C A U

Kiel University
Christian-Albrechts-Universität zu Kiel

Research Group
Distributed Systems

Distributed Systems Course
Labs Presentation

What you will do during the labs

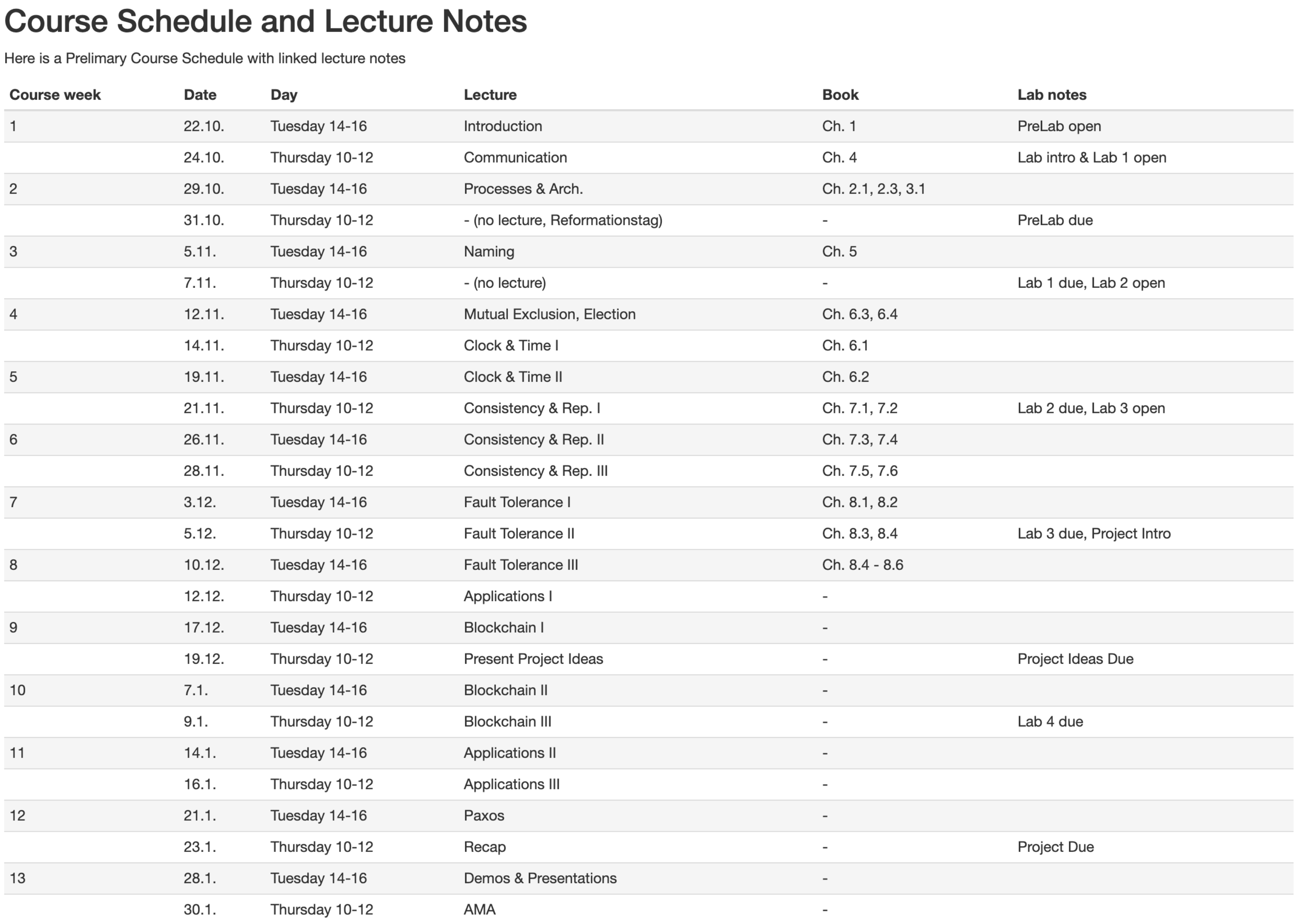
What, how, when

- Every Thursdays
  - From 14:15 to 15:45

- We use iLearn
  - Does everybody have access to it?

