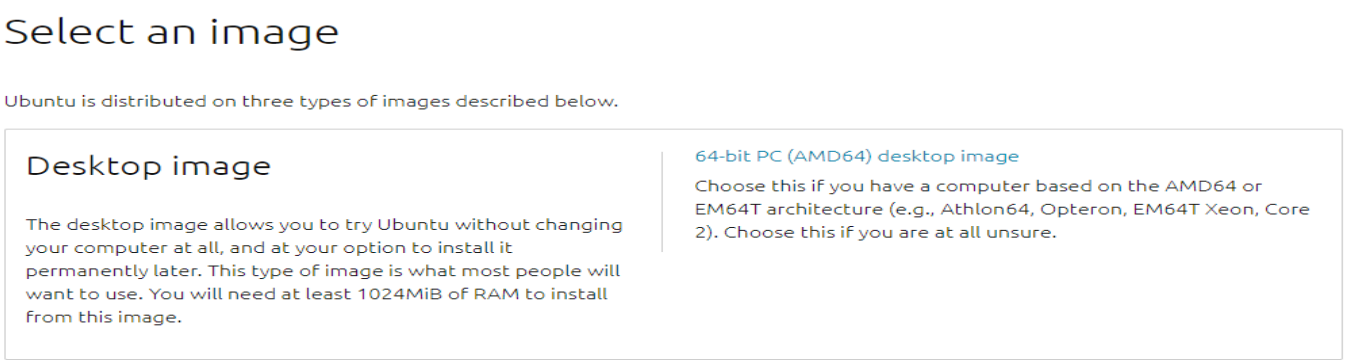
ENVIRONMENT SETUP

We have already given a VM for each of you which is a Windows 10 machine. Log in to your VM and install a new Ubuntu-20.04 in the VirtualBox we already installed on your machine. Although Ububtu-20.04 is installed there you will install another one by your own and work on the machine you built inside the VirtualBox.

Here is the download link for Ubuntu-20.04 https://releases.ubuntu.com/focal/ . Go to the link and select the following one:



Here is the tutorial for installing Ubuntu in VirtualBox https://ubuntu.com/tutorials/how-to-run-ubuntu-desktop-on-a-virtual-machine-using-virtualbox#1-overview

Step 1: Build required toolchain

1.1 Base dependencies

The basic build package on Ubuntu is the build-essential package. To install run:
$ sudo apt-get update
$ sudo apt-get install build-essential

1.2 Additional base dependencies for building seL4 projects on Ubuntu include installing:

$ sudo apt-get install cmake ccache ninja-build cmake-curses-gui
$ sudo apt-get install libxml2-utils ncurses-dev
$ sudo apt-get install curl git doxygen device-tree-compiler
$ sudo apt-get install u-boot-tools

1.3 For Ubuntu20.04, install cross-compiling for ARM targets:

$ sudo apt-get install python3-dev python3-pip python-is-python3
$ sudo apt-get install protobuf-compiler python3-protobuf

1.4 Simulating with QEMU:

To run seL4 projects on a simulator you need to install QEMU on VM:
$ sudo apt-get install qemu-system-arm qemu-system-x86 qemu-system-misc

```
zhliao@Ubuntu2004:~$ sudo apt-get install qemu-system-arm qemu-system-x86 qemu-s
ystem-misc
[sudo] password for zhliao:
Reading package lists... Done
Building dependency tree
Reading state information... Done
qemu-system-arm is already the newest version (1:4.2-3ubuntu6.24).
qemu-system-misc is already the newest version (1:4.2-3ubuntu6.24).
qemu-system-x86 is already the newest version (1:4.2-3ubuntu6.24).
0 upgraded, 0 newly installed, 0 to remove and 6 not upgraded.
```

Build seL4 manual (optional):

If you would like to build the seL4 manual, you will need the following LaTeX packages:
$ sudo apt-get install texlive texlive-latex-extra texlive-fonts-extra

1.5 Python dependencies:

Python dependencies are required to build seL4, the manual and its proofs. To install you can run:
$ pip3 install --user setuptools
$ pip3 install --user sel4-deps
# Currently seL4 foundation duplicate dependencies for python2 and python3 as a python3 upgrade is in process
$ pip install --user setuptools
$ pip install --user sel4-deps

Step 2: CAmkES Build Dependencies

To build a CAmkES based project on seL4, additional dependencies need to be installed on your host machine. Projects using CAmkES (the seL4 component system) need Haskell and some extra Python libraries in addition to the standard build tools. The following instructions cover the CAmkES build dependencies for Ubuntu.

