Homa

Data Center Transport Protocol

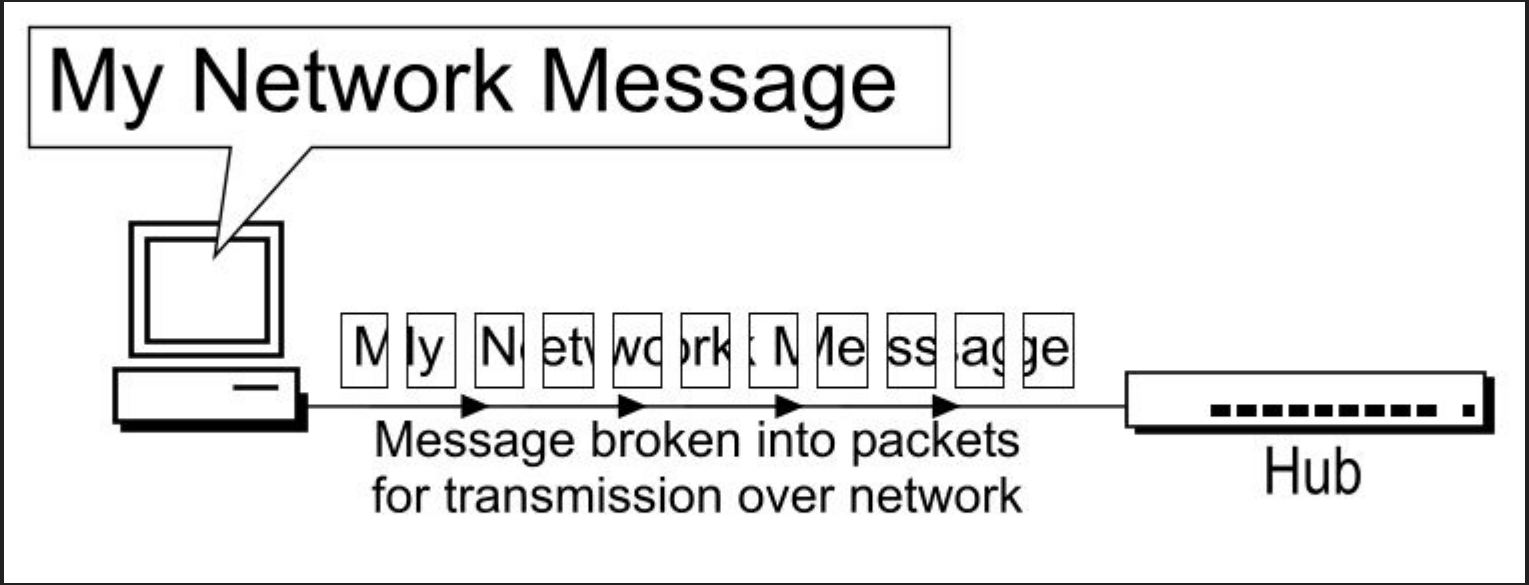
Presentation 2

Harsh Kapadia

Recap: Problems with TCP in the Data Center

- Stream orientation
- Connection orientation
- Bandwidth sharing (Fair Scheduling)
- Sender-driven Congestion Control
- In-order packet delivery

Recap: Message vs Packet



Recap: Homa Features

- Message-oriented (RPCs)
- Connectionless
- Shortest Remaining Processing Time (SRPT) Scheduling
- Receiver-driven Congestion Control
- High out-of-order packet tolerance
- No per-packet acknowledgements
- At-least-once semantics

Recap: Sender vs Receiver

- Client → Server
  - Sender: Client
  - Receiver: Server
- Server → Receiver
  - Sender: Server
  - Receiver: Client

