops (writes) vs non-transactional ops (reads)
- Client-Cloud RTT >> Server RTT
- Timed-out reads may cause application errors that are visible to end-users
- Timed-out writes will abort the underlying transaction
- Connection-based affinity for server-side sessions and caching, with conn-pools to distribute sessions



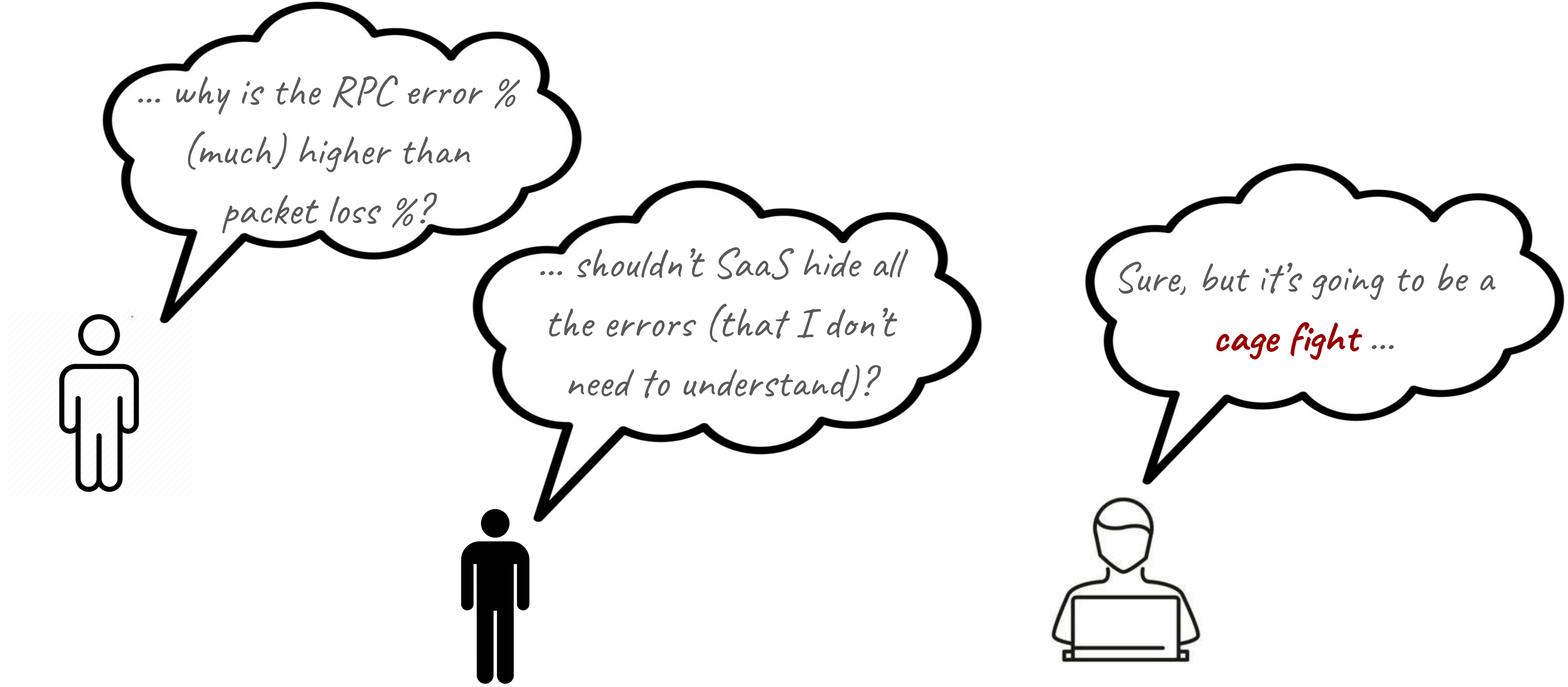
Failure Modes

L3 failures between the client and Cloud

- Failures affect only a small fraction of “flows” (hence TCP connections) and flows are unidirectional
- Failures often result in short-lived black holes but partial packet losses are also possible
- Transient server or proxy failures (overloading) may happen which are not correlated to L3 failures



