Universität Stuttgart

INSTITUT FÜR KOMMUNIKATIONSNETZE UND RECHNERSYSTEME
Prof. Dr.-Ing. Dr. h. c. mult. P. J. Kühn

Experience with Simulating Real TCP/IP-Protocol Stacks

Michael Scharf, Christina Zeeh
Institute of Communication Networks and Computer Engineering
University of Stuttgart
michael.scharf@ikr.uni-stuttgart.de

ITG Fachgruppe 5.2.1 Workshop "Network Engineering" - Essen, November 22, 2007

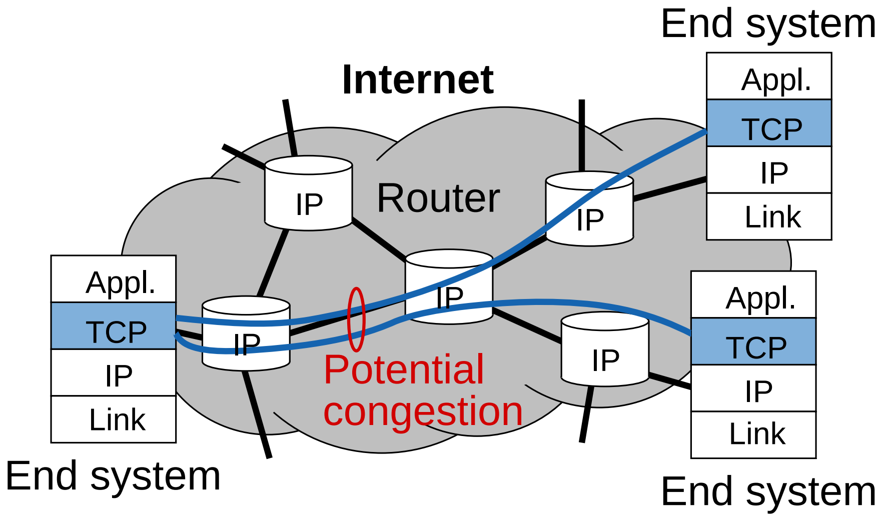
This work is partly funded by the German Research Foundation (DFG) through the Center of Excellence (SFB) 627 "Nexus".

Outline

- **Motivation: Current and future Internet**
- **Towards accurate simulators**
- **Network Simulation Cradle**
- **Accuracy and performance tests**
- **Work-in-progress**
- **Conclusion and future work**

