UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

CONTROLADOR DE REDES EM CÓDIGO ABERTO

Tiago Luís Calha Maurício

MESTRADO EM INFORMÁTICA

Trabalho de projeto orientado por:
Prof. Doutor Hugo Alexandre Tavares Miranda
e pelo Eng. Gustavo Alberto Vouga de Carvalho Homem

2018

Agradecimentos

Gostaria de agradecer ao professor Hugo Miranda, ao Engenheiro Gustavo Homem e à Ângulo Sólido pela paciência, ajuda e orientação no desenvolvimento deste projeto.

Quero agradecer à minha família que sempre estiveram presentes para apoiar e ajudar mesmo nos momentos mais difíceis. Quero também agradecer aos meus amigos por todo o bom ambiente criado e também por toda a ajuda.

Este trabalho foi parcialmente suportado pela FCT, através do LASIGE, ref. UID/-CEC/00408/2013

Resumo

As redes de computadores são constituídas por diversos componentes que desempenham papéis diferentes, tornando a gestão da rede uma tarefa relativamente complexa. Uma solução para este problema é a centralização dos mecanismos de gestão de alguns componentes da mesma através de um controlador de rede.

Neste projeto foi desenvolvido um controlador de redes chave na mão, baseado em código aberto, utilizando a distribuição Ubuntu 16.04 LTS como sistema operativo. O controlador implementa diversas funcionalidades tais como encaminhamento (*routing*), antepara de segurança (*firewall*) e ponto de acesso sem fios. Este controlador suporta diversas configurações e permite configurações com redundância de ligações WAN para implementação de tolerância a uma falta de ligação à *Internet*. O *software* do controlador desenvolvido é instalado e configurado de forma automática e é destinado a ambientes empresariais.

O desenvolvimento de um controlador baseado em tecnologias de código aberto permite à instituição acolhedora, Ângulo Sólido, providenciar aos seus clientes uma solução vantajosa em relação às soluções proprietárias, reduzindo os custos associados à sua aquisição e providenciando uma configuração específica para o cliente em questão.

Palavras-chave: código aberto, encaminhamento, antepara de segurança, automático, controlador

Abstract

Computer networks are composed of distinct components that have different roles, making the network management a fairly complex task. One solution to this problem is the centralization of the management mechanisms of some of the network components through a network controller.

In this project a turnkey open-source network controller was developed using the Ubuntu 16.04 LTS Linux distribution as the operating system. The controller implements a number of features such as routing, firewall and wireless access point. This controller supports several configurations and allows configurations with redundancy of WAN connections for implementation of fault tolerance. The controller's software is installed and configured automatically and is intended for enterprise environments.

The development of a controller based on open source technologies allows the welcoming institution, Ângulo Sólido, to provide its customers with an advantageous solution over proprietary solutions, reducing the costs associated with the acquisition of the controller and providing a specific configuration for each customer.

Keywords: open source, routing, firewall, automatic, controller

Conteúdo

Lista de Figuras

Lista de Tabelas

Listagens

Capítulo 1

Introdução

1.1 Motivação

As redes de computadores são constituídas por diferentes componentes, cada um desempenhando um papel. Encaminhadores, comutadores, anteparas de segurança e pontos de acesso são exemplos de equipamentos necessários para garantir comunicação segura entre computadores. A coexistência dos diversos equipamentos dentro de uma rede obriga a uma manutenção e gestão especializada e diversificada.

O equipamento que faz a ligação física e a transição do tráfego interno para a rede exterior denomina-se *gateway*, neste documento considera-se que um controlador de redes é uma *gateway*. Este componente desempenha um papel fundamental na rede, sendo o único ponto de entrada e saída. Se este componente operar defeituosamente pode prejudicar todo o normal funcionamento da rede e ter repercussões significativas na instituição. Um controlador de rede pode albergar diversos componentes de uma rede local, permitindo a centralização de alguma parte dos serviços da rede em questão, por exemplo encaminhamento, autenticação e segurança, facilitando assim a sua gestão, a gestão dos dispositivos pertencentes a essa rede e dos dispositivos hospedados na mesma.

As soluções existentes no mercado, no que toca a controladores de rede, são soluções proprietárias de custo elevado ou soluções providenciadas pelos ISP (*"Internet Service Providers"*), que desempenham algumas das funcionalidades desejadas. No entanto, o equipamento não é customizável, não existindo, à luz do meu conhecimento, soluções diferenciadas com custos razoáveis. As soluções providenciadas pelos ISP não permitem ao utilizador efetuar configurações para além do fornecido pela interface de gestão do dispositivo. O facto de estes equipamentos executarem *software* proprietário impede a customização do mesmo, não permitindo adicionar componentes de *software* para desempenhar funcionalidades necessitadas pelos clientes, por exemplo um servidor de VPN (*"Virtual Private Network"*). Considerando ainda que a heterogeneidade, em termos de versões do *software*, modelos e fabricantes, eleva a complexidade da gestão destes equipamentos e a pouca escalabilidade, em número de clientes, de uma solução utilizando

os dispositivos providenciados pelos ISP, leva a colocar esta hipótese de lado. Mesmo existindo soluções no mercado baseadas em código aberto, os dispositivos são configurados pelo fabricante e não desempenham as funcionalidades pretendidas.

A impossibilidade de providenciar aos seus clientes uma solução de controlador de redes de custo moderado com configuração específica, constitui a motivação para o desenvolvimento de um controlador de redes em código aberto por parte da instituição acolhedora.

1.2 Objetivos

O objetivo final deste projeto é o desenvolvimento de uma solução chave na mão, baseada em tecnologias de código aberto, que ofereça não só as funcionalidades providenciadas por um dispositivo fornecido pelos ISP, mas também serviços e funcionalidades à medida para o cliente. A solução desenvolvida deve ser escalável no sentido do número de clientes e na heterogeneidade das suas necessidades. Cada cliente necessita de uma configuração específica, pelo que a sua instalação, configuração e reposição têm de ser procedimentos simples para que possa escalar. Pretende-se desenvolver uma solução com custo moderado, que facilite a gestão e manutenção da rede local do cliente, incluindo partes da rede com pontos de acesso sem fios, e beneficiando da utilização de código aberto.

Pretende-se que a solução permita abstrair, do cliente, pormenores técnicos pois este pode não ser perito na área, para tal a solução deverá ter as seguintes funcionalidades: dupla ligação WAN (*"Wide Area Network"*) para tolerância a faltas, instalação automática, em prol da escalabilidade da solução, encaminhamento (*routing*), antepara de segurança (*firewall*) e tradução de endereços (NAT), assegurando todos os serviços necessários ao funcionamento de uma rede, bem como disponibilizar serviços de autenticação e suportar utilização de redes privadas virtuais. A solução irá também efetuar modelação de tráfego para garantir qualidade de serviço no acesso à *Internet*.

