**OS/161**

OS/161 is an instructional operating system, sys161 is a simulated execution environment designed for IS/161.

* **workstation simulator**[sys161]: the simulated workstation is implemented in System/161 and includes a MIPS processor (R3000), main memory, and a collection of devices including a timer, disk(s), a network interface, and input and output consoles.
* **operating system**[os/161]: the OS/161 operating system runs on System/161 and manages the simulated workstation, and implements a set of system calls for user programs
* **user programs**: OS/161 user programs run on the simulated machine, and use services provided by the OS/161 operating system

This guide will help you to set up your OS/161 development environment (get the source codes and tool chain), understand the source code structure of OS/161, to be able to, modify, build and run your first OS/161 kernel using Git and GDB.

**Requirements**

* **OS/161**
  1. **source code** of the OS/161 operating system. You will be designing, implementing, running and debugging the OS.
  2. **a toolchain** for cross compiling and running your kernel, including a machine simulator called sys161.
* **Vagrant:** an open-source software product for building and maintaining portable virtual development environments
* **Git**: a distributed source control tool. You will need this to tracks your changes to OS/161.
* **GDB**: the GNU debugger which allows you to control and inspect your running system.

**Installation and Setup**

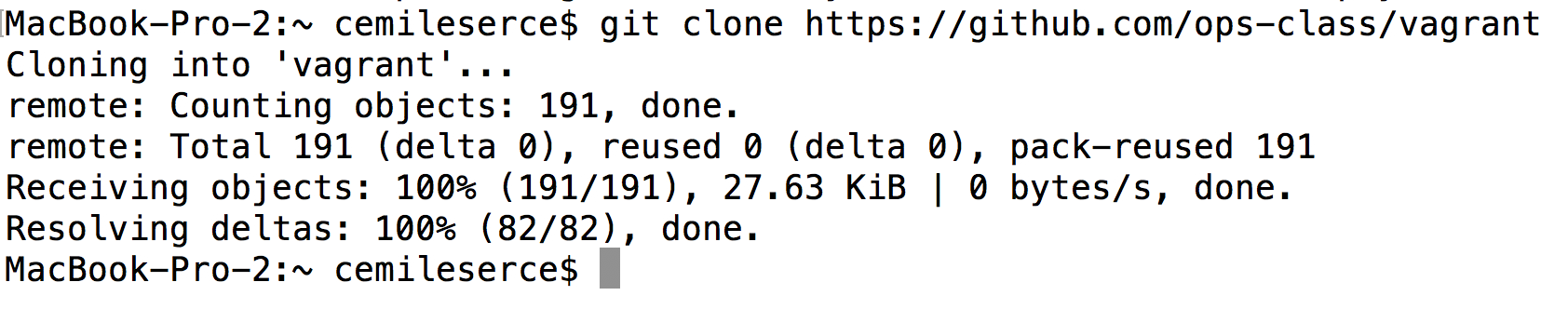
***-Vagrant-***

- Download and install Vagrant. <https://www.vagrantup.com/downloads.html>

***-Toolchain-***

* Clone vagrant file created for OS/161

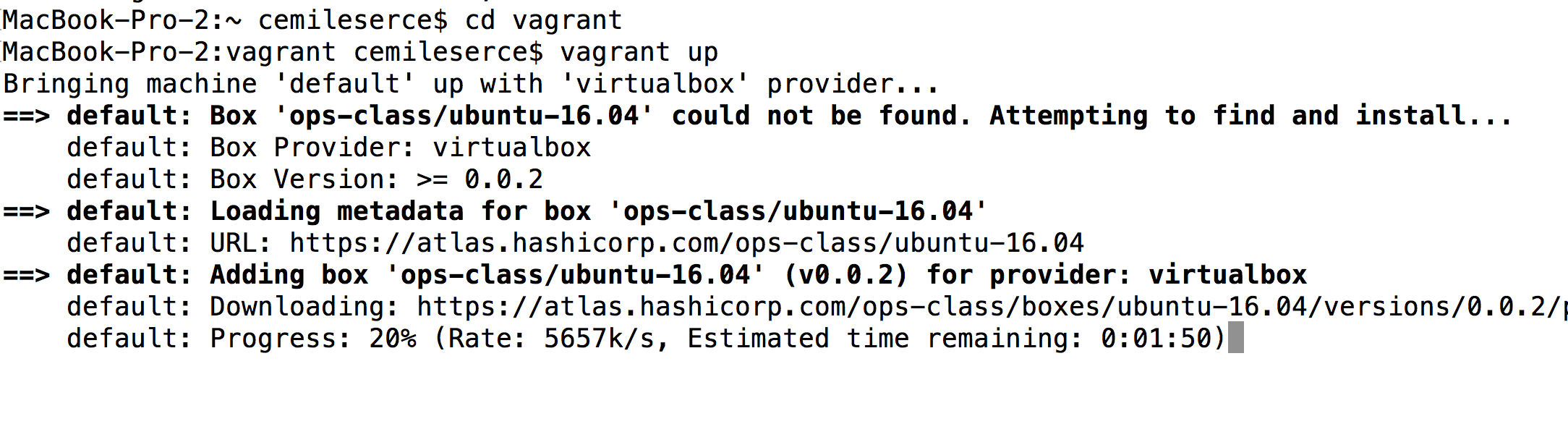
**git clone** [**https://github.com/ops-class/vagrant**](https://github.com/ops-class/vagrant)

