IKR SimLib-QEMU: TCP Simulations Integrating Virtual Machines

ICCRG – 87. IETF Berlin – July 31, 2013

Thomas Werthmann <thomas.werthman@ink.uni-stuttgart.de>
Mirja Kühlewind <mirja.kuehlewind@ikr.uni-stuttgart.de>
Matthias Kaschub <matthias.kaschub@ikr.uni-stuttgart.de>
Sebastian Scholz <sebastian.scholz@ikr.uni-stuttgart.de>
David Wagner <david.wagner@ikr.uni-stuttgart.de>

Universität Stuttgart
Institute of Communication Networks
and Computer Engineering (IKR)
Prof. Dr.-Ing. Andreas Kirstädter

Overview

- Motivation: Realistic TCP Simulations
- IKR SimLib-QEMU: Concept and Implementation
  - QEMU Extensions
  - Relaying Operating System Interfaces
  - QEMU Adapter in the Simulation
  - Control Flow and Reproducibility of Simulation Results
- Initial Evaluation against ns-3/NSC
- Conclusion

Realistic TCP Simulations

Possible Approaches

- Abstract model of application and transport (TCP) behavior

  → Do often not reflect real systems well due to complexity and rapid development of TCP
- Emulation in real time or slower (computers connected to simulated networks)

  → Dependency on the real clock base and possible influences of other processes
- Re-implementation of standards or re-use of OS code (e.g. ns-2/3, ns-3/NSC)

  → High maintenance effort and validation is difficult

→ Use of unmodified OS and application code

ns-3 with Network Simulation Cradle (NSC)

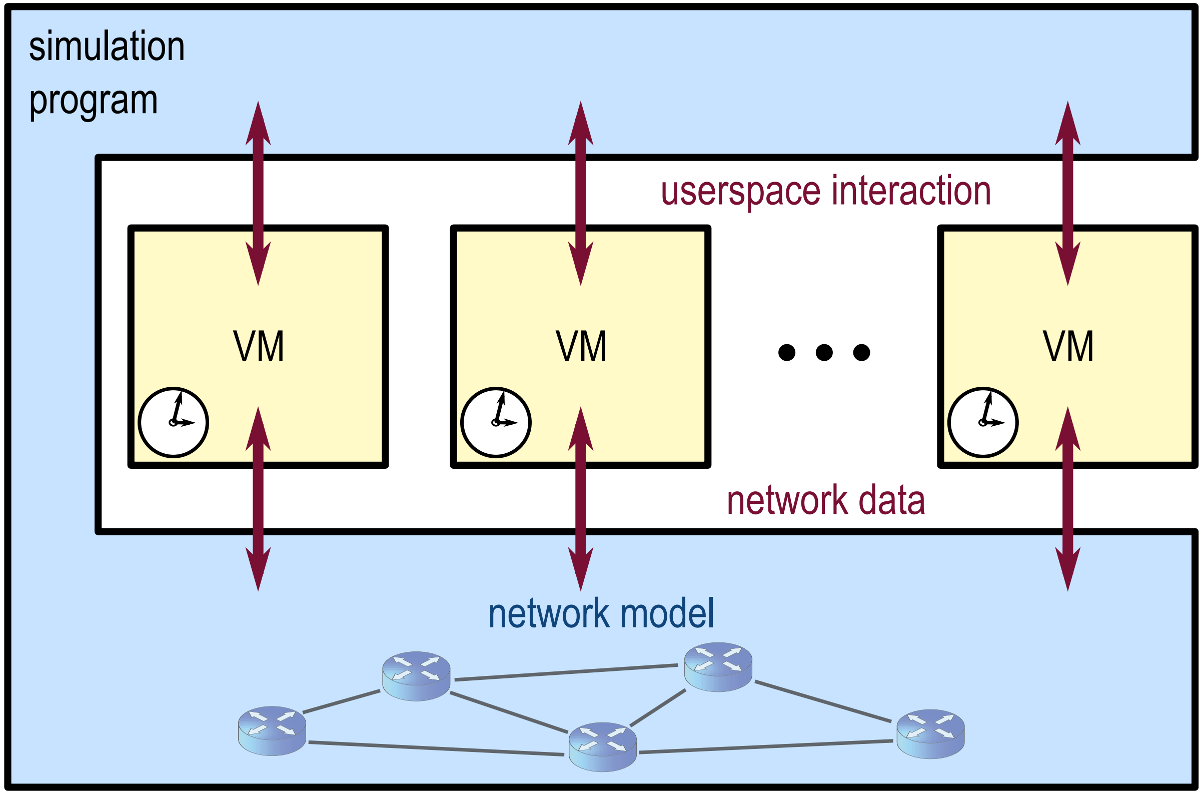
- ns-3: Most common used network simulator supporting integration of real kernel code
- Network Simulation Cradle (NSC): Framework to convert kernel code into user-space code and interface for simulation integration