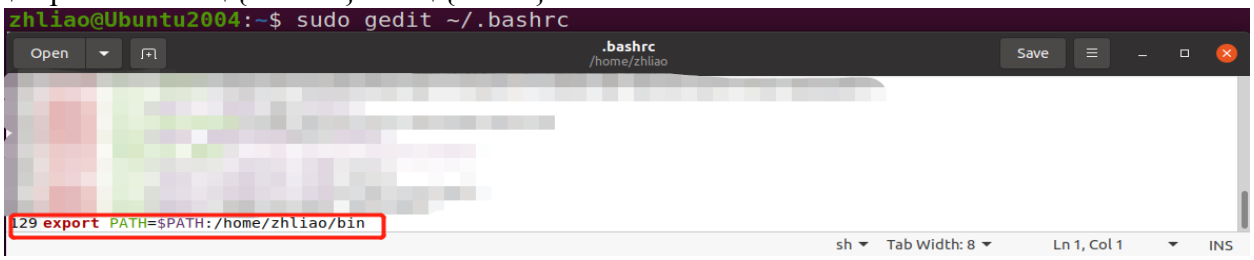
2.1 Install Google's Repo tool

$ sudo apt-get update
$ sudo apt-get install repo
You might see an error while installing the repo. If the above commands do not work, please install it manually as follows:

$ mkdir -p ~/.bin
$ PATH="${HOME}/.bin:${PATH}"
$ curl https://storage.googleapis.com/git-repo-downloads/repo > ~/.bin/repo
$ chmod a+rx ~/.bin/repo

#Then add the ~/.bin directory to your PATH variable to make it work:

$nano ~/.bashrc (Please learn how to save and exit in nano)
Go to the end of the file and add the following line
$export PATH="${HOME}/.bin:${PATH}"



Do not forget to execute the source command to make it work:
$ source ~/.bashrc

Next, run this command in a regular directory (A regular directory isn't part of a repo client; for example, it's your home directory.) to check if the installation is successful:
$repo version