- Groups of 2

Timeplan (see iLearn for up-to-date infos)



Course Schedule and Lecture Notes

Here is a Prelimary Course Schedule with linked lecture notes

Course week	Date	Day	Lecture	Book	Lab notes
1	22.10.	Tuesday 14-16	Introduction	Ch. 1	PreLab open
	24.10.	Thursday 10-12	Communication	Ch. 4	Lab intro & Lab 1 open
2	29.10.	Tuesday 14-16	Processes & Arch.	Ch. 2.1, 2.3, 3.1	
	31.10.	Thursday 10-12	- (no lecture, Reformationstag)	-	PreLab due
3	5.11.	Tuesday 14-16	Naming	Ch. 5	
	7.11.	Thursday 10-12	- (no lecture)	-	Lab 1 due, Lab 2 open
4	12.11.	Tuesday 14-16	Mutual Exclusion, Election	Ch. 6.3, 6.4	
	14.11.	Thursday 10-12	Clock & Time I	Ch. 6.1	
5	19.11.	Tuesday 14-16	Clock & Time II	Ch. 6.2	
	21.11.	Thursday 10-12	Consistency & Rep. I	Ch. 7.1, 7.2	Lab 2 due, Lab 3 open
6	26.11.	Tuesday 14-16	Consistency & Rep. II	Ch. 7.3, 7.4	
	28.11.	Thursday 10-12	Consistency & Rep. III	Ch. 7.5, 7.6	
7	3.12.	Tuesday 14-16	Fault Tolerance I	Ch. 8.1, 8.2	
	5.12.	Thursday 10-12	Fault Tolerance II	Ch. 8.3, 8.4	Lab 3 due, Project Intro
8	10.12.	Tuesday 14-16	Fault Tolerance III	Ch. 8.4 - 8.6	
	12.12.	Thursday 10-12	Applications I	-	
9	17.12.	Tuesday 14-16	Blockchain I	-	
	19.12.	Thursday 10-12	Present Project Ideas	-	Project Ideas Due
10	7.1.	Tuesday 14-16	Blockchain II	-	
	9.1.	Thursday 10-12	Blockchain III	-	Lab 4 due
11	14.1.	Tuesday 14-16	Applications II	-	
	16.1.	Thursday 10-12	Applications III	-	
12	21.1.	Tuesday 14-16	Paxos	-	
	23.1.	Thursday 10-12	Recap	-	Project Due
13	28.1.	Tuesday 14-16	Demos & Presentations	-	
	30.1.	Thursday 10-12	AMA	-	

Exercise sessions

- When a lab or projects are due / presented
  - Attendance is mandatory!
    - You need a good reason for not joining these sessions
- The other sessions
  - Are for Q&A, feedback, etc.
  - But
    - If you do not join, do not complain if you miss out on discussions
- How to contact me?
  - During the exercise sessions ← **Best**
  - Otherwise per mail, but don't expect a quick answer!
  - Don't knock on my door! I will not answer your questions

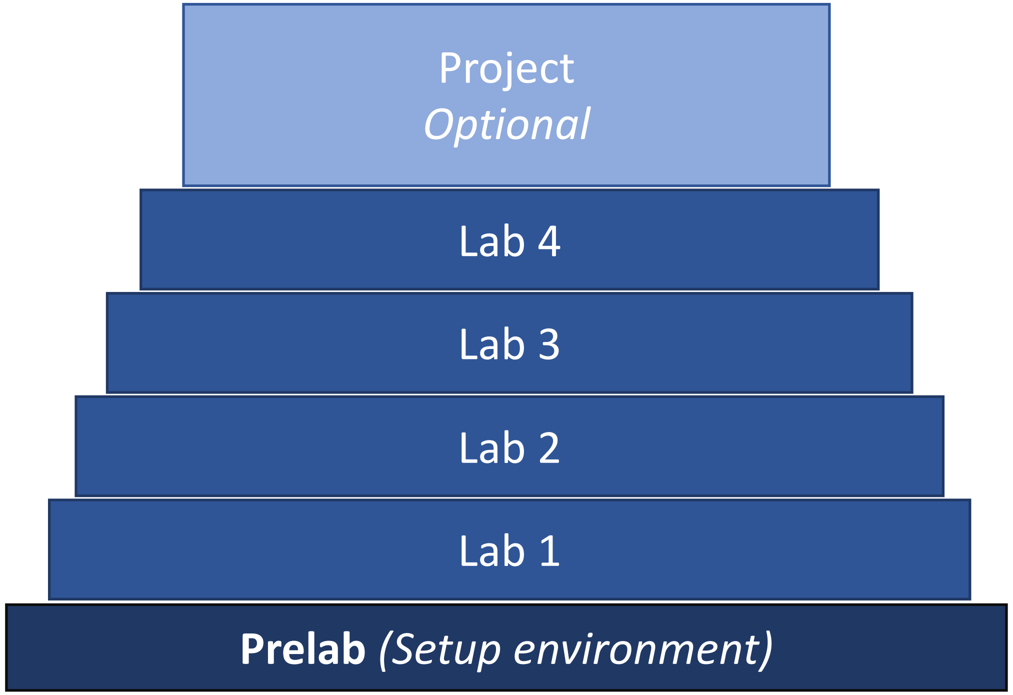
Example:

