UNIVERSIDADE
FEDERAL RURAL
DE PERNAMBUCO

Thiago Valentim Bezerra

Interface de comunicação assíncrona *multihomed* para telemetria de plataformas de coleta de dados ambientais

Recife

2015

Thiago Valentim Bezerra

Interface de comunicação assíncrona *multihomed* para telemetria de plataformas de coleta de dados ambientais

Monografia apresentada ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Universidade Federal Rural de Pernambuco – UFRPE

Departamento de Estatística e Informática

Curso de Bacharelado em Sistemas de Informação

Orientador: Victor Wanderley Costa de Medeiros

Coorientador: Glauco Estácio Gonçalves

Recife

2015

À Deus, minha amada esposa, meus pais e a todos os professores da graduação.

Agradecimentos

Agradeço à Deus e dedico este trabalho a muitas pessoas que foram - e ainda são - importantes para mim. Sei que não citarei o nome de todos, mas farei o máximo para dedicar uma homenagem apropriada. Agradeço de coração a:

A minha amada esposa, Milena Rodrigues, por ter a compreensão e a paciência do mundo durante a minha graduação.

Aos meus pais Francisco e Damares, pois sem eles não chegaria aqui. Ao meu irmão Filipe, aos meus amigos de graduação que fiz durante seus longos 5 anos.

A todos os professores do BSI que durante a graduação passaram todo o conhecimento.

Ao meu orientador Victor Medeiros e ao meu coorientador Glauco Gonçalves por passarem o conhecimento e orientação para a realização deste trabalho.

“As leis da natureza nada mais são que
pensamentos matemáticos de Deus”
(Johannes Kepler)

Resumo

A coleta de dados ambientais é uma ferramenta importante em diversas áreas de pesquisa. No entanto, para que os dados coletados em ambientes remotos e de difícil acesso sejam transmitidos integralmente é importante que a transmissão dessas informações seja feita através de canais redundantes de comunicação. O desenvolvimento deste trabalho está focado na criação de uma interface de comunicação assíncrona *multihomed* para telemetria que é parte fundamental na transmissão confiável de dados. A comunicação *multihomed* consiste na utilização de diferentes tecnologias de transmissão a fim de aumentar a disponibilidade do canal de comunicação. A interface desenvolvida neste trabalho foi construída de maneira modular permitindo a incorporação simplificada de diferentes tecnologias de comunicação. Como resultado é apresentada a avaliação da interface funcionando em uma plataforma Raspberry Pi utilizando dois tipos diferentes de tecnologias de comunicação, o 3G e a Ethernet, em diferentes cenários de funcionamento.

Palavras-chave: Coleta de dados ambientais, telemetria, Raspberry Pi, *multihomed*.

Abstract

The gathering of weather data is an important tool on several research fields. However, in order to transmit data collected in remote environment and areas with difficult access, it is important that the transmission could be done through redundant communication channels. The development of this work is focused on creating an asynchronous communication interface for multihomed telemetry which is a fundamental part in the reliable data transmission. The multihomed communication consists in using different transmission technologies in order to increase the availability of the communication channel. The interface developed in this work has been built in a modular fashion allowing streamlined incorporation of different communication technologies. As result is presented a evaluation of the interface running on the Raspberry Pi platform using two different types of communications technologies, 3G and Ethernet, in different operating scenarios.

Keywords: Environmental data collection, telemetry, Raspberry Pi, multihomed.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

UDP	User Datagram Protocol
TCP	Transmission Control Protocol
ARP	Address Resolution Protocol
GSM	Global System for Mobile Communications
GPRS	General Packet Radio Service
UMTS	Universal Mobile Telecommunications Service
LTE	Long Term Evolution
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
MIMO	Multiple-Input/Multiple-Output Enhaced
OFDM	Orthogonal frequency-division multiplexing
Wi-Fi	Wireless Fidelity
RTT	Round-trip time
PPP	Point-to-point protocol
WLAN	Wireless Local Area Network

Sumário

1 Introdução

1.1 Cenário

A coleta de dados ambientais é fundamental para a previsão de fenômenos naturais. As informações obtidas nestas previsões podem ser aplicadas em diversas áreas. Na agricultura pode-se destacar a melhoria da eficiência do plantio, cultivo e colheita quando realizados em condições ambientais mais próximas às ideais. O trabalho de (Edson B. Teracine, 2009) demonstra um acréscimo de 5% a 20% na produtividade, proveniente da aplicação de previsões meteorológicas. Outra aplicação está relacionada a mitigação de catástrofes climáticas em grandes cidades como mostra o Capítulo 2 do Livro de (Carlos EM Tucci, 2005) “inundações em cidades urbanas decorrentes de fortes chuvas”.

No entanto, para que as previsões sejam precisas é essencial que não haja interrupção na aquisição dos dados. Um dos principais pontos de falha em sistemas de telemetria de dados ambientais é a interface de comunicação. É importante que o dispositivo faça a transferência confiável dos dados entre o cliente e o servidor por diferentes interfaces de rede e utilizando diferentes meios de transmissão.

No mercado existem outros dispositivos com estas mesmas características porém, em sua maioria são proprietários e possuem custo elevado de aquisição e manutenção. No entanto, com a popularização das plataformas de desenvolvimento de sistemas embarcados como o Arduino[1] e o Raspberry Pi[2] foi possível desenvolver uma plataforma completa de baixo custo que atende a vários dos requisitos necessários a coleta e transmissão eficiente de dados ambientais. Espera-se que a solução desenvolvida neste trabalho possa ser utilizada em outros cenários que apresentem características semelhantes.

A abordagem *multihomed* é uma das formas de aumentar a disponibilidade do canal de comunicação através da utilização de canais redundantes para transmissão dos dados. O uso de múltiplos canais exige a definição de um mecanismo de escolha do canal preferencial que ficará ativo e que só será substituído em caso de falha ou pela qualidade da canal.

Na literatura encontramos alguns trabalhos que abordam alguns destes critérios de seleção do melhor canal de comunicação. O trabalho de (Alberto Cortes Martin et al., 2011) apresenta diversos critérios de seleção como: informações sobre a largura

[1] Site do Arduino - arduino.cc
[2] Site da Raspberry Pi - www.raspberrypi.org

de banda e perda de pacotes do canal, segurança da rede e tempo de funcionamento sem interrupções. Ele propõe que a escolha da melhor interface de rede esteja dentro destes critérios para que além da garantia da integridade dos dados transmitidos o usuário possa ter uma experiência satisfatória.

A mudança da interface de rede utilizada durante uma transmissão de dados podem ocasionar diversos problemas. O trabalho de (Shahriar Nirjon et al., 2012) aborda alguns dos problemas com base na mudança de interface como: perda de dados, possíveis mensagens de erros e transtornos ao usuário. O maior problema da mudança de interface de rede é a perda de dados fazendo com que os dados não cheguem de forma íntegra ao seu destino. Para contornar esse problema, uma solução é o uso de *buffers*, desta forma, o conteúdo será armazenado até que os dados sejam enviados adequadamente.

Outro aspecto que deve ser considerado quando usamos múltiplos canais de comunicação é o aumento do consumo energético do dispositivo. Se o dispositivo estiver sendo alimentado por bateria a vida útil será reduzida como mostra o estudo de (LEE, 2011).

Existem diversas vantagens de se usar um dispositivo com múltiplas interfaces de comunicação (*multihomed*), a tese de doutorado de (Alberto Cortés Martín, 2012) aborda algumas destas vantagens. Dentre elas destacam-se resiliência, disponibilidade e maior tolerância a falhas.

1.2 Descrevendo o problema

É importante para os pesquisadores e empresas que trabalham direta ou indiretamente com dados ambientais saberem com antecedência como o ambiente irá se comportar para melhor prever seus investimentos. Contudo, para que estes dados sejam coletados e enviados da maneira adequada é essencial contar com um canal de comunicação confiável. As estações de coleta, muitas vezes, estão instaladas em locais remotos e de difícil acesso. Porém, com os avanços em tecnologias de comunicação, mesmo nestes lugares, é possível obter canais de comunicação via celular (GPRS, EDGE, 3G e 4G). Este tipo de tecnologia tem um custo bem mais baixo do que a comunicação via satélite propiciando uma maior utilização dos sistemas de telemetria.

Com a maior disseminação da filosofia *Open Source* e com a popularização de plataformas de desenvolvimento de sistemas embarcados surgiram no mercado dispositivos como o Raspberry Pi. O Raspberry Pi é um microcomputador desenvolvido pela Raspberry Pi Foundation, uma instituição sem fins lucrativos do Reino Unido, com o intuito de oferecer uma plataforma de baixo custo com grande foco em ensino. Seu pro-

pósito inicial foi o incentivo ao aprendizado da ciência da computação especialmente nas escolas. O Raspberry Pi possui várias interfaces de conexão com o mundo externo como por exemplo: saídas de áudio e vídeo, pinos de entrada e saída (GPIO) e interfaces de comunicação serial (UART, I2C e SPI). O Raspberry Pi também possui capacidade computacional para executar distribuições do sistema operacional Linux. Estas características fazem do Raspberry Pi uma excelente alternativa de plataforma para telemetria de dados a ambientais. Neste trabalho optou-se pelo uso da distribuição Raspbian[3] do sistema operacional Linux e pela versão B do Raspberry Pi.

1.3 Objetivo

Este trabalho tem como objetivo principal desenvolver e avaliar uma interface de comunicação assíncrona *multihomed* para telemetria de plataformas de coleta de dados ambientais de baixo custo.

Durante este desenvolvimento foi empregado conceitos e técnicas que abrangem as áreas de sistemas embarcados, redes de computadores e sistemas operacionais. De forma específica, pretende-se:

- Desenvolver a interface de comunicação;
- Compreender os requisitos existentes na coleta e transmissão de dados ambientais;
- Compreender as tecnologias usadas na transmissão como Ethernet, 3G e Wi-Fi;
- Testar e avaliar a interface de comunicação desenvolvida.

[3] Sistema operacional Raspbian - www.raspbian.org

1.4 Organização do documento

O presente trabalho está organizado em cinco capítulos dos quais o primeiro é a introdução e os outros quatro estão descritos abaixo:

- No capítulo 2 é descrita a fundamentação teórica onde serão apresentados os principais conceitos envolvidos no desenvolvimento do projeto;
- No capítulo 3 é apresentada a solução desenvolvida;
- No capítulo 4 é apresentada a modelagem matemática;
- No capítulo 5 e 6 são apresentados os cenários de teste da interface de comunicação e os resultados obtidos para cada uma das estratégias;
- No capítulo 7 são apresentadas as conclusões e os trabalhos futuros.

2 Fundamentação Teórica

Este capítulo apresenta os principais conceitos e componentes envolvidos no desenvolvimento e validação da interface de comunicação assíncrona *multihomed*.