Homa Packet Types

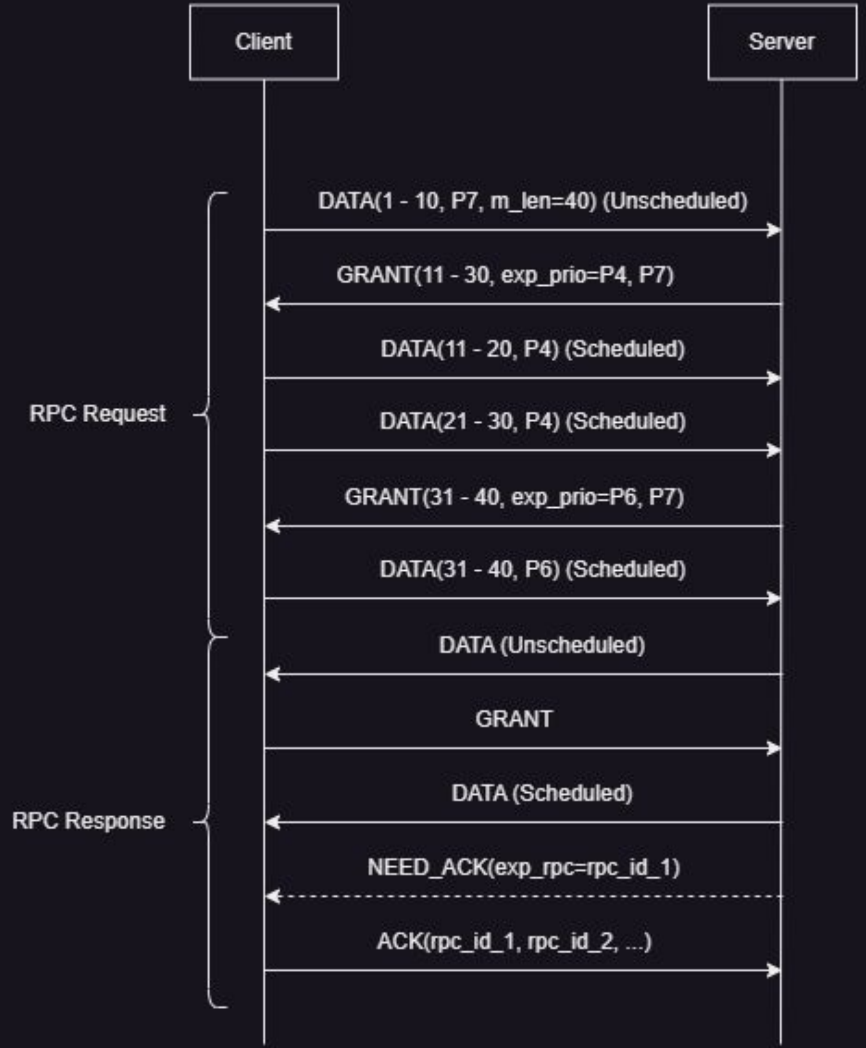
- `DATA(rpc_id, data, offset, self_prio, m_len)`
  - Sent by sender or receiver
  - Can `ACK` one RPC
- `GRANT(rpc_id, offset, exp_prio)`
  - Sent by receiver
- `RESEND(rpc_id, offset, len, exp_prio)`
  - Sent by receiver
- `UNKNOWN(rpc_id)`
  - Sent by sender or receiver
- `BUSY(rpc_id)`
  - Sent by sender
- `CUTOFFS(rpc_id, exp_unsched_prio)`
  - Sent by receiver
- `ACK(rpc_id)`
  - Sent by sender
  - Can `ACK` multiple RPCs
- `NEED_ACK(rpc_id)`
  - Sent by receiver
- `FREEZE`
- `BOGUS`

Homa API

- `homa_send()`
  - Send a request message to initiate a RPC.
- `homa_reply()`
  - Send a response message for a RPC previously received.
- `homa_abort()`
  - Terminate the execution of a RPC.

