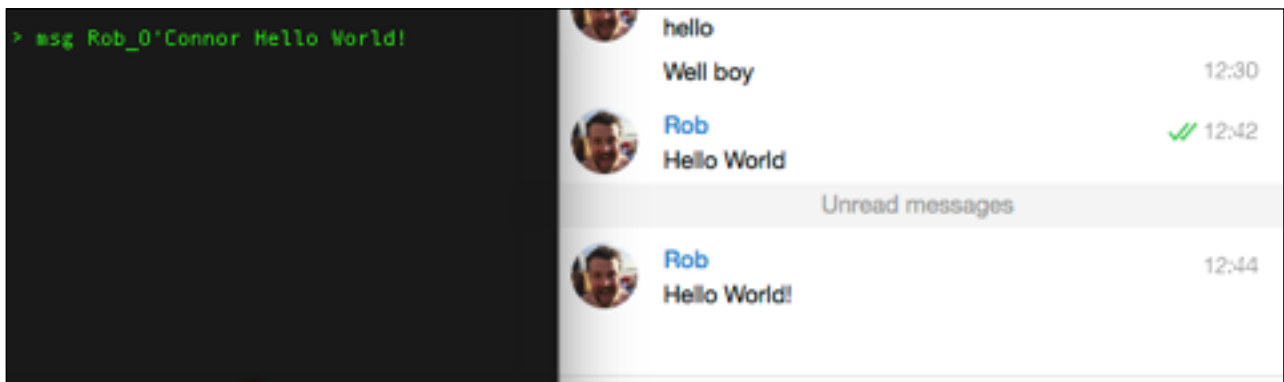

Waterford Institute of Technology

Computer Summer Camp

Lab - Telegram Sam



Introduction

Upon completion of this lab, you will have:

- Set up a Raspberry Pi
- Installed and configured an application using the command line
- Successfully sent an instant message from the Pi to a smartphone/desktop

What You'll Need

- A Raspberry 3 Model B
- Ethernet Cable
- HDMI monitor cable
- Keyboard/Mouse
- Monitor
- Power supply
- Use of a laptop/desktop/smartphone running Telegram
- SD card



In this lab, we're going to piggyback on Telegram, an open source cloud-based messaging system (similar to WhatsApp) to set up a messaging system from the Pi to one of our consumer-level devices (e.g. an iPhone). In order to do this, you'll be installing an operating system and configuring the Telegram application from the command line. Don't worry if you don't understand every step as you're going along - that's absolutely fine. The important thing is to complete the exercise and afterwards, reflect on what you've done. If you have any difficulty with the lab at any point, ask for assistance from the lab supervisor.

[Note: this lab has absolutely nothing to do with 1970s glam rockers T-Rex. It's just some free word association and I like the song 'Telegram Sam']

Steps

Step 1 - Set Up the Raspberry Pi

Connect up all of the required components for your Raspberry Pi. **DO NOT plug in the power until the lab supervisor has checked it is connected up properly.**

You should have a HDMI monitor cable, a network cable (ethernet), USB mouse and USB keyboard. You will also need to put the SD card (which contains Raspbian) in the card reader slot on the Pi.

To reiterate - **DO NOT plug in the power until the lab supervisor has checked it is connected up properly.**

