Advanced Computer Networks 2022
Vrije Universiteit Amsterdam
Lin Wang, George Karlos, Florian Gerlinghoff

Lab0: Welcome and Warm-up

Type:	Individual
Maximum points:	0
Submission:	None
Deadline:	None
Contact:	vu.acn.ta@gmail.com

The goal of this lab (not graded) is to introduce you to the logistics of the course labs and get you started on the basics of Mininet—a network emulator we will be using for most of the other labs throughout the course. Please make sure you read this document carefully.

1 Code of Conduct

We have **zero tolerance** to cheating, fraud, and plagiarism.

You are encouraged to discuss specific issues and share knowledge with peer students (e.g., on Canvas Discussion forum). However, in any circumstances you are expected to write your own code. Do not buy or sell code. Do not copy code from anywhere. Do not give out code to other students. It is unlikely that the labs will result in two students/groups writing almost identical code. Any similarities in the code will lead to further investigation, with a possibility of reporting to the examination board.

2 Labs Organization

2.1 Logistics

The course includes in total **five graded labs** (i.e., lab1 through lab5). All of them will involve coding with Python and/or C/C++. If you are not familiar with these programming languages, please find online resources to prepare yourself before you work on the labs. Advanced knowledge of these programming languages is not required, so you can just follow online tutorials (e.g., https://docs.python.org/3/tutorial/ for Python) to learn the basics. After all, this is a course about networking, not about advanced use of programming languages.

All labs except lab1 will be based on group efforts where you can form **a group of up to three students** to work on these labs. Having fewer students in a group is allowed. You should start looking for your group mates as soon as the course starts. Once you have found group mates, you can enroll yourself in a lab group on Canvas (People → Lab Group). You should have joined a group by the deadline which will be announced in the first lecture and on Canvas. Lab1 will be assessed individually, based on a Canvas quiz. With this quiz, you can roughly assess your and your group mates' capabilities before forming groups.

If your group, unfortunately, dissolves in the middle of the course (e.g., other members have dropped the course), you could choose to join another group given that you can find a group willing to accept you and the group is not full yet. You can also continue the rest labs on your own if you cannot find another group to join. You should inform us with such group changes in

advance. Once we approve the change, you should move yourself to the new group on Canvas. You are not allowed to switch the group without a legitimate reason.

For each lab in which you are expected to submit your code, please make sure you follow strictly all the guidelines we provide you. Failing to follow the guidelines may lead to penalties or even direct rejections to your submission with zero points. Together with your code submission, you are expected to write a small report documenting your ideas for the implementation and observations from your experimental results. We will provide templates for the reports and you are expected to follow the templates closely. The report templates will be based on LaTeX. If you have never used LaTeX before, please check Overleaf—an online tool for working with LaTeX. Overleaf also provides a nice LaTeX tutorial in case you need to learn the basics. You have access to Overleaf Professional through your UvA account (see https://nl.overleaf.com/edu/uva).

2.2 How to Ask for Help

If you are stuck and you are looking for help, please follow these steps.

- Read the code given to you carefully and check the online documentations for the tools (e.g., Mininet usage, Ryu APIs, P4 language specifications). For debugging, please look at the output and error logs carefully. Most of the time, your problem would be solved if you spend enough time doing the above.
- Ask Google. Issues with network setups and API usage could be solved with proper Googling most of the time. Make sure you have done sufficient search. Also consider searching the GitHub issues of the project.
- Ask in the Discussion forum on Canvas. If the above two steps have not solved your problem, you can post your problem in the Discussion forum on the course Canvas. However, you should never ask for a direct solution to a specific lab. Try to narrow down your problem and ask that specific aspect only. It is possible that other students have already posted the same problem before you, so make sure you have gone through all the problems discussed there and your problem has not been covered before you create a new post. We will attend to the Discussion forum periodically to answer questions. However, you should be prepared to hear that "debugging such an issue is part of the lab assignment". We encourage you to constructively participate in Canvas discussions. Please respect each other and do not give out the solution to the lab directly.
- Contact us through email. After exhausting all other possibilities, if you believe you are really stuck and there is something fundamentally wrong, please reach out to us via the email vu.acn.ta@gmail.com. However, do not just paste the print-screen and say that something does not work. Please show us you have put in the effort (e.g., the above steps) to solve the problem. Please include details like the ones below in your email to us.
  - What is the problem?
  - How to reproduce it? Is it nondeterministic or does it happen every time?
  - What possibilities have you tried to narrow down the error conditions?
  - Have you searched on Google the error message? Did you find anything useful?
  - What do you suspect is happening?