2.1 Comunicação assíncrona *multihomed*

Uma comunicação é assíncrona quando o emissor e o receptor não precisam de um controle antes que as informações possam ser enviadas. Em um ambiente assíncrono o receptor tem que estar pronto para aceitar dados sempre que for necessário. (Douglas E. Comer, 2007)

Sendo assim ao usar uma comunicação assíncrona o remetente e o receptor não sincronizam antes de enviar cada informação e sempre o remetente precisar estar livre para novas requisições.

2.2 A pilha de protocolos da Internet - O modelo TCP/IP

Os protocolos utilizados na Internet estão organizados através de uma estrutura em camadas denominada modelo em camadas TCP/IP. Este modelo consiste em 5 camadas, cada camada é independente das demais ou seja tarefas associadas a uma camada pode ser modificada sem que as demais sofram qualquer alteração.

A camada mais alta é a de aplicação e a mais próxima do usuário final. A camada abaixo da aplicação é a de transporte, essa camada é responsável pela segmentação e reconstrução dos fluxos de dados provenientes de camadas superiores. A camada logo abaixo é a de rede, responsável pelo roteamento de datagramas de origem para o seu destino.

Abaixo da camada de rede encontra-se a camada de enlace responsável em fazer a transferência de dados entre vizinhos da rede. A primeira camada é a física que é responsável por definir os meios de acesso e os conectores físicos.(TANENBAUM, 2003)

2.3 *Socket*

O *socket* é uma interface de comunicação bidirecional, independente de sistema operacional, utilizada em arquiteturas cliente-servidor. Foi desenvolvida como parte do sistema operacional BSD UNIX. A Universidade da Califórnia, Berkeley desen-

volveu e distribuiu uma versão UNIX que continham protocolos de ligação interredes TCP/IP.(LEFFLER; KARELS; MCKUSICK, 1989)

O *socket* trabalha entre as camadas de transporte e aplicação do modelo TCP/IP, pois conversa com protocolos da camada de transporte, TCP e UDP, e com os protocolos da camada de aplicação, FTP, HTTP e SMTP (Douglas E. Comer, 2007). Uma vez que o *socket* é criado, é possível transmitir e receber informações a partir de comandos internos como *send* para enviar e *recv* para recebimento de informações. Ao final de uma comunicação o *socket* é fechado.

Para se transmitir uma mensagem através de uma rede TCP/IP, primeiramente, é preciso que seja verificado se existe alguma interface de rede (enlace) disponível e, necessariamente, este enlace precisa ter uma rota (*gateway*) para se comunicar com a internet. Em seguida, é preciso criar o *socket*. Para isso é necessário especificar o endereço IP do servidor de destino, a porta e o protocolo da camada de transporte que será utilizado, TCP ou UDP.

O TCP(*Transmission Control Protocol*) é um protocolo da camada de transporte que é orientado a conexão isto é, para um programa se comunicar com outro é necessário primeiro solicitar uma conexão para depois utilizar. Outra característica do TCP é que ele é confiável, ele garante que os dados enviados sejam entregues sem perdas e na mesma ordem em que foram enviados.

O UDP(*User Datagram Protocol*) utiliza o paradigma de comunicação sem conexão, ou seja um programa que usa o UDP não precisa preestabelecer comunicação antes de enviar os dados. Uma característica importante é que se dois programas estiverem enviando dados e eles pararem, nenhum outro dado será trocado pelos programas pois, o UDP não usa mensagens de controle.

2.4 Implementação do envio de pacotes internamente no linux

Igualmente aos protocolos de rede a Figura 1 mostra que o linux implementa os endereços de protocolos de internet como um conjunto de softwares de camadas interligados. O BSD socket é uma interface genérica que suporta várias formas de comunicação em redes e também é um mecanismo de comunicação entre processos.

Apoiando a camada BSD socket está a camada INET socket, que gerência os pontos finais de comunicação para os protocolos TCP e UDP baseado no IP.

O UDP é um protocolo sem conexão, enquanto TCP é um protocolo orientado a conexão. Quando os pacotes são transmitidos por UDP, o linux não garante que eles cheguem íntegros ao seu destino. Quando os pacotes são transmitidos por TCP, eles tem um número de sequência e tanto o emissor quanto o receptor certificam-se de que

os dados transmitidos são recebidos na ordem correta.

A camada IP contém os código de implementação do protocolo de Internet. Este código esta no início do cabeçalho IP, e quando transmitidos o protocolo IP entende como rotear os pacotes recebidos.

Abaixo da camada IP, encontra-se todos os dispositivos de rede que o linux suporta, por exemplo PPP e Ethernet. Os dispositivos de rede nem sempre representam dispositivos físicos; alguns, como o dispositivo de *loopback* são dispositivos de software.

Por fim, o protocolo ARP(*Address Resolution Protocol*) fica entre a camada IP e os dispositivos de rede. O protocolo ARP tem o papel de fazer as traduções de endereços IP em endereços físicos de hardware, tais como endereços ethernet. O ip precisa dessa tradução pouco antes de passar os dados para o dispositivo de rede para transmissão.(RUSLING, 1999)

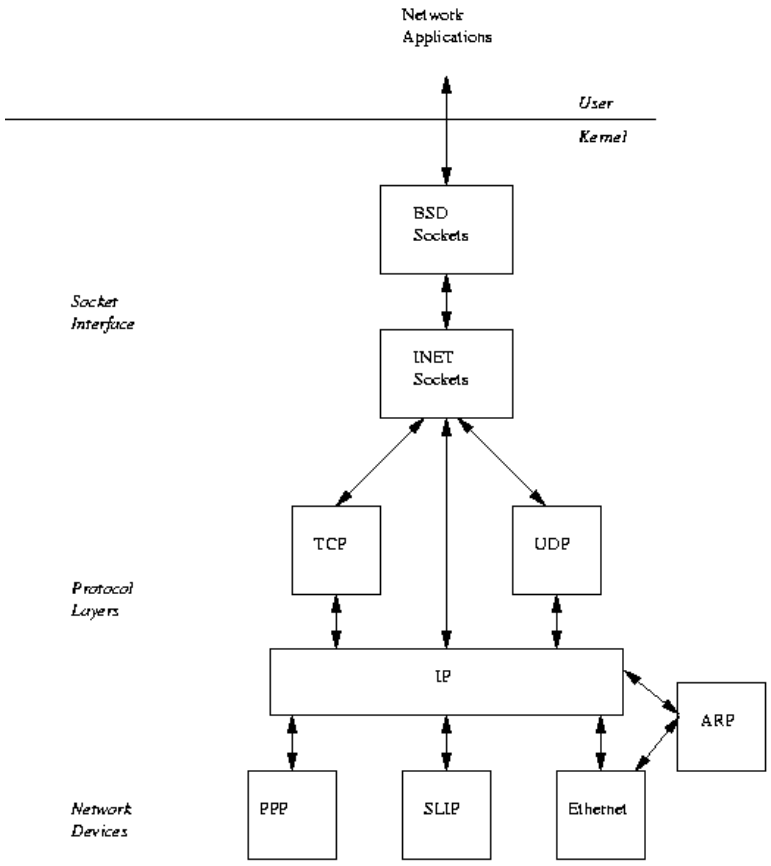

Figura 1 – Camadas de redes linux
(RUSLING, 1999)

2.5 Protocolos da camada de enlace de dados

A camada de enlace tem como objetivo oferecer serviços a camada de rede como: enquadramento de pacotes, acesso ao enlace, entrega confiável, controle de fluxo, detecção de erros e correção de erros. A camada de enlace faz a ligação entre as camada física e a de rede, existem diversos protocolos e equipamentos que atuam nesta camada. A seguir são descritos com mais detalhes alguns dos protocolos de enlace utilizados neste trabalho.

2.5.1 Ethernet

O padrão Ethernet baseado na norma IEEE 802.3 é um exemplo de uma tecnologia que atua na camada de enlace, com transmissão de dados baseado em pacotes. O modo de transmissão half-duplex utiliza a técnica CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*) para reduzir as colisões e as detectar quando elas acontecem, permitindo que dispositivos que compartilham o mesmo meio físico transmitam dados simultaneamente sem haver colisões.

O modo de transmissão do Ethernet possui dois tipos de configurações bidirecionais podendo ser *half-duplex*, onde cada ativo transmite ou recebe os dados de forma que não aconteca transmissões simultâneas e a *full-duplex* em que os ativos podem transmitir ou receber informações simultaneamente.

O padrão *10-Gigabit Ethernet* é a versão mais recente do padrão *Ethernet* que opera a 10Gbps, segue as mesmas características, como detecção de colisões, regras de repetidores e aceita transmissão *half-duplex* e *full-duplex*. (Marco A. Filippetti, 2014)

2.5.2 802.11 (Wi-Fi)

O protocolo 802.11 (Wi-Fi) faz a comunicação entre a camada física e de enlace assim como o *Ethernet*. A IEEE(*Institute of Electrical and Electronics Engineers*), uma organização que foi criada para padronizar tecnologias e protocolos relacionados a telecomunicações, aprovou vários padrões relacionados à redes sem fio. Os padrões mais comuns que definem uma Wi-fi são: 802.11, 802.11b , 802.11a, 802.11g, 802.11n e 802.11ac.

O padrão 802.11 com taxas de transferência entre 1 e 2 Mbps, pode utilizar infravermelho ou sinais de rádio.

O 802.11b tem taxas de transferência entre 5,5 Mbps e 11 Mbps e opera na faixa de frequência de 2,5 GHz.

O 802.11a utiliza a tecnologia OFDM (*Orthogonal frequency-division multiplexing*) que divide o canal de comunicação entre um número de subcanais, dividindo a

informação em cada um deles. Como cada canal atua de forma independente, não acontece interferência entre os canais. Possui taxa de transferência máxima de 54 Mbps operando na faixa de frequência de 5 GHz.

O 802.11g combina as melhores características dos padrões 802.11a e 802.11b pois trabalha com a modulação OFDM, opera em uma frequência de 2,4 GHz e atinge uma taxa de transferência de 54 Mbps.

O 802.11n é compatível com as versões anteriores, pode operar nas frequências de 2,4 GHz e 5 GHz. A maior diferença entre os padrões anteriores é que este padrão introduz o MIMO (*Multiple-Input, Multiple-Output Enhaced*), podendo atingir uma taxa de transferência de 600 Mbps com o uso de 4 antenas receptoras e de 4 antenas transmissoras.

O padrão 802.11ac pode alcançar uma velocidade de 1,3 Gbps utilizando a frequancia de 5 GHz e pode chegar a uma velocidade de 600 Mpbs quando opera com múltiplas antenas. (Marcos Flávio Araújo Assunção, 2013)

2.5.3 Protocolos de telefonia móvel

A evolução dos protocolos de telefonia móvel foi dividida em 5 gerações. A primeira geração é composta por sistemas analógicos com funcionalidades básicas como *roaming* e *handover* entre células.(KRANSMO, 2003)

A segunda geração (2G) tem como características a utilização de comutação de circuitos, fornecendo serviços auxiliares ao usuário como mensagens e identificador de chamadas. Um exemplo de padrão da segunda geração é o GSM(*Global System for Mobile Commmunications*).

