`certs` directory.

```
[student@workstation ~]$ tree certs
certs
├── classroom-ca.pem
├── control2.lab.example.com.crt
├── control2.lab.example.com.key
├── controller.lab.example.com.crt
├── controller.lab.example.com.key
├── db.lab.example.com.crt
├── db.lab.example.com.key
├── exec1.lab.example.com.crt
├── exec1.lab.example.com.key
├── exec2.lab.example.com.crt
├── exec2.lab.example.com.key
├── exec3.lab.example.com.crt
├── exec3.lab.example.com.key
├── hop1.lab.example.com.crt
├── hop1.lab.example.com.key
├── hub.lab.example.com.crt
└── hub.lab.example.com.key

0 directories, 17 files
```

You do not use all of these files in this exercise.

1.2. The `controller` directory contains the manifest file to use for subscribing to the automation controller. Use the `tree` command to verify this.

```
[student@workstation ~]$ tree controller
controller
└── manifest.zip

0 directories, 1 file
```

2. The `lab` command also downloads the Red Hat Ansible Automation Platform Bundle installer. Extract it. For simplicity, rename the extracted directory to `aap2.2-bundle`. Change to the directory that contains the extracted contents.

```
[student@workstation ~]$ tar xzf \
> ansible-automation-platform-setup-bundle-2.2.0-6.1.tar.gz
[student@workstation ~]$ mv ansible-automation-platform-setup-bundle-2.2.0-6.1 \
> aap2.2-bundle
[student@workstation ~]$ cd aap2.2-bundle
```

3. Modify the inventory file to specify the following details:

- The three servers to install
- The passwords for the administrator account on each server
- The container registry for the automation controller
- The variables related to the database
- The variables related to the certificates

3.1. Edit the `inventory` file to specify the fully qualified domain name (FQDN) of the automation controller, the private automation hub, and the database servers as follows:

Section of the inventory	FQDN
[automationcontroller]	controller.lab.example.com
[automationhub]	hub.lab.example.com
[database]	db.lab.example.com

3.2. Set the values for the following variables related to the automation controller, its container registry, and its database.

Variable	Value
admin_password	redhat
pg_host	db.lab.example.com
pg_password	redhat
registry_url	hub.lab.example.com
registry_username	admin
registry_password	redhat

> **NOTE**
>
> You use the private automation hub as the container registry for the automation controller.

3.3. Set the values for the following variables related to the private automation hub and its database:

Variable	Value
automationhub_admin_password	redhat
automationhub_pg_host	db.lab.example.com
automationhub_pg_password	redhat

3.4. Set the variables for the signed certificates for the `controller`, `hub`, and `db` servers, the associated private keys, and the GLS Training Classroom CA certificate as follows:

Variable	Value
custom_ca_cert	/home/student/certs/classroom-ca.pem
web_server_ssl_cert	/home/student/certs/controller.lab.example.com.crt

Variable	Value
web_server_ssl_key	/home/student/certs/controller.lab.example.com.key
automationhub_ssl_cert	/home/student/certs/hub.lab.example.com.crt
automationhub_ssl_key	/home/student/certs/hub.lab.example.com.key
postgres_use_ssl	True
postgres_ssl_cert	/home/student/certs/db.lab.example.com.crt
postgres_ssl_key	/home/student/certs/db.lab.example.com.key

3.5. When modified, the uncommented content of the `inventory` file displays as follows:

```
[automationcontroller]
controller.lab.example.com
[automationcontroller:vars]
peers=execution_nodes
[execution_nodes]
[automationhub]
hub.lab.example.com
[automationcatalog]
[database]
db.lab.example.com
[sso]
[all:vars]
admin_password='redhat'
pg_host='db.lab.example.com'
pg_port=5432
pg_database='awx'
pg_username='awx'
pg_password='redhat'
pg_sslmode='prefer'  # set to 'verify-full' for client-side enforced SSL
registry_url='hub.lab.example.com'
registry_username='admin'
registry_password='redhat'
receptor_listener_port=27199
automationhub_admin_password='redhat'
automationhub_pg_host='db.lab.example.com'
automationhub_pg_port=5432
automationhub_pg_database='automationhub'
automationhub_pg_username='automationhub'
automationhub_pg_password='redhat'
automationhub_pg_sslmode='prefer'
automationcatalog_pg_host=''
automationcatalog_pg_port=5432
automationcatalog_pg_database='automationservicescatalog'
automationcatalog_pg_username='automationservicescatalog'
automationcatalog_pg_password=''
custom_ca_cert=/home/student/certs/classroom-ca.pem
web_server_ssl_cert=/home/student/certs/controller.lab.example.com.crt
web_server_ssl_key=/home/student/certs/controller.lab.example.com.key
automationhub_ssl_cert=/home/student/certs/hub.lab.example.com.crt
automationhub_ssl_key=/home/student/certs/hub.lab.example.com.key
postgres_use_ssl=True
postgres_ssl_cert=/home/student/certs/db.lab.example.com.crt
postgres_ssl_key=/home/student/certs/db.lab.example.com.key
sso_keystore_password=''
sso_console_admin_password=''
```

Save and close the file.

> **NOTE**
>
> You can use the `~/install-installation/inventory` file for comparison.

