3 Creating new labs

The most challenging and critical part of designing a new cybersecurity lab is the design of the lab itself, i.e., identifying learning objectives and organizing exercises to achieve those objectives. The Labtainer framework does not specifically address any of that. Rather, the framework is intended to allow you to focus more time on the design of the lab and less time on mitigating and explaining system administration and provisioning burdens you would otherwise place on students and instructors.

Typical steps for developing a new lab are:

1. Give the lab a name and create its computers using the `new_lab_setup.py` script;
2. Choose the starting baseline configuration for each computer and add software packages within a Dockerfile;
3. Define networks and connections to the lab computers in the lab's `start.config` file.
4. Populate the user's HOME directory and system directories with lab-specific files.

The remainder of this section covers the fist step and provides an example. The following section 4, covers the other three steps. After a lab is created, you can then optionally parameterize it per section 5 and/or define criteria for automated assessment per section 6

3.1 Create the first lab computer

Labtainer exercises each have their own directory under the "labs" directory in the project repository. The first step in creating a new lab within the framework is to create a directory for the lab and then cd to it. The directory name will be the name used by students when starting the lab. It must be all lower case and not contain spaces.

```
cd $LABTAINER_DIR/labs
mkdir <new lab name>
cd <new lab name>
```

After the new lab directory is created, run the "new_lab_setup.py" script. [4]

```
new_lab_setup.py
```

This will create a set of template files that you can then customize for the new lab. These template files are referenced in the discussion below. The result of running `new_lab_setup.py` is a new labtainer lab that can be immediately run. While this new lab will initially only present you with a bash shell to an empty directory on a Linux computer, it is worth testing the lab to understand the workflow.

3.2 Testing the new lab

Once a new lab directory is created, and the new_lab_setup.py has been run, then you can test the new, (currently empty) lab. All student labs are launched from the labtainer-student directory. Lab development workflow is easiest if at least two terminals or tabs are used, one in the new lab directory, and one in the labtainer-student directory. So, open a new tab or window, and then:

[4] The `$LABTAINER_DIR` will have been defined in your .bashrc file when you installed Labtainers. It should point to the `labtainers/trunk` directory. You may need to start a new `bash` shell to inherit the environment variable.