A terceira geração (3G) tem como características a utilização de comutação de pacotes na transmissão dos dados. Tem como princípios prover tanto serviço de voz quando serviço de dados. Um exemplo de tecnologia que atua na terceira geração é o UMTS(*Universal Mobile Telecommunications Service*) que tem como características o tráfego de diferentes mídias a partir de um mesmo ponto de transmissão.

A quarta geração (4G), com taxa de transmissão maiores que 20 Mbps, é chamada de evolução de taxas elevadas, tem como padrão o LTE(*Long Term Evolution*). Um diferencial é que esse padrão prioriza a transmissão de dados em relação a transmissão de voz(HOLMA; TOSKALA, 2009)

2.6 Protocolos da camada de aplicação

A camada de aplicação tem como objetivo fornecer a interface entre as aplicações de comunicação e a rede por onde os dados serão transmitidos. Os protocolos

de comunicação são utilizados para a troca de mensagens entre dois computadores. A seguir serão descritos todos os protocolos da camada de aplicação que serão usados na transmissão dos dados.

2.6.1 Protocolo FTP

O protocolo FTP(*File Transfer Protocol*) é usado para a troca de arquivos através da Internet. É baseado na arquitetura Cliente-Servidor, onde o cliente fornece informações de acesso. Após ser passado as informações de segurança e validado pelo servidor é possível ser feita a transferência dos arquivos local para o servidor remoto.

Como mostra a Figura 2 o usuário interage com o FTP por meio de uma interface. É necessário que seja fornecido o endereço do servidor FTP, que faz com que o processo cliente FTP estabeleça uma conexão TCP com o servidor remoto. Depois de passado as informações é autorizado a transferência dos arquivos do cliente para o servidor remoto.

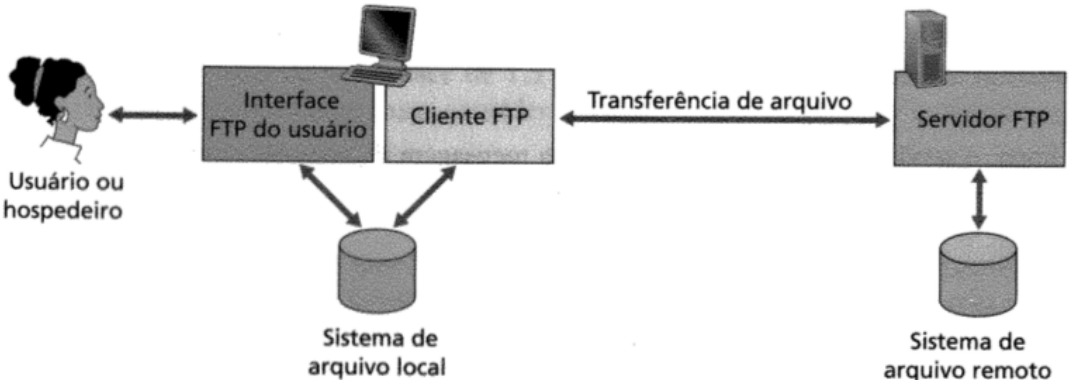

Figura 2 – Exemplo de envio de arquivos com o protocolo FTP
(James F. Kurose; Keith W. Ross, 2005)

2.6.2 Protocolo SMTP

O protocolo SMTP(*Simple Mail Transfer Protocol*) é usado para envio de mensagens de e-mail através da rede. Ele permite transferência de um correio a outro com um serviço ponto a ponto utilizando um canal de dados confiável, o SMTP é um protocolo apenas de envio.

A Figura 3 mostra o exemplo do sistema de correios na internet. Neste exemplo existem 3 componentes na transferência de correio eletrônico. Agente de usuário, servidores de correio e SMTP. Quando um usuário quer enviar um e-mail para outro usuário, é o agente de usuário que permite que a mensagem seja escrita, armazenada e enviada. Depois que o e-mail é enviado ele vai para uma fila de envio, quando o destinatário quer visualizar o e-mail recebido ele extrai da caixa de correio do seu servidor de correio.

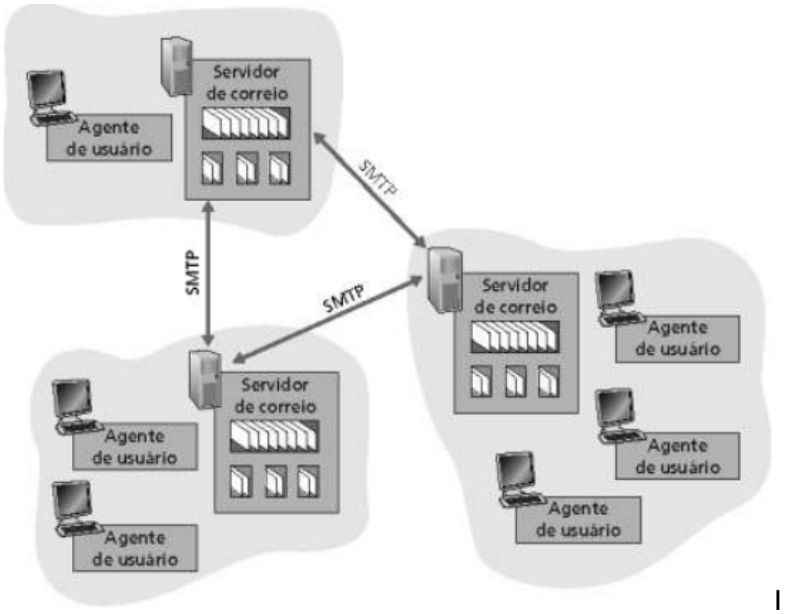

Figura 3 – Exemplo de envio de E-mail com o protocolo SMTP
(James F. Kurose; Keith W. Ross, 2005)

2.6.3 Protocolo HTTP

O protocolo HTTP(*Hyper Text Transfer Protocol*) é um protocolo da camada de aplicação. Ele é executando em dois programas distintos, no cliente que solicita, recebe e apresenta os objetos e o servidor que envia sobe demanda. O cliente e o servidor "conversam" trocando mensagens HTTP.

O funcionamento é ilustrado na Figura 4 quando o cliente faz uma requisição de uma página web, é aberto um socket com uma conexão TCP na porta 80 do servidor. O *browser* envia para o servidor mensagens de requisição HTTP e o servidor *web* responde a requisição em mensagens HTTP que contém os objetos que serão exibidos no *browser*.

Figura 4 – Exemplo de requisição o pelo protocolo HTTP
(James F. Kurose; Keith W. Ross, 2005)

3 Desenvolvimento da interface de comunicação assíncrona *multihomed*

3.1 Visão geral da interface *multihomed*

Para que se tenha um melhor uso de todas as interfaces de rede de um dispositivo é necessário que exista uma forma de gerenciar todas elas, de modo a se obter a maior disponibilidade de transferência possível.

Mas para que este controle seja realizado é necessário que o módulo tenha a gerencia de todas as interfaces e possa controlar os protocolos envolvidos, pois será a partir das interfaces e dos protocolos que os dados serão enviados.

Pode-se observar que um gerenciamento adequado das interfaces fará com que elas fiquem menos tempo ociosas e a transmissão das informações tenha uma maior probabilidade de chegar íntegra ao seu destino. Implementar um gerenciador de interfaces de rede em uma arquitetura de coleta de dados ambientais fará com que as informações coletadas possam ser encaminhadas por mais de uma rota de saída possível.

Neste contexto, o trabalho proposto consiste no desenvolvimento e avaliação de uma interface de comunicação baseada na arquitetura cliente-servidor que é utilizada em um dispositivo móvel de coleta de dados para a transferência confiável dos dados para um servidor remoto através de diferentes interfaces de rede.

A interface desenvolvida é capaz de utilizar para a transmissão vários meios de comunicação como: GPRS, Wi-Fi, Ethernet e conexão via satélite, porém nesse estudo foram utilizadas duas, o 3G que utiliza a infraestrutura da rede celular e a Ethernet que irá utilizar uma conexão através de um cabo par trançado. A escolha do canal de comunicação a ser utilizado será realizada pela interface de comunicação através de sua política de seleção do canal ativo.

Outro aspecto importante é que a interface de comunicação assíncrona *multihomed* foi desenvolvida baseada no princípio de independência na transmissão dos dados. Sempre que a aplicação de coleta necessita enviar os dados, ela apenas informa a interface de comunicação o que será enviado e a interface envia sem qualquer dependência entre a aplicação de coleta de dados e a interface de comunicação. Além disso, a interface de comunicação gerencia vários canais redundantes de comunicação para aumentar a disponibilidade no envio dos dados.

A interface de comunicação assíncrona foi desenvolvida na linguagem de pro-

gramação Python[1]. Todo o código desenvolvido da interface, bem como os códigos de teste, estão disponíveis no GitHub[2] através do endereço http://bit.ly/1IGQ06r.

3.2 Interface de comunicação assíncrona *multihomed*

Com a finalidade de escolher a interface de rede que estiver ativa, a interface de comunicação será uma parte fundamental no processo de coleta de dados ambientais, pois é a partir dele que serão transmitidos os dados que foram coletados em regiões remotas através da internet.

Uma visão geral da arquitetura da coleta de dados é apresentada na Figura 5. Os sensores irão capturar os dados ambientais que serão armazenados na aplicação de coleta de dados. Esta, por sua vez, irá chamar a interface de comunicação que terá o papel de transmitir os dados coletados e armazenados através da Internet. O envio dos dados é totalmente transparente para aplicação de coleta.

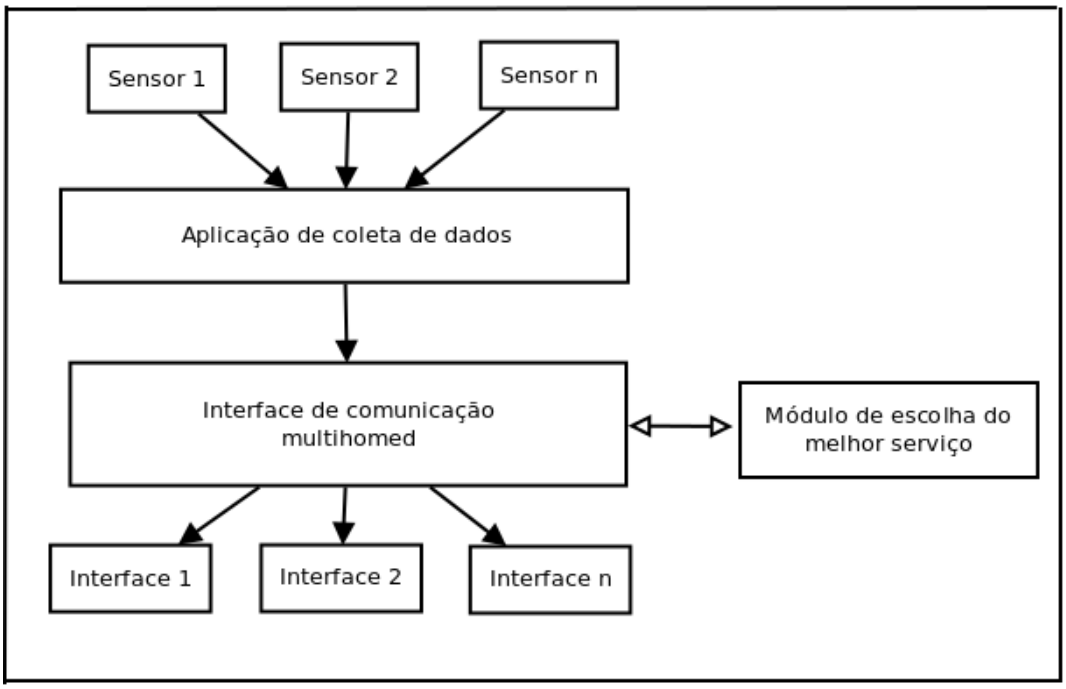

Figura 5 – Arquitetura da coleta e transmissão dos dados ambientais

Podemos observar que a quantidade de sensores (variáveis ambientais) na plataforma de coleta é configurável, da mesma forma a quantidade de interfaces de rede integradas com o Raspberry Pi também é configurável podendo ser integrado quantas forem necessárias para aumentar a disponibilidade na transmissão das informações que foram coletadas.

[1] Python - www.python.org
[2] GitHub - www.github.com

A interface de comunicação *multihomed* tem como principal objetivo escolher a interface a partir de 2 estratégias. A primeira é apenas observar se o canal esta operante ou inoperante e a 2ª estratégia é observar a qualidade do canal através do módulo de escolha do melhor serviço. Após a escolha da interface será enviados os dados coletados por ela como mostra a Figura 6. Os dados serão armazenados na aplicação de coleta de dados através de arquivos e uma referência para estes arquivos será enviada para a interface de comunicação. A gerência de envio tem a finalidade de informar se existe algum enlace disponível como também informar se a conexão está ativa.

Para que a interface de comunicação seja configurada corretamente, os parâmetros de configuração dos enlaces de comunicação, dos servidores e dos protocolos que serão utilizados, deverão ser previamente definidos pelo usuário a partir de um arquivo de configuração. Por exemplo, caso o usuário deseje utilizar um servidor FTP para enviar os dados de coleta, deverão ser definidos no arquivo de configuração o endereço do servidor FTP, o *login* e senha para acesso.

As referências para os arquivos com os dados coletados, especificadas pela aplicação de coleta de dados, serão armazenadas pela interface de comunicação nas filas de envio. No momento do envio a interface de comunicação irá buscar os arquivos e enviá-los.

Na arquitetura interna do módulo existe uma fila de envio para cada protocolo, pois o envio através de cada um dos protocolos é gerenciado individualmente. A interface de comunicação baseada na referência passada pela aplicação de coleta, irá buscar os arquivos e enviar por cada um dos protocolos.

A interface de comunicação gerencia cada uma das filas através de *polling*, ou seja, em intervalos predeterminados de tempo as filas são verificadas para determinar se existem dados disponíveis para envio. Caso o envio de todos os arquivos da fila tenha sido confirmado, todas as referências da fila em questão são apagadas. Caso não haja confirmação de envio de algum dos arquivos, a referência para este arquivo permanece na fila até que o envio seja realizado com sucesso.

Todos os dados coletados são armazenados em arquivos. A opção por utilizar arquivos levou em consideração a persistência dos dados em caso de falhas de energia e a simplicidade de se trabalhar com eles.

No momento em que a gerente de envio informa que a interface e o enlace estão prontos, é aberto um *socket* para cada um dos protocolos em seus respectivos servidores de destino, podendo ser por meio de qualquer uma das interfaces, com a finalidade de minimizar a perda de dados.

É importante enfatizar a função do gerente de envio, pois é ele quem verifica a

disponibilidade dos enlaces e dos *links* dentro da interface de comunicação.

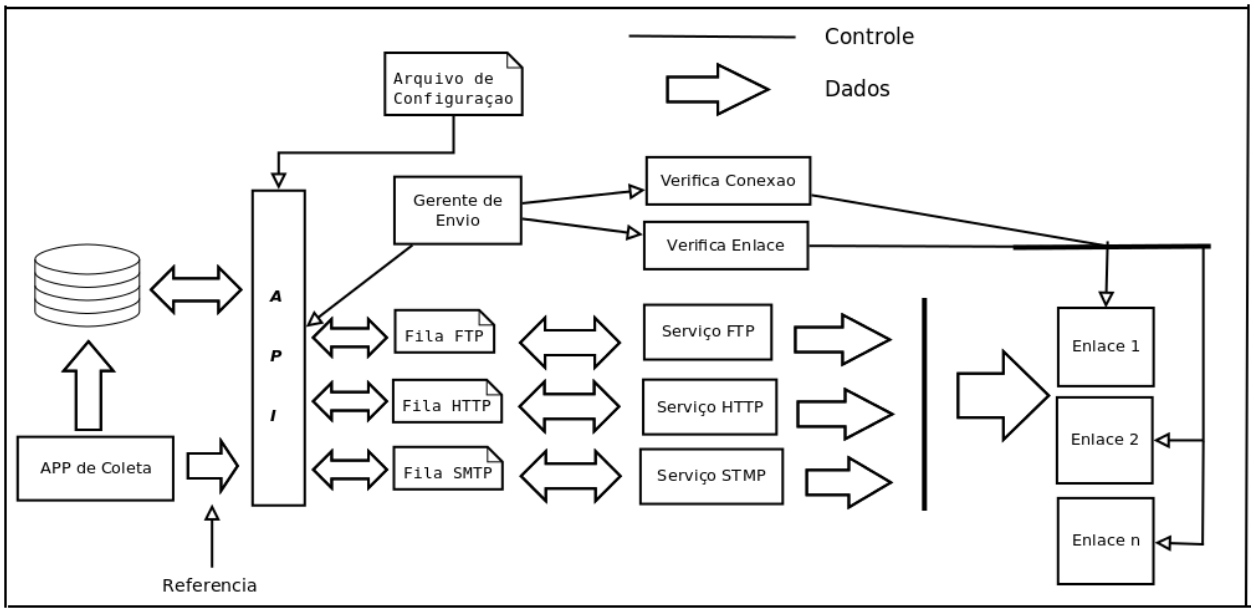

Figura 6 – Arquitetura interna do funcionamento do modulo de comunicacão

A interface de comunicação assíncrona utilizada nos testes foi configurada para utilizar três protocolos de comunicação distintos: FTP, SMTP e HTTP. A escolha destes protocolos está relacionada a ampla utilização destes protocolos nas soluções comerciais disponíveis no mercado.

Quando os dados são enviados pelo protocolo *FTP* será armazenado no servidor FTP os mesmos arquivos criados pela aplicação de coleta. Utilizando o protocolo *SMTP* os arquivos coletados serão enviados como anexo de um e-mail enviado pelo Raspberry Pi. Desta forma, estes arquivos podem ser acessados por qualquer dispositivo que tenha acesso ao e-mail de destino. Por fim, quando é utilizado o protocolo *HTTP*, os dados são enviados para a plataforma *Thingspeak* [3] capaz de gerar gráficos em tempo real dos dados coletados.

Esta funcionalidade da plataforma permite que os dados percorram caminhos distintos aumentando a possibilidade de que todos os arquivos sejam transmitidos e que estejam íntegros. Outra vantagem é a possibilidade da redução do tempo de espera pois, não haverá dependência de apenas um protocolo de comunicação.

A Figura 7 mostra o fluxo de funcionamento da interface de comunicação na primeira estratégia. No primeiro momento a aplicação de coleta instancia a interface de comunicação. Em seguida, a interface irá ler os parâmetros do arquivo de configuração e configurar adequadamente a interface de comunicação. Após a configuração, a *thread* que é responsável pela gerência da interface de comunicação é inicializada. Por fim, o módulo estará pronto para receber chamadas da função *sendArquivo()* que

[3] Thingspeak - www.thingspeak.com

é responsável por inserir as referências dos arquivos coletados nas filas internas da interface para cada um dos protocolos de comunicação utilizados.

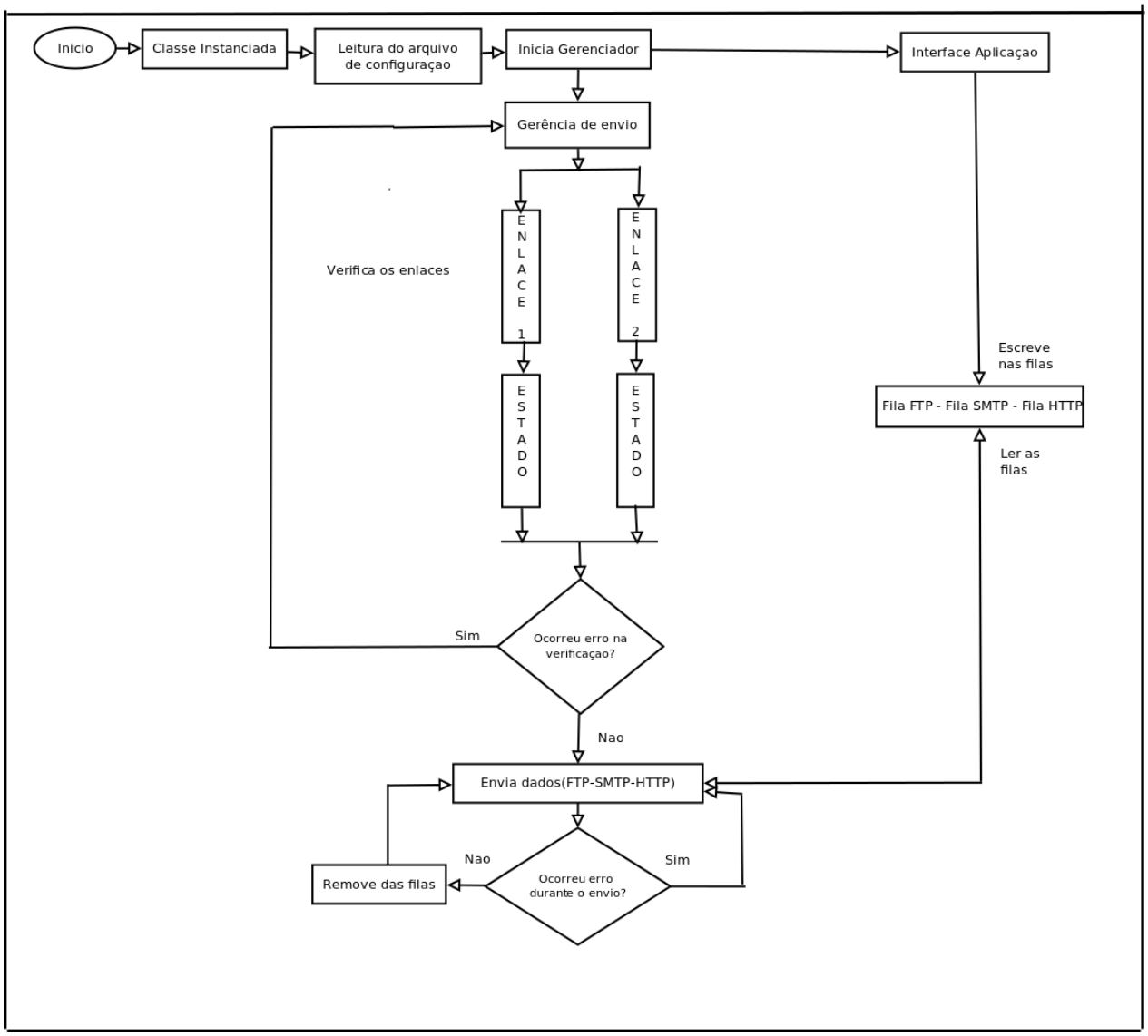

Figura 7 – Fluxo do funcionamento da 1ª estratégia

A Figura 8 mostra o fluxo da interface de comunicação na segunda estratégia. Com funcionamento igual ao primeiro fluxo a única mudança será no momento da escolha do melhor canal para a transmissão dos dados. Esta estratégia tem como base analisar 2 parâmetros. O primeiro é o RTT(*Round-trip time)* que é o tempo em que um pacote seja enviado de um computador de origem para um computador de destino. O segundo é o se o canal e o serviço estão com perdas de pacotes.

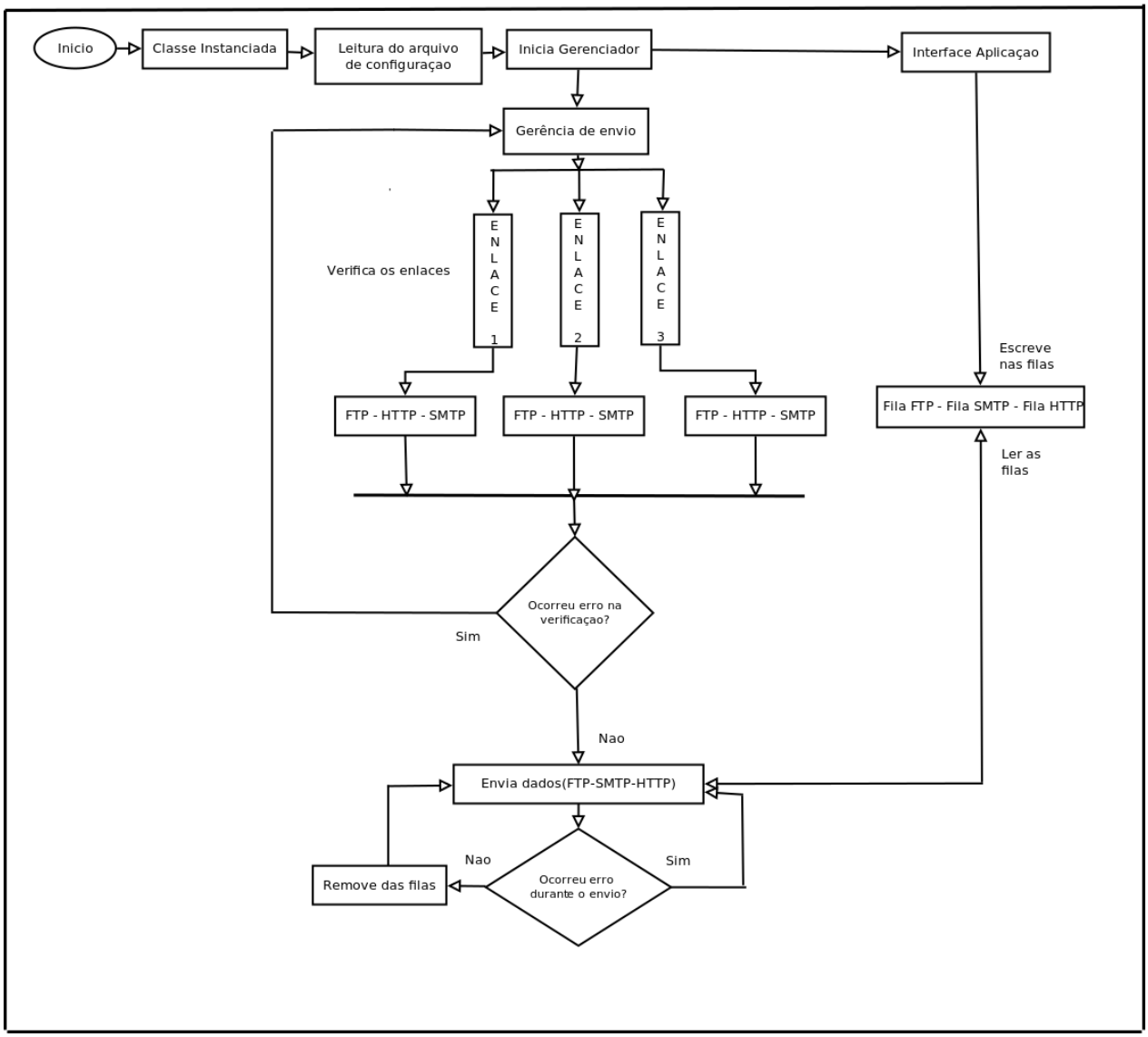

Figura 8 – Fluxo do funcionamento da 2ª estratégia

4 Modelagem matemática da 2ª estratégia

O modelo matemático que será apresentado a seguir tem como objetivo, encontrar o melhor índice de cada protocolo e cada serviço com a finalidade de dar maior disponibilidade a interface e também a diminuição do consumo energético. Porém todas as restrições de encontrar o melhor índice devem ser atendidas. Estas restrições e os parâmetros empregados no modelo de programação linear inteira são esclarecidos a seguir.

4.1 Formulação do problema

4.1.1 Variáveis do problema

Os conjuntos de variáveis de entrada para a escolha do melhor serviço são definidas como:

- Conjunto de Enlaces = {1,...,E}
- Conjunto de Protocolos = {1,...,P}

O numero total de variaveis de decisão para os 3 enlaces é definido pelo produto cartesiano dos conjuntos de entrada do problema X(Enlace, Protocolos). São 3 enlaces, 3 protocolos, totalizando 9 variáveis de decisão.

4.1.2 Função objetivo do problema

A funcão objetivo deve dar preferência a escolha do melhor serviço considerando a premissa abaixo.

- Minimizar a média do RTT de cada enlace e protocolo somado com o fator de erro de cada enlace e cada protocolo.

4.1.3 Restrições do problema

As restricões do modelo estão descritas a seguir:

- Cada interface deve ter apenas um protocolo.
- Não deve ser ultrapassado o tamanho da janela.

4.1.4 Modelagem matemática do problema

Para minimizar a escolha do melhor serviço é necessário realizar primeiro o cálculo da média do RTT para cada enlace e protocolo, como mostra a equação (4.1).

$$\overline{RTT}_t = \alpha\overline{RTT}_(t-1) + (1-\alpha)RTT_t \tag{4.1}$$

Onde:

$\overline{RTT}_t-_1$, é a média do RTT no instante anterior

RTT_t, é RTT no instante atual.

α, é o fator de amortecimento.

E após, calcular o fator de erro de cada enlace e cada protocolo como mostra a equação (4.2).

$$f = p(2^{tam} - 1) \tag{4.2}$$

Onde:

p , é o fator de peso para descarte do serviço.

tam , é o tamanho da janela enquanto o erro irá permanecer no serviço.

Minimizar:

$$\mathbb{Z} = \sum_{e,p} \left(\overline{RTT}_(e,p) + f_(e,p)\right) x_(e,p) \tag{4.3}$$

Onde:

e , é o índice do conjunto de enlaces.

p , é o índice do conjunto de protocolos.

$\overline{RTT}_{e,p}$, é o RTT médio de 1 enlace e 1 protocolo.

$f_{e,p}$, é o fator de erro de 1 enlace e 1 protocolo.

$x_{e,p} \in \{0, 1\}$, é 1, se existir 1 enlace e 1 protocolo. 0, caso contrário.

Sujeito a restrições:

- Cada enlace deve ter apenas um protocolo

$$\sum x_{e,p} \leq 1, \forall_(e,p) \tag{4.4}$$

- Não deve ser ultrapassado o tamanho da janela.

$$f_(e,p)x_(e,p) \leqslant p\left(2^{tam}-1\right)\forall_(e,p) \tag{4.5}$$

5 Descrição dos testes da 1ª estratégia

Para a realização dos testes na interface de comunicação *multihomed* foram definidos 4 cenários distintos. Em todos estes cenários as interfaces de comunicação ficaram inativas por 30 segundos e só a partir daí os testes tiveram início. Os testes no módulo foram automatizados com dados ambientais reais coletados previamente. Os dados foram enviados a partir de cada um dos cenários especificados, e para fazer a medição dos resultados utilizamos as seguintes métricas: análise da integridade dos dados recebidos, tempo de envio e tempo de envio quando há falhas.

Os primeiros testes utilizaram servidores *SMTP* e *FTP*. A massa de dados utilizada nos testes correspondem a cerca de 5 dias de coletas com uma coleta feita a cada hora. Desta forma, não se pode perder nenhum dos dados enviados pois, cada um dos arquivos contém informações referentes a pelo menos uma hora de coleta. Após o envio dos dados, a própria plataforma Raspberry Pi realiza o download de todos os arquivos, tanto do servidor *SMTP* quanto do servidor *FTP*.

Após o download é feita a verificação da integridade a partir da análise da soma de verificação MD5, um algoritmo de *HASH* para verificação da integridade de arquivos. Esta soma de verificação gera uma chave única de 128 bits. Para confirmar a integridade é suficiente comparar as chaves geradas através do arquivo antes e depois do envio. Se as chaves forem iguais, o arquivo é idêntico(RIVEST et al., 1992).

A seguir serão descritos em detalhes os cenários avaliados:

5.1 Cenário 1

5.1.1 Descrição

Neste primeiro cenário iremos verificar uma interface ativa de cada vez como mostra a Figura 9. Primeiro testaremos a interface eth0 com o padrão Ethernet e logo após a interface ppp0 com a tecnologia 3G. Como no início dos testes as interfaces estão desativadas o módulo irá escrever no *buffer* até que a primeira interface se encontre ativa. Assim que a interface de comunicação for ativada os dados dos arquivos armazenados no *buffer* começarão a ser enviados.