Motivation: Current and Future Internet

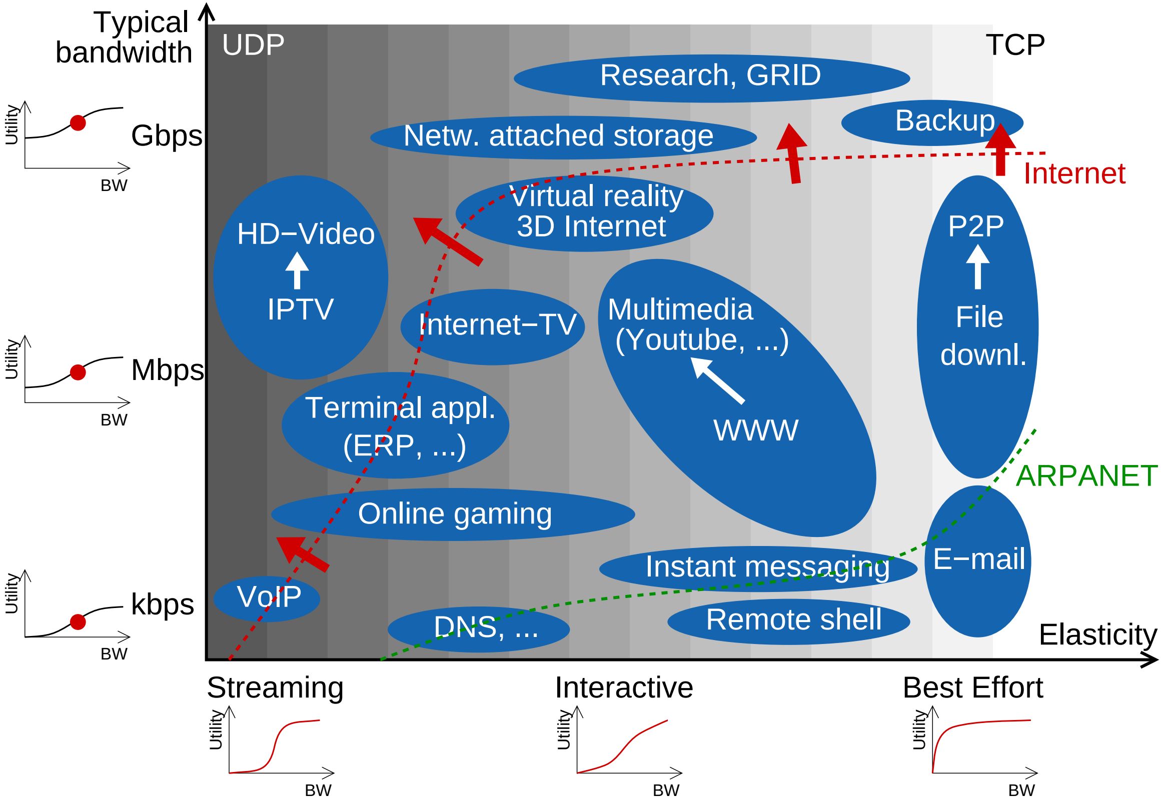
The Internet



- **A highly decentralized, imperfect, and extremely successfull system**
- **Design principles**
  - End-to-end argument
  - Resource management by TCP congestion control (no QoS!)

Motivation: Current and Future Internet

Internet Applications



Further Details: Michael Scharf, "Future Internet Transport Layer - Heading towards a Post-TCP Era?", Future Internet Design Workshop, ECOC, Sept. 2007

Motivation: Current and Future Internet

Design Principles of the Internet Congestion Control

- **Sender-side control of data rate by TCP congestion window**
- **Greedy probing of available bandwidth on path (window increase)**
- **Implicit congestion feedback by packet loss (window decrease)**

➥ **Basics almost unchanged since V. Jacobson's proposal from 1988**

Challenges

- **Broadband wide area networks**
  - Large bandwidth-delay product
  - Extreme variety and dynamics (sensor networks to highspeed photonics)
- **Fairness (network neutrality debate)**
- **Network demanding applications**
- **...**

Further details: Michael Welzl, Dimitri Papadimitriou, Michael Scharf, "Open Research Issues in Internet Congestion Control", IETF internet draft, work in progress, July 2007, draft-irtf-iccrg-welzl-congestion-control-open-research-00.txt

Motivation: Current and Future Internet

Recent TCP Research and Standardization

Incremental → Evolution → Revolution

- TCP tweaking (always)
- Highspeed−TCP (2003)
- AQM (1998)
- ECN (2001)
- Quick−Start (2007)
- re−ECN (?)
- XCP (?)

➥ **Lots of ongoing work!**

The "Credibility Gap" of Internet Simulations

- **TCP is the basic Internet transport protocol... and inherently complex**
- **How to investigate new network and transport protocols?**
  - Real development in testbeds not always possible
  - Faster "time-to-paper" of simulations
- **However: Lack of accurate (TCP) simulators**
  - Many missing features
    - Unidirectional transfer only
    - Constant packet size
    - No flow control
    - ...
  - Seldomly validated
  - Real-world stacks always differ to specs and permanently evolve