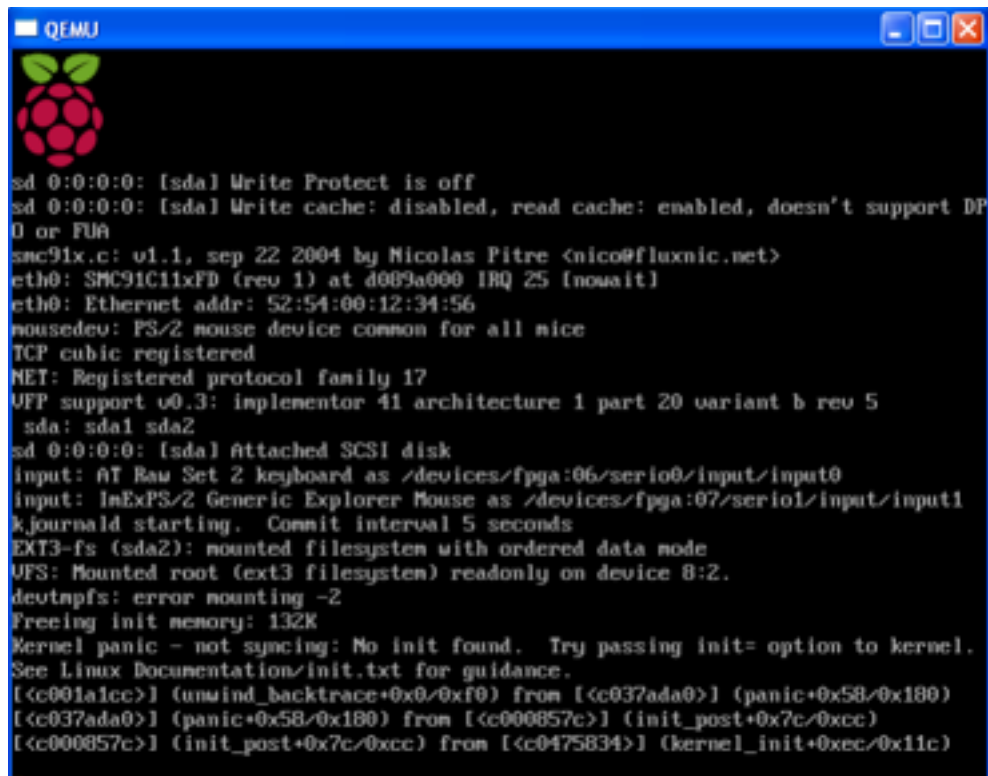
Once you've got this bit signed off by your lab supervisor (i.e. Me!), you're good to turn on the device (Step 3).

Questions to Ask Yourself:

- Can you identify each of the components on the Pi board?
- Which connectors link up which parts?
- Why must you wait before turning on the power?