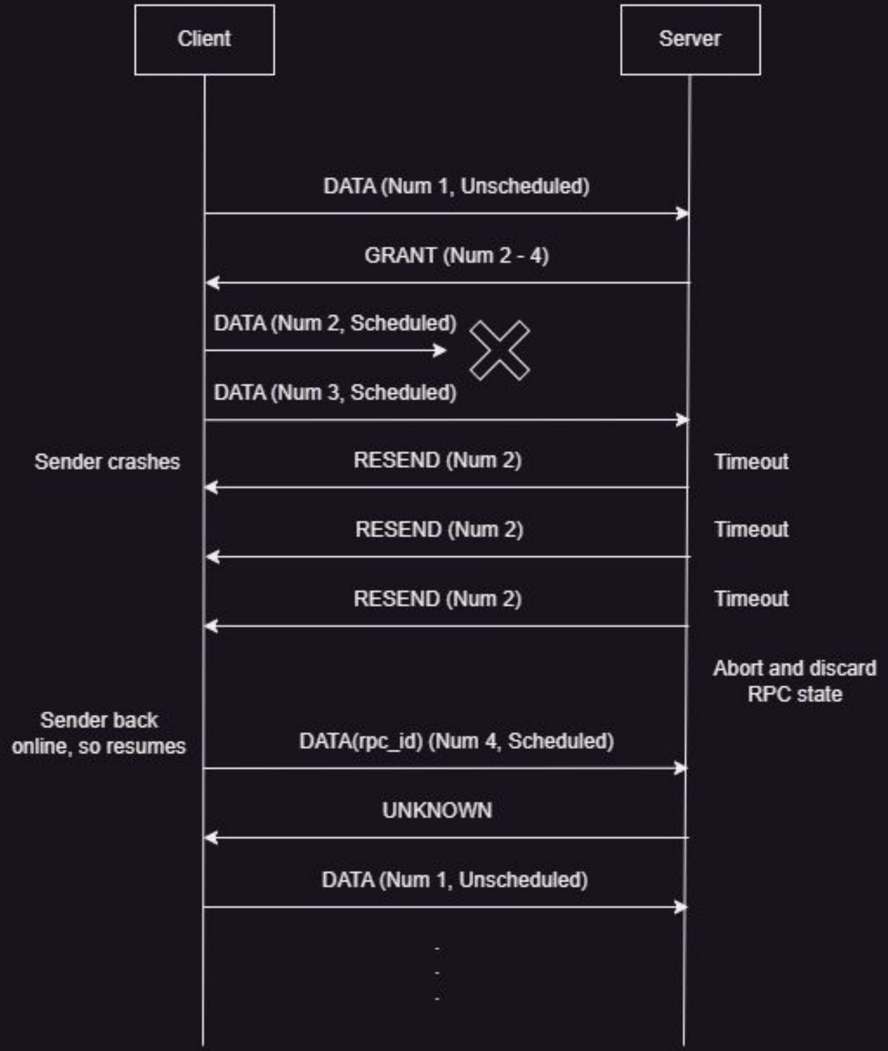
Homa Working

- RPC Request, RPC Response
- `DATA`, `GRANT`, `ACK`, `NEED_ACK`
- Priority levels: `P0` (lowest) to `P7` (highest)


Client
Server
RPC Request
RPC Response
DATA(1 - 10, P7, m_len=40) (Unscheduled)
GRANT(11 - 30, exp_prio=P4, P7)
DATA(11 - 20, P4) (Scheduled)
DATA(21 - 30, P4) (Scheduled)
GRANT(31 - 40, exp_prio=P6, P7)
DATA(31 - 40, P6) (Scheduled)
DATA (Unscheduled)
GRANT
DATA (Scheduled)
NEED_ACK(exp_rpc=rpc_id_1)
ACK(rpc_id_1, rpc_id_2, ...)