Este projeto tem também como objetivo a integração do aluno no contexto da instituição acolhedora, introduzindo-o a conceitos novos como DevOps.

1.3 Instituição de Acolhimento

A Ângulo Sólido é uma empresa de prestação de serviços informáticos, fundada em 2005, que trabalha nas áreas de arquitectura, implementação e administração de sistemas e é especializada em tecnologias de código aberto. A Ângulo Sólido presta diversos serviços a pequenas e médias empresas e instituições governamentais. Nos últimos anos tem-se especializado na metodologia de *"DevOps"*, que consiste na unificação do desenvolvimento e da operação de *software*, aplicada a *startups* tecnológicas.

A Ângulo Sólido está localizada em Lisboa e Évora e presta serviços como virtualização, gestão de Linux empresarial, instalação, configuração e manutenção de redes, desenvolvimento *web* e também formação em tecnologias de código aberto.

1.4 Estrutura

O presente documento apresenta a seguinte estrutura:

- **Introdução**, onde é feita uma introdução ao tema do projeto. É apresentada a motivação deste projeto, os objetivos, a instituição de acolhimento e a estrutura do documento;
- **Estado da Arte**, onde é feita uma contextualização do estado atual da tecnologia relacionada bem como de trabalho existente na área;
- **Controlador de Redes**, onde é descrito todo o trabalho realizado, tecnologias e ferramentas utilizadas.
- **Avaliação**, onde são descritos os testes realizados e apresentados os resultados da avaliação realizada;
- **Conclusões e Trabalho Futuro**, onde são discutidos os pontos mais relevantes do projeto e as perspetivas em relação a trabalho futuro.

Capítulo 2

Estado da Arte

A tecnologia utilizada nas infraestruturas de rede, tanto em LAN como em WAN, evoluiu bastante na última década, progredindo, por exemplo, de tecnologias como o DSL/ADSL (*"Asymmetric Digital Subscriber Line"*) para fibra ótica e comunicação utilizando tecnologia 4G. Em termos de serviços de *Internet* por parte dos ISP, evoluímos, por exemplo, de larguras de banda como 5 Mbit/s de *download* para 1 Gbit/s[19].

A melhoria dos serviços de *Internet* e das tecnologias das infraestruturas de rede, permitiu um aumento do número de instituições que utilizam sistemas informáticos para gerir e manter o seu negócio, necessitando de redes locais mais complexas e com mais dispositivos. A dificuldade da gestão dessas redes aumenta com o crescimento da própria rede. Dado este problema, a utilização de controladores de rede permite reduzir alguma complexidade na gestão de uma rede local.

2.1 Equipamento de Rede

A figura 2.1 demonstra uma composição genérica de uma rede local com duas ligações à *Internet* utilizando dois ISP diferentes. Esta configuração é constituída por duas ligações independentes, criando redundância física e implementando tolerância a faltas de serviço. Considerando a figura e os elementos nesta representados, há quatro conceitos que nos interessam particularmente: encaminhadores (*routers*), pontes (*bridges*), comutadores (*switches*) e pontos de acesso (*access points*). Estes quatro conceitos apresentam funcionalidades implementadas por diversos equipamentos. Nesta secção é feita a distinção entre estes conceitos e explicadas as suas funções.

2.1.1 Encaminhadores (*Routers*)

O papel de um encaminhador consiste em encaminhar pacotes de dados entre redes de computadores distintas. São os dispositivos que conectam toda a *Internet*.

Os encaminhadores possuem uma tabela de rotas (*"routing tables"*). Estas tabelas contêm uma lista de redes que dita para que interface de rede devem ser encaminhados