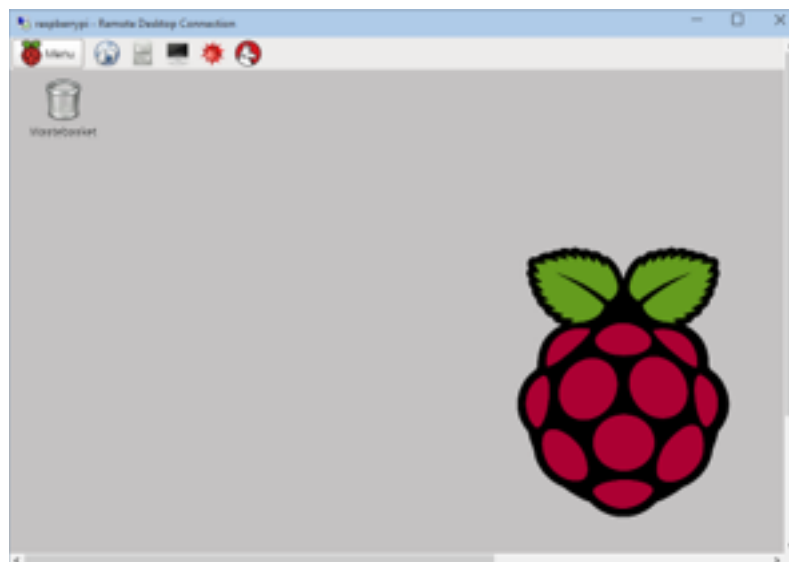
Step 2 - Turn on the Pi

Once the Pi starts to receive power, it will attempt to boot into Raspbian. You should see a screen that looks something like this:



```
QEMU
sd 0:0:0:0: [sda] Write Protect is off
sd 0:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DP
0 or FUA
smc91x.c: v1.1, sep 22 2004 by Nicolas Pitre <nico@fluxnic.net>
eth0: SMC91C11xFD (rev 1) at 4009a000 IRQ 25 (nowait)
eth0: Ethernet addr: 52:54:00:12:34:56
mousedev: PS/2 mouse device common for all mice
TCP cubic registered
NET: Registered protocol family 17
UFS support v0.3: implementor 41 architecture 1 part 20 variant b rev 5
sda: sda1 sda2
sd 0:0:0:0: [sda] Attached SCSI disk
input: AT Raw Set 2 keyboard as /devices/fpga:06/serio0/input/input0
input: ImExPS/2 Generic Explorer Mouse as /devices/fpga:07/serio1/input/input1
kjournald starting. Commit interval 5 seconds
EXT3-fs (sda2): mounted filesystem with ordered data mode
UFS: Mounted root (ext3 filesystem) readonly on device 8:2.
devtmpfs: error mounting -2
Freeing init memory: 132K
Kernel panic - not syncing: No init found. Try passing init= option to kernel.
See Linux Documentation/init.txt for guidance.
[<c001a1cc>] (unwind_backtrace+0x0/0xf0) from [<c037ada0>] (panic+0x58/0x180)
[<c037ada0>] (panic+0x58/0x180) from [<c000857c>] (init_post+0x7c/0xccc)
[<c000857c>] (init_post+0x7c/0xccc) from [<c0475834>] (kernel_init+0xec/0x11c)
```

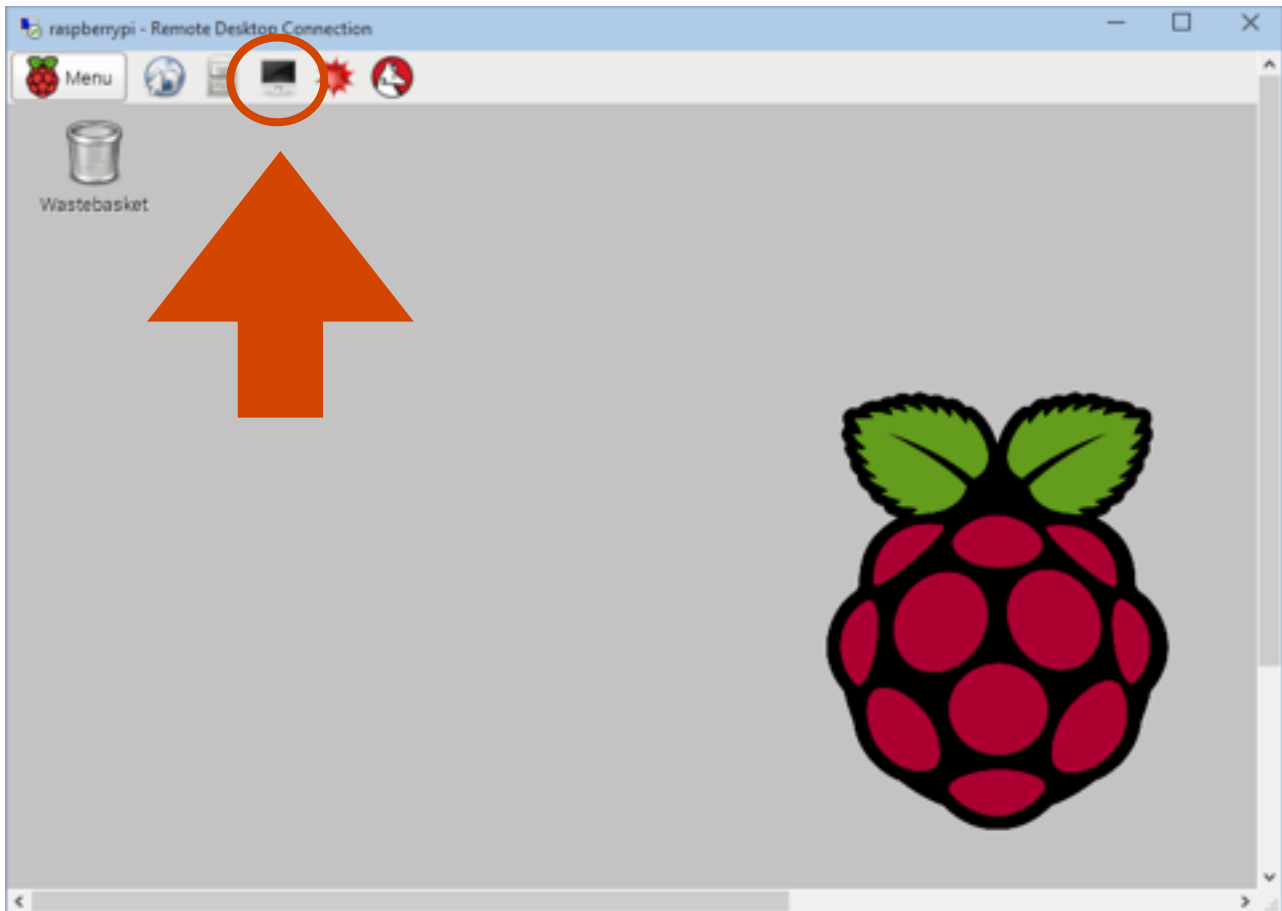
Once the machine boots up, it will launch into the Graphical User Interface (GUI)