2.3 Slip Days

You have in total **three slip days** which you can use to extend the deadline of lab submissions. You can split these slip days for multiple deadline extensions, e.g., two days for lab2 and one day for lab5. There will be no further extensions after that and you will be awarded zero points

if you miss the deadline. In case of a group reorganization, the individual slip days will be counted and divided by the members of the group and rounded up. For example, suppose a three-member group has {1, 1, 3} slip days left among the members, the whole group will have (1+1+3) / 3 = 2 days worth of slip days.

Exceptions: We will only consider exceptions for medical situations (e.g., COVID-19 related interruptions). However, you must inform your study advisor about such situations and obtain their approval. Without the approval from the study advisor, we will not be able to grant any unilateral deadline extensions.

2.4 How to Work in a Group

The lab deadlines are spread out based on the perceived complexity and the amount of code that needs to be written. If you show up one day before the deadline saying that your group members decided not to work or dropped the course, we cannot help you in such a short notice. You should inform us early enough to find a solution.

All of you must be responsible for the group work. Here are some good practices to follow:

- Schedule one or more weekly meetings to discuss the lab. If one or more members do not show up after multiple reminders, inform us immediately and notify them that they are not part of your group any more.
- Meet early and decide in advance who does what and when to synchronize the progress. Use a version control system like git to keep track of issues and updates.
- We will not be responsible for resolving disputes in your group. If you have trouble in your group, contact us early enough with detailed evidence of what happened.

3 Set Up the Course Virtual Machine

The first step is to set up the virtual machine (VM) you will be using for the rest of this course. This will make it easy to install all dependencies for the labs, saving you the tedium of installing individual packages and ensuring your development environment is correct.

3.1 Install VirtualBox

VirtualBox is a VM provisioner (hypervisor). For macOS or Windows users, please visit this page https://www.virtualbox.org/wiki/Downloads. The download links are under the heading "VirtualBox 7.x.x platform packages". For Windows users, use all the default installation settings, but you can uncheck the "Start Oracle VirtualBox 7.x.x after installation" checkbox. For Linux users, you can install VirtualBox by running the command

```
sudo apt install virtualbox
```

This will install the VirtualBox application with GUI on your computer.

Note: We have not tested the VM setup on Windows Subsystem for Linux or M1/M2-based Macs. Please try to avoid such platforms or make it work on your own. In case of issues on these platforms, we will not be able to provide help.

3.2 Import the Course VM Image in VirtualBox

Please download the course VM image at

```
https://surfdrive.surf.nl/files/index.php/s/MPz4Idpic30qmbC
```

The image is around 5.9GB in size (MD5 checksum: `41db6b39f269796d5142b9f0bca0acc1`). Please make sure you have a good Internet connection before you download it.

After that, please follow the steps in the VirtualBox User Manual to import the VM. Please keep most of the configurations as default, but you can adjust the number of processors and amount of memory allocated to the VM based on the available resources of your host machine. Once the import is finished, you can boot up the VM. The user name for VM login is `acn22` and the password is `@msterdam2022`.

3.3 Download the Course Labs Code Base

The course labs code base is hosted on GitHub at https://github.com/vuhpdc/acn22-labs. Please use the following command to clone the project repository to your VM

```
git clone https://github.com/vuhpdc/acn22-labs.git acn22-labs
```

You can also fork the repository to your own GitHub account and collaborate with your group members using the forked repository. If you have never worked with git before, please check the slides about version control and git on Canvas at

```
Files/references/tutorial-git.pdf
```

In the code base repository, labs are organized in different folders. Lab0 is already available under folder `lab0`. When we release a new lab, this code base repository will be updated with a new folder (e.g., `lab1`) for the new lab. You can simply run

```
git pull
```

to retrieve the new lab. You should follow the prompt and simply apply a merge with your local repository.

3.4 Advanced Setup

For those of you who would like to work from your host machine (with your favorite editor), you can set up a shared folder between the host and the VM and use SSH to access the VM from the host. This section provides the instructions for that. If you plan to access your VM through the VirtualBox GUI and do the development directly inside the VM (you will need to install your favorite editor for that), then you can skip this section.