➥ **Possible remedy: Direct execution of real TCP/IP stack code in simulations**

Towards Accurate Simulators

Recent Research Efforts

← Difference from real code

Abstract model	Reimplementation	Code extraction	Stack integration	Runtime emulation
− Mathematical model − High−level behavior, e. g. fluid−flow model − Greedy sources	− Separate code basis − Simplified functions (flow control, conn. management, ...) − Synthetic app. models	− Hybrid code basis − Only parts directly affecting performance (cong. control) − Synthetic app. models	− Network stack ported to sim. environment − Based on FreeBSD or Linux kernel − Synthetic app. models	− Kernel or user space emulation − May require modifi− cation of host kernel − Real applications
− Various tools, often Matlab based	− Plain ns−2 simulator − Various other special TCP simulators, such as OPNET, ... − IKR tcplib	− NS−2 TCP Linux (Caltech, since 2006)	− NSC (U. of Waikato, since 2003) − OppBSD (Karlsruhe, since 2004) − Lunar (Virginia Tech, 2004)	− NCTUns (Chiao Tung/ Harvard, since 1999) − UML Simulator (2003) − dummynet, NIST Net, netem, ...

Complexity/cost →

Challenges of the Direct Code Execution Approach

- **Moving code from kernel-space to user-space**
  - No priviliged CPU instructions
  - Many kernel subsystems not needed (e. g., memory management)
- **Multiple stack instances in simulators**
  - No global variables
  - Multi-tasking, threading, and scheduler difficult to model
- **Simulator interface**
  - Virtual time: Timer interrupt replaced by simulator events
  - Full packets transport, instead of function calls
  - Byte-stream socket interface vs. message-oriented simulators
  - Programming language mismatch (e. g., C vs. C++ code)

Comparison of Recent Solutions

Approach	Code basis	Maturity	Accuracy	Performance	Extensibility
NS−2 TCP Linux	Linux 2.6.13+	+	o	++	o
NSC	Linux, *BSD, ...	+	+	+	+
OppBSD for OMNeT++	FreeBSD 6.0	+	+	+	o
Lunar	Linux 2.4.3	−	o	?	−
NCTUns	Linux/BSD	+	+	o	o
UMLSimulator	User−mode Linux	−	?	−	o

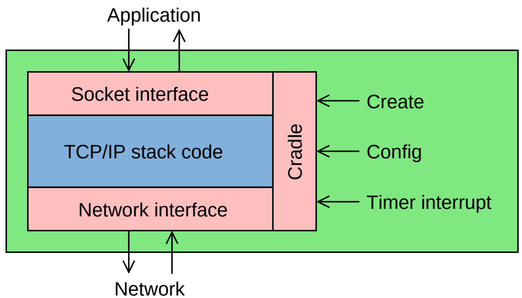
Network Simulation Cradle (NSC)

Overview

- **Developed by Sam Jansen at University of Waikato, Hamilton, New Zealand, since 2003**
- **Supports TCP stacks of Linux (2.4, 2.6.14.2), FreeBSD, OpenBSD, lwIP**
- **Frontend to ns-2 simulator**

