networks. Students see and interact with Linux computers, primarily via bash shell commands and GUI-based applications. In general, the Labtainer framework implementation is not visible to the student, and the Linux environment as seen by the student is not noticeably augmented to support the framework.

Labtainers are intended for use on individual student computers, e.g., a laptop, or potentially a VM allocated to the student from within a VM farm. [2] The computer utilized by a student must include the Linux operating system, e.g., as a single VM. This Linux operating system, referred to herein as the *Linux host*, can be any distribution and version which supports Docker. Students download and expand a tarball, and run an installation script as described in the *Labtainer Student Guide* [3] Alternately, students can use a Linux VM that is pre-configured with Labtainers and Docker, and is available at our website.

It is suggested that the student's Linux host be a virtual machine that is not used for purposes requiring trust. Software programs contained in cybersecurity lab exercises are not, in general, trusted. And while Docker containers provide namespace isolation between the containers and the Linux host, the containers run as privileged.

Labtainer exercises can include networking to external hosts, e.g., a Windows VM running alongside the Linux host VM, as described in section 8.3.

Students initiate any and all labs from a single workspace directory on the Linux host. To perform a specific Labtainer exercise, the student runs a *labtainer* command from the Labtainer workspace, naming the lab exercise. This results in one or more containers starting up along with corresponding virtual terminals via which the student will interact with the containers. These virtual terminals typically present a bash shell. Each container appears to the student as a separate computer, and these computers may appear to be connected via one or more networks.

When a student starts a given exercise for the first time, the framework fetches Docker images from the Docker registry. Docker manages container images as a set of layers, providing efficient storage and retrieval of images having common components. The initial Labtainer installation step pulls a few baseline images (about 1.5 GB) from the public Docker registry, known as the *Docker hub*. Images for specific labs are pulled from the Docker hub by downloading only those additional layers required by that lab, and which had not been previously pulled from the hub. This is transparent to the student, other than waiting for downloads to complete.

After the student performs the lab exercise, artifacts from the container environments are automatically collected into an archive, (a zip file), that appears on the student's Linux host. The student forwards this archive file to the instructor, e.g., via email or a learning management system (LMS). The instructor collects student archive files into a common directory on his or her own Linux host, and then issues a command that results in automated assessment of student lab activity, (if the lab is designed for that), and the optional creation of an environment in which the instructor can review the work of each student.

Many cybersecurity lab exercises are assessed through use of reports in which students describe their activities and answer specific questions posed by the instructor. Labtainers are intended to augment, rather than supplant this type of reporting. The framework includes mechanisms for automating the collection of student lab reports into the artifact archive files that are collected by instructors.

[2] Labtainers can also support labs in which students collaborate (or compete) on shared infrastructure. Please see section 12 for information on multi-user environments.

[3] This tarball may someday be replaced by standard Linux distribution packages, e.g., Debian and/or RPM packages.