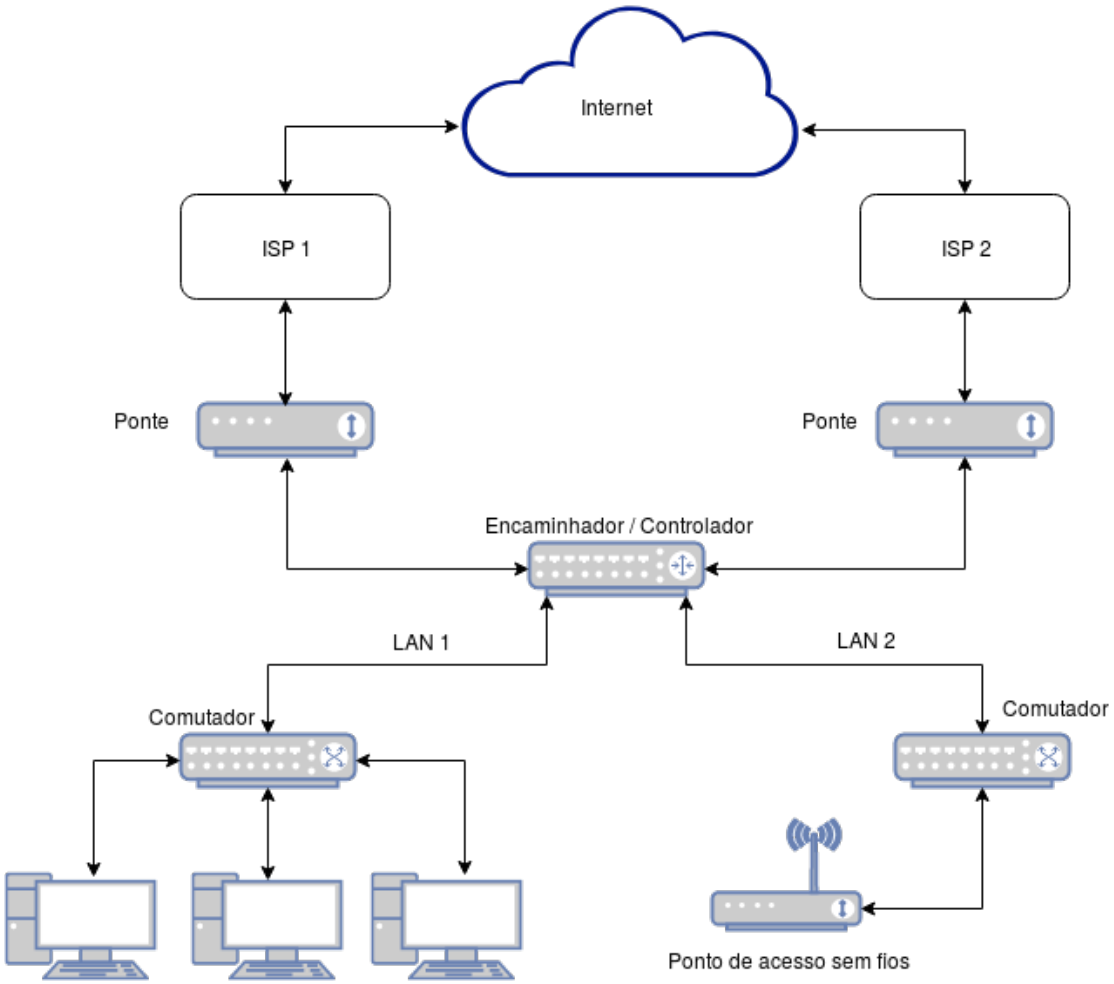

Figura 2.1: Diagrama genérico de uma rede local.

pacotes com destino a essas redes. A tabela 2.1 apresenta um exemplo simplificado dos conteúdos de uma tabela de rotas que incluem o destino do pacote, a interface de rede para onde deve ser encaminhado e o respetivo endereço IP do *gateway* a ser utilizado.

Destination	Network mask	Gateway	Network Interface	Metric
192.168.5.0	255.255.255.0	-	eth1	100
192.168.1.1	255.255.255.0	-	eth0	100
default	0.0.0.0	192.168.1.1	eth0	100

Tabela 2.1: Exemplo de uma tabela de rotas

Ao receber um pacote, o encaminhador identifica o seu endereço de destino e compara com as suas entradas da tabela. Se o endereço pertencer a alguma rede registada na sua tabela, por exemplo 192.168.5.13 pertence à rede 192.168.5.0, com máscara de 24 bits, encaminha o pacote para a respetiva interface de rede. Neste caso, seria a interface eth1. Irá sempre existir uma linha de encaminhamento por omissão para que, caso o

endereço destino do pacote não pertença a nenhuma rede listada na tabela, o pacote seja encaminhado para outro encaminhador. Neste exemplo, o endereço IP 192.168.1.1 que se encontra acessível pela interface de rede eth0. O encaminhador por omissão assegura que, mesmo não tendo conhecimento de onde se encontra a rede de destino, o pacote será encaminhado para um dispositivo capaz de assegurar a continuação da rota.

2.1.2 Pontes (*Bridges*)

Existem várias tecnologias e protocolos que permitem transportar informação entre redes. Por exemplo, dentro de uma rede local, utiliza-se, maioritariamente *Ethernet* para fazer as ligações entre dispositivos e para o transporte de dados. As pontes fazem a projeção entre diferentes tecnologias, permitindo assim que a heterogeneidade existente nas infraestruturas seja transparente. As pontes localizam-se em extremidades de redes, por exemplo pontos de acesso sem fios contêm pontes que efetuam a projeção do protocolo IEEE 802.11 (redes sem fios) para o protocolo IEEE 802.3 (*Ethernet*). A distribuição de serviço, por parte dos ISPs (*"Internet Service Provider"*), é feita utilizando DSL (*"digital subscriber line"*), fibra ótica ou redes 4G. Para que seja possível estabelecer uma ligação, com os serviços do ISP, é necessário instalar uma ponte para fazer o mapeamento da tecnologia utilizada na infraestrutura do provedor para a tecnologia utilizada na rede local. No caso de DSL, utiliza-se um *Modem* de DSL. Caso seja fibra ótica, utiliza-se um ONT (*"optical network terminal"*). Estes dois equipamentos fazem a conversão entre DSL e *Ethernet* e entre fibra ótica e *Ethernet*, respetivamente, sendo dois exemplos de pontes.

2.1.3 Comutadores (*Switches*)

Enquanto que um encaminhador conecta computadores entre redes diferentes, um comutador faz a interligação dos computadores dentro de uma rede. Os comutadores contêm uma tabela que dita quais os dispositivos que estão ligados em cada uma das suas portas. No entanto, ao invés de uma tabela de rotas, a tabela de um comutador é gerada com utilização. Quando um comutador arranca, a sua tabela está vazia. Ao receber uma trama de uma porta, o comutador identifica o endereço MAC (*"Media Access Control"*) de origem e de destino e compara com as linhas da tabela. Caso o endereço de origem não esteja na tabela, adiciona-o à tabela da porta por onde a trama foi recebida. Se o endereço de destino já estiver registado, encaminha a trama para a porta associada ao endereço. Caso contrário, envia para todas as portas têm um cabo ligado.

2.1.4 Ponto de Acesso Sem Fios

Tipicamente referimo-nos a estes dispositivos como AP apesar da sua designação ser WAP (*"Wireless Access Point"*). Este equipamento consiste numa placa de rede com uma antena rádio ligada a uma componente com fios da rede. Este equipamento

permite acesso à LAN (*"Local Area Network"*) a partir de uma porção da rede sem fios designada por WLAN (*"Wireless Local Area Network"*). Os APs têm limitações, tanto por questões legais, por não se poder transmitir ondas rádio fora de uma determinada gama de frequências ou acima de uma determinada potência e por questões físicas, sendo difícil, por exemplo, fazer o sinal trespassar paredes. Obstáculos como estes são um problema para a cobertura do sinal emitido pelos pontos de acesso, constituindo um motivo para a instalação de diversos dispositivos destes em locais estratégicos.

2.2 Soluções Integradas

Os equipamentos anteriormente referidos servem diferentes propósitos dentro de uma rede. A gestão de quatro tipos diferentes de equipamentos não é um trabalho simples, devendo ser feita por uma equipa de profissionais. Numa LAN simples, por exemplo doméstica, torna-se desnecessariamente complicado fazer a gestão e manutenção de todos estes dispositivos. Existem então algumas combinações destes equipamentos que permitem reduzir a complexidade da sua gestão e os custos associados à sua aquisição:

- Ponte + Encaminhador
- Encaminhador + Comutador
- Ponte + Encaminhador + Comutador
- Comutador + Ponto de acesso
- Ponte + Encaminhador + Comutador + Ponto de acesso

Tipicamente os ISP fornecem aparelhos do último tipo, coloquialmente designados por *"router"*, onde um único aparelho contém todas estas funcionalidades.

2.2.1 *Gateways*

Numa rede, existe um nó que opera como o ponto de ligação da rede interna com outras redes e tradutor entre diferentes protocolos. Esse nó é denominado *gateway*. Uma *gateway* pode desempenhar várias funções para além da sua função primária. Encaminhamento e comutação dos pacotes de dentro para fora da rede e vice-versa são funcionalidades tipicamente desempenhadas por uma *gateway*.