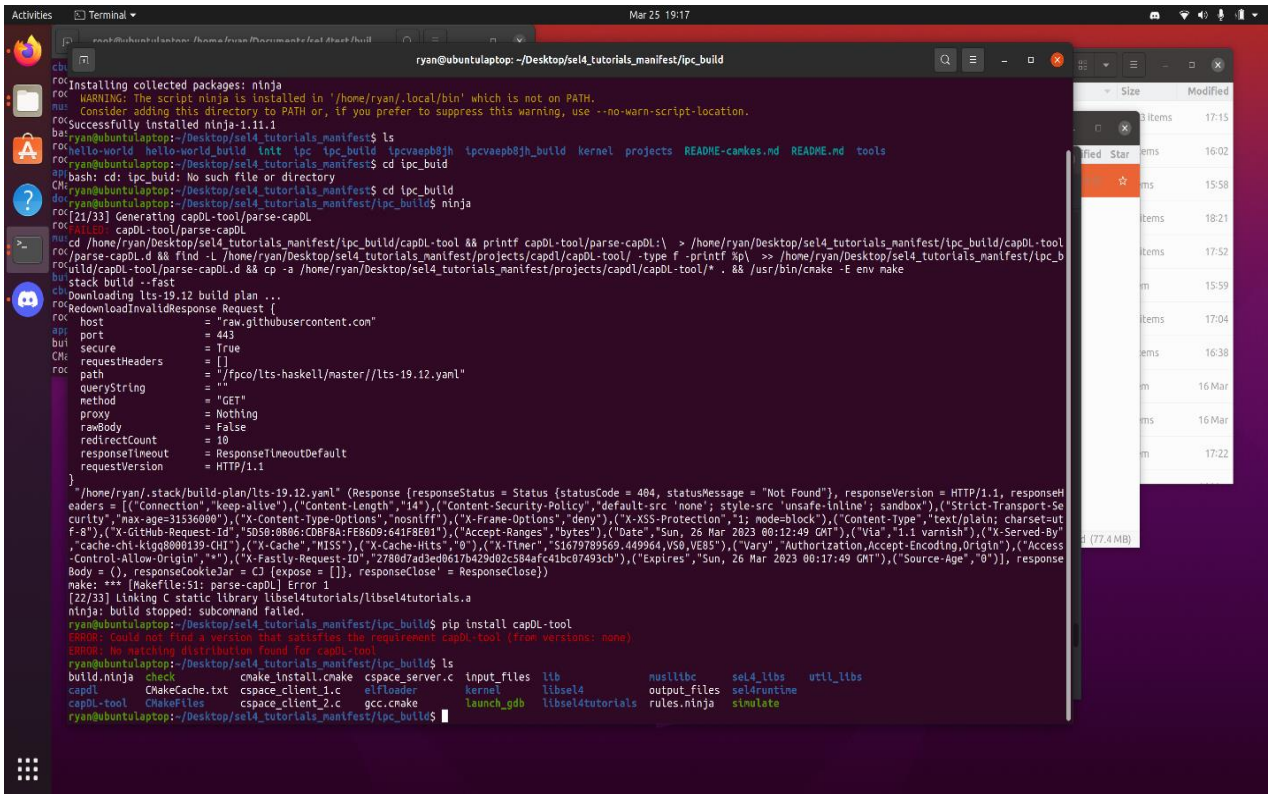
```
zhliao@Ubuntu2004:~$ repo version
<repo not installed>
repo launcher version 2.30
       (from /home/zhliao/bin/repo)
git 2.25.1
Python 3.8.10 (default, Nov 14 2022, 12:59:47)
[GCC 9.4.0]
OS Linux 5.15.0-60-generic (#66~20.04.1-Ubuntu SMP Wed Jan 25 09:41:30 UTC 2023)
CPU x86_64 (x86_64)
Bug reports: https://bugs.chromium.org/p/gerrit/issues/entry?template=Repo+tool+issue
```

2.2 Install Python dependencies required by the CAmkES build toolchain:

$ pip3 install --user camkes-deps
# Currently seL4 duplicate dependencies for python2 and python3 as a python3 upgrade is in process
$ pip install --user camkes-deps

2.3 The CAmkES build toolchain additionally requires Haskell. You can install the Haskell stack on your distribution by running:

$ sudo apt-get install haskell-stack



For the above figure: It seems that there is an issue with downloading the Haskell build plan for the stack tool. The error message indicates that the stack tool is trying to download a specific version of the build plan file (lts-19.12.yaml), but it cannot find it.

One possible reason for this could be that the build plan file has been moved or deleted from the source repository. You can try updating the stack tool and retrying the build process to see if that resolves the issue: **$ sudo stack upgrade**

2.4 Install build dependencies:

$ sudo apt-get install clang gdb
$ sudo apt-get install libssl-dev libclang-dev libcunit1-dev libsqlite3-dev
$ sudo apt-get install qemu-kvm

Step 3: Fetching, Configuring and Building seL4test on QEMU

This section presents a case study, by the end of which you can run seL4test on a simulator (QEMU). To build a project, you need to:

- check out the sources using Repo,
- configure a target build using CMake,
- build the project using Ninja.