- 7th Nov. is presentation day
→ attendance mandatory

- 14th Nov is Q&A
→ Not mandatory

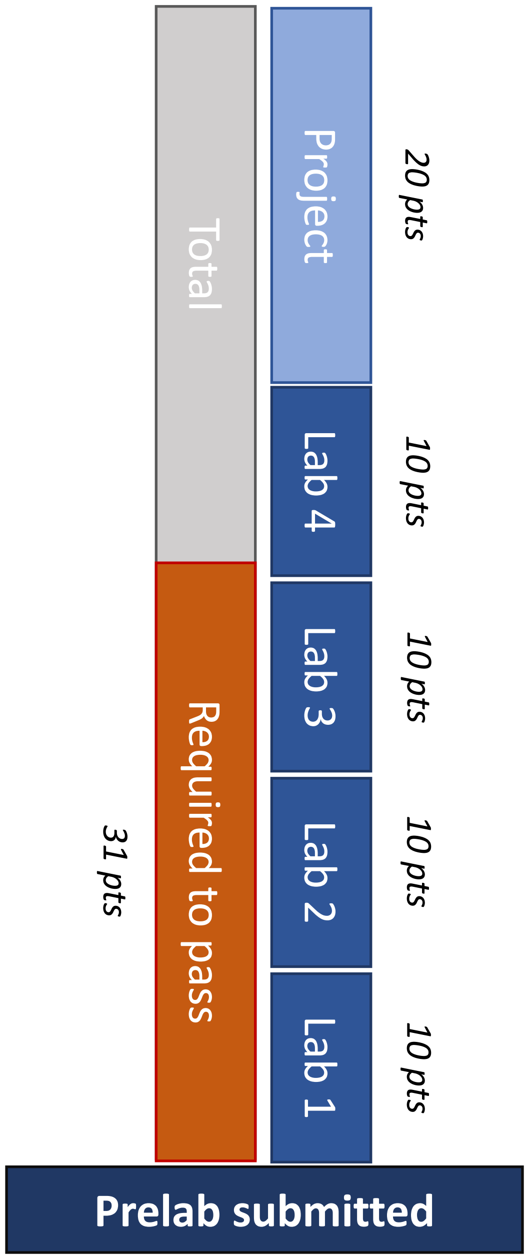
In a nutshell

- Prelab
  - Learn the basics of Python, setup your environment
- Labs
  - Four labs, build a distributed system, and improve it everytime
- Project *(optional!)*
  - You propose the topic
  - Implement your nice idea
  - Present it to the other groups at the end of the course!



How to pass the labs?

- Submitting Prelab is mandatory
- You need 50% of the points
  - One lab is 10 points
    - Sometimes with bonus points
  - The optional project is worth 20 points
  - You get more than 30 points, you pass!