Tendo em conta o equipamento disponibilizado pelos ISP, pode afirmar-se que este desempenha a função de *gateway*. Este dispositivo desempenha as inúmeras funções desempenhadas pela *gateway* e funções acrescidas como tradução de nomes, ponto de acesso sem fios, antepara de segurança, provedor de endereços IP para os dispositivos

internos e, mais recentemente, provedor de sinal de televisão. A *gateway* de uma rede é, regularmente, o encaminhador principal da rede, visto ser o ponto de entrada e saída da rede.

Em termos de mercado, a funcionalidade das *gateways* é introduzida em dispositivos destinados a encaminhamento. A Cisco e a TP-Link produzem vastas opções de encaminhadores, destinados a um conjunto amplo de consumidores finais, desde encaminhadores domésticos a encaminhadores destinados a centros de dados de grande escala. No entanto, estes equipamentos operam com um sistema operativo proprietário, introduzindo um custo acrescido associado e restringindo as opções de *software* às já embebidas no equipamento.

Por outro lado, existem equipamentos que operam com sistemas de código aberto. Apesar de ser um sistema operativo de código aberto, o Linux OpenWrt é um sistema especializado para dispositivos embebidos e com funções de encaminhamento. Exemplificando este ponto, o Turris Omnia[6] é um encaminhador destinado a efetuar funções de *gateway* e ponto de acesso sem fios, entre outras. A Mikrotik[12] também fabrica dispositivos destinados a desempenhar funcionalidades de encaminhador e *gateway*, no entanto, o sistema operativo utilizado neste equipamento é uma implementação do OpenWrt, denominada RouterOS, desenvolvida pela própria Mikrotik, ou seja, é um sistema operativo proprietário desenhado com o objetivo de desempenhar estas funções.

Foram efetuados esforços no sentido de encontrar uma solução de mercado, de custo moderado, que tirasse partido da versatilidade de um sistema operativo Linux. O Turris Omnia é a solução mais próxima encontrada, possui inúmeras funcionalidades das desejadas e outras funcionalidades adicionais. No entanto, o custo da solução é elevado e como o seu sistema operativo é desenhado especificamente para o dispositivo, é necessário contratar um perito para efetuar a customização do mesmo, acrescentando complexidade à utilização deste dispositivo. À luz do conhecimento atual, não existem soluções comercializadas de custo moderado e customizáveis que integrem um sistema operativo de propósito geral, em código aberto, num equipamento que desempenhe as funções de *gateway* e encaminhador, entre outras.

2.2.2 Controladores de Pontos de Acesso

Neste trabalho, designa-se controlador de rede como uma combinação de *hardware* e *software* que centraliza a gestão de uma rede local, que possui funcionalidades de encaminhamento e autenticação, bem como gestão de redes privadas virtuais ou VPN (*"Virtual Private Network"*) e gestão de pontos de acesso.

Um exemplo mais específico de um controlador de rede, é um controlador de pontos de acesso sem fios. Estes controladores têm como objetivo mover toda a gestão e controlo dos pontos de acesso sem fios das redes locais para um só dispositivo, centralizando e facilitando a gestão. Ao passar a gestão dos pontos de acesso para o controlador, podem introduzir-se funcionalidades mais complexas que surgem da cooperação e coexistência de

múltiplos pontos de acesso. Um exemplo é o *"access point roaming"*. Esta funcionalidade consiste na transferência da associação dos dispositivos móveis de um ponto de acesso para outro sem que seja necessária alteração de endereço e consequente interrupção das ligações já estabelecidas pelo dispositivo. Empresas como a Cisco e a TP-Link, conhecidas pelo seu peso e influência no mundo dos equipamentos de rede, para além de encaminhadores (*routers*), dispositivos de comutação (*switches*) e pontos de acesso sem fios, produzem também controladores de pontos de acesso. Sendo dispositivos proprietários, todo o seu *software* e protocolos pertencem à empresa em questão e são utilizados em conjunto com pontos de acesso do mesmo fabricante que o controlador. Para além de estes dois exemplos, a Mikrotik produz encaminhadores que em conjunto com *software* podem assumir funcionalidades de gestão de pontos de acesso. Tendo como exemplo um controlador da Cisco, este dispositivo tem capacidade para atingir taxas de transferência de 40 Gigabit/s, conseguem gerir até 6000 pontos de acesso e 64000 clientes, e possuem portas *ethernet* capazes de comunicar a velocidades de 10 Gigabits/s [3]. Em termos de funcionalidades, estes controladores efetuam autenticação, permitem *"access point roaming"* e são capazes de configurar a força de sinal e os canais em que os pontos de acesso transmitem, com o intuito de evitar sobreposições de sinal.

O protocolo *"Control And Provisioning of Wireless Access Points"* (CAPWAP) é um standard que permite a gestão centralizada de pontos de acesso sem fios. Este protocolo utiliza UDP, com encriptação baseada em *"Datagram Transport Layer Security"* (DTLS), na comunicação entre o controlador e os pontos de acesso. O CAPWAP define a comunicação entre o ponto de acesso e o controlador, bem como processos de entrada na rede por parte de pontos de acesso onde, ao iniciar o seu funcionamento, o ponto de acesso envia uma mensagem de descoberta, que será respondida por qualquer controlador da rede que a receba, sendo posteriormente integrado na rede em questão. Este protocolo é compatível com os padrões de redes sem fios (IEEE 802.11) e tem uma implementação em código aberto denominada OpenCAPWAP [1].

2.3 Enquadramento da Solução

Apesar de existirem diversas soluções para implementar uma rede local, consideremos apenas os extremos opostos do espectro de soluções. A figura 2.2 ilustra, por um lado, uma solução onde os equipamentos da rede desempenham apenas uma função, por outro, uma solução onde apenas um único dispositivo desempenha todas as funções, à semelhança das soluções implementadas e fornecidas pelos ISP.