3.1 Use repo to check sel4test out from GitHub. Its manifest is located in the sel4test-manifest repository:

$ mkdir seL4test
$ cd seL4test
$ repo init -u https://github.com/seL4/sel4test-manifest.git

```
shafiqul@shafiqul-VirtualBox:~/seL4test$ repo init -u https://github.com/seL4/sel4test-manifest.git
Downloading Repo source from https://gerrit.googlesource.com/git-repo
repo: Updating release signing keys to keyset ver 2.3

Traceback (most recent call last):
  File "/home/shafiqul/seL4test/.repo/repo/main.py", line 874, in <module>
    _Main(sys.argv[1:])
  File "/home/shafiqul/seL4test/.repo/repo/main.py", line 850, in _Main
    result = repo._Run(name, gopts, argv) or 0
  File "/home/shafiqul/seL4test/.repo/repo/main.py", line 294, in _Run
    result = run()
  File "/home/shafiqul/seL4test/.repo/repo/main.py", line 275, in <lambda>
    lambda: self._RunLong(name, gopts, argv, git_trace2_event_log) or 0
  File "/home/shafiqul/seL4test/.repo/repo/main.py", line 442, in _RunLong
    execute_command()
  File "/home/shafiqul/seL4test/.repo/repo/main.py", line 408, in execute_command
    execute_command_helper()
  File "/home/shafiqul/seL4test/.repo/repo/main.py", line 374, in execute_command_helper
    result = cmd.Execute(copts, cargs)
  File "/home/shafiqul/seL4test/.repo/repo/subcmds/init.py", line 395, in Execute
    self._ConfigureUser(opt)
  File "/home/shafiqul/seL4test/.repo/repo/subcmds/init.py", line 212, in _ConfigureUser
    name = self._Prompt("Your Name", mp.UserName)
  File "/home/shafiqul/seL4test/.repo/repo/project.py", line 778, in UserName
    self._LoadUserIdentity()
  File "/home/shafiqul/seL4test/.repo/repo/project.py", line 791, in _LoadUserIdentity
    u = self.bare_git.var("GIT_COMMITTER_IDENT")
  File "/home/shafiqul/seL4test/.repo/repo/project.py", line 3803, in runner
    p.Wait()
  File "/home/shafiqul/seL4test/.repo/repo/git_command.py", line 553, in Wait
    self.VerifyCommand()
  File "/home/shafiqul/seL4test/.repo/repo/git_command.py", line 543, in VerifyCommand
    raise GitCommandError(
git_command.GitCommandError: GitCommandError: 'var GIT_COMMITTER_IDENT' on manifests failed
stderr:
*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
```

You might see this error while running this command:

$repo init -u https://github.com/seL4/sel4test-manifest.git

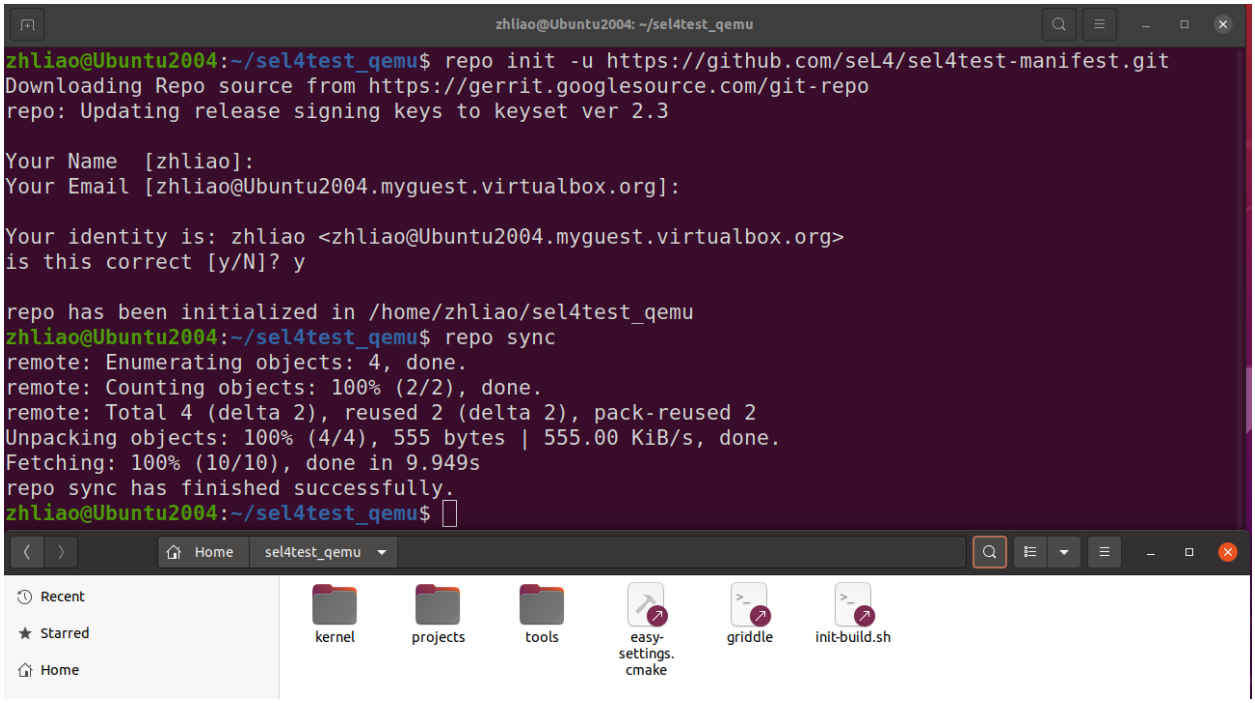
To resolve this issue run the commands with your email and name.

```
Run
  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"
```