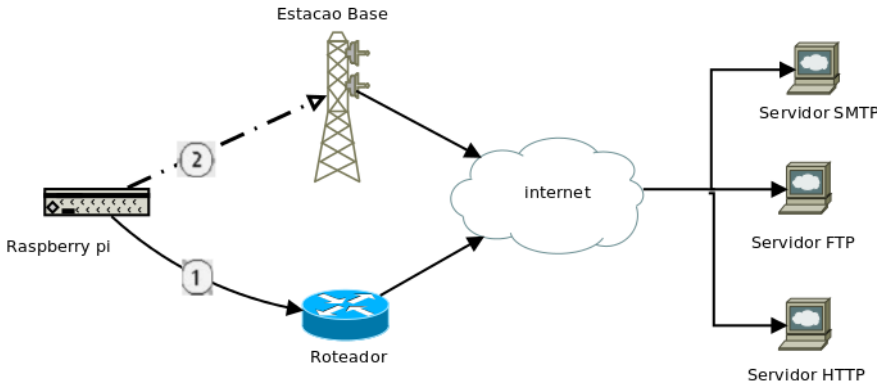

Figura 9 – Cenário enviando dados usando uma interface por vez

5.1.2 Objetivo

O objetivo deste teste é analisar o tempo de envio que cada protocolo e enlace leva para sair do seu destino e chegar no destino

5.1.3 Resultados

A partir da Tabela 1 que contém informações referente ao tempo gasto no envio dos arquivos pelo protocolo SMTP podemos observar que o envio pela tecnologia Ethernet é mais eficiente.

Tabela 1 – Tempo de envio dos dados pelo protocolo SMTP usando as interfaces Ethernet e 3G

Nome do arquivo	Duração do envio (Ethernet)	Duração do envio (3G)
1 Anexo com 10 arquivos	3.721 s	4.946 s
Total	3.721 s	4.946 s

Analisando o protocolo FTP, a tecnologia Ethernet também é mais rápida que a 3G, enquanto o protocolo FTP pela Ethernet envia os 10 arquivos em 8,9 segundos, pela 3G esse tempo é de 19,6 segundos, aumentando em mais de 100% o tempo de envio dos arquivos, como podemos verificar na Tabela 2.

Tabela 2 – Tempo de envio dos dados pelo protocolo FTP usando as interfaces Ethernet e 3G

Nome do arquivo	Duração do envio (Ethernet)	Duração do envio (3G)
1 Table1 2015-06-13 04-00-00.dat	0.945 s	3.478 s
2 Table1 2015-06-13 03-00-00.dat	0.896 s	1.829 s
3 Table1 2015-06-13 02-00-00.dat	0.899 s	1.648 s
4 Table1 2015-06-13 01-00-00.dat	0.897 s	1.897 s
5 Table1 2015-06-13 00-00-00.dat	0.895 s	1.719 s
6 Table1 2015-06-12 23-00-00.dat	0.896 s	1.919 s
7 Table1 2015-06-12 22-00-00.dat	0.897 s	1.747 s
8 Table1 2015-06-12 21-00-00.dat	0.893 s	1.569 s
9 Table1 2015-06-12 20-00-00.dat	0.892 s	1.670 s
10Table1 2015-06-12 19-00-00.dat	0.872 s	2.138 s
Total	8.9 s	19.6 s

5.2 Cenário 2

5.2.1 Descrição

Neste cenário as duas interfaces de rede estão ativas simultaneamente. Após os dados terem sido enviados verificamos qual interface foi utilizada para o envio. Com base no sistema operacional quando mais de uma interface está ativa, todas as interfaces se tornam *default*, desse modo a transmissão dos dados será feita pela interface que foi ativada por último, a interface de comunicação não tem controle sobre essa escolha.

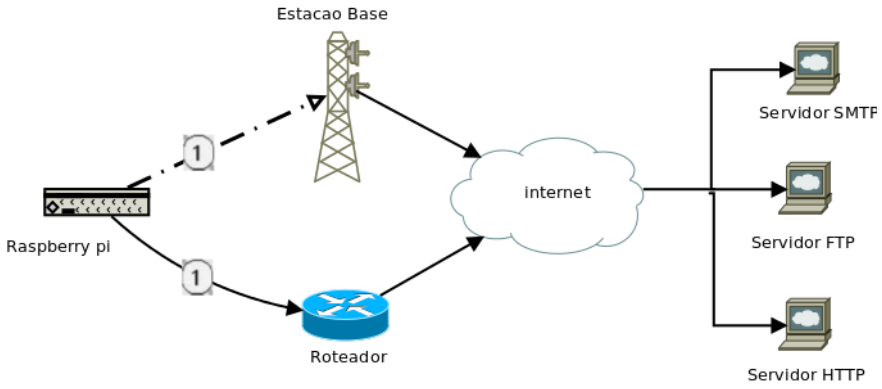

Figura 10 – Cenário com duas interface ativas

5.2.2 Objetivo

O objetivo deste cenário é visualizar por qual interface os dados serão transmitidos, visto que o equipamento estará com duas interfaces de rede ativa.

5.2.3 Resultados

Na Figura 11 podemos observar que na tabela de roteamento constam as duas interfaces de rede, ETH0 e PPP0, com suas respectivas rotas 192.168.2.1 e 10.64.64.64.

Na Figura 12 pode-se observar inicialmente a verificação da interface de rede e que as duas se encontram ativas. Em seguida é feito o teste de conectividade nas duas interfaces: a interface ETH0 com rota 192.168.2.1 tem o IP 192.168.2.108; e a interface PPP0 com rota 10.64.64.64 tem o IP 179.173.85.71. No final verifica-se por qual das interfaces foi realizada a comunicação.

```
root@raspberrypi:~/projeto# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.2.1     0.0.0.0         UG    0      0        0 eth0
0.0.0.0         10.64.64.64     0.0.0.0         UG    0      0        0 ppp0
10.64.64.64     0.0.0.0         255.255.255.255 UH    0      0        0 ppp0
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

Figura 11 – Tabela de rotas

```
lo
eth0
ppp0
Interface Ativa
------------------------------------------------------------------------
PING 8.8.8.8 (8.8.8.8) from 192.168.2.107 eth0: 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_req=1 ttl=52 time=55.3 ms
64 bytes from 8.8.8.8: icmp_req=2 ttl=52 time=55.3 ms
64 bytes from 8.8.8.8: icmp_req=3 ttl=52 time=55.1 ms
64 bytes from 8.8.8.8: icmp_req=4 ttl=52 time=54.6 ms
64 bytes from 8.8.8.8: icmp_req=5 ttl=52 time=54.9 ms

--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 54.636/55.070/55.360/0.372 ms
------------------------------------------------------------------------
PING 8.8.8.8 (8.8.8.8) from 179.173.85.71 ppp0: 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_req=1 ttl=40 time=1038 ms
64 bytes from 8.8.8.8: icmp_req=2 ttl=40 time=688 ms
64 bytes from 8.8.8.8: icmp_req=3 ttl=40 time=498 ms
64 bytes from 8.8.8.8: icmp_req=5 ttl=40 time=478 ms

--- 8.8.8.8 ping statistics ---
5 packets transmitted, 4 received, 20% packet loss, time 4000ms
rtt min/avg/max/mdev = 478.303/675.904/1038.556/224.901 ms, pipe 2
Conexao Verifica e OK
------------------------------------------------------------------------
Saindo SMTP pelo 192.168.2.107
Saindo FTP pelo 192.168.2.107
```

Figura 12 – Gerenciamento de envio e rota de saída

5.3 Cenário 3

5.3.1 Descrição

No cenário 3 os testes serão realizados a partir de um longo período de inatividade das interfaces de comunicação (24h).

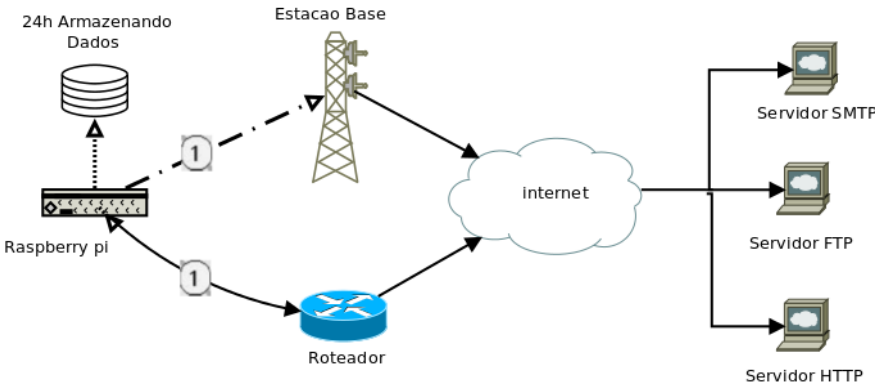

Figura 13 – Cenário com as interfaces inativas por um longo período

5.3.2 Objetivo

O objetivo deste cenário é analisar que mesmo o equipamento não tendo interface ativa, os dados não serão perdidos.

5.3.3 Resultados

A cada uma hora será feito um envio de um arquivo. Como não existe rota para enviar os dados os as referências para os arquivos ficarão armazenadas nas filas como ilustrado na Figura 13. A gerência de envio tem como função a verificação dos enlaces e a checagem dos links, essa verificação é feita de forma constante para que quando as interfaces ficarem ativas o módulo consiga enviar os dados.

Após a ativação do enlace e a validação da interface de comunicação, a interface irá enviar todos os arquivos referenciados nas filas de envio de cada protocolo, como mostra a Figura 14.

