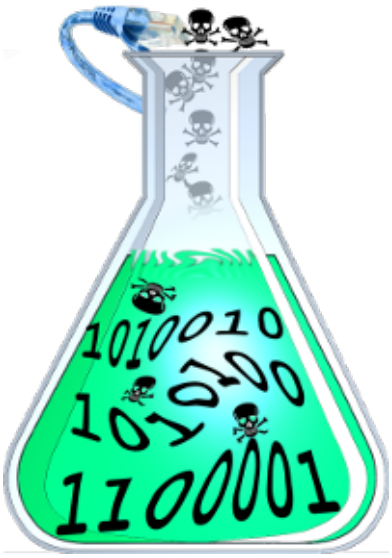
Labtainers Instructor Guide

Fully provisioned cybersecurity labs

May 25, 2021



1 Introduction

This manual is intended for use by instructors who assign and/or grade labs using Labtainers.

Labtainers provide a consistent execution environment for performing laboratory exercises, and can include execution of several different computers interconnected via virtual networks. Refer to our published papers at `https://nps.edu/web/c3o/labtainers` for additional information on the use of Labtainers. See 5 for information on creating and maintaining Labtainer exercises.

The easiest way to get Labtainers is to use our pre-built VM available at the Labtainer website `https://nps.edu/web/c3o/virtual-machine-images`. Note that any Linux system can be used as long as it supports Docker, and it can be used on *Docker Desktop* installations on Mac and Windows computers as an alternative to using the VM appliance. If Labtainers is to be used on a system other than the pre-built VM, refer to the Labtainer Student Guide for information on installing Labtainers.

Running Labtainers on servers, e.g., cloud deployments is discussed in section 4.4

2 Assigning Labs

Pior to assigning a lab, become familiar with it by reviewing the lab and its manual.

Student instructions for using Labtainers are in the Labtainer Student Guide. Students work from the `labtainer-student` directory, i.e.,

```
cd ~/labtainer/trunk/scripts/labtainer-student
```

2.1 Selecting Labs

Available labs are listed via the `labtainer` script:

```
labtainer
```

Use the `-k` option to see a list of searchable keywords, and the `-f <keyword>` option to view a summary of labs having that keyword.

Lab exercises are also organized into *Labpacks*. These are ordered collections of multiple related labs that you may wish to assign to students. Use this command:

```
labpack
```

to view a list of Lab Packs, and provide the name of a Labpack as an argument to see a list of the labs within a Labpack. You may also create your own Labpacks as described in **??**.

Available labs are also summerized and organized into broad categories at `https://nps.edu/web/c3o/labtainer-lab-summary1`.

Additional lab exercises created by instructors are available as IModules, which are listed at `https://nps.edu/web/c3o/imodules`. Students can get access to those labs using:

```
imodule <url>
```

where `url` is that provided on the IModules web page.

2.2 Try the Lab

Start a lab by providing its name as an argument to the `labtainer` command. This will typically display a link to a lab manual, or will display a lab manual in one of the resulting virtual terminals. You can interact with the resulting computers just as a student would.

3 Assessing Lab Performance

When the student stops a lab, i.e., using `stoplab`, Labtainers creates a zip file of student artifacts (including lab reports) and then displays the path to this zip file to the student. The easiest way for the student to forward this zip file to you is by starting a browser on the Linux VM and either emailing you the zip file, or uploading the file into an LMS, (e.g., Sakai). Alternately, the student can define a shared folder in the VM and copy the zip to the host computer.

Collect all of the lab zip files from each student into your Labtainer transfer directory, which is typically at

```
~/labtainer_xfer/<labname>
```

where labname is the name of the lab. Do not unzip the files. Alternately student assignments can be bulk-collected from a learning management system (LMS) per Appendix A and the resulting zip would be copied into the Labtainer transfer directory. Again, do not unzip files and do not change the file names of zip files. See subsection 4.3 for how to manage the student zip files on your host computer.

Instructor assessment of labs takes place from the `labtainer-instructor` directory, i.e.,

```
cd ~/labtainer/trunk/scripts/labtainer-instructor
```

Use the `gradelab` command to assess results for a given lab:

```
gradelab <labname>
```

A table of lab results with one row per student and a column for each goal will be displayed. A description of the goals follows the table. A web-based display of that data is available as described in subsection 3.1.

Note that not all labs include automated assessment. For those labs, you will see this message:

```
No automated assessment for this lab
```

Even when no automated assessment is performed, you can still observe student performance artifacts, e.g., the `.bash_history` file and files created by the student as described below in 3.1.

Use the `-r` option to perform a fresh grading, e.g., if you've removed files from the `labtainer_xfer` directory. Sometimes zip files within the `labtainer_xfer` directory are corrupted. If error messages indicate a bad zip file, try removing it from the directory and then use the `-r` option to perform a fresh grading. Use the `-u` option to update your gradelab to the latest image.