Este projeto tem como objetivo desenvolver uma solução que desempenhe variadas funções. Para melhor enquadramento do projeto é necessário aprofundar a razão pela qual não é possível utilizar um equipamento fornecido pelo ISP. A configuração e personalização que a Ângulo Sólido pretende garantir não é compatível com um equipamento procedente

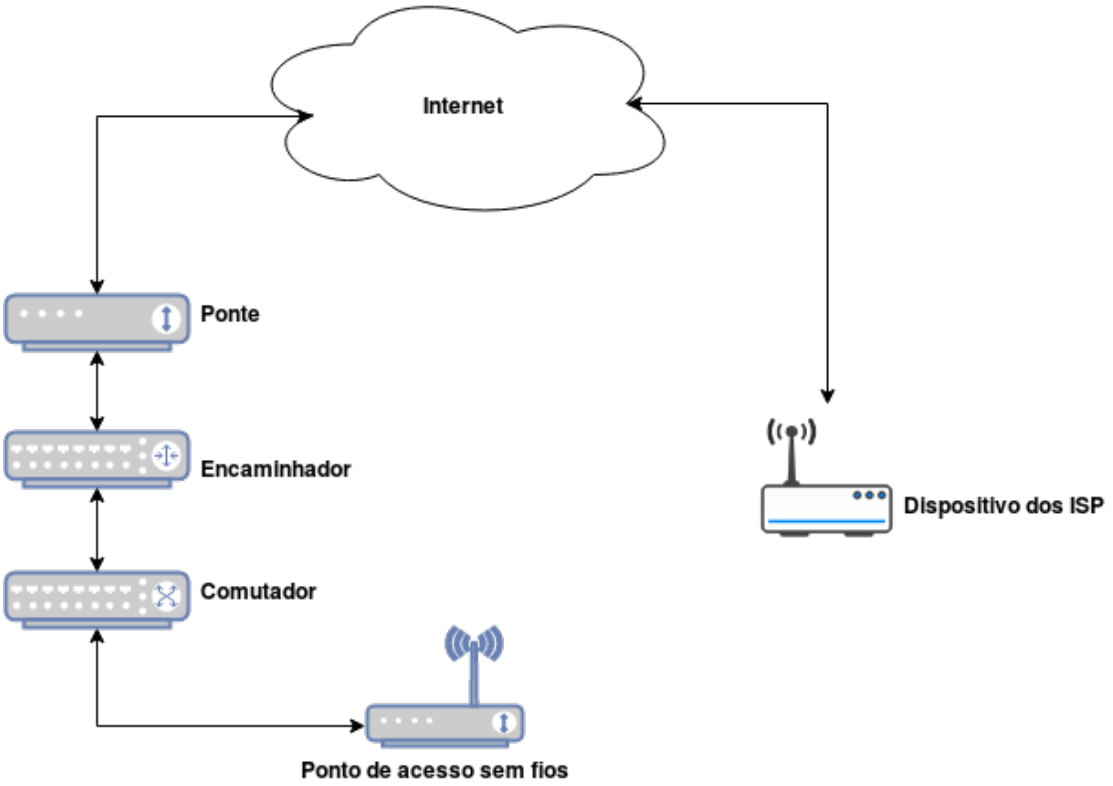

Figura 2.2: Solução diferenciada para rede local (esquerda) e solução dos ISP (direita)

de um ISP, uma vez que estes equipamentos são configurados pelo proprietário. Por exemplo, não é possível configurar as portas de rede nestes equipamentos e também não é possível instalar *software*. Para além das limitações na configuração do equipamento, visto que o seu proprietário é o ISP, há certos aspetos que a Ângulo Sólido não consegue garantir ao cliente. Por exemplo, qualidade de serviço personalizada para o cliente, manutenção do equipamento e substituição em caso de avaria, são pontos que são garantidos por parte do proprietário do equipamento. Ao desenvolver uma solução própria, a instituição acolhedora pode dar essas garantias. Noutra perspectiva, se fossem utilizados equipamentos provenientes de ISP, estaríamos a lidar com pelo menos um equipamento diferente por cada ISP existente e, talvez, diversos modelos em cada ISP. A gestão de equipamentos diferentes torna-se complexa. Seria necessário configuração individual por cada dispositivo diferente, reduzindo a escalabilidade da solução, em número de clientes, e aumentando os tempos de intervenção. Ao possuir uma solução própria, a instituição consegue uniformizar todos os equipamentos, ou seja, independentemente do cliente ou do ISP contratado pelo cliente, o equipamento instalado pela Ângulo Sólido é sempre idêntico, permitindo facilitar a sua instalação e configuração.

Capítulo 3

Controlador de Redes

As diversas etapas do processo da desenvolvimento de um controlador de redes passam pela escolha do *hardware* e do sistema operativo, instalação e configuração de serviços e pacotes de *software*, a implementação de funcionalidades, realização testes de validação e desempenho e correções de problemas.

Numa primeira fase, será necessário procurar e escolher *hardware* que satisfaça da melhor forma as necessidades do produto. Escolhido o *hardware*, é necessário decidir que sistema operativo e que conjunto de pacotes de *software* devem ser instalados para garantir os serviços necessários para o funcionamento do produto. Segue-se a implementação dos procedimentos automáticos de instalação e configuração do *software* necessário, a partir de uma instalação base.

Por fim, serão efetuados testes com diversas configurações possíveis e instalações em ambientes reais.

3.1 Requisitos

De modo a desenvolver um controlador de redes em código aberto, é necessário combinar *hardware*, capaz de suportar as funcionalidades pretendidas, com um sistema operativo que seja compatível, instalando e configurando os pacotes de *software* necessários. Posteriormente, é necessário efetuar a instalação e configuração dos pacotes de *software* necessários. É imperativo que o *hardware* tenha diversas portas de rede capazes de suportar velocidades de pelo menos 1 Gigabit/s para que seja possível garantir uma taxa de transmissão alinhada com as expectativas atuais (R1). Tendo em conta que o cliente alvo poderá não possuir instalações devidas para acomodar este tipo de equipamento, o armazenamento não deve conter partes móveis, pois estas desgastam-se mais facilmente e são mais susceptíveis ao impacto (R2). A componente *hardware* deve suportar Linux, pois o controlador será baseado numa implementação deste sistema operativo (R3).

É necessário definir e configurar as redes locais e acesso à WAN (*Wide Area Network*) (R4). O controlador tem de providenciar acesso à *Internet*, para tal será configurado NAT

(*Network Address Translation*) (R5), resolução de nomes (R6), através de um servidor DNS (*"Domain Name System"*), e encaminhamento. Deverá existir isolamento entre as diferentes redes locais, não sendo permitida comunicação entre duas redes LAN. O isolamento entre as redes locais e a implementação de segurança nas mesmas deverão ser atingidos através da configuração de regras da antepara de segurança (*firewall*). O controlador deve também possibilitar configurações com redundância de ligações WAN (R8), de modo a tolerar faltas de ligação à *Internet*.

