Advanced Computer Networking (ACN)

Router Project – Description

Prof. Dr.-Ing. Georg Carle

Sebastian Gallenmüller

Chair of Network Architectures and Services
Department of Informatics
Technical University of Munich

Registration

Projects

- Projects are optional but highly recommended to gain deeper hands-on experience about a specific topic
- We offer two projects this semester:
  - Router project
  - QUIC project
- We have moved to a different infrastructure for the router project this year
- Router project has limited capacity (10 students)
- All students that do not get a slot in the router project may participate in the QUIC project

Application process

- Apply for the router project by writing your name & gitlab user ID to acn@net.in.tum.de
- **No** application is necessary for the QUIC project
- You can find your gitlab user ID at https://gitlab.lrz.de/-/profile
- Application deadline: **Nov 9, 18:00 (CET)**
- You will get a reply by the end of the week if you got a slot in the router-project

Registration

How do you participate?

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

  ```
  git remote add template git@gitlab.lrz.de:acn/terms/2023ws/template.git
  git remote update
  git merge --allow-unrelated-histories template/router-project
  ```
- You are only allowed to participate in one project (either QUIC or router)

How to make clear on which project you are working?

- Merging the `router-project` branch creates the following file: `project.yml`
- We will only consider your submission for the router project iff the file contains only the following line:

  ```
  project: router
  ```
- We use the content of this file to decide which project we correct for a certain deadline
- If you do not follow these instructions, we will not correct and grade your submission

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

Router Project
Userspace packet processing

Known Userspace-Frameworks

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

DPDK

DATA PLANE DEVELOPMENT KIT

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

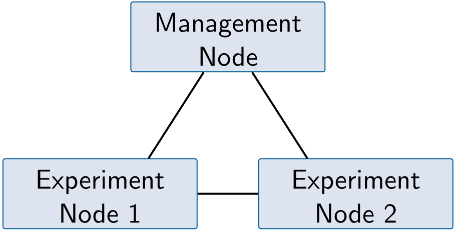
Plain orchestrating system (pos)

- 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

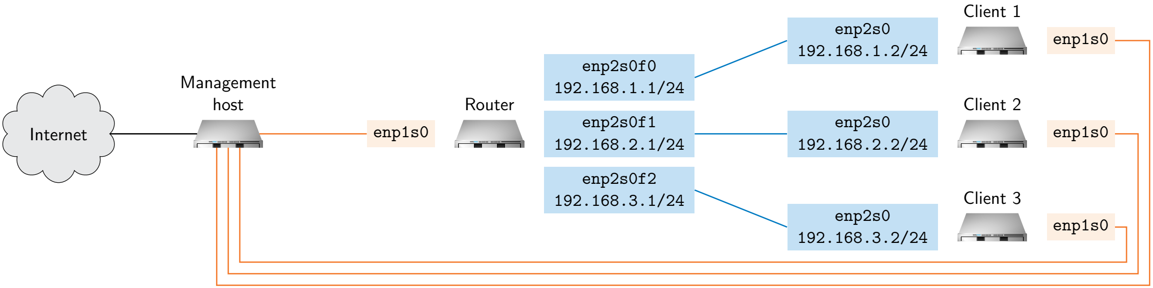
➔ Experiments become reproducible



Minimal experiment topology

Testbeds in Computer Science

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

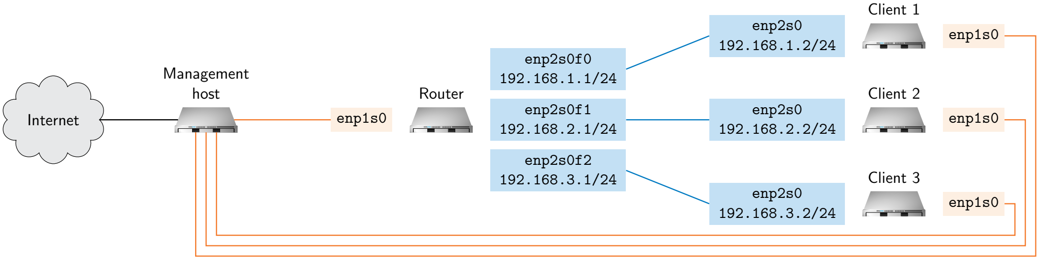
- Submissions using git repository (the same repo used for tutorial hand-ins)
- Project deliverables are graded