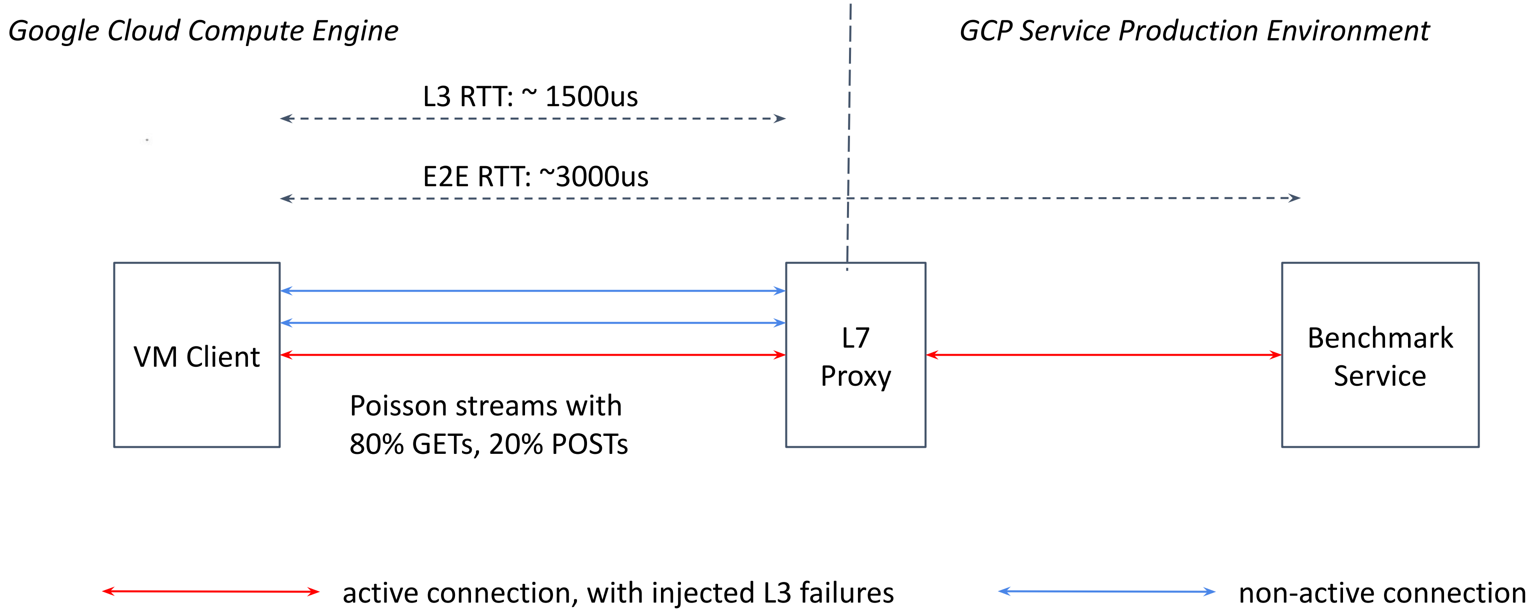
Key Questions



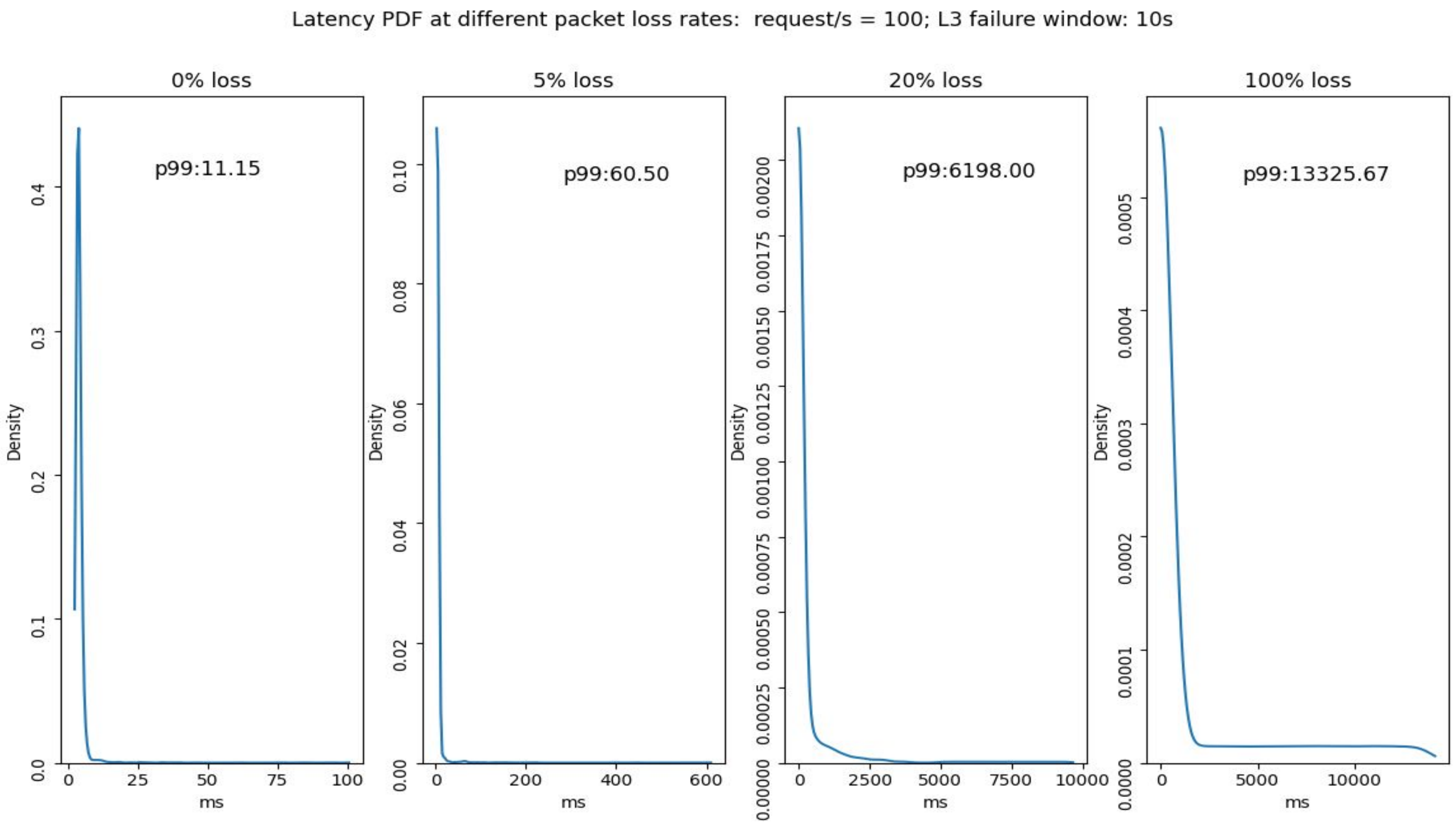
Evaluations

A general-purpose benchmark framework

- Evaluate impacts of different workload and environment parameters
- Evaluate effectiveness of different solutions for minimizing visible errors

