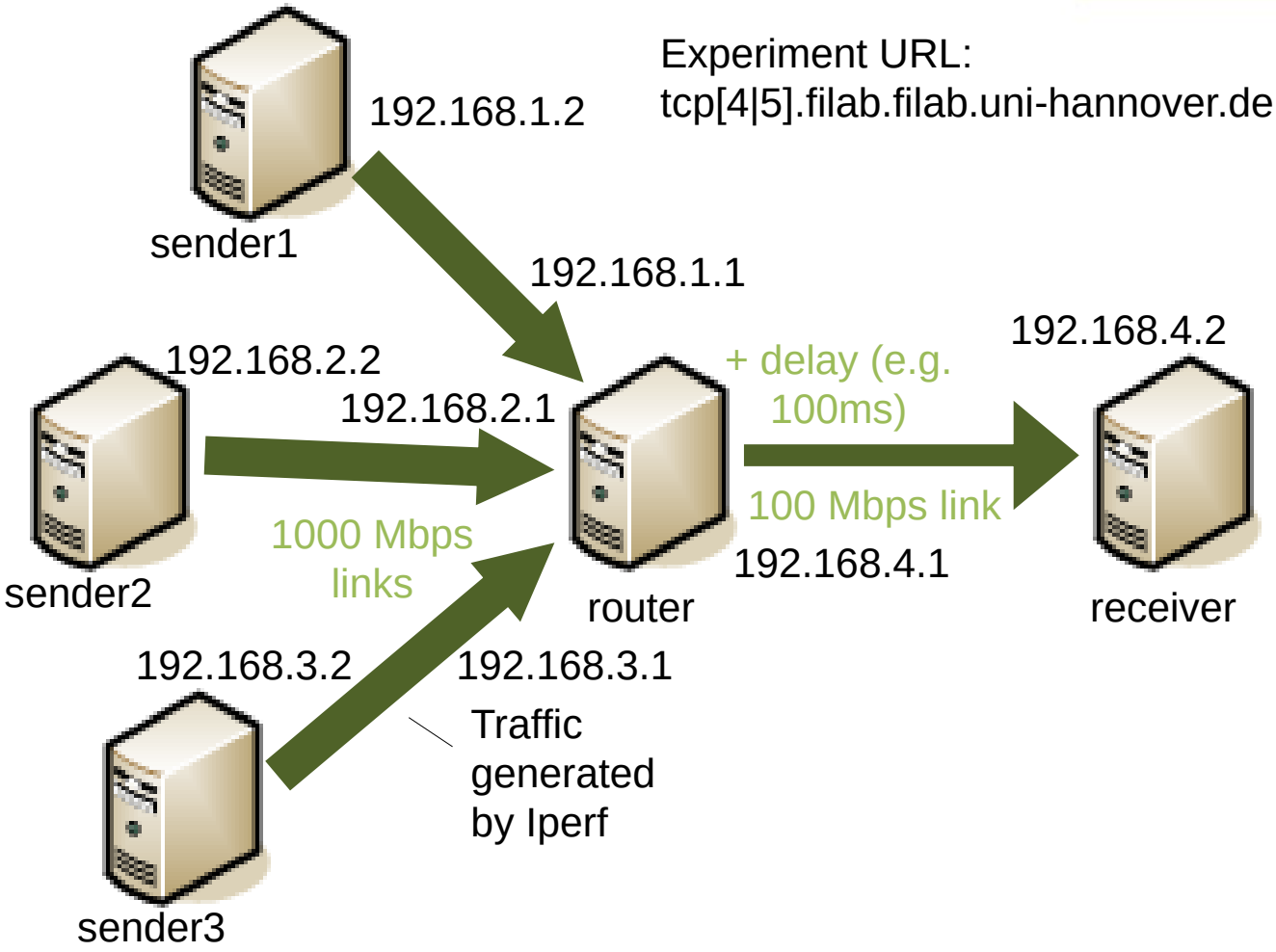
Institut für Kommunikations-Technik

Experiments with
TCP Congestion Control

Future Internet Communication Technologies

Leibniz Universität Hannover

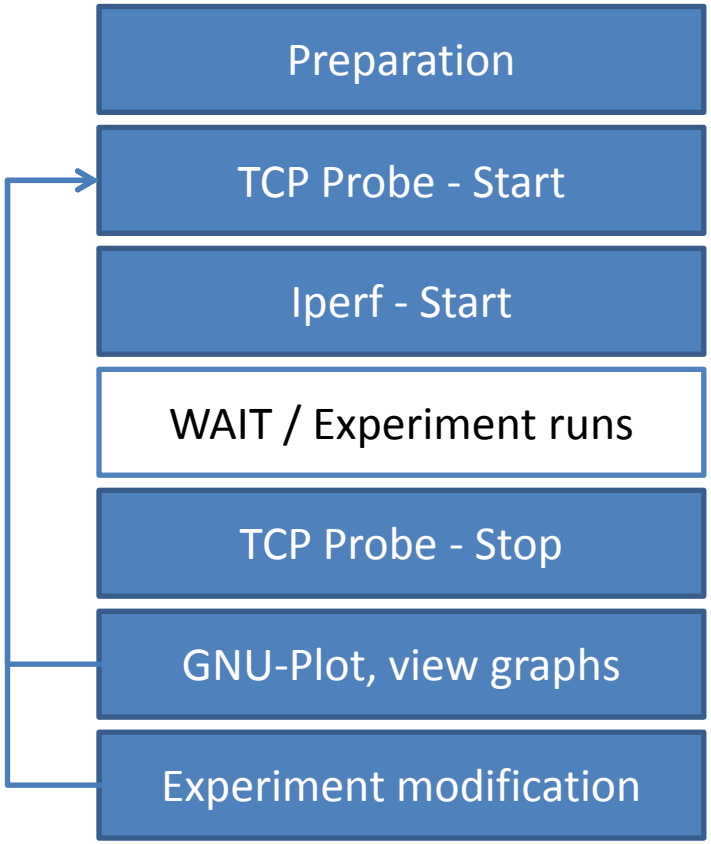
Overview / Experiment Topology



Login, e.g. ssh [-X] student@sender1.tcp4.filab.filab.uni-hannover.de , password: “fi2016”

Workflow and Groups

Experiment workflow:



Group/node assignments:

Laptop ID	Group	Experiment	Node
..11	1	tcp4	sender1
..12	1	tcp4	sender2
..13	1	tcp4	sender3
..14	1	tcp4	router, receiver
..15	2	tcp5	sender1
..16	2	tcp5	sender2
..17	2	tcp5	sender3
..18	2	tcp5	router, receiver

Overview / Components and Tasks

- Open terminal connections according to your node / task assignments
  - *Iperf* servers and *Gnuplot* will be blocking the terminal
- At the receiver: *Iperf* servers for receiving traffic (TCP/UDP)
- At the router: *Linux traffic control (TC)* for generating loss and delay, modifying buffer size
- At the senders:
  - *Iperf* clients for sending traffic
  - *TcpProbe* for packet capturing
  - *SysCtl* for changing the congestion control model
  - Any sender: *GnuPlot* for generating graphs and *XView* for displaying (Note: Latter requires ssh login with -X)

- Preparation
  - In general: cd to your working directory,e.g.: *cd ~/group2*
  - Receiver: *Iperf* server for receiving packets, keeps listening until closed
    - Example: TCP-Port=5001 (default), UDP-Port=5006

      For TCP: *iperf -s -p 5001*

      For UDP: *iperf -s -u -p 5006*
  - Sender: Configure TcpProbe, example for port 5001

    sudo sysctl -w net.ipv4.tcp_no_metrics_save=1

    sudo modprobe tcp_probe port=5001

    sudo chmod 444 /proc/net/tcpprobe

Preparation (2)

- Preparation
  - Router: Prepare traffic control for the corresponding interface to reach receiver (=network 192.168.4.0)