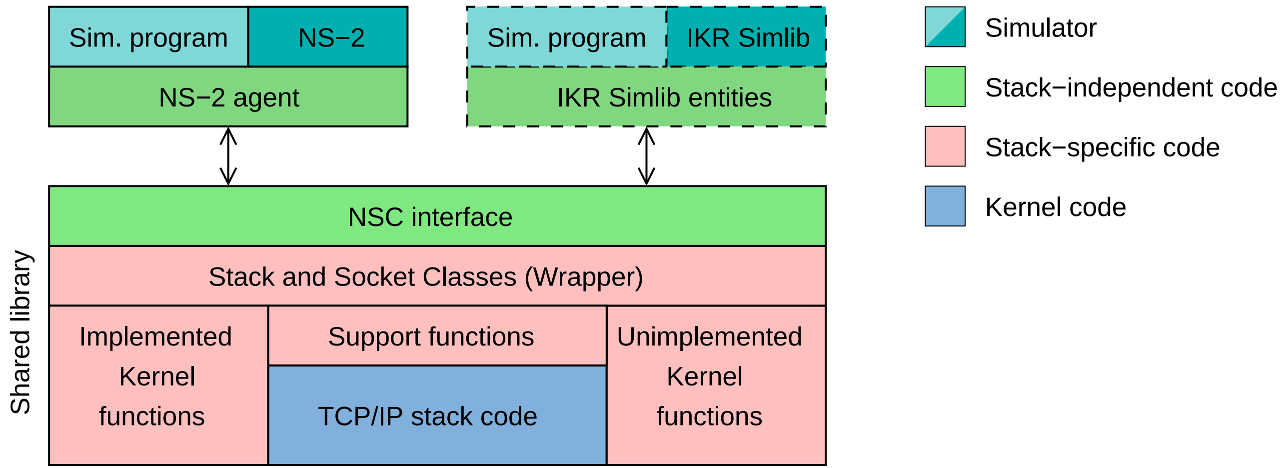
Features

- **Powerfull automatic C parser**
  - Substitution of global variables
  - Can be adapted to new stacks
- **Supports execution of different stacks in parallel**
- **Good documentation**
- **Well validated**



Network Simulation Cradle (NSC)

Simulator Structure

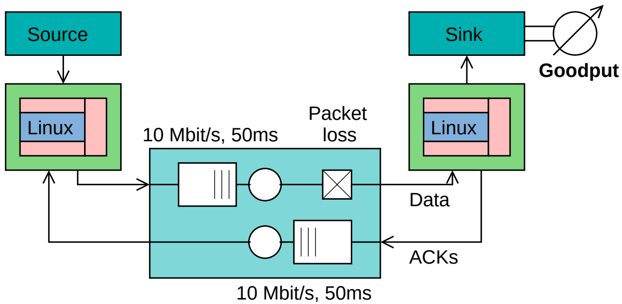


Extensions

- **Flexible towards other simulators**
  ➥ Implementation of new frontend to IKR simlib with rather limited effort
- **Integration of new protocol stacks**
  ➥ Can be quite time-consuming

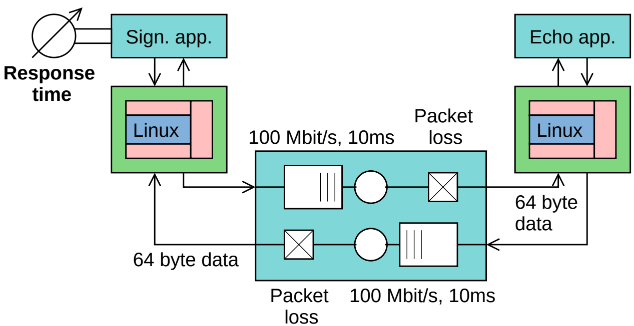
Accuracy and Performance Tests

Scenario 1: Goodput of Greedy Source



- One TCP connection with greedy source
- Ethernet with MTU of 1500 byte
- Buffer size of 1000 packets
- Simulation: Linux 2.6.14.2
- Measurement: Linux 2.6.20.20 on P4 PC, "netem" network emulation

