Bridging Devices onto the OFELIA Testbed

David R Newman

December 10, 2013

1 Introduction

The OFELIA testbed makes it possible to bridge devices onto slices created via an island's Expedient site. This document will provide instructions on how to bridge both Linux and Windows devices.

OpenVPN is the best suited application for setting up an Ethernet bridge from a Linux virtual machine on the OFELIA testbed. It can be installed as a server package on most Linux distributions and as a client on most Linux and Windows distributions. OpenVPN's Linux instructions on how to do setup an Ethernet bridge using OpenVPN can be found at:

`http://openvpn.net/index.php/open-source/documentation/miscellaneous/76-ethernet-bridging.html`

This guide is an adaptation of these instructions for setting up an Ethernet bridge to an OFELIA island virtual machine. It assumes that you have already:

1. Created your own OFELIA account.
2. Connected to the OFELIA testbed network over VPN.
3. Created a project through an OFELIA island's Expedient website .
4. Added a slice to that project.
5. Created a couple of virtual machines on the OFELIA island's virtual machine servers.
6. Configured a flowspace connecting together these virtual machines with one or more switches.

Section 2 describes the project, test slice and flowspace used as an exemplar in this guide. If you need help getting to a similar point follow the instructions at:

`https://alpha.fp7-ofelia.eu/doc/index.php/Working_with_the_OFELIA_Control_Framework`

Section 3 describes how to generate the server-side configuration for Ethernet bridging and section 4 how to generate the client-side configuration.

Section 5 provides instructions on how to bridge first Linux and then Windows client devices. Section 5.2 as well as describing how to bridge a Windows device also explains how to initially connect to the OFELIA testbed network over VPN.

Section 6 describes how to use the automated script to generate server and client configurations to save time.

Section 7 provides first some basic and then some more advanced testing to confirm the Ethernet bridge has been set up successfully.

2 OFELIA Test Slice Details

Testbed Island Create-Net (Trento, Italy)[1]

Project name OFERTIE-Southampton

Test slice name OFERTIE Test Slice

Flowspace [h1]–(s1)–(s2)–[h2]

VLAN ID 112

Virtual Machine 1

[1] See page 1 of `http://www.fp7-ofelia.eu/assets/IslandsinventoryPhaseIOpenCall.pdf`

Name h1

On server vm1

Management IP 10.216.33.103

Flowspace Interface eth2

OpenFlow controller port number: 6633

Virtual Machine 2

Name h2

On server vm2

Management IP 10.216.33.104

Flowspace Interface eth2

N.B. By default the root password on virtual machines is *openflow*.

3 Generating Server Configuration

The first step to setting up an Ethernet bridge is to generate server configuration. Typically, the OpenVPN package on your Linux distribution will provide a set of scripts called *easy-rsa* on Debian/Ubuntu these can be found in:

`/usr/share/doc/openvpn/examples/easy-rsa/`

If you cannot find these scripts you may need to do a web search to find out how to install them on your Linux distribution. To make use of these scripts, on one of the virtual machines on your project slice, copy this folder to your home folder. Then follow these instructions:

1. Edit /ofelia/users/*OFELIA-USERNAME*/easy-rsa/vars and update *KEY_COUNTRY*, *KEY_PROVINCE*, *KEY_CITY*, *KEY_ORG* and *KEY_EMAIL*. For the project slice described in section 2. The following values were set:

   ```
   export KEY_COUNTRY="IT"
   export KEY_PROVINCE="Trentino"
   export KEY_CITY="Trento"
   export KEY_ORG="CREATE-NET"
   export KEY_EMAIL="username@your.domain"
   ```

2. From your copy of the *easy-rsa* folder, run the following commands to generate the server key, certificate and certificate authority:

   ```
   $ source ./vars
   $ sh ./clean-all
   $ sh ./build-dh
   $ sh ./pkitool --initca
   $ sh ./pkitool --server server
   $ cd keys
   $ /usr/sbin/openvpn --genkey --secret ta.key
   $ su root
   # cp server.crt server.key ca.crt dh1024.pem ta.key /etc/openvpn/
   ```

3. Uncompress the OpenVPN example server configuration to */etc/openvpn/*:

   ```
   # zcat /usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz > /etc/openvpn/server.conf
   ```

4. Edit */etc/openvpn/server.conf* ensure there is one uncommented *dev* and *server-bridge* lines that look as follows:

   ```
   dev tap0
   server-bridge 192.168.1.32 255.255.255.0 192.168.1.64 192.168.1.128
   ```