Student reports (if any) are copied into

```
~/labtainer_xfer/<labname>/docs
```

on the Linux host. If LMS assignment collection is used, then student reports should be looked for in

```
~/labtainer_xfer/<labname>/reports
```

which also includes reports separately uploaded into the LMS.

3.1 Review lab artifacts

An early release of a web-based tool for viewing details of student assessment results and student artifacts is available by use of the `-w` flag with the `gradelab` command. That causes the grader container to listen on port 8008 of the Labtainer VM. You can then open a browser on that VM and go to `localhost:8008`. Alternately, use your host machine's browser by setting port forwarding on your VM, (e.g., in VirtualBox, use Machine / Settings / Network / Advanced / Port Forwarding to set host IP 127.0.0.1:8008 to map to guest IP 0.0.0.0:8008).

The table of goals displayed in the browser includes links to details of artifacts created by the student when performing the lab. For example, clicking on the student name displays a table of all timestamped result artifacts. That page includes a **History** heading with links to the `.bash_history` files one each container. And it includes a table with links to files in the student home directory and links to result files, e.g., stdout from selected commands issued by the student.

Links within each goal table cell lead to pages whose content depends on the type of goals defined. For example, a goal whose value is defined by a boolean expression will lead to a table of all boolean values for each timestamp for which results are present.

Definitions of different goal types and result types can be found in the *Lab Designer Guide*. Note that you need not understand all of the displayed data in order to gain useful insight into student progress. Some of the displayed information requires an understanding of the Labtainers automated assessment configuration directives, and is made available in the displays primarily in support of those developing automated assessment for labs.

3.1.1 Artifacts on the grader container

You can view all student results, including their original artifacts by using the `-d` flag with the `gradelab` command. This results in a virtual terminal connected to a grading container that contains all student artifacts and results. If you have not first run the `gradelab` command without the "-d" option, run `instructor.py` from within the virtual terminal to cause the zip files to be extracted. A student's home directory can then be found in

```
<student email>/<lab>.<container>.student
```

There you will find the `.bash_history` file along with the student-created files. Student artifacts collected by the framework are found in

```
<student email>/<lab>.<container>.student/.local/result
```

The `-d` option is also used when debugging automated assessment configuration files. You can create additional virtual terminals into the grading container by reissuing the gradelab command with the `-a` flag. When you are finished, or wish to stop working, type:

```
stoplab
```

4 Managing Labtainer Installations and Updates

Any given Labtainers installation can be brought up to date to the latest version by using the

```
update-labtainer.sh
```

command from the `labtainer-student` directory. The current version of a Labtainer installation is seen by using:

```
labtainer -v
```

The first time any given lab exercise is started, the latest version of that lab is automatically pulled from the Docker Hub registry. Note however that any given lab is not updated by the `update-labtainer.sh` command once the lab has been started. To update a specific lab to the latest version after it has been started the previous version of that lab must be deleted using:

```
removelab.py <labname>
```

The next time the lab is started, the latest version will be retrieved from the Docker registry.

If you want to update the labtainer.grader docker image (and delete the previous image and grader containers) use:

```
gradelab -u <labname>
```

4.1 Suggestions for student workflow

A student's work on any given lab is preserved until and unless the student restarts the lab using the "-r" option on the `labtainer <labname> -r` command. When taking a break from work on a lab, the student can either stop the lab using `stoplab`, or simply pause the VM. However, if the student wishes to perform other Labtainer-related work on the VM, (e.g., revisit a previous lab), they should first use `stoplab` for the current lab. When they restart the lab, none of their work will be lost.

4.2 Deploying without the Internet

Labtainers pulls Docker images from Docker Hub when a student first runs any given lab. You can deploy Labtainers within environments that have no Internet connection by first creating your own VM template. Start with the standard Labtainers VM, and run the script at

```
$LABTAINER_HOME/setup_scripts/pull_lab.py
```

to pull images for your desired labs onto the VM. Then replicate that VM for each user, e.g., by exporting it as an appliance.

Note that a few labs deliberately access the Internet, e.g., the public key lab. You can either avoid use of those labs, or alter them to direct students to alternate network addresses.