Scenario 2: Head-of-Line Blocking (HOL) in Signaling

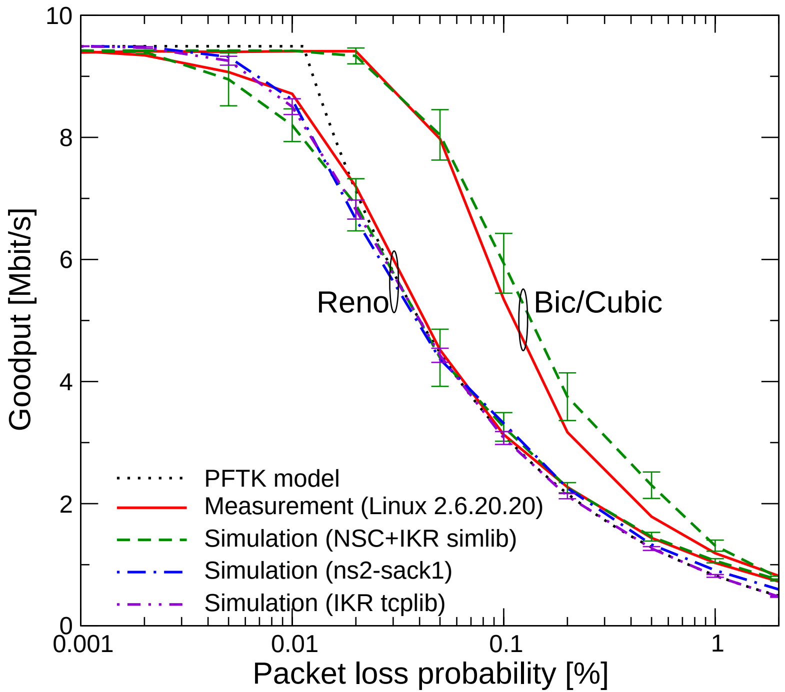


- One TCP connection
- Request-response of 64 byte messages with neg.-exp. IAT of 10 ms
- Ethernet with MTU 1500 byte
- Buffer size of 1000 packets
- Socket option "NODELAY"
- Simulation: Linux 2.6.14.2
- Measurement: Linux 2.6.20.20 on P4 PC, "netem" network emulation