5. By default, the OpenVPN server will be setup not to allow connecting clients to talk to each other. If this is required, also uncomment (remove the leading ;) from the following line in */etc/openvpn/server.conf*:

   ```
   ;client-to-client
   ```

6. Copy *bridge-start* and *bridge-stop* scripts to */usr/local/bin/*:

```
# cp /usr/share/doc/openvpn/examples/sample-scripts/bridge-* /usr/local/bin/
```

7. Edit *bridge-start* and change *eth*, *eth_ip* and *eth_broadcast* to an interface that is not currently in use (and will not be used as part of your project slice's flowspace) and give it an IP address in the range *192.168.1.0/24*. E.g.

```
eth="eth1"
eth_ip="192.168.1.32"
eth_broadcast="192.168.1.255"
```

8. Add the following *iptables* rules at the command line:

```
# iptables -A INPUT -i tap0 -j ACCEPT
# iptables -A INPUT -i br0 -j ACCEPT
# iptables -A FORWARD -i br0 -j ACCEPT
```

9. Save the *iptables* rules to a file:

```
# iptables-save > /etc/iptables.dat
```

10. Add an *iptables-restore* to */etc/rc.local* to restore rules on reboot:

```
iptables-restore < /etc/iptables.dat
```

11. Run the *bridge-start* script and start OpenVPN:

```
# /usr/local/bin/bridge-start
# service openvpn start
```

4 Generating Client Configuration

Still on the virtual machine configured in section 3 generate the client keyset:

```
$ cd /ofelia/users/OFELIA USERNAME/easy-rsa
$ source ./vars
$ ./build-key-pass client1
```

You will be prompted to set a PEM passphrase and then you will need to set the credentials for the client certificate. For the project slice described in section 2 the following credentials were used. (There is no need to set a *challenge password* or *optional company name*):

```
Country Name (2 letter code) [ES]:GB
State or Province Name (full name) [Catalonia]:Hampshire
Locality Name (eg, city) [Barcelona]:Southampton
Organization Name (eg, company) [i2CAT]:UniversityOfSouthampton
Organizational Unit Name (eg, section) []:ECS
Common Name (eg, your name or your server's hostname) [client1]:client1.ecs.soton.ac.uk
Name []:ECS Client
Email Address [HIDDEN@ecs.soton.ac.uk]:HIDDEN@ecs.soton.ac.uk
```

Now tarball up the client key, certificate and certificate signing request with the certificate authority certificate:

```
$ cd /ofelia/users/OFELIA-USERNAME/easy-rsa/keys
$ tar cvf ../../client1.tar client1.{crt,csr,key} ca.crt
```

This tarball can now be copied to the device you want to bridge onto the OFELIA testbed.

5 Setting Up Client Device

5.1 Linux Devices

1. Make sure the OpenVPN package is installed.
2. Copy the tarball for the client off the server and unpack in /etc/openvpn/:

```
# cd /etc/openvpn/
# scp OFELIA-USERNAME@10.216.33.103:client1.tar .
# tar xvf client1.tar
```

3. Copy the client configuration[2] to /etc/openvpn:

   ```
   # /usr/share/doc/openvpn/examples/sample-config-files/client.conf /etc/openvpn/ofelia_client.conf
   ```

4. Edit */etc/openvpn/ofelia_ client.conf* and replace with the following configuration, changing the *ca*, *cert* and *key* as appropriate based on the filenames you unpacked and *remote* to the IP address of your OFELIA virtual machine's 10.216.x.y address:

   ```
   client
   dev tap
   proto udp
   remote 10.216.33.103 1194
   resolv-retry infinite
   nobind
   persist-key
   persist-tun
   ca ca.crt
   cert client1.crt
   key client1.key
   ns-cert-type server
   comp-lzo
   verb 3
   mute 20
   ```

5. Run OpenVPN as root using this client configuration. (You will be prompted for a password as set as the *PEM passphrase* in section 4):

   ```
   # /usr/sbin/openvpn /etc/openvpn/ofelia_client.conf
   ```

Your bridge should now have been successfully set up. Refer to section 7 for instructions on how to test this.