A rede local deve ser capaz de acomodar dispositivos móveis e dispositivos sem endereço IP estático (R9). Tendo este ponto em conta, será configurado um servidor DHCP (*"Dynamic Host Configuration Protocol*), permitindo aos utilizadores usufruírem de dispositivos móveis sem necessidade de configurar manualmente os mesmos.

Requere-se que o controlador tenha capacidade para autenticar utilizadores de rede (R10), para tal será configurado um mecanismo de interligação com sistemas de autenticação, em particular LDAP (*"Lightweight Directory Access Protocol"*). Os utilizadores deverão ser capazes de aceder a serviços internos à rede local, mesmo estando geograficamente distantes da mesma, para tal será configurado um servidor de VPN (*"Virtual Private Network"*) (R11).

Em prol de providenciar aos utilizadores uma boa experiência de utilização da *Internet*, deve ser configurado um mecanismo de qualidade de serviço (QoS) com o intuito de impedir que transferências de dados afetem a navegação na *Internet*, garantindo que o acesso a máquinas exteriores tenha uma latência inferior a 100ms (R12). Devem também ser garantidas taxas de transferência não inferiores a 80% da largura de banda máxima contratada ao ISP (R13).

Os procedimentos de instalação e configuração devem ser automatizados para reduzir a complexidade e o tempo de configuração (R14). A infraestrutura de automação dos procedimentos deverá ser alimentada por um ficheiro de variáveis personalizado à medida do cliente.

O conjunto de requisitos é resumido na tabela 3.1.

Requisito	Descrição
R1	O *hardware* deve ter pelo menos 4 portas de rede capazes de suportar pelo menos 1 Gigabit/s
R2	O armazenamento não pode conter partes móveis
R3	O *hardware* deve suportar Linux
R4	O controlador deve providenciar acesso à *Internet*
R5	Tradução de nomes
R6	Resolução de nomes
R7	Isolamento entre redes locais
R8	O controlador deve possibilitar redundância de ligações WAN para tolerância a falta de ligação à *Internet*
R9	Capacidade de alojamento de dispositivos móveis
R10	Autenticação de utilizadores através de sistemas de autenticação externos (LDAP)
R11	Disponibilização de um serviço VPN
R12	Garantir acesso a máquinas externas com latência abaixo de 100ms
R13	Garantir taxas instantâneas de transferência não inferiores a 80% da largura máxima de banda contratada ao ISP
R14	Os procedimentos de instalação e configuração devem ser automáticos e alimentados por um ficheiro com variáveis de configuração

Tabela 3.1: Tabela de Requisitos

3.2 Arquitetura

Como já foi referido, o controlador irá desempenhar diversas funcionalidades na rede onde será colocado. A figura 3.1 ilustra uma representação lógica dos componentes do controlador e a forma como se integra com outros componentes da rede.

O controlador poderá ser configurado de diversas formas, permitindo diferentes conjugações de ligações a redes internas e externas. A configuração mínima obrigatória consiste em apenas 1 ligação WAN, 1 LAN e 1 WLAN. São permitidas configurações com redundância de ligações WAN, garantindo tolerância a uma falta de ligação à *Internet*. A solução permite também configurações com duas redes LAN, uma para serviços internos e a segunda rede para dispositivos móveis convidados, acesso VPN ou para a implementação de uma zona desmilitarizada. A rede WLAN hospedada no controlador tem o intuito de dar acesso ao serviço VPN sem que seja necessária uma ligação física a uma rede. Esta rede existe é utilizada pela da equipa da Instituição Acolhedora para fins de gestão e manutenção.

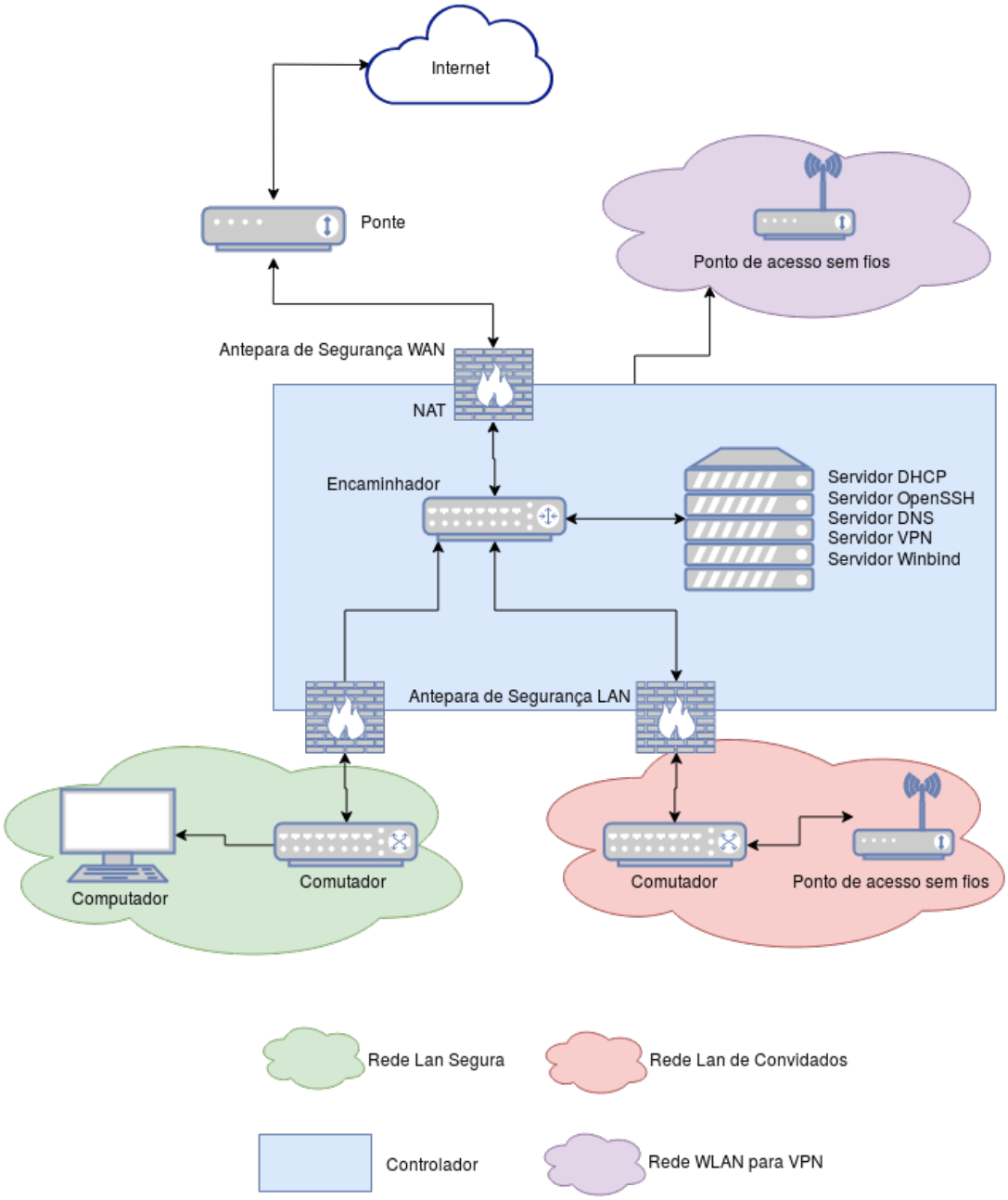

Figura 3.1: Arquitetura final da solução.

