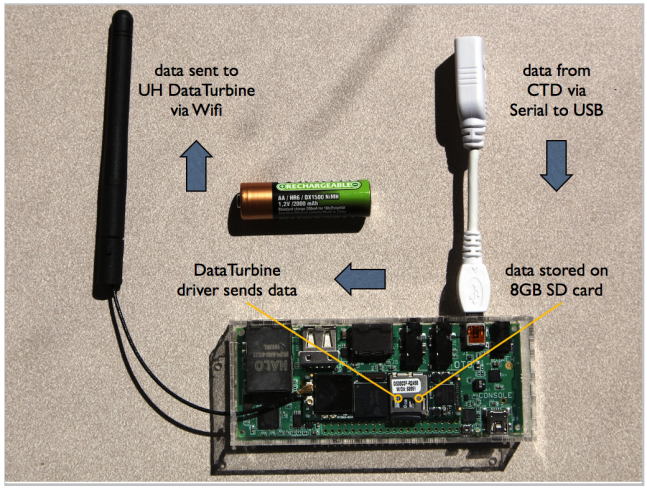
PacIOOS Remote Telemetry Deployment Guide

1. Overview

This document describes how to create a remotely deployed telemetry system for the PacIOOS nearshore sensor project using a low cost, Linux-based WiFi telemetry setup. The deployment guide assumes familiarity with the Linux operating system, environment, and commands, and the software setup is done on a modern Linux distribution (Ubuntu, Redhat, CentOS, Debian, etc.).



2. Hardware Requirements

A typical deployment uses the following hardware:

1. Gumstix Overo Fire COM module ($219)
2. Tobi expansion board ($69)
3. Acrylic case for Gumstix Tobi board ($33)
4. 5.0 volt US power supply ($10)
5. 8GB class 4 MicroSD Memory Card ($15)
6. USB miniB to USB standardA cable ($8)
7. USB standardA to RS232 serial adapter ($19)
8. USB miniA to USB standardA female adapter ($6)
9. Four stainless steel machine screws, #0-80, 1" (pack of 100 $8)
10. Twelve stainless steel hex nuts, #0-80, (pack of 100 $6)
11. Four nylon spacers, 6/6, round, 7.5mm (pack of 100 $7)

3. Software Dependencies

Installing software onto the Gumstix will require a number of software downloads, including the OpenEmbedded Linux distribution itself. We currently are using the Overo precompiled binary from 5 February, 2010. Other third party software is also needed. See below.

3.1 Download the Gumstix Overo pre-compiled binaries

```
http://www.sakoman.com/feeds/omap3/glibc/images/overo
/201002051458/MLO-overo-201002051458
http://www.sakoman.com/feeds/omap3/glibc/images/overo
/201002051458/u-boot-overo-201002051458.bin
http://www.sakoman.com/feeds/omap3/glibc/images/overo
/201002051458/uImage-overo-201002051458.bin
http://www.sakoman.com/feeds/omap3/glibc/images/overo
/201002051458/omap3-desktop-image-overo-201002051458.tar.bz2
```

3.2 Download the FTDI USB to RS232 serial driver

The FTDI Driver is installed on your workstation in order to connect to the Gumstix serial console via USB. Choose the right driver for your operating system.

3.3. Install third party software

3.3.1 To function correctly and to have standard command line conveniences, install the following opkg software packages on the Gumstix:

```
cron, jamvm, lib-rxtx-jni, lsof, minicom, ntp-bin, openvpn,
perl-module-strict, perl-module-exporter-heavy, perl-module-
sys-hostname,
perl-module-xsloader, perl-module-io-socket, perl-module-io,
perl-module-integer, perl-module-io-select, python-logging,
python-subprocess, screen, vim-syntax
```

3.3.2 Build and add the latest *pacioos-gumstix.tar.gz* distribution to */usr/local/bbl* from the BBL Subversion repository.