5.2 Windows Devices

5.2.1 VPN Connection to OFELIA Testbed Network

If you have a Windows device that you have yet to connect to the OFELIA testbed network follow these instructions:

1. Download and install OpenVPN from `http://openvpn.net/index.php/download/community-downloads.html`. Make sure you download the Windows Installer for the appropriate type of operating system (either 32 or 64 bit). This guide has been tested using OpenVPN version 2.3.2.
2. Open the *Network and Sharing Center* through the *Control Panel* and click on *Manage Adapter Settings* in the left-hand menu bar.
3. Right-click on the the *Local Area Connection* that is described underneath as *TAP-Windows Adapter V9* and then click on *Properties* in the menu that appears.
4. Click on the *Internet Protocol Version 4 (TCP/IPv4)* listed option, being careful not to un-tick the adjacent tick box. Then click on the *Properties* button.
5. Change from *Obtain DNS server address automatically* to *Use the following DNS server addresses* and enter *10.216.24.2* for *Preferred DNS server*. Then click *OK* and then *Close* on the original network connections properties window.
6. Run *Windows Explorer* as administrator. I.e. use the search bar in the start menu to find *Windows Explorer*, then right-click on it and select *Run as administrator*.
7. Navigate to

   ```
   C:\Program Files\OpenVPN\config\
   ```

8. Create a new *Text Document* named *ofelia.ovpn*, edit using *Notepad*, adding the following lines before saving:

   ```
   mode p2p
   topology subnet
   proto udp
   port 1194
   ```

[2] Location of client configuration may vary between Linux distributions. This guide references the location on Debian/Ubuntu distributions.

```
remote 157.193.215.150
dev tun
pull
script-security 2
tls-client
ca Ca-ofeliarouted.crt
auth-user-pass
ping 10
ping-restart 60
```

9. Go to `https://alpha.fp7-ofelia.eu/doc/index.php/VPN_setup` in a web browser, right click on the link to *Ca-ofeliarouted.crt* click on *Save Link As...* and save it.
10. In the *Windows Explorer* window you ran as administrator, copy the certificate file from where you downloaded it to *C:|Program Files|OpenVPN|config|*
11. Go to the Start menu and search for *cmd*. Under *Programs* it should find *cmd.exe*. Right click on this and select *Run as administrator* from the menu that appears.
12. In the black window that appears, type in the following commands:

    ```
    C:\Windows\systen32> cd "C:\Program Files\OpenVPN\config"
    C:\Progran Files\OpenVPN\config> ..\bin\openvpn.exe ofelia.ovpn
    ```

    If the connection is successful the last message displayed in this black window that appears will be (preceded by a timestamp):

    ```
    Initialization Sequence Completed
    ```

13. To test that connection is working as expected, try connection to the Expedient site, e.g. `https://exp.create-net.fp7-ofelia.eu/`. If the connection was successful you will be asked to authorise a self-signed SSL certificate and then will be taken to a login screen with the message *Welcome to Expedient*.

5.2.2 Setting up Ethernet Bridge

Before you can set up the Ethernet bridge you need to create a second TAP interface, which OpenVPN can use to connect to the OpenVPN server running on the OFELIA virtual machine. If you are running a 64-bit version of Windows 7 you will need to run the program in Windows compatibility mode. To do this:

1. Load up the *Control Panel* and then in the top right had corner do a search for *troubleshooter*.
2. Then click on *Troubleshooting* under the list of results returned.
3. In the window that appears click on *Next*. Then from the list presented click on *Not Listed* and click *Next* again.
4. Then type in *C:|Program Files|TAP-Windows|bin|devcon.exe* and click on *Next*.
5. In the next window, tick *The program worked in earlier versions of Windows but won't install or run now* and *The program requires additional permissions* tick boxes and click *Next*.
6. Now, select the *Windows Vista* option and click *Next*.
7. Then click on *Start the program...*, a black windows will briefly appear. Once this disappears click *Next*.
8. Finally, click on *Yes, save these settings for this program*. You the see a final window saying *Troubleshooting has completed* from which you can click on the close button.

Now, to create a second TAP interface go to the start menu and run *Add a new TAP virtual Ethernet adapter* from the *Utilities* sub-folder in the *TAP-Windows* folder that can be found under *All Programs*. A black window will appear. If you are on a 64-bit system, you may be prompted with a window to confirm you are happy to run in compatibility mode. If the interface installs successfully, if you go to *Manage Adapter Settings* (see step 2 in section 5.2.1), you should see a new *Local Area Connection* with the type *TAP-Windows Adapter V9 #2*.

Now, you can setup the VPN connection to the virtual machine on your OFELIA project slice:

1. First, make sure that the VPN connection to OFELIA is active. You can use the "Can I access an OFELIA island's Expedient site?" test as described in the last step in section 5.2.1.