3.2.1 Seleção de *Hardware*

Respeitando os requisitos anteriormente descritos, é desejável um sistema que não tenha partes móveis, o que implica que o armazenamento interno deve ser do tipo SSD (*"Solid State Drive"*) ou CF (*"Compact Flash"*). São necessárias pelo menos quatro portas de rede capazes de suportar comunicação *Ethernet* com velocidade de pelo menos 1 Gigabit/s, duas das quais para ligação WAN de forma a criar redundância física para a ligação exterior, uma porta para LAN e, idealmente, outra para uma rede dedicada de administração. Este sistema tem de conter uma interface de rede sem fios, ter uma caixa

robusta de metal e ter boa resistência a mudanças de temperatura e à humidade, devido à possibilidade da inexistência de uma sala devidamente preparada para alojar este tipo de equipamentos, e deve ser compatível com um sistema operativo Linux.

A primeira fase de procura passa por encontrar placas que possuam no mínimo quatro portas de rede físicas, capazes de suportar *Ethernet* com velocidade de pelo menos 1 Gigabit/s, interface de rede sem fios e compatíveis com Linux, sendo estas características eliminatórias à partida. Foram também consideradas placas com três portas de rede físicas, existindo a possibilidade da utilização de um adaptador externo para obter a quarta porta de rede. Ao longo deste documento o conjunto de uma placa com três portas de rede físicas e um adaptador *Ethernet* externo é considerado uma placa de rede com quatro portas de rede físicas. Tendo em conta os pontos referidos, surge uma lista de seis placas candidatas:

- AAEON FWB-2250
- AAEON FWB-7250
- Soekris net5501
- Soekris net6501
- BananaPi BPI-R2
- PC-Engines APU (+ adaptador *Ethernet* externo)

Em termos de armazenamento interno é necessário ter em consideração o tipo de armazenamento que a placa permite. As placas a cima têm conectores *Compact Flash*, microSD e/ou portas SATA e mSATA, permitindo por isso integrar armazenamento do tipo SSD. Todas satisfazem, no que toca ao armazenamento, os requisitos referidos anteriormente. No que respeita a memória RAM, a placa com menos memória contém 512MB, considerados suficientes ainda assim, pelo que nesta fase de avaliação não houve nenhuma exclusão óbvia.

Apesar das placas escolhidas terem todas quatro portas de rede físicas, verifica-se que as placas AAEON FWB-2250, AAEON FWB-7250 e BananaPi BPI-R2 apenas possuem duas interfaces de rede independentes, sendo as restantes utilizadas para comutação. Na sequência desta análise restam as placas Soekris net5501, Soekris net6501 e PC-Engines APU. A tabela 3.2 faz uma comparação sumária das mesmas.

A placa Soekris net6501 é uma versão mais recente da Soekris net5501, contendo tecnologias diferentes e tem melhor desempenho. O custo adicional da Soekris net6501 face ao melhoramento do desempenho do *hardware* não é considerado excessivo, logo a Soekris net5501 é excluída das opções. Tendo em conta estas considerações, chega-se a duas soluções possíveis sendo que uma se afirma melhor em termos de componentes. As

Nome	**Portas de Rede**	**Processador**	**RAM**	**Armazenamento**
net5501	4	500 Mhz AMD Geode LX	521MB	Compact Flash
net6501	4	1.6 Ghz Intel Atom E6xx	2GB	USB or mSata
APU 3b2	3 + 1	AMD Embedded GX-412TC 1GHz	2GB	SD Card

Tabela 3.2: Placas candidatas a integrar o sistema

duas combinações em avaliação, incluindo já todos os componentes adicionais necessários são:

- Soekris net6501 + Caixa de metal
- Kingston mSata 60GB SSD
- Carta wireless Atheros AR9280 MiniPCIe
- Antena + Pigtail
- Fonte de alimentação IEC320-C8 para net6501

e:

- PC-Engines APU 3b2 + Caixa de metal
- SD Card 4GB
- Carta wireless mini PCI-E
- Antena + Pigtail
- Fonte de alimentação 12V
- Adaptador externo USB-*Ethernet*

A tabela 3.3 mostra que o custo das duas soluções difere em quase 100%:

Optou-se pela solução baseada na placa APU uma vez que o custo é mais baixo e as suas características cumprem os requisitos de *hardware* e são mais que suficientes para o desempenho das tarefas, satisfazendo os requisitos R1, R2 e R3.

Solução	Custo
Soekris	$\sim$ 505€
APU	$\sim$ 215€

Tabela 3.3: Custo das soluções propostas

3.2.2 Sistema Operativo

O Sistema Operativo é o *software* base sobre a qual serão implementadas todas as funcionalidades do controlador, tornando a sua seleção um passo importante do projeto. Procura-se uma distribuição Linux, que seja compatível com o *hardware* escolhido de forma a garantir um nível de estabilidade desejado em ambiente empresarial e que possa desempenhar as funcionalidades a implementar. É essencial escolher uma distribuição com suporte a longo prazo [10, 14], pois o esforço associado à substituição do sistema operativo é elevado e dispendioso. O suporte de longo prazo reduz essa a necessidade de substituição. Para tal foram consideradas duas candidatas que cumprem estes requisitos: CentOS e Ubuntu LTS. Estas distribuições têm suporte de 7 e 5 anos, respetivamente, garantindo assim o suporte a longo prazo pretendido. Quanto às especificações, ambas têm versões de 64bit, sendo compatíveis com o processador da placa em questão. Em termos de RAM e armazenamento, a solução de *hardware* cobre as exigências das duas distribuições [11, 14].