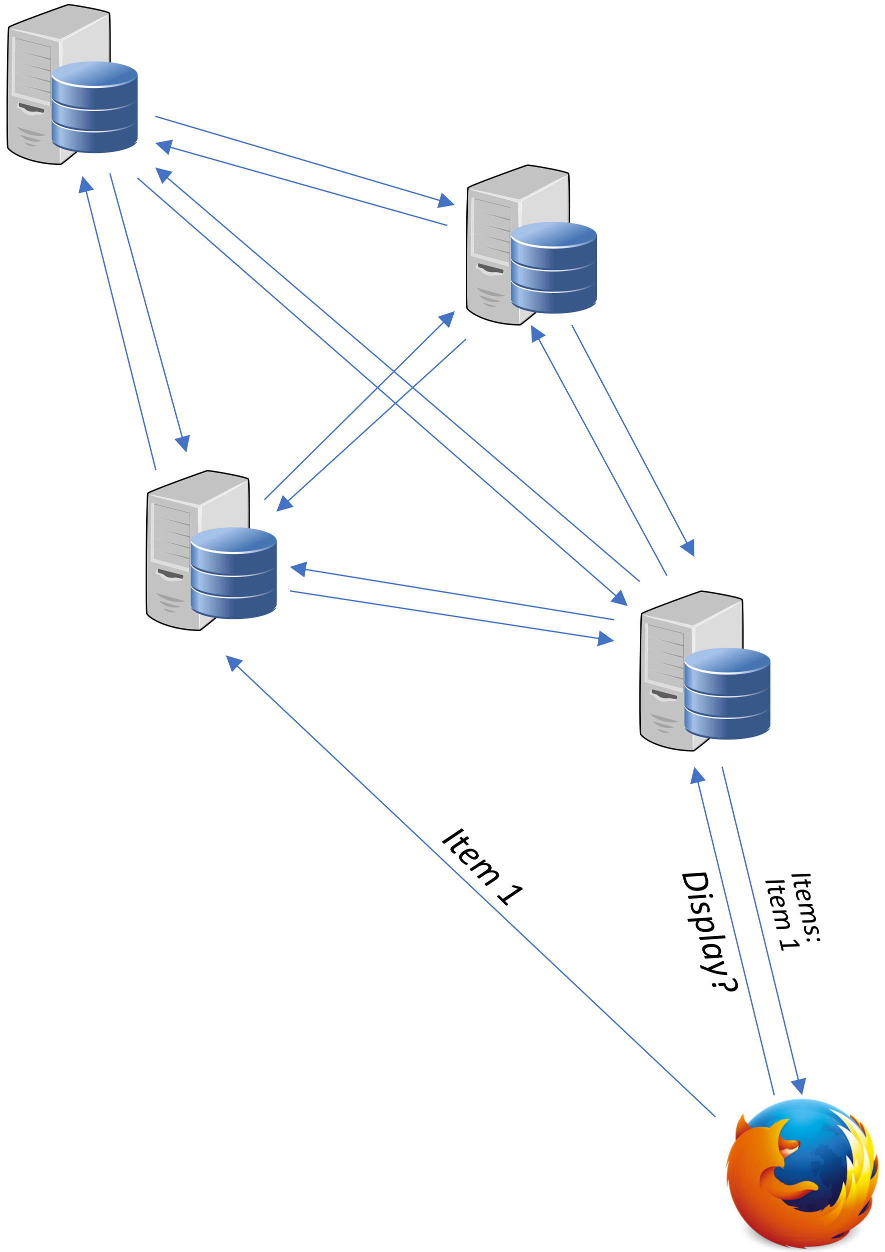


In a nutshell – the labs

- Design and implement a distributed system

- **RESTful distributed blackboard**
  - Clients send notes/messages to any server
  - Servers do distributed systems magic to provide a reliable, consistent, efficient, fault-tolerant service
  - Clients from all other the world should see the same result on all servers!

- In short, we apply and implement the concepts you'll see in the lectures

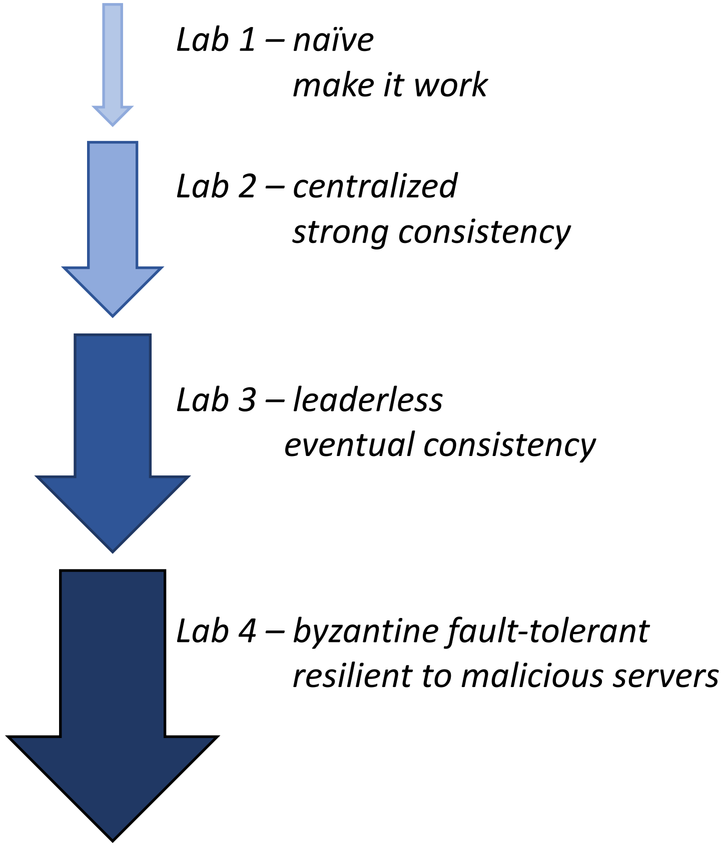


How we will do it

- Incremental steps

- Explore different design choices

- Make your code base grow
  - Learn your lessons on the way



Lab 1 – naïve
make it work

Lab 2 – centralized
strong consistency

Lab 3 – leaderless
eventual consistency

Lab 4 – byzantine fault-tolerant
resilient to malicious servers

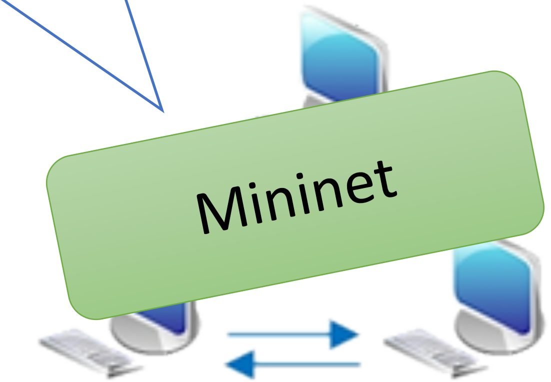
In a nutshell – the labs

Lab overview

During the labs you will make this application:

- Reliable
- Consistent
- Efficient
- Fault-tolerant
- …

You will build an application over this network



Mininet

You will use a platform for emulating computers around the world

Mininet

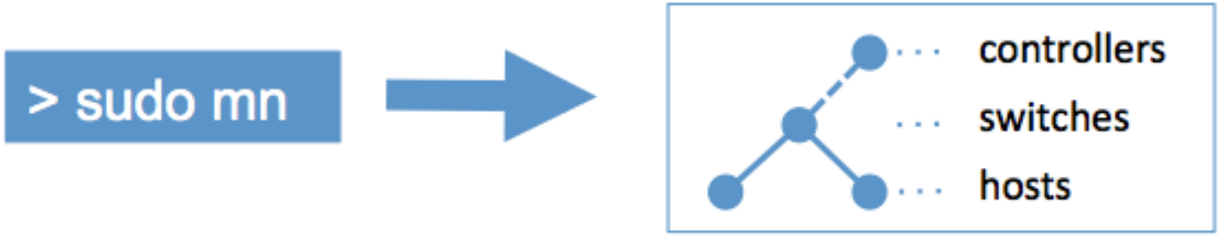
The platform

Mininet

- Network emulator
  - Emulate hosts (machines), switches, controllers, and network links
  - All of that, on one PC!

- Often used in research
  - Especially in computer networking

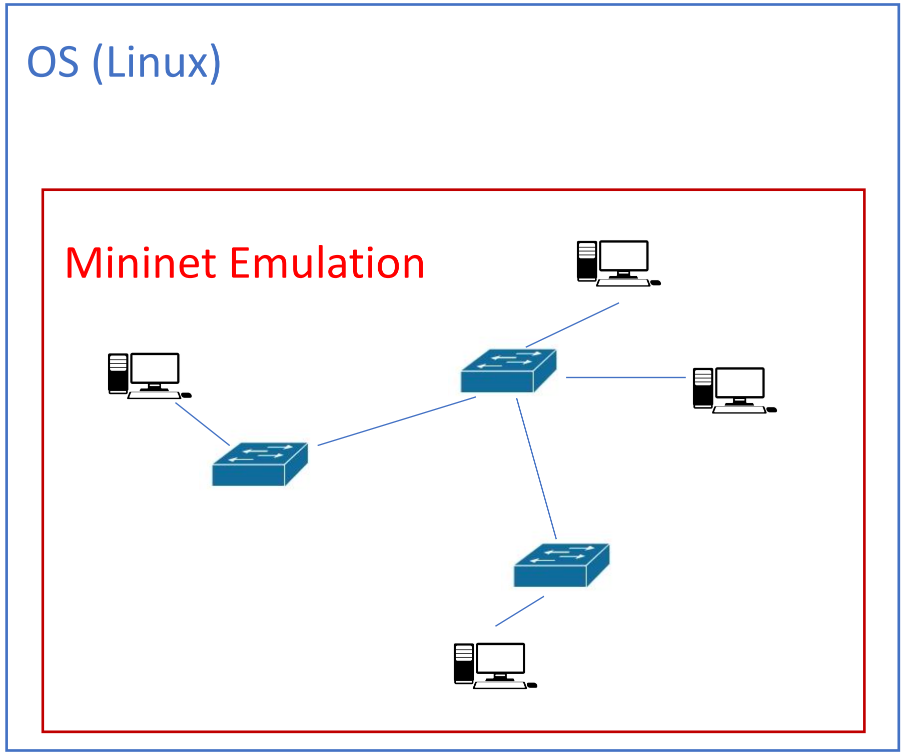
- Can handle large scale networks
  - If your machine is powerful enough