2. Now download the tarball with client configuration you created in section 4. WinSCP[3] is a suitable client for doing this. When installing WinSCP, if prompted, select *Norton Commander* as the style of user interface you want.
3. Start WinSCP and when configuring your login session set the *Host name* as the IP address of the OFELIA virtual machine you are running as the VPN server, the *User name* and *Password* as your OFELIA username and password. Once you have filled in all these details, click on the *Login* button.
4. If WinSCP connects successfully navigate to the directory where you saved you client OpenVPN configuration in the right-hand pane. If you are using the automated script that will be */tmp/openvpn*. Then click on the tarball you want to copy and press the F5 key.
5. Once the tarball is downloaded you will need to unpack it. WinRAR[4] or 7-Zip[5] are suitable tools for doing this..
6. Once unpacked, again using your *Windows Explorer* window running as administrator copy all the files you unpacked from the tarball into the *C:\Program Files\OpenVPN\config* folder.
7. Create a new text document in the same folder called ofelia-island.ovpn and copy the following lines into it, changing the *ca*, *cert* and *key* as appropriate based on the filenames you unpacked and *remote* to the IP address of your OFELIA virtual machine's 10.216.x.y address:

```
client
dev tap
proto udp
remote 10.216.33.103 1194
resolv-retry infinite
nobind
persist-key
persist-tun
ca ca.crt
cert client1.crt
key client1.key
ns-cert-type server
comp-lzo
verb 3
mute 20
route-method exe
route-delay 5
reneg-sec 0
```

8. After saving this file, go to the Start menu and search for *cmd*. Under *Programs* it should find *cmd.exe*. Right click on this and select *Run as administrator* from the menu that appears.
9. In the black window that appears, type in the following commands:

```
C:\Windows\systen32> cd "C:\Program Files\OpenVPN\config"
C:\Progran Files\OpenVPN\config> ..\bin\openvpn.exe ofelia-island.conf
```

   If the connection is successful the last message displayed in this black window that appears will be (preceded by a timestamp):

```
Initialization Sequence Completed
```

Your bridge should now have been successfully set up. Refer to section 7 for instructions on how to test this.

6 Setting up Ethernet Bridge Using Automated Script

To save having to manually type in all the commands described in sections 3 to 5, an accompanying automated script for this setup has been written. It can be downloaded from:

`https://raw.github.com/drn05r/ofertie-scripts/master/ofelia_tap/setup-tap`

The following instructions explain how to make use of this script:

[3] http://winscp.net/eng/index.php
[4] http://www.win-rar.com/download.html
[5] http://www.7-zip.org/

1. Copy the script to your home folder on one of the virtual machines you have created for your project slice.
2. Modify the scripts file permissions so it can executed:

   ```
   $ chmod +x setup-tap
   ```

3. Modify the variables at the start of the file. If you are creating a bridge on a newly created project slice virtual machine, all you have to replace the C and D in the *VPN_SERVER_IP* with the 10.216.0.0/16 address of the virtual machine. However, you should also make sure *BRIDGE_IFACE* is a defined but unrequired interface on the virtual machine. I.e. check this interface to make sure it is not to be connected to one of the switches in your project slice flowspace.
4. Run the script (as root) in server mode:

   ```
   # ./setup-tap server
   ```

5. If the script complete successfully you used see the message:

   ```
   Server setup COMPLETE
   Now you need to run client setup
   ```

6. Now copy the script onto the client. The best way to do this is by using the following command from the client device (adapting as appropriate):

   ```
   $ scp OFELIA-USERNAME@10.216.33.103:setup-tap .
   ```

7. Run the script (as root) in client mode (on the client device):

   ```
   # ./setup-tap client
   ```

8. If this is successful you should get a message saying *CLIENT SETUP COMPLETE* followed by some instructions on how to start the bridge.
9. Back on the virtual machine run the following command as root:

   ```
   # bridge-start ; service openvpn start
   ```

10. On the client device run the following command as root:

    ```
    # /usr/sbin/openvpn  /etc/openvpn/client1/client1.conf
    ```

Your bridge should now have been successfully set up refer to section 7 for instructions on how to test this.