****

* Use Vagrant file to boot a headless Ubuntu virtual machine with the toolchain installed.

**cd vagrant**

**vagrant up**

****

* The vagrant up may take a few minutes. Once it completes, you should be able to log in to the virtual machine and run the OS/161 tools:

**vagrant ssh**

**trinity@zion:~$ sys161**

***-OS/161 Source Code-***

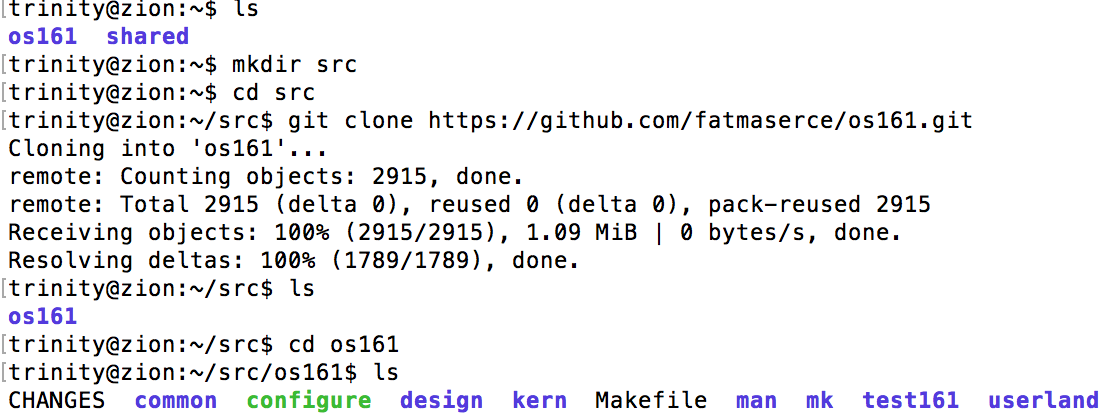
* Create a new folder/directory “src” inside trinity@zion [/home/trinity/src]

**mkdir src**

**cd src**

* Using git, clone the source code for the operating system, OS/161 [https://github.com/ops-class/os161]

**git clone** [**https://github.com/fatmaserce/os161.git**](https://github.com/fatmaserce/os161.git)

****

**Building the Kernel**

The kernel sources for OS/161 are in the kern subdirectory, which has its own configuration script. Change into kern/conf and look around. You should notice a configuration script, a base configuration file (conf.kern), and a couple of configurations that include conf.kern.

You should take a look at conf.kern and one of the configurations to get a sense of the format. But for now, the only thing we’re concerned about is ensuring that we enable dumbvm for this assignment.

There are three steps:

1. **Building the dependencies (**bmake depend**)**. This scans all of the source files that you have configured to be part of your kernel and ensures that all their header files are also included.
2. **Building the kernel (**bmake**)**. This generates your kernel binary.
3. **Installing the kernel (**bmake install**)**. This installs your kernel into the root directory that you configured above.

Run these three commands now and check that they complete successfully. Then change into your root directory and look around. You should see a fresh kernel. If you don’t, review the steps above until your kernel builds successfully.