3.4.1 Development on the Host Machine

To do the development on your host machine and have changes visible to your VM, the easiest way is to share the `acn22-labs` directory between your host and the VM:

1. Open up a terminal (on Linux or macOS) and, first, shut down the VM (if running)

```
VBoxManage controlvm acn22 poweroff
```

For Windows, you can run `VboxManage` from the CLI (e.g., Command Prompt) with

```
C:\Program Files\Oracle\VirtualBox\VBoxManage.exe
```

2. Share the folder with the VM

Warning: The command below will wipe out the directory (i.e., `<guest_path>`) in the VM. Please make sure you have backed up that directory before you run this command.

```
VBoxManage sharedfolder add acn22 --name shared-acn2-dir --automount \
--hostpath <host_path> --auto-mount-point <guest_path>
```

where you replace

- `<host_path>` with the path of the `acn22-labs` directory on your host machine
- `<guest_path>` with the desired path of `acn22-labs` in the VM

3. Verify with

```
VBoxManage showvminfo acn22 | grep "Host path"
```

3.4.2 Remote Access to the VM

To access your VM remotely you have to establish an SSH connection to it from the host:

1. Open up a terminal and, first, shut down the VM (if running)

```
VBoxManage controlvm acn22 poweroff
```

2. By default we have already set up a port forwarding rule which maps the port 22 of the VM (for SSH) to port 4242 of the host. In case port 4242 is already occupied or you want to use a different port number (e.g., 4243), you can set up port forwarding with

```
VBoxManage modifyvm acn22 -natpf1 "ssh,tcp,,4243,,22"
```

You can now access the VM with SSH. However, some of the assignments require launching a GUI from the VM. To do this remotely, you need an X server installed on your host machine. If you are using Linux as your host OS you already have an X server installed (good job!). If you are using Windows or macOS, **before continuing with the next steps, make sure that**

- **macOS**: XQuartz is installed. Follow the XQuartz website for instructions.
- **Windows**: Xming is installed. Follow the Xming website for instructions. Use the default options and uncheck "Launch Xming" at the end

Your VM is already configured to use the X server. Thus, the following steps should work. If not, then please open a thread on Canvas or contact us.

3. Start the VM in headless (no GUI, not needed) mode

```
VBoxManage startvm acn22 --type headless
```

4. Connect to it with SSH

```
ssh -X -p 4242 acn22@127.0.0.1
```

5. Test the X forwarding with

```
sudo xterm
```

4 Mininet 101

Mininet is a powerful network emulator and we will use it throughout the labs. Mininet is pre-installed in the VM you have just set up. Please complete the online Mininet walkthrough at http://mininet.org/walkthrough/. In some parts of the walkthrough, you may see software-defined networks. We will discuss it later in this course, so you do not need to understand what they mean right now. You just need to know how to run and interact with Mininet.

4.1 Build a Customized Network Topology

Now using the Mininet APIs you have just learnt, build the network topology illustrated in Figure 1. Hosts `h1 - h4` are represented by squares and switches `s1` and `s2` are represented by circles. The name of the devices you build in Mininet should match the names in the diagram. The hosts are assigned IP addresses `10.0.0.x` where `x` should match the host number $(1-4)$, e.g., `10.0.0.2` for `h2`. The properties (bandwidth, delay) of the links are specified as: (15Mbps, 10ms) for links `e1 - e4` and (20Mbps, 45ms) for link `e5`.

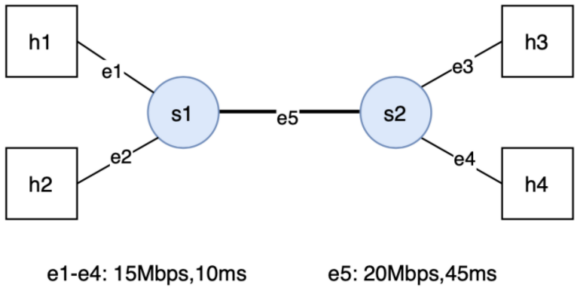

Figure 1: A simple bridge network topology.

One you have completed the topology file, launch it in Mininet using the following command.

```
sudo mn -custom /path/to/topo/file -topo=topo name -link=tc
```

4.2 Measure the Performance of the Network

Use Xterm to start terminals for the hosts in the network.

```
mininet> xterm h1 h3
```

Now, please measure the latency and throughput between hosts `h1` and `h3` using `ping` and `iperf`, respectively. For ping you should send 20 or more packets and average the results and for `iperf`, you should measure for 20s or longer. Measure with both TCP and UDP connections.

```
h1$ iperf -s (-u)
h3$ iperf -c ip.to.server -t 20 (-u)
```

4.3 Effect of Multiplexing

In the above measurement we only have one connection going at the same time. Now, let us try with simultaneous connections where we let `h1` talk to `h3` and `h2` talk to `h4` at the same time. Try to predict the latency and throughput. Use ping and iperf to measure the latency and throughput of the two connections. Check if your predictions are correct and think about why or why not. What if we have two connections all destined to `h4` (i.e., `h1 - h4` and `h2 - h4`)?