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

Testbeds in Computer Science

Problem 1 (1 credits, deadline: November 28, 2023, 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

Testbeds in Computer Science

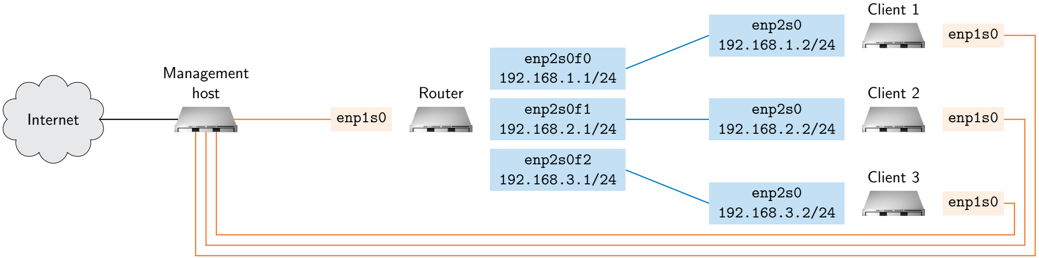
1a) Default route

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

Testbeds in Computer Science

1b) Experiment script

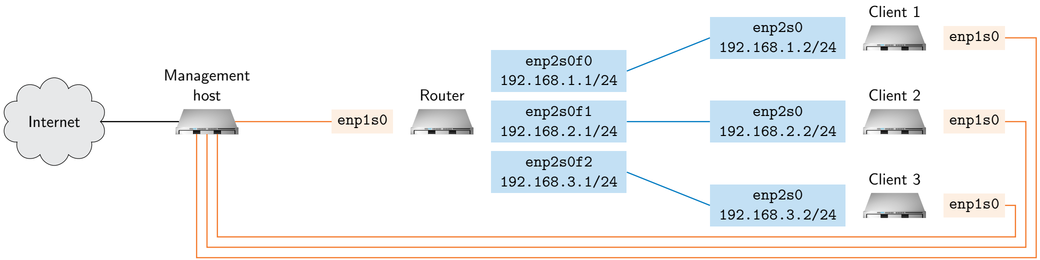
- Nodes are not booted:
  - allocate nodes
  - configure image
  - reboot machines
  - execute scripts for each node
- **Hint:** Have a look at the pos-examples repo



Testbed setup

1c) Client configuration

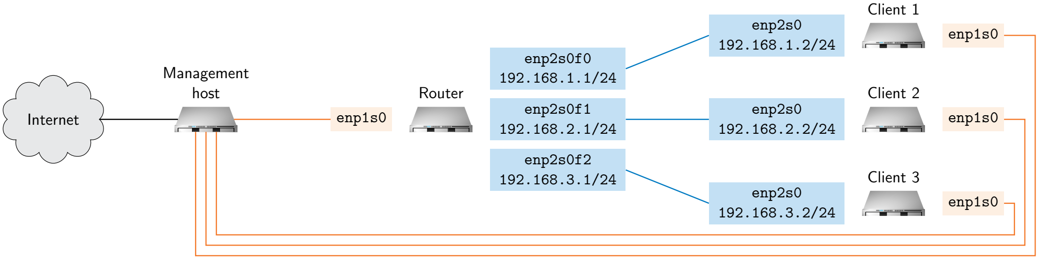
- Nodes are not configured:
  - regular Linux
  - config tool to use `ip` (do **NOT** use `ifconfig`)
  - start `eth1` interfaces
  - set correct addresses
  - configure routes to other clients



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 `eth1` and `eth2`)
- Ping client2 node from client1 node
- Observe packets on client2 using `tcpdump`

1f) Bidirectional Forwarder

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

Testbeds in Computer Science

Problem 2 (4 credits, deadline: December 19, 2023, 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 16, 2024, 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 30, 2024, 4:00 PM)

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