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

Advanced Computer Networking

IN2097, Winter Semester 2025/26

Exercise Procedure

Last updated: October 27, 2025 at 6:07pm

Content of the Exercise

The instructions in this worksheet help prepare the tools necessary to participate in the ACN tutorial. The first tool presented is the version control system Git[1], which is used for the hand-in of your submissions. The second tool is Jupyter, which offers an environment for creating the actual submissions. We provide notebooks, which are processed by Jupyter and can be edited in your browser. There you have to complete problems, such as answering questions, creating short snippets of code, or plotting graphs. Additionally, we provide virtual machines (VMs) used for the exercise and the projects.

Academic Misconduct

We check your submissions for plagiarism. Participants violating the academic code of conduct will be excluded from the bonus system.

The use of AI for replacing your own thought process is not allowed. If you use AI, its use must be clearly disclosed.

It is allowed and encouraged to discuss the assignment with other students. However, the creation of the submission itself has to be done by each student individually. Group work is not allowed. Using Google, StackOverflow, and other Internet sources is allowed (as long as no license is violated). If your submission contains copied information, it has to be clearly marked as such, and the original source has to be referenced. For example, StackOverflow provides a share link. Use that to obtain a link, which must be added to your source code. In any case, you have to understand the code you submitted, which means you must be able to explain how it works.

For details, please refer to the official guidelines of the school and university:

- Official TUM citation guidelines: https://mediatum.ub.tum.de/1723332?show_id=1236069
- Code of conduct of the TUM School of Computation, Information and Technology:
  - English Version: http://go.tum.de/103707
  - German version: http://go.tum.de/750259

[1] see e.g., `http://www.ndpsoftware.com/git-cheatsheet.html`

Getting Access

We provide you with Git repositories hosted at the LRZ GitLab[2]. Every TUM student already has an account. Use your LRZ ID (e.g., ga23wat) to log in if you have never used GitLab before to initialize your account. We have a service to link your student account to your GitLab account. There you authenticate yourself both against TUMonline and the LRZ GitLab. Make sure you registered for the Advanced Computer Networking course in TUMonline. Follow the steps listed below to get access to the course infrastructure:

1. Log in to `https://gitlab.lrz.de` if you have never logged in before and accept the terms and conditions
2. Make sure your SSH keys are uploaded to GitLab (see below for details)
3. Visit `https://acn.net.cit.tum.de/auth` to access the authentication service
4. You are prompted to log in to TUMonline (make sure you are enrolled in the course)
5. Then you are prompted to log in to the LRZ GitLab

Once you followed these steps, you are granted access to several Git repositories.

Repositories

Your personal repository: `[read-write access]`
This is your personal working repository. There you have write access on the `main` branch and you hand in your exercise and project submissions. You repository is **not** automatically cloned onto the testbed nodes.

Material: `[read-only access]`
In this repository, we distribute course materials like lecture slides, exercise slides and solutions, and old exams. The content of the material repository is also available at `https://acn.net.cit.tum.de`.

Template: `[read-only access]`
This repository is used to distribute content for the exercise and projects to you. You are supposed to merge corresponding Git branches into your repository as explained below.

Tools

To access the Git repositories and to SSH into your VM you need to install tools on your local machine. For Unix-based systems there are command line clients for Git and SSH available via your distribution's package manager. For Windows we recommend using the Windows Subsystem for Linux (WSL)[3].