3.3.3 From ddclient on SourceForge, download, install, and configure the ddclient software. See below.

4. Gumstix Installation

The OpenEmbedded Linux operating system needs to be installed onto the 8GB microSD card such that it is bootable.

4.1 Setting up the microSD card

With the device in a USB microSD card reader and inserted into a USB port on your machine, unmount the existing filesystem if there is one and use fdisk expert mode to partition the card.

4.1.1. Set the disk geometry to have 255 heads, 63 sectors, 512 bytes per sector. The number of cylinders (for an 8GB card) is calculated by:

```
(disk size in bytes) / heads / sectors / (bytes per sector)

7,973,371,904 / 255 / 63 / 512

= 969.37, or 969 cylinders
```

4.1.2. Create a 32MB FAT partition 1 (id 0C), and the remainder of the disk a 7.9GB ext3 partition 2 (id 83). Mark the first partition as bootable. Format the partitions as FAT and ext3 respectively.

4.1.3. Finally, copy the boot files and the OpenEmbedded distribution to the FAT and ext3 partitions respectively.

```
$ sudo mount /dev/sdc1 /media/FAT
$ sudo mount /dev/sdc2 /media/linux
$ sudo cp MLO-overo-201002051458 /media/FAT/MLO
$ sudo cp u-boot-overo-201002051458.bin /media/FAT/u-boot.bin
$ sudo cp uImage-overo-201002051458.bin /media/FAT/uImage
$ cd /media/linux
$ sudo tar xvjf omap3-desktop-image-overo-201002051458.tar.bz2
```

4.2 Boot the Gumstix

Once the microSD card is imaged, boot the Gumstix off of it. Connect via USB serial to gain a console prompt. Any terminal emulator will work, but I've used screen here. The communications settings should be set to a baud rate of 115200, 8 data bits, no flow control, 1 parity bit.

```
screen /dev/ttyUSB0 115200, cs8, ixoff
```

Then apply power to the Gumstix.

4.3 Set up the operating system and services

4.3.1. Change the root password.
4.3.2. Set the system to boot into runlevel 3.
4.3.3. Add in the *pacioos* user, which will run the instrument driver software.
4.3.4. Change syslog.conf to "file" rather than "buffer".
4.3.5. Networking depends on a small change to the *hosts* file. Add *overo* to the localhost line:

```
127.0.0.1 overo localhost.localdomain localhost
```

4.3.6. Add *networking* and *openvpn* to run level 3 and remove the following services since they aren't needed:

```
apmd, avahi-daemon, bluetooth, caps, NetworkManager, samba
```

4.3.7. Configure networking over the 802.11b/g Wifi interface *wlan0*. The exact setup will differ per site depending on the local network settings. Some require WEP or WPA authentication, while others are open.

For open networks:

```
auto lo
iface lo inet loopback

allow-hotplug wlan0
auto wlan0
iface wlan0 inet dhcp
pre-up /sbin/iwconfig wlan0 essid “your-essid”
wireless_mode managed
```

For WPA2 networks:

```
auto lo
iface lo inet loopback

allow-hotplug wlan0
auto wlan0
iface wlan0 inet dhcp
pre-up wpa_supplicant -Dwext -iwlan0 -c
/etc/wpa_supplicant.conf -B
down killall wpa_supplicant
wireless_mode managed
```

Also add the wpa_supplicant.conf:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
eapol_version=1
ap_scan=1
fast_reauth=1

network={
ssid=”your-essid”
proto=WPA2
key_mgmt=WPA-PSK
pairwise=CCMP TKIP
group=CCMP TKIP
scan_ssid=1
psk=”your-wpa-passkey”
priority=10
}
```

For WEP networks:

```
auto lo
iface lo inet loopback

allow-hotplug wlan0
auto wlan0
iface wlan0 inet dhcp
iwconfig wlan0 essid “your-essid”
iwconfig wlan0 key your-hex-key
ifdown wlan0
ifup wlan0
```

Once the configuration is finished, restart the networking service to bring up the WiFi interface.

4.3.8. Add the following repository to the opkg configuration file:

```
http://www.angstrom-distribution.com/feeds/2008/ipk/glibc
/armv7a/base
```

4.3.9 Set up the timezone files correctly for Pacific time zones. First, download the Pacific timezone files from the NIH FTP site:

```
elsie.nci.nih.gov/pub/tzdata*.tar.gz
```

Open the archive and use the standard Linux timezone compiler to compile the zone files into a directory called *zoneinfo*. Once compiled, replace */usr/share/zoneinfo* with this new directory.

Save the old */etc/localtime* file and symbolically link to the appropriate new zone file.

4.3.10. Create the following directories, which will be used for the data streaming, archiving, and logging. Be sure that they are writable by the *pacioos* user and group.

```
/usr/local/rbnb/log
/var/log/rbnb (symbolically link to above)
/usr/local/wifi/log
/var/log/wifi (symbolically link to above)
/usr/local/openvpn
/var/log/openvpn (symbolically link to above)
/usr/local/dyndns
/etc/ddclient
/data/raw
/data/spool
/dev/net
```

4.3.11. Set up NTP now that the new zone files are in place. First, stop the *ntpd* service and then add

time.apple.com or any other reliable NTP server to the NTP configuration file. Restart the service.

4.3.12. Prepare the OS for openVPN connections by creating the correct device files and loading the modules:

```
$ sudo mknod /dev/net/tun c 10 120
$ sudo modprobe tun
```

Add the *tun* module to the *modules* configuration file.

4.3.13. Set up the dynamic DNS client. First, install the ddclient download into */usr/local/dyndns*. Copy the configuration file provided into the *ddclient.conf* file. Configure the client to use web IP detection with the *pacioos* user and password:

```
pid=/var/run/ddclient.pid
ssl=no
use=web, web=checkip.dyndns.org/, web-skip='IP Address'
login=pacioos
password=pacioos-password-goes-here
server=members.dyndns.org, \
protocol=dyndns2 \
pixx01.dyndns.org
```

Hostnames so far are set up as:

```
pias01.dyndns.org (American Samoa)
pifm01.dyndns.org (Micronesia)
pimi01.dyndns.org (Marshall Islands)
pipl01.dyndns.org (Palau)
```

Set up the ddclient service in run level 3 with the script provided.

5. Set up WiFi authentication

Certain deployments will require username and password-based authentication if the upstream network doesn't support MAC-based authentication. So far, only American Samoa supports it. For the others, install the *authwifi* service script found in the bin directory of the BBL Subversion repository. Then, customize the getWifiAuth.sh script for the particular site's network login.

6. Connecting to the VPN server

For each PacIOOS remote deployment, IP addresses are obtained in many ways. Services providers such as Blue Sky in American Samoa provide publicly accessible IP addresses, however others do not and only assign an IP address on a private network. To be able to manage the deployment remotely, each device on a private network must establish a connection to the PacIOOS VPN server running on *bbl.ancl.hawaii.edu*.

6.1 Configuring the VPN client

Copy the PKI keys and client.conf files from the server that correspond with the particular deployed client to the openvpn configuration directory. For instance, for *pipl01*, copy *pipl01.conf*, *keys/pipl01**, and *keys/ca.crt* to the client. For convenience, the server should contain *pixx01.tgz* archives of all files for each client.

7. Troubleshooting the Gumstix installation

There may be times when the Gumstix device won't boot properly. Here are some possible ways to troubleshoot it.

7.1 Possible corrupt filesystem

The Linux OS is installed on an ext3 partition, and the partition may have errors. Fix any errors if possible:

```
$ sudo umount /dev/sdc2
$ sudo fsck.ext3 /dev/sdc2
```

Otherwise, there may be hardware problems with the microSD card itself. Test for bad blocks, or bad memory i/o:

```
$ sudo badblocks /dev/sdc2
```