  → NSC does not support a more recent kernel version than 2.6.26
- Direct Code Execution (DCE) Cradle supports version 3.4

IKR SimLib-QEMU: Concept

Overview

Integration of modified VMs running unmodified kernel and application code



Upper Layer Control and configure VMs and applications (e.g. write data to socket)

Lower Layer Intercept in/output (e.g. forward Ethernet frame to simulated topology)

IKR SimLib-QEMU: Concept

Integration into the Simulation Program

Extended QEMU intercepts all I/O and redirects it to the QEMU Adapter

e.g. for packets sent/received, keyboard/mouse events, serial/parallel connectors

QEMU Adapter provides an interface to send control commands created by simulation events

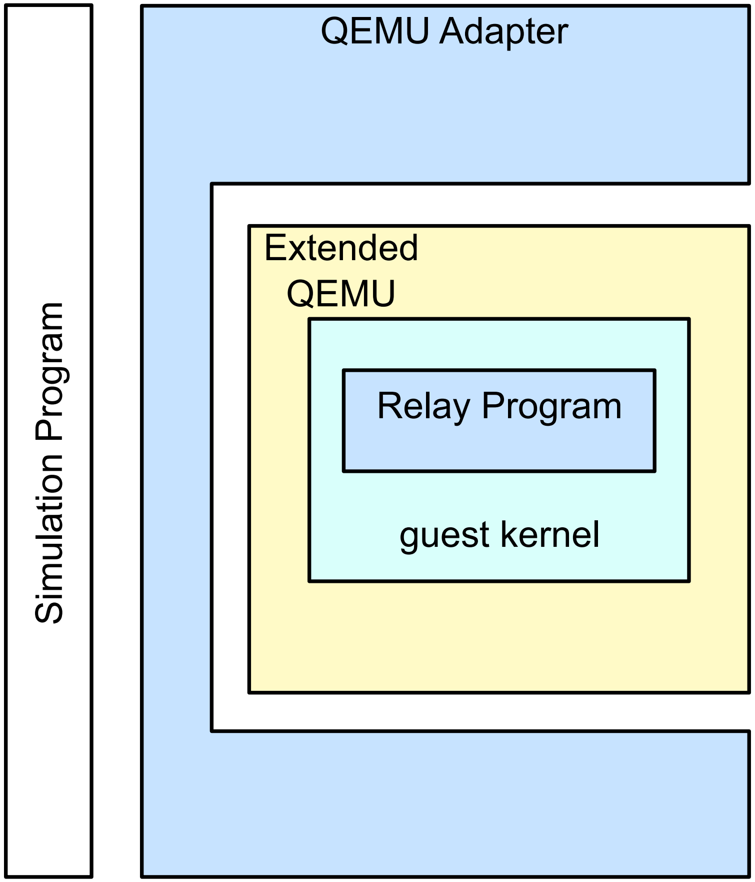
e.g. connect network interface of the VM to the simulated network

Relay Program allows to access the POSIX socket API from the simulation program

e.g. to open TCP connections and send data

Simulation Program executes global control flow and synchronizes all VMs with simulation clock

maintains events in a central calendar (event-based simulation)



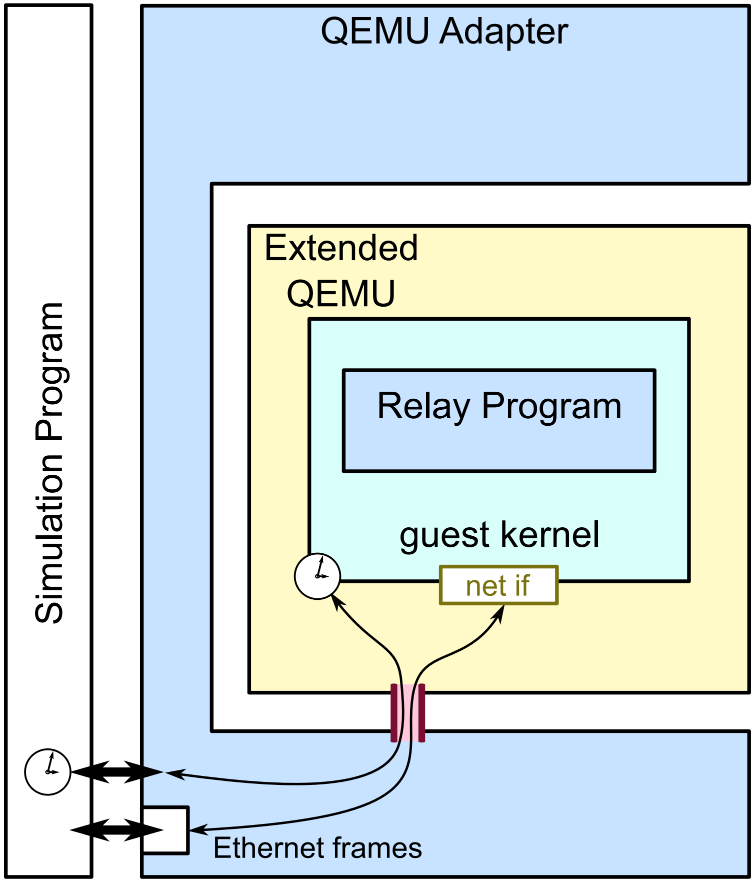
IKR SimLib-QEMU: Concept

Extended QEMU

Input and Output Interception

VM must not interact with the host’s devices!

- QEMU architecture: virtual hardware components with a virtual device and device backend
- Added own backends for network, serial I/O etc.
- All data is forwarded to or provided by the QEMU Adapter



Clock Control

Simulated time and emulated time in the VM need to match exactly!

- Virtual clock chip in VM: High Precision Timer (HPET) usually aligned to wall clock time
- Simulated time: independent of wall clock time (simulation runs slower or faster than real time)
- Patched QEMU timer code to
  - when the timer is read, simulated time from the simulation calendar is returned
  - timer interrupts of the VM schedule simulation events

IKR SimLib-QEMU: Concept

Relay Program

Relaying Socket Interface

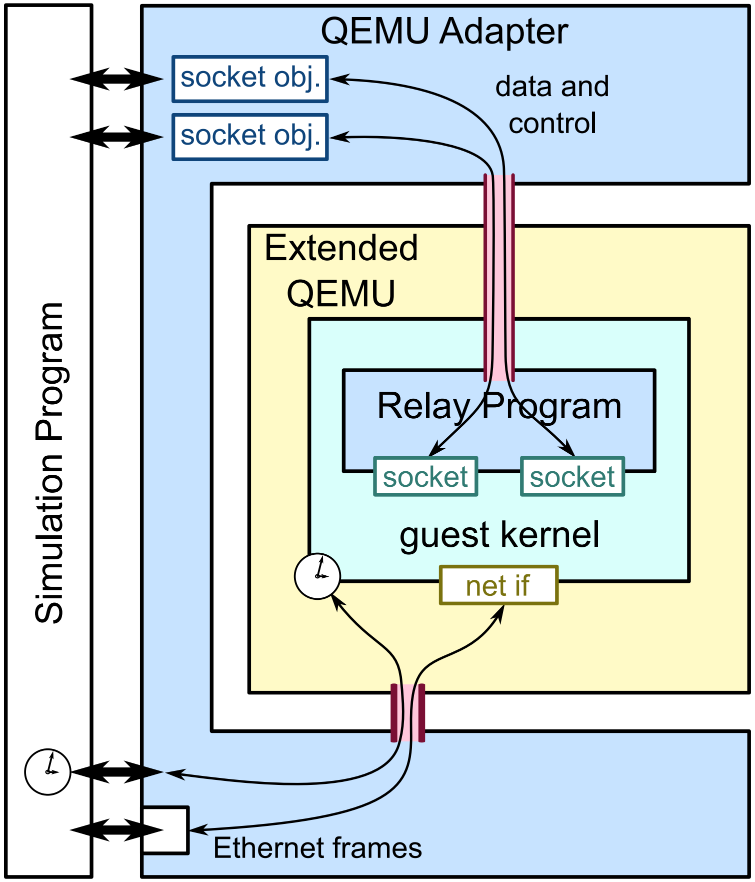
Socket interface should be usable from the simulation program for easy traffic model implementations

- Relay Program runs in the user space of the VM
- Communicates with the QEMU Adapter via a virtual serial interface
- Currently supports IPv4 / v6 API `socket()`, `bind()`, `listen()`, `setsockopt()`, `getsockopt()`, `read()`, and `write()`

System Configuration and Measurement

System should be accessible from the simulation program

- Relay Program can be used to set sysctl values
- Socket info struct can be read to extract cwnd etc.

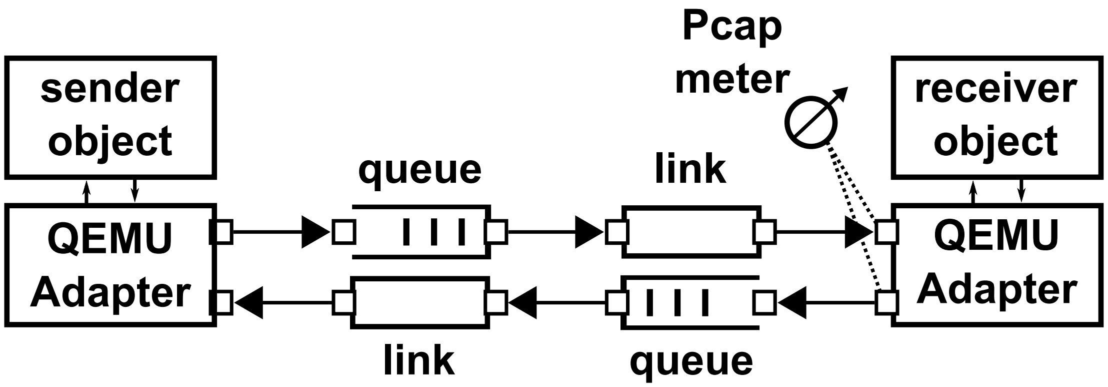


IKR SimLib-QEMU: Concept

Embedding the QEMU Adapter in the Simulation Program

IKR SimLib Simulation Environment

- Event-driven, java-based, freely available
- Based on a calendar that advances time and maintains events to trigger entities
- Entities exchange messages via unidirectional interfaces
- Library provides a set of entities to handle and manipulate messages (e.g. queues)



Usage of the QEMU Adapter

- Sender/receiver interact with socket interface of QEMU Adapter
- QEMU Adapter redirecs Ethernet frames from the VM to the simulation entities
- PcapMeter can be inserted at any point to capture Ethernet frames in pcap format