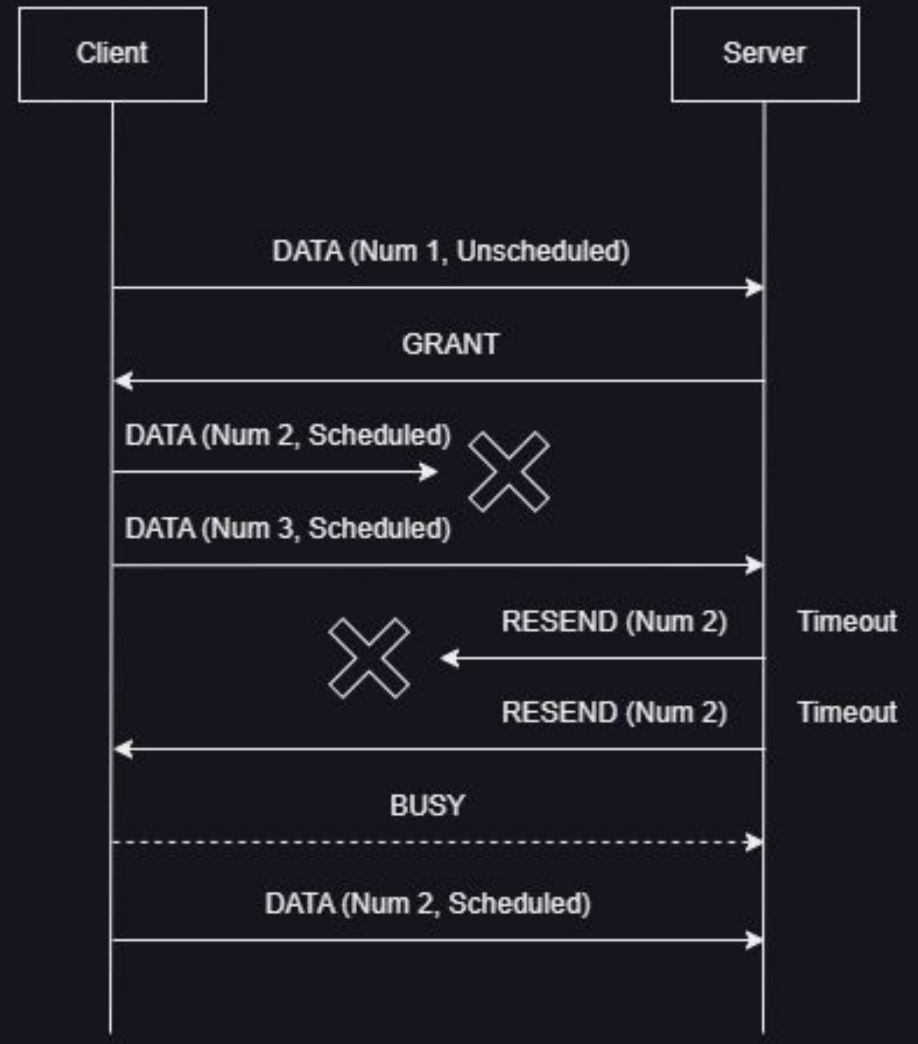
Homa Working

- RPC Request
- `DATA, GRANT, RESEND, UNKNOWN`



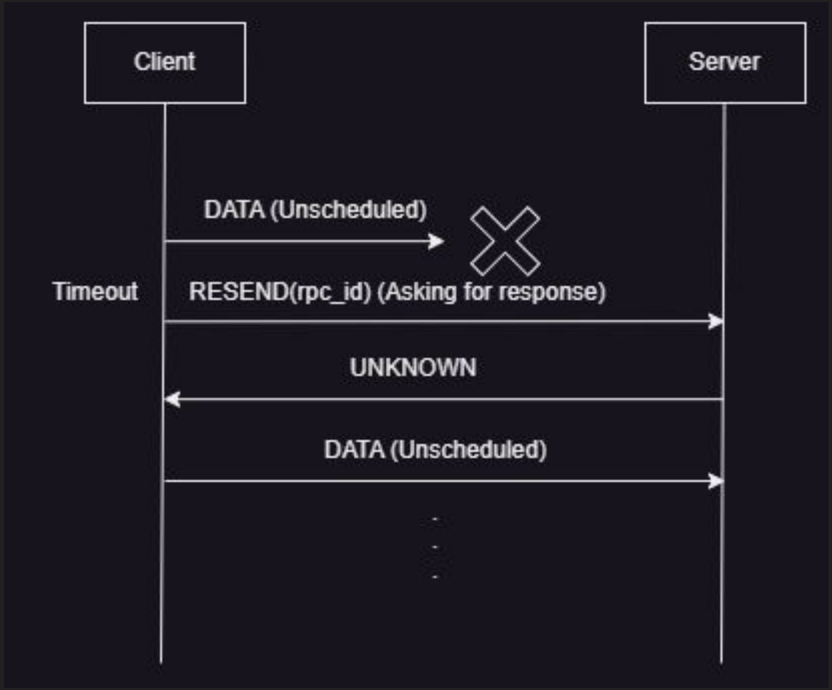
Homa Working

- RPC Request
- `DATA, GRANT, RESEND, BUSY`



Homa Working

- RPC Request
- `DATA`, `RESEND`, `UNKNOWN`
- Need to confirm this scenario



Recap: Homa Protocol Overview

- On receiving message from top layer, sender blindly sends unscheduled portion
- Sender can send further scheduled `DATA` packets only if receiver authorises through `GRANT` packet
- `GRANT` usually requests for '`RTT` bytes' worth outstanding data to keep transmission uninterrupted

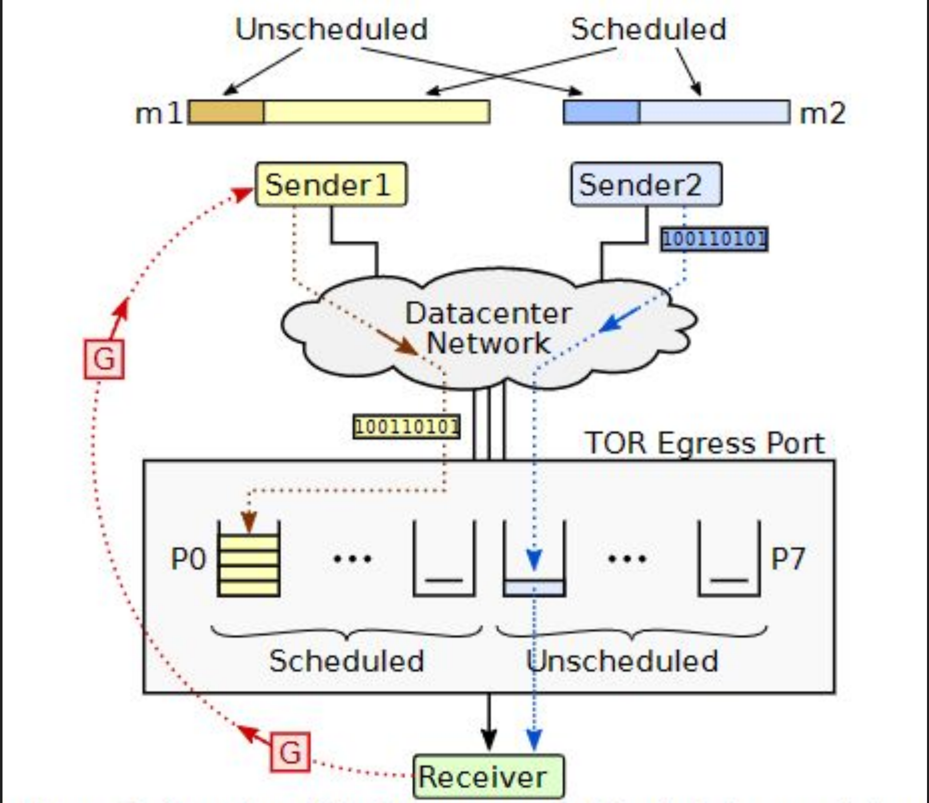


Figure 2: Overview of the Homa protocol. Sender1 is transmitting scheduled packets of message *m1*, while Sender2 is transmitting unscheduled packets of *m2*.