6.1 Creating Additional Client Configurations

If you wish to create additional client configurations for more devices, you need to do the following:

1. Edit the script on the server, changing the values for *CLIENT* and *CLIENT_LOCAL*. You might typically change these from *client1* to *client2*.
2. Run the script in *clientconf* mode:

   ```
   # ./setup-tap clientconf
   ```

3. Copy the setup script to you second client device and run in client mode like before:

   ```
   # ./setup-tap client
   ```

4. On the client device run the following command as root:

   ```
   # /usr/sbin/openvpn  /etc/openvpn/client2/client2.conf
   ```

7 Testing Ethernet Bridge Setup

7.1 Basic Testing

You can use the *mtr* command to perform a basic test on whether the Ethernet bridge setup has been successful. This may take a while to complete (approximately 2 minutes).

```
mtr -r -c 100 192.168.1.32
```

Once finished you should see an output something like:

```
HOST: c           Loss%  Snt  Last  Avg   Best  Wrst   StDev
1.|-- 192.168.1.32   0.0%   100  55.2  55.5  54.4  109.4  5.5
```

7.2 Advanced Testing

To test that the Ethernet bridge has been configured as expected and that it will be suitable for bridging external devices for your use cases, there follows several use case scenarios demonstrating the setup and providing additional configuration where necessary. The scenarios assume you have already bridged your external devices onto the OFELIA testbed network.

7.2.1 Linux Game Server on an OFELIA Virtual Machine with an External Client

1. On the OFELIA Virtual Machine download a copy of the Open Transport Tycoon Deluxe (OpenTTD) DEB package:

   ```
   $ wget http://binaries.openttd.org/releases/1.3.2/openttd-1.3.2-linux-debian-squeeze-amd64.deb
   ```

2. As root install the following packages:

   ```
   # apt-get install liblzo2-2 libsdl1.2debian libasound2 libcaca0 libpulse0 libasyncns0 \
   libjson0 libsndfile1 libflac8 libogg0 libvorbis0a libvorbisenc2 openttd-data \
   openttd-opengfx openttd-openmsx openttd-opensfx
   ```

3. As root, install the OpenTTD DEB package using *dpkg*:

   ```
   # dpkg -i openttd-1.3.2-linux-debian-squeeze-amd64.deb
   ```

4. Change directory to where OpenTTD has been installed. This is typically */usr/share/games/*
5. Download the OFERTIE test configuration file for OpenTTD:

   ```
   # wget https://raw.github.com/drn05r/ofertie-scripts/master/ofelia_tap/openttd.cfg
   ```

6. Run OpenTTD in daemon mode using the following command:

   ```
   # /usr/games/openttd -D -c openttd.cfg
   ```

7. On the external client device, download the version of OpenTTD appropriate to the client's operating system from: `http://www.openttd.org/en/download-stable`
8. If your external client device runs Linux then install the same packages as those listed in step 2. You may need to figure out the equivalent packages for non Debian/Ubuntu versions of Linux.
9. If your external client device is running Debian/Ubuntu you can use the same *dpkg* command as in step 3 to install OpenTTD. If you are using another version of Linux refer to the instructions at: `http://wiki.openttd.org/Installation_FAQ`

   If you are installing on Windows, just follow the instructions of the automated MSI installer. There is no need to change any of the configuration options.
10. Once OpenTTD is installed on Linux it can be run from the command line, by just simply typing:

    ```
    $ /usr/games/openttd
    ```

On Windows, OpenTTD can be started by clicking on the shortcut icon on the desktop.

11. When you first start OpenTTD you might be prompted with a number of error messages, you can generally just dismiss these. After doing this to connect to the OpenTTD server click on the *Multiplayer* menu option. In the window that appears, check that *Connection* is set to *LAN* and then click the *Find Server* button. If the setup has been successful you should see a server called *OFERTIE* listed. Click on this server and then click on *Join game* at the bottom of the right-hand pane in this window to connect to the server game.

7.2.2 Linux Game Server on an External Device with an External Client connected through the same OFELIA Virtual Machine

Assuming you have followed the server setup instructions to allow client-to-client communication (step 5 in section 3) or used the automated script described in 6, all you should need to do is to create a second client configuration, as detailed in section 4 and deploy it on the external device you want to run as the Linux game server. If you have used the automated script, (see section 6), just change the value of the *CLIENT* and *CLIENT_LOCAL* variables near the beginning of the *setup-tap* before running the script in *clientconf* mode, (see section 6.1).

After you have generated and deployed this second client configuration, just follow the instructions in section 7.2.1 for installing OpenTTD in daemon mode on this second external device rather than the OFELIA virtual machine. You should be able to use the same process as described in section 7.2.1 to confirm that the setup has been successful.