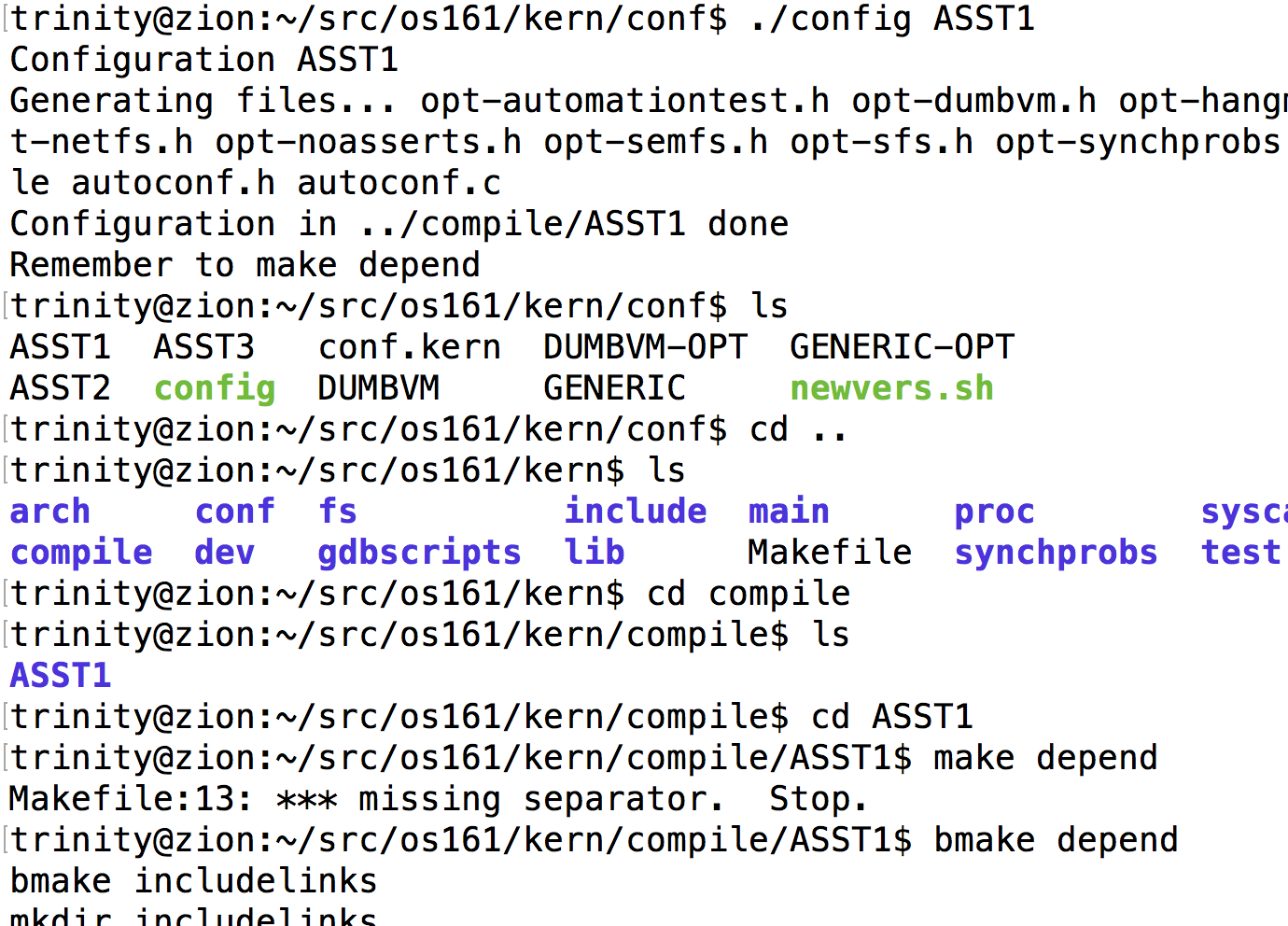
*Building the dependencies:*

**cd kern/conf**

**./config ASST1**

**cd ../compile/ASST1**

**bmake depend**

****

*Building the kernel:*

**bmake**

*Installing the kernel:*

**bmake install**

This will create the ASST0 build directory and then actually build a kernel in it.

*Build the user-level utilities*

* **cd ~/os161/src**
* **bmake**

**Configuring sys161 and Running the Kernel**

If you have made it this far, you have built and installed the entire OS. Now it is time to run it. But how? Given that your kernel doesn’t yet have any useful features, it would be impossible to run it on real hardware, or even in a fully-featured VM like VirtualBox. Instead, OS/161 kernels are built to be run by a special-purpose system simulator called sys161. Compared to other VMs or full-system simulators, sys161 is simpler and faster but retains enough realism to enable OS/161 kernel development. Apart from floating point support and certain issues relating to RAM cache management, it provides an accurate emulation of a MIPS processor.

* Download the sample **sys161.conf** and install it as ~/os161/root/sys161.conf.