```
lo
eth0
Interface Ativa
------------------------------------------------------------------------
PING 8.8.8.8 (8.8.8.8) from 192.168.2.107 eth0: 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_req=1 ttl=52 time=47.8 ms
64 bytes from 8.8.8.8: icmp_req=2 ttl=52 time=47.5 ms
64 bytes from 8.8.8.8: icmp_req=3 ttl=52 time=47.2 ms
64 bytes from 8.8.8.8: icmp_req=4 ttl=52 time=47.8 ms
64 bytes from 8.8.8.8: icmp_req=5 ttl=52 time=47.6 ms

--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 47.215/47.631/47.859/0.268 ms
------------------------------------------------------------------------
ping: unknown iface ppp0
Conexao Verifica e OK
------------------------------------------------------------------------
Saindo SMTP pelo 192.168.2.107
Saindo FTP pelo 192.168.2.107
```

Figura 14 – Verificação da interface e envio dos dados

5.4 Cenário 4

5.4.1 Descrição

No último cenário será feito um teste enviando um arquivo de grande dimensão e será realizada uma interrupção de um dos canais do envio durante a transferência.

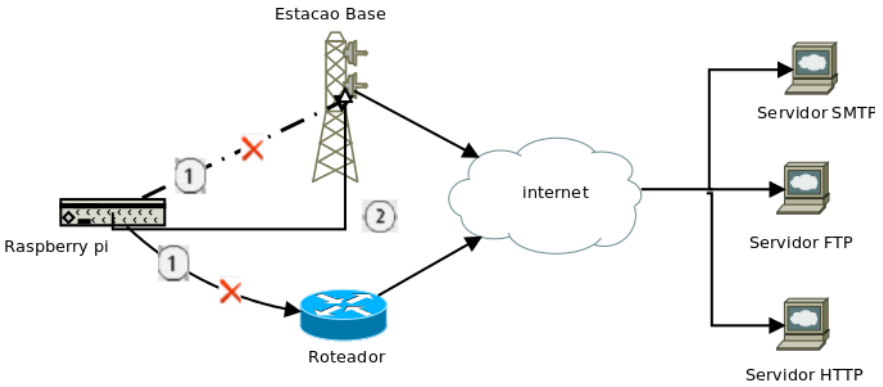

Figura 15 – Cenário com interrupção drástica durante o envio

5.4.2 Objetivo

O objetivo deste cenário é verificar que mesmo acontecendo uma interrupção durante o envio dos dados, a interface irá trocar o enlace de modo que a transmissão continue e que os dados não sejam perdidos.

5.4.3 Resultados

Este cenário consiste em deixar as duas interfaces ativas como mostra a Figura 16 e durante o envio provocar uma interrupção brusca da comunicação. A Figura 17

mostra a interrupção durante o envio e a mudança de interface acontecendo automaticamente. É possível observar que os dados continuam sendo enviados saindo pela segunda interface que estava disponível. Ao término é verificado a integridade dos dados como mostra a Figura 18 comprovando que nenhum dado foi perdido apesar de ter acontecido uma mudança repentina na rota de envio dos dados.

```
root@raspberrypi:~/projeto# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.2.1     0.0.0.0         UG    0      0        0 eth0
0.0.0.0         10.64.64.64     0.0.0.0         UG    0      0        0 ppp0
10.64.64.64     0.0.0.0         255.255.255.255 UH    0      0        0 ppp0
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

Figura 16 – Duas interfaces ativas

```
lo
eth0
ppp0
Interface Ativa
------------------------------------------------------------------------
PING 8.8.8.8 (8.8.8.8) from 192.168.2.107 eth0: 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_req=1 ttl=52 time=55.3 ms
64 bytes from 8.8.8.8: icmp_req=2 ttl=52 time=55.0 ms
64 bytes from 8.8.8.8: icmp_req=3 ttl=52 time=55.1 ms
64 bytes from 8.8.8.8: icmp_req=4 ttl=52 time=55.2 ms
64 bytes from 8.8.8.8: icmp_req=5 ttl=52 time=55.3 ms

--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4005ms
rtt min/avg/max/mdev = 55.028/55.225/55.339/0.280 ms
------------------------------------------------------------------------
PING 8.8.8.8 (8.8.8.8) from 179.173.85.71 ppp0: 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_req=1 ttl=40 time=1205 ms
64 bytes from 8.8.8.8: icmp_req=2 ttl=40 time=446 ms
64 bytes from 8.8.8.8: icmp_req=3 ttl=40 time=605 ms
64 bytes from 8.8.8.8: icmp_req=4 ttl=40 time=935 ms
64 bytes from 8.8.8.8: icmp_req=5 ttl=40 time=595 ms

--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3999ms
rtt min/avg/max/mdev = 446.475/757.793/1205.995/275.306 ms, pipe 2
Conexao Verifica e OK
------------------------------------------------------------------------
E-mail enviado
Saindo SMTP pelo 192.168.2.107
Ocorreu um erro durante o envio
Saindo FTP pelo 179.173.85.71
```

Figura 17 – Interrupção e mudança de interface durante envio

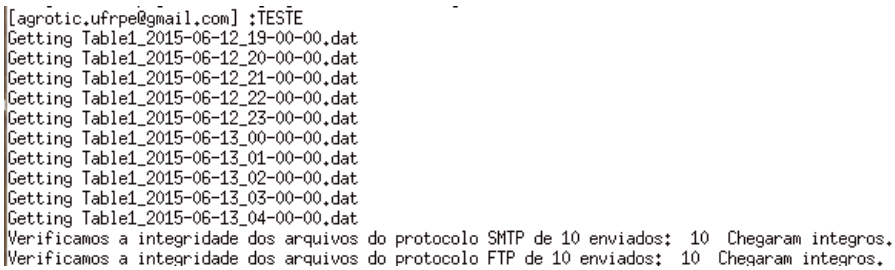

Figura 18 – Integridade dos dados após a mudança de interface

6 Descrição dos testes 2ª estratégia

Nos testes da segunda estratégia foi utilizado 3 interfaces para a verificação, com período entre as verificações de 60 segundos. A partir da soma da média do RTT de cada enlace e cada protocolo e com o fator de erro de cada enlace e cada protocolo irá gerar um índice de qualidade para cada um dos serviços. Com o resultados dos índices a interface de comunicação fará a escolha de qual serviço utilizar para enviar os dados. Nessa estratégia uma das preocupações é a economia da energia pois so irá utilizar apenas um serviço para transferência.

6.1 Verificação 1

Nesta verificação será observado o índice de qualidade de cada um dos serviços do enlace ETH0. Também será comparado com o melhor índice.

6.1.1 Resultados verificação 1

A Figura 19 mostra os índices de qualidade da interface ETH0 por cada serviço de envio. Os altos picos que a figura mostra revela que houver perda de pacotes durante a verificação e com isso elevou o índice de qualidade dos serviços.

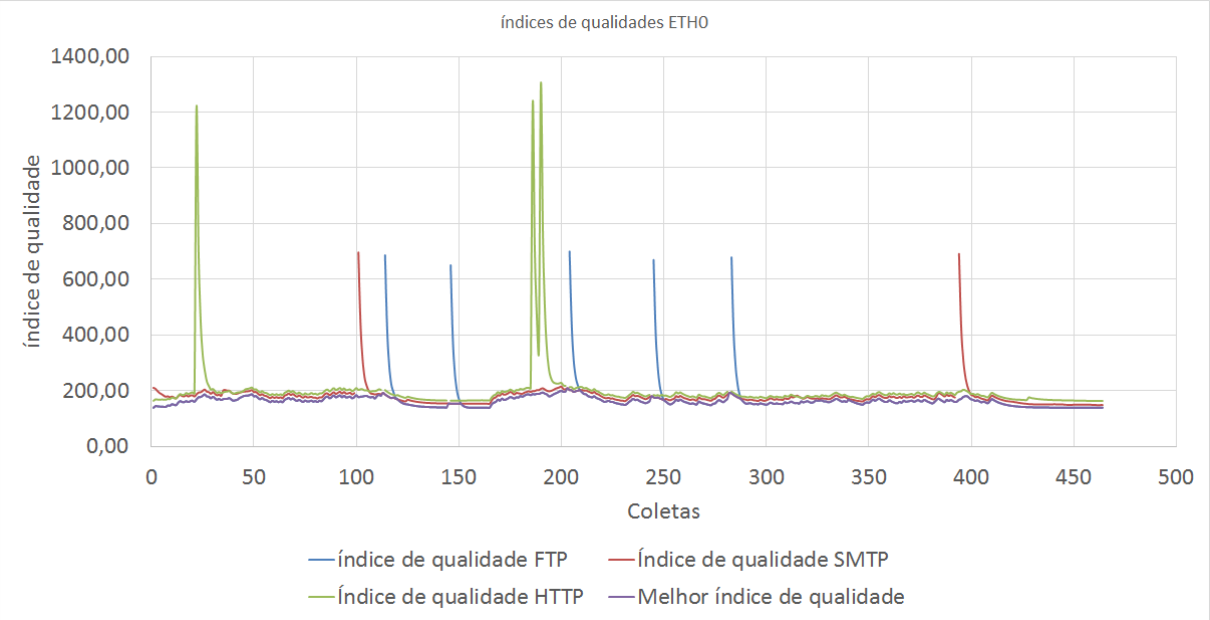

Figura 19 – índices de qualidade ETH0

6.2 Verificação 2

Nesta verificação será observado o índice de qualidade de cada um dos serviços do enlace PPP0. Também será comparado com o melhor índice.

6.2.1 Resultados verificação 2

A Figura 20 mostra os índices de qualidade da interface PPP0 por cada serviço de envio. Nesta verificação é possível diferenciar o melhor índice dos demais, pois por essa interface o índice de qualidade dos serviços não foram bons.

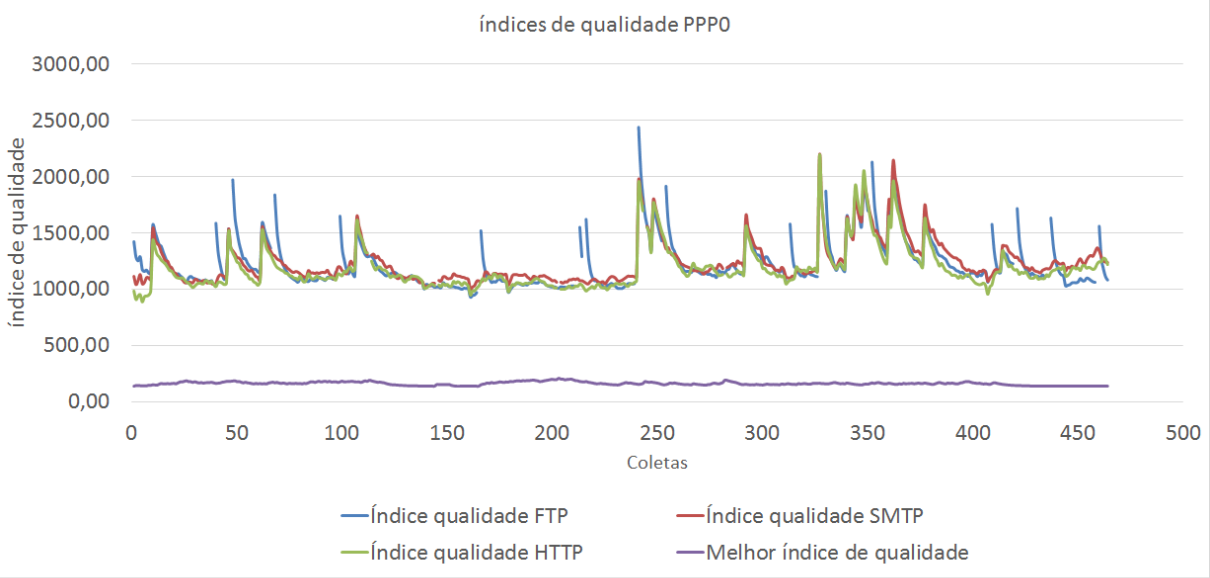

Figura 20 – índices de qualidade PPP0

6.3 Verificação 3

Nesta verificação será observado o índice de qualidade de cada um dos serviços do enlace WLAN0. Também será comparado com o melhor índice.

6.3.1 Resultados verificação 3

A Figura 21 mostra os índices de qualidade da interface WLAN0 por cada serviço de envio. Observa-se que nesta verificação aconteceu mais erros de perda de pacotes consequentemente gerou altos picos do índice de qualidade.

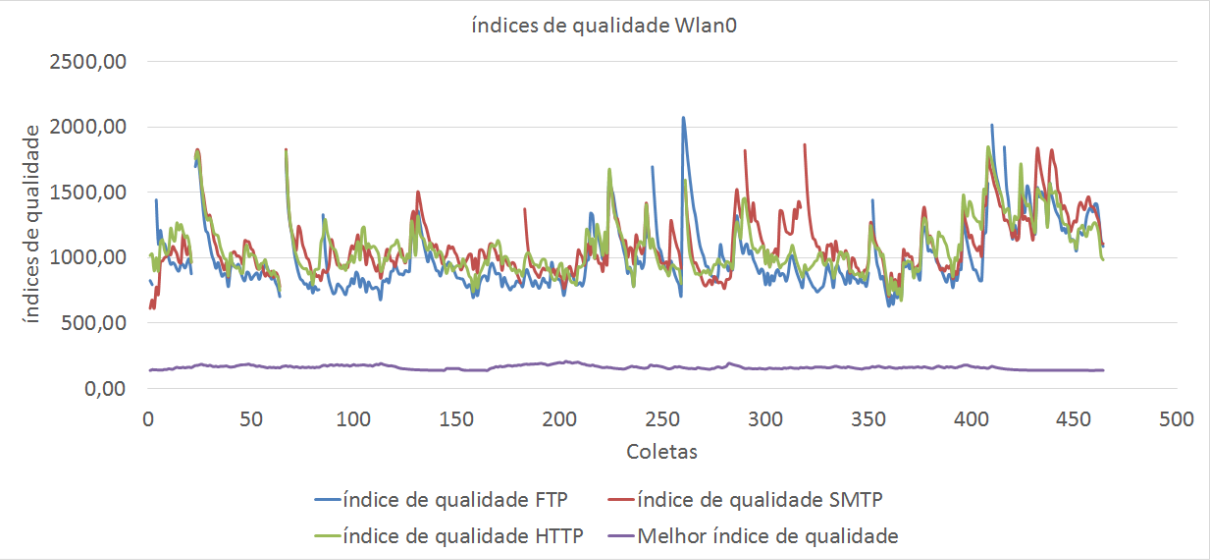

Figura 21 – índices de qualidade WLAN0

6.4 Resultados com todos os índices juntos

A Figura 22 mostra o gráfico com todos os índices de qualidade juntos.

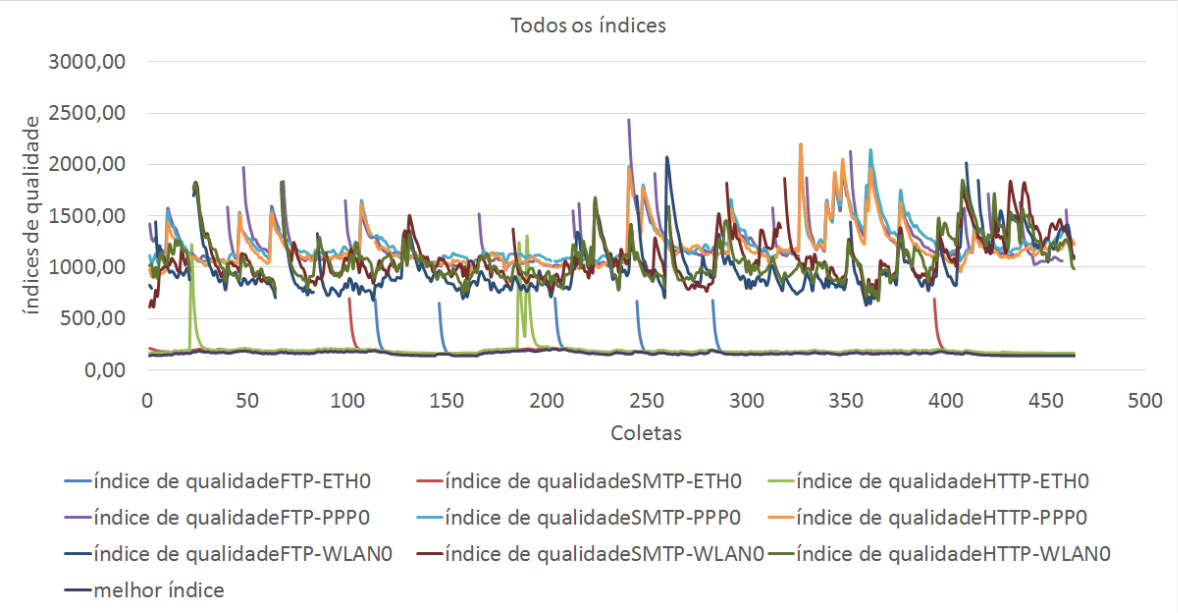

Figura 22 – Todos os índices

7 Conclusão

O presente trabalho teve como principal objetivo fazer com que uma estação de coleta de dados ambientais possa melhorar sua disponibilidade e minimizar a perda de dados através de uma interface de comunicação *multihomed*.

Esse trabalho foi baseado em um problema real, específico na transmissão de dados a partir de áreas remotas, onde observamos que é imprescindível a integridade dos dados coletados para que os pesquisadores sejam capazes de prever com precisão fenômenos naturais.

7.1 Dificuldades encontradas

Durante a execução do trabalho foram encontradas algumas dificuldades. A primeira está relacionada com a preparação do ambiente de trabalho que envolve o *hardware* utilizado e a plataforma de programação. No decorrer do trabalho não tivemos como fazer testes em campo, visto que envolveria vários fatores. Porém foram realizados testes em laboratório com cenários mais próximos da realidade.

Uma segunda dificuldade envolve os custos para realização dos testes visto que não dispúnhamos de verba própria para uso da rede 3G da operadora.

7.2 Lições aprendidas

Com o desenvolvimento do trabalho de conclusão de curso pude absorver diversos aprendizados. Dentre eles destaca-se o de compreender o funcionamento de um sistema operacional em uma plataforma embarcada, a configurar e usar um microcomputador embarcado, compreender com detalhes o funcionamento dos protocolos de comunicação em redes e por fim a programação de sistemas no contexto de redes de computadores.

A última lição aprendida e de muita valia foi participar de um projeto importante que tem potencial de contribuir para a sociedade, pois a previsão através de dados ambientais pode melhorar a eficiência na agricultura e podemos fazer com que vidas sejam salvas antecipando informações sobre catástrofes nas grandes cidades ou em regiões remotas.

7.3 Trabalhos futuros

Como trabalhos futuros podemos aperfeiçoar o módulo de comunicação implementado uma escolha mais inteligente das interfaces de rede através de critérios como: qualidade da banda, perda de pacotes, disponibilidade, energia e etc.

Um segundo trabalho poderá ser a avaliação de mais tecnologias para envio de dados dentro no módulo de comunicação, como por exemplo, a comunicação via satélite que é essencial em alguns locais bastante isolados.

Um terceiro trabalho possível é a análise de desempenho de diferentes bibliotecas que trabalham com envio de dados através dos protocolos de aplicação e constatar quais são as melhores a serem utilizadas. Neste trabalho foram utilizadas as bibliotecas *smtplib, ftplib* e *httplib*.

Referências

Alberto Cortes Martin et al. Selection and publication of network interface cards in multihomed pervasive computing devices. In: . IEEE, 2011. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5766876>. Citado na página 12.

Alberto Cortés Martín. *Mejoras al aprovechamiento de dispositivos multiconectados*. Tese (Doutorado) — Universidad Carlos III de Madrid. Departamento de Ingeniería Telemática, Madrid, 2012. Citado na página 13.

Carlos EM Tucci. *Gestão de águas pluviais urbanas*. Programa de Modernização do Setor Saneamento, Secretaria Nacional de Saneamento Ambiental, Ministério das Cidades, 2005. Disponível em: <http://eventos.caf.com/media/21762/gestiao-aguas-pluviais-carlos-tucci2.pdf>. Citado na página 12.

Douglas E. Comer. *Redes de Computadores e Internet*. 4. ed. [S.l.]: Bookman, 2007. ISBN 978-85-60031-36-8. Citado 2 vezes nas páginas 16 e 17.

Edson B. Teracine. O Brasil e as atividades espaciais. *Parcerias Estratégicas*, v. 4, n. 7, p. 05–06, 2009. Disponível em: <http://seer.cgee.org.br/index.php/parcerias_estrategicas/article/viewArticle/76>. Citado na página 12.

HOLMA, H.; TOSKALA, A. *LTE for UMTS-OFDMA and SC-FDMA based radio access*. [S.l.]: John Wiley & Sons, 2009. Citado na página 20.

James F. Kurose; Keith W. Ross. *Redes de computadores e a Internet*. 3. ed. [S.l.]: Pearson, 2005. ISBN 85-88639-18-1. Citado 2 vezes nas páginas 21 e 22.

KRANSMO, J. L. *Broadcasting of two generation cellular system control channel information over a three generation control channel to support roaming and handover to two generation cellular networks*. [S.l.]: Google Patents, 2003. US Patent 6,594,242. Citado na página 20.

LEE, J. C. J. Energy-efficient rate allocation for multi-homed streaming service over heterogeneous access networks. In: . Houston, TX, USA: Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE, 2011. Citado na página 13.

LEFFLER, S. J.; KARELS, M. J.; MCKUSICK, M. K. *The design and implementation of the 4.3BSD Unix operating system*. Reading, MA: Addison-Wesley, 1989. Disponível em: <https://cds.cern.ch/record/113321>. Citado na página 17.

Marco A. Filippetti. *CCNA 5.0 Guia Completo de Estudo*. [S.l.]: Visual Books, 2014. ISBN 978-85-7502-284-9. Citado na página 19.

Marcos Flávio Araújo Assunção. *WIRELESS HACKING Ataques e segurança de redes sem fio Wi-Fi*. 1. ed. Florianópolis: Visual Books, 2013. ISBN 987-85-7502-282-5. Citado na página 20.

RIVEST, R. L. et al. Rfc 1321: The md5 message-digest algorithm. *Internet activities board*, v. 143, 1992. Citado na página 32.

RUSLING, D. A. *The linux kernel*. 1999. Citado na página 18.

Shahriar Nirjon et al. MultiNets: Policy Oriented Real-Time Switching of Wireless Interfaces on Mobile Devices. In: . IEEE, 2012. p. 251–260. ISBN 978-1-4673-0883-0. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6200056>. Citado na página 13.

TANENBAUM, A. S. *Redes de computadores*. [S.l.]: Pearson Educación, 2003. Citado na página 16.