As distribuições CentOS e Ubuntu derivam de distribuições base diferentes, respetivanente RedHat e Debian. Isto significa que alguns componentes diferem entre estes sistemas. Por exemplo, a definição e configuração das interfaces de rede é feita de forma diferente nestas duas distribuições. Mesmo que algumas diferenças não tenham impacto direto sobre a forma como o sistema opera, é necessário ter em consideração estas nuances a fim de escolher a distribuição mais favorável para o projeto. A instituição de acolhimento, Ângulo Sólido, gere um número elevado de sistemas baseados em Ubuntu pelo que tem experiência relativamente a esta distribuição. Ao escolher esta distribuição está a caminhar-se no sentido da uniformização dos sistemas geridos pela Ângulo Sólido, facilitando a manutenção e gestão desses sistemas. Tendo em conta os pontos de avaliação anteriormente referidos, a distribuição Ubuntu revela ser a mais vantajosa para o projeto.

3.3 Concretização

Nas secções seguintes serão apresentados em detalhe os procedimentos utilizados para a concretização das funcionalidades introduzidas na solução.

3.3.1 Redes

A estrutura de uma rede local simples com acesso exterior implica, no mínimo, que o controlador de rede possua duas interfaces de rede: uma para ligação LAN (*"Local Area Network"*) e outra para ligação WAN (*"Wide Area Network"*). Mantendo o requisito mínimo de 1 interface WAN e 1 interface LAN e considerando a existência também uma rede WLAN (*"Wireless Local Area Network"*), pretende-se que o controlador desenvolvido suporte as seguintes combinações de redes:

- 1 WAN + 1 LAN + 1 WLAN
- 1 WAN + 2 LAN + 1 WLAN
- 1 WAN + 3 LAN + 1 WLAN
- 2 WAN + 1 LAN + 1 WLAN
- 2 WAN + 2 LAN + 1 WLAN

De notar que as configurações apresentadas não consideram a utilização de pontos de acesso externos ao controlador, que serão adicionados apenas na fase de configuração. As diferentes configurações apresentadas têm em consideração cenários reais.

O objetivo de suportar diferentes configurações recai sobre a heterogeneidade das necessidades das instituições onde o controlador será implementado. É necessário suportar tanto uma configuração de rede simples, como cenários mais complexos. O controlador pode ser instalado em instituições com sistemas críticos em que não podem perder acesso à *Internet*. A configuração com dupla ligação WAN está destinada a suportar essa necessidade. No caso de uma instituição ter máquinas acessíveis a partir da *Internet*, é possível configurar uma das redes LAN como DMZ (*"Demilitarized Zone"*), isolando assim outras redes internas de possíveis ameaças.

A configuração base apresentada na Listagem 3.1 contém o mínimo necessário para proceder ao processo de instalação, estando definidas apenas duas interfaces de rede, uma WAN, utilizando DHCP por omissão, e uma interface destinada a LAN, com endereço IP estático para efetuar acessos por ssh ao controlador. Durante o processo de instalação, todas as interfaces do controlador serão reconfiguradas de acordo com as necessidades do cliente em questão.

Considera-se que a interface WLAN, incorporada no próprio controlador, está destinada primariamente para ligação utilizando VPN.

Listagem 3.1: Ficheiro das interfaces de rede na imagem base

```
# This file describes the network interfaces available on
   your system
# and how to activate them. For more information, see
   interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The LAN interface
auto enp1s0
iface enp1s0 inet static
        address 192.168.5.100
        network 192.168.5.0
        netmask 255.255.255.0
        broadcast 192.168.5.255

# The primary WAN interface
auto enp2s0
iface enp2s0 inet dhcp
```

A utilização do adaptador USB-Ethernet à solução poderia ser problemática em termos de automatização, no que toca à previsibilidade na denominação das interfaces. No entanto, atendendo à implementação do *kernel* Linux, as interfaces de rede possuem nomes previsíveis. O nome atribuído às interfaces será sempre o mesmo, por exemplo, as interfaces de rede dos adaptadores serão sempre denominadas obedecendo ao modelo "enx<MAC-ADDRESS>", permitindo a implementação de uma solução automática.

3.3.2 NAT e Encaminhamento

Pretende-se que as redes LAN do cliente tenham acesso à *Internet*, para tal é necessário implementar NAT no controlador e permitir encaminhamento entre interfaces de rede, nomeadamente, ativar o encaminhamento IPv4 e definir regras na antepara de segurança que permitam esse encaminhamento.

São definidas duas tabelas de encaminhamento quando o controlador é configurado com duas ligações WAN, *wan_table* e *wan2_table*. Se o controlador apenas estiver configurado com uma ligação WAN apenas a tabela *wan_table* é definida. A tabela *wan_table* trata do encaminhamento principal, tento como objetivo encaminhar todo o tráfego entre as interfaces LAN e a WAN principal. A tabela *wan2_table* serve para fazer o encaminhamento

em caso de falta na interface de WAN primária e para albergar tráfego vindo de servidores específicos, definidos pelo cliente.

É possível que o cliente possua serviços disponibilizados para o público, ou seja, que sejam acessíveis a partir da *Internet*. Contemplando as possíveis necessidades dos clientes, a solução suporta encaminhamento de ligações exteriores, a partir de determinados portos, para servidores internos, permitindo assim a prestação desses mesmos serviços. Para proceder à execução deste tipo de encaminhamento, foi implementado um mecanismo que facilita a aplicação das regras necessárias. A solução disponibiliza uma infraestrutura onde um ficheiro, que contém a informação necessária para execução das diferentes regras de encaminhamento, alimenta uma função que executa essas regras.

As redes LAN são isoladas, não sendo possível que computadores alojados em redes locais diferentes comuniquem entre si, existindo por vezes exceções. Tendo em consideração essas situações, foi implementada uma infraestrutura similar à referida anteriormente para introduzir regras de encaminhamento entre redes locais, suportando essas exceções.

Tolerância a faltas de ligação à *Internet*

A redundância resultante da configuração com duas ligações exteriores permite garantir tolerância à falta de uma das ligações exteriores. A solução apresentada possui um mecanismo de monitorização, deteção e reação a faltas de *Internet* numa das ligações.

Assumindo que um cliente possui um controlador configurado com duas ligações WAN, primária e secundária, colocam-se os seguintes cenários. Ao ser detetada falta de ligação à *Internet* na ligação secundária, o mecanismo redireciona todo o tráfego anteriormente encaminhado por essa ligação para a ligação primária e notifica o sistema de monitorização da instituição acolhedora. Se for detetada falta de ligação à *Internet* na ligação primária, o mecanismo redireciona todo o tráfego dessa ligação para a ligