Advanced Computer Networking (ACN)

Router Project – Description

Prof. Dr.-Ing. Georg Carle, Sebastian Gallenmüller

Chair of Network Architectures and Services
School of Computation, Information, and Technology
Technical University of Munich

Registration

Project

- Participation in the ACN project is optional but highly recommended to gain hands-on experience with computer networks

How do you participate?

- First, request a GitLab repository if you have not requested a repository for the exercise yet:
  `https://acn.net.cit.tum.de/auth`
- Merge requires resources from template repository:

```
git remote add template git@gitlab.lrz.de:acn/terms/2025ws/template.git
git remote update
git merge --allow-unrelated-histories template/router-project
```

Router Project
Packet processing software

Usually the network stack is part of your OS

- Entire network stack provided
- Standardized socket interface

Reasons for poor network performance over BSD sockets:

- Dynamic memory allocation
- Costly context switches (user space - kernel space)
- Copying of packet data

Known Userspace-Frameworks

- Data Plane Development Kit (DPDK)
- PF_RING ZC
- netmap
- Linux eXpress Data Path (XDP)

Acceleration techniques:

- Memory allocation only done once
- No copying of packet data
- Batch processing of packets
- Detect new packets by polling the NIC (lower number or no interrupts)
- Reduced functionality (raw Ethernet frames)

Testbeds in Computer Science

Scientific testbeds

- Platforms to implement, debug, and evaluate ideas and concepts
- Execution of experiments, e.g., benchmarking hardware and software components
- Important property: reproducibility of experiments

Testbeds in Computer Science

Scientific testbeds

- Platforms to implement, debug, and evaluate ideas and concepts
- Execution of experiments, e.g., benchmarking hardware and software components
- Important property: reproducibility of experiments

Plain orchestrating system (pos)

- pos is a framework for operating scientific testbeds developed in our research group

Testbeds in Computer Science

Scientific testbeds

- Platforms to implement, debug, and evaluate ideas and concepts
- Execution of experiments, e.g., benchmarking hardware and software components
- Important property: reproducibility of experiments

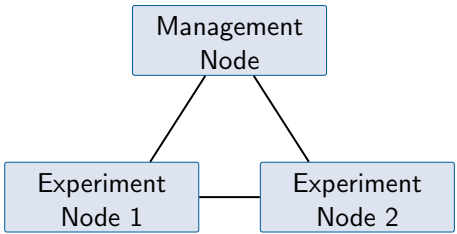
Plain orchestrating system (pos)

- pos is a framework for operating scientific testbeds developed in our research group

Features of pos

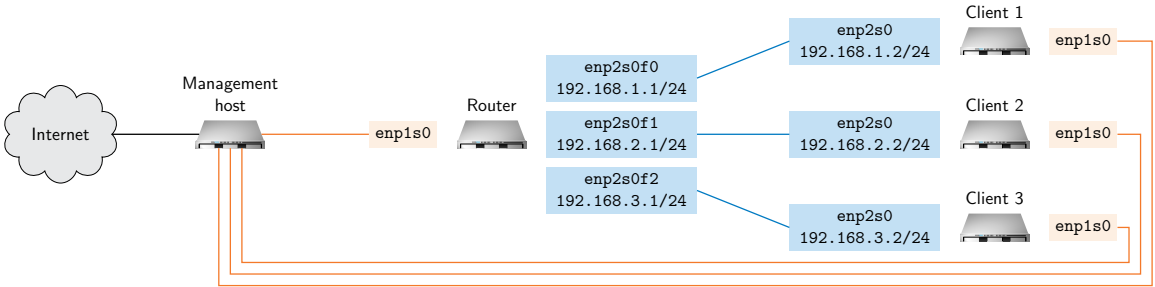
- Automation of experiment workflow
- Live images
  - Experimenters **must** automate configuration
  - No residual state between reboots on experiment nodes
- Other researchers can easily (re-)run experiment

➔ Experiments become reproducible



Minimal experiment topology

Infrastructure for the router project



- Testbed consists of two node types:
  - Management node: Providing SSH and Internet connectivity to experiment nodes
  - Experiment nodes (router and clients): Used for the actual experiment
- Separate management (orange) and experiment (blue) networks
  - Separation ensures measurements that are not impacted by management traffic

Project software router

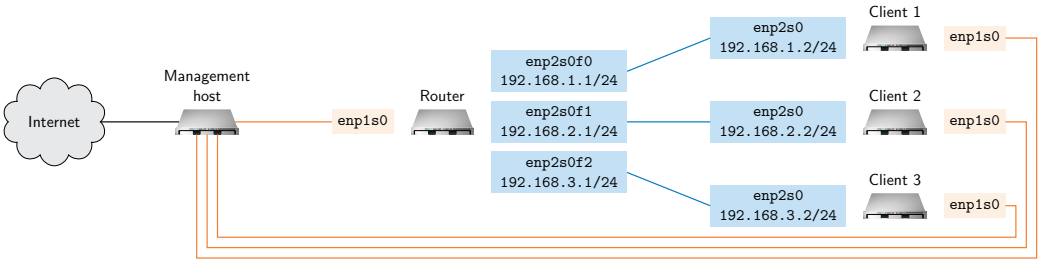
- Implement a software router
- Using the packet processing framework DPDK
- Programming language: C/C++
- You get virtual machines for setting up your router

- Submissions using Git repository (the same repo used for tutorial hand-ins)
- Project deliverables are graded

- Project description available: https://acn.net.cit.tum.de

Problem 1 (1 credits, deadline: November 20, 2025, 4:00 PM)

- Login into your virtual machines
- Configure the testbed setup
- Compile & configure DPDK
- Test your setup with a simple DPDK forwarding example
- Submission: scripts configuring router and clients



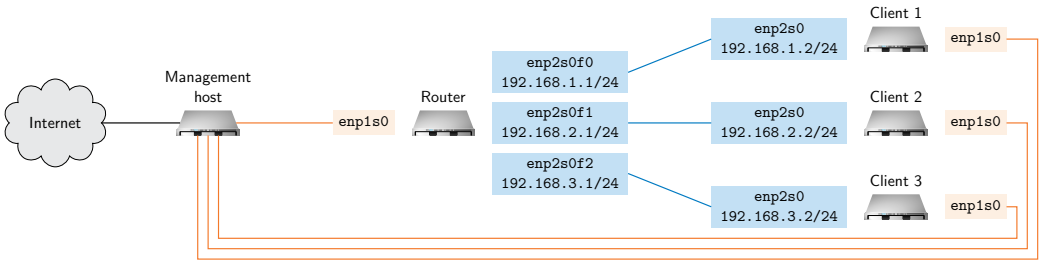
Testbed topology and containing addresses for router and client nodes

1a) Default route

- Your are connected via SSH to a experiment node
- The SSH connection uses the default route
- **Warning:** removing the default route is a **bad idea**

1b) Experiment script

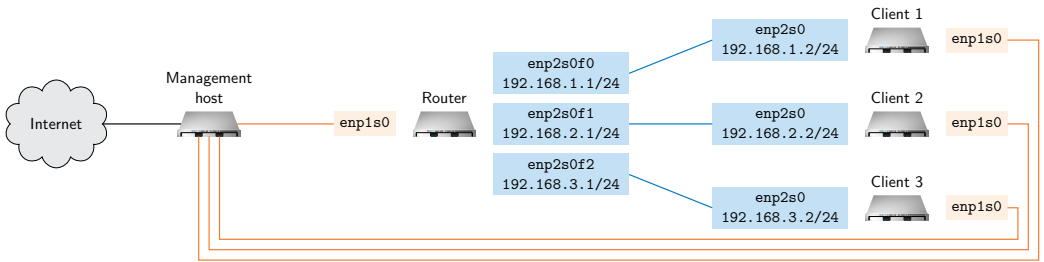
- Nodes are not booted:
  - Allocate nodes
  - Configure image
  - Reboot machines
  - Execute scripts for each node
- **Hint:** Have a look at the pos-examples repo



Testbed setup

1c) Client configuration

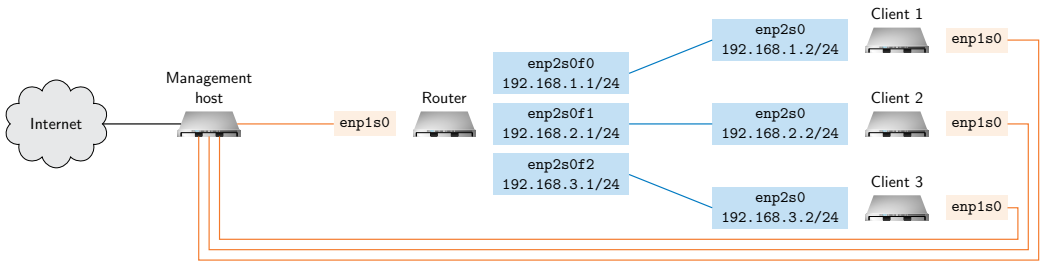
- Nodes are not configured:
  - Regular Linux
  - Config tool to use `ip` (do **NOT** use `ifconfig`)
  - Start `enp1s0` interfaces
  - Set correct addresses
  - Configure routes to other clients



Testbed setup

1d) Router configuration

- Nodes are not configured:
  - Router interfaces controlled by DPDK
  - Regular Linux tools cannot be used for configuration
  - Use the DPDK we provide *(see exercise sheet)*
  - Read the README to compile and install DPDK
  - Try out the forwarding app (`fwd`)



Testbed setup

1e) Test Forwarder

- Configure client nodes
- Run forwarder on router node (forwarding between `enp2s0f0` and `enp2s0f1`)
- Ping Client 2 from Client 1
- Observe packets on Client 2 using `tcpdump`

1f) Bidirectional Forwarder

- The forwarder forwards traffic unidirectionally
- Extend the forwarder to forward in both directions
- Use a second thread

Problem 2 (4 credits, deadline: December 18, 2025, 4:00 PM)

- Command line interface
- Router should answer the clients' ARP requests
- Sanity checks on IP packets
- Do routing decision and forward packets accordingly

Problem 3 (3.5 credits, deadline: January 15, 2026, 4:00 PM)

- Implement a routing table
- Algorithm of choice: DIR-24-8
- Integrate routing table into your software router

Problem 4 (1.5 credits, deadline: January 29, 2026, 4:00 PM)

- Measure performance
- Plot your measurement results
- Create a test report of your findings