1

« CrossRSS »

Stateless CPU-aware Datacenter Load-Balancing

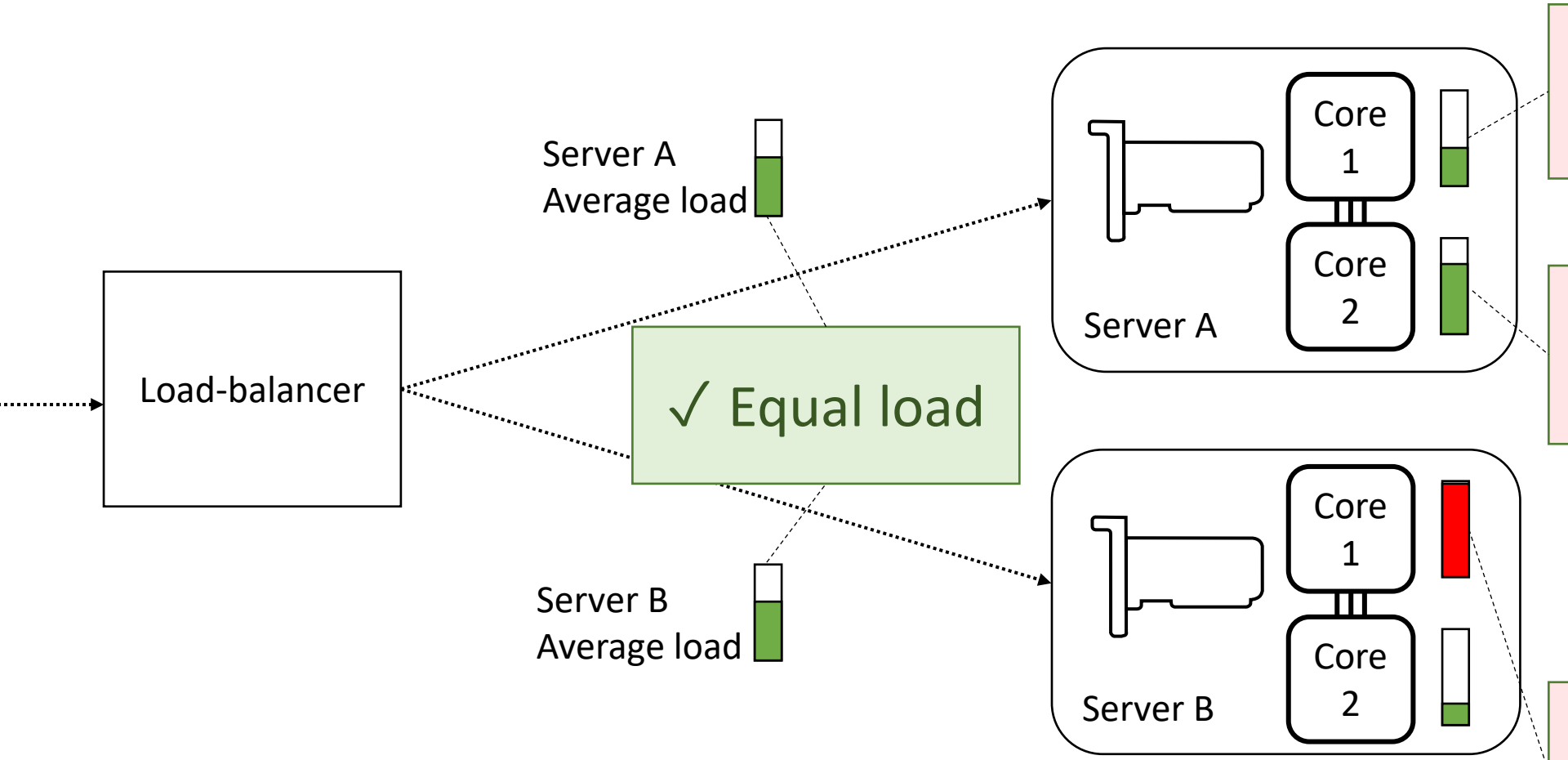
Tom Barbette, Marco Chiesa, Gerald Q. Maguire Jr. and Dejan Kostić

KTH Royal Institute of Technology

2

The problem

Load-balancing **between** servers is not enough



✗ Underutilisation

$ $ $

✗ High tail latency

✗ Dropping packets

→ poor balancing **within** servers

3

The challenge

Current solutions require either:

- Many context switches, inter-core communications

  ⇒ OS scheduler, Shenango[NSDI'19], Shinjuku[NSDI'19]

- Server and/or application modifications

  ⇒ Metron[NSDI'18], RSS++[CoNEXT'19]

- SmartNICs

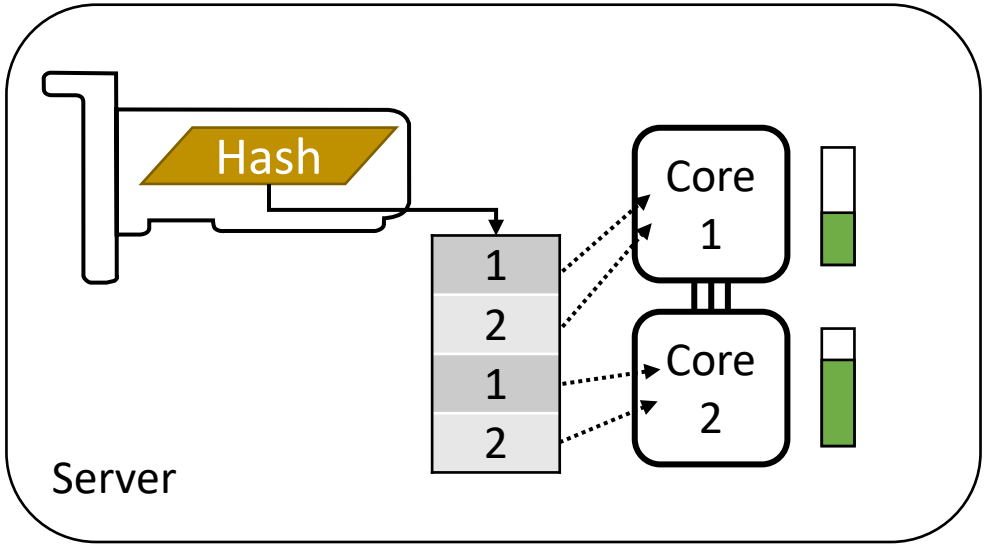
  ⇒ eRSS[APNET'19]

"Can we devise a load balancing mechanism that takes into account the per CPU **core** load – to achieve more uniform load balancing without requiring modifications to the servers?"

4

CrossRSS: Leverage knowledge of the dispatching done inside servers (RSS)

- The server's core handling the packet is given by the last bits of the hash of the 4-tuple
- Therefore when we select a server, we implicitly select a core too