From here, have a play around with some of the desktop and menu icons (play Tetris if you like!)

Check that you have a valid network connection by opening up the web browser and visiting some page on the internet. I like to use <http://news.google.ie> as my tester page and I can usually tell if the browser is displaying a cached page.

Next, open up the Terminal program. This is where you can enter command-line instructions. You can see this in the top left hand dock of the screen, titled Terminal



Enter the command "ifconfig" in the terminal. This will display all of the network configuration information. Look for the line labelled "eth0". This should display the ethernet information. The "inet address" line displays your IP address. This isn't necessary for this lab, but it's good information to have.

Questions to Ask Yourself:

- What is the relationship between the command-line and the GUI?
- What is a "cached page"?
- What is an IP address?

Step 3 - Install and Configure Telegram

Now it's time to get down and dirty with the command line. We're going to execute a bunch of commands here - a lot of this may seem like gibberish to you, but just go with it for the time being. In a few months, you'll be fluent gibberish speakers (typists)!

We'll be installing Telegram as per instructions found on the web, largely from two sources:

- <http://www.instructables.com/id/Telegram-on-Raspberry-Pi/>
- <http://www.emmeshop.eu/blog/node/44>

I have them summarised here, but it's good to acknowledge our sources.

In the command line, run the following commands. You must type these EXACTLY. Some commands will take longer to execute than others. While they're executing, open a browser and see if you can figure out what each is doing. Each time you execute a command (by pressing the return key), the terminal will output feedback to the screen - however, it may seem like more gibberish ... but it's not!

```
sudo apt-get update
```

```
sudo apt-get install libreadline-dev libconfig-dev libssl-dev  
lua5.2 liblua5.2-dev libevent-dev make
```

```
git clone --recursive https://github.com/vysheng/tg.git && cd tg
```

```
./configure
```

This one might take a while. If you have any configuration errors, you'll need to address them. If it kicks up an error about Python, you might need to execute the following command, which manually installs the correct version of Python.

```
sudo apt-get install python2.7-dev
```

You may also get an error about JSON. The following command should fix this:

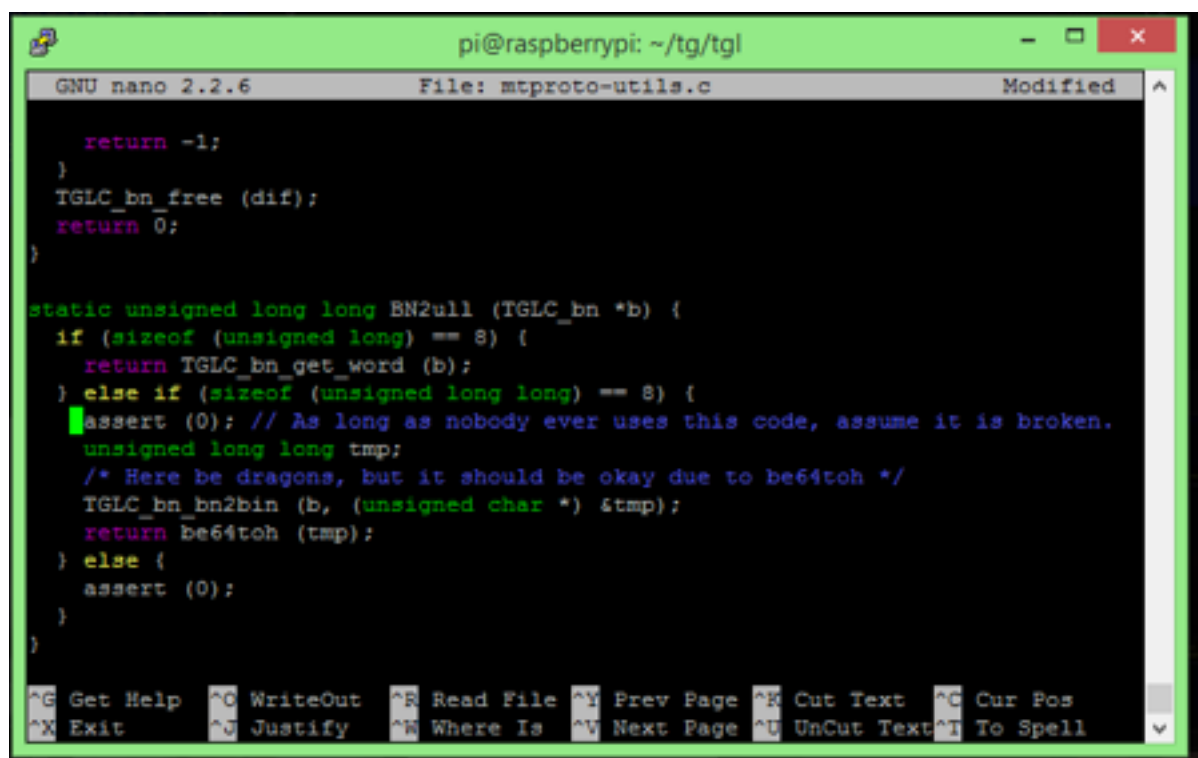
```
sudo apt-get install libjansson*
```

If you've received either or both of those messages, it's because certain software libraries that Telegram needs are missing and need to be installed. Assuming all is well, **run ./configure again**. If you still have errors, ask your lab supervisor.

The next step is tricky. A slight inconsistency has arisen with the latest version of the Raspbian Operating System and the telegram-cli application. In order to fix this, you're going to need to edit two lines of code. You don't need to understand the code, you just need to make teeny-tiny modification (i.e. hack!) so it will work!

```
nano tgl/mtproto-utils.c
```

This will open up the **nano** text editor. You'll need to scroll down to find the function named **static unsigned long long BN2ull**

A screenshot of a terminal window on a Raspberry Pi. The window title is 'pi@raspberrypi: ~/tg/tgl'. The terminal shows the GNU nano 2.2.6 text editor editing the file 'mtproto-utils.c'. The code visible includes a 'return -1;' statement, a 'TGLC_bn_free (dif);' statement, and a 'return 0;' statement. Below these is the 'static unsigned long long BN2ull (TGLC_bn *b) {' function. Inside this function, there is an 'if (sizeof (unsigned long) == 8) {' block with a 'return TGLC_bn_get_word (b);' statement. This is followed by an 'else if (sizeof (unsigned long long) == 8) {' block. Inside this block, the first line is 'assert (0); // As long as nobody ever uses this code, assume it is broken.' The cursor is positioned at the end of this line. Below the 'assert' line is a comment '/* Here be dragons, but it should be okay due to be64toh */' followed by 'TGLC_bn_bn2bin (b, (unsigned char *) &tmp);' and 'return be64toh (tmp);'. The function ends with an 'else {' block containing 'assert (0);' and a closing brace. At the bottom of the terminal, there is a status bar with various nano editor shortcuts like '^G Get Help', '^O WriteOut', etc.