[2] `https://gitlab.lrz.de`
[3] https://en.wikipedia.org/wiki/Windows_Subsystem_for_Linux

Uploading SSH Keys

For pulling or pushing the above repositories, you can authenticate yourself with your password or a public key. Key authentication provides a more secure and convenient way. For example, you can use the following command to create an SSH key pair:

```
your-pc% ssh-keygen -t ed25519 -C "ACN key" -f acn_key
```

Then upload the **public** key (in the above case `acn_key.pub`) to GitLab https://gitlab.lrz.de/-/profile/keys. This enables you to clone the repositories using SSH. Note that you might need to add the generated private key to your default keys or specify it in your SSH config file (`~/.ssh/config`):

```
Host gitlab.lrz.de
    User git
    IdentityFile path/to/your/acn_key
```

Accessing Exercise and Project Materials

We use the **Template** repository mentioned above to distribute the Jupyter notebooks for the exercise and resources for the projects. For this, these materials are pushed on certain branches to the **Template** repository:

`tutorial` for the exercise,

By using the following commands **within your own repository** the resources are merged into your repository and you can work on them. You will have a closer look at them in the first tutorial. For now, just copy them.

```
your-pc% git remote add template git@gitlab.lrz.de:acn/terms/2025ws/template.git
your-pc% git remote update
your-pc% git merge --allow-unrelated-histories template/tutorial
```

For the projects, replace the branch name `tutorial` with the corresponding project branch in the third command.

Accessing the Testbed

For this course, a testbed is used to provide you with virtual machines that you can use for the project and the tutorials.

Scientific testbeds are used to execute experiments, such as benchmarking hardware or software components. We aim to create reproducible experiments. Therefore, we automate the entire experiment workflow to minimize the chance of human error or misconfigurations during experiments. Our research group created a framework to manage testbeds called the *plain orchestrating system (pos)* [1]. pos ensures the creation of reproducible experiments using a specific experiment workflow. For the purpose of this lecture, we created a testbed providing a small network of four connected nodes (1 router/general purpose host, 3 clients) for every student. During the course of this project, you will get to know the pos framework, its experiment workflow, and its features.

Reservation of Testbed Resources

A reservation is needed to use the resources of the testbed. Therefore, we created a calendar to enter the time slots and the type of resources for a planned testbed usage. The calendar can be accessed from the CLI or via the web interface[4]. Multiple identical setups are available for testing, each setup consists of a topology with a single router/general purpose host and three clients as depicted in Figure 1. The router and clients with the same prefix (e.g., `i1`) are directly connected. Each testbed user has access to multiple setups.

[4] https://testbed.acn.net.cit.tum.de/calendar

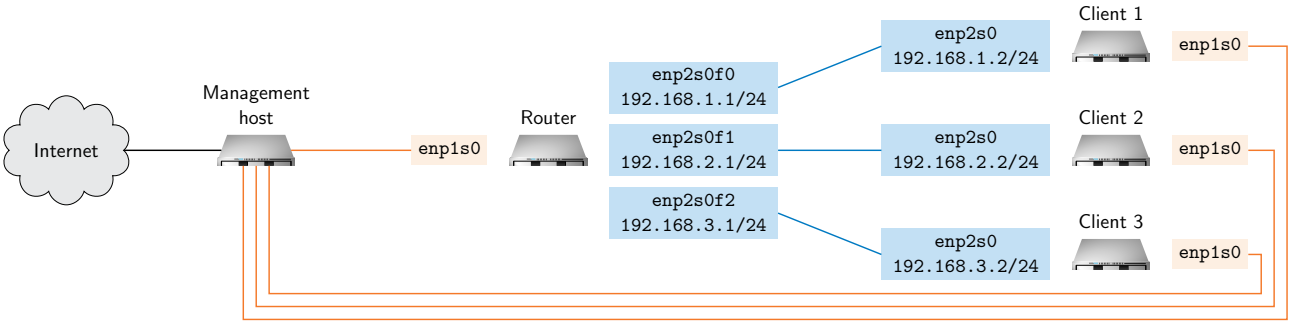

Figure 1: Testbed with management host, clients and router

Fair use policy: Every user can only reserve two future time slots with a maximum length of 6 h. Please only allocate the minimum amount of resources necessary for testing to also give other people a chance to use the testbed.

Accessing the Management Node of the Testbed

The testbed consists of a single management node and multiple experiment nodes. Figure 1 shows the management host and the four experiment nodes used for implementing this project. The management node acts as the gateway to the testbed. All experiment nodes are connected to the management node via a management network (orange connections). This management network is separate from the experiment network (blue connections). This separation is necessary to avoid any impact of management traffic on measurements using the experiment network. The management node of the testbed can be accessed via SSH using the following command in Listing 1.

```
your-pc% ssh -p 10022 sgitlabUserID@testbed.acn.net.cit.tum.de
```

Listing 1: Connecting to the management node (replace `gitlabUserID` with your actual user ID before trying to log in)

Activating your SSH key To be able to log into the management node via SSH, you need to activate your SSH key via the testbed's web interface.

1. Log into the testbed web interface: https://testbed.acn.net.cit.tum.de/
2. Navigate to "User Profile": https://testbed.acn.net.cit.tum.de/profile
3. Click on "Fetch SSH keys from GitLab"
4. Click on "Activate key" for each SSH key you want to activate

Determining the GitLab user ID We use your GitLab user ID prepended with an `s` as the username and the SSH keys you uploaded to the LRZ GitLab for authentication. You can find your GitLab user ID in the web interface (cf. Figure 2). Make sure that at least one of your personal SSH keys known to GitLab is available on your local machine before trying to log in.

Alternatively, you can log into the testbed web interface and find your username under the "Username" setting under "User Profile".

pos CLI After logging in to the testbed host, you can access the pos CLI, the central tool to use the testbed. The pos CLI offers extensive documentation; just type in the pos command. An example can be found in Listing 2.

