Multicarrier Vehicle Telemetry

Claiton Francisco, Gabriel Langer e Gabriel Luz

Motivação

O Ônibus Elétrico Alimentado por Energia Solar da UFSC foi inaugurado em dezembro de 2016, iniciando o serviço regular de transporte entre o Campus Trindade e o Sapiens Parque em março de 2017. O veículo foi proposto como um projeto de deslocamento produtivo, e possui poltronas confortáveis, duas mesas de reunião, tomadas 220V e USB, além de ar-condicionado. Foi proposto também que a rede wi-fi da UFSC continuasse funcionando mesmo dentro do ônibus, porém ainda não foi implementado. Um grande desafio para resolver o problema do wi-fi no ônibus é garantir a qualidade do sinal em todo o trajeto, utilizando roteamento de internet proveniente de uma rede 4G. Além disso, é muito importante que dados sobre a situação do ônibus sejam disponibilizados em tempo real para análises e tomadas de decisão.

Objetivos

O objetivo deste trabalho é desenvolver uma solução que forneça a melhor rede possível de wi-fi para os passageiros do Ônibus Elétrico em cada parte do trajeto, além de enviar os dados do veículo em tempo real para um servidor da UFSC. Além disso, se possível, será feita uma análise em cima dos dados obtidos.

Metodologia

Para fornecer uma rede wi-fi com qualidade, utilizaremos um Raspberry PI conectado à módulos 4G de diferentes operadoras de telefonia, escolhendo sempre a rede com mais consistência para cada parte do trajeto.

Os dados do ônibus deverão ser obtidos através de um módulo CAN conectado a um EPOSMote III já instalado no interior do veículo.

Tarefas

- Entregar a revisão bibliográfica e detalhamento do projeto
- Demonstração de viabilidade tecnológica do módulo CAN/Telemetria e qualidade do sinal/roteamento
- Conexão com o EPOSMote do ônibus e envio de telemetria
- Realizar o roteamento do 4G no ônibus
- Teste de coleta de dados e roteamento

- EXTRA: Aplicação de técnicas de Data Science nos dados obtidos

Entregáveis

D1: Detalhamento do projeto
D2: Teste das tecnologias do projeto
D3: Demonstração do funcionamento da rede Wi-Fi
D4: Demonstração da confiabilidade de dados extraídos
D5: Demonstração dos dados recebidos no servidor
D6 (Optional): Estudo de novas conclusões a respeito dos dados do ônibus.

Cronograma

Task	**20/09**	**27/09**	**04/10**	**25/10**	**18/10**	**25/10**	**01/11**	**08/11**	**15/11**	**22/11**	**29/11**
Tarefa1	x	D1									
Tarefa2		x	x	D2							
Tarefa3				x	x	x	D3				

Task	20/09	27/09	04/10	25/10	18/10	25/10	01/11	08/11	15/11	22/11	29/11
Tarefa4							x	x	D4		
Tarefa5								x	x	x	D5

CONCEITOS INICIAIS DO PROJETO

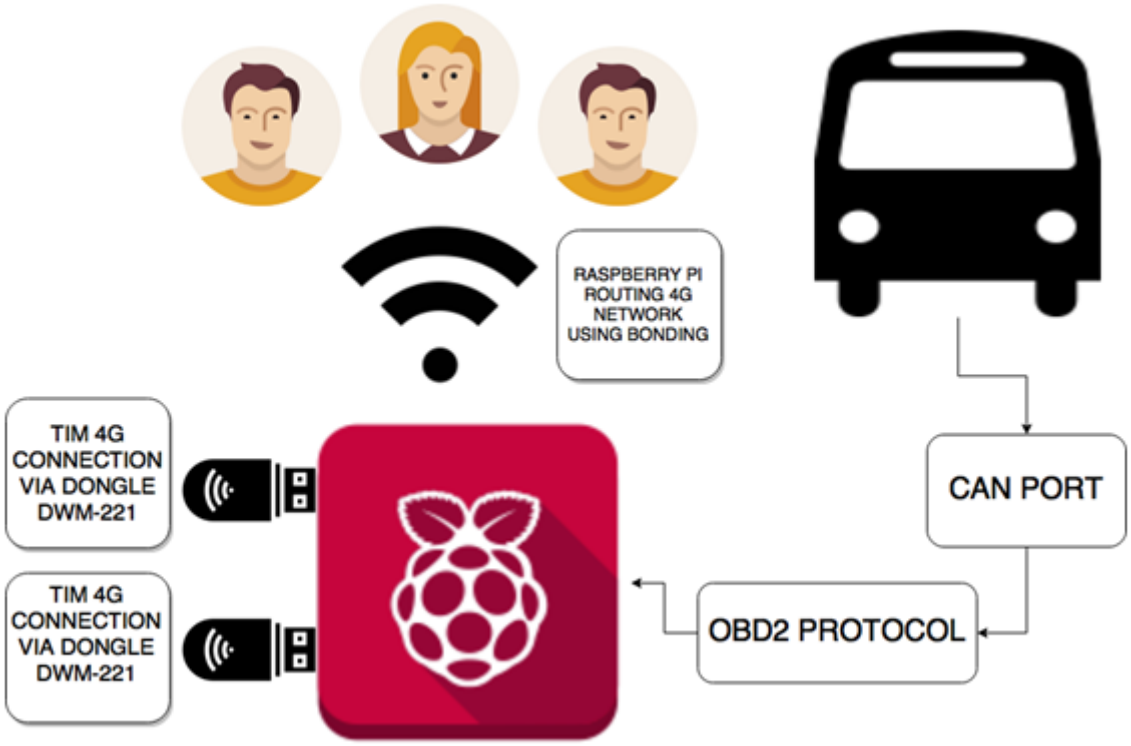

Figura 1. Diagrama geral do projeto

LEITURA DE DADOS DO ÔNIBUS

1.1 - Introdução

Neste trabalho iremos fazer a leitura de dados da injeção eletrônica do ônibus elétrico da Universidade e transmitir em tempo real para um servidor. Os dados serão lidos na porta CAN do ônibus utilizando o EPOS, transmitidos usando o protocolo OBD2 via serial para o raspberry PI, onde interpretará os dados e transmitirá os dados para o servidor.

Os dados fornecidos pela injeção eletrônica de um veículo são diversos, vão desde o status do limpador de parabrisas, até a velocidade atual.

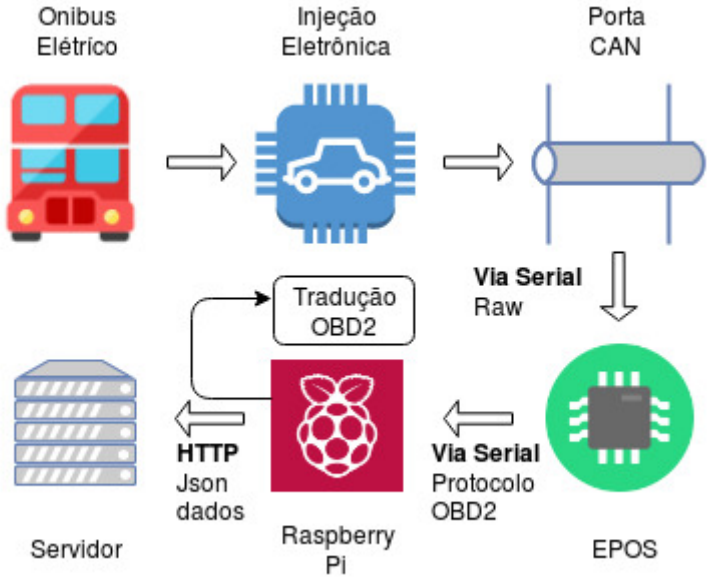

Figura 2. Diagrama Leitura CAN

1.2 - Porta CAN
A porta CAN, Controller Area Network, é um standard na indústria automobilística para a comunicação de dispositivos e microcontroladores sem um host. Todos os dispositivos do carro se comunicam utilizando esse standard
1.3 - Protocolo OBD2
O OBD, On-board diagnostics, é um termo automotivo referenciando a capacidade de um veículo de se auto diagnosticar e fornecer dados do veículo. Existem várias implementações do protocolo, o OBD2 é um standard para o protocolo implementado porém existem muitas variações, cada montadora usa a sua implementação do protocolo. Quando você conecta um scanner OBD2 ele passa a enviar mensagens na CAN Bus onde os outros dispositivos estão escutando por essas mensagens, e respondem com o status requisitado por quem enviou a mensagem.
1.4 - ELM 327
O ELM 327 vem para ajudar, produzido pela ELM Electronics para traduzir protocolos OBD na maioria dos carros modernos. Ele é uma interface para 10 diferentes protocolos
• SAE J1850 PWM (41.6 kbit/s)
• SAE J1850 VPW (10.4 kbit/s)
• ISO 9141-2 (5 baud init, 10.4 kbit/s)
• ISO 14230-4 KWP (5 baud init, 10.4 kbit/s)
• ISO 14230-4 KWP (fast init, 10.4 kbit/s)
• ISO 15765-4 CAN (11 bit ID, 500 kbit/s)
• ISO 15765-4 CAN (29 bit ID, 500 kbit/s)
• ISO 15765-4 CAN (11 bit ID, 250 kbit/s)
• ISO 15765-4 CAN (29 bit ID, 250 kbit/s)
• SAE J1939 (250kbps)
• SAE J1939 (500kbps)
Porém, o ônibus elétrico utiliza uma implementação experimental no standard OBD2 e não poderá ser utilizado o ELM 327, por enquanto temos testado utilizando nossos próprios veículos os quais estão incluídos na lista dos protocolos que o ELM327 traduz.

Figura 3. Adaptador ELM327

1.5 - Metodologia do teste
Estamos primeiramente tentando ler dados do nosso carro utilizando a interface ELM327 e enviar esses dados para um servidor e ficar disponível via web para o usuário. Após concluirmos esta etapa, iremos traduzir a implementação do OBD2 utilizada pelo ônibus elétrico para as informações disponíveis.
1.6 - Resultados do teste da leitura via bluetooth usando Raspberry PI
No momento conseguimos ler os dados de um dos carros utilizando a interface ELM327 porém encontramos alguns problemas para portar isso para o raspberry pi, no momento estamos trabalhando para resolver esses problemas e poder ler os dados diretamente pelo raspberry pi conectado via serial a porta CAN do carro.

CONEXÃO COM A INTERNET USANDO BONDING DE MÚLTIPLAS INTERFACES DE CONEXÃO MÓVEL (4G)

`Atualização: Gateways EPOS que requerem conexões 4G estão utilizando Zeroshell, conforme documentado no guia do desenvolvedor`

(http://epos.lisha.ufsc.br/IoT+Internals#Zeroshell_IoT_Gateway_Installation).

1.1 - Introdução

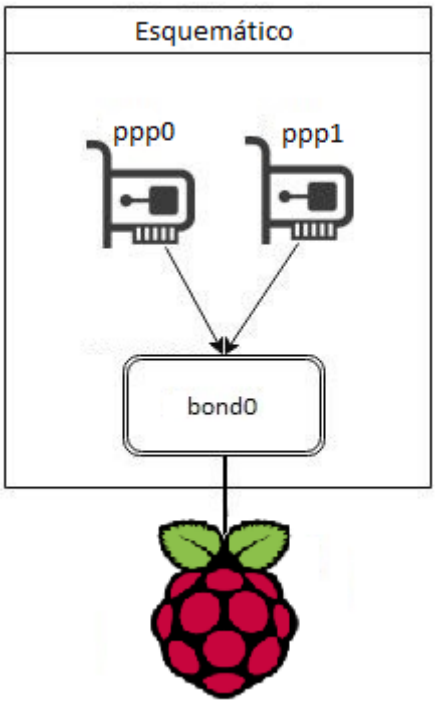

Figura 4. Diagrama de bonding

Usando os dois equipamentos DONGLES disponibilizados pelo lisha para conexão com a internet, estamos configurando o raspberry pi para funcionar como um roteador wifi, configurando um DHCP SERVER e colocando essas duas interfaces em bonding para suporte a queda de internet por uma das interfaces. Até o momento, o grupo está tendo dificuldade em subir as duas interfaces ao mesmo tempo e ativar o servidor DHCP para distribuição dos IP's.

1.2 - O que é bonding?

Ethernet bonding, regulado pela norma IEEE 802.3ad com o título link aggregation é uma técnica em redes de computadores usada para o acoplamento de dois ou mais canais ethernet em paralelo para produzir um único canal de maior velocidade e/ou aumentar a disponibilidade e redundância desse canal.

Em relação ao objetivo do trabalho, o intuito é garantir a redundância do canal de comunicação com a internet, fazendo com que ao se detectar a indisponibilidade de rede através da interface de comunicação principal (detecção feita de forma automática pelo sistema operacional), o interface secundário assumirá a responsabilidade de transmitir os pacotes de rede.

Ferramenta usada para configuração do bonding: ifenslave

As configurações estão sendo feitas e testes estão em andamento. Sem dados para inserir nesse entrega.

1.3 - Sobre dongles e recebimento do sinal

Figura 5. Dongle TP-Link DWM-221
Os dongles utilizados para a viabilidade do projeto foram ambos da marca TP-Link, modelo DWM-221, ambos com chips 4G. Estes dongles vêm pré-configurados de fábrica com dados da operadora Oi conforme abaixo:

Nome = Oi Dados
APN = gprs.oi.com.br
Usuário = oi
Senha = oi

Estes dados são inseridos em uma aplicação configurada no raspberry pi, chamada sakis3g, que é responsável por toda a parte da criação de uma interface de comunicação de rede, a ppp0. Tendo as duas interfaces comunicação com a rede, podemos posteriormente fazer as configurações de bonding e de dhcp.

Figura 6. Conexão bem sucedida do raspberry com a rede 4G da operadora TIM através da aplicação sakis3g.

1.4 - Roteamento do sinal de um módulo 4G

Para a parte do roteamento dessa rede 4G através do adaptador Wi-Fi interno do Raspberry Pi foram utilizados dois pacotes muito importantes:

- **hostapd**: Este pacote permite a utilização do Wi-Fi interno como um ponto de acesso.
- **dnsmasq**: Este pacote é uma combinação entre o DHCP e servidor DNS com fácil configuração.

No sistema operacional utilizado (Raspbian, uma distribuição de Debian para RPi), a configuração padrão de interface é feita pelo dhcpcd. Então foi necessário configurar wlan0 para ser ignorada, pois a partir de agora será feita através um IP estático. Esse endereço IP estático foi configurado na parte de interfaces (/etc/network/interfaces).

Então foram feitas as configurações de roteamento, configurando wlan0 como a interface, o nome da rede, a senha e outras. Esta parte é realizada através do hostapd, no arquivo /etc/hostapd/hostapd.conf. Nesta etapa já conseguimos visualizar a rede por outro dispositivo, porém ainda resta a conexão com a rede 4G. Também foi feita a configuração do dnsmasq, onde especificamos o endereço ip para receber as requisições, além de definir o alcance do dhcp da interface.

Para realizar o Forwarding, isto é, direcionar o tráfego de ppp0 para wlan0, configuramos o arquivo sysctl.conf para fazer ip forwarding ajustando uma flag pré-definida. Além disso precisamos configurar a

NAT (Network Address Translation) entre as interfaces através de iptables, onde foi configurado o POSTROUTING e FORWARD. Então foram iniciados os serviços hostapd e dnsmasq, além de configurados para serem inicializados no boot do sistema.

Após estes passos, o Raspberry Pi está servindo como roteador de 4G e dependendo da necessidade, pode ser utilizado como ponte para um roteador externo através da porta Ethernet (interface eth0).

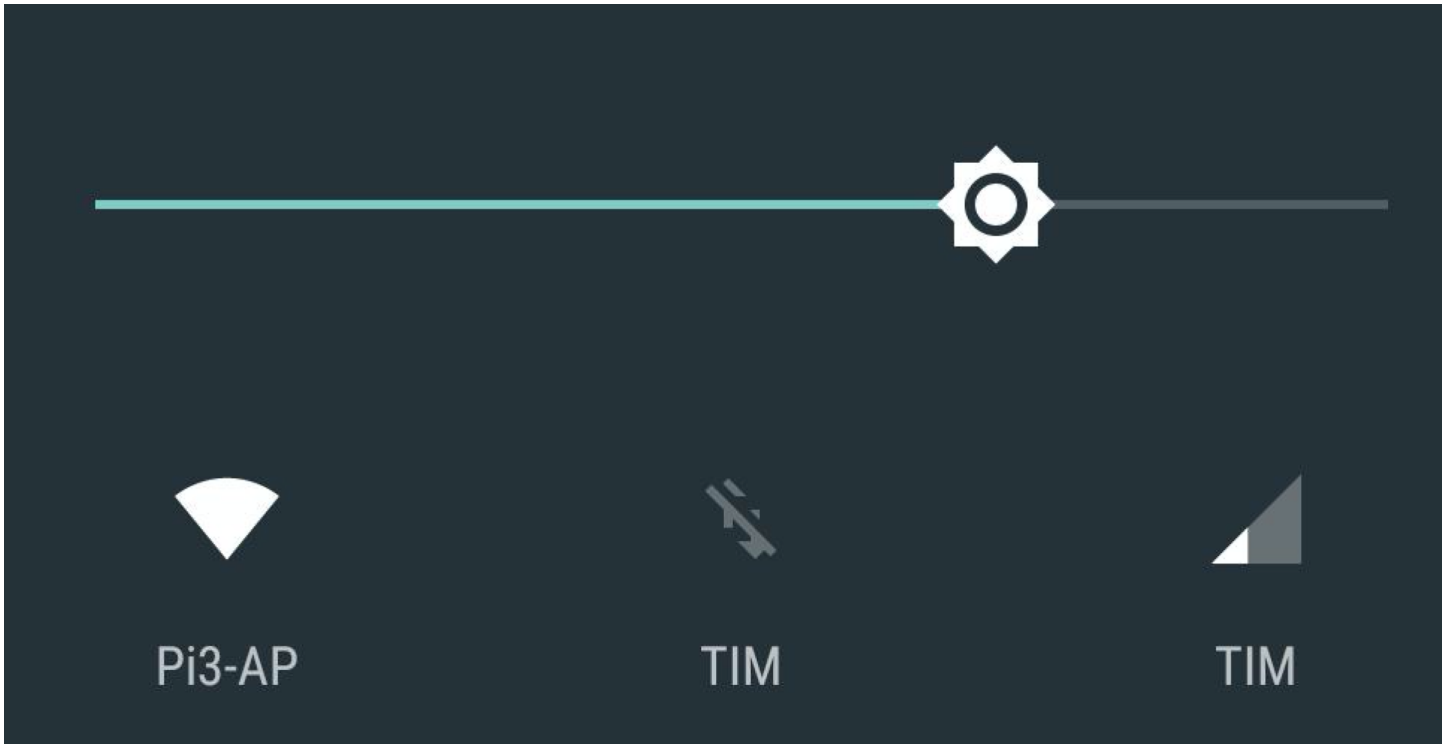

Figura 7. Conexão com a internet através da rede Wi-FI configurada (Pi3-AP).

1.5 - Controle de rede usando bonding
Este controle será feito na próxima etapa, onde será criada uma interface chamada bond0, que substituirá a ppp0 e fará o controle de qualidade do sinal proveniente de dois dongles.

ATIVAÇÃO DOS MODEMS PARA ROTEAMENTO VIA IP ROUTE

1 - Conexão dos modem 3G simultaneamente:

Com o avanço do trabalho foi identificado algumas limitações relacionado ao Sakis3G, onde o mesmo apresentou problema de manter dois modems ativos simultaneamente.

