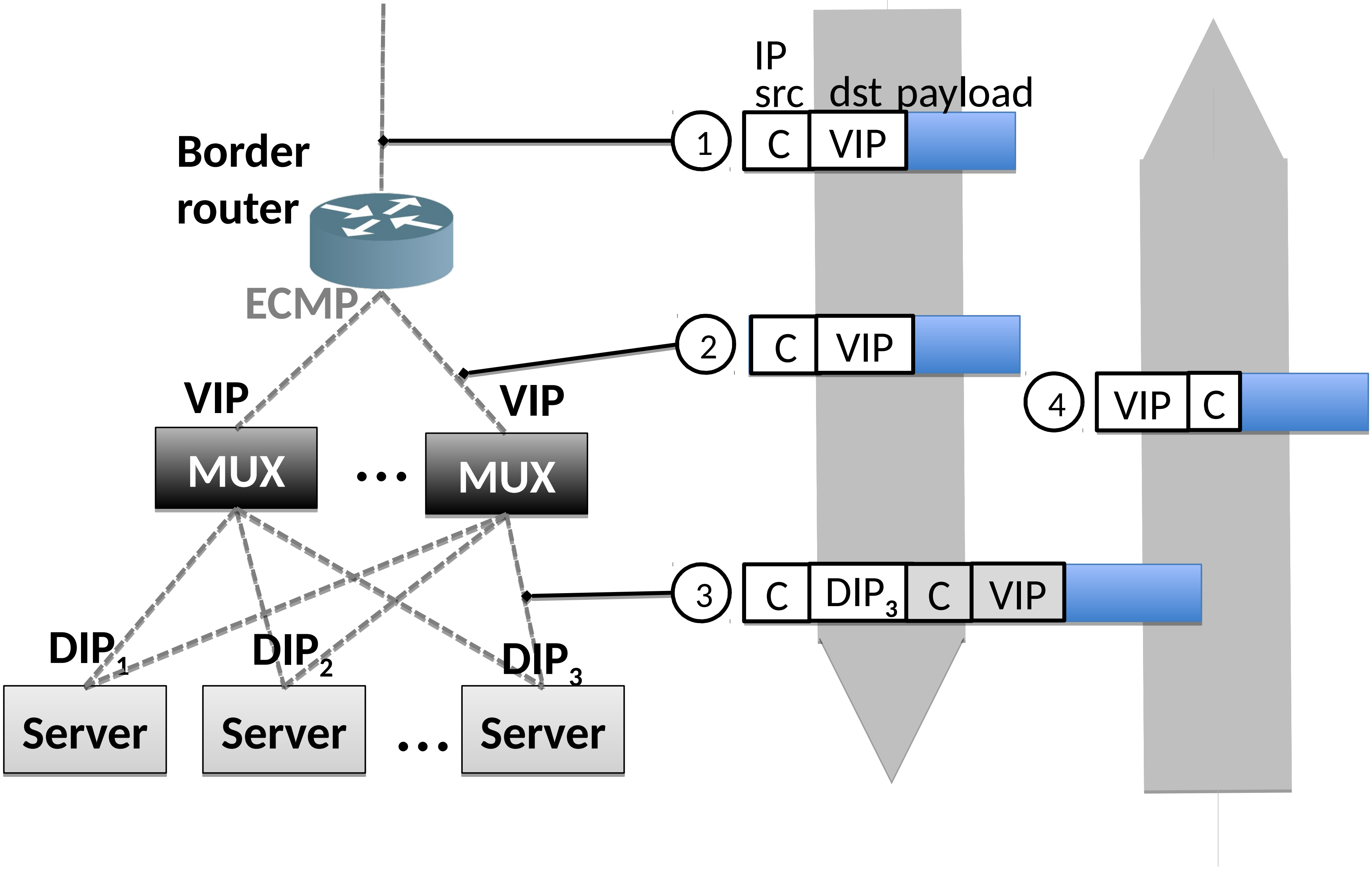
Datacenter-scale load balancing for Multipath TCP

Costin Raiciu

Joint work with Vladimir Olteanu

University Politehnica of Bucharest

Problem statement



Load balancing at scale today

- MUXes apply hash to each packet’s 5 tuple to select appropriate DIP

  => All packets of the same TCP connection will arrive at the same DIP is selected

Load balancing MPTCP

- Hashing the 5 tuple will sent additional subflows to different servers

  => All subflows but the first one will break.

Towards a solution

- **Part 1:** steer SYNs to the appropriate server, without requiring coordination across MUXes

- **Part 2**: steer data packets to the appropriate server
  - Easy if you install state at the MUX after SYN

Solution 1 [HotMiddlebox 2016]

- PRE:
  - Split 32 bit space across all DIPs
- On SYN(MPC)
  - MUX generate new random key, computes B’s token
  - Uses token to select DIP
  - Encapsulates key in SYN and sends it to DIP
  - DIP recovers key from SYN and uses it
- On SYN(JOIN)
  - MUX uses token to select appropriate DIP

Solution 2

- PRE: split 16 bit port space across all servers
- On SYN (MPC) towards **(VIP, 80)**
  - Hash 5 tuple and select DIP
  - Forward to DIP
  - DIP establishes connection
  - DIP sends ADD_ADDR with **(VIP,NEW_PORT)** where NEW_PORT is in DIPs assigned port range

Solution 2

- On SYN (JOIN) towards **(VIP, 80)**
  - Send RST or drop SYN
- On SYN (JOIN) towards (VIP,NEW_PORT)
  - Send to appropriate DIP

Handling Data Packets Statelessly

- **Solution 1**
  - Must encode server ID in every packet
  - We use 12 least significant bits from timestamp option
- **Solution 2**
  - Simply use port number to decide
  - Dst port=80? Hash and select DIP
  - Dst port!=80? Use dst port to select DIP.

Conclusions

- There is more than one viable way to handle datacenter-scale load balancing for MPTCP
- Can even load balance statelessly
- Our prototype
  - Is completely stateless
  - Handles ~30Gbps per MUX.

- To discuss: security issues?