SEATTLE Lab: Distributed ping and NAT exploration

This Lab is an introduction to Seattle. You will use Seattle to measure the delays between pairs of computers distributed around the world. In the first part of the lab, you will familiarize yourself with Seattle. In the second part, you will run ping among computers distributed around the world. In the third part, you'll be investigating how to determine whether or not a computer is behind a NAT (network address translator).

What is Seattle?

Seattle is a platform for networking and distributed systems research. It's free, community-driven, and offers a large deployment of computers spread across the world. Users who choose to install Seattle share the resources on their computer with other users. This makes it possible for you to obtain resources from other computers (typically up to 10) distributed around the world.

Because Seattle nodes are distributed around the world, you have the ability to use it in application contexts that include cloud computing, peer-to-peer networking, mobile computing, and distributed systems.

Users install and run Seattle with minimal impact on system security and performance. Sandboxes are established on the user's computer to limit the consumption of resources (such as CPU, memory, storage space, and network bandwidth) by other Seattle nodes. In particular, programs installed by other Seattle nodes on your computer are only allowed to operate inside of a sandbox, ensuring that your files and programs on your computer are kept private and safe.

Getting Started with Seattle

You first need to install the Seattle client on your computer so that you can access resources on other computers. (This does not make your computer available to other Seattle users, it merely allows you to access resources on users that have chosen to make them available.) Installing the Seattle client will allow you to obtain resources from other computers (typically up to 10) distributed around the world.

To acquire the Seattle client, you first need to register at the Seattle Clearinghouse, https://seattleclearinghouse.poly.edu/html/register and then log-in. After logging in, click on the my vessels tab. **Under number of vessels select 10, and under environment select WAN** and then click 'get'. This will give you access to 10 virtual machines (vessels) on 10 different computers distributed around the world.

Next go to the profile tab and download the toolkit. The toolkit contains the Seattle client among other files and programs. Unpack the zip file to a folder of your choosing on your computer. Unpacking the zip file will create a directory that contains the Seattle shell (i.e., the client) called 'seash.py' as well as other files you will need for this assignment. Download your public and private key from the same webpage where you unzipped the demokit/toolkit.

On Windows first open the Command Prompt (i.e., run cmd.exe) and then change directories to the folder where you unzipped the demokit. On Mac / Linux, open a terminal window and change to the directory where you unzipped the demokit.

Then start the Seattle client by typing "python seash.py" or "seash.py". You will see a prompt like "!> " once seash starts.

At that prompt you then type "loadkeys *username*", with username replaced with your login name with the Clearinghouse, e.g.,

!> loadkeys johnsmith

Then type

!> as *username*

You will see "*username*@ !> " which indicates you are now acting with your credentials.

Now type "browse" to see the vessels that you have access to:

johnsmith@ !> browse

The output from the browse command will look something like the following:

johnsmith@ !> browse

Added targets: %1(IP:port:vesselname), %2(IP:port:vesselname), %3(IP:port:vesselname), ...

Added group 'browsegood' with X targets

Running a Simple Application on Another Computer

You will now upload a "hello world" program to one of your vessels (e.g., vessel 5) and execute that program on vessel 5. This program simply prints "hello world" to the log file on vessel 5. Uploading and executing is accomplished with the Seattle run command:

johnsmith@ !> on %5 run helloworld.repy

To see the results of this command type

johnsmith@ !> on %5 show log

To activate all the vessels in our browsegood group, type

johnsmith@ !> on browsegood

To run "hello world" on all of these vessels, type:

johnsmith@browsegood !> run helloworld.repy

johnsmith@browsegood !> show log

NOTE: You can use use "list" to show the ip addresses of all the vessels and "show location" to show the physical locations of all the vessels. Try typing both of these commands. Take screen shots and hand in.

UDP Ping

Now you are ready to measure the connectivity between computers. You will need to check which UDP port you should use. For convenience, Seattle Clearinghouse assigns the same UDP port to all of a user's vessels. To figure out which port you should use, look in the profile tab on the Clearinghouse Website that you logged into. **You should write down your port number as you'll need this value repeatedly.**

We'll measure the connectivity between two vessels say %1 and %2 . You can also use list to learn the ip address of all the vessels.

johnsmith@browsegood !> list

Run udppingserver on %2 and udpping on %1 as shown below (replace port with your port number and IP_of_**%**2 with the IP address of vessel 2).