    - Identify interface: *ifconfig*
    - Register interface (example = eth43) at TC:

      sudo tc qdisc add dev eth43 root netem

Preparation (3)

- Preparation
  - At one node of your choice: Start the *GnuPlot - shell* (keep open in separate window)

    gnuplot -persist

  - Configure GnuPlot

    set terminal png

    set style data linespoints

    show timestamp

    set xlabel "time (seconds) “

    set ylabel "Segments (cwnd) “

    set datafile separator ” ”

TCP Probe

- At the sender: Run TcpProbe
  - While tcpprobe is running, probe data will be written into a file
- Start; probe data file = *./tcpcap[num sender].out*

  e.g., sender2: cat /proc/net/tcpprobe >./tcpcap2.out & set TCPCAP=$!
- Stop (later!)
  - $TCPCAP has stored the process ID at the start

  kill $TCPCAP
- Not sure if there are still other procs?

  ps -ef | grep tcpprobe; echo $TCPCAP

Iperf

- Let Iperf send data from the sender to the receiver
  - Remember the port numbers you have used in the preparation phase (to configure the Iperf-servers and TCP Probe) and the IP adress of the receiving host/interface

- TCP
  - Example: Receiver has IP 192.168.4.2; Iperf listens at port 5001; Packets will be sent for a duration of 60 seconds

  iperf -c 192.168.4.2 -p 5001 -i 1 -t 60

- UDP
  - Example: Receiver has IP 192.168.4.2; Iperf listens at port 5006; Datagrams will be send at 100Mbps for a duration of 30 seconds

  iperf -c 192.168.4.2 -p 5006 -i 1 -u -b 100M -t 30

GnuPlot

- In the GnuPlot-shell: Generate a plot using the probe data; example for sender 2:

  set output "./plot2.png“

  plot "./tcpcap2.out" using 1:7 title "snd_cwnd“

  - Zoom in, e.g. 10..15 seconds: *set xrange [10:15]* and repeat steps above

- Not sure if there are data in the probe file?