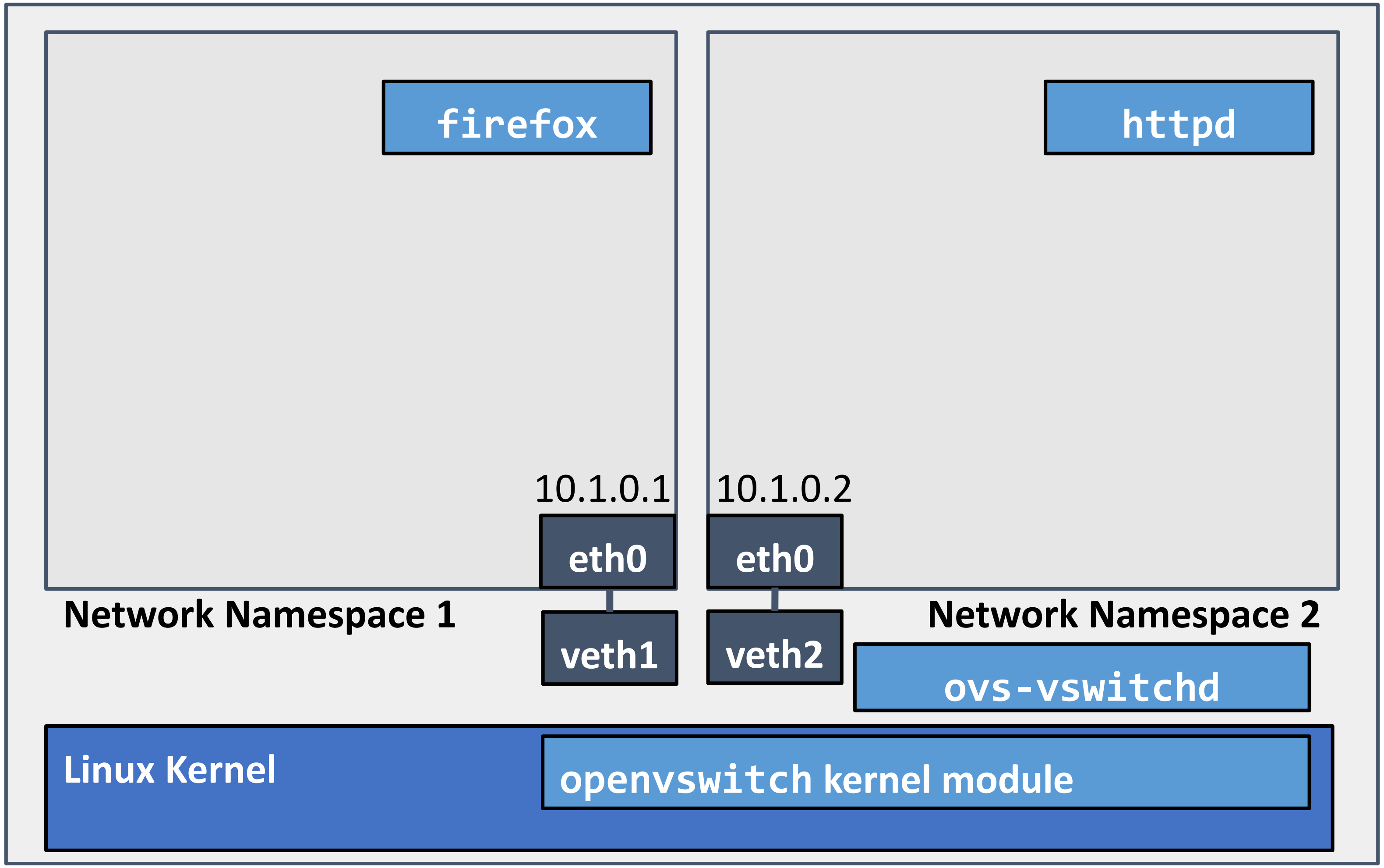
```
> sudo mn
```

Mininet: How does it work?

- Container-based virtualization
- Nodes are emulated as a UNIX process
  - Cannot access any other processes locally
    - webserver on machine 1 does not share memory with webserver on machine 2
  - Run on an emulated network
    - With latency, packet drops, …!
  - Use the host network stack
    - Real IP, real TCP!