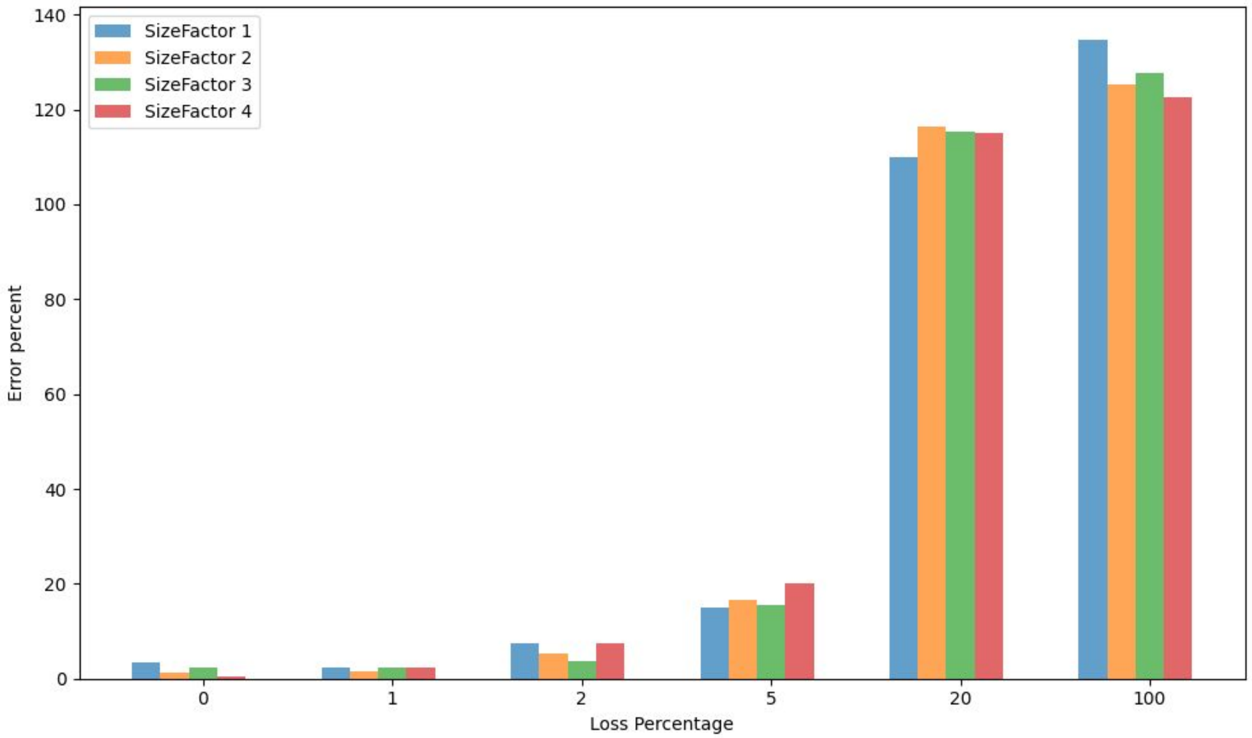


Tail Latencies



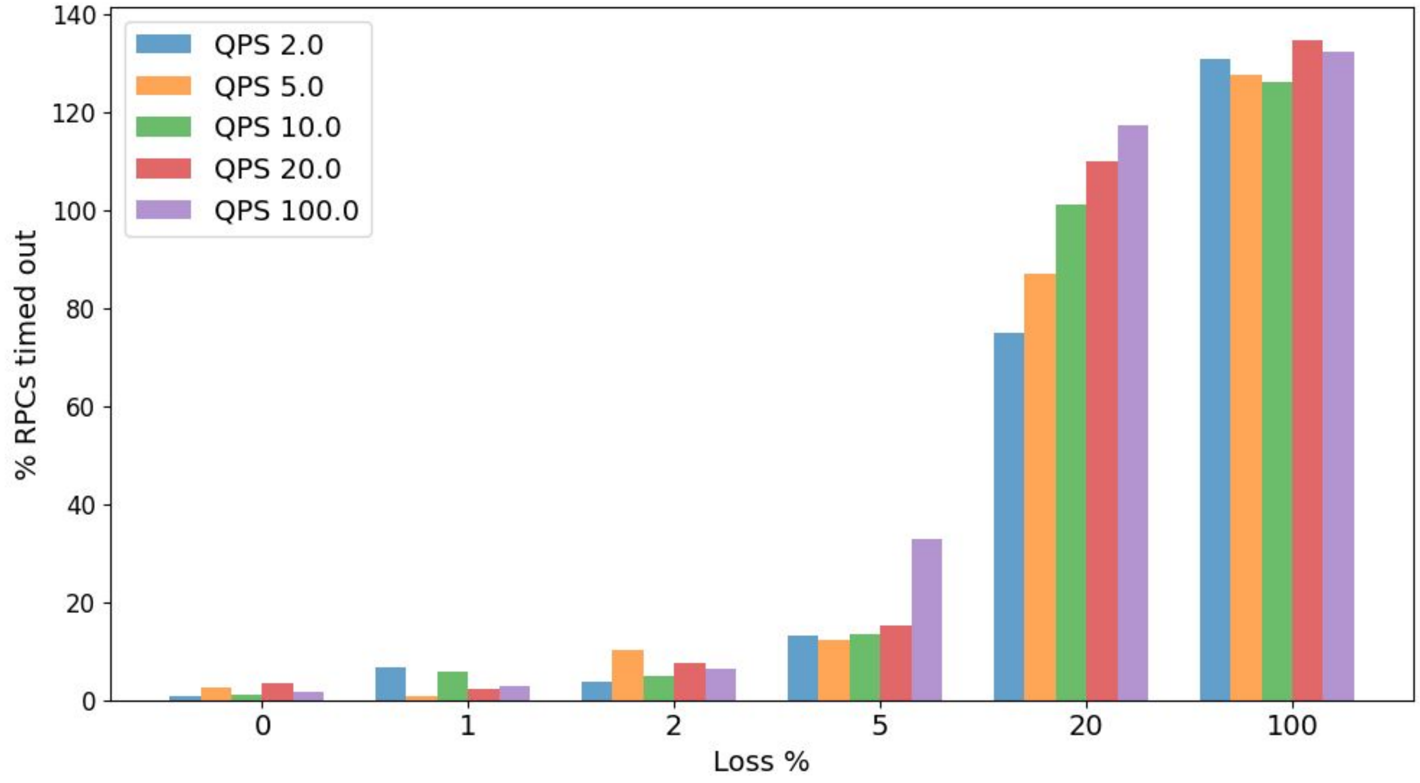
RPC Timeouts (Payload Sizes)

Timeout is set to 4 *RTT (median E2E). RPCs are not cancelled.