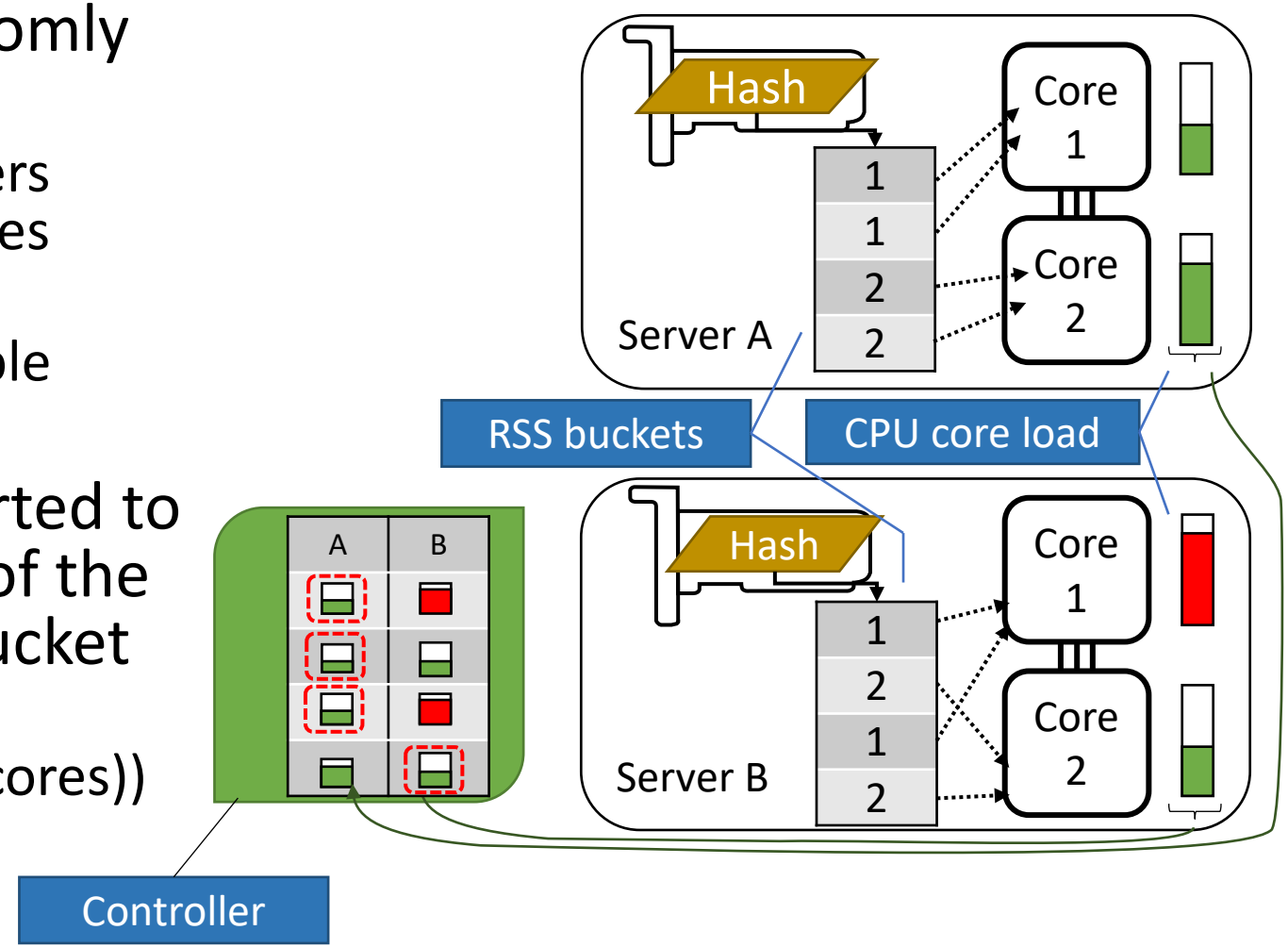


→ When selecting a server to handle a new flow, instead of putting servers in competition to select a least loaded one, put the « final » cores in competition

5

CrossRSS design 1/2

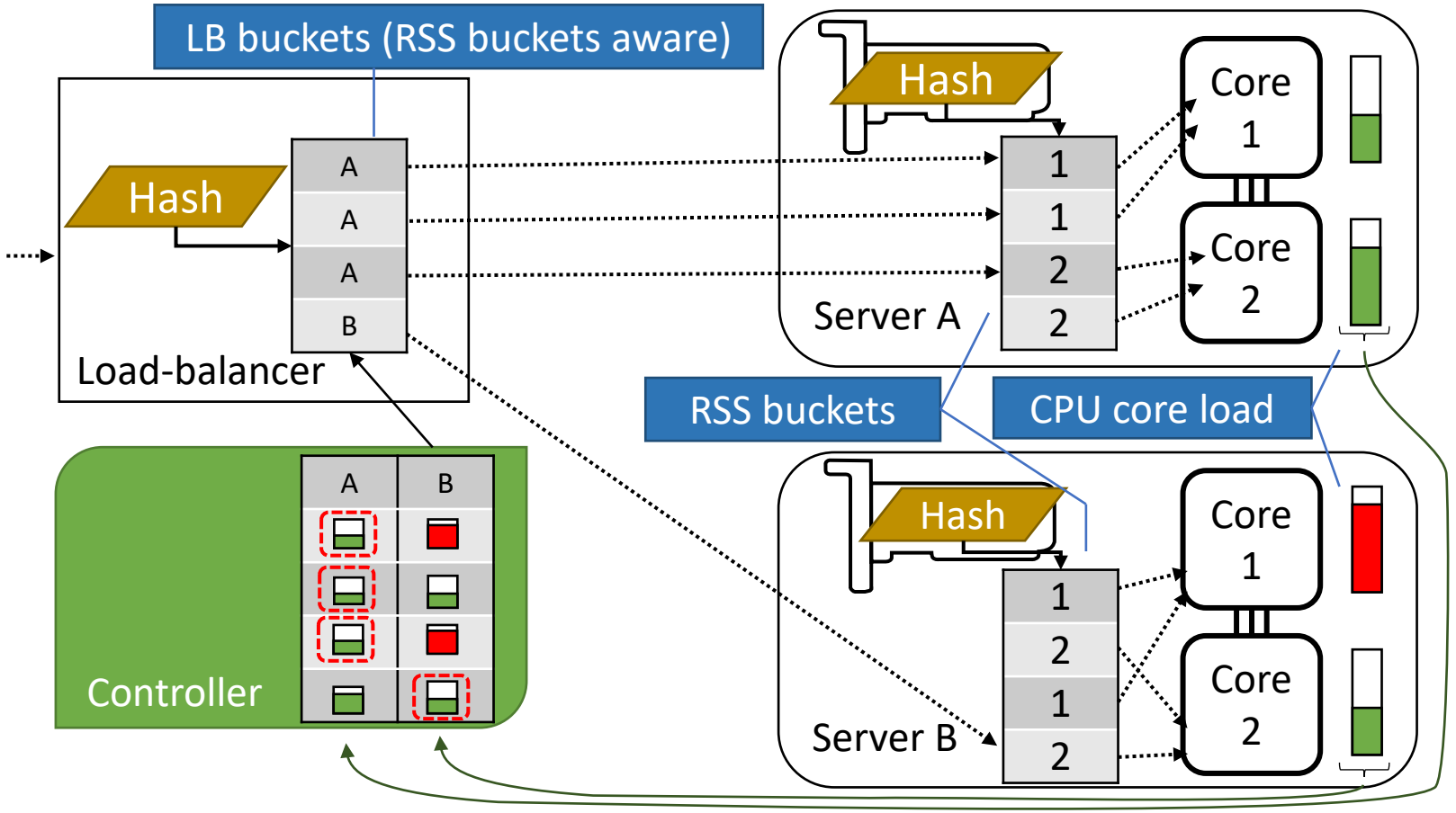
- RSS buckets are pseudo-randomly assigned to cores in server
  - Each bucket index across servers considers a different set of cores
    → power-of-K choices
  - The server can re-build the table easily
- The load of each core is reported to a controller that keeps track of the least loaded core for every bucket
  - Scalable as min(cores) == min(min(sub-cores), min(sub-cores))