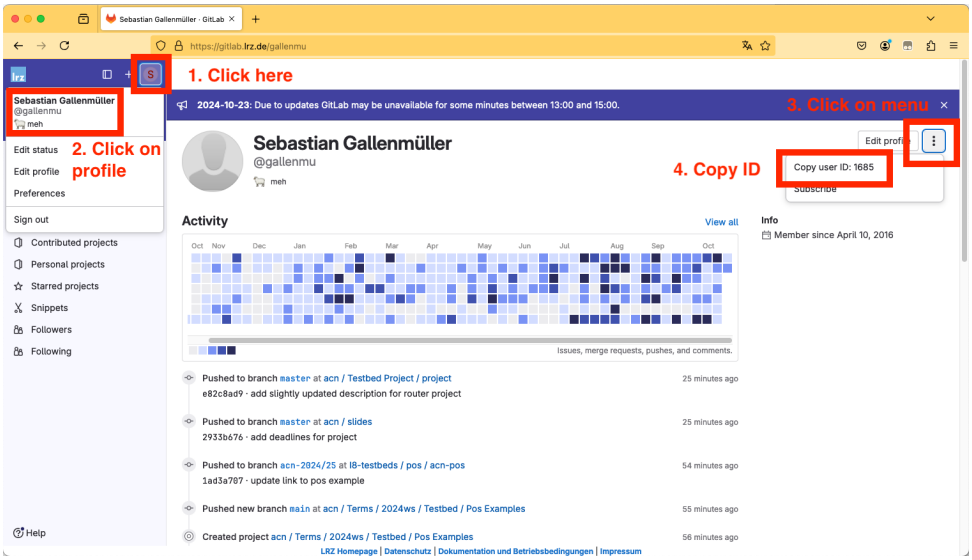

Figure 2: Location of GitLab ID in the web interface

```
testbed.acn.net.cit.tum.de% pos
Usage: pos [OPTIONS] COMMAND [ARGS]...

Plain-orchestrating service to access and manipulate the test nodes of this
testbed.

Quickstart:

0. Look at available nodes:
$ pos nodes list
$ pos allocations list

1. Allocate testnodes you want to use for one experiment:
$ pos allocations allocate nodeA nodeB [...]

2. Configure the nodes (per node):
$ pos nodes image nodeA debian-stretch
$ pos nodes reset nodeA

3. Execute commands on the nodes (per node):
$ pos commands launch nodeA -- echo $(hostname)

You may use command prefixes, e.g. "pos n l" for "pos nodes list"

Options:
--version             Show the version and exit.
--color / --no-color
-h, --help            Show this message and exit.

Commands:
allocations  Define nodes taking part in an experiment.
calendar     Testbed calendar
commands     Execute commands on testbed nodes or roles.
hooks        Configure hooks in posd that can be used as callbacks
images       List available/add new images.
jobs         Jobs (scripts) to be executed by pos at a given time
nodes        Access testbed nodes or roles.
roles        Group nodes into logical experiment roles.
```

Listing 2: pos CLI example on management node with help text output

Accessing the Experiment Nodes

Figure 1 shows the topology offered. A router/general purpose node and three client nodes are available. The router/general purpose node is connected to each client via a separate connection. The router/general purpose node has more RAM (8 GB) and three CPU cores compared to the client nodes, which have 4 GB of RAM and a single virtual CPU core. Every machine has a management interface called `enp1s0` to connect the respective experiment node to the management node. This interface provides SSH and Internet connectivity, i.e., this connection will not be interrupted when messing around with the other interfaces and DPDK.

Participants get their own set of experiment nodes for the project and tutorials. The names of the experiment nodes can be queried using the pos CLI, shown in Listing 3. This command outputs a table that lists the names (`id`s) of all available nodes and additional information, such as the configured image.

```
testbed.acn.net.cit.tum.de% pos nodes list

id          type  status       allocation                 image           updated

i0-client1  host  ERR booting  1685_231103_210856_811477  debian-trixie   48h
i0-client2  host  booted       1685_231103_213724_568848  debian-trixie   48h
i0-client3  host  unknown      None                       boot-local      6d
i0-router0  host  unknown      None                       boot-local      6d
```

Listing 3: pos command to list the available nodes of the testbed

To log in to a specific node, use the SSH command on the management node followed by the ID of the respective experiment node. Keep in mind that the node has to be booted before it can be accessed. Listing 4 shows several nodes. According to the printed output, only the experiment node with the ID `i0-client2` is currently in a booted state and can be accessed via SSH.

```
testbed.acn.net.cit.tum.de% ssh i0-client2
```

Listing 4: Logging in to a booted experiment node

To set up the nodes that are currently in a non-booted state, several steps have to be performed:

1. Allocate the respective node(s)
2. Configure an image to be booted
3. Reset the machines to load the configured image
4. Wait for the machine to become available

Listing 5 lists the respective commands to perform the previously listed steps. We also provide a more extensive example script at `git@gitlab.lrz.de:acn/terms/2025ws/testbed/pos-examples.git`.

```
testbed.acn.net.cit.tum.de% pos allocations allocate i0-client1
Allocation ID: s27161_251021_151240_258728 (i0-client1)
Results in /srv/testbed/results/s27161/default/2025-10-21_15-12-40_258728
testbed.acn.net.cit.tum.de% pos nodes image i0-client1 debian-trixie
testbed.acn.net.cit.tum.de% pos nodes reset i0-client1
```

Listing 5: Allocating and preparing an experiment node with id `i0-client1`

Rebooting Experiment Nodes

Nodes can be rebooted from the management node using the command in Listing 6. This also works if the connection between the experiment node and the management node is no longer available, e.g., if the NIC driver was removed.
Important note: All experiment nodes use RAM disks as storage; **resetting a node erases everything that was written to this storage**. Only your home folder on the management node is permanent.

```
testbed.acn.net.cit.tum.de% pos nodes reset i0-client1
```

Listing 6: pos command to reboot a node with id `i0-client1`

Using Jupyter Notebook

The exercises use Jupyter notebooks, which can simply be accessed via your browser. We provide empty notebooks in the **Template** repository (see *Accessing Exercise and Project Materials*). You can run a Jupyter server locally or use our prepared images. The following section provides you a detailed step-by-step guide on how to use it.