IKR SimLib-QEMU: Concept

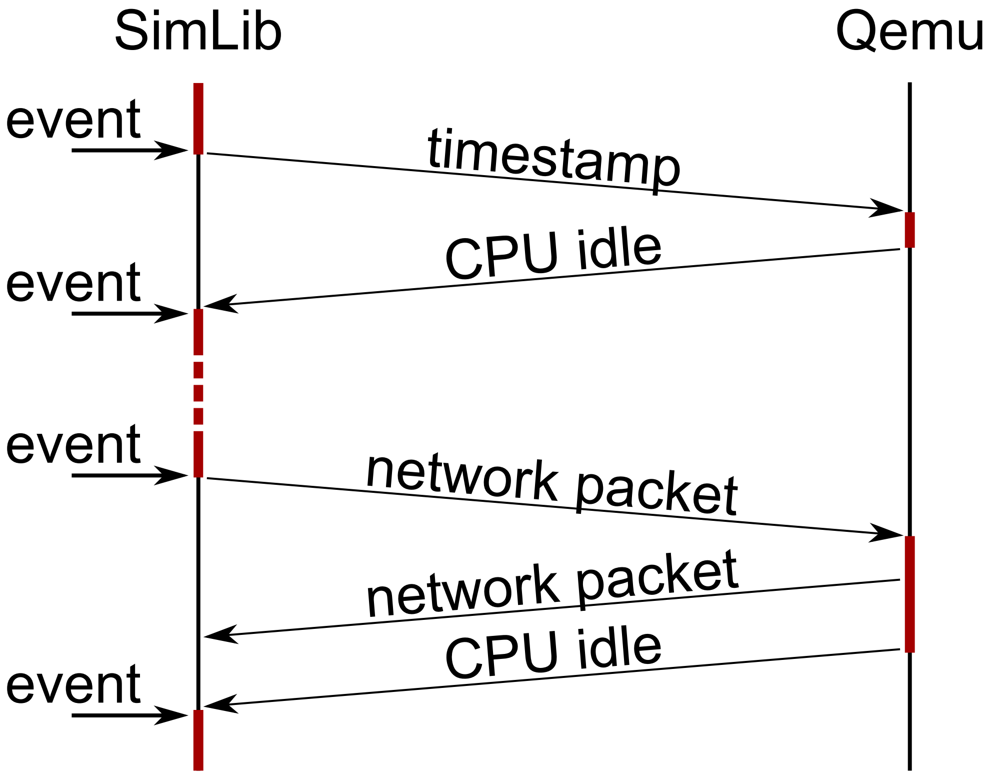
Control Flow: Interaction of SimLib and QEMU

Either the simulation or one single QEMU instance executes at a time

- Timer events correspond to events in the simulation calendar
- Simulation program wakes up the respective VM and updates the time to the current simulated time
- Each execution cycle of VM is performed in zero simulated time and time between events is skipped

  → follows paradigm of event-driven simulation

  → VM can spend nearly infinite time for computations



- Simulation program provides input and waits until CPU of VM becomes idle

→ **Strictly synchronous interaction**

→ **Virtual computer is not restricted by host CPU power**

IKR SimLib-QEMU: Concept

Reproducibility of Simulation Results

- All I/O of the VMs is controlled by the simulation program

  → No interaction with the host's devices
- The VM timer chip is directly linked to the simulation calendar

  → Simulated time is independent of wall clock time of the host
- Each VM operation happens as a result of a simulation event (socket API action, ethernet frame, timer)

  → VMs are controlled by the simulation program
- The VM is given infinite time for processing

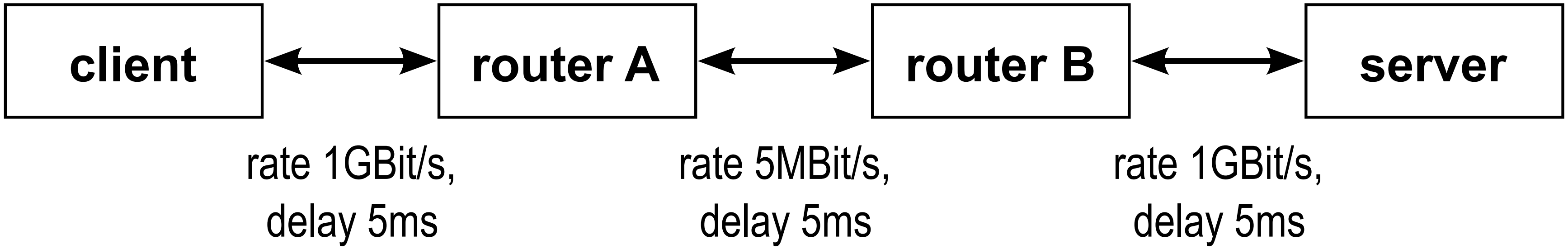
  → No interaction with the CPU load on the host

→ **VMs are fully integrated into the simulation program and are fully isolated from the host**

→ **Timing and results of the operations in the VM cannot be influenced by the environment (parallel processes, wall clock time, ...)**

Initial Evaluation

Scenario for Comparison with ns-3/NSC and different kernels



- One TCP connection using Reno congestion control
- Single bottelneck with access link limitation

