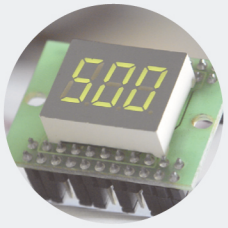
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES






SERVIÇO DE CORREIO ELETRÓNICO EM AMBIENTE MARÍTIMO UTILIZANDO REDES TOLERANTES AO ATRASO

JOÃO MIGUEL DIAS RODRIGUES
Novembro de 2015

POLITÉCNICO
DO PORTO

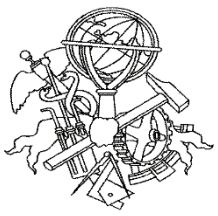
SERVIÇO DE CORREIO ELETRÓNICO EM AMBIENTE MARÍTIMO UTILIZANDO REDES TOLERANTES AO ATRASO

João Miguel Dias Rodrigues

Mestrado em Engenharia Eletrotécnica e de Computadores
Área de Especialização de Telecomunicações
Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
2015

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Disciplina de Tese/Dissertação, do 2º ano, do Mestrado em Engenharia Eletrotécnica e de Computadores

Candidato: João Miguel Dias Rodrigues, Nº 1130201 1130201@isep.ipp.pt
Orientação científica: Jorge Botelho Costa Mamede, jbm@isep.ipp.pt
Empresa: INESC TEC
Supervisão: Jaime Sousa Dias, Jaime.dias@inesctec.pt



Mestrado em Engenharia Eletrotécnica e de Computadores
Área de Especialização de Telecomunicações
Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
2 de novembro de 2015

Agradecimentos

A elaboração desta dissertação não seria possível sem ajuda de algumas pessoas que, direta ou indiretamente, contribuíram para o sucesso da mesma. Assim sendo, venho por este meio expressar a minha gratidão para com essas pessoas.

Em primeiro lugar, gostaria de agradecer ao meu orientador, Professor Doutor Jorge Mamede, por me ter dado a oportunidade de desenvolver esta dissertação inserido na equipa do INESC TEC, e pelo incondicional apoio dado durante a realização da mesma.

Em segundo lugar, quero agradecer ao meu coorientador, Mestre Jaime Dias, pela total disponibilidade, pelos conselhos dados e por ter ajudado a resolver os problemas encontrados durante a fase de implementação. Gostaria também de dar uma palavra de agradecimento ao Mestre Mário Lopes, pois sempre que lhe foi solicitada ajuda, pôs ao dispor o seu conhecimento.

Em terceiro lugar quero agradecer aos meus colegas e amigos, pelo apoio dado durante toda a dissertação e durante o decorrer do curso. Sem eles não era possível ultrapassar os dois anos de mestrado da forma que o fiz.

Por último, mas não menos importante, quero agradecer à minha família, principalmente aos pais e irmão, pelo apoio incondicional que sempre me deram durante esta e outras etapas da minha vida.

Resumo

Delay Tolerant Network (DTN) é uma arquitetura de redes que procura resolver os problemas associados à conetividade intermitente de sistemas e possibilita a existência de comunicações em ambientes onde o conjunto de protocolos tradicionais TCP/IP não funciona. A arquitetura DTN é adequada a cenários com uma topologia de rede dinâmica, densidade de nós reduzida, conetividade intermitente e de curta duração entre os nós, e em que as aplicações são tolerantes ao atraso.

Nesta dissertação é apresentada uma solução de baixo custo recorrendo ao conceito DTN que permite a utilizadores de embarcações utilizarem o serviço de correio eletrónico no mar. A solução estende o sistema de correio eletrónico ao cenário marítimo recorrendo a estações na costa, comunicação sem fios entre embarcações e entre estas e a estações na costa, e à capacidade das embarcações funcionarem como meios de transporte de dados.

Para proceder à validação da proposta apresentada, foi implementado um protótipo com o sistema de correio eletrónico adaptado ao cenário marítimo. O protótipo é constituído por vários nós, configurados de forma a assumir o papel de embarcações, estação da costa e um servidor de *e-mail* presente na Internet.

Os resultados dos testes experimentais realizados em ambiente controlado mostram que os objetivos do trabalho foram alcançados. O serviço de *e-mail* assente sobre a arquitetura DTN e adaptado ao cenário de comunicações marítimo foi testado em diferentes contextos, e em todos eles, as experiências tiveram resultados positivos.

Palavras-Chave

Redes Tolerantes ao Atraso, *E-mail*, Comunicações Marítimas

Abstract

Delay tolerant network (DTN) is a network architecture that aims at solving the problems associated with intermittent connectivity, and enables communications in environments where traditional TCP/IP protocol do not work. The DTN architecture is suitable for scenarios with a dynamic network topology, lower node density, intermittent short duration connectivity and, where applications are delay tolerant.

This dissertation presents a low-cost solution using the DTN concept that allows vessel users to use the e-mail service at sea. The solution extends the e-mail system to the maritime scenario using stations on the shore, wireless communication between vessels and between vessels and coast stations, and the ability of the vessels to transport data.

To proceed with the validation of the proposal, a prototype with the e-mail system adapted to the maritime scenario was implemented. The prototype consists in multiple nodes, configured to assume the role of vessels, shore station and a mail server present on the Internet.

The results of experimental tests performed in a controlled environment show that the objectives of this work were achieved. The e-mail service based on DTN architecture and adapted to maritime communications scenario was tested in different contexts, and in all of them, the experiments had good result.

Keywords

Delay Tolerant Network, E-mail, Maritime Communications

Índice

AGRADECIMENTOS ..... **I**

RESUMO ..... **III**

ABSTRACT ..... **V**

ÍNDICE ..... **VII**

ÍNDICE DE FIGURAS ..... **IX**

ÍNDICE DE TABELAS ..... **XI**

ÍNDICE DE EXCERTOS DE CÓDIGO ..... **XIII**

ACRÓNIMOS ..... **XV**

1. INTRODUÇÃO **1**

1.1. CONTEXTUALIZAÇÃO 1
1.2. OBJETIVOS 3
1.3. ORGANIZAÇÃO DO RELATÓRIO 3

2. ESTADO DA ARTE **5**

2.1. SISTEMA DE *E-MAIL* 5
2.2. DELAY TOLERANT NETWORK 6
2.3. DESEMPENHO DOS PROTOCOLOS DE ENCAMINHAMENTO EM DTN 9
2.4. SOLUÇÕES DE CORREIO ELETRÓNICO EM DTN 12
2.5. RESUMO E DISCUSSÃO 15

3. ARQUITETURA PROPOSTA E IMPLEMENTAÇÃO **19**

3.1. ESTRUTURA DO SISTEMA 20
3.2. *HARDWARE* 21
3.3. SOFTWARE 22
3.3.1 PLATAFORMA DE VIRTUALIZAÇÃO 22
3.3.2 SISTEMA OPERATIVO 22
3.3.3 AGENTE DE TRANSPORTE DE *E-MAIL* (MTA) 23
3.3.4 CLIENTE DE *E-MAIL* 23
3.3.5 SERVIDOR DNS 23
3.3.6 PROTOCOLO BUNDLE 23
3.3.7 INTEGRAÇÃO DO SISTEMA DE *E-MAIL* COM A ARQUITETURA DTN 24
3.4. INSTALAÇÃO E CONFIGURAÇÃO 25
3.4.1 PROTOCOLO BUNDLE 26
3.4.2 INTEGRAÇÃO DO POSTFIX COM PROTOCOLO BUNDLE 29
3.5. TESTES PRELIMINARES 36

3.5.1 DESEMPENHO DO PROTOCOLO BUNDLE 37
3.5.2 INTEGRAÇÃO DO POSTFIX COM A ARQUITETURA DTN 39

4. AVALIAÇÃO EXPERIMENTAL 43

4.1. PARÂMETROS EM AVALIAÇÃO 43
4.2. TESTES DE DESEMPENHO ENTRE DOIS NÓS COM CONETIVIDADE PERMANENTE (CENÁRIO 1) 44
4.2.1 CONDIÇÕES DE TESTE 45
4.2.2 ANÁLISE DE DESEMPENHO NO CENÁRIO 1 46
4.3. TESTES DE DESEMPENHO ENTRE DOIS NÓS COM CONETIVIDADE INTERMITENTE (CENÁRIO 2) 47
4.3.1 CONDIÇÕES DE TESTE 49
4.3.2 ANÁLISE DE DESEMPENHO NO CENÁRIO 2 49
4.4. TESTES DE DESEMPENHO ENTRE SEIS NÓS COM CONETIVIDADE PERMANENTE (CENÁRIO 3) 50
4.4.1 CONDIÇÕES DE TESTE 51
4.4.2 ANÁLISE DE DESEMPENHO NO CENÁRIO 3 51
4.5. TESTES FUNCIONAIS ENTRE TRÊS NÓS COM ARQUITETURAS DISTINTAS 52
4.5.1 CONDIÇÕES DE TESTE 53
4.5.2 ANÁLISE AO COMPORTAMENTO DA SOLUÇÃO PROPOSTA 54
4.6. CONCLUSÕES E DISCUSSÃO 57

5. CONCLUSÕES E TRABALHO FUTURO 59

5.1. TRABALHO FUTURO 60

REFERÊNCIAS DOCUMENTAIS 61

ANEXO A. FICHEIROS DE CONFIGURAÇÃO DA ARQUITETURA DTN 71

Índice de Figuras

Figura 1 Serviço de correio eletrónico no mar. 2
Figura 2 Mecanismo *Store-carry-forward*. 7
Figura 3 Camadas protocolares arquitetura DTN. 8
Figura 4 Comparação da taxa de entrega. 9
Figura 5 Probabilidade de entrega de *bundles* em função do tamanho 11
Figura 6 Probabilidade de entrega de *bundles* em função do tempo de vida. 11
Figura 7 Cenário de testes. 12
Figura 8 Tempo de transmissão. 13
Figura 9 Sumário dos testes. 14
Figura 10 Esquema N4C. 15
Figura 11 Esquema do cenário marítimo. 19
Figura 12 Topologia lógica da *testbed*. 20
Figura 13 Consola *daemon* DTN. 29
Figura 14 Configuração DNS do domínio seamail.com. 36
Figura 15 Configuração DNS do domínio internete.com. 37
Figura 16 Testes na consola do *daemon* DTN. 38
Figura 17 Exemplo de receber e ler uma *bundle*. 39
Figura 18 Excerto do ficheiro dtn_pymail.log da origem da *bundle*. 40
Figura 19 Listagem de bundles na consola do *daemon* DTN. 40
Figura 20 Diagrama de sequência do nó origem de uma *bundle*. 40
Figura 21 Excerto do ficheiro dtn_pymail.log do destino da *bundle*. 41
Figura 22 Diagrama de sequência do nó destino de uma *bundle*. 41
Figura 23 Excerto do ficheiro dtn_pymail.log da origem da *bundle*. 41
Figura 24 Informação recebida no nó origem da *bundle*. 42
Figura 25 Cenário 1 – Dois nós DTN com conexão estabelecida. 44
Figura 26 Topologia de rede do cenário 1. 45
Figura 27 Tempo de entrega de *e-mails* com 4 kbytes. 46
Figura 28 Tempo de entrega de *e-mails* com tamanho variável. 47
Figura 29 Cenário 2 - Dois nós DTN sem conexão previamente estabelecida. 48
Figura 30 Topologia de rede do cenário 2. 48
Figura 31 Tempo total de sincronização e entrega de *e-mails*. 49
Figura 32 Cenário 3 – Seis nós DTN com conexão estabelecida. 50
Figura 33 Topologia de rede do cenário 3. 51
Figura 34 Tempo de entrega de *e-mails* com número de saltos variável. 52

Figura 35 Cenário 4 – Três nós com arquiteturas distintas. 53
Figura 36 Topologia de rede do cenário 4. 53
Figura 37 Criação de *e-mail*. 54
Figura 38 Excerto do ficheiro mail.log. 54
Figura 39 Leitura do *e-mail*. 54
Figura 40 Diagrama de sequência da primeira experiência do cenário 4. 55
Figura 41 Criação de *e-mail*. 55
Figura 42 Excerto do ficheiro mail.log. 56
Figura 43 Leitura de *e-mail*. 56
Figura 44 Diagrama de sequência da primeira experiência do cenário 4. 57

Índice de Tabelas

Tabela 1 Especificações Toshiba L755-104. 21
Tabela 2 Protocolo Bundle e dependências. 27
Tabela 3 Ficheiros do Pymail reutilizados. 31

Índice de Excertos de Código

Excerto de código 1 Configuração do ficheiro `Dtn.conf`. 28
Excerto de código 2 Configuração ficheiro `Main.cf` do Postfix. 30
Excerto de código 3 Configuração ficheiro `Master.cf` do Postfix. 30
Excerto de código 4 Configuração de Transporte do Postfix de embarcações 30
Excerto de código 5 Configuração de Transporte do Postfix da costa. 31
Excerto de código 6 Algoritmo do ficheiro `Dp_dtn.py` presente em embarcações. .. 33
Excerto de código 7 Algoritmo do ficheiro `Dp_dtn.py` presente na costa. 33
Excerto de código 8 Alterações feitas ao ficheiro `Dp_pfmailout.py`. 35
Excerto de código 9 Alterações feitas ao ficheiro `Dp_pfmailin.py`. 35

Acrónimos

ADU – Application Data Unit

API – Application Programming Interface

AODV – Ad Hoc On-Demand Distance Vector

BIND – Berkeley Internet Name Domain

DNS – Domain Name System

DTN – Delay Tolerant Network

DTNRG – Delay Tolerant Networking Research Group

EID – Endpoint Identifier

IMAP – Internet Message Access Protocol

MDA – Mail Delivery Agent

MTA – Mail Transfer Agent

MUA – Mail User Agent

MX – Mail Exchanger

N4C – Networking for Communications Challenged Communities

OLSR – Optimized Link State Routing

POP3 – Post Office Protocol 3

PRoPHET – Probabilistic Routing Protocol using History of Encounters and Transitivity

SMTP – Simple Mail Transfer Protocol

TTL – Time To Live

UDP – User Datagram Protocol

URI – Uniform Resource Identifier

1. INTRODUÇÃO

1.1. CONTEXTUALIZAÇÃO

O *e-mail* é uma das aplicações da Internet mais utilizadas em todo mundo. Contudo, nem todas as pessoas podem usufruir desse serviço, especialmente quando se encontram em zonas remotas. Enquanto que em solo terrestre o acesso dos utilizadores ao serviço de correio eletrónico é relativamente fácil de assegurar, no cenário marítimo devido sobretudo às caraterísticas geográficas, isso não acontece. Tipicamente, no ambiente marítimo, o acesso à banda larga e a utilização de serviços da Internet, estão limitados às zonas próximas da costa através de redes móveis, ou via satélite disponível em qualquer lugar, sendo que este último apresenta custos muito elevados para a maioria dos utilizadores. Existe portanto a necessidade de aproximar as comunicações marítimas ao cenário de comunicações em terra, proporcionando aos utilizadores de pequenas embarcações o acesso a serviços da Internet a um custo acessível.

Nesse sentido, surgiu com esta dissertação, a intenção de estender o sistema de correio eletrónico ao cenário marítimo, tirando partido das tecnologias sem fios existentes, da presença de uma estação junto à costa, bem como da mobilidade e capacidade de armazenamento de dados nas embarcações. Esta proposta visa fornecer o serviço de correio eletrónico aos utilizadores das embarcações aproveitando a conetividade temporária e intermitente estabelecida entre os nós (Figura 1).

Figura 1 Serviço de correio eletrónico no mar.

Partindo do princípio que cada embarcação apenas consegue comunicar com outro nó, seja ele outra embarcação ou o nó localizado na costa, quando está a uma distância reduzida, é evidente que num ambiente amplo como é o marítimo, as oportunidades de comunicação são escassas [1] [2]. Assim sendo, caso os *e-mails* tivessem que ser entregues diretamente ao nó destino, cada nó teria de ter a capacidade de os guardar por um largo período de tempo, até que fosse possível passá-los ao nó destino, isto, se porventura surgisse essa oportunidade. De forma a aumentar as probabilidades de entrega, pretende-se que cada embarcação tenha a capacidade de transportar e passar os *e-mails* a outras embarcações, repetindo esse processo até que, eventualmente, estes cheguem ao destino. Mesmo assim, visto que a rede marítima é caraterizada por contatos de curta duração entre os nós, uma topologia dinâmica e uma baixa densidade de nós, o cenário de comunicações marítimo é propenso a interrupções e a significativos atrasos nas ligações. Desta forma, e apesar do serviço de correio eletrónico ser tolerante a atrasos, a conetividade intermitente, a inexistência de ligações extremo-a-extremo e a ausência de uma infraestrutura DNS no cenário marítimo, impossibilitam a utilização de protocolos utilizados em terra no transporte de *e-mails*, como o Simple Mail Transfer Protocol (SMTP) [3]. Por esse motivo, recorreu-se à utilização de uma arquitetura de redes tolerante a atrasos e conetividade intermitente, denominada *Delay Tolerant Network* (DTN) [4], que possibilita a existência de comunicações em ambientes extremos, utilizando na entrega de dados o princípio de funcionamento salto-a-salto (*hop-by-hop*). A utilização da arquitetura de redes DTN foi a opção colocada logo de início neste trabalho, visto que a mesma já foi aplicada com sucesso em diversos projetos realizados pela

comunidade científica, que procuraram resolver os problemas associados à conetividade intermitente em ambientes remotos [5] [6] [7].

De forma a avaliar o desempenho e viabilidade da solução proposta, foi realizado em ambiente controlado um conjunto de testes ao protótipo implementado na dissertação. Os testes experimentais consistiram no envio e receção de *e-mails* em diferentes cenários, planeados de forma a aproximar as condições do ambiente controlado às condições das comunicações marítimas.

1.2. OBJETIVOS

O principal objetivo deste trabalho foi desenvolver uma solução de correio eletrónico baseada em DTN, que visa permitir a utilização desse serviço em zonas marítimas remotas. Para alcançar esta meta, foram considerados os seguintes objetivos específicos:

- Fazer um estudo sobre o que já foi proposto até agora, sobretudo envolvendo serviço de *e-mail* em DTN;
- Estudar a arquitetura DTN, nomeadamente o seu conceito, a sua implementação e o seu modo de funcionamento;
- Fazer o levantamento do *software* necessário para implementar a proposta de correio eletrónico;
- Integrar o serviço de correio eletrónico com a arquitetura DTN em dois tipos de nós (embarcações e estação da costa);
- Implementar uma *testbed* e definir os cenários e condições dos testes, tendo em atenção o panorama marítimo;
- Efetuar testes funcionais e de desempenho nos diferentes cenários;
- Analisar resultados e verificar a viabilidade da proposta desenvolvida em ambiente real.

1.3. ORGANIZAÇÃO DO RELATÓRIO

Este documento está estruturado em 5 capítulos. No Capítulo 2, é apresentado o estado da arte onde são expostos conceitos, estudos e trabalhos realizados no âmbito do tema abordado. O capítulo 3 apresenta a montagem do protótipo, abrangendo o *hardware* e *software* utilizado, juntamente com uma avaliação preliminar ao funcionamento do sistema. No capítulo 4 é feita a avaliação experimental ao protótipo nos diferentes cenários elaborados. No capítulo 5 são apresentadas as conclusões finais e trabalho futuro.

2. ESTADO DA ARTE

A arquitetura DTN foi originalmente desenvolvida para comunicações interplanetárias. Contudo, com o passar dos anos, a comunidade científica percebeu a importância que a arquitetura DTN poderia ter nas comunicações terrestres e na expansão dos serviços da Internet a zonas remotas [8]. Nesta dissertação é feita uma investigação para conhecer o sistema de *e-mail*, bem como a arquitetura DTN e o seu modo de funcionamento.

Neste capítulo são também apresentados estudos realizados sobre a arquitetura DTN, nomeadamente sobre desempenho dos seus protocolos de encaminhamento. Somado a isto, são retratadas soluções propostas até hoje que envolvem o serviço de correio eletrónico com DTN.

2.1. SISTEMA DE *E-MAIL*

Nesta secção é caraterizado o sistema de correio eletrónico, dando a conhecer os componentes do mesmo, os protocolos utilizados e o seu funcionamento básico. O sistema básico de *e-mail* é constituído por três componentes principais: cliente de *e-mail*, servidor de e-mail e o sistema de nomes de domínios (DNS).

O cliente de *e-mail*, é um software que permite ao utilizador o acesso ao serviço de correio eletrónico. Os clientes de *e-mail* [9], também conhecidos por *mail user agent* (MUA), podem

ser aplicações locais, como Microsoft Outlook, Mozilla Thunderbird e Mutt, ou podem ser aplicações remotas com interface *web,* como o Gmail ou Yahoo. Seja qual for o tipo de cliente, geralmente cada um deles tem a capacidade de:

- Mostrar lista de todas as mensagens que estão na caixa de correio do utilizador;
- Selecionar mensagens e ler o seu conteúdo;
- Criar e enviar novas mensagens.

Sempre que é criada uma mensagem a partir de um cliente de *e-mail*, essa mensagem é entregue a um servidor de *e-mail*. Um servidor de *e-mail* ou *mail transfer agent* (MTA) [10] é uma aplicação que está instalada num computador e é responsável pela transferência de *e-mail* entre utilizadores. A principal funcionalidade de um MTA é receber *e-mails* de utilizadores locais ou remotos e encaminhá-los localmente ou remotamente. Os MTAs comunicam entre si através do protocolo *Simple Mail Transfer Protocol* (SMTP), sendo que o objetivo dessas comunicações é propagar as mensagens até ao MTA do destinatário. Exemplo de MTA é o *software* Exim, Postfix, Sendmail, entre outros.

O sistema de nomes de domínios (DNS) tem um papel essencial no sistema de *e-mail*. Os servidores DNS guardam registos MX (*mail exchanger*), que especificam para onde é que devem ser encaminhados os *e-mails* destinados a um determinado domínio. Portanto, sempre que um MTA recebe um *e-mail*, e o destinatário não é um utilizador local, é feita uma consulta a um servidor DNS para saber o endereço IP do MTA para o qual o *e-mail* deve de ser enviado [11].

Assim que uma mensagem chega ao MTA final, isto é, ao servidor de *e-mail* responsável pelo domínio do destinatário, esta é passada a um agente de entrega de *e-mail* ou *mail delivery agent* (MDA), que posteriormente a armazena numa caixa de correio (mailbox) presente no servidor e reservada ao utilizador que recebeu o *e-mail*. Uma vez presente na caixa de correio, as mensagens podem ser lidas localmente com um cliente de *e-mail* ou acedidas remotamente e descarregadas com os protocolos *Post Office Protocol* 3 (POP3) [12] ou Internet *Message Access Protocol* (IMAP) [13].

2.2. DELAY TOLERANT NETWORK

Delay tolerant network [4] é uma arquitetura de rede que possibilita a existência de comunicações em ambientes com conectividade intermitente, atrasos significativos e alta taxa de erros de transmissão, utilizando o princípio de funcionamento salto-a-salto na entrega

de dados. As comunicações são assentes em contatos oportunísticos que acontecem entre os nós da rede, que proporcionam a transferência de dados através do mecanismo *store-carry-forward*.

O mecanismo *store-carry-forward* [14], muitas vezes denominado *store-and-forward*, permite que as mensagens sejam transferidas pela rede saltando de um nó para o outro até ao destino, tendo como suporte o armazenamento persistente de cada nó (Figura 2).

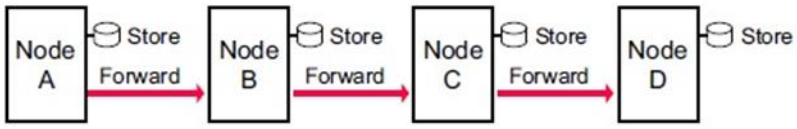


Figura 2 Mecanismo *Store-carry-forward*.

A arquitetura DTN é tolerante a atrasos na entrega dos dados, de horas ou até dias, devido ao armazenamento persistente (disco rígido, memória *flash*) existente em cada nó. Este tipo de armazenamento é necessário em cada nó pelas seguintes razões:

- A ligação com outro nó e o envio de pacotes pode demorar um longo período de tempo;
- As velocidades de transmissão e a eficiência de atuação dos dois nós podem não ser iguais;
- Após a transmissão de uma mensagem, devido aos erros que podem ocorrer, ou caso o nó esteja configurado para inundar a rede (*flooding*), pode ser necessário retransmitir a mesma mensagem.

A arquitetura DTN implementa em cada nó o mecanismo *store-carry-forward* através da adição da camada protocolar Bundle [15]. Essa camada Bundle, que atentando aos modelos de referência, está entre a camada de transporte e a camada de aplicação, serve para abstrair as camadas protocolares inferiores, de modo a permitir que as aplicações consigam comunicar ao longo dos nós. Os nós podem ser constituídos por arquiteturas protocolares semelhantes ou por arquiteturas diferentes, sendo que o protocolo Bundle atua em cada nó como se fosse uma extremidade de uma conexão extremo-a-extremo. Ou seja, em DTN, cada ligação entre dois nós é gerida como uma conexão extremo-a-extremo, permitindo dessa forma que os dados se propaguem na rede, mesmo quando não existe conetividade constante entre a origem e o destinatário. A camada protocolar Bundle é também responsável por gerir as mensagens trocadas entre os nós da rede DTN, sendo que essas mensagens são

denominadas de *bundles*. A partir da Figura 3 é possível perceber o funcionamento básico do protocolo Bundle, e como é feita a interação do mesmo com as camadas inferiores.

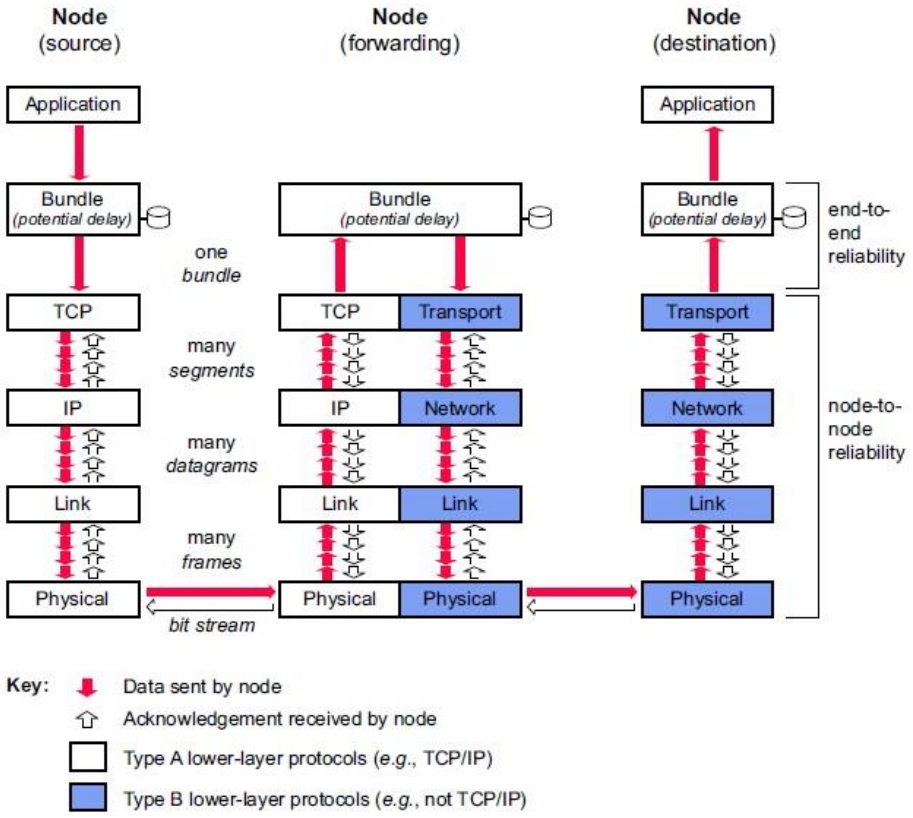


Figura 3 Camadas protocolares arquitetura DTN.

As *bundles* são criadas pela camada protocolar Bundle e consistem num conjunto de dados aplicacionais (ADU - Application Data Unit), agrupados juntamente com um cabeçalho, que contém informação de controlo requerida pela arquitetura DTN. Numa rede DTN, cada nó é um elemento independente que contém um agente do protocolo Bundle e está preparado para a qualquer momento gerar, receber e retransmitir *bundles*. Os nós DTN são identificados através de um *Endpoint Identifier* (EID), que é o equivalente a um endereço de uma máquina. Estes endereços têm uma sintaxe própria da arquitetura DTN e estão associados a um determinado identificador uniforme de recursos (URI - Uniform Resource Identifier) [8] [16].

O protocolo Bundle é instalado através do *software* DTN2 [17], que foi desenvolvido pelo grupo *Delay Tolerant Networking Research Group* (DTNRG) [18], que por sua vez foi criado pela *Internet Research Task Force* (IRTF), que é uma organização que promove a investigação para o desenvolvimento da Internet. O DTN2 é a implementação de referência

do protocolo Bundle, e até à data é o único *software* que implementa a arquitetura DTN no cenário terrestre.

2.3. DESEMPENHO DOS PROTOCOLOS DE ENCAMINHAMENTO EM DTN

Em [19], os autores estudaram o desempenho das comunicações em ambiente marítimo, comparando a eficiência de diferentes protocolos de encaminhamento tais como os tradicionais extremo-a-extremo Ad Hoc On Demand Distance Vector (AODV) [20] e Optimized Link State Routing (OLSR) [21], e os protocolos de redes tolerantes ao atraso, *Epidemic* [22] e *Spray and wait* [23]. O estudo realizado tem por base dados fornecidos pela autoridade portuária de Singapura, que permitiu aos autores desenvolver no simulador de comunicações QualNet, o ambiente de comunicações marítimo do estreito de Singapura de forma precisa. Os protocolos testados utilizam um método de transmissão com vários saltos, exceto o protocolo DTN *Spray and wait* que, nesta simulação, foi configurado para ter apenas um salto durante a transmissão.

No simulador para comparar o desempenho dos vários protocolos de encaminhamento, a topologia da rede marítima foi dividida em quatro sectores, sendo que nos sectores 2 e 3 os barcos estão a aproximar-se da base station (BS) e nos sectores 1 e 4 os barcos estão a afastar-se da base station que está na costa. Em cada simulação, dois barcos de cada sector foram aleatoriamente escolhidos como fontes para gerar tráfego, com destino à base station. Na Figura 4 é possível observar que a percentagem de pacotes entregues varia nos diferentes sectores, sendo que nos sectores 1 e 4, o número de pacotes entregues é reduzido comparativamente aos sectores 2 e 3.

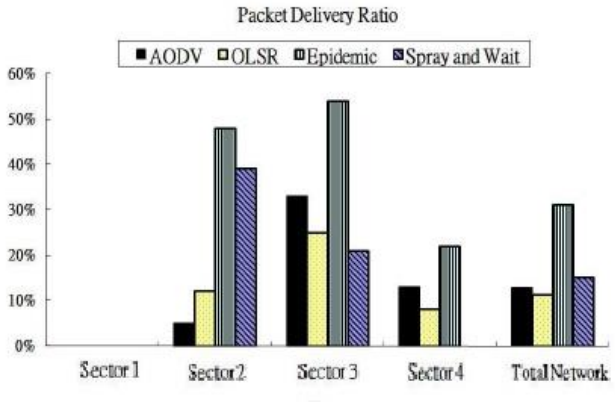


Figura 4 Comparação da taxa de entrega.

Os resultados concluíram que os protocolos DTN conseguem, globalmente, atingir um melhor desempenho do que os protocolos de encaminhamento tradicionais. Deste modo, verifica-se que o protocolo *epidemic* utilizado em redes tolerantes ao atraso, garante uma maior fiabilidade na entrega dos pacotes, e caso não existam limitações de recursos, pode atingir 100% no que se refere à taxa de entrega.

Em [24], foi avaliado o desempenho dos protocolos de encaminhamento existentes em redes DTN, mais especificamente numa rede DTN rodoviária (*Vehicular Delay Tolerant Network*), para perceber a eficiência dos vários protocolos quando existe uma topologia de rede altamente dinâmica, contatos curtos entre os nós e conectividade intermitente. Os protocolos avaliados podem ser agrupados em duas categorias: *Single-copy* ou *Multiple-copy*. Os protocolos *single-copy* fazem circular apenas uma cópia de cada *bundle* na rede, sendo que os métodos *multiple-copy* replicam as *bundles* cada vez que há um contato entre nós. Neste estudo, do tipo *single-copy* foram testados os protocolos *Direct Delivery* [25] e *First Contact* [26]. No *Direct Delivery*, o nó origem transporta a *bundle* até chegar ao nó destino, enquanto que no *First Contact*, a *bundle* é entregue de forma aleatória, ou seja, o nó origem transmite a *bundle* ao primeiro nó que encontra. Do tipo *multiple-copy*, foram testados os protocolos *Epidemic*, *Spray and Wait* e PRoPHET (*Probabilistic Routing Protocol using History of Encounters and Transitivity)* [27]. O método *Epidemic* e o *Spray and Wait* copiam as *bundles* para outros nós cada vez que há um contato, sendo que este último, limita o número de cópias, controlando a inundação da rede. O PRoPHET é um método probabilístico que utiliza informação obtida nos contatos anteriores para saber a quem deve entregar as *bundles*.

A *testbed* definida integrou computadores, portáteis e robôs, sendo que os robôs durante a experiência transportaram os portáteis simulando um veículo. Os computadores foram usados como nós de retransmissão fixos e nós terminais, estes últimos simulando um local isolado. O cenário montado teve 36,5 m² e consistiu em 3 nós terminais, 2 nós de retransmissão e 4 nós móveis. Neste estudo foram considerados dois panoramas. No primeiro foram geradas *bundles* de diferentes tamanhos com um *Time to live* (TTL) fixo, enquanto no segundo foram geradas *bundles* com o mesmo tamanho, mas com diferentes TTL. No primeiro cenário deste estudo foi possível observar que o tamanho das *bundles* teve influencia direta na probabilidade de entrega de todos os protocolos (Figura 5).

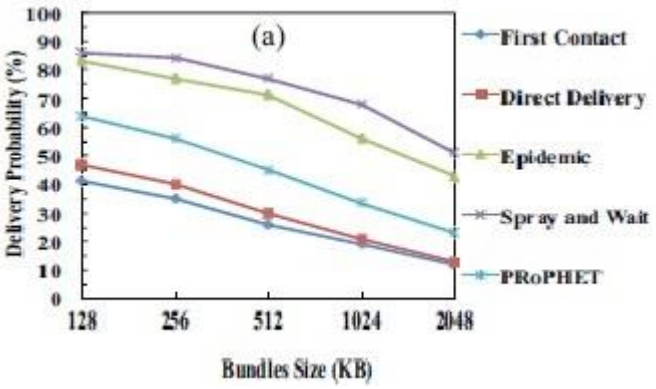


Figura 5 Probabilidade de entrega de *bundles* em função do tamanho.

É possível também notar que os protocolos de *routing multiple-copy* apresentam resultados melhores face aos métodos *single-copy*, especialmente o protocolo *Spray and wait*. Este protocolo é semelhante ao *epidemic*, mas com a particularidade de ter um limite de cópias definido, neste caso são 3, obtendo desta forma um melhor desempenho quando existe menos recursos (armazenamento e largura de banda).

No segundo cenário como é possível ver na Figura 6, aumentando o TTL, isto é, aumentando o tempo de vida de uma *bundle* e permitindo que os nós as armazenem por um maior período de tempo sem as descartar, a probabilidade da entrega aumenta consideravelmente.

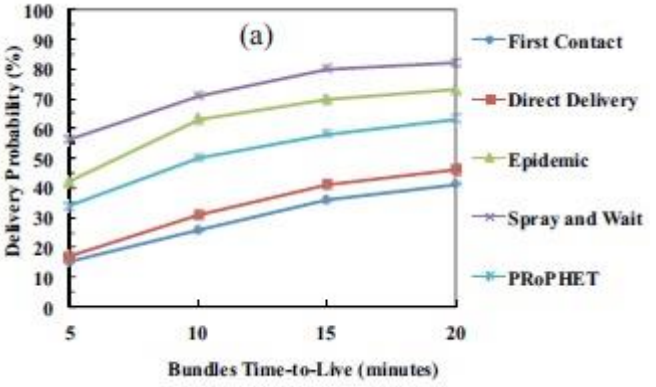


Figura 6 Probabilidade de entrega de *bundles* em função do tempo de vida.

Mais uma vez, o protocolo *Spray and wait* é o método com melhor desempenho na entrega. Os métodos de encaminhamento *single-copy* (*Direct Delivery* e *First Contact*) tiveram o pior desempenho, pois manter apenas uma cópia na rede resulta frequentemente em bundles perdidas antes de as mesmas chegarem ao destino.

2.4. SOLUÇÕES DE CORREIO ELETRÓNICO EM DTN

Em [28] foi feita a modelização de um sistema baseado numa rede ferroviária, que utiliza a arquitetura DTN para permitir a utilização de serviços da internet a utilizadores de aldeias isoladas. Esta solução passa por ter um servidor principal a receber as notícias e *e-mails* endereçados às aldeias. O servidor encontra-se fora da área isolada, pelo que não pertence à rede DTN. Existem ainda estações ferroviárias que recebem os serviços a partir do servidor principal, sendo que estas estações estão encarregues de descarregar os dados para os comboios. Ao passar pelas aldeias, os comboios transmitem os dados destinados às aldeias e recebem ao mesmo tempo os dados provenientes das aldeias. Os principais elementos da arquitetura modelizada são: servidores de notícias e de *e-mails*, uma interface de acesso aos utilizadores, o software necessário para processar todos os dados que circulem na rede segundo o protocolo DTN, bem como um router DTN e a capacidade armazenamento no comboio.

A simulação realizada foi composta por um conjunto de portáteis e *acess points*, com base na arquitetura do sistema. Os portáteis foram equipados com a arquitetura DTN, sendo que 3 portáteis estão fixos e 1 portátil encontra-se em movimento durante a experiência. Este último, simula a movimentação de um comboio, passando na área de cobertura de 3 *access points* ligados aos três outros portáteis e realiza a troca de informação (Figura 7).

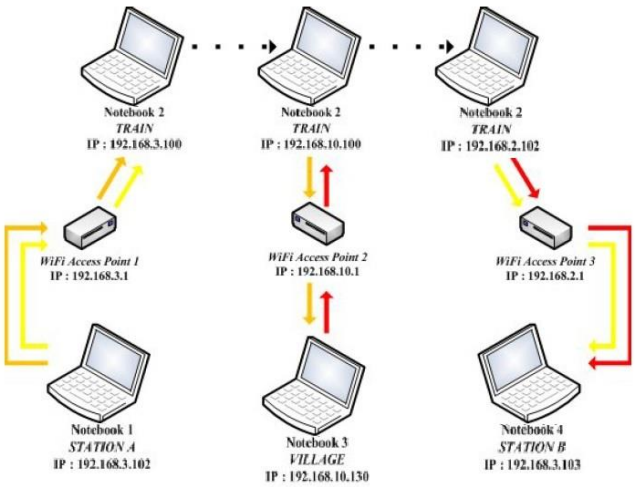


Figura 7 Cenário de testes.

O gráfico presente na Figura 8 mostra os resultados obtidos nas experiências realizadas.

Figura 8 Tempo de transmissão.

Em [29], os autores puseram em prática o que tinham em parte idealizado dois anos antes em [28], e implementaram um sistema que permitiu fornecer o serviço de correio eletrónico a aldeias isoladas. Esse serviço foi fornecido através da instalação de uma rede baseada no conceito DTN, utilizando como parte da infraestrutura um sistema de transportes ferroviário real. O principal objetivo do estudo foi avaliar o desempenho do sistema implementado, verificando a qualidade do serviço de *e-mail* em ambiente real, mesmo quando não existe ligação extremo-a-extremo entre a origem e o destino. Nesta implementação foram utilizados alguns comboios como mulas de transportes de dados, bem como 4 estações ferroviárias. Uma das estações funciona como *gateway*, entre a parte do sistema que utiliza a rede DTN e a parte que não utiliza. No sistema implementado existe ainda um servidor fora da rede ferroviária, designado por Utama, que processa todos os *e-mails* do seu domínio, quer sejam provenientes da internet ou das estações ferroviárias.

Os testes realizados consistiram no envio de um conjunto de *e-mails* provenientes de utilizadores gmail para utilizadores das aldeias e vice-versa. Este processo foi composto por várias fases, tais como os *e-mails* destinados às aldeias serem recebidos pela estação ferroviária, os *e-mail* serem recebidos pelos servidores dos comboios, que por sua vez quando estivessem a passar pelas aldeias realizassem a troca de *e-mails* devida. Os testes realizados mostraram que o serviço de *e-mail* implementado operou corretamente, mostrando que com a utilização de redes DTN é possível transmitir dados através do princípio salto-a-salto, permitindo a utilizadores em áreas remotas efetuar comunicações. Na Figura 9 estão sumarizados os resultados dos testes.

Tests' Location	Amount of E-mail Sent	Total Data Sent (Mbps)	Connection Time	Transmission Time (sec)	Mean Throughpu (Mbps)
Cimekar Sation (Village_11)					
to Router Server on Trip1	300	8.2	1 minutes 52 seconds	11.7277	5.5981
to Router Server on Trip2	400	10.9	2 minutes 34 seconds	27.9866	3.1280
Haurpugur Station (Village_12)					
to Router Server on Trip1	274	7.5	1 minutes 52 seconds	30.1049	1.9919
to Router Server on Trip2	274	7.5	2 minutes 43 seconds	32.6166	1.8385
Baraya Geulis Diesel Train (Router)					
to Cimekar Station on Trip2	246	6.7	2 minutes 44 seconds	14.3104	3,7674
to Haurpugur Station on Trip1	157	4.3	2 minutes 2 seconds	52.0230	0,6674

Figura 9 Sumário dos testes.

Em [30], foi proposta uma solução baseada em DTN que permite estender o legado da internet a regiões remotas, onde não é barato nem viável a existência de infraestruturas de comunicações permanentes. Esta proposta nasceu de um projeto de 36 meses conhecido por N4C (*Networking for Communications Challenged Communities***),** desenvolvido por um conjunto de 12 parceiros europeus, que consistiu no estudo, desenvolvimento e experimentação de uma arquitetura, uma infraestrutura e várias aplicações em ambiente real.

O objetivo principal do projeto foi desenvolver um *software* baseado no conceito de redes tolerantes ao atraso que permitisse estender os serviços da internet a regiões remotas, fornecendo aos utilizadores uma experiência semelhante à vivida na internet. O projeto N4C foi dividido em diversos planos de trabalho, realizados ao longo dos 36 meses de trabalho. Esses planos de trabalho estão descriminados tecnicamente e cientificamente ao longo de dezenas de documentos, que estão disponíveis gratuitamente no *site* do projeto.

As arquiteturas, protocolos, bem como o *hardware* e software desenvolvidos no N4C foram testados e validados em duas *testbeds* reais. Várias séries de testes foram realizadas ao longo dos 3 invernos e 3 verões que existiram durante o prazo do projeto. Testes esses que ocorreram em diversos países, e serviram para aferir o desempenho de diversas aplicações desenvolvidas, bem como da infraestrutura implementada. Entre as aplicações testadas, está um serviço de correio eletrónico, um serviço de conversas instantâneas, transmissão de imagens, recolha de dados meteorológicos, transmissões de ficheiros, entre outras. A infraestrutura utilizada nos testes foi composta por diversos elementos, tais como:

smartphones, computadores, *modems*, câmaras de vídeo, *tablets*, veículos, helicópteros, montanhistas, estações meteorológicas, entre outros [31]. A Figura 10 ilustra o esquema do projeto N4C.

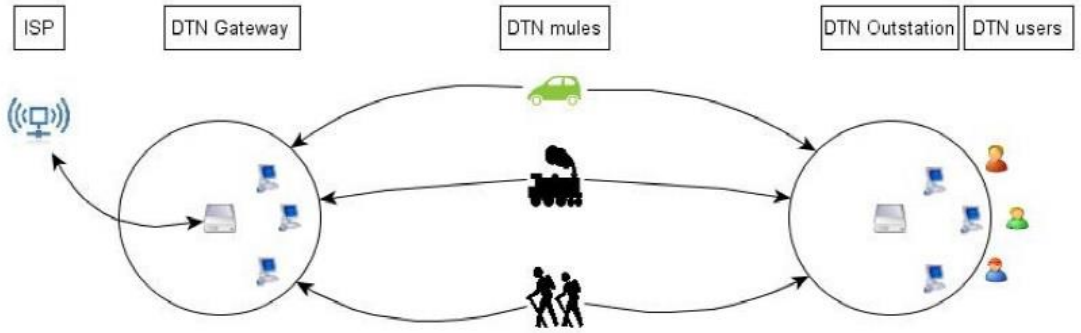


Figura 10 Esquema N4C.

Nos testes realizados, as aplicações desenvolvidas obtiveram os resultados pretendidos, nomeadamente o serviço de *e-mail* denominado Pymail, que foi posteriormente disponibilizado à comunidade no repositório do projeto, e pode ser dessa forma utilizado em trabalhos pós N4C, tal como esta dissertação. Sendo assim, para esta dissertação foi aproveitado o código da aplicação Pymail presente na "gateway" (ver Figura 10), nomeadamente os *scripts* em *python*. O restante código do Pymail, utilizado em outros nós do projeto N4C, para o cenário marítimo abordado não é necessário, pois entre outras funcionalidades, são os scripts de código da gateway que transformam um *e-mail* numa *bundle* e vice-versa.

2.5. RESUMO E DISCUSSÃO

No capítulo 2, para além do sistema de *e-mail* e do conceito de DTN, foram apresentados trabalhos de investigação que envolveram a arquitetura DTN, nomeadamente estudos e experiências realizadas pela comunidade científica nos últimos anos. A partir da investigação realizada foi possível tirar conclusões e apontamentos que se revelaram fundamentais para o trabalho desta dissertação.

No estudo [19], abordado na secção 2.3, a simulação que foi realizada com base no estreito de Singapura, serviu para concluir que os protocolos utilizados em DTN obtêm melhor desempenho no cenário marítimo do que os protocolos tradicionais AODV e OLSR, comprovando que a escolha da arquitetura DTN nesta dissertação foi uma decisão acertada.

Em [24], foi possível através de experiências realizadas num ambiente controlado, verificar que os protocolos de encaminhamento em DTN que utilizam o método *multiple-copy* apresentam melhores resultados face aos protocolos que utilizam *single-copy*.

Na restante investigação efetuada, foi possível verificar que não existe nenhuma proposta que cumpra o principal objetivo desta dissertação, ou seja, até à data não foi desenvolvida qualquer solução de baixo custo que forneça às pessoas que estão em embarcações o serviço de *e-mail*, e que simultaneamente resolva os problemas associados à conetividade intermitente no cenário marítimo. Ainda assim, a comunidade científica já desenvolveu algumas propostas onde o serviço de correio eletrónico é integrado com a arquitetura DTN. Nos cenários das propostas [28], [29] e [30], existem infraestruturas, seja estações presentes nas aldeias e nas cidades, seja o sistema ferroviário, que garantem alguma estabilidade ao sistema, sendo que em todas essas propostas existe um ponto em comum, que é o facto de a origem e destino das mensagens serem nós imóveis. Isto, somado ao facto da movimentação dos nós intermédios ser sempre idêntica, torna o comportamento dos intervenientes e do sistema previsível. Outro ponto em comum entre esses cenários, é o facto de existirem mulas de transporte, isto é, nós que apenas servem para transportar *bundles* de um local para outro, como comboios, pessoas, carros, entre outros, sendo que no caso desta dissertação, não existem exatamente mulas de transporte, pois não existem nós que apenas façam esse transporte de dados. No cenário marítimo abordado nesta dissertação, as embarcações para além de fazer o transporte de dados, têm de ter a capacidade de processar *e-mails* endereçados a um determinado domínio e de proporcionar aos seus utilizadores o serviço de correio eletrónico.

Em contraste com os cenários das propostas apresentadas, no ambiente marítimo a previsibilidade é reduzida, pois não existem trajetos definidos ou estações fixas no oceano que processem o tráfego. A falta de previsibilidade no comportamento das embarcações faz com que as propostas apresentadas na secção 2.4 não possam ser simplesmente transpostas para o cenário marítimo. Nessas propostas, os *e-mails* são encaminhados na rede DTN sobre rotas previamente estabelecidas em cada um dos nós. Rotas essas, que são compostas por um par de endereços, isto é, um "destino" e uma "*gateway*". Esse mecanismo configurado manualmente em cada máquina DTN, indica ao protocolo Bundle a que nó DTN tem de entregar uma *bundle*, para que esta possa chegar a determinado endereço DTN. Posto isto, no cenário marítimo sem existir nós fixos e sem trajetos definidos entre os nós móveis, não

é possível estabelecer rotas pré-definidas entre os mesmos. Sendo assim, e partindo do princípio que todas as embarcações estão equipadas com a arquitetura DTN, optou-se pela utilização de um protocolo de encaminhamento que espalhe cópias das mensagens pelos nós da rede, para que estas possam eventualmente chegar ao seu destino. Porventura, a maior diferença em termos funcionais do sistema de *e-mail* dos cenários terrestres apresentados, para o sistema de *e-mail* do cenário marítimo, é o facto de neste último não ser totalmente garantida a entrega das mensagens ao destino, enquanto que em terra, devido aos percursos definidos e aos nós intermédios que os percorrem, com maior ou menor atraso, os dados são entregues.

Apesar das referidas diferenças, neste trabalho foi possível aproveitar parte do código fonte utilizado noutras propostas, nomeadamente parte do código da aplicação Pymail desenvolvida no âmbito do projeto N4C [30], sendo que esse código já foi reaproveitado em outros projetos, nomeadamente em [29]. Existe portanto a necessidade de desenvolver uma solução em alguns pontos semelhante às que foram aqui apresentadas, isto é, uma solução que permita a utilização do serviço de correio eletrónico sobre a arquitetura DTN, porém, adaptada a um panorama necessariamente diferente como é o caso do cenário marítimo composto por embarcações.

3. ARQUITETURA PROPOSTA E IMPLEMENTAÇÃO

Neste capítulo é apresentada a estrutura do protótipo desenvolvido, o *hardware*, o *software* e o processo de implementação. O protótipo instalado foi desenvolvido tendo por base um cenário real constituído por embarcações, um nó presente na costa marítima e a Internet, como está ilustrado na Figura 11.

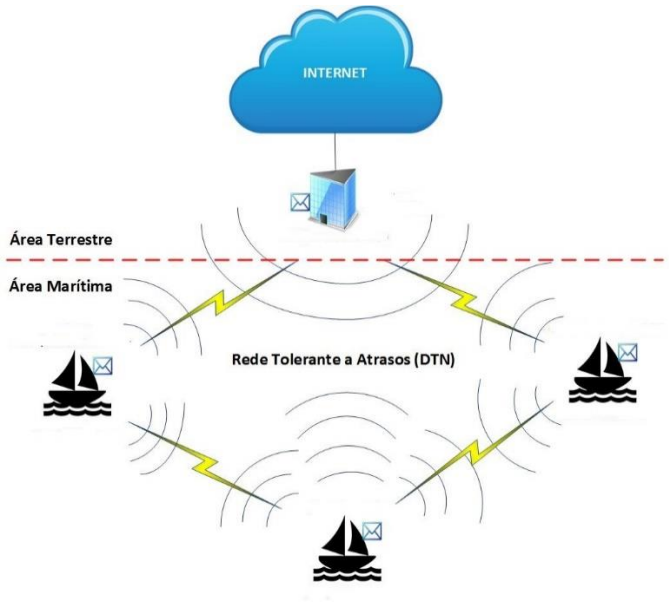


Figura 11 Esquema do cenário marítimo.

O ambiente ilustrado na figura anterior foi modelizado neste trabalho através da utilização de máquinas virtuais. Portanto, todos os nós presentes na *testbed* implementada, sejam eles

representativos de uma embarcação, do nó costeiro ou da Internet, são máquinas virtuais independentes que interagem entre si através da rede interna da plataforma de virtualização.

Visto que esta dissertação teve como principal objetivo a implementação de um serviço de *e-mail* baseado em DTN adaptado ao cenário marítimo, neste capítulo será dada mais ênfase à implementação e configuração do serviço de correio eletrónico nos "barcos" e no nó da costa, e não tanto no nó que representa a Internet na *testbed*.

3.1. ESTRUTURA DO SISTEMA

Antes de avançar com qualquer implementação, teve de ser definida a estrutura da *testbed* e dos elementos que a compõem. Após algum tempo de investigação e reflexão, foi decidido que a *testbed* que viria a servir de plataforma à avaliação do desempenho do serviço de *e-mail* baseado em DTN teria a topologia lógica ilustrada na Figura 12.

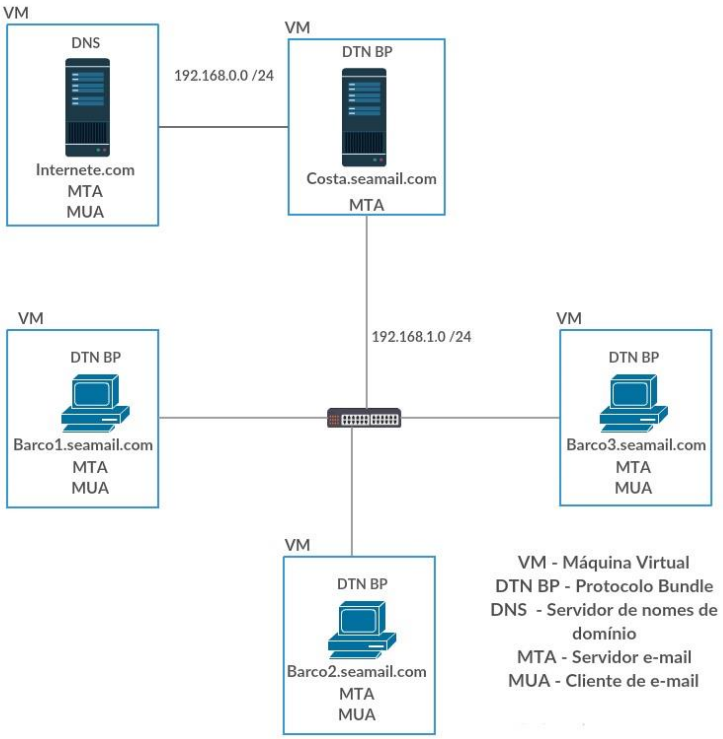


Figura 12 Topologia lógica da *testbed*.

Como se percebe pela Figura 12, o sistema é dividido em três partes, nomeadamente, um conjunto de máquinas com designação "BarcoX.seamail.com" onde o "X" é um número, uma máquina com endereço "Costa.seamail.com" e outra com o nome "Internete.com". Cada uma destas máquinas opera sobre diferentes arquiteturas e deve fazer o tratamento dos *e-mails* conforme a sua função no sistema.

Os nós que representam embarcações apenas conseguem comunicar através da rede DTN. Em cada embarcação, todos os *e-mails* que chegam ao MTA, caso tenham como destino um utilizador de outro domínio, são colocados em *bundles* para entregar. No processo inverso, todas as *bundles* que chegam à embarcação, e têm como destino outro nó, são armazenadas para que seja possível mais tarde fazer a sua propagação na rede. Caso as *bundles* tenham como destino um utilizador da própria embarcação, as *bundles* são entregues a uma aplicação que fará a extração dos *e-mails* e posteriormente entregá-los-á ao MTA.

O nó da costa funciona como uma *gateway* para os *e-mail*s que estão de entrada ou de saída da rede DTN. O nó costeiro tem duas interfaces de rede, que permitem simultaneamente a interação com a rede DTN marítima e com a rede terrestre. Os *e-mail*s que são recebidos pelo servidor da costa, caso tenham como destinatário um endereço com o formato "xxx@barcoX.seamail.com", são entregues à interface DTN na forma de *bundles* para serem transmitidos. Caso o endereço destino dos *e-mail*s não seja nenhum utilizador de domínio "seamail.com", a gestão dos *e-mail*s é feita de forma padrão e estes são entregues através do protocolo SMTP.

A Internet foi representada na *testbed* por um nó que se encontra fora da rede DTN. O nó foi implementado com o propósito de simular um servidor de *e-mail* presente na Internet. Este nó foi configurado como sendo simultaneamente um servidor DNS e um MTA. Para efeitos experimentais foi utilizado este nó para interagir com o nó da costa, de forma a simular situações onde são enviados ou recebidos *e-mail*s de e para a rede DTN.

3.2. *HARDWARE*

Esta dissertação teve como único componente físico o computador portátil Toshiba L755-104 [32]. As fases de investigação, implementação e execução de testes desta dissertação foram na sua totalidade realizadas neste computador. O computador portátil utilizado dispõe de todos os requisitos necessários para o sistema que se pretende implementar e testar. A Tabela 1 tem algumas das caraterísticas do portátil.

Tabela 1 Especificações Toshiba L755-104.

Componente do computador	**Detalhe**
Processador	Intel Core i5-2410M
Memória	4 GB
Placa Gráfica	NVIDIA GeForce GT 525M

Armazenamento	640 GB

3.3. SOFTWARE

Nesta secção é apresentado o *software* que suporta o sistema de e-*mail* implementado, bem como alguma informação acerca da sua utilização no trabalho da dissertação.

3.3.1 PLATAFORMA DE VIRTUALIZAÇÃO

Inicialmente, a escolha da plataforma dividiu-se entre o VirtualBox e o VMware Player por várias razões. Sobretudo porque essas duas aplicações são gratuitas e podem ser instaladas em qualquer sistema operativo [33] [34]. Na pesquisa efetuada verificou-se que as duas plataformas são semelhantes, contudo, o VirtualBox tem algumas funcionalidades que não estão disponíveis na versão Player do VMware, como a clonagem de máquinas virtuais. Apesar do desempenho das duas aplicações ser idêntico, alguns estudos demonstraram que o do VirtualBox é ligeiramente superior [35] [36]. Sendo assim, para a implementação da proposta e para fazer a sua validação foi utilizada a plataforma de virtualização VirtualBox da Oracle [37].

Ao longo do tempo e com o avançar da dissertação, foi possível constatar que o VirtualBox foi uma plataforma suficientemente robusta e que permitiu sem qualquer problemas desenvolver o trabalho que se pretendia. Neste trabalho, a utilização de máquinas virtuais foi uma vantagem, pois foi possível replicar o número de nós sem ter quaisquer custos, sendo mais fácil também simular um cenário com vários nós.

3.3.2 SISTEMA OPERATIVO

O sistema operativo utilizado durante esta implementação foi o Ubuntu 14.10 de 32-bit, versão *desktop*. Este sistema operativo foi utilizado tanto para os nós dos barcos como para o nó da costa. Para o nó que se encontra fora da rede DTN foi utilizado o Ubuntu 14.04, versão *server*. Optou-se por este sistema operativo por várias razões, nomeadamente por ser o sistema mais utilizado em projetos deste tipo, mas também por ter a maior comunidade das plataformas Linux, sendo esta última uma mais-valia no que se refere à resolução de problemas. No que se refere ao desenvolvimento de aplicações em código aberto, a escolha de um sistema operativo com base Linux/Unix é a escolha ideal e indispensável.

3.3.3 AGENTE DE TRANSPORTE DE *E-MAIL* (MTA)

Durante a fase de pesquisa, foram consideradas para servidor de *e-mail*, as aplicações Exim, Sendmail e Postfix. Na pesquisa realizada não foram encontradas diferenças significativas entre os MTAs, contudo, verificou-se que o Postfix é o MTA com a configuração e administração mais acessível, para além de ser uma aplicação com uma vasta documentação de suporte *online* [38]. O Postfix foi então o MTA escolhido para este projeto, porque, essencialmente, cobre todas as necessidades que são exigidas aos servidores de *e-mail* que estão presentes no âmbito deste trabalho. O Postfix foi instalado em todos os nós do sistema, sejam eles embarcações, nó da costa ou nós presentes em terra.

3.3.4 CLIENTE DE *E-MAIL*

Para a realização de testes foi escolhido o cliente de *e-mail* Mutt. Este cliente foi o eleito porque é uma aplicação leve e simples de utilizar que serve perfeitamente para a realização de testes. Contudo, em algumas fases, por ser uma solução mais rápida, foi utilizado durante os testes o utilitário *mailutils* do Postfix, que permite, entre outras coisas, a criação e leitura de *e-mails* com apenas uma linha de código na consola do Ubuntu.

3.3.5 SERVIDOR DNS

De forma a efetuar alguns testes específicos foi necessário instalar um sistema DNS numa das máquinas. Sendo assim, foi instalado o servidor BIND que é o mais utilizado em sistemas operativos Unix e tem um grande suporte *online*.

3.3.6 PROTOCOLO BUNDLE

O protocolo Bundle tem como atual implementação de referência o *software* DTN2 [17]. Esta implementação, desenvolvida pelo *Delay Tolerant Networking Research Group* (DTNRG), incorpora os componentes da arquitetura DTN, enquanto ao mesmo tempo oferece uma plataforma estável e robusta para testes e implementações no mundo real. Contudo, antes de instalar o DTN2 nesta dissertação foi necessário ter em atenção alguns requisitos. O DTN2 é instalado juntamente com um *software* denominado Oasys [39], que foi desenvolvido para dar suporte ao código do DTN2. O protocolo Bundle precisa ainda de uma base de dados persistente, sendo que por omissão é utilizada a base de dados Berkeley [40]. Apesar da base de dados pré-definida ser a Berkeley, é possível utilizar outro tipo de armazenamento, como Mysql [41], PostgreSQL [42], ou mesmo o sistema operativo em si

[43]. Contudo, não existiram razões para se optar por outro tipo de base de dados. Por isso manteve-se a opção pré-definida, e a base de dados instalada foi a Berkeley DB versão 5.3 da Oracle [44]. A instalação do DTN2 tem ainda algumas dependências, pelo que necessita que algumas bibliotecas sejam previamente instaladas.

Visto que o protocolo Bundle está preparado para operar em cenários de comunicações intermitentes, está naturalmente preparado para o cenário marítimo. Sendo assim, em cada nó apenas foi preciso configurar o ficheiro de configuração principal da arquitetura DTN conforme as necessidades. Neste caso, com exceção da configuração de alguns parâmetros obrigatórios, apenas foram adicionados a esse ficheiro agentes de descoberta, que servem basicamente para encontrar vizinhos DTN de forma automática. A outra opção era registar o endereço de todos os nós juntamente com as respetivas rotas, em cada uma das máquinas, que como mencionado anteriormente neste documento, não seria viável. A grande vantagem deste mecanismo é que, caso exista um novo nó na rede DTN, basta configurar essa máquina, não sendo necessário fazer qualquer configuração nas restantes. Um dos parâmetros alterados no ficheiro de configuração foi o tipo de encaminhamento das *bundles*, onde foi definido o método Epidemic. Com este protocolo de encaminhamento, cada vez que existe um contacto entre dois nós DTN, todas as *bundles* armazenadas nas máquinas são replicadas na máquina oposta (flooding). Num cenário composto por embarcações, a grande vantagem deste protocolo é que caso seja necessário enviar uma *bundle* para um nó que está fora de alcance, ao espalhar pela rede cópias dessa *bundle*, a probabilidade de entrega é significativamente maior. É importante referir que o algoritmo de *flooding* utilizado pelo *software* de referência DTN2 permite ter uma inundação de pacotes minimamente controlada, visto que não possibilita a redundância de *bundles* no mesmo nó. Os detalhes da instalação e configuração do protocolo Bundle são apresentados na secção 3.4.1.

3.3.7 INTEGRAÇÃO DO SISTEMA DE *E-MAIL* COM A ARQUITETURA DTN

Os autores do projeto N4C [30] desenvolveram um serviço de correio eletrónico baseado em DTN. Esse serviço, existente na forma de uma aplicação denominada Pymail, contém algumas funcionalidades que são úteis na presente dissertação. Esta é uma aplicação do tipo multitarefa que tem vários processos a correr em simultâneo e funciona continuadamente como um *daemon*.

Uma parte dessa aplicação Pymail é utilizada na "gateway" do projeto N4C (ver Figura 10), e é responsável pela integração do serviço de *e-mail* com a arquitetura DTN, nomeadamente, por colocar os *e-mails* vindos da Internet em *bundles*, encaminhar estas de seguida para a rede DTN, e fazer exatamente o processo inverso, isto é, extrair os *e-mails* das *bundles* e encaminhar estes via SMTP para a Internet. Sendo assim, e visto que no cenário marítimo abordado todos os nós com a arquitetura DTN (embarcações e estação na costa) têm um servidor de *e-mail* incluído, os *scripts* em *python* da aplicação Pymail foram utilizados neste trabalho, pois a principal função destes nós que estão equipados com a arquitetura DTN é igualmente colocar *e-mails* em *bundles* e o processo inverso. Contudo, o código da "gateway" não pôde ser integralmente transposto para os nós DTN que representam embarcações e o nó da costa, pois o tratamento dado às *bundles* tem de ser forçosamente diferente, devido às diferentes caraterísticas de cada cenário. Assim sendo, as alterações substanciais feitas à aplicação, tanto nas embarcações, como na estação da costa, consistiram na modificação de um algoritmo presente num *script* que processa o endereço de destino das *bundles*.

Originalmente, o algoritmo que define, a partir do cabeçalho do *e-mail*, o endereço DTN do destino da respetiva *bundle*, era composto por uma estrutura de condições, delineada de acordo com o cenário de comunicações DTN do projeto N4C. O que foi feito, foi alterar o algoritmo consoante o tipo de nó (embarcação ou nó da costa) e conforme as necessidades do cenário marítimo abordado.

Para além da modificação desse algoritmo, foi necessário fazer alterações adicionais à estrutura de alguns *scripts* devido a problemas encontrados no funcionamento da aplicação desenvolvida pelo N4C. Os problemas encontrados e as respetivas alterações efetuadas à aplicação Pymail estão detalhadas na secção 3.4.2.

3.4. INSTALAÇÃO E CONFIGURAÇÃO

Nesta dissertação, durante a fase de implementação, foram feitos *backups* de máquinas virtuais várias vezes, pois era necessário ter pontos de restauro devido aos problemas encontrados. Algum *software*, nomeadamente o DTN2 e a aplicação que integra o Postfix com o protocolo Bundle, colocou muitos entraves à sua execução, pois devido a incompatibilidades, repositórios obsoletos e outros problemas, não foi possível fazer a sua compilação imediatamente. Os *backups* serviram de base para implementar o sistema,

utilizando durante a instalação, quando necessário, o método tentativa e erro, tentando de várias formas a correta compilação dos programas. Posteriormente, após ter colocado todas as aplicações a funcionar, cada problema que foi aparecendo no envio de *e-mails* teve de ser depurado, e a única forma de fazer a depuração foi enviar mais *e-mails* até descobrir e solucionar o problema.

É importante referir que esta secção apresenta as principais etapas da instalação e configuração do serviço de *e-mail*, nas máquinas que representam embarcações. Ou seja, para esta secção do relatório não se tornar demasiado redundante, parte do trabalho exposto neste capítulo é apenas relativo às embarcações. Contudo, nos momentos em que for relevante apresentar-se-á as configurações realizadas no nó da costa.

3.4.1 PROTOCOLO BUNDLE

A primeira etapa da implementação do sistema consistiu na instalação da plataforma de virtualização VirtualBox e da posterior instalação do sistema operativo. O sistema operativo Ubuntu, bem como os tutoriais de instalação do mesmo estão disponíveis em [45]. Nesta fase não surgiram quaisquer problemas durante a instalação, ficando o sistema operativo preparado para albergar as restantes aplicações. Cada máquina virtual tem as mesmas caraterísticas: 8 GB de armazenamento, 1GB de memória RAM e um núcleo de processamento.

O passo seguinte foi instalar o *software* DTN2. Nesta fase surgiram alguns entraves à compilação e instalação do *software*, nomeadamente a falta de compatibilidade entre alguns elementos, principalmente entre o DTN2 e as suas próprias dependências. A solução utilizada foi testar várias versões do diferente *software*, como o DTN2, Berkeley, Oasys e dependências, de forma a ultrapassar os erros que iam aparecendo aquando da compilação do DTN2. Apesar de se ter tido problemas, foi possível colocar o protocolo Bundle a funcionar nas máquinas, sendo que para isso foram instalados os pacotes que se encontram na Tabela 2, tendo como referência os documentos [46] [47].

Tabela 2 Protocolo Bundle e dependências.

Pacotes instalados	**Tipo**
tcl8.4, tcl8.4-dev, tclx834	Dependências Protocolo Bundle
build-essential	
libdb5.3, libdb5.3-dev	
libxerces-c2-dev	
libxerces-c28	
tclreadline	
Berkeley DB 5.3.28	Base de dados persistente
Oasys 1.6.0	Protocolo Bundle
DTN-2.9.0	

Os pacotes da tabela anterior foram instalados pela ordem listada. As dependências são pacotes que foram instalados a partir do Ubuntu Synaptic Package Manager, sendo que a base de dados Berkeley foi descarregada de [44] e compilada através dos seguintes comandos:

```
../dist/configure
Make
sudo make install
```

Após compilar o Berkeley, foi a vez de instalar o *software* Oasys e o DTN2. Após descarregar o código fonte do Oasys 1.6.0 e do DTN2 a partir de [39], foram utilizados os seguintes comandos para fazer a compilação do código do *software*, executando primeiro no diretório do pacote Oasys e depois no diretório do pacote dtn-2.9.0:

```
./build-configure.sh
CC=gcc CXX=g++ ./configure
Make
Sudo make install
```

Após fazer a compilação com sucesso do DTN2, foi necessário configurar a arquitetura DTN de forma a possibilitar o envio de *bundles* entre os nós. A arquitetura DTN tem um ficheiro de configuração em cada máquina que é analisado cada vez que o *daemon* que ativa a arquitetura é arrancado. Esse ficheiro, denominado "`Dtn.conf`", tem diversos parâmetros que são configuráveis e que permitem ao protocolo Bundle, implementado na máquina, entre outras coisas, assumir diferentes comportamentos na rede. As configurações presentes no ficheiro `Dtn.conf` são utilizadas independentemente das aplicações que correm por cima do protocolo. As principais linhas de configuração do ficheiro `Dtn.conf` estão

discriminadas abaixo, e foram definidas com a ajuda dos documentos [43] [48].

```
Route set type flood
Route local_eid "dtn://barco.seamail.com"
Discovery add discovery_bonjour ip port=2000
Discovery announce hello discovery_bonjour tcp interval=5
cl_addr=192.168.1.5
```

Excerto de código 1 Configuração do ficheiro `Dtn.conf`.

As quatro linhas do Excerto de código 1 são as mais relevantes do ficheiro de configuração, contudo existem outras que podem ser visualizadas no Anexo A. O tipo de base de dados ou o diretório onde são guardadas as *bundles* são algumas das definições configuráveis do ficheiro que não estão especificadas em cima. Das linhas que estão presentes no Excerto de código 1, as duas primeiras são por si só esclarecedoras. Na primeira é definido o tipo de encaminhamento de *bundles*, sendo que o parâmetro flood é relativo ao método de inundar a rede com bundles (*flooding*), e na segunda é definido o endereço do nó DTN (*Endpoint Identifier*). As duas últimas linhas do excerto anterior são comandos responsáveis por criar e gerir agentes de descoberta, que por sua vez são responsáveis por descobrir vizinhos DTN de forma autónoma através da emissão de datagramas UDP. Resumindo, nas máquinas onde foram definidos estes dois últimos comandos funciona um agente de descoberta na porta 2000 gerando anúncios a cada 5 segundos. De máquina para máquina foram alterados alguns parâmetros, entre os quais, o identificador do nó DTN e o endereço IP que aparece na última linha do excerto anterior, retirado do ficheiro de configuração. Este ficheiro de configuração está também incluído no repositório da aplicação pymail [49], contudo o ficheiro `Dtn.conf` utilizado foi configurado a partir de um *template* presente no diretório "dtn-2.9.0/daemon".

Após configurar o ficheiro `Dtn.conf`, o seguinte comando foi executado com o propósito de iniciar a base de dados, uma vez presente no diretório do ficheiro:

```
sudo dtnd –c Dtn.conf --init-db
```

Finalizada a configuração e depois de criar a base de dados, já foi possível testar o funcionamento da arquitetura DTN, utilizando para isso o comando:

```
sudo dtnd –c Dtn.conf
```

Este comando arranca o *daemon* da tecnologia DTN na porta 5050 e lança uma nova consola DTN ao utilizador (Figura 13). Esta consola interpreta comandos específicos reconhecidos

pela arquitetura DTN, que servem para diversos propósitos, como listar vizinhos ou listar *bundles*.

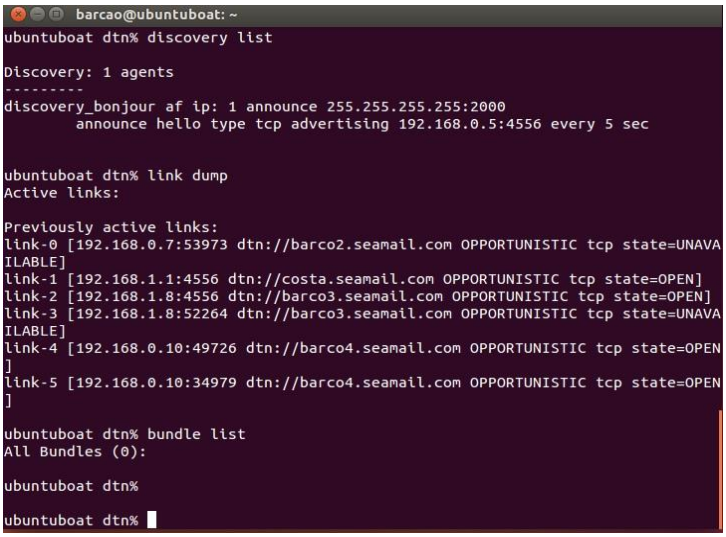


Figura 13 Consola *daemon* DTN.

Até aqui, já era possível verificar o desempenho da arquitetura DTN nas máquinas, o envio de *bundles*, envio de *pings*, etc. Contudo, tendo em conta os objetivos, ainda faltava adaptar o serviço de *e-mail* à arquitetura DTN. Essa fase começou com a configuração do servidor de correio eletrónico em cada máquina.

3.4.2 INTEGRAÇÃO DO POSTFIX COM PROTOCOLO BUNDLE

Nesta subsecção estão presentes detalhes da configuração do Postfix nas embarcações e na estação da costa, bem como da sua integração com a arquitetura DTN. Nesta subsecção, está também exposta toda a adaptação feita ao código da aplicação Pymail, bem como especificadas as diferenças algorítmicas entre a aplicação implementada nos barcos e na costa. De forma a ser facilmente percetível, no que falta deste documento, a aplicação que faz a integração do Postfix com o protocolo Bundle, será várias vezes tratada como Pymail. Nesta subsecção é detalhada a integração do Postfix com o protocolo Bundle numa máquina com nome de domínio barco1.seamail.com, sendo que para outra qualquer embarcação, durante a configuração, bastou incrementar o número associado ao barco, por exemplo "barco2.seamail.com".

A primeira etapa da configuração do Postfix foi a edição do ficheiro `Main.cf` presente no diretório "/etc/postfix". Nesse ficheiro de configuração foram alterados e adicionados alguns parâmetros, como se pode visualizar no Excerto de código 2.

```
Myhostname = barco1.seamail.com
mydestination = barco1.seamail.com, localhost
mynetworks = 127.0.0.0/8 192.168.1.0/24
dtnout_destination_recipient_limit = 1
transport_maps = regexp:/etc/postfix/transport_regexp
```

Excerto de código 2 Configuração ficheiro `Main.cf` do Postfix.

Os dois últimos parâmetros do excerto anterior não pertencem normalmente a este ficheiro de configuração, contudo, foram adicionados tendo em conta a integração que se pretende obter entre o Postfix e o protocolo Bundle. Para o Postfix interagir com a aplicação intermédia, e conseguir entregar *e-mails* à arquitetura DTN implementada, foi necessário configurar o servidor com um mecanismo conhecido por "pipe transport". Este mecanismo do Postfix permite recorrer a outro tipo de transporte, utilizando um parâmetro denominado "transport_maps" no ficheiro `Main.cf`. Neste caso, o servidor de *e-mail* de cada embarcação foi configurado para invocar uma instância do ficheiro Pypop_smtpout.py sempre que apareça um *e-mail* com um determinado destinatário. O primeiro passo para implementar este mecanismo foi adicionar as linhas presentes no Excerto de código 3 ao ficheiro master.cf presente no diretório "/etc/postfix".

```
dtnout unix - n n - - pipe -v
flags=D user=barcao argv=/home/barcao/dtn/python/pypop_smtpout.py
```

Excerto de código 3 Configuração ficheiro `Master.cf` do Postfix.

O passo seguinte foi criar um ficheiro denominado "transport_regexp" no diretório "/etc/postfix". O conteúdo deste ficheiro foi configurado conforme as necessidades da embarcação, e pode ser visualizado no seguinte Excerto de código 4.

```
/^.*\@barco1.seamail.com$/      :
/^.*\.com$/                dtnout:
```

Excerto de código 4 Configuração de Transporte do Postfix de embarcações

Como se pode observar na caixa de texto anterior, o processo `dtnout` é invocado sempre que chegue ao servidor Postfix um *e-mail* com destinatário ".com", sendo que posteriormente é invocado o ficheiro pypop_smtpout.py da aplicação Pymail (ver Excerto de código 3). Neste caso foi definida a terminação ".com" porque o objetivo é transferir para

a rede DTN qualquer *e-mail*, sendo que o ".com" é globalmente o domínio mais utilizado. Contudo este ficheiro é consultado hierarquicamente, portanto, caso o *e-mail* tenha como destinatário um utilizador da própria embarcação, não é invocado qualquer processo e o *e-mail* é entregue na caixa correio do utilizador.

O mesmo ficheiro foi criado e configurado no servidor da costa, porém com algumas diferenças. A leitura hierárquica mantém-se, contudo o processo `dtnout` apenas é invocado quando o servidor tem para entregar *e-mails* a destinatários com um endereço que acabe em "seamail.com". No Excerto de código 5, estão as configurações que foram definidas no ficheiro "transport_regexp" do servidor localizado na costa.

```
/^.*\@costa.seamail.com$/       :
/^.*\.seamail.com$/         dtnout:
```

Excerto de código 5 Configuração de Transporte do Postfix da costa.

Para fazer a ligação entre o Postfix e a arquitetura DTN foi utilizada uma aplicação desenvolvida no projeto N4C denominada Pymail, que entre outras funcionalidades serve para encapsular e descapsular *bundles*. A aplicação necessitou da instalação de alguns requisitos, como os pacotes python, python-dev e python-pysqlite2. A implementação da aplicação teve como suporte os artigos [46] [50]. Como foi mencionado anteriormente, apenas foi utilizada uma parte da aplicação Pymail, sendo que esses ficheiros foram descarregados e adaptados ao protótipo da dissertação, e encontram-se especificados na Tabela 3.

Tabela 3 Ficheiros do Pymail reutilizados.

Ficheiros utilizados do repositório Pymail	
Dtn_pymail_log.conf	Dp_dtn.py
SocketServer2_6.py	Dp_main.py
Dp_msg_send_evt.py	Dp_pfmailout.Py
Dp_pfmailin.py	Pypop_Maildir.Py
Pypop_smtpout.py	Start_dtnd.sh
Stop_nomadic_mail.sh	Start_mail_link.sh
Start_nomadic_mail.sh	

Antes de tentar executar os *scripts* do Pymail, teve de ser instalada uma API do python, conhecida por pythonapi, presente no subdiretório "applib" da pacote dtn-2.9.0, com os seguintes comandos:

```
Make pythonapi
Make pythonapi_install
```

Esta API é necessária pois permite que a tecnologia DTN esteja preparada para trabalhar com a aplicação Pymail. Resumindo, esta interface de programação de aplicações (API) permite à aplicação Pymail utilizar funcionalidades da arquitetura DTN sem se envolver em detalhes do *software* DTN2.

Nesta fase houve vários problemas que consumiram um tempo considerável no trabalho da dissertação. Inicialmente, não estava a ser possível executar a aplicação Pymail devido a alguns erros na abertura do *socket*. Após algum tempo e algumas alterações aos ficheiros `SocketServer2_6.py` e `Dp_pfmailout.py`, foi possível suprimir esse erro e executar a aplicação Pymail. Após conseguir executar a aplicação, foi preciso alterar alguns *scripts* de forma a transformar uma *gateway* do projeto N4C numa potencial embarcação. O primeiro ficheiro a ser alterado foi o `Dp_main.py`, que é o ficheiro que controla todo o funcionamento da aplicação. Este ficheiro é responsável pela gestão de diversos processos importantes para o funcionamento da aplicação, sendo que apenas foram alterados alguns parâmetros de configuração, como o domínio de *e-mail* ou o caminho do ficheiro que guarda os *logs*.

No ficheiro `Dp_dtn.py`, na classe que gere os e-mails que estão de saída da máquina, o algoritmo que determina o endereço DTN do nó destino a partir do cabeçalho do *e-mail*, teve de ser alterado substancialmente, de forma a funcionar num sistema que tem a costa como *gateway* da rede DTN. O Excerto de código 6 mostra o algoritmo alterado presente em `Dp_dtn.py` que faz o processamento do endereço DTN do destinatário do *e-mail*. Este excerto foi retirado de um nó representativo de uma embarcação.

```
destn = evt.destination()
self.logdebug("Message for destination(s) %s" % destn)
if len(destn) == 0:
    self.logerror("No destinations specified: message ignored")
    continue
if '@' in destn[0]:
    [user, dest_domain] = destn[0].split("@")
    if self.domain in dest_domain:
        destn_eid = "dtn://%s/%s" % (dest_domain, EMAIL_IN)
    else:
        destn_eid = "dtn://costa.seamail.com/%s" % (EMAIL_IN
    else:
        self.logwarn("Destination %s does not contain a domain name" %
                     destn[0])
        user = destn[0]
        dest_domain = ""
```

Excerto de código 6 Algoritmo do ficheiro `Dp_dtn.py` presente em embarcações.

Por outro lado, o Excerto de código 7, retirado do mesmo ficheiro `Dp_dtn.py`, presente no nó da costa, mostra que existem diferenças face ao código que existe nas embarcações.

```
destn = evt.destination()
self.logdebug("Message for destination(s) %s" % destn)
if len(destn) == 0:
    self.logerror("No destinations specified: message ignored")
    continue
if '@' in destn[0]:
    [user, dest_domain] = destn[0].split("@")
else:
    self.logwarn("Destination %s does not contain a domain name" %
                 destn[0])
    user = destn[0]
    dest_domain = ""
destn_eid = "dtn://%s/%s" % (dest_domain, EMAIL_IN)
```

Excerto de código 7 Algoritmo do ficheiro `Dp_dtn.py` presente na costa.

Para além deste algoritmo, neste ficheiro foram ainda alterados alguns parâmetros nas embarcações e na costa, tal como se sucede com outros ficheiros. Um dos parâmetros configurados foi o tempo de validade das *bundles*, que foi definido para 24 horas para efeitos experimentais. Sendo assim, cada *bundle* só pode circular pela rede durante 24 horas, sendo descartada após esse tempo. Neste ficheiro `Dp_dtn.py` foi ainda implementado código que possibilita ao utilizador saber se o *e-mail* que foi enviado, foi entregue. Este mecanismo que

foi implementado apenas para efeitos experimentais, consiste numa informação dada, em forma de texto na consola, quando o *e-mail* é enviado e quando é recebido o aviso de receção.

Com a instalação e configuração concluídas, o sistema deveria funcionar eficientemente. Contudo, surgiram alguns problemas e os *e-mails* não estavam a ser passados ao Postfix no nó destino. É importante referir que a aplicação Pymail coloca todo o conteúdo do *e-mail*, ou seja, cabeçalho e corpo de texto, numa *bundle* como se fosse uma simples mensagem. Posto isto, no lado do recetor a aplicação extrai o *e-mail* do corpo da *bundle*, e mais tarde, extrai do *e-mail* os dados que precisa, nomeadamente os endereços eletrónicos do emissor e do destinatário, para passar devidamente o *e-mail* para o Postfix. Após perceber que o problema estava na leitura do cabeçalho do *e-mail* no lado do recetor, foram feitas algumas alterações ao código, tanto no ficheiro que processa os *e-mails* vindos do Postfix, como no ficheiro que entrega, aquando da receção, os e-mails ao Postfix. Os ficheiros em questão são o *Dp_pfmailout.py* e *Dp_pfmailin.py*. No código original do *Dp_pfmailin.py*, na função que faz a entrega do *e-mail* ao postfix são enviados três parâmetros via SMTP, o endereço eletrónico do emissor, endereço eletrónico do destinatário e o *e-mail* em si. Os endereços eletrónicos, principalmente o endereço do emissor, não estavam a ser extraídos corretamente do *e-mail*. Para corrigir esse problema foi feito um acréscimo de informação ao cabeçalho do *e-mail* no lado emissor, para que no lado do recetor exista a possibilidade de saber exatamente quem enviou o *e-mail*. Aqui assuma-se que as alterações feitas ao emissor e recetor, foram realizadas nos ficheiros Dp_pfmailout.py e Dp_pfmailin.py respetivamente. Os Excerto de código 8 e Excerto de código 9 contêm as alterações feitas nos ficheiros Dp_pfmailout.py e Dp_pfmailin.py respetivamente.

```
DTN_Send_HDR = "Received: by"
DTN_SEND_USR = "From: "
DTN_SEND_USRCAPS = "from: "
(…)
if data.startswith(DTN_Send_HDR):
    [rece, domain] = data.split(" (")
    [domains, recet] = rece.split("by ")

if data.startswith("Subject: "):
    variav = 1

if data.startswith(DTN_SEND_USR) or data.startswith(DTN_SEND_USRCAPS):
    if (variav == 1):
       [froms, hostn] = data.split("@")
       [fromms, nome] = froms.split(" ")
       data = "From: %s@%s" % (nome, recet)
msg.lineReceived(data)
```

Excerto de código 8 Alterações feitas ao ficheiro `Dp_pfmailout.py`.

```
DTN_RCPTS_HDR = "To"
DTN_RCPTS_FRO = "from"
DTN_RCPTS_FROO = "From"
(…)
if email_msg.has_key(DTN_RCPTS_HDR):
    rcpts_str = email_msg.get(DTN_RCPTS_HDR)
(…)
if email_msg.has_key(DTN_RCPTS_FRO):
    rcpts_from_str = email_msg.get(DTN_RCPTS_FRO)
elif email_msg.has_key(DTN_RCPTS_FROO):
    rcpts_from_str = email_msg.get(DTN_RCPTS_FROO)
else:
    self.logerror("Email message in %s from DTN doesn't have %s header" %
                                (evt.filename(), DTN_RCPTS_FRO))
(…)
try:
    self.logdebug("Opening link to Postfix")
    smtp_link.connect()
    smtp_link.sendmail(rcpts_from_str, rcpts_str, email_msg.as_string(True))
    self.loginfo("Email in %s passed to Postfix for sending to %s" %
            (evt.filename(), " ".join(rcpts_str)))
except:
    self.logerror("Unable to send message in %s to Postfix." %
            evt.filename())
```

Excerto de código 9 Alterações feitas ao ficheiro `Dp_pfmailin.py`.

Para colocar o sistema a funcionar foi necessário ter a correr o *daemon* DTN e a interface lógica escrita em *python*. Estes dois serviços podem ser iniciados um de cada vez. Contudo a maneira mais fácil de os iniciar é através da execução do script start_nomadic_mail.sh utilizando o seguinte comando:

```
sudo ./start_nomadic.sh
```

Sem iniciar estes dois serviços, a arquitetura DTN não entra em funcionamento, e o Postfix quando invocar o processo `dtnout` não obterá qualquer resposta e não poderá despachar os *e-mails* que tem na caixa de saída.

3.5. TESTES PRELIMINARES

Para a realização de alguns testes específicos foi utilizado um nó localizado fora da rede DTN, configurado com um servidor de *e-mail* e um servidor DNS. A implementação deste nó com interface apenas terrestre, deve-se à necessidade de avaliar o desempenho do nó da costa, no que se refere ao processamento de *e-mails* de e para a rede DTN. Enquanto no cenário marítimo devido às condições do meio, o serviço DNS não pode ser utilizado, na área terrestre a sua utilização é fundamental. Portanto, para o servidor da costa conseguir redirecionar os *e-mails* em terra, é necessário ter um servidor DNS ao qual possa pedir informações. Neste sistema o servidor DNS é o nó localizado fora da rede DTN e foi configurado com o servidor de *e-mail* Postfix e com o software DNS bind9. O Postfix foi configurado com o domínio "mail.internete.com", sendo que no BIND foram configuradas as várias zonas e os seus vários registos, nomeadamente o registo MX. As Figura 14 e Figura 15 mostram como foi configurada cada zona no servidor DNS

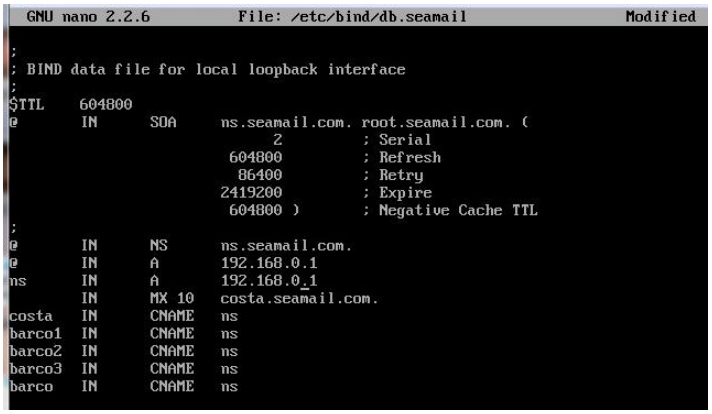


Figura 14 Configuração DNS do domínio seamail.com.

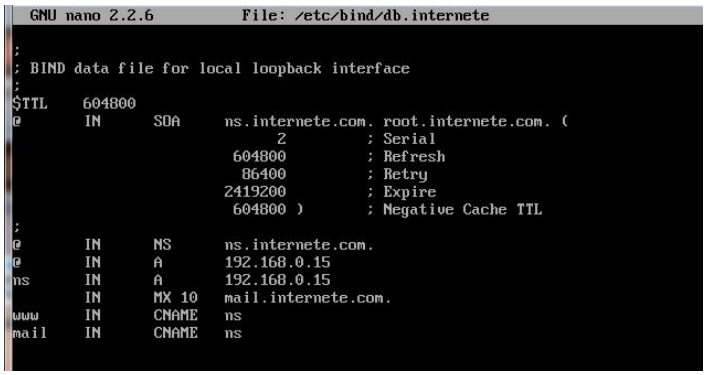


Figura 15 Configuração DNS do domínio internete.com.

Como se pode perceber pelas figuras anteriores, as duas zonas foram configuradas de forma semelhante. Todavia no ficheiro da zona "seamail.com", foram registados alguns "barcos", para que o servidor DNS consiga responder a pedidos de MTAs que têm *e-mails* para entregar a "barcos" do domínio "seamail.com". Após algumas configurações adicionais, o sistema ficou completo e pronto a ser testado.

Antes de verificar se o protótipo desenvolvido está a funcionar como pretendido, foram realizados alguns testes preliminares que tiveram como objetivo aferir o desempenho das aplicações implementadas, nomeadamente o protocolo Bundle (DTN2) e a aplicação que permite a interação do Postfix com o *daemon* DTN. Estes testes feitos às aplicações foram realizados durante a parte final da implementação do sistema, e serviram para perceber se o sistema estava corretamente configurado e pronto para receber um conjunto de testes de maior complexidade.

3.5.1 DESEMPENHO DO PROTOCOLO BUNDLE

As experiências relatadas nesta secção serviram para perceber antecipadamente se o protocolo Bundle instalado, ou seja o *software* DTN2, teria a capacidade de se adaptar ao cenário marítimo. A primeira experiência foi realizada com duas máquinas virtuais em rede e o objetivo foi avaliar o desempenho dos nós, no que se refere à descoberta automática de vizinhos na rede DTN. Este processo de descoberta de vizinhos é a base de todo o sistema implementado. Portanto, o correto funcionamento deste mecanismo é fundamental para todo o projeto. Com este primeiro teste, procurou-se perceber se em cada um dos nós existia informação acerca do seu vizinho, quando estavam todos em rede, ou seja, com o *daemon* DTN a correr em ambos.

A partir das experiências realizadas, foi possível retirar algumas conclusões acerca do comportamento da arquitetura DTN. Se duas máquinas estiverem em rede, e tiverem no seu sistema o respetivo *daemon* DTN a correr, os dois nós irão estabelecer momentaneamente um contato oportunista. A Figura 16 recolhida a partir do nó com o endereço "dtn://barco.seamail.com" mostra uma informação que foi obtida na consola do *daemon* DTN com o comando "link dump", na qual se podem ver os contatos ativos e os inativos.

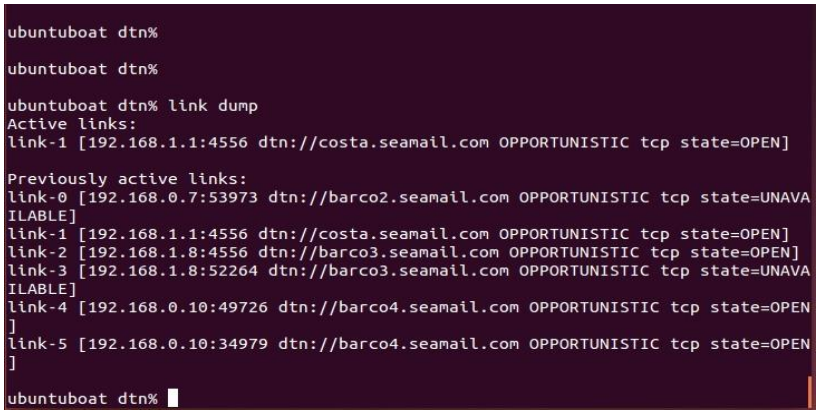


Figura 16 Testes na consola do *daemon* DTN.

Caso um dos nós saia da rede ou o *daemon* DTN seja parado, a ligação passa a indisponível e o contato desaparece das ligações ativas. Como se pode então perceber pela figura anterior, o registo de vizinhos que ocorre de forma automática ficou a funcionar corretamente.

Uma das principais caraterísticas do protocolo Bundle é a transmissão de dados através de *bundles*. Portanto, no sistema que foi implementado neste trabalho os *e-mails* são necessariamente transmitidos na forma de *bundles*. De forma a verificar se o *software* DTN2 instalado estava a funcionar de acordo com o pretendido, foi testado de forma manual o envio de uma *bundle* sobre a plataforma DTN entre dois nós. A *bundle* testada foi uma simples mensagem de texto, escrita e enviada manualmente a partir do nó "dtn://barco.seamail.com", utilizando o seguinte comando na consola:

```
Dtnsend –s dtn://barco.seamail.com –d
dtn://costa.seamail.com/teste –t m –p "ola
costa"
```

Para receber a *bundle* que tinha sido enviada para o endereço "dtn://costa.seamail.com/teste" e para ler o conteúdo da mesma foi necessário utilizar o comando "dtnrecv dtn://costa.seamail.com/teste", como se pode visualizar na Figura 17.

Figura 17 Exemplo de receber e ler uma *bundle*.

Com esta experiência, foi possível verificar que as *bundles* estavam a ser corretamente transmitidas através da rede DTN, e foi simultaneamente possível ter a certeza que o protocolo *bundle* implementado estava totalmente preparado para albergar o resto das aplicações.

3.5.2 INTEGRAÇÃO DO POSTFIX COM A ARQUITETURA DTN

Nesta subsecção estão retratados os testes preliminares que foram realizados de forma a averiguar a interação existente entre o servidor de *e-mail* Postfix e o protocolo Bundle. Ou seja, são verificadas as condições em que um *e-mail* é colocado numa *bundle* e vice-versa. Como mencionado anteriormente, para fazer a integração do Postfix com a arquitetura DTN, foi necessário implementar uma aplicação composta por *scripts* escritos em *python*, desenvolvida originalmente no âmbito do projeto N4C. Após fazer a configuração da aplicação, que está relatada na secção 3.4, foram feitas algumas experiências entre dois nós DTN, de forma a saber se o envio de *e-mails* através do protocolo Bundle estava a funcionar corretamente e se era possível no futuro executar um conjunto de testes de maior complexidade ao sistema no seu todo. A primeira experiência realizada foi o envio de um *e-mail*. O objetivo era verificar se o *e-mail* estava a ser automaticamente convertido numa *bundle* nesse mesmo nó. Por exemplo, na experiência aqui reproduzida, foi enviado do nó "dtn://barco.seamail.com" um *e-mail* com o endereço destino "root@barco2.seamail.com". Na Figura 18 é possível observar o ficheiro "dtn_pymail.log" com os eventos ocorridos na aplicação intermediária após ser criado o *e-mail* e invocado o processo `dtnout`.

```
22:26:55,593 Mailout receiver - 1 INFO     New mailout connection from 127.0.0.1
22:26:55,593 Mailout receiver - 1 INFO     Receiving message for delivery to root@barco2.seamail.com
22:26:55,593 Mailout receiver - 1 INFO     Message to be sent by DTN stored as /home/barcao/maildir/dtnout
5615.M5932769P4732Q1.ubuntuboat,S=391
22:26:55,594 dtn-send           INFO     Sending message to dtn://barco2.seamail.com/email/in
22:26:55,829 dtn-send           INFO     dtn://barco.seamail.com/email/out sent at 494630815, seq no 1391
22:26:55,829 Mailout receiver - 1 INFO     Mailout receiver finishing
```

Figura 18 Excerto do ficheiro dtn_pymail.log da origem da *bundle*.

Para verificar se foi criada efetivamente uma nova *bundle*, foi feita uma listagem das *bundles*, observada através da consola DTN (Figura 19) após o *e-mail* ter sido criado.

```
ubuntuboat dtn%

ubuntuboat dtn% bundle list
All Bundles (1):
        1391: dtn://barco.seamail.com/email/out -> dtn://barco2.seamail.com/email/in length 391

ubuntuboat dtn%
```

Figura 19 Listagem de bundles na consola do *daemon* DTN.

O processo de conversão de um *e-mail* numa *bundle* resultou da interação dos vários componentes instalados na máquina. Na Figura 20 é possível observar um diagrama que mostra de forma geral o comportamento de um nó DTN, aquando da receção de um *e-mail* por parte do MTA e da criação de uma *bundle* sobre a arquitetura DTN.

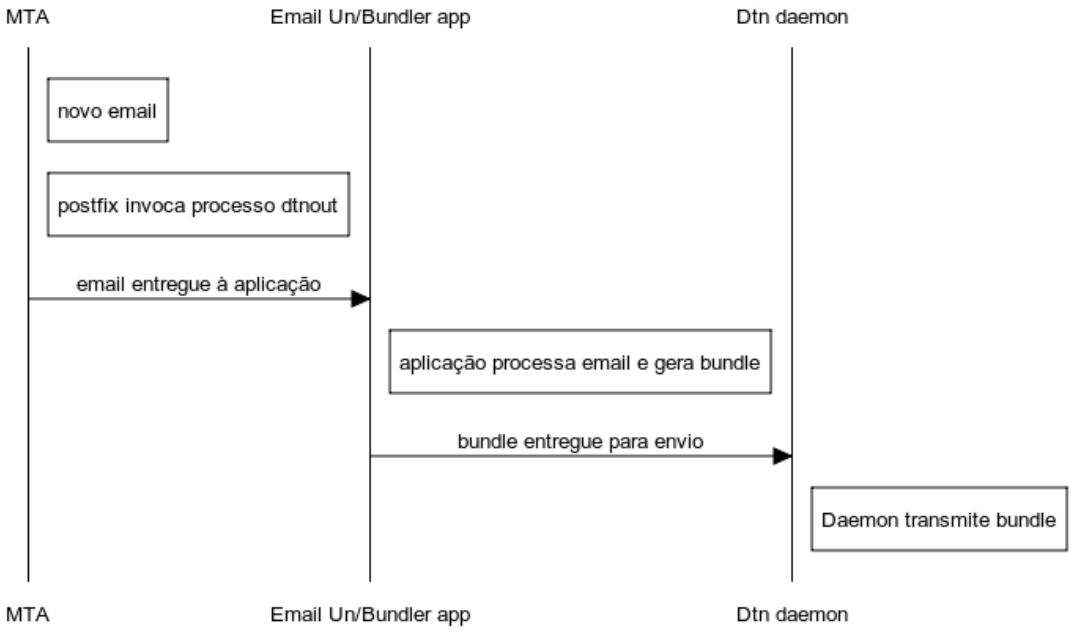


Figura 20 Diagrama de sequência do nó origem de uma *bundle*.

Ainda na continuação da primeira experiência, foi avaliado o desempenho do nó "dtn://barco2.seamail.com", nomeadamente o processamento da *bundle* que foi recebida. Na

Figura 21 é possível observar os eventos presentes no ficheiro "dtn_pymail.log" do "barco2", onde se pode ver que o *e-mail* foi entregue com sucesso ao Postfix.

```
22:29:46,386 dtn-receive        INFO     Received email bundle from dtn://barco.seamail.com/email/out
0815 seq 1391
22:29:46,386 dtn-receive        INFO     New email stored as /home/barcao/maildir/dtnin/new/1441325986
3321Q1.ubuntuboat,S=391
22:29:46,450 postfix-send       INFO     Email in /home/barcao/maildir/dtnin/new/1441325986.M3861510P4
tuboat,S=391 passed to Postfix for sending to < r o o t @ b a r c o 2 . s e a m a i l . c o m >
```

Figura 21 Excerto do ficheiro dtn_pymail.log do destino da *bundle*.

Os eventos que ocorreram no nó "barco2.seamail.com" quando foi recebida a *bundle*, foram praticamente o inverso do que aconteceu no nó que emitiu originalmente a *bundle*. Contudo existiram algumas diferenças, como se pode ver no diagrama da Figura 22.

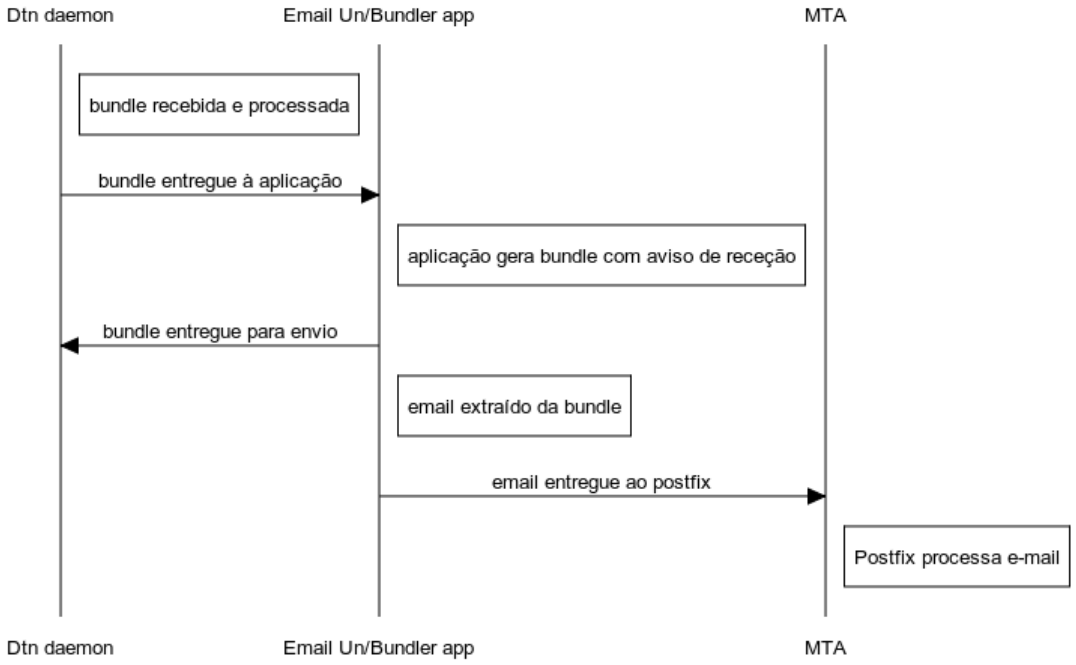


Figura 22 Diagrama de sequência do nó destino de uma *bundle*.

Como se pode visualizar na figura anterior, assim que a aplicação intermediária recebeu a *bundle* com o *e-mail*, criou outra com destino oposto contendo a informação que o *e-mail* foi entregue. Na Figura 23 é possível ver um excerto do ficheiro "dtn.pymail.log", capturado no nó emissor, que contém a informação que o *e-mail* foi entregue.

```
01:19:41,000 MainThread         INFO     DTN Report handler running in thread: dtn-report
01:19:46,817 dtn-report         INFO     Received delivery report re from dtn://barco.seamail.com/email/ou
630815 seq 1391
```

Figura 23 Excerto do ficheiro dtn_pymail.log da origem da *bundle*.

Na consola da máquina que emitiu originalmente o *e-mail*, apareceu ainda em forma de texto na consola uma informação, ilustrada na Figura 24.

Figura 24 Informação recebida no nó origem da *bundle*.

Os testes funcionais que foram retratados nesta secção, demonstraram que o protótipo foi corretamente implementado, e está pronto para se sujeitar a um conjunto de testes específicos baseados em situações que ocorrem no cenário marítimo.

4. AVALIAÇÃO EXPERIMENTAL

Neste capítulo são apresentados os resultados dos testes feitos à solução proposta nesta dissertação. De forma a avaliar o desempenho e viabilidade do protótipo implementado, foi realizado em ambiente controlado um conjunto de testes. Esses testes experimentais consistiram no envio e receção de *e-mails* em diferentes cenários, planeados de forma a aproximar as condições do ambiente controlado às condições das comunicações marítimas. Neste capítulo são também descritas as especificidades de cada cenário utilizado nos testes, nomeadamente as semelhanças de cada um com o contexto real, e as respetivas alterações feitas à topologia de rede das máquinas virtuais. De modo a facilitar a leitura e interpretação da informação presente neste capítulo, os resultados da avaliação experimental foram divididos por várias subsecções, juntamente com a contextualização e caracterização de cada cenário ondes os mesmos foram obtidos.

4.1. PARÂMETROS EM AVALIAÇÃO

Os testes foram feitos num ambiente controlado. Portanto, todas as adversidades existentes no cenário marítimo real não interferiram nos testes, que se focaram na parte aplicacional do sistema. Porém, a execução de testes num ambiente controlado constituído por máquinas

virtuais faz com que não seja vantajoso analisar uma série de parâmetros. Por exemplo, num cenário real seria possível obter medições como a taxa de *e-mails* entregues, o número de *bundles* repetidas ao longo da rede, ou a duração dos encontros com outros nós, ao contrário do que acontece nos cenários utilizados nesta dissertação, onde esses valores são controlados durante a realização dos testes. Por outro lado, num cenário controlado, é possível testar exaustivamente a parte aplicacional do sistema em diversos cenários, sem ter qualquer custo na sua execução, ao contrário do que se sucederia se os testes fossem levados a cabo no ambiente marítimo.

As medições obtidas nos testes, foram basicamente os tempos de entrega dos *e-mails* nos diversos cenários utilizados. Esses tempos de entrega são essencialmente os tempos de processamento dos *e-mails* e *bundles* por parte das aplicações, pois efetivamente, os tempos de transmissão entre os nós são reduzidos devido à alta taxa de débito da interface de rede da plataforma de virtualização. Os tempos foram recolhidos a partir dos ficheiros log presentes no sistema.

4.2. TESTES DE DESEMPENHO ENTRE DOIS NÓS COM CONETIVIDADE PERMANENTE (CENÁRIO 1)

O primeiro cenário de testes foi constituído por dois nós DTN totalmente conectados, simulando uma situação onde dois barcos permanecem algum tempo em contato um com o outro, ou um cenário onde uma embarcação estabelece um contato duradouro com o nó da costa (Figura 25). Este foi um dos cenários escolhidos pela razão de ser uma situação facilmente passível de acontecer no cenário marítimo.

Figura 25 Cenário 1 – Dois nós DTN com conexão estabelecida.

Este cenário foi montado com a ajuda de duas máquinas virtuais configuradas em rede. Em cada máquina virtual foi utilizada uma *interface* de rede ligada a uma rede interna criada no Virtualbox. Na Figura 26 é possível visualizar a topologia de rede utilizada neste primeiro conjunto de testes.

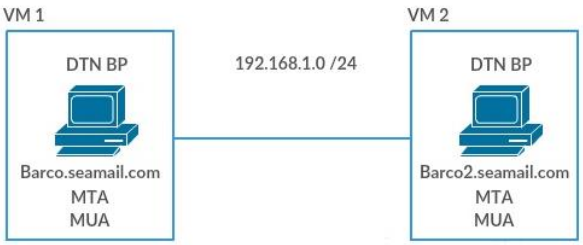


Figura 26 Topologia de rede do cenário 1.

4.2.1 CONDIÇÕES DE TESTE

Os testes consistiram no envio de um conjunto de *e-mails* de um nó para o outro, tendo a ligação DTN sido estabelecida entre os dois nós antes dos *e-mails* serem criados. O objetivo foi testar o funcionamento básico do serviço e avaliar o tempo de entrega dos *e-mails*, sendo que neste cenário foram realizadas duas séries de testes.

Na primeira série de testes, foram enviados *e-mails* até 500 bytes, que foram convertidos em *bundles* de tamanho reduzido. O tamanho padrão para as *bundles* com conteúdo reduzido é 4 kbytes, sendo esse o tamanho das *bundles* geradas que foram transmitidas no primeiro conjunto de testes do cenário 1. Neste caso, o objetivo dos testes foi verificar o tempo que demora a processar um tradicional *e-mail* de texto por parte das aplicações presentes nos dois nós.

Neste cenário foi ainda realizado um outro conjunto de testes. De forma a avaliar o desempenho do sistema quando é sujeito a um maior processamento, foram enviados *e-mails* de diferentes tamanhos entre dois nós DTN, tendo esses *e-mails* um tamanho superior face aos que já tinham sido enviados na experiência anterior. O objetivo dos testes foi igualmente obter o tempo de entrega dos *e-mails* e verificar o desempenho dos nós quando são sujeitos a um maior processamento. Tal como no conjunto de testes anterior, o tempo de entrega de um *e-mail* foi contabilizado desde o momento em que o Postfix entrega o *e-mail* à aplicação intermédia no lado do emissor, até ao momento reverso no lado do recetor. Os *e-mails* que foram analisados nesta série de testes foram concebidos através da anexação de imagens, que deram corpo e tamanho à mensagem. Em avaliação estiveram 4 diferentes tamanhos de *bundles*, começando com 1,1 Mbytes, e indo de forma gradual até aos 8,5 Mbytes, sendo que foram enviadas séries de 10 *e-mails* em cada situação.

4.2.2 ANÁLISE DE DESEMPENHO NO CENÁRIO 1

Neste cenário, foram executadas duas séries de testes descritas na secção anterior. Em cada série de testes foram enviados manualmente 40 *e-mail*s entre dois nós, sendo que na primeira série os *e-mail*s foram constituídos apenas por alguns bytes de texto, e na segunda série foram constituídos por texto e imagens. Os resultados obtidos na primeira série de testes estão ilustrados no gráfico da Figura 27.

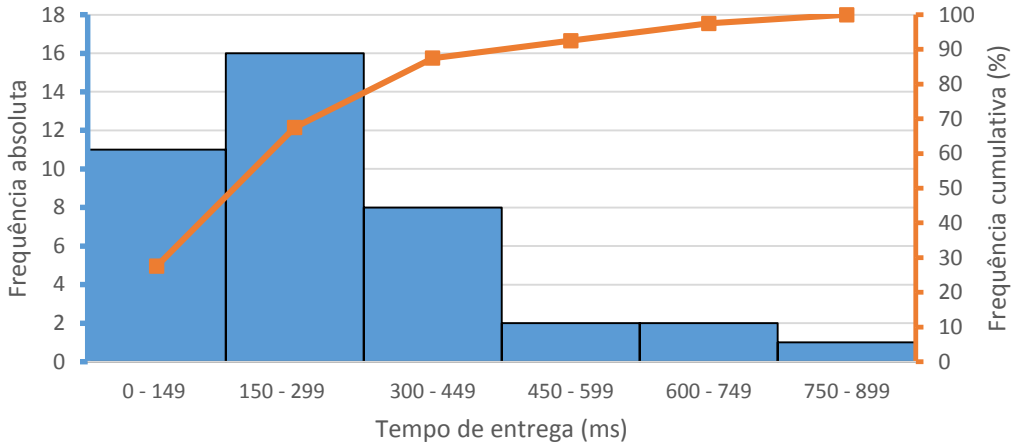


Figura 27 Tempo de entrega de *e-mails* com 4 kbytes.

Na Figura 27, é possível observar que a maior parte dos *e-mails* demoraram menos de meio segundo a serem entregues ao destino, quando as respetivas *bundles* têm 4 kbytes de tamanho. Nas 40 experiências realizadas é possível ver que existiu uma ligeira variação de valores nos resultados obtidos. Isto deve-se necessariamente à capacidade de processamento instantânea das máquinas, aquando da realização dos testes. Contudo, os 40 *e-mails* não foram enviados todos em série, pelo que os primeiros *e-mails* enviados após o estabelecimento da ligação entre os nós, demoraram mais algum tempo a serem entregues.

O segundo conjunto de experiências realizadas no cenário 1, tiveram como resultados os valores ilustrados no gráfico da Figura 28, onde é indicado um intervalo de confiança de 90%.

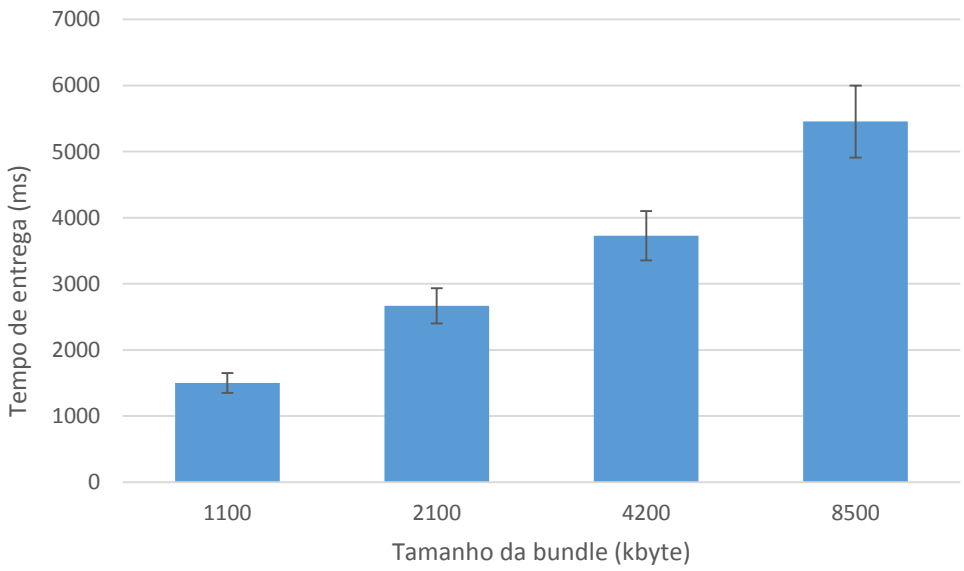


Figura 28 Tempo de entrega de *e-mails* com tamanho variável.

A partir do gráfico presente na Figura 28 é possível visualizar que, apesar do aumento não ser totalmente proporcional, quanto maior for o tamanho do *e-mail* enviado, maior é o tempo despendido na sua entrega. Este facto deve-se essencialmente à quantidade de informação que é necessário processar em cada situação e naturalmente à capacidade de processamento do instrumento de trabalho utilizado nos testes. Como foi referido anteriormente, os tempos de transmissão das *bundles* entre os nós foram reduzidos, portanto, grande parte do tempo gasto na entrega dos *e-mails* foi consumido pelas aplicações do sistema, tanto no envio como na receção.

4.3. TESTES DE DESEMPENHO ENTRE DOIS NÓS COM CONETIVIDADE INTERMITENTE (CENÁRIO 2)

O cenário utilizado nas experiências aqui relatadas foi constituído igualmente por apenas dois nós DTN. Contudo teve algumas diferenças face ao cenário 1, especialmente nas condições em que foram realizados os testes. Os testes realizados neste cenário foram projetados de forma a simular uma situação recorrente das redes DTN, onde dois nós encontram-se e trocam entre si as mensagens que têm armazenadas há algum tempo. No cenário marítimo esta situação poderia facilmente acontecer entre embarcações, ou entre embarcações e o nó da costa (Figura 29), pois devido às caraterísticas do meio, é pouco provável que exista conetividade entre os nós no momento em que as *bundles* são geradas para envio.

Figura 29 Cenário 2 - Dois nós DTN sem conexão previamente estabelecida.

Na Figura 29 é possível visualizar um exemplo enquadrado no cenário testado, onde uma embarcação aproxima-se do nó da costa, criando a possibilidade de comunicação com o mesmo, tanto para receber *e-mails* como para enviar. O cenário 2 assentou em duas máquinas virtuais configuradas, tal como no cenário anterior, em rede. Contudo, antes de criar cada *e-mail*, foi desmarcado um visto nas definições do Virtualbox que garante a conetividade entre as duas máquinas virtuais. Após conceber o *e-mail*, esse visto foi remarcado de forma a permitir conetividade entre as duas máquinas. As constantes alterações feitas às definições das máquinas virtuais durante os testes, tiveram basicamente a mesma funcionalidade que desligar e ligar um cabo de rede entre dois nós. Na Figura 30 é possível observar a topologia de rede do cenário apresentado, dividida em duas fases.

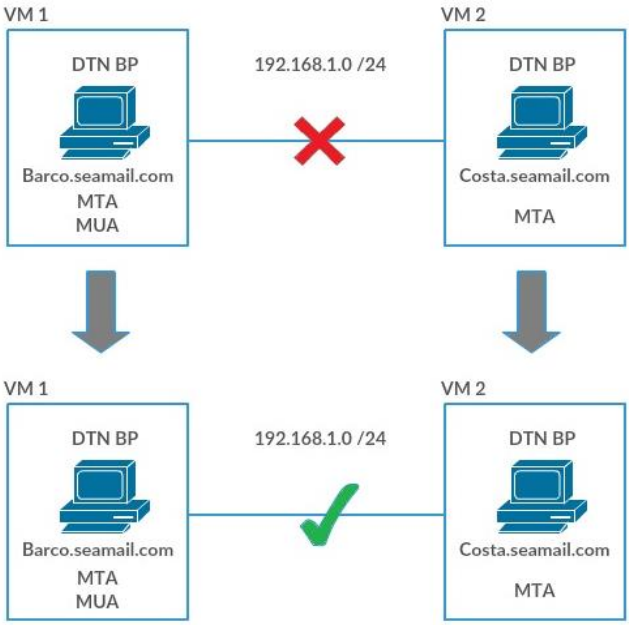


Figura 30 Topologia de rede do cenário 2.

4.3.1 CONDIÇÕES DE TESTE

Nesta série de testes, no lado do emissor, cada *e-mail* foi criado sem haver nesse instante conetividade com o nó recetor. Após o *e-mail* ser criado, a conexão DTN foi restabelecida entre os dois nós. Os *e-mails* foram criados contendo apenas algum texto, originando *bundles* com 4 kbytes de tamanho. O objetivo principal dos testes, desta vez não foi verificar o tempo de processamento dos *e-mails*, foi sim analisar o tempo que dois nós DTN demoram a sincronizar-se e a transmitir os *e-mails* que têm armazenados. Nestes testes, o tempo de entrega de um *e-mail*, foi contabilizado desde o momento em que foi reposta a ligação entre as duas máquinas virtuais, até ao momento em que a aplicação que extrai o *e-mail* entregou o mesmo ao Postfix. Neste cenário não se justificou fazer testes com *bundles* superiores a 4 kbytes, pois os resultados seriam semelhantes aos obtidos com *bundles* de tamanho padrão, visto que o tempo de entrega é maioritariamente composto pelo período de sincronização dos dois nós.

4.3.2 ANÁLISE DE DESEMPENHO NO CENÁRIO 2

Neste cenário foram criados manualmente 40 *e-mail*s, originando respetivas *bundles* com 4 kbytes tamanho. Os procedimentos tomados antes e durante a realização de cada teste estão especificados nas secções anteriores. Na Figura 31 estão ilustrados os resultados obtidos.

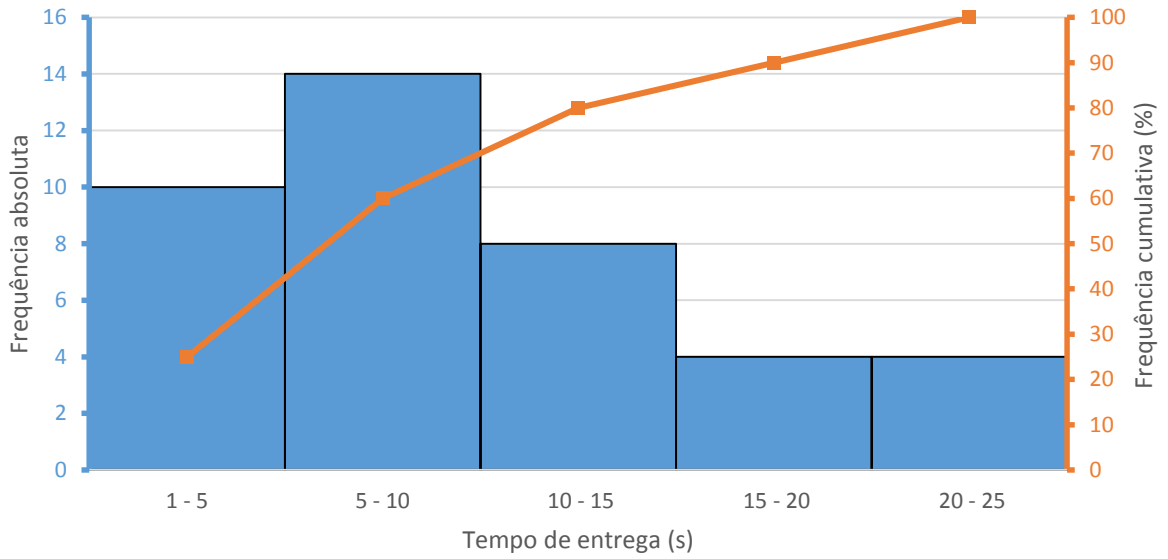


Figura 31 Tempo total de sincronização e entrega de *e-mails*.

A partir do gráfico da Figura 31 é possível verificar que nos *e-mails* enviados, existiu uma grande disparidade de valores nos tempos de entrega dos *e-mails*, visto que 60% deles

levaram menos de 10 segundos a serem entregues e 10 % deles demoraram mais de 20 segundos. Isto deve-se essencialmente ao tempo gasto pelos nós na sua sincronização. Os agentes de descoberta presentes nos nós DTN estão configurados para emitir datagramas UDP a cada 5 segundos. Contudo a sincronização com um vizinho pode demorar mais algum tempo, até porque, muitas vezes, apesar da ligação DTN entre os nós estar criada, leva alguns segundos até ficar ativa.

