Instruction for Test Remote Servers

last update: September 15, 2025

This document is written for CSCI 5451 Introduction to parallel computing. It provides a quick instruction about how to use the package `test.zip` to test OpenMP, MPI, PThreads and Cuda on the Cuda and Plate Clusters server. It is highly recommended to run the tests on the clusters before starting working on the homework.

Step 0: Basic Knowledge about the Clusters For CSCI 5451, we will use the Cuda and Plate clusters for different homework tasks. Check the following table for a brief summary of the clusters. For each homework assignment, the homework description will let you know which cluster you should be attaching to.

Cluster	Nodes	Server Names	Tasks
Cuda	5	**csel-cuda-01.cselabs.umn.edu** to **csel-cuda-05.cselabs.umn.edu**	Cuda, NCCL
Plate	4	**csel-plate01.cselabs.umn.edu** to **csel-plate04.cselabs.umn.edu**	OpenMP, MPI

To work efficiently on the remote server, we strongly recommend using **VS Code** together with the **Remote SSH** extension. For more details, please refer to this page.

Step 1: Cluster Assignment In order to ensure that each of the servers above is used approximately equally, please refer to the table below. Students should only use the server listed in the column corresponding to the first letter(s) of their last name. For example, if your last name is Nelson, you should use the servers in column 03: cuda-03 (L–N) and plate03 (Mo–So). If your last name is Hancock, then you should use cuda-02 (H-K), and plate01 (A-Ha).

	01	02	03	04	05
cuda-	A-G	H-K	L-N	P-S	T-Z
plate	A-Ha	He-Me	Mo-So	St-Z	

Step 2: Connect to the Clusters First, verify your network connectivity with the machines in the target clusters using the following command:

```
ping csel-cuda-01.cselabs.umn.edu
# Replace with the appropriate server name for other clusters if needed
```

If the connection is successful, the server will respond promptly. Once the network connectivity is confirmed, you can establish an SSH connection to the server with the following command:

```
ssh your_x500_id@csel-cuda-01.cselabs.umn.edu
# Put your_x500_id as the user name here.
```

You may need to provide your UMN password to log in the machine.

Step 3: Prepare the Test File on Remote Server First, transfer the `test.zip` file to the remote machine using one of the methods below. These are the most common approaches, but you may use any other method you prefer.

- **Using SCP**: Open a **new terminal** on your local machine and run the following command to upload `test.zip` to the remote server:

```
scp path/on/your/local/machine/to/test.zip \
    your_x500_id@csel-cuda-01.cselabs.umn.edu:~/
# You need to modify this and put the path and your_x500_id accordingly.
```

- **Using VS Code**: After logging into the remote server via VS Code, simply drag and drop `test.zip` into your home directory on the server.

Once the file is placed in your home directory on one server, you do **NOT** need to repeat this step for the others, since all remote servers share the same storage system.

After you finished the transferring process, you may use the following command to unzip the files on the remote server:

```
unzip test.zip
```

This command will create a folder called `test_device`. Then you may move into that folder using

```
cd test_device
```

Step 4: Perform Test Based on the task allocations across the clusters, we will run **three tests** on the **Plate Cluster** and **one test** on the **Cuda Cluster**.

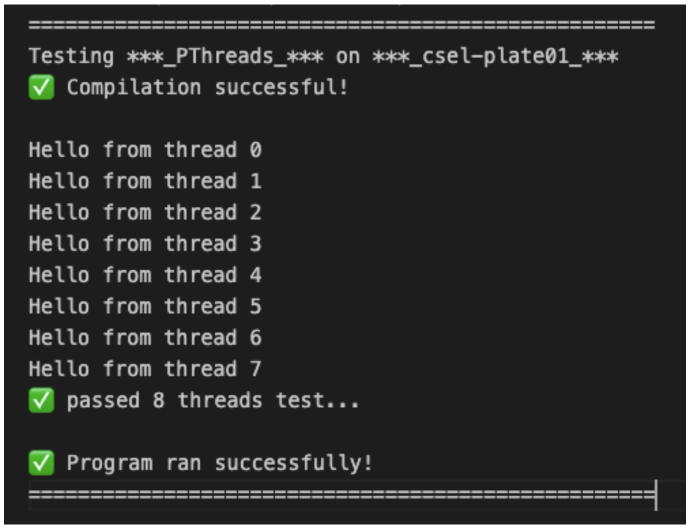
- On **Plate Cluster**, run:

```
bash test_mpi.sh
bash test_openmp.sh
bash test_pthreads.sh
```

- On **Cuda Cluster**, run:

```
bash test_cuda.sh
```

For each bash file, we will first compile the code and then run the 'hello world' test. Below is an example of the expected terminal output when the `test_pthreads.sh` script passes successfully.



```
==================================================
Testing ***_PThreads_*** on ***_csel-plate01_***
✅ Compilation successful!

Hello from thread 0
Hello from thread 1
Hello from thread 2
Hello from thread 3
Hello from thread 4
Hello from thread 5
Hello from thread 6
Hello from thread 7
✅ passed 8 threads test...

✅ Program ran successfully!
==================================================
```

*** Note that for some cases the output may not be ordered.**

We also offered example outputs of all the test cases, please refer to the `*.txt` files located in the `passed_examples` folder.

After completing all the steps in this tutorial, you are fully prepared to start Homework 1! Good luck ☺!