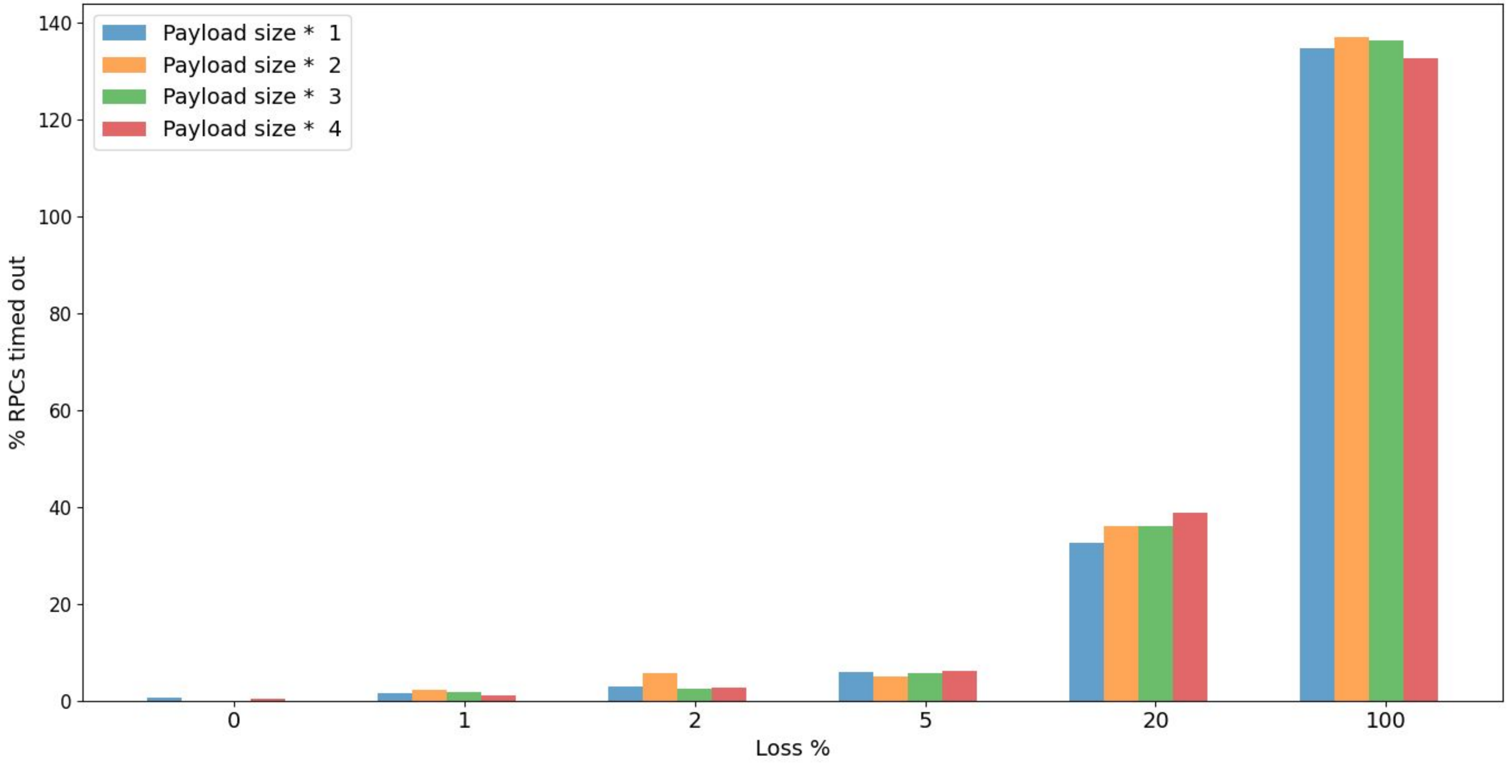


RPC Timeouts (Request Rates)