➥ **Scenarios similar to validation tests of Sam Jansen**

Accuracy and Performance Tests

Result Scenario 1: Goodput of Greedy Source

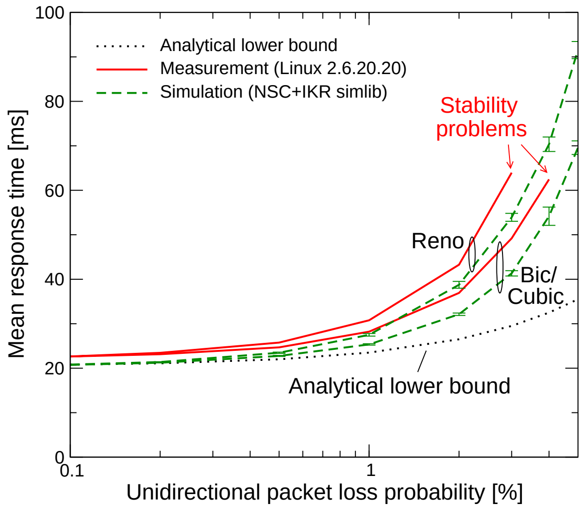


➥ **High accuracy, also for new congestion control algorithms**

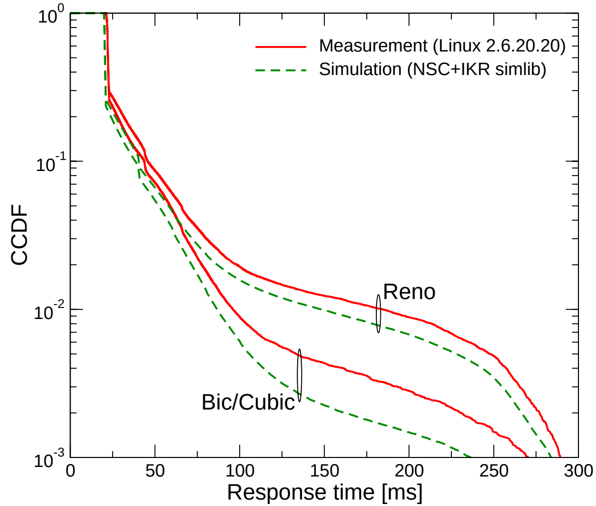
Accuracy and Performance Tests

Result Scenario 2: HOL in Signaling

Mean response time



Distribution at 1% loss

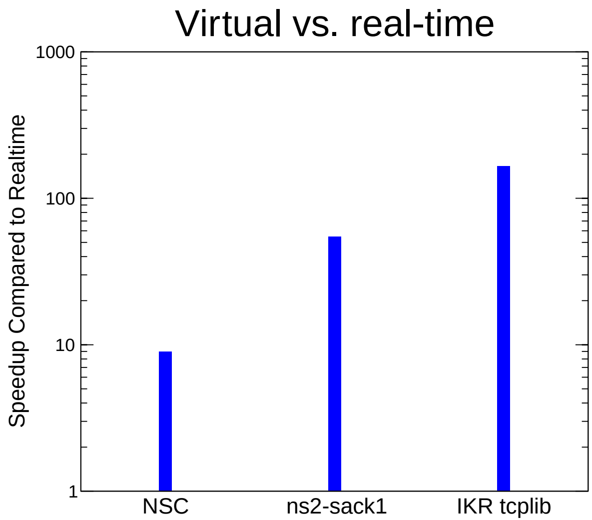


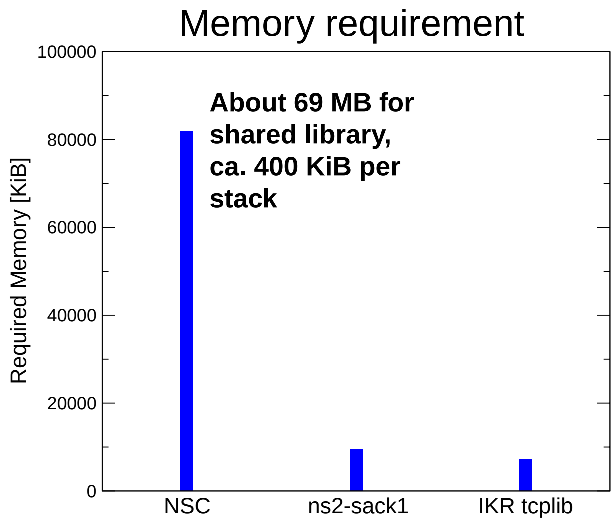
➥ **Response time in simulations less than in reality**

➥ **Reasons: Kernel scheduler delays, network emulation errors**

Further Details: Michael Scharf, Sebastian Kiesel: "Head-of-Line Blocking in TCP and SCTP: Analysis and Measurements", Proc. IEEE Globecom, San Francisco, CA, USA, Nov. 2006

Overhead and Cost





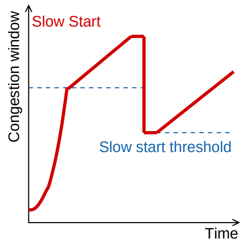
Scenario: Test case 1 at 0.1% loss; Plattform: Intel P4 2.8 Ghz, 2 GB RAM, Ubuntu 7.04

- **Significant overhead of NSC compared to abstract simulators**
- **Improvement possible**
  - Decrease timer interrupt frequency
  - Usage of pseudo data as payload of packets

Work-in-Progress

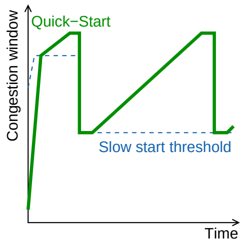
Quick-Start TCP Extension (RFC 4782)

Slow-Start:



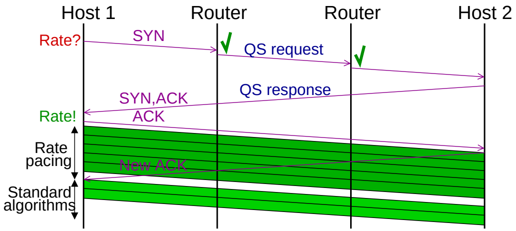
- One pillar of TCP congestion control
- Exponential window growth

Quick-Start:



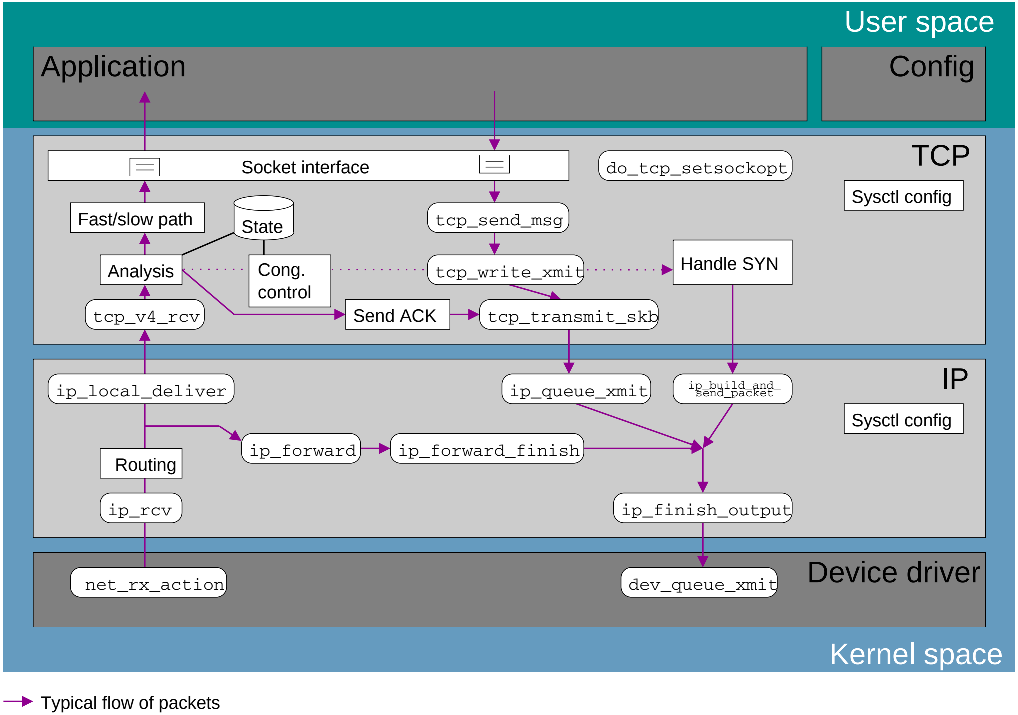
- Recent experimental TCP extension
- (Almost) immediately use large window

- **Speeds up interactive WAN applications**
  - After connection setup or idle periods
  - For large bandwidth-delay products
- **Reality check**
  - Requires support in *all* routers
  - Some open (research) issues



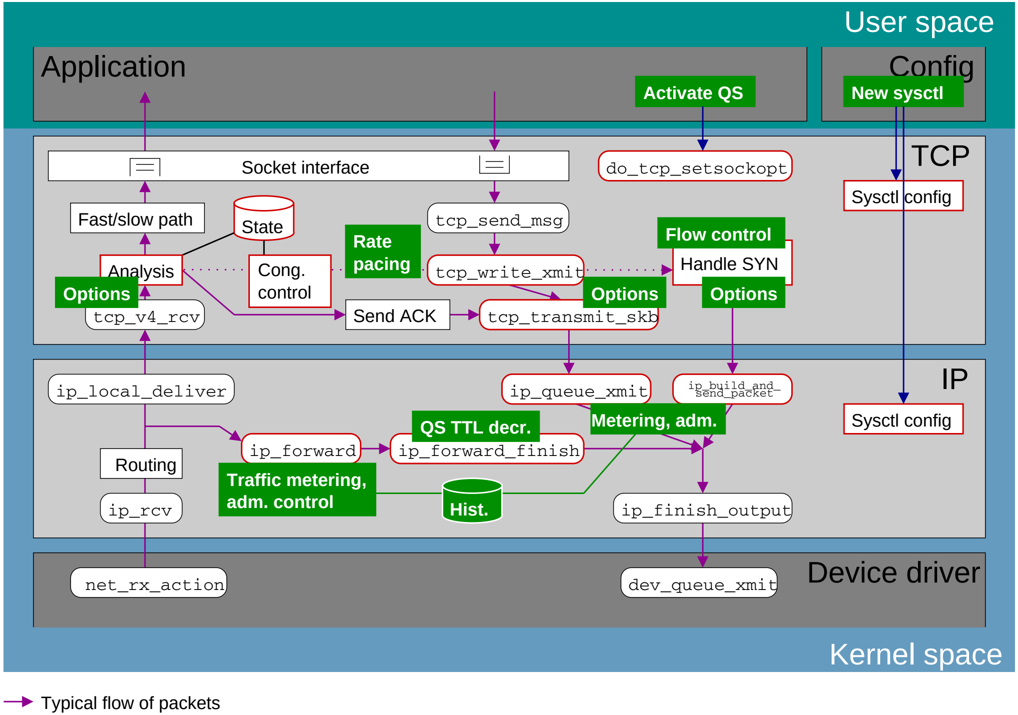
➥ **May be an option for future broadband IP networks**