  *ls -al *cap**

- In any terminal: Display the generated graphs

  xview ./plot2.png &

  Regenerate plots with different x range if needed

Experiment Modifications

- Change congestion control model (at the source)

  sudo sysctl -w net.ipv4.tcp_congestion_control=reno

  (or vegas, bic, cubic)

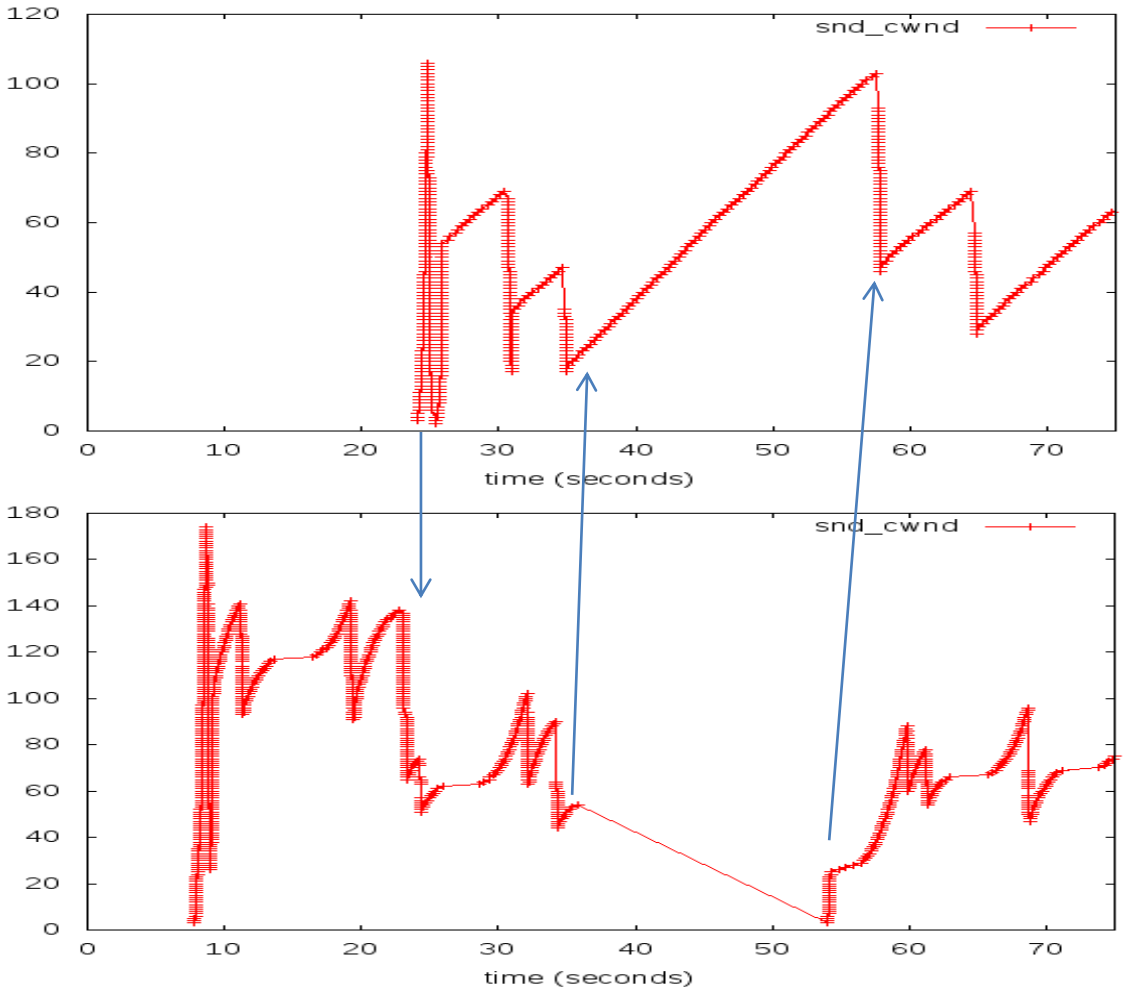
  - Retrieve the current by using: *sysctl net.ipv4.tcp_congestion_control*

- Modify buffer size, packet loss and delay (at the router), example: device=eth43, loss=0.1%, delay=10ms, buffer size = 17 packets

  sudo tc qdisc change dev eth43 root netem limit 17 loss 0.1% delay 10ms

Example

- Sender 1
  - TCP-Reno
  - Start at ~25s
- Sender 2
  - TCP-Cubic
  - 1. Start at ~8s for 30s
  - 2. Start at ~55s



Experiments

- Single traffic source
  - Watch the graphs for the congestion window at
    - different congestion control models (cubic, bic, vegas)
    - At modified delay at the router: 0, 10, 100ms
- Multiple traffic source
  - Generate UDP cross traffic from another sender (Keep the TCP source from the previous experiment sending)
    - Modify the rate of the UDP traffic to 50, 5, 1, 0 Mbps
  - TCP: Generate TCP cross traffic from other sources
    - Start parallel capturing at the senders at the same time and compare the graphs