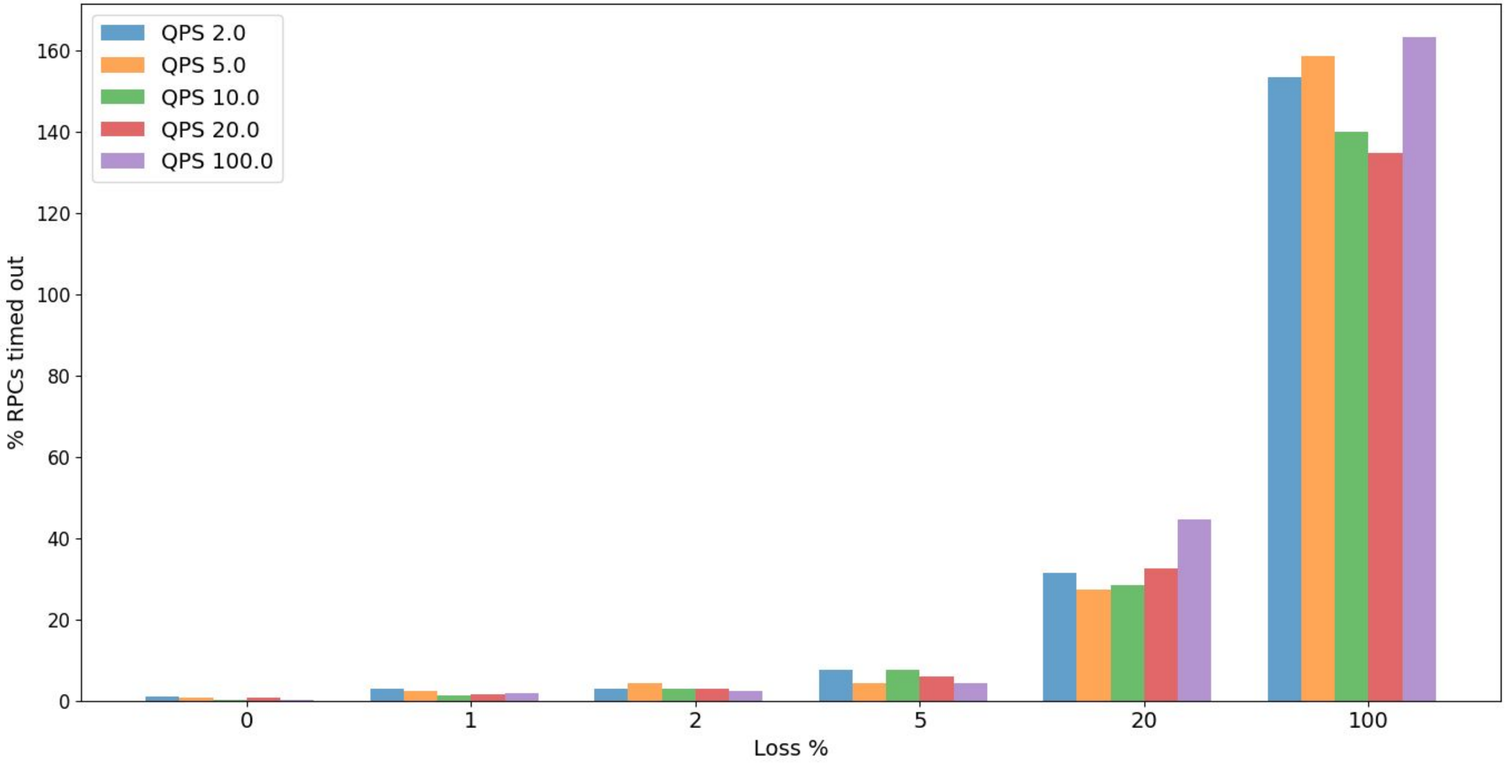
Timeout is set to 4 *RTT (median E2E). RPCs are not cancelled.