7.2.3 Linux Game Server on an External Device with an External Client connected through a different OFELIA Virtual Machine

Figure 1 shows a network with two OFELIA virtual machines and two external devices, (a game server and a game client), bridged onto their own virtual machine over a VPN connection.

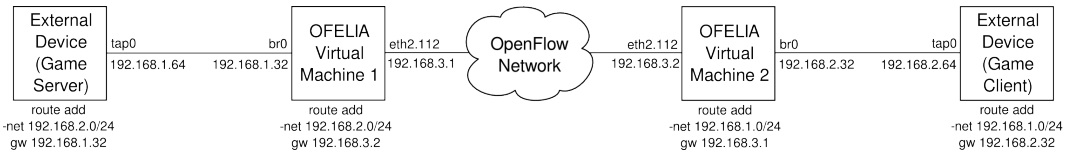

Figure 1: Network with two external devices bridged via different OFELIA virtual machines

To build this network, a VPN server on a second OFELIA virtual machine needs to be setup. This can be done by either following the server configuration instructions in section 3 or using the automated script as described in section 6. You will need to use a different subnet for the VPN server. If you are doing this manually, in step 4 in section 3, use the following configuration for */etc/openvpn/server.conf*:

```
dev tap0
server-bridge 192.168.2.32 255.255.255.0 192.168.2.64 192.168.2.128
```

In step 7 you will need to use following values in *bridge-start* remembering to ensure the eth value is for an interface that is not in use:

```
eth="eth1"
eth_ip="192.168.2.32"
eth_broadcast="192.168.2.255"
```

If you are using the automated script instead, all you need to do is change *BRIDGE_IP* and *BRIDGE_BCAST* to the following:

```
BRIDGE_IP="192.168.2.32"
BRIDGE_BCAST="192.168.2.255"
```

BRIDGE_IFACE may also need to be changed to a different interface that is not in use.

Once you have setup the VPN server and bridged an external device, if you have not already done so you need to connect the OFELIA virtual machines to the flowspace. Before you can do this you need to make sure, (as root), you have installed the *vlan* DEB package and enabled the *8021q* kernel module on both virtual machines:

```
# apt-get install vlan
# modprobe 8021q
```

As you can see from section 2 both virtual machines are joined to the flowspace on their *eth2* interfaces and the VLAN ID for the flowspace is *112*. Therefore run the following commands, (as root), to add and bring up the *eth2.112* interface, (substituting *X* for *1* for the OFELIA Virtual Machine 1 and *2* for OFELIA Virtual Machine 2):

```
# vconfig add eth2 112
# ifconfig eth2 up
# ifconfig eth2.112 192.168.3.X netmask 255.255.255.0 broadcast 192.168.3.255 up
```

Now you can route traffic between the two external devices. However, before defining these routes you will need to enable IPv4 forwarding on both OFELIA virtual machines and the external devices. On Debian/Ubuntu systems, this can be done by editing */etc/sysctl.conf* and uncommenting *net.ipv4.ip_forward*, making sure its value is set to 1. You will then need to restart the *procps* service using the following command as root:

```
/etc/init.d/procps restart
```

Now you can add the following route commands to each machine:

OFELIA Virtual Machine 1 `route add -net 192.168.2.0/24 gw 192.168.3.2`

OFELIA Virtual Machine 2 `route add -net 192.168.1.0/24 gw 1192.168.3.1`

External Device (Game Server) `route add -net 192.168.2.0/24 gw 192.168.1.32`

External Device (Game Client) `route add -net 192.168.1.0/24 gw 192.168.2.32`

To add these permanently, these commands also needs to be added to */etc/rc.local*, prior to *exit 0*. If you external devices are running Windows you will need to run a command prompt as administrator and type the following command, adapting as appropriate:

```
route ADD -p A.B.C.D/24 E.F.G.H
```

The *-p* option will add this route permanently, without this it will only be added until the device is rebooted.

Now you should be able to send traffic between the two external devices. To test with a real world network game scenario, on the first device run *OpenTTD* in daemon mode, (see section 7.2.1). Next, on the second device run *OpenTTD* in regular mode and follow the instructions in section 7.2.1 to connect to the multiplayer game on the first device. You will need to explicitly specify the server using the *Add Server* button and typing in *192.168.1.64* (assuming that is the IP address the first device acquired over VPN). Once added you should be able to connect to game running on the game server as before, (see the end of section 7.2.1).