4.3 Collecting student zip files on the host computer

It is often more convenient for instructors to gather student zip files on the computer that hosts a Labtainers VM rather than on the VM. For example, email clients and/or LMS interfaces may run more easily on the host than they do on the VM. This can be achieved by defining a shared folder for use by the VM guest, as follows:

- Pick a directory on your host that you will share with the VM. Somewhere within that directory create a `labtainer_xfer` subdirectory.
- Make that directory accessible by anyone on your host machine.
- Define a shared folder for your guest VM, e.g., on VirtualBox, use Machine / Settings / Shared folders, map it to your selected directory.
- On VirtualBox, ensure your user ID is within the vboxsf group, and reboot.

```
sudo usermod -G vboxsf -a $USER
sudo reboot
```

- On the virtual machine, identify the path to the `labtainer_xfer` directory in the shared folder. For example, if you shared a directory called `mydir` on VirtualBox, that might be found at `/media/sf_mdir/labtainer_xfer`.
- From the $HOME directory on the virtual machine, remove the `labtainer_xfer` directory and replace it with a symbolic link to the shared `labtainer_xfer` directory, e.g.,

  ```
  cd
  ln -s /media/sf_mydir/labtainer_xfer
  ```

  depending on where you created the new `labtainer_xfer` directory within the shared folder.
- Now place student zip files on the host within the `labtainer_xfer/<labname>` directory. Note the lab subdirectory will be created when you start the lab to run it yourself – or you can create it manually.

4.4 Deploying on servers

Labtainers can be deployed on servers and accessed by students using a web browser. This assumes you have access to suitable infrastructure and IT support. Two general approaches are:

1. Virtual Desktop Infrastructure – Use VDI products such as VMWare Horizon to run Labtainer VMs. In these environments, each student is allocated a VM, and that VM's desktop is seen by the student in the browser. Students deliver their results to instructors by starting a browser within the VM, e.g., to access an LMS or web-mail account.
2. *Headless Labtainers* – Labtainers are deployed as servers in a cloud and a *NOVNC* desktop is rendered using a web browser. Access to the Labtainer server instance is via HTTP through an SSH tunnel. Please see `https://raw.githubusercontent.com/mfthomps/Labtainers/master/headless-lite/README.md` for additional information, including a sample cloud-config file.

5 Customizing Labtainers

5.1 New and custom lab exercises

Creating new labs and modifying existing labs is described in the *Labtainers Lab Designer User Guide*. `https://github.com/mfthomps/Labtainers/raw/master/docs/labdesigner/labdesigner.pdf`

That guide also describes how to use *IModules* to provide your students with custom versions of the lab manuals, and how to publish new labs so that they can be incorporated into your student's Labtainers instances, and shared with other educators.

5.2 Create new Labpacks

You can organize lab exercises into your own Labapcks using the `makepack` command. First go to the

```
$LABTAINER_HOME/scripts/labtainer-instructor
```

directory and then use the `makepack` command, providing the name of the Labpack that you wish to create or modify.[1]

```
makepack mypack1
```

This results in a shell that accepts makepack comands. Use either `h` or `?` to get help. Note that chages to Labpacks are stored immediately, there are no save/quit options.

Labpacks are stored in the `$LABTAINER_HOME/labpacks` directory. To publish one or more Labpacks so that they are available to your students, go to the `labpacks` directory and use tar to create a tarball containing each of your Labpacks. For example:

```
tar tf mypacks.tar mypack1 mypack2
```

Include only the names of your custom Labpacks that you wish your students to receive. Then post the resulting tarball on a website and provide your students with the URL. Students will then provide that URL to the `labpack` command:

```
labpack -a <url>
```

to get access to your Labpacks.

[1] Do not modify other Labpacks, only modify those that you've created.

A
LMS Assignment Collection

A.1 Sakai

In the Sakai Assignments section, select the "In / New" entry for the appropriate assignment. The resulting page should enumerate each student who has submitted an assignment. In the upper right, click the "Download All" link, and then click the "Student submission attachment(s)" option and click the "Download" button. Copy the resulting zip into the lab transfer directory on the Linux host, i.e.,

```
~/labtainer_xfer/<labname>
```

Do not unzip the file and do not change its file name. You can then run the `gradelab <labname>` command from the `labtainer-instructor` directory. In addition to the assessment summary, any student lab reports will be available in:

```
~/labtainer_xfer/<labname>/reports/<student name>
```

Those reports will include any that the student separately uploaded into Sakai (it is important to remind students to NOT change the name of lab report documents.)

A.2 Moodle

See the Moodle user guide at
`https://moodleuserguides.org/guides/bulk-download-assignment-submissions/` for information on getting a bulk download, but DO NOT unzip the file. Copy the resulting zip into the lab transfer directory on the Linux host, i.e.,

```
~/labtainer_xfer/<labname>
```

Do not unzip the file and do not change its file name. You can then run the `gradelab <labname>` command from the `labtainer-instructor` directory. In addition to the assessment summary, any student lab reports will be available in:

```
~/labtainer_xfer/<labname>/reports/<student name>
```

Those reports will include any that the student separately uploaded into Moodle (it is important to remind students to NOT change the name of lab report documents.)

A.3 Other LMS

Send me a sample of the bulk download file from other LMS systems and we'll roll it into a future Labtainers release. (mfthomps at nps.edu)