H3/QUIC (Payload sizes)



H3/QUIC (Request Rates)

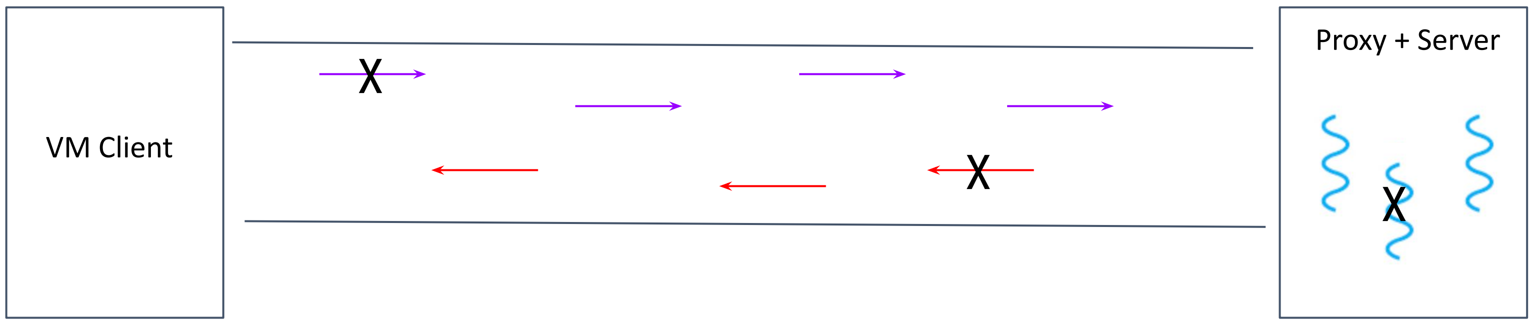


Failure Detection

Fail-fast: detect (suspect) a failed connection (or server) timely

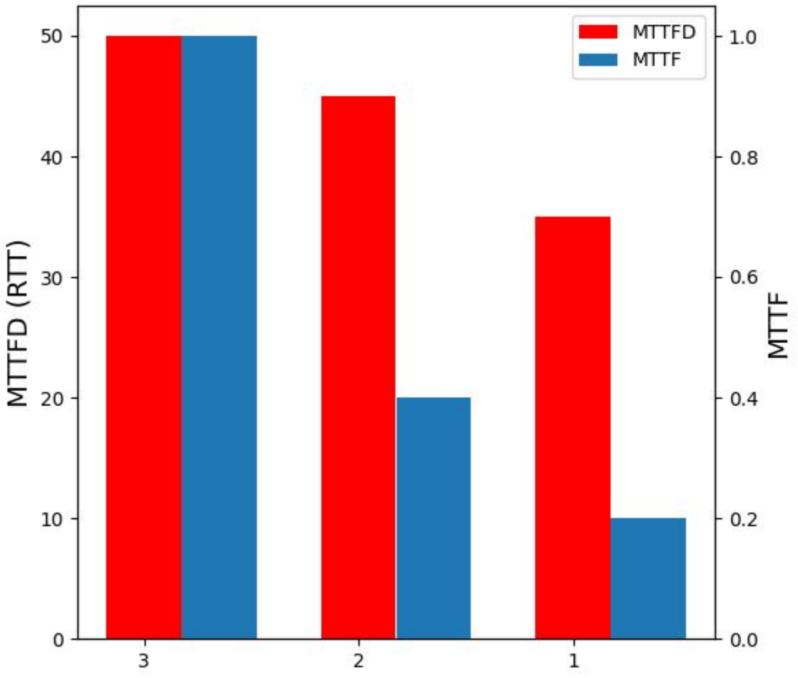
Detection params

- A: a factor of median RTT
- B: maximum number of pending acks
- ack_num: number of pending L7 acks in the runtime
- timeout = 12 * A * RTT - 2 * A * RTT * min(B, ack_num)



