Local Networking Between Two BBBs

by Brandon Mills, Roksan Yazman, Justin Medwid
Last update: Dec 5, 2015

This document guides the user through:

1. Setting up BBBs to ping one another from the same computer.
2. Basic C code for sending UDP packets to hard-coded IP address.

Table of Contents

Note: Guide not yet tested in the SFU Surrey Linux Lab (SUR4080). Some changes may be needed.

Formatting:

1. Commands starting with `$` are Linux console commands on the host PC: `$ echo "Hello world!"`
2. Commands starting with `#` are Linux commands on the target (BeagleBone Black):
   ```
   # echo "Hello embedded world!"
   ```
3. Almost all commands are case sensitive.
4. When referring to “7.x” it is the short version of “192.168.7.x”
   When referring to “8.x” it is the short version of “192.168.8.x”

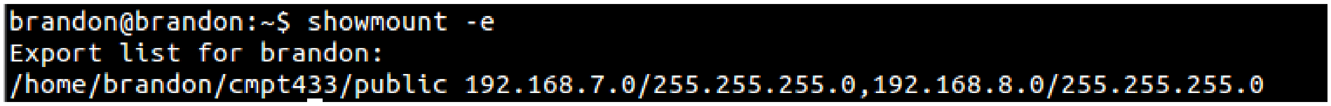
Revision History:

- None so far.

1. Target BBB Setup

1. First pick one BBB to change the host/target IP address of from 192.168.7.1/192.168.7.2 to 192.168.8.1/192.168.8.2. Once a BBB has been chosen and connected to the computer via micro-USB cable there are a few files to change in the BBB. These files and changes include:
   - At the end of the file /etc/network/interfaces change:
     - “address” from 7.2 to 8.2
     - “network” from 7.0 to 8.0
     - “gateway” from 7.1 to 8.1
   - In /opt/scripts/boot/am335x_evm.sh change:
     - 7.1 to 8.1 in 2 places (around where you see echo start and echo end)
     - 7.2 to 8.2 in the /sbin/ifconfig command (almost immediately below the first changes)
   - At the end of the file /etc/udhcpd.conf change:
     - “Start” and “End” from 7.1 to 8.1
   - If you have a script to add the default gateway (route add default gw 192.168.7.1) change:
     - From 7.1 to 8.1
   - In the mount NFS script/command change:
     - From 7.1 to 8.1
   - After all these files are changed reboot your BBB and run the add route command and mount NFS command (or updated scripts):
     - `# route add default gw 192.168.8.1`
     - `# busybox mount -o tcp -t nfs -o nolock 192.168.8.1:/home/user/cmpt433/public /mnt/remote`
       - Change “user” in path to your user name on the host.
       - This is one command not two.

2. After the BBB files have been changed there is still one host file that will need to be updated as well.
   - In /etc/exports add the line to end of the file:
     - "/home/user/cmpt433/public 192.168.8.0/255.255.255.0(rw,sync,no_subtree_check)"
   - Run the commands:
     - `$ sudo exportfs -ra`
     - `$ sudo /etc/init.d/nfs-kernel-server restart`
   - Can view the mount via:
     - `$ showmount -e command`
     - Expected output:



```
brandon@brandon:~$ showmount -e
Export list for brandon:
/home/brandon/cmpt433/public 192.168.7.0/255.255.255.0,192.168.8.0/255.255.255.0
```

3. To test and see if the new IP Address has correctly updated run:
   - From host:
     - `$ ping 192.168.8.2`
   - From target:
     - `# ping 192.168.8.1`

4. Troubleshooting
   - This is the step that is the easiest to mess up on since it requires changes to many files with the proper IP Address:
   - After running the mount NFS command, if you see:

```
Mount: 192.168.8.1:/home/user/cmpt433/public failed, reason given by server: Permission denied
Mount: mounting 192.168.8.1:/home/user/cmpt433/public on /mnt/remote failed: Bad file descriptor
```

     - This likely means that you have incorrectly added the line to the exports file from Step 2 above. Make sure to run “showmount –e” to verify.
   - Make sure if a command from above has the word “user” in the path that it should be replaced with the username on your host/target.

2. IP Address and Route Forwarding

1. After the target BBB has been set up there are a few things that will need to be done on the host, first off IP forwarding should be enabled using the command:In /etc/network/interfaces change:
   - `$ sudo sh –c “echo 1 > /proc/sys/net/ipv4/ip_forward”`
2. The next step is to add two routes to the file /etc/network/interfaces, this is crucial as it allows both of the BBB to route through the host to the other BBB. Add the following lines to the end of the file:
   - up route add -net 192.168.8.0 netmask 255.255.255.252 gw 192.168.8.2
   - up route add -net 192.168.7.0 netmask 255.255.255.252 gw 192.168.7.2

3. Connection and Ping

1. Open up 2 terminals and SSH into both of the BBBs via:
   - `$ ssh root@192.168.7.2` in terminal 1
   - `$ ssh root@192.168.8.2` in terminal 2

2. Check if forwarding is working by pinging targets via:
   - From 8.2: `ping $ 192.168.7.2`
   - From 7.2: `ping $ 192.168.8.2`

4. UDP Packet Sending Code

- Here are two functions that can be called to continuously send a “ping” message to the receiverIP which can be passed in as either 192.168.7.2 or 192.168.8.2 and receive message to whichever IP the socket was bound to (This depending on which target you run the code from).
- This code assumes that a UDP socket has already been set up.

```
void sendPackets(char* receiverIP)
{
               int BUFFER_LENGTH = 1024;
               char responseMessageBuffer[BUFFER_LENGTH] = "ping"
               struct sockaddr_in addrDest;
               memset((char *) &addrDest, 0, sizeof(addrDest));
               addrDest.sin_family = AF_INET;
               addrDest.sin_port = htons(senderPort);
               addrDest.sin_addr.s_addr = inet_addr(receiverIP);

               while(true) {
                      sleep(1); //send every 1 second

                     int responseMessageLength = sendto(UDPSocket,
                                                        responseMessageBuffer,
                                                        strlen(responseMessageBuffer),
                                                        0,
                                                        (struct sockaddr*) &addrDest,
                                                        (socklen_t) sizeof(addrDest));

                     if (responseMessageLength == -1) {
                             die("Message sent failed.\n");
                     }

                     memset(responseMessageBuffer, 0, BUFFER_LENGTH * sizeof(*responseMessageBuffer));
               }
}

void listenForPackets()
{
       struct sockaddr_in receiverSocketAddress;
       int socketLength = sizeof(receiverSocketAddress);

       while(true) {
               char messageBuffer[BUFFER_LENGTH];
               fflush(stdout);
               int messageLength = recvfrom(UDPSocket,
                                            messageBuffer,
                                            BUFFER_LENGTH,
                                            0,
                                            (struct sockaddr *) &receiverSocketAddress,
                                            (socklen_t*) &socketLength);

               if (messageLength == -1) {
                      die("Message receive failed.\n");
               }

               memset(messageBuffer, 0, BUFFER_LENGTH * sizeof(*messageBuffer));
       }
}
```