Distributed Systems
Pre-Assignment

In this series of labs, you will learn how to distribute an application logic over distributed (web) servers, and how to deal with different consistency and ordering problems. To do so, we will use several tools: *Mininet*, a network emulator; *Bottle*, a RESTful web framework; and the programming language *Python* (version 2.7).

This prelab will introduce you to these three main building blocks. If you haven't used python before, we strongly recommend you to learn the basics of the language before the real labs start. We will go through the basics of Bottle, as you'll need to create a RESTful application, and explain how to setup Mininet.

Submitting the prelab is mandatory, but the prelab does not count in the final grade.

1 Python

1. Google offers a basic Python course at `https://developers.google.com/edu/python/`. Visit the course and learn basic Python's features if you are not familiar with it.

2. Finish all three basic exercises in the Google's Python course `https://developers.google.com/edu/python/exercises/basic`, i.e., string1.py, list1.py and wordcount.py. Submit your source code of the functions YOU wrote for each of the exercise in your submission. (Note: We know that the solutions for these exercises are also provided online. However you should do them yourself to practice your Python programming skills, which are required in all the labs).

2 Bottle

In the labs you will use Bottle to create web servers. Please check this Bottle tutorial `http://bottlepy.org/docs/dev/tutorial.html`.

Create and submit a web server using Bottle. Your server should display some text on the index (route to '/'). In addition, your server should serve POST requests (route to '/input'), and display all POST parameters received.

3 Mininet

Mininet is a network emulator which creates a network of virtual hosts, switches, controllers, and links. It enables you to create and manage custom network topologies with several hosts, all in the same physical machine. Mininet is a full-blown emulator with many capabilities, but in this course we will use it for a single purpose: to create and connect multiple servers for your Python code to run on.

Visit `http://mininet.org` to learn more about Mininet. Then complete the following tasks:

Task A: Install Mininet. In this link `http://mininet.org/download/` you will find information on how to setup Mininet on your machine. We recommend following option 1 (VM installation). That includes:

- Installing one of the following: *VirtualBox*, *VMware* or *qemu*. Note: You might find working on your own laptop or desktop with *VirtualBox*/*VMware* a bit more convenient (compared to *qemu*).
- Downloading the MininetVM image: https://github.com/mininet/mininet/wiki/Mininet-VM-Images . Note: if you are using *qemu*, download the 32-bit version.
- Go through the VM setup notes: http://mininet.org/vm-setup-notes/

Task B (optional): GUI . After the previous task, you should have successfully installed a Mininet VM with access to the Internet. At this point, you might also want to add a graphic interface to the Mininet image. In this case, follow this section from the mininet FAQ `https://github.com/mininet/mininet/wiki/FAQ#vm-console-gui`.

Note: if you are using *qemu* on Linux, add these extra options when you boot your machine: *-enable-kvm -m 2G -usb -usbdevice tablet -show-cursor*

Task C: Read the walkthrough. An introduction to Mininet can be found here `http://mininet.org/walkthrough/`. It shows how to run Mininet using simple typologies and run basic commands on the hosts.

Task D: Testing topologies. Start a default topology with two hosts and a switch (*sudo mn* should do just that), start a ping from h1 to h2 and mark down the results. Then start Mininet again with *sudo mn --link tc,bw=10,delay=10ms* and do the same ping. Do you see different values in round trip time?