4.4. TESTES DE DESEMPENHO ENTRE SEIS NÓS COM CONETIVIDADE PERMANENTE (CENÁRIO 3)

O cenário 3 foi constituído por 6 nós DTN parcialmente conectados. Este cenário foi projetado de forma a simular um ambiente marítimo composto por 6 nós, em que por exemplo, 5 nós são embarcações e o restante é o nó presente na costa, sendo que cada nó tem conetividade limitada, pois apenas consegue comunicar com um ou dois nós vizinhos. Num cenários marítimo destes, caso uma embarcação ou o nó da costa pretenda enviar um *e-mail* que tenha como destino um nó que não esteja imediatamente a seu alcance, esse *e-mail* terá de ser passado a outros barcos, que o reencaminharão na rede marítima até ao destino (Figura 32).

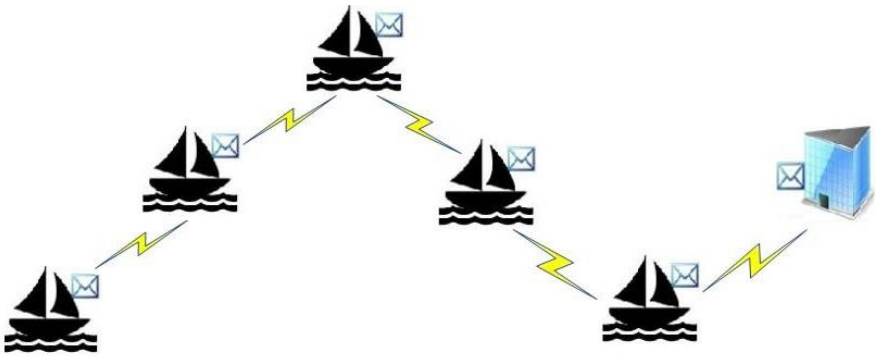


Figura 32 Cenário 3 – Seis nós DTN com conexão estabelecida.

O cenário 3 assentou em 6 máquinas virtuais configuradas em rede. Em cada nó foram utilizadas uma ou duas *interfaces* de rede ligadas às redes internas da plataforma de virtualização. Na Figura 33 está representada a topologia de rede das máquinas virtuais, utilizada no cenário 3.

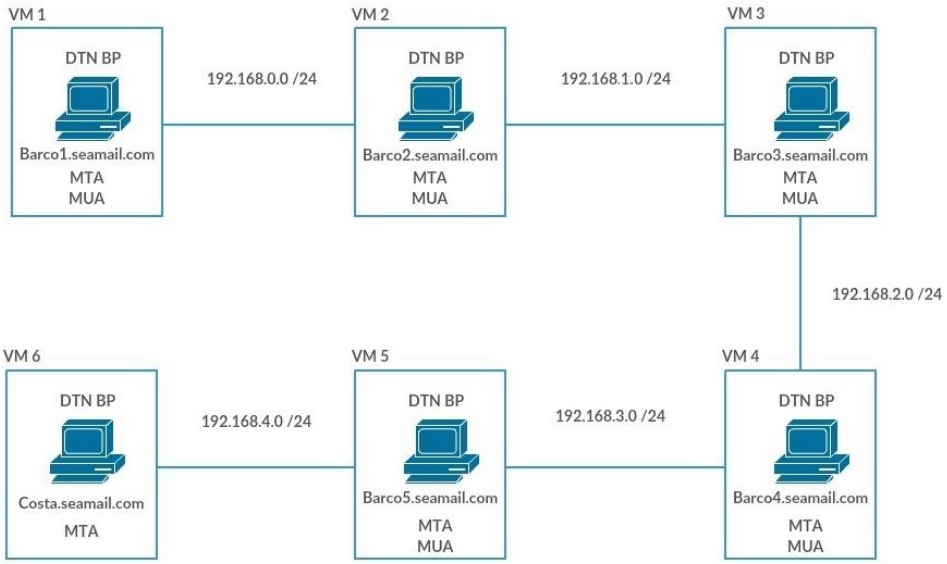


Figura 33 Topologia de rede do cenário 3.

4.4.1 CONDIÇÕES DE TESTE

No cenário 3 foram feitos 5 tipos de testes, cada um deles envolvendo um número de nós diferente. Os testes consistiram no envio de *e-mails* entre um número variável de nós DTN, ou seja, em alguns testes foram utilizados 2 nós, em outros testes 3 nós e assim sucessivamente até aos 6 nós, sendo que o objetivo dos testes foi avaliar o desempenho do serviço, nomeadamente a rapidez, quando o número de *hops* na entrega dos *e-mails* é superior a 1. Em cada conjunto de testes, excetuando os testes com apenas 2 nós, foram utilizadas mulas de transporte entre a origem e o destino, ou seja, nós DTN que apenas são responsáveis por fazer o transporte das *bundles*, não tendo qualquer papel a nível aplicacional. Neste cenário foram enviados *e-mails* de texto, originando *bundles* de tamanho padrão.

4.4.2 ANÁLISE DE DESEMPENHO NO CENÁRIO 3

O terceiro cenário foi elaborado com o objetivo de avaliar a capacidade do sistema, no que se refere à transmissão de *bundles* com a ajuda de nós intermédios. Para isso foram enviados 25 *e-mails* em cada contexto presente no cenário 3. Os resultados obtidos estão presentes graficamente na Figura 34, com um intervalo de confiança de 90%.

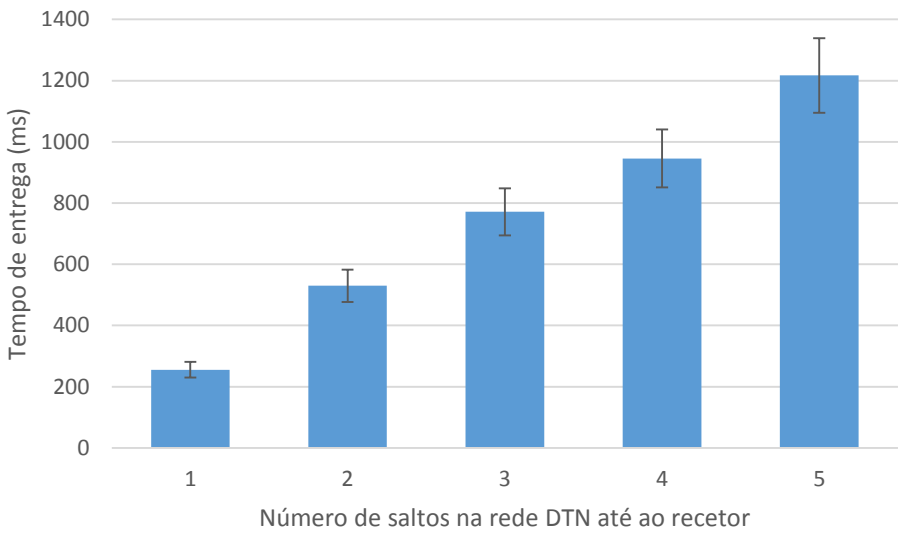


Figura 34 Tempo de entrega de *e-mails* com número de saltos variável.

O gráfico presente na Figura 34 demonstra que o tempo de entrega dos *e-mails* foi praticamente proporcional ao número de saltos realizados na rede DTN até ao recetor, sendo que o atraso médio introduzido por cada salto foi 243,4 ms. Esta proporcionalidade acontece porque todos os nós estão configurados de forma semelhante e são constituídos pelas mesmas arquiteturas. Contudo, o facto de neste cenário terem sido utilizadas várias máquinas virtuais em simultâneo no mesmo computador, pode ter tido alguma influência nos resultados obtidos.

4.5. TESTES FUNCIONAIS ENTRE TRÊS NÓS COM ARQUITETURAS DISTINTAS

O quarto cenário foi constituído por 2 nós DTN totalmente conectados, estando um deles simultaneamente ligado a um terceiro nó que não está equipado com a arquitetura DTN. Este cenário foi projetado de forma a simular uma situação onde um utilizador de uma embarcação troca *e-mails* com um utilizador localizado em terra, utilizando o nó da costa como *gateway* da rede DTN (Figura 35). O objetivo destes testes foi averiguar, em termos funcionais, a eficiência do nó da costa, no que se refere à gestão de *e-mails* que são trocados entre dois nós que possuem diferentes arquiteturas, bem como avaliar na totalidade a solução proposta nesta dissertação.

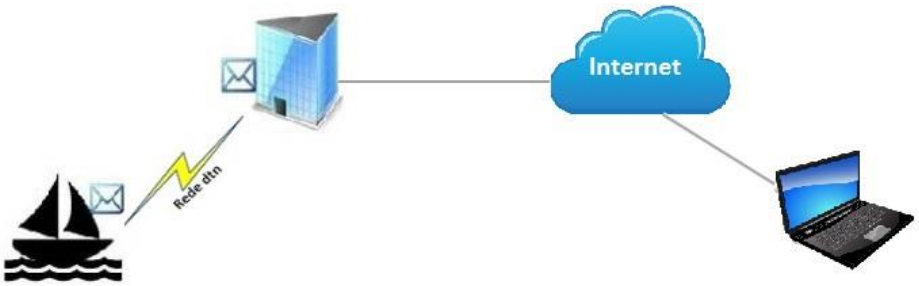


Figura 35 Cenário 4 – Três nós com arquiteturas distintas.

O cenário 4 foi montado com a ajuda de 3 máquinas virtuais configuradas em rede. Contudo de modo a criar o ambiente ilustrado na figura anterior, foi configurada a *interface* de rede das máquinas virtuais, de modo a isolar o nó embarcação do nó que se encontra na Internet, tornando o nó da costa o único ponto de contato entre os dois nós. A Figura 36 mostra a topologia lógica de rede das máquinas virtuais no quarto cenário.

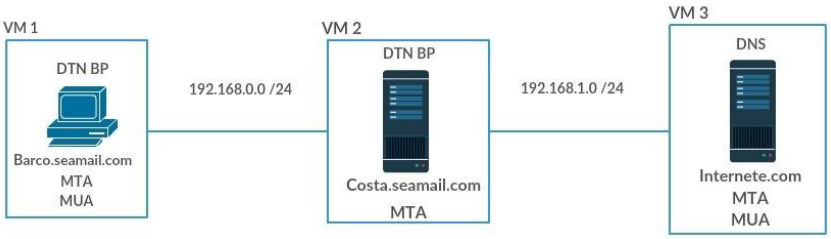


Figura 36 Topologia de rede do cenário 4.

4.5.1 CONDIÇÕES DE TESTE

O cenário 4 foi utilizado para a realização de testes funcionais, que serviram para verificar se o serviço baseado em DTN está preparado para operar em conjunto com o sistema de correio eletrónico utilizado em terra. As experiências realizadas neste cenário, avaliaram a *testbed* no seu todo, incluindo o nó Internete.com, que até aqui não tinha sido utilizado em qualquer teste. Ou seja, neste cenário, a validação feita ao serviço de *e-mail* baseado em DTN, incluiu a utilização do protocolo SMTP para transferir *e-mails*, visto que a conexão entre os nós Costa.seamail.com e Internete.com não assenta sobre a arquitetura DTN. Os testes funcionais realizados neste cenário, corresponderam ao envio de dois *e-mails* a partir dos nós localizados nas extremidades da rede, tendo como destinatário o nó oposto e obrigando à intervenção do nó da costa.

4.5.2 ANÁLISE AO COMPORTAMENTO DA SOLUÇÃO PROPOSTA

Os resultados alcançados no cenário 4 são apresentados nesta secção com recurso a imagens de capturas de ecrã aquando do envio e receção dos *e-mails*, de forma a dar uma melhor perceção do comportamento e eficiência do sistema implementado. Na primeira experiência realizada no cenário 4, foi enviado um *e-mail* do nó Barco.seamail.com, tendo como destino o endereço "internete@internete.com" (Figura 37).

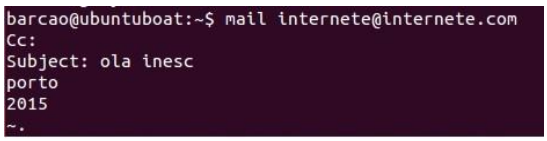


Figura 37 Criação de *e-mail*.

De forma a verificar se o serviço operou como pretendido e se o *e-mail* chegou ao destino, foi consultado o conteúdo do ficheiro mail.log presente no nó Internete.com (Figura 38).

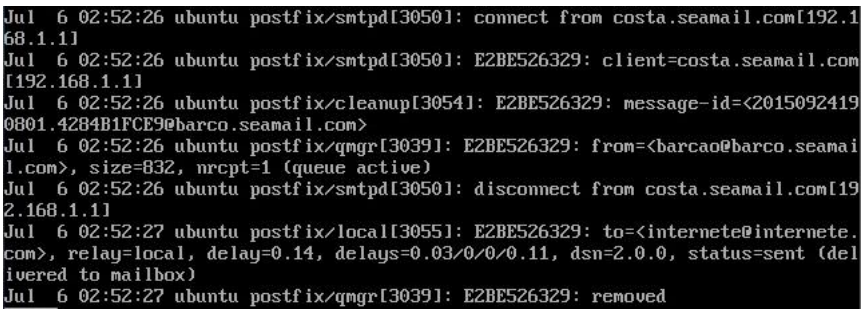


Figura 38 Excerto do ficheiro mail.log.

Como se pode perceber pelas anotações presentes na Figura 38, o *e-mail* enviado originalmente pela máquina "barco.seamail.com, foi entregue ao MTA Internete.com via SMTP por parte do nó Costa.seamail.com, sendo posteriormente colocado na caixa de correio do utilizador (Figura 39).

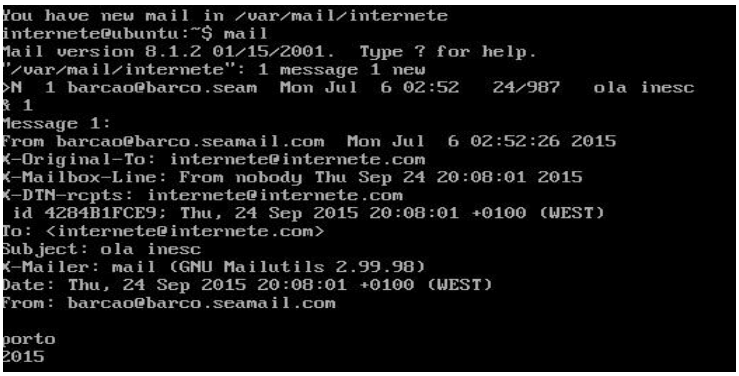


Figura 39 Leitura do *e-mail*.

O primeiro teste funcional realizado no cenário 4, retratado nesta secção com suporte a imagens, teve um resultado globalmente positivo, visto que o *e-mail* foi corretamente entregue ao destinatário. O envio bem-sucedido de um *e-mail*, a partir de um nó que opera com a arquitetura DTN, para um nó que utiliza o protocolo tradicional SMTP, resultou de um conjunto de eventos que ocorreram em várias máquinas. Na Figura 40 é possível observar um diagrama que mostra de forma abstrata o comportamento das três máquinas do cenário 4, aquando da experiência que foi relatada anteriormente.

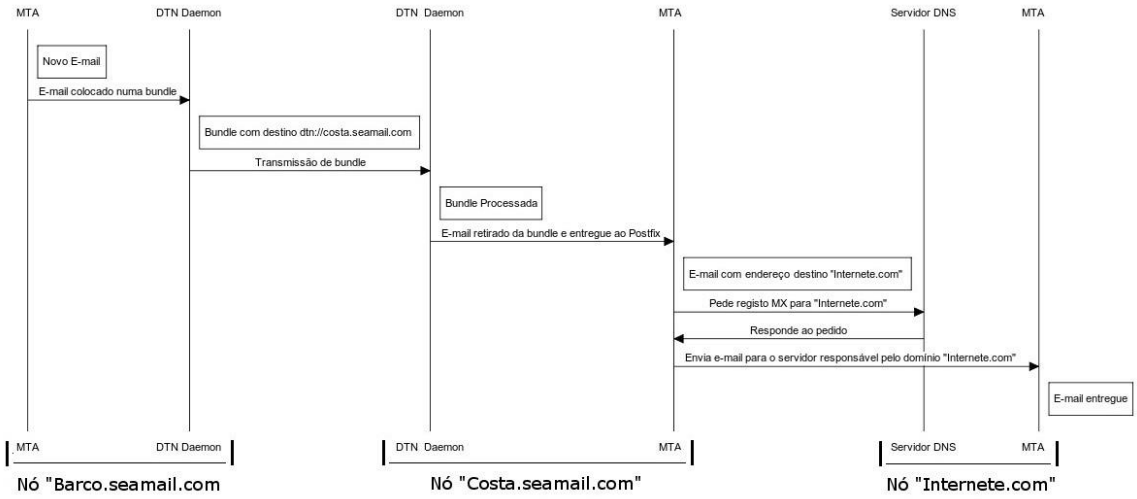


Figura 40 Diagrama de sequência da primeira experiência do cenário 4.

A partir do diagrama anterior, é interpretável que numa primeira fase, o *e-mail* foi colocado numa *bundle* destinada ao nó da costa. Uma vez presente no nó da costa, o *e-mail* foi extraído da *bundle* e foi passado ao servidor Postfix presente na máquina, que posteriormente utilizou o serviço DNS e o protocolo SMTP para entregar o *e-mail* ao nó Internete.com.

Na segunda experiência realizada no cenário 4, foi executado o procedimento inverso, isto é, foi enviado um *e-mail* a partir do nó "Internete.com" com destino ao nó "Barco.seamail.com" (Figura 41).

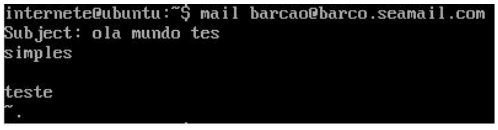


Figura 41 Criação de *e-mail*.

Após algum tempo, o *e-mail* foi entregue no destinatário, como se pode comprovar pelas anotações presentes no ficheiro mail.log do nó "barco.seamail.com" (Figura 42).

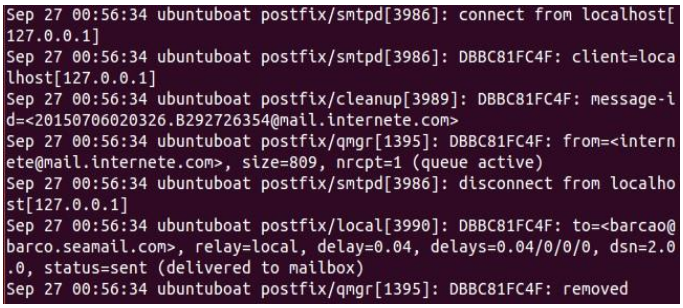


Figura 42 Excerto do ficheiro mail.log.

Mais uma vez, o envio de um *e-mail* entre dois nós com diferentes arquiteturas foi bem-sucedido, visto que o *e-mail* chegou de forma íntegra ao destinatário (Figura 43).

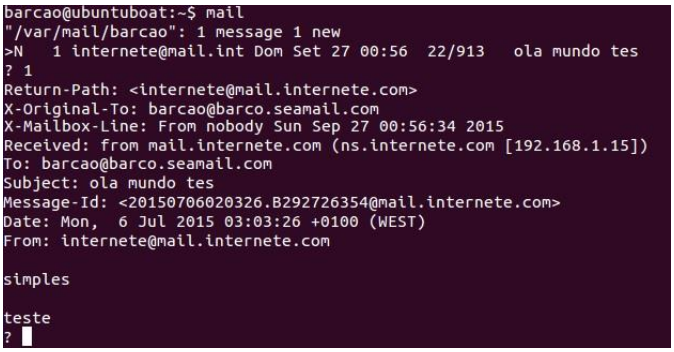


Figura 43 Leitura de *e-mail*.

O envio de um *e-mail* no segundo teste funcional realizado ao cenário 4, resultou novamente de uma série de interações que ocorreram entre os três nós. Tal como no primeiro teste funcional, o *e-mail* foi entregue ao destinatário com o auxílio do nó da costa, sendo que neste caso, o nó costeiro foi responsável por encaminhar o *e-mail* para a rede DTN. Na Figura 44, está representado na forma de um diagrama as interações que se sucederam entre os elementos da *testbed*, aquando da experiência anteriormente relatada.