Then again run this command

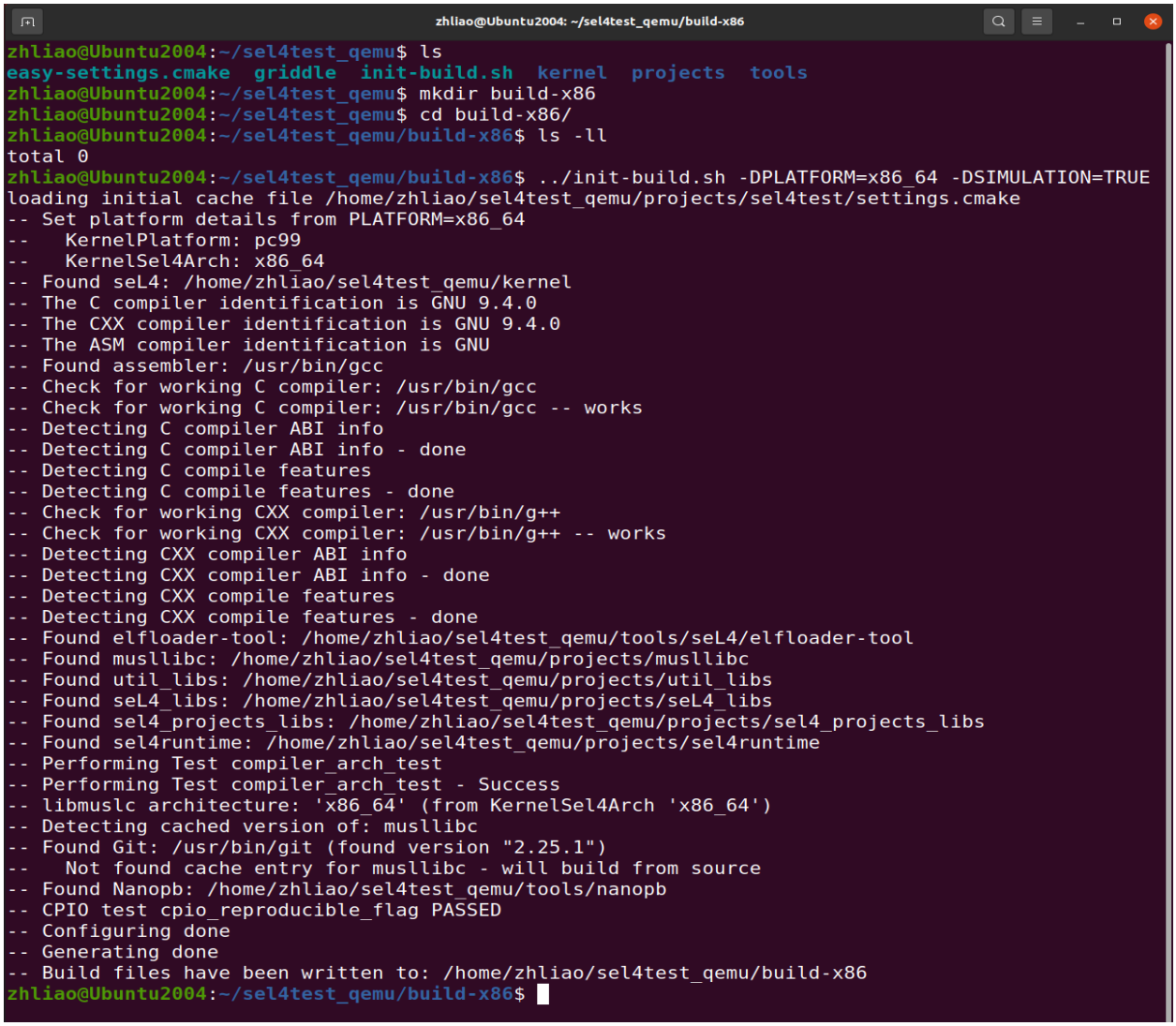
$repo init -u https://github.com/seL4/sel4test-manifest.git