Homa Linux Architecture

- Transmit (top): `homa_send()` → copy packets → TSO/GSO → IP layer → NIC
- Receive (bottom): NIC (RSS) → Interrupt → NAPI (GRO, SoftIRQ core choosing) → SoftIRQ (network stack traversal) → copy packets → `homa_recv()`

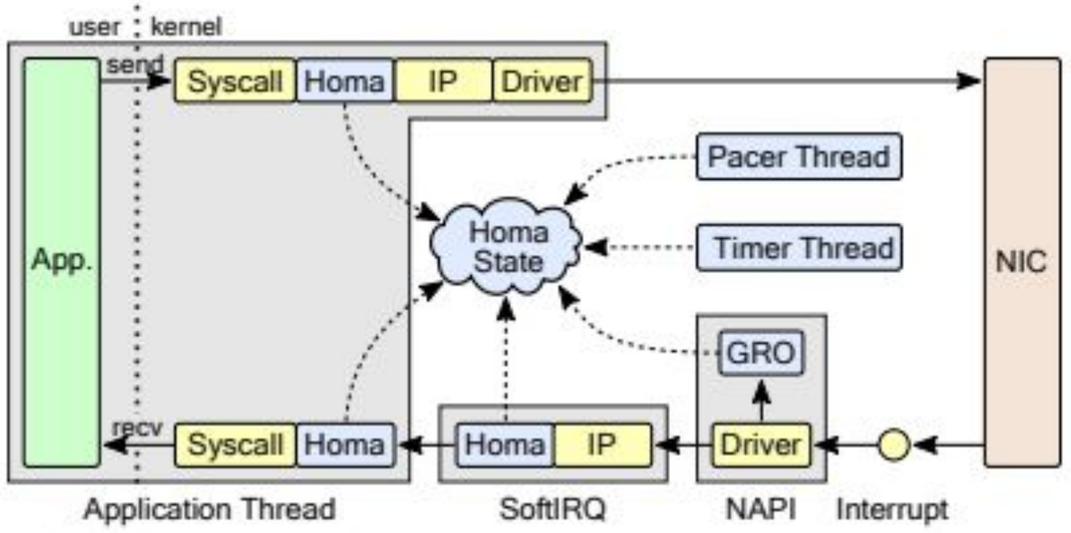


Figure 2: Structure of Homa/Linux. Homa components are shown in blue; existing Linux kernel modules are in yellow. Gray areas represent different cores. Only the primary sending and receiving paths are shown; other Homa elements such as the pacer thread and timer thread also transmit packets.

Data Center TCP (DCTCP)

- Makes use of Explicit Congestion Notification (ECN) to calculate how many bytes were affected and slows down based on that, rather than the fixed backoff that generic versions of TCP (Reno, CUBIC, etc.) have.

Experimentation Problems

- I was not able to get GENI TCP throughput above 100 Mbps.
  - Need ~25 Gbps
  - CloudLab
- I was not able to build the Homa module due to missing Linux kernel header files in Ubuntu 18.04 and 20.04 images on GENI.
  - Needs latest Ubuntu releases
    - Ubuntu 22.04 LTS image was not available on GENI
    - Broke multiple VMs trying to upgrade
  - Using CloudLab could solve problems

Tentative Future Plans

- Set up CloudLab environment
- Build Homa module
- Understand Homa module experiments and scripts
- Conduct Homa experiments
- Wireshark analysis of Homa
- Homa module code dive
- Try to form algorithm
- Try to make state machine

Resources

- networking.harshkapadia.me/homa

Thank you!