You'll need to change the line 101 (where the cursor is above) from **assert(0);** to **//assert(0);**

This comments out that line so the compiler ignores it.

You'll need to do the same thing at line 115 in the function named **static void ull2BN**. If you run into any difficulty here, ask your lab supervisor for some help.

Once you're finished editing those TWO lines. Press CTRL-X to exit nano and type Y to save the changes (when promoted).

The next command is the big one. This compiles the code and creates the Telegram executable.

```
make
```

This will take a while. Now is a good time to grab some fresh air or a coffee (or both!). While the system is "making", you can also move onto the next step and set up your Telegram account.

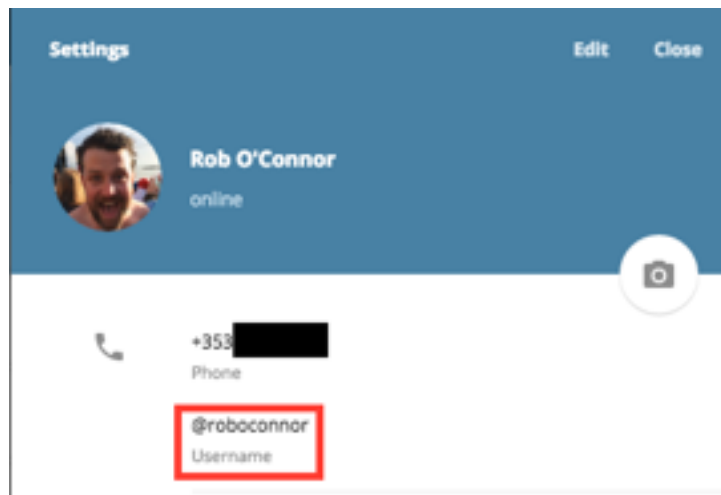
Questions to Ask Yourself:

- What is sudo?
- What is apt-get?
- What are libraries?
- What is git?
- What does compiling do?

Step 4 - Register with Telegram

Visit www.telegram.org and set up a Telegram account (this is free). You should also download one of the free Telegram client apps for phones. Once this is completed, you should have an account set up with a username and phone number.

It is very important that you set up a username with your account. You can do this in the Settings page:



You'll need the username so that people can message you without knowing your phone number.

Step 5 - Connect the Pi to your Telegram Account

Back to the Pi. Once the system has "made" itself, you should switch into the Telegram directory.

```
cd tg
```

then

```
bin/telegram-cli -k tg-server.pub -W
```

The first time we start telegram you must enter the phone number, including international code (for Ireland it's +353). So if your phone number is 0871234567, you'll have to enter +353871234567 (drop the first zero)

You should then receive on your phone a sms message with an authentication code. Enter it at the prompt on the Pi.

Now you are ready to use Telegram. You can only message someone in your friends list. Add the classmate next to you to your friends list (this is very easy!) on Telegram and then send them a message.

You will have successfully completed the lab, if you can send your friend a message from the command line. The format for the msg command is:

msg @<username> message

So you could message me with the following command:

```
msg @roboconnor Hello World!
```

Assuming other classmates get set up too, send some messages to one another.

Telegram is a lot more powerful than sending simple text messages. A list of other commands is available here - <http://www.emmeshop.eu/blog/node/44> - can you successfully send a photo to your classmates?

That's it! But wait

It's important that we shut down the Raspberry Pi before turning off the power. To do this, you can select "shutdown" from the GUI menu. Or if you want to be hard-core, enter the following command at the terminal.

```
sudo shutdown -h now
```

Once the Pi switches off, remove all the connections and tidy away all of the components.

Questions to ask yourself:

- Why did Telegram require authentication?
- What other uses could this have, besides simple messaging to humans?

Conclusion

Even though it may have seemed simple, there was a lot going on here. You've worked with a new operating system, then installed and configured an app from the command line, linked it to a cloud service and successfully sent a message to another person.

That deserves a prize - <https://www.youtube.com/watch?v=du7jfDYe5LI>