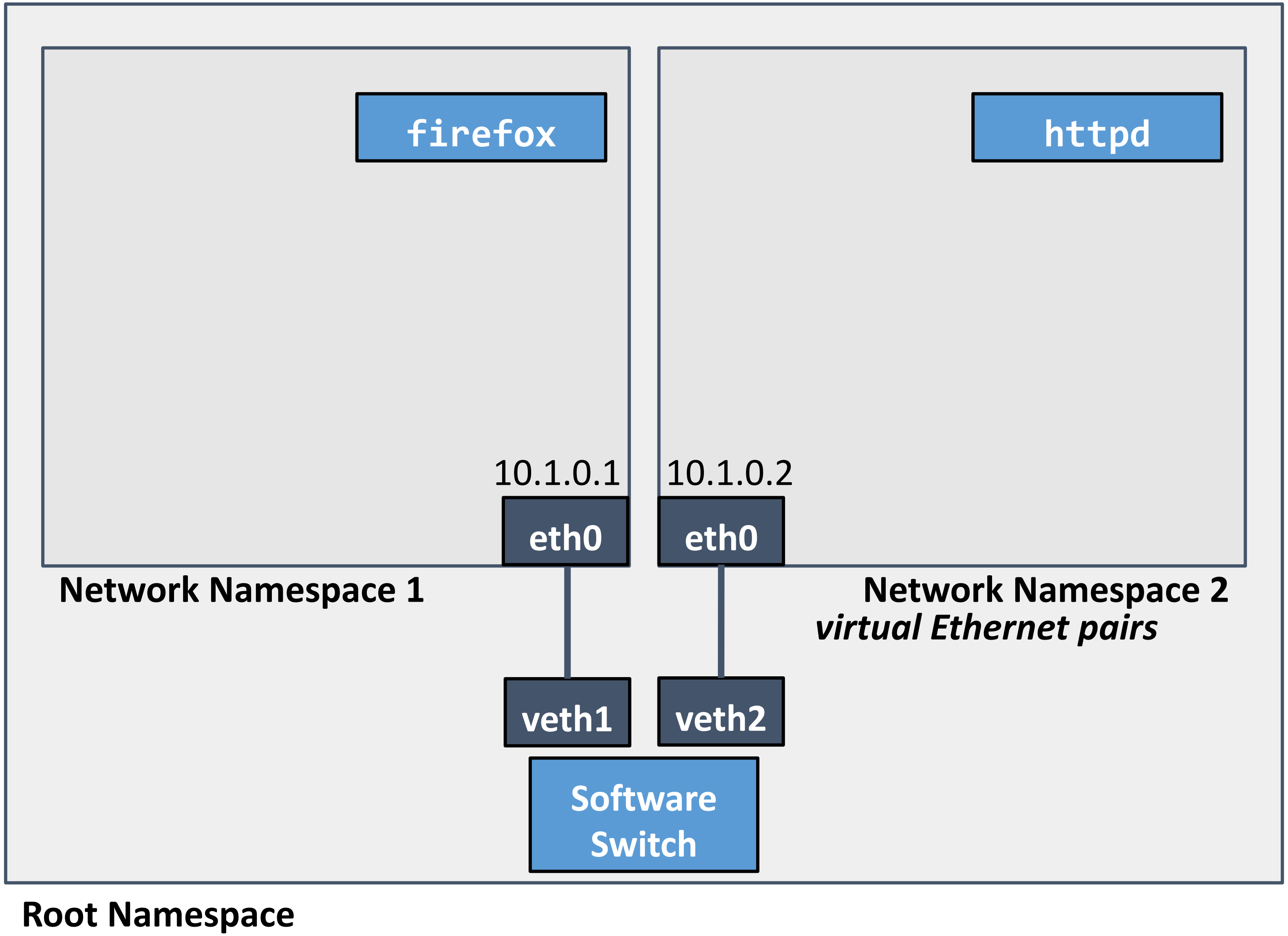
Very Simple Network using Lightweight Virtualization



Server (or VM!)

Credits: SIGCOMM 2014 tutorial

Mechanism: Network Namespaces and Virtual Ethernet Pairs



Credits: SIGCOMM 2014 tutorial

Some basic commands to test

- Running tests
  - pingall
  - iperf
  - h1 ping h2
  - h1 ifconfig
  - h1 xterm
    - opens a terminal on the machine called h1
    - very useful to start a server yourself!
- Stopping the network
  - exit
- Your network crashed?
  - sudo mn –c
    - Will clear all mininet files

Some useful links

- How to install mininet http://mininet.org/download/
- Walkthrough from the basic commands to custom scripts http://mininet.org/walkthrough/
- SIGCOMM 2014 tutorial
  - https://docs.google.com/a/onlab.us/presentation/d/1Xtp05ILQTEFGICTxzV9sQI28wW_cAZz6B1q9_qZBR_8/edit
- Some code examples (advanced): https://github.com/mininet/mininet/tree/master/examples

Labs Deliverables

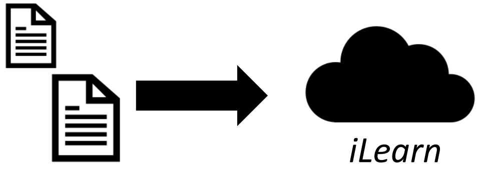
What you NEED to give/show us

(Prelab deliverable is described in prelab)

(Project deliverables will be explained another time)