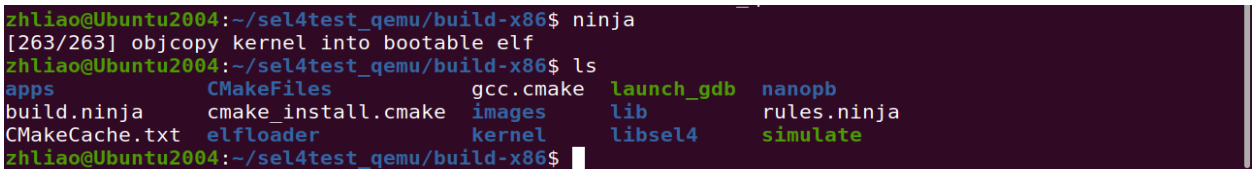
$ repo sync



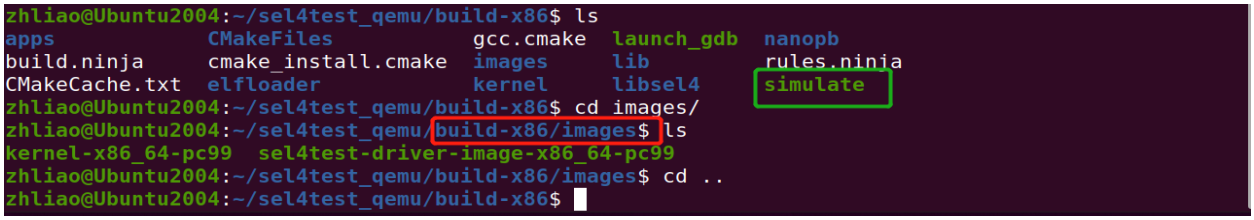
3.2 Configure an x86_64 build directory, with a simulation target to be run by QEMU. QEMU is a generic and open-source machine emulator and virtualizer, and can emulate different architectures on different systems.

$ mkdir build-x86
$ cd build-x86
$ ../init-build.sh -DPLATFORM=x86_64 -DSIMULATION=TRUE
$ ninja