Para corrigir o problema foi usado **WVDIAL**, outra ferramenta para gerenciamento dos dongles. O Wvdial é ideal para automatizar a tarefa de discagem, rediscagem e autenticação de login e senha. Uma vez invocado, o programa telefona para seu provedor e quando atendido, faz automaticamente a efetuação de login no mesmo. Uma vez que você se ausentou do computador e deixou uma janela de download aberta, caso sua conexão com o provedor caia, o WvDial redisca para o mesmo automaticamente.

Tal ferramenta possibilita subir 2 ou mais modems de forma simultânea sem problema de queda de conexão, mantendo todas as interfaces ativas por todos os período de testes.

- Ferramenta de uso: **wvdial**.

É necessário adicionar algumas informações no arquivo de configuração do wvdial:

```
[Dialer Defaults] Init1 = ATZ Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0 Init3 =
```

AT+CGDCONT=1,"IP","gprs.oi.com.br" Username = oi Password = oi Phone = *99# Modem Type = Analog Modem Baud = 115200 Stupid Mode = 1 CarrierCheck = no Check DNS = 0 Auto DNS = 1 Dial Command = ATD New PPPD = yes ISDN = 0 Dial Timeout = 0 [Dialer phone] Modem = /dev/gsmmodem [Dialer phone2] Modem = /dev/gsmmodem2

- Cada modem conectado ao raspberry deve ter sua interface **Dialer phone** no arquivo de configuração, onde é apontado para o um link que contém o device de cada dongle conectado.

Após configurar o wvdial é necessário rodar o seguinte comando:

1. **wvdial phoneX**

Tal comando irá ativar a interface selecionada carregando uma INTERFACE ppp deixando-a disponível;

Abaixo segue um exemplo com a configuração de subida dos dois dongle:

```
pi@raspberrypi:~ $ sudo wvdial phone
--> WvDial: Internet dialer version 1.61
--> Initializing modem.
--> Sending: ATZ
ATZ
OK
--> Sending: ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
OK
--> Sending: AT+CGDCONT=1,"IP","gprs.oi.com.br"
AT+CGDCONT=1,"IP","gprs.oi.com.br"
OK
--> Modem initialized.
--> Sending: ATD*99#
--> Waiting for carrier.
ATD*99#
CONNECT
--> Carrier detected.  Starting PPP immediately.
--> Starting pppd at Sun Nov 12 20:21:59 2017
--> Pid of pppd: 1082
--> Using interface ppp0
--> pppd: ��h[01]
--> pppd: ��h[01]
--> pppd: ��h[01]
--> pppd: ��h[01]
--> pppd: ��h[01]
--> local  IP address 100.65.68.62
--> pppd: ��h[01]
--> remote IP address 10.64.64.64
--> pppd: ��h[01]
--> primary   DNS address 189.40.198.80
--> pppd: ��h[01]
--> secondary DNS address 8.8.4.4
--> pppd: ��h[01]
```

```
pi@raspberrypi:~ $ sudo wvdial phone2
--> WvDial: Internet dialer version 1.61
--> Initializing modem.
--> Sending: ATZ
ATZ
OK
--> Sending: ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
OK
--> Sending: AT+CGDCONT=1,"IP","gprs.oi.com.br"
AT+CGDCONT=1,"IP","gprs.oi.com.br"
OK
--> Modem initialized.
--> Sending: ATD*99#
--> Waiting for carrier.
ATD*99#
CONNECT
--> Carrier detected.  Starting PPP immediately.
--> Starting pppd at Wed Nov 15 11:47:24 2017
--> Pid of pppd: 1170
--> Using interface ppp1
--> pppd: ???
--> pppd: ???
--> pppd: ???
--> pppd: ???
--> pppd: ???
--> pppd: ???
--> local  IP address 100.70.218.81
--> pppd: ???
--> remote IP address 10.64.64.65
--> pppd: ???
--> primary   DNS address 189.40.198.80
--> pppd: ???
--> secondary DNS address 8.8.4.4
--> pppd: ???
```