johnsmith@browsegood !> on **%**2 run udppingserver.repy *port*

johnsmith@browsegood !> on **%**1 run udpping.repy *IP_of_**%**2 port*

johnsmith@browsegood !> on **%**1 show log

johnsmith@browsegood !> on **%**2 show log

You will see that %1 was able to send UDP ping packets to %2 and that %2 received the ping packets. Now stop the udppingserver on vessel 2 by typing

johnsmith@browsegood !> on **%**2 stop

Sending pings between a group of computers

Write all the ip addresses to a local file called neighboriplist.txt as follows:

johnsmith@browsegood !> show ip to neighboriplist.txt

Then upload the file to all the vessels as follows:

username@browsegood !> upload neighboriplist.txt

Examine the connectivity between a pair of vessels by running a program called allpairsping.repy, replacing port with your port number.

johnsmith@browsegood !> run allpairsping.repy *port*

Finally, check the result of all of the nodes pinging each other by opening your web browser and pointing it at one of the vessels in your neighboriplist:port (for example, type in the address bar of your browser something like 129.15.78.31:63124). If you can't see this from a specific node, try to refresh a few times. If you still don't get a result, you should try to load the webpage of a different vessel. You should observe in your browser a connectivity table between all the nodes, like the one below. Take screenshot and hand in.

Latency information from 129.15.78.31

	198.82.160.238	129.15.78.31	128.8.126.79	128.111.52.63	130.83.244.166	169.229.50.15	202.125.215.12	202.112.28.100	155.225.2.71
198.82.160.238	0.40s	0.04s	0.01s	0.08s	0.10s	0.43s	0.45s	0.31s	Unknown
129.15.78.31	0.29s	0.40s	0.04s	0.04s	0.12s	0.05s	0.50s	0.38s	Unknown
128.8.126.79	0.49s	0.04s	0.00s	0.07s	0.09s	0.08s	0.24s	0.62s	Unknown
128.111.52.63	0.34s	0.04s	0.07s	0.00s	0.16s	0.01s	0.46s	0.23s	Unknown
130.83.244.166	0.10s	0.13s	0.09s	0.16s	0.00s	0.16s	0.32s	0.39s	Unknown
169.229.50.15	0.09s	0.07s	0.49s	0.01s	0.50s	0.00s	0.22s	0.60s	Unknown
202.125.215.12	0.34s	0.21s	0.43s	0.17s	0.60s	0.22s	0.00s	0.16s	Unknown
202.112.28.100	0.31s	0.28s	0.30s	0.23s	0.38s	0.87s	0.47s	0.00s	Unknown
155.225.2.71	No Data Reported								

Detecting presence of a network address translator (NAT)

Remove all the vessels that you have by clicking remove all in the my vessels tab on the main clearinghouse website. To do this, select WAN as environment and 1 in number of vessels and click “Get”. Now under seash at the "username@ !>" prompt, type:

johnsmith@ !> browse

You should see the IP of at least one WAN node. Write down the IP address of the WAN node assigned to browsegood. Write down the IP address of your computer. Start the pingserver on the WAN node with the following command in seash (replacing *port* with your port number):

johnsmith@ !> on %1

johnsmith@**%**1 !> run udppingserver.repy *port*

Ping the WAN node from your local computer by typing the following command at a terminal window (not in seash!) after you cd to the directory the demokit is in. Remember to replace *WANIP* with the WAN node's IP address and *port* with your port. If you are running this on a remote system, you should use seash and replace the python repy.py restrictions.allowallports portion of the command with 'run'.

```
python repy.py restrictions.allowallports udpping.repy WANIP port
```

Now look at the log of the WAN computer.

```
johnsmith@%1 !> show log
```

Stop the pingserver on the WAN node:

```
johnsmith@%1 !> stop
```

Start the ping server on the local computer by running the following command from the terminal window (not in seash!). Once again, if you are running this on a remote system, you should use seash and replace the python repy.py restrictions.allowallports portion of the command with 'run'.

```
python repy.py restrictions.allowallports udppingserver.repy port
```

Next ping the local node from the WAN node

```
johnsmith@%1 !> run udpping.repy localIP port
```

Then check the log by running

```
johnsmith@%1 !> show log
```

What to Hand in

a. Screenshots of IP addresses and locations in the "hello world" portion of the lab.
b. Attach the screenshot of the latency information table that you got when you ran allpairsping.
c. In the last part detecting the presence of a NAT, Is there a NAT present? Based on the results you obtained give reasons based which you conclude whether there is a NAT or not. Justify your answer. Include screen shots of the show log command on the WAN node (seash) and the local command prompt.
d. In the folder demokit, there should be a file allpairsping.repy. Open the file as text and give a brief overview of what exactly is happening here. You will have to refer to https://seattle.cs.washington.edu/wiki/PythonVsRepy and https://seattle.cs.washington.edu/wiki/RepyApi to answer this.