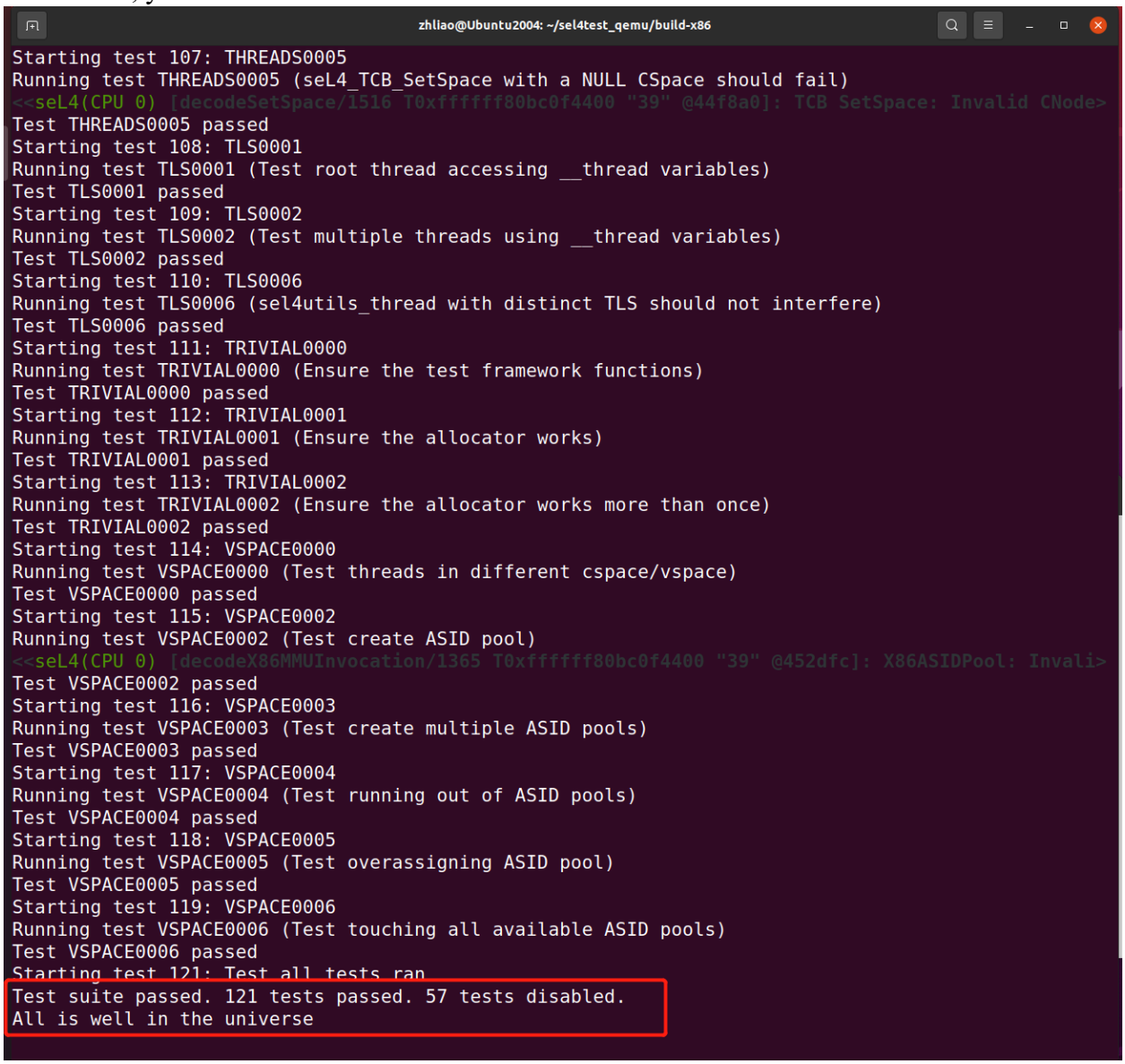


3.3 The build images are available in build-x86/images, and a script build-x86/simulate that will run Qemu with the correct arguments to run seL4test.



Run simulate as follows:
$ ./simulate

On success, you should see:



Then, press “Ctrl” + “A” and then press “X” in your terminal, you will exit QEMU.

Congratulations! You have successfully ported seL4 to QEMU! Next, you can develop applications on it and test them.

References:

1. seL4 Building for the BeagleBone Black https://docs.sel4.systems/Hardware/Beaglebone.html

2. seL4 Docs: Getting Started https://docs.sel4.systems/GettingStarted#getting-cross-compilers

3. seL4 Docs: Host Dependencies https://docs.sel4.systems/projects/buildsystem/host-dependencies.html

4. seL4 Docs: seL4Test https://docs.sel4.systems/projects/sel4test/

5. gtkterm https://postimg.cc/VSHvWJQj

6. seL4 and RefOS on BBB https://mokshasoft.com/seL4/seL4-beaglebone-black.html

7. Booting sequence of Beaglebone Black hardware https://fastbitlab.com/linux-device-driver-programming-lecture-6-booting-sequence-beaglebone-black-hardware/

8. http://qdosmsq.dunbar-it.co.uk/blog/2015/09/getting-arduino-working-from-a-windows-7-virtualbox-guest/

9. Cp2102 Usb-to-Serial Driver Installation https://exploreembedded.com/wiki/Cp2102_Usb-to-Serial_Driver_Installation

10. Linux Device Driver Programming Lecture 6- Booting sequence of Beaglebone Black hardware https://fastbitlab.com/linux-device-driver-programming-lecture-6-booting-sequence-beaglebone-black-hardware/

11. Linux device driver lecture 3 : Beaglebone black boot sequence https://www.youtube.com/watch?v=uJWWzdsL4tA

12. Minicom no cmd on start up https://stackoverflow.com/questions/16672829/minicom-no-cmd-on-start-up

13. fatload command https://u-boot.readthedocs.io/en/v2022.04/usage/cmd/fatload.html

14. Linux / UNIX minicom Serial Communication Program https://www.cyberciti.biz/tips/connect-soekris-single-board-computer-using-minicom.html