Network simulation

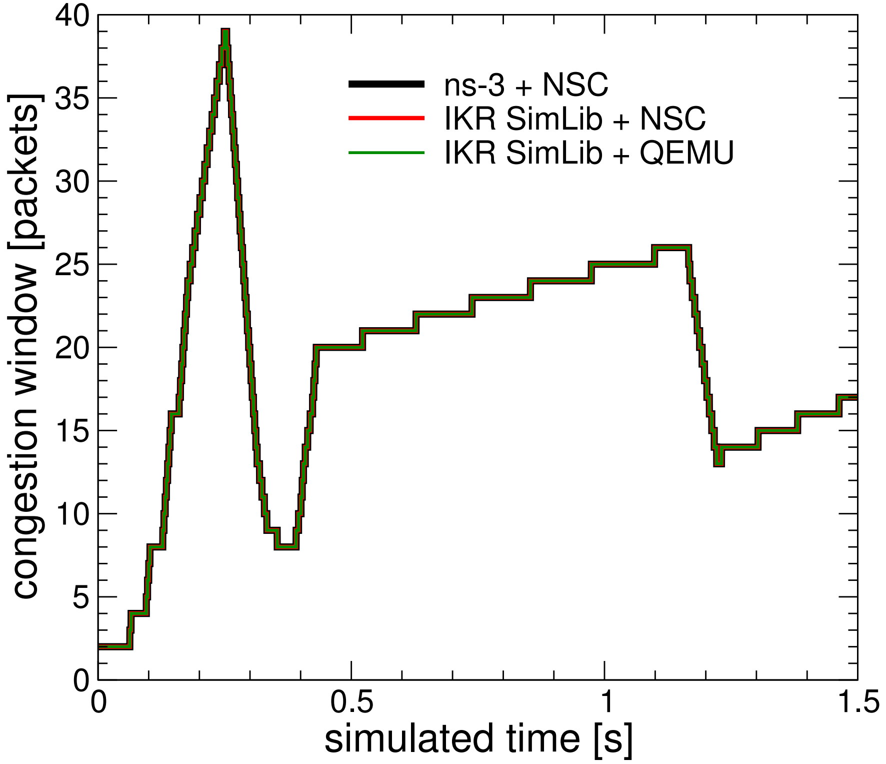
1. IKR SimLib-QEMU
2. IKR SimLib-NSC

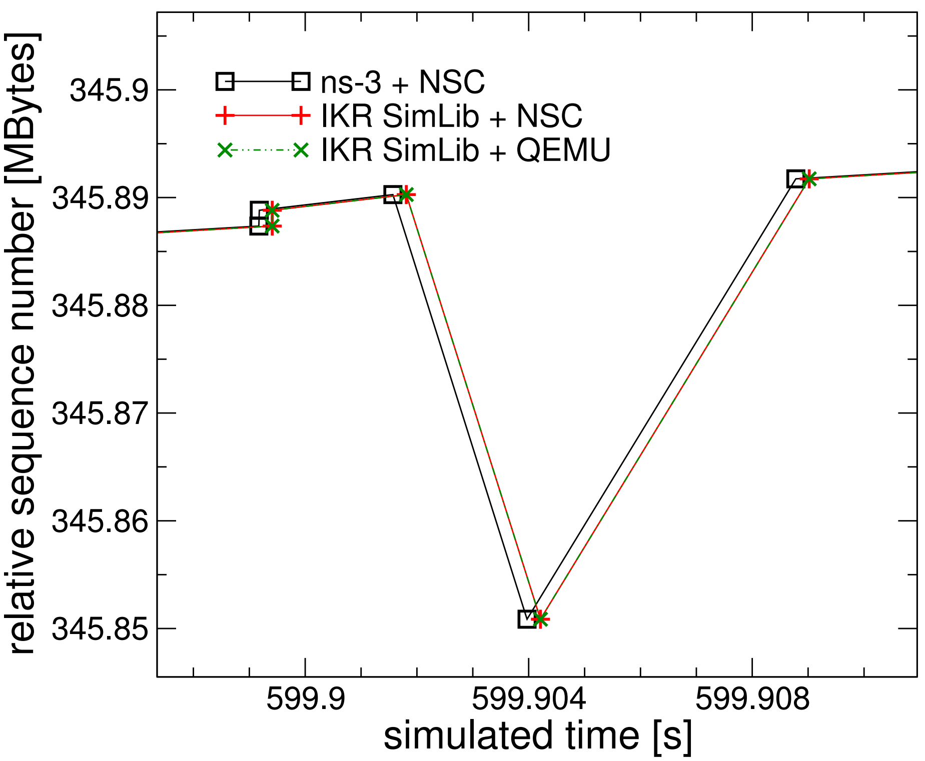
   → Remember: NSC does not support a more recent kernel version than 2.6.26
3. ns-3 + NSC