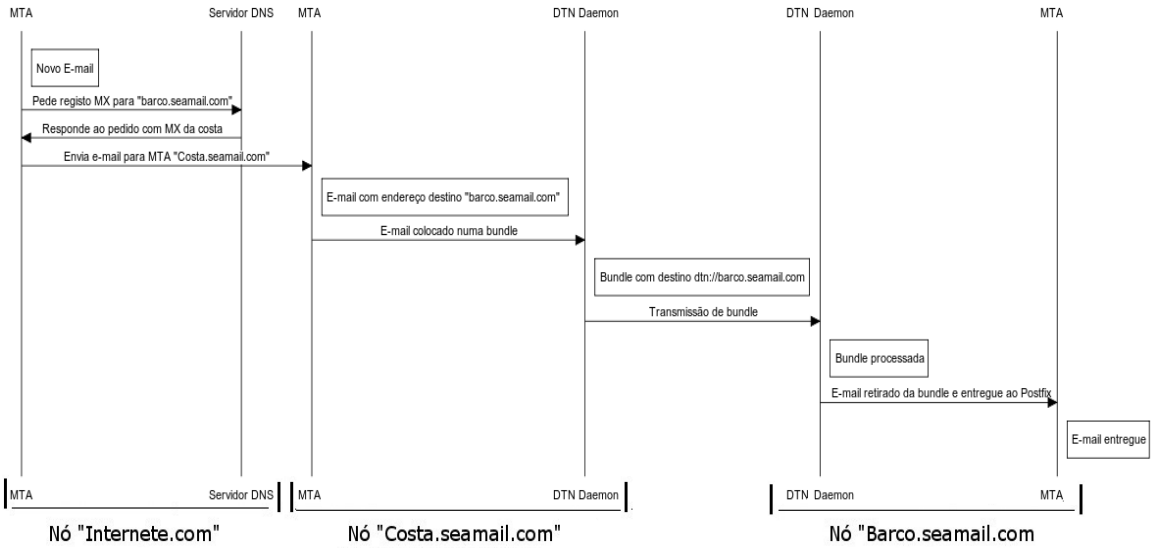


Figura 44 Diagrama de sequência da primeira experiência do cenário 4.

Como se pode perceber pelo diagrama anterior, numa primeira fase o nó Internete.com consultou o sistema DNS para saber o registo MX do servidor responsável pelo domínio "barco.seamail.com". O servidor DNS respondeu ao pedido, informando que o nó Costa.seamail.com, é o responsável pelos *e-mails* endereçados a tal domínio. Uma vez presente no nó da costa, o *e-mail* foi colocado numa *bundle* destinada ao nó Barco.seamail.com. Após a transmissão da *bundle*, o *e-mail* foi extraído da mesma e passado ao servidor Postfix, que posteriormente o deixou na caixa de correio do utilizador.

Os dois testes funcionais realizados no cenário 4 foram conclusivos no que respeita à validação do protótipo desenvolvido. Os dois *e-mails* foram entregues com eficiência, mostrando que o serviço de *e-mail* baseado em DTN que foi adaptado ao cenário marítimo está a funcionar corretamente, e está inclusivamente preparado para operar com o sistema de correio eletrónico existente nas comunicações terrestres, isto é, com o protocolo SMTP.

4.6. CONCLUSÕES E DISCUSSÃO

Após analisar os resultados dos testes práticos realizados à solução proposta, conclui-se que o serviço de correio eletrónico desenvolvido nesta dissertação ficou a funcionar conforme pretendido. Os resultados obtidos mostraram que o sistema de *e-mail* ficou adaptado à arquitetura DTN e ao cenário marítimo. Os cenários utilizados na validação do protótipo desenvolvido foram delineados de forma a recriar situações passíveis de acontecer em ambiente real, sendo que todos os cenários foram igualmente importantes para aferir o desempenho do sistema.

O conjunto de testes funcionais executado no cenário 4, testou de uma forma completa o funcionamento do sistema, pois requereram a transmissão de *e-mails* utilizando o protocolo Bundle, bem como a transmissão de *e-mails* utilizando o protocolo SMTP. Os testes do cenário 4 mostraram ainda que o nó da costa é capaz de gerir o tráfego proveniente de duas redes distintas sem qualquer problema.

Os resultados obtidos das experiências realizadas nos cenários 2 e 3 demonstraram que o serviço de *e-mail* baseado em DTN ficou totalmente preparado para funcionar num ambiente onde a conetividade extremo-a-extremo não está assegurada e onde o modo de funcionamento na entrega dos dados é necessariamente sob o princípio salto-a-salto. No cenário 2 verificou-se que cada nó equipado com o protocolo Bundle está preparado para esperar o tempo que for preciso na entrega dos *e-mails*. Por outro lado, no cenário 3 verificou-se que o método de encaminhamento *flooding* está a funcionar como era expectável, e que o protocolo Bundle viabiliza a utilização de terceiros na propagação das mensagens, quando o nó origem não está ao alcance do nó destino.

Uma das situações que se verificou durantes a realização dos testes, foi que com o avançar das experiências, o tempo consumido pelas aplicações na transmissão dos *e-mails* foi sendo cada vez mais reduzido. Este facto pode dever-se a diversos fatores, nomeadamente, ao estabelecimento de algum tipo de rotinas nas aplicações, que de alguma forma aceleram o processamento dos dados.

Nas experiências realizadas foram testados *e-mails* com texto, *e-mails* com anexos, *e-mails* com texto e anexos, sendo que todos eles foram processados e entregues sem qualquer problema. Isto mostra que a proposta implementada nesta dissertação está preparada para fornecer aos utilizadores de embarcações um serviço semelhante ao que é oferecido em terra, escondendo dos utilizadores a complexidade dos protocolos e das arquiteturas que sustentam o serviço.

5. CONCLUSÕES E TRABALHO FUTURO

No cenário marítimo o acesso à banda larga e a serviços da Internet está limitado às ligações via satélite ou às redes móveis em zonas próximas da costa. O propósito desta dissertação foi desenvolver uma solução de baixo custo recorrendo ao conceito DTN que permita a utilizadores de embarcações utilizar o serviço de correio eletrónico no mar, tirando partido das comunicações sem fios e de estações na costa. Para atingir tal meta, foram definidos vários objetivos, entre os quais, estudar a arquitetura DTN e todas as propostas onde a mesma foi envolvida com o serviço de *e-mail*, fazer um levantamento do software necessário para implementar a proposta, integrar o sistema de *e-mail* com a arquitetura DTN, efetuar testes funcionais e de desempenho, e verificar a viabilidade da proposta desenvolvida para o ambiente marítimo real. Os objetivos foram plenamente alcançados, visto que foi possível implementar e avaliar um sistema de correio eletrónico a operar sobre a arquitetura DTN, tendo em conta um cenário marítimo composto por embarcações. A proposta implementada foi avaliada em termos funcionais e de desempenho em quatro cenários, que foram projetados tendo em conta situações passíveis de acontecer no ambiente marítimo real.

Ao visualizar os resultados obtidos, é possível concluir que o serviço de correio eletrónico implementado sobre a arquitetura DTN está a funcionar corretamente e está adaptado às

condições de comunicação impostas pelo cenário marítimo. A prova disso, é que nas diversas experiências realizadas, quer tenham sido enviados *e-mails* de texto ou *e-mails* com anexos, os mesmos foram sempre entregues e processados sem qualquer problema nos vários cenários.

O facto de a *testbed* ter sido totalmente implementada no mesmo computador sobre uma plataforma de virtualização, tem alguma preponderância nos valores obtidos a partir dos testes de desempenho, visto que a capacidade de processamento do instrumento de trabalho, foi o principal fator que determinou a rapidez de execução na entrega dos *e-mail*s. Contudo, mais importante do que os valores obtidos nas experiências, é constatar a eficiência do sistema proposto ao executar o seu principal propósito, transmissão de *e-mail*s entre dois nós sem existir permanentemente uma conexão *extremo-a-extremo*.

5.1. TRABALHO FUTURO

A validação da solução proposta neste trabalho, indica que o serviço de *e-mail* que foi implementado em ambiente controlado, pode ser transportado para o cenário real, visto que o protocolo Bundle está totalmente preparado para lidar com o ambiente de comunicações marítimo e a parte aplicacional do sistema está totalmente funcional. Posto isto, a solução apresentada nesta dissertação, que é constituída apenas por *software* isento de licença, pode ser no futuro aproveitada no ambiente marítimo real a um custo acessível para os utilizadores, especialmente se for integrada num sistema que utilize tecnologias sem fios sem grandes custos para o utilizador, como por exemplo o WI-FI.

Referências Documentais

[1] M. Lopes, "Wi-Fi broadband maritime communications using 5.8 GHz band," [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7017139&newsearch=true&queryText=m%C3%A1rio%20lopes. [Acedido em 14 Fevereiro 2015].

[2] L. Santos, "Wi-Fi Maritime Communications using TV White Spaces," Faculdade de Engenharia do Porto, Porto, 2013.

[3] J. Klensin, "Simple Mail Transfer Protocol," [Online]. Available: https://www.ietf.org/rfc/rfc2821.txt. [Acedido em 16 Abril 2015].

[4] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall e H.Weiss, "Delay-Tolerant Networking Architecture," [Online]. Available: https://tools.ietf.org/pdf/rfc4838.pdf. [Acedido em 10 fevereiro 2015].

[5] IRTF, "DTNRG," [Online]. Available: https://sites.google.com/site/dtnresgroup/home/about. [Acedido em 16 Maio 2015].

[6] Wikipedia, "Delay-tolerant networking," [Online]. Available: https://en.wikipedia.org/wiki/Delay-tolerant_networking#Research_efforts. [Acedido em 20 Maio 2015].

[7] D. V. Simões, "Delay and Disruption Tolerant Networks - DTNs," [Online]. Available: http://www.gta.ufrj.br/grad/08_1/dtn/ProjetosRelacionados.html. [Acedido em 12 maio 2015].

[8] S. Farrel, V.Cahill, D. Geraghty, I. Humphreys e P. McDonald, "When TCP Breaks Delay- and Disruption-Tolerant Networking," em *Internet Computing, IEEE (Volume:10 , Issue: 4)* , 2006.

[9] H. Arora, "How Email Works? – Email Basic Concepts Explained," [Online]. Available: http://www.thegeekstuff.com/2013/05/how-email-works/. [Acedido em 12 Julho 2015].

[10] How-To-Geek, "HTG Explains: How Does Email Work?," [Online]. Available: http://www.howtogeek.com/56002/htg-explains-how-does-email-work/. [Acedido em 15 Julho 2015].

[11] M. Brain e T. Crosby, "How E-mail Works," [Online]. Available: http://computer.howstuffworks.com/e-mail-messaging/email3.htm. [Acedido em 10 Julho 2015].

[12] J. Myers e M. Rose, "Post Office Protocol - Version 3," [Online]. Available: https://www.ietf.org/rfc/rfc1939.txt. [Acedido em 10 Abril 2015].

[13] M. Crispin, “INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1,” [Online]. Available: https://tools.ietf.org/html/rfc3501. [Acedido em 12 abril 2015].

[14] F. Warthman, “Delay- and Disruption-Tolerant,” [Online]. Available: https://www.ietf.org/mail-archive/web/dtn-interest/current/pdfXSoxd3q0eO.pdf. [Acedido em 1 Março 2015].

[15] K. Scott e S. Burleigh, “Bundle Protocol Specification,” [Online]. Available: https://tools.ietf.org/pdf/rfc5050.pdf. [Acedido em 12 Fevereiro 2015].

[16] G. A. v. Winckler, “DTN - Delay- and Disruption- Tolerant Networking,” [Online]. Available: http://grenoble.ime.usp.br/~gold/cursos/2012/movel/mono-1st/1805-1_Gabriel.pdf. [Acedido em 14 Abril 2015].

[17] DTNRG, “DTN2,” [Online]. Available: https://sites.google.com/site/dtnresgroup/home/code/dtn2documentation. [Acedido em 20 Fevereiro 2015].

[18] IRTF, “DTNRG,” [Online]. Available: www.dtnrg.org. [Acedido em 14 Fevereiro 2015].

[19] H.-M. Lin, Y. Ge, A.-C. Pang e J. Pathmansuntharam, “Performance study on delay tolerant networks in maritime communication environments,” em *OCEANS 2010 IEEE*, Sydney, 2010.

[20] C. Perkins, E. Belding-Royer e S. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing,” [Online]. Available: https://www.ietf.org/rfc/rfc3561.txt. [Acedido em 15 Junho 2015].

[21] T. Clausen e P. Jacquet, “Optimized Link State Routing Protocol (OLSR),” [Online]. Available: https://www.ietf.org/rfc/rfc3626.txt. [Acedido em 20 junho 2015].

[22] A. Vahdat e D.Becker, “Epidemic Routing for Partially-Connected Ad Hoc Networks,” 2000. [Online]. Available: http://issg.cs.duke.edu/epidemic/epidemic.pdf. [Acedido em 10 Março 2015].

[23] T. Spyropoulos, K. Psounis e C. S. Raghavendra, “Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks,” em *IEEE SIGCOMM Workshop*, 2005.

[24] J. Dias, J. Isento, V.N.G.J., F. Farahmand e J. Rodrigues, “Testbed-based performance evaluation of routing protocols for vehicular delay-tolerant networks,” em *GLOBECOM Workshops (GC Wkshps), 2011 IEEE* , Houston, TX, 2011.

[25] T. Spyropoulos, K. Psounis e C. S. Raghavendra, “Single-copy Routing in Intermittently Connected Mobile Networks,” em *First IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (IEEE SECON 2004)*, Santa Clara, California, 2004.

[26] S. Jain, K. Fall e R. Patra, “Routing in a Delay Tolerant Network,” em *ACM SIGCOMM 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, Portland, Oregon, 2004.

[27] A. Lindgren, A. Doria, E. Davies e S. Grasic, “Probabilistic Routing Protocol for Intermittently Connected Networks,” [Online]. Available: http://tools.ietf.org/html/draft-irtf-dtnrg-prophet-06. [Acedido em 25 Junho 2015].

[28] E. M. Husni e A. Sumarmo, “Delay Tolerant Network utilizing train for news portal and email services,” em *2010 International Conference on Information and Communication Technology for the Muslim World (ICT4M),*, Jakarta, 2010.

[29] E. Husni e A. Wibowo, “E-mail System for Delay Tolerant Network,” em *2012 International Conference on System Engineering and Technology (ICSET),*, Bandung, 2012.

[30] E. C. S. F. Programme, “N4C,” [Online]. Available: http://www.n4c.eu/. [Acedido em 1 Fevereiro 2015].

[31] N4C, “D2.1 N4C System Architecture,” [Online]. Available: http://www.n4c.eu/Download/n4c-wp2-004-sys-arch-10.pdf. [Acedido em 12 Março 2015].

[32] Toshiba, “Satellite L755-104,” [Online]. Available: http://www.toshiba.pt/discontinued-products/satellite-l755-104/. [Acedido em 25 Julho 2015].

[33] J. Fitzpatrick, “Five Best Virtual Machine Applications,” [Online]. Available: http://lifehacker.com/5714966/five-best-virtual-machine-applications. [Acedido em 15 Março 2015].

[34] J. O'Gara, “When one operating system is not enough: The five best virtual machine applications,” [Online]. Available: http://www.digitaltrends.com/computing/best-virtual-machine-apps-for-mac-linux-and-windows-pcs/. [Acedido em 12 Março 2015].

[35] Y. A. Khan e M. Amro, “PERFORMANCE EVALUATION OF VIRTUAL MACHINES,” [Online]. Available: http://www.academia.edu/5497134/Performance_Evaluation_of_Virtual_Machines. [Acedido em 14 Março 2015].

[36] S. G. Langer e T. French, “Virtual Machine Performance Benchmarking,” [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3180542/. [Acedido em 25 Março 2015].

[37] Oracle, “VirtualBox,” [Online]. Available: https://www.virtualbox.org/. [Acedido em 19 Março 2015].

[38] E. Simard, “What is the best mail server on Linux?,” [Online]. Available: http://www.gtcomm.net/blog/what-is-the-best-mail-server-on-linux/. [Acedido em 25 Fevereiro 2015].

[39] Dtnrg, “Delay tolerant networking,” [Online]. Available: http://sourceforge.net/projects/dtn/files/. [Acedido em 16 Fevereiro 2015].

[40] Oracle, “Berkeley DB,” [Online]. Available: http://www.oracle.com/technetwork/database/database-technologies/berkeleydb/overview/index.html. [Acedido em 10 fevereiro 2015].

[41] “MySQL,” [Online]. Available: https://www.mysql.com/. [Acedido em 25 Maio 2015].

[42] “PostgreSQL,” [Online]. Available: http://www.postgresql.org/. [Acedido em 16 Maio 2015].

[43] N4C, “D2.2: Functional Specification for DTN Infrastructure Software,” [Online]. Available: http://www.n4c.eu/Download/n4c-wp2-023-dtn-infrastructure-fs-12.pdf. [Acedido em 16 Março 2015].

[44] ORacle, “Berkeley Download,” [Online]. Available: http://www.oracle.com/technetwork/products/berkeleydb/downloads/index-082944.html. [Acedido em 15 Fevereiro 2015].

[45] Ubuntu, “Ubuntu,” [Online]. Available: http://www.ubuntu.com/. [Acedido em 14 Abril 2015].

[46] N4C, “D7.4 Evaluation and updates of the integration platform,” [Online]. Available: D7.4 Evaluation and updates of the integration platform. [Acedido em 10 Março 2015].

[47] A. Azfar, “ByteWalla: DTN on Android phones DTN2.6 Installation Guidelines,” [Online]. Available: https://archive.ssvl.kth.se/csd2010/www.tslab.ssvl.kth.se/csd/projects/1011248/sites/default/files/DTN2.6_Installation_Guidelines_revised.pdf. [Acedido em 10 Março 2015].

[48] A. Azfar, “ Bytewalla: DTN2 with Neighbor Discovery and Bundle Flooding,” [Online]. Available: https://archive.ssvl.kth.se/csd2009/www.tslab.ssvl.kth.se/csd/projects/092106/sites/default/files/Neighbor_Discovery_and_Bundle_flooding.pdf. [Acedido em 10 Março 2015].

[49] n4C, “Pymail Files,” [Online]. Available: http://info.n4c.eu/code/PyMail/file/0765c38ab38a/gateway/dtn. [Acedido em 16 Março 2015].

[50] A. Azfar, “ Bytewalla: Integration of POSTFIX with DTN2,” [Online]. Available: https://archive.ssvl.kth.se/csd2009/www.tslab.ssvl.kth.se/csd/projects/092106/sites/default/files/Postfix_DTN2_Integration.pdf. [Acedido em 11 Março 2015].

Anexo A. Ficheiros de configuração da arquitetura DTN

Nó embarcação, ficheiro `Dtn.conf`:

```
log/dtnd info “dtnd parsing configuration…”
console set addr 127.0.0.1
console set port 5050
set shorthostname [lindex [split [info hostname] .] 0]
console set prompt "$shorthostname dtn% "
storage set type berkeleydb
storage set server_port 62345
storage set schema /etc/DS.xsd
set dbdir "/home/dtn/bundlestore"
if {$dbdir == ""} {
    foreach dir {/var/dtn /var/tmp/dtn} {
   if {[file isdirectory $dir]} {
           set dbdir $dir
           break
   }
    }
}
if {$dbdir == ""} {
    puts stderr "Must create /var/dtn or /var/tmp/dtn
storage directory"
    exit 1
}
storage set payloaddir home/barcao/bundlestore
storage set dbname DTN
storage set dbdir /home/barcao/db
route set type flood
route local_eid “dtn://barco.seamail.com”
interface add tcp0 tcp local_port=4556
interface add udp0 udp
discovery add discovery_bonjour ip port=2000
discovery announce tcp0 discovery_bonjour tcp interval=5
cl_addr= 192.168.1.5
log /dtnd info "dtnd configuration parsing complete"
```

Nó costa, ficheiro `Dtn.conf`:

```
log/dtnd info “dtnd parsing configuration…”
console set addr 127.0.0.1
console set port 5050
set shorthostname [lindex [split [info hostname] .] 0]
console set prompt "$shorthostname dtn% "
storage set type berkeleydb
```

```
storage set server_port 62345
storage set schema /etc/DS.xsd
set dbdir "/home/dtn/bundlestore"
if {$dbdir == ""} {
    foreach dir {/var/dtn /var/tmp/dtn} {
   if {[file isdirectory $dir]} {
           set dbdir $dir
           break
   }
    }
}
if {$dbdir == ""} {
    puts stderr "Must create /var/dtn or /var/tmp/dtn
storage directory"
    exit 1
}
storage set payloaddir home/barcao/bundlestore
storage set dbname DTN
storage set dbdir /home/barcao/db
route set type flood
route local_eid “dtn://costa.seamail.com”
interface add tcp0 tcp local_port=4556
interface add udp0 udp
discovery add discovery_bonjour ip port=2000
discovery announce tcp0 discovery_bonjour tcp interval=5
cl_addr= 192.168.1.1
log /dtnd info "dtnd configuration parsing complete"
```