6

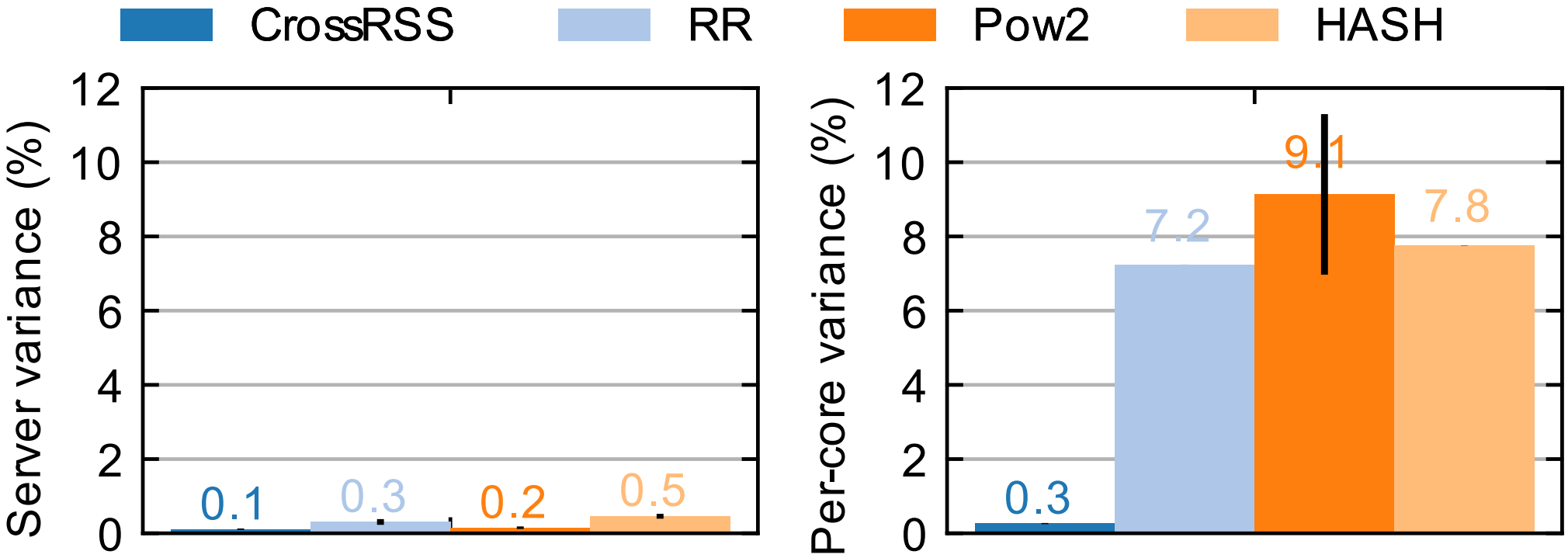
CrossRSS design 2/2

- The controller writes the mapping of *buckets* → *servers (→ core* is implicit*)* to the LB
- When a new flow (connection) is received, the LB compute the hash, and selects the server in the mapping
- To avoid breaking connections, one can use techniques proposed in Cheetah[NSDI'20]



7

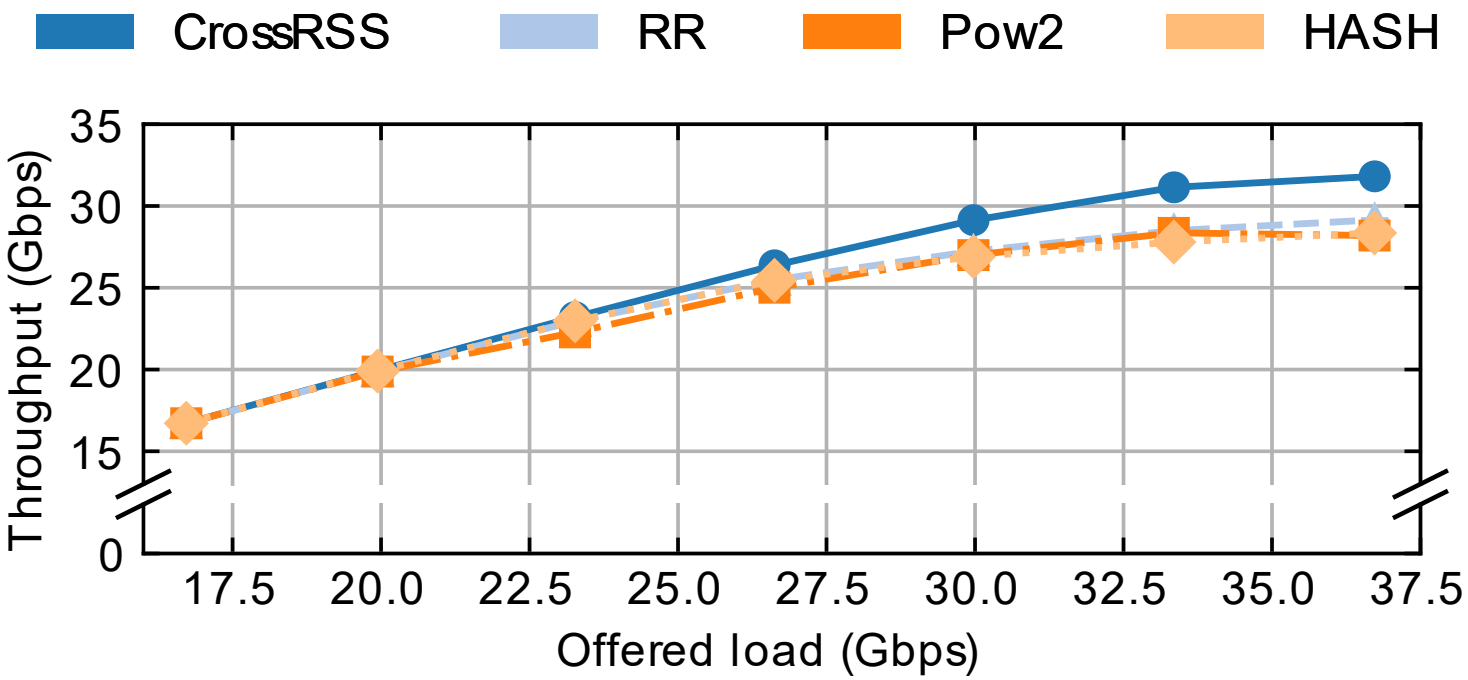
Evaluation: Variance



✓ All methods are more or less fine in term of **server** variance
But…

CrossRSS is 20X more effective in keeping **core** variance low !

Evaluation: Throughput



By achieving a better load balancing of packets towards a FW+NAT+DPI, CrossRSS allows to handle 12% more throughput