   → Most common used network simulator also supporting NSC integration

Initial Evaluation

Three different simulators



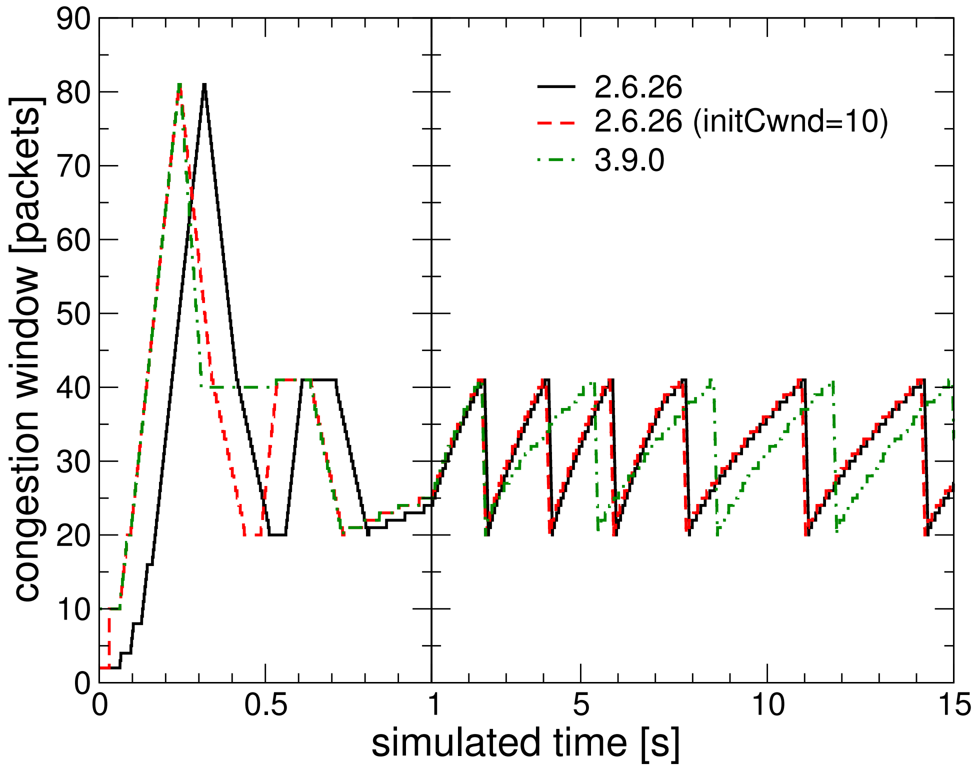


→ IKR SimLib+NSC and IKR SimLib+QEMU match exactly

→ Small drift of ns-3+NSC

Initial Evaluation

Three different kernels



→ New algorithms change TCP behavior

- Initial window of 10
- Proportional Rate Reduction

Summary and Conclusion

- VMs integrated into packet-level network simulation to investigate TCP behavior

  → Immediate use of an unmodified network stack of most recent kernel versions possible
- Extended QEMU adapts VM clock to simulated time and redirects I/O

  → Full isolation of the VM from host system guarantees reproducibility
- Relay program provides a minimal abstracting of the OS

  → Easily access from the simulation program to perform socket operation or capture network traffic

→ IKR-SimLib-QEMU provides comparable results to ns-3+NSC

→ Computation effort and memory usage are viable for TCP simulations, even in large scale network (results not shown) and further optimizations still possible

SimLib-QEMU: TCP Simulations integrating VMs

Thank you for your attention!

Questions?