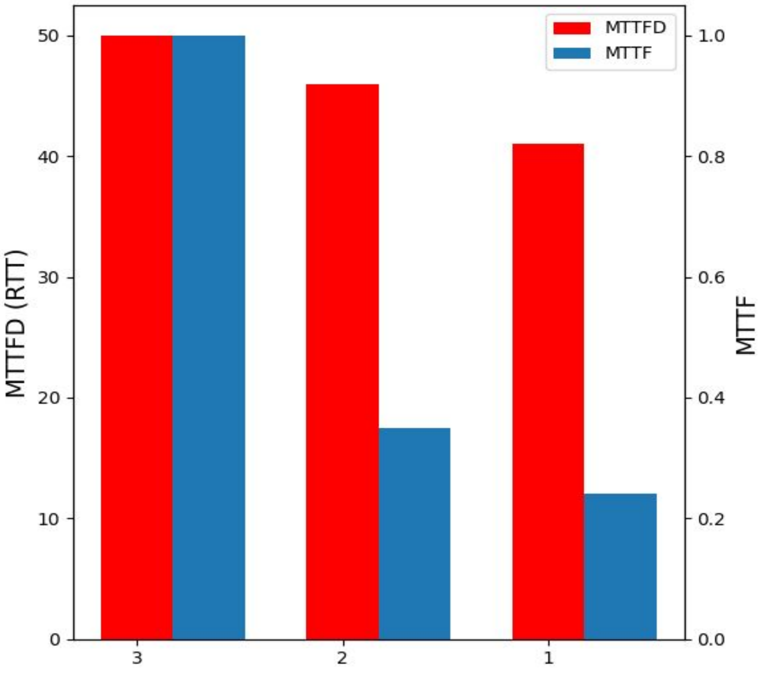
Evaluation Results

Timeliness: MTTFD with injected failures



A (B = 3)

False-positive: MTTF with failure-free runs



B (A = 3)

Failure Recovery

Decoupled from failure detection timeout

- Shorter timeout for reads and longer timeout for writes, as decided by applications

Upon a failed connection

- New requests will use a different connection
- Pending reads will be auto-retried immediately on a different connection but only once
- Pending requests will run to completion or will be cancelled after a timeout

Minimum visible errors

- Read timeout is larger than 2 * MTTFD, e.g. 300us
- Write timeout is longer than the failure window, e.g. 5s

Cost of false-positive detection

- Duplicated reads and misrouted writes

End-to-End Arguments

<u>E2E Principle</u>: functions such as failure detection and mitigation may "completely and correctly be implemented only" at an application level, and implementing those functions completely and correctly from a lower level is not possible.

<u>Key takeaways</u>

- Leverage service-level RPC semantics in failure detection and recovery
- Passive detections to adapt to runtime conditions
- Graceful recovery to handle short failure windows

<u>Low-level recovery</u>

- Desired if it can be done quickly and transparently (without tearing down TCP connections)
- An optimization aligned with the E2E principle

Production Environments?

No silver bullet

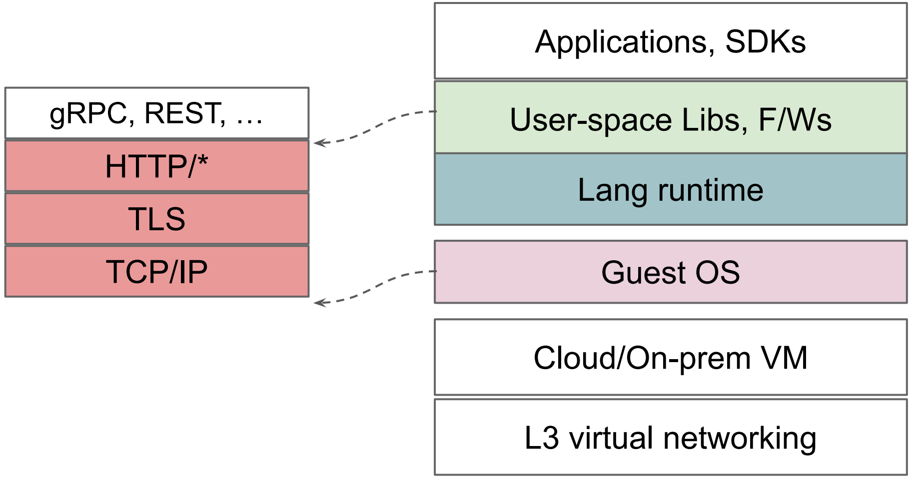
Monitoring data is critical to identify root causes and to evaluate solutions

- Client-side transport layer
- Correlation between userspace/guest transports and L3 networking

eBPF Transport Monitoring

github.com/google/ebpf-transport-monitoring

- A standalone toolkit
- Correlation between kernel events (TCP) and user-space events (TLS, HTTP/2)
- Future work: more languages, compact event encoding, QUIC/H3 …



Contact Us

Feedback and contributions

- eBPF transport monitoring
- Benchmark framework
- Example data sets, analytical methods

Discussion: *github.com/google/ebpf-transport-monitoring/issues*

Email: vlm@google.com, web@google.com

Please scan the QR Code above to leave feedback on this session