<https://www.ops-class.org/files/sys161.conf>

Copy file into vagrant folder.

**ls /vagrant**

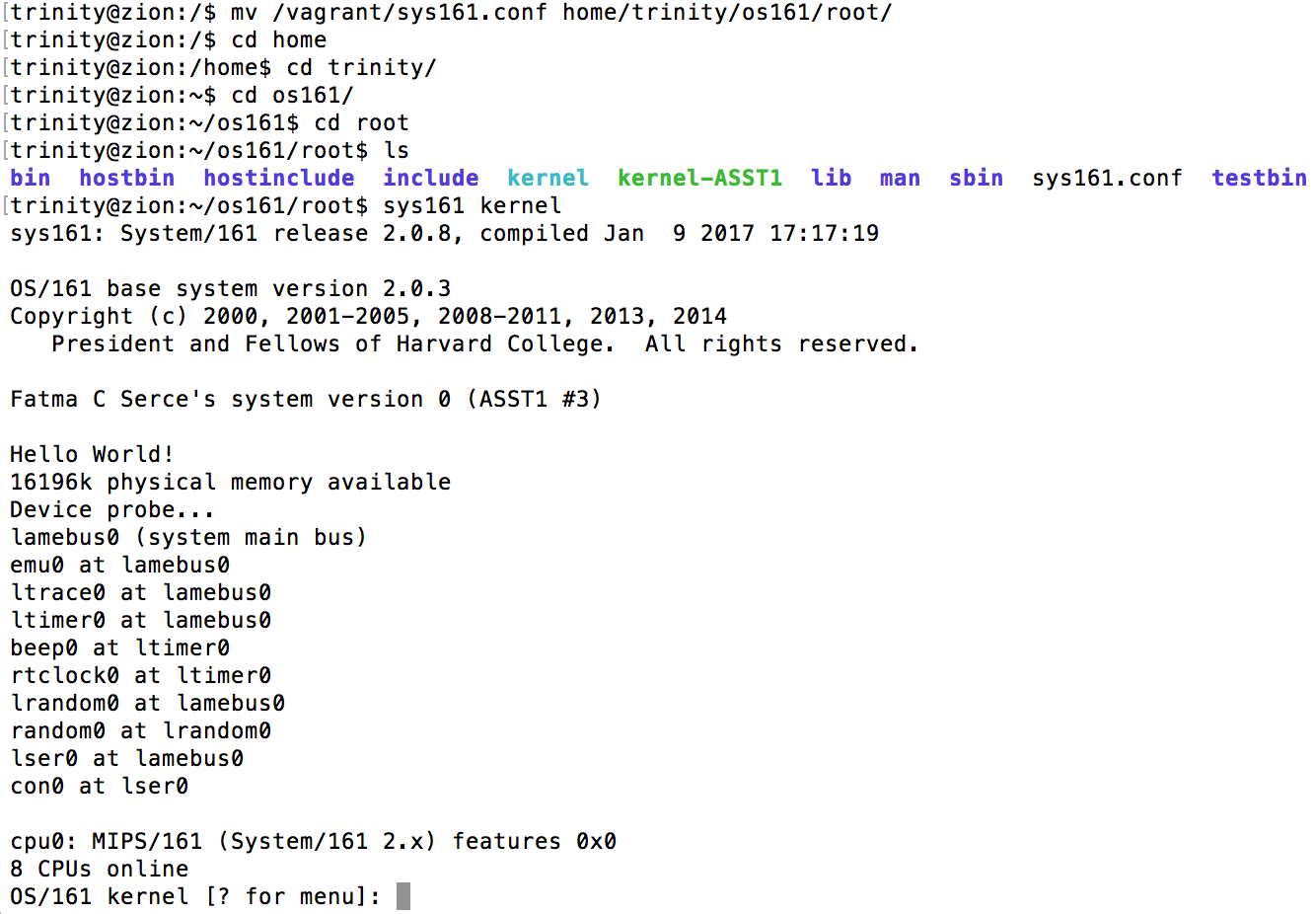
**mv /vagrant/sys161.conf home/trinity/os161/root/**

* Change to the root directory of your OS.

**cd ~/os161/root**

* Now run system/161 (the machine simulator) on your kernel.

**sys161 kernel**

****

* Power of the machine by typing q at the menu prompt.

**References:**

<https://www.ops-class.org/asst/0/>

<https://github.com/ops-class/os161>

<https://www.student.cs.uwaterloo.ca/~cs350/common/WorkingWith161.html>

http://os161.eecs.harvard.edu/resources/building.html