teste de ping com saída pelas interfaces:

```
pi@raspberrypi:/ $ ping -I ppp0 8.8.8.8
PING 8.8.8.8 (8.8.8.8) from 191.170.163.216 ppp0: 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=57 time=153 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=57 time=74.2 ms
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 2 received, 33% packet loss, time 2002ms
rtt min/avg/max/mdev = 74.225/113.636/153.047/39.411 ms
pi@raspberrypi:/ $ ping -I ppp1 8.8.8.8
PING 8.8.8.8 (8.8.8.8) from 100.70.218.81 ppp1: 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=56 time=220 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=56 time=106 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=56 time=185 ms
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 106.772/171.172/220.877/47.730 ms
pi@raspberrypi:/ $
```

2 - Configuração de roteamento com a internet:

Ao tentarmos criar uma interface bonding com os dois **PPP's** gerados a partir do **wvdial**, nos deparamos com outro problema. O bonding apresentou diversos problemas de compatibilidade com as interfaces ppp, fazendo com que a possibilidade de realizarmos tal configuração ficasse inviável.
Foi observado que algumas empresas como **CISCO**, **HUAWEI** ou **NIKROTIK** disponibilizam equipamentos especializados para realizar a configuração que estávamos tentando, o que nos levou a concluir que é necessário um hardware específico para agregação de interfaces ppp para uso em bonding.

Com isso usamos outra abordagem para resolução do problema.

Ativamos os dois dongles e usamos a configuração de roteamento com detecção de queda de conexão para fazer a seguinte configuração:

- para habilitar a conexão com a internet através dos dois modems, foi necessário adicionar à tabela de roteamento default a saída via PPP's com o comando **ip route add default dev pp0** e **ip route add default dev pp1**
    - O Kernel do Linux, a partir da versão 2.2, possui toda a funcionalidade de gerenciamento de largura de banda, através de controle de tráfego. Isto é feito por filtros e filas. Os filtros colocam o tráfico em filas, e nestas há o encaminhamento de quais pacotes serão enviados primeiro ou depois ou, ainda, rejeitados.
- Com isso conseguimos manter dois links com a internet, um principal ao qual todo o tráfego está passando e outro de backup, caso o primeiro caia o segundo irá "tomar" o lugar dele como rota default.

MANTENDO DOWNLOADS ATIVOS APÓS QUEDA DE REDE

Utilizando regras de roteamento conseguimos chegar em resultados esperados, o usuário não notaria a troca de link quando acontece falha, porém, qualquer download ativo cairia pois há troca de IP externo trocando entre os modem GSM. Uma solução que idealizamos seria o uso de uma VPN para manter o IP externo fixo, dessa maneira, mesmo que os links caiam, se o link com o servidor da vpn estiver ativo, o download não cai.

Um teste que fizemos foi conectar a rede roteada pelo raspberry pi, e então conectar a uma VPN, os resultados foram promissores, mesmo derrubando os dois links gsm, o download para porém não cancela, quando um dos links retorna, o download volta a baixar, a conexão não é perdida.

Para incorporar ao raspberry pi tentamos conectar a um servidor OpenVPN gratuito no raspberry pi e então rotear via a interface de rede criada pelo OpenVPN tun0 porém por alguma flag de configuração errada, quando o link cai o client do OpenVPN não tenta reconectar. Pela necessidade de focar na outra parte do projeto que é ler os dados de telemetria deixamos para arrumar esse ponto adicional em configurar para o raspberry conectar via VPN após amadurecermos a parte do projeto sobre a telemetria.

Bibliografia

1. http://epos.lisha.ufsc.br/EPOSMote+III+Programming
2. https://www.raspberrypi.org/documentation/
3. https://wiki.linuxfoundation.org/networking/bonding
4. http://www.can-wiki.info/
5. http://www.dcs.warwick.ac.uk/~nathan/resources/Publications/aii-2016.pdf
6. https://tools.ietf.org/html/rfc1332
7. https://linux.die.net/man/5/wvdial.conf