Configure SSH on Your Machine

In order to access the Jupyter notebooks, you need to start a Jupyter notebook server on an experiment node, and tunnel a local port (e.g., 1337) to the appropriate port on the experiment node (1337). When you use the correct testbed image (`debian-trixie-acn-tutorial`), a Jupyter notebook server is automatically started using a systemd service (`jupyter.service`) on port 1337.

As there is no direct connection between your PC and the chosen experiment node, you have to use the management host as an SSH proxy jump host. For this, you can use the following SSH sample configuration. Replace the values in bold print with the appropriate values of your user and setup.

```
# Configuration to access the management host
Host testbed.acn.net.cit.tum.de
       Hostname testbed.acn.net.cit.tum.de
       User sgitlabUserID
       IdentityFile path/to/your/acn_key
       Port 10022

# Configuration to access a particular experiment node
Host *.testbed
       HostName %h.acn.net.cit.tum.de
       User root
       IdentityFile path/to/your/acn_key
       ProxyJump testbed.acn.net.cit.tum.de
       StrictHostKeyChecking no
       UserKnownHostsFile /dev/null
```

Listing 7: Exemplary SSH configuration file to access experiment nodes (usually located at `~/.ssh/config`)

Start and Access the Jupyter Notebook

In order to start the Jupyter notebook server on a VM, follow these steps, replacing bold values suitably:

1. Log into the management node

   ```
   your-pc% ssh testbed.acn.net.cit.tum.de
   ```

2. Reserve an appropriate node (e.g., **`i8-router0`**) for a specified duration (here **120** minutes)

   ```
   testbed.acn.net.cit.tum.de% pos calendar create --start now -d 120 i8-router0
   ```

3. Allocate the node

   ```
   testbed.acn.net.cit.tum.de% pos allocations allocate i8-router0
   ```

   In case the allocation fails, as the previous user has not freed their allocation, execute the following command

```
testbed.acn.net.cit.tum.de% pos allocations free -k i8-router0
        # -k ensures that your reservation is not deleted
```

4. Configure the image to use (`debian-trixie-acn-tutorial` contains the Jupyter notebook server)

   ```
   testbed.acn.net.cit.tum.de% pos nodes image i8-router0 debian-trixie-acn-tutorial
   ```

5. Reboot your node

   ```
   testbed.acn.net.cit.tum.de% pos nodes reset i8-router0
   ```

6. Log onto the node and clone your personal repository

   ```
   testbed.acn.net.cit.tum.de% ssh i8-router0
   i8-router0% git clone git@gitlab.lrz.de:acn/terms/2025ws/students/sgitlabUserID.git
   ```

   Please note that this works because the testbed management node adds an SSH deploy key to a forwarded SSH agent. If you connect to the testbed node via the jump host (in the SSH config), this will not work, since the SSH agent is not forwarded.

7. Create an SSH tunnel to your VM (using the provided SSH config)

   ```
   your-pc% ssh -L 1337:127.0.0.1:1337 i8-router0.testbed
   ```

8. In your browser of choice visit `http://localhost:1337/`

9. Commit your changes using the Git CLI

   Remember: The VMs are ephemeral. Therefore, when they are shut down, which occurs after 12 hours of inactivity, **all local data is lost**. Commit and push your work, or use some other non-ephemeral storage to prevent data loss.

10. Free your allocation

    ```
    testbed.acn.net.cit.tum.de% pos allocations free i8-router0
    ```

References

[1] Sebastian Gallenmüller, Dominik Scholz, Henning Stubbe, and Georg Carle. The pos Framework: A Methodology and Toolchain for Reproducible Network Experiments. In *CoNEXT '21: The 17th International Conference on emerging Networking EXperiments and Technologies, Virtual Event, Munich, Germany, December 7 - 10, 2021*, pages 259–266. ACM, 2021.