Required Linux Kernel Modifications



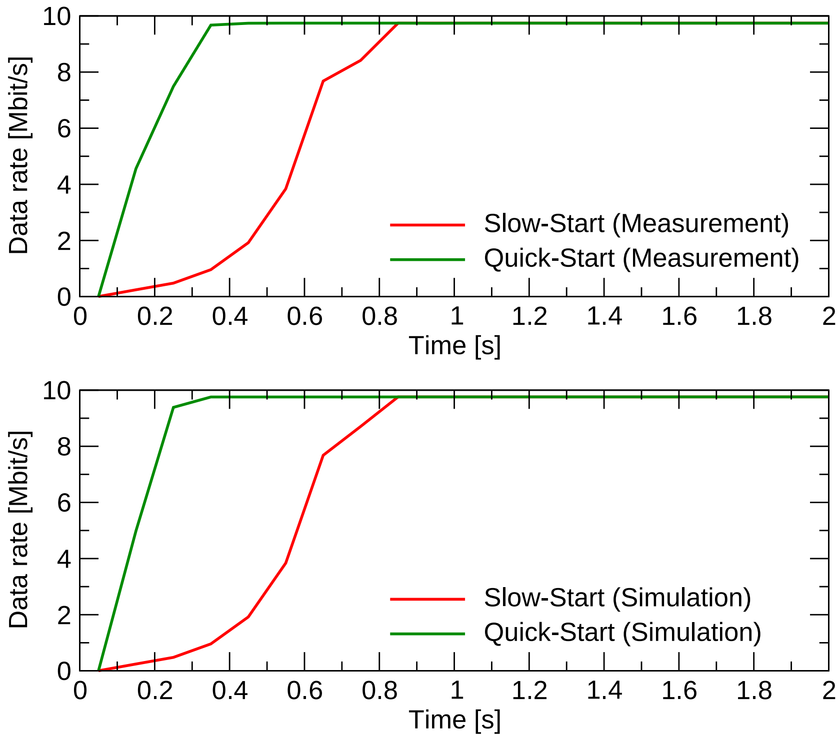
Further details: Michael Scharf, Haiko Strotbek, "Experiences with Implementing Quick-Start in the Linux Kernel", Presentation at IETF 69, TSVAREA, Chicago, IL, USA, July 2007

Required Linux Kernel Modifications



Further details: Michael Scharf, Haiko Strotbek, "Experiences with Implementing Quick-Start in the Linux Kernel", Presentation at IETF 69, TSVAREA, Chicago, IL, USA, July 2007

Measurement vs. Simulation



Scenario

- **10 Mbps Ethernet**
- **100 ms RTT**
- **Simulation: Linux 2.6.14.2, Reno congestion control**
- **Measurement: Linux 2.6.20.20, Reno congestion control, "netem" network emulation**
- **(Almost) same kernel patch**
- **Quick-Start request for 5.12 Mbit/s**

➥ **Support of kernel prototype software development**

Conclusions and Future Work

Conclusions

- **Simulating TCP/IP networks is challenging**
- **(More) Accurate simulators by direct execution of TCP/IP stack code**
- **Promising solution: Network Simulation Cradle**
  - Extensible, supports many stacks
  - Frontends to different simulators possible (ns-2, IKR simlib)
  - Can support experimental protocol development
- **Limitation: Less scalable than abstract simulators**

Future Work

- **Adaptation to newest stack versions**
- **Improvement of scalability and addition of features**
- **Better models for applications, kernel schedulers, ...**

Acknowledgements

Contributors

- Haiko Strotbek
- Sebastian Kiesel
- Marc Necker
- Sebastian Gunreben

The "Network Simulation Cradle" (NSC) is developed by Sam Jansen at University of Waikato, Hamilton, New Zealand.