Deliverables

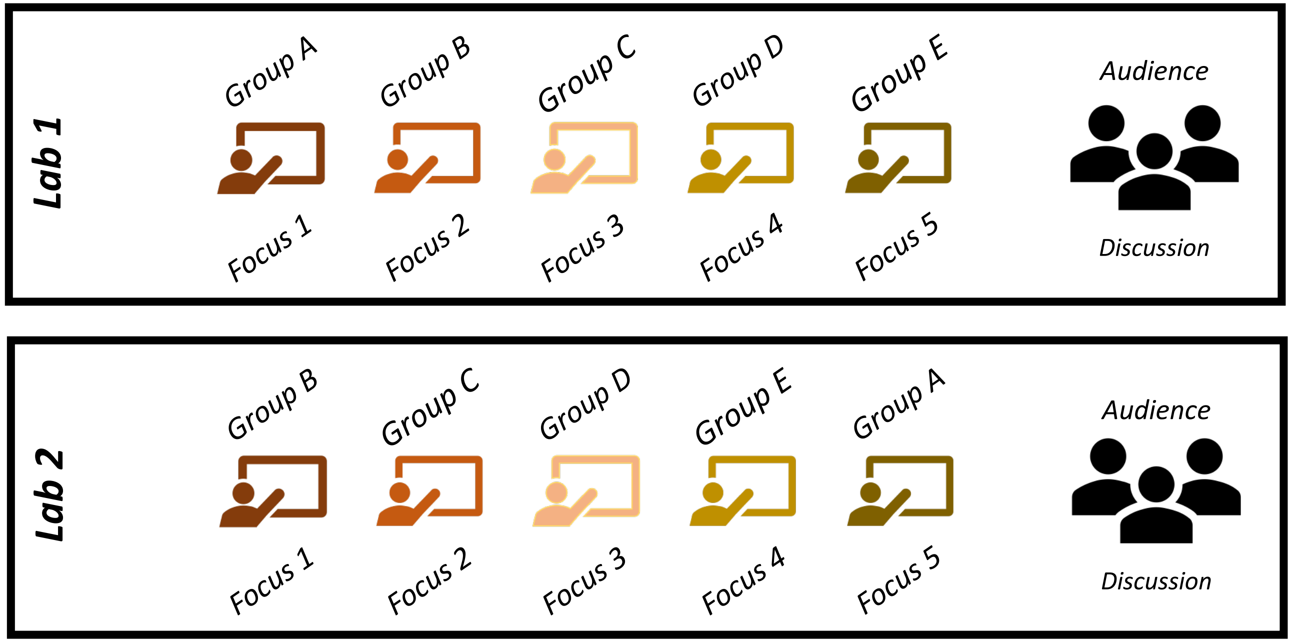
- Your implementation
  - Including any script you used to test consistency!
  - Upload it on iLearn



- A presentation of your work
  - 3 minutes max per group!
  - Use slides (you can use figures, screenshots, videos…)
  - Focus on the *focus* given to your for this lab (see next slides)
  - Followed by a discussion

Focus – what, how

- At the beginning of each lab, we give you a *focus*
- Your presentation should mainly be about this part, other groups will present the other focuses!
- You get a new focus for each lab
- Some focuses are harder than others!



Focus list

- You have been given one of the following focuses:
  - Design
  - Corner Cases
  - Implementation
  - Demo
  - Evaluation

Example:
For lab 1:

- *groups 1 and 6 will present their design*
- *Groups 2 and 7, their corner cases*
- *3 and 8, their implementation*
- *4 and 9 will demo their lab*
- *5 and 10 will evaluate their solutions*

For lab 2, we will change focus!

Focus - Design

- Explain how you designed your system
- What is your algorithm?
- What complexity (message, time)?
- Advantages/Cons of your design vs what we saw in lectures?

Focus - Corner Cases

- Sometimes, protocols need to handle special cases
  - Also known as corner cases

- What corner cases could break your design?

- How did you deal with them?

- Concurrent *Modify* often causes many corner cases…

Focus - Implementation

- Explain how you have implemented your system

- You can focus on some interesting parts

- Did you have to do any tweaks/tricks to make it work?

Focus - Demo

- Show us your system!
- You can record a video or do a live show
  - But live experiments often fail… ☺
- Don’t use screenshots only!
  - We might believe you are hiding a broken system…

Focus - Evaluation

- Evaluate the performance of your system

- Time to convergence?
  - (total amount of time from the first request until all servers have their final board)

- Impact of the number of servers?
  - You can modify start_topology.py for that

- Number of messages?

- This is the hardest focus, but evaluation is a key element for your master thesis/research, so you should learn how to evaluate!

Goto Lab 1

Prelab deadline:

Friday 1st, November 2019!

Credits – based on slides from

- Beshr Al Nahas
- Charalampos Stylianopoulos
- Olaf Landsiedel
- Christos Profentzas
- Iosif Salem
- Ioannis Nikolakopoulos
- And many others