4. As the `root` user, run the installation script.

4.1. Use the `sudo` command to change to the `root` user, using `student` as the password. Change to the `/home/student/aap2.2-bundle` directory and execute the `setup.sh` installation script. The script can take up to 20 minutes to complete.

```
[student@workstation aap2.2-bundle]$ sudo -i
[sudo] password for student: student
[root@workstation ~]# cd /home/student/aap2.2-bundle
[root@workstation aap2.2-bundle]# ./setup.sh -e ignore_preflight_errors=true
Using /etc/ansible/ansible.cfg as config file
...output omitted...

PLAY RECAP *********************************************************************
controller.lab.example.com : ok=289  changed=136  ...  failed=0  ...  ignored=6
db.lab.example.com         : ok=77   changed=31   ...  failed=0  ...  ignored=1
hub.lab.example.com        : ok=213  changed=89   ...  failed=0  ...  ignored=1
localhost                  : ok=3    changed=1    ...  failed=0  ...  ignored=0

The setup process completed successfully.
[warn] /var/log/tower does not exist. Setup log saved to setup.log.
```

IMPORTANT

You run the installation script as the `root` user because the `root` user on the `workstation` machine has root access to the `controller`, `hub`, and `db` machines in this lab environment.

Failure to run the installation script as the `root` user in the classroom environment results in the following error messages:

```
fatal: [controller.lab.example.com]: FAILED! => {"changed": false, "ms
g": "UID on remote machine is 1000 (0 required). Check Ansible connectio
n and become settings."}
fatal: [hub.lab.example.com]: FAILED! => {"changed": false, "msg": "UID
on remote machine is 1000 (0 required). Check Ansible connection and bec
ome settings."}
fatal: [db.lab.example.com]: FAILED! => {"changed": false, "msg": "UID o
n remote machine is 1000 (0 required). Check Ansible connection and beco
me settings."}
```

Because the lab environment does not satisfy the minimum memory requirements for the automation controller and the private automation hub, set the `ignore_preflight_errors` Ansible variable to `true` to ignore prerequisite checks made before installation starts. This should not be set in a production environment.

4.2. After the installer finishes successfully, exit from the `root` session.

```
[root@workstation aap2.2-bundle]# exit
```

5. On the `workstation` machine, navigate to the automation controller web UI to finish the initial configuration. Verify that no warning messages are displayed about the authenticity of the certificate for the automation controller.

   5.1. Open a browser and navigate to `https://controller.lab.example.com` (`https://controller.lab.example.com`). Because you configured the certificate for the automation controller in the `inventory` file, and because the `workstation` machine trusts the CA that signed the certificate, no warning message is displayed about the authenticity of the certificate.

   > **IMPORTANT**
   >
   > The machines in the classroom environment trust any certificate signed by the GLS Training Classroom CA. If your organization uses a corporate or an enterprise certificate authority, then you can configure your machines to trust certificates signed by that certificate authority. Copy the file identified by the `custom_ca_cert` variable to the `/etc/pki/ca-trust/source/anchors` directory and run the `update-ca-trust` command.
   >
   > These steps are not necessary if you use certificates signed by a publicly recognizable certificate authority.

   5.2. Click the lock icon next to the URL, and then on the pop-up menu, verify this is a secure connection.

   Optional. Click **Connection secure**, **More Information**, and then click **View Certificate** to view the certificate.

   5.3. Log in to the web UI as the `admin` user with `redhat` as the password.

   5.4. The screen automatically displays the option to upload the subscription manifest file. Click **Browse** and select the `manifest.zip` file located in the `/home/student/controller` directory. Click **Next**.

   5.5. Do not make any changes to the **User and Insights analytics** step. Click **Next**.

   > **NOTE**
   >
   > Although the red asterisk in the **Username** and **Password** fields might suggest that it is mandatory to provide your Red Hat credentials, it is not. When you provide the credentials the automation controller can send the collected data to https://cloud.redhat.com (https://cloud.redhat.com). This is not necessary for this exercise.

   5.6. Click **Submit** to accept the End User License Agreement.

   5.7. The automation controller web UI displays the Dashboard.

6. On the `workstation` machine, navigate to the private automation hub web UI to ensure that the installation was successful. Verify that there is no warning message about the authenticity of the certificate for the private automation hub.

   6.1. Open a new tab in the browser and navigate to `https://hub.lab.example.com` (`https://hub.lab.example.com`). As with the controller server, there is no warning message about the authenticity of the certificate. Click the lock next to the URL, and then verify the secure connection on the pop-up menu.

   6.2. Log in to the web UI as the `admin` user with `redhat` as the password.

   6.3. The private automation hub web UI displays the dashboard.

7. As the `postgres` user on the database server, verify the existence of the `awx` and the `privateautomationhub` databases.

   7.1. Log in to the `db.lab.example.com` database server as the `student` user. Use the `sudo` command to become the `postgres` user, and then list the databases using the `psql` command. Verify that both databases appear in the list.

```
[student@workstation aap2.2-bundle]$ ssh student@db.lab.example.com
Warning: Permanently added 'db.lab.example.com' (ECDSA) to the list of known hosts.
...output omitted...
[student@db ~]$ sudo su - postgres
[sudo] password for student: student
[postgres@db ~]$ psql -U postgres -l
                                  List of databases
     Name      |     Owner     | Encoding |   Collate   |    Ctype    |  ...
---------------+---------------+----------+-------------+-------------+-----
 automationhub | automationhub | UTF8     | en_US.UTF-8 | en_US.UTF-8 |  ...
 awx           | awx           | UTF8     | en_US.UTF-8 | en_US.UTF-8 |  ...
 postgres      | postgres      | UTF8     | en_US.UTF-8 | en_US.UTF-8 |  ...
 template0     | postgres      | UTF8     | en_US.UTF-8 | en_US.UTF-8 |  ...
 template1     | postgres      | UTF8     | en_US.UTF-8 | en_US.UTF-8 |  ...
(5 rows)
```

   7.2. Exit from the `postgres` session and from the `db` machine.

```
[postgres@db ~]$ exit
logout
[student@db ~]$ exit
logout
Connection to db.lab.example.com closed.
```

Finish

On the `workstation` machine, change to the `student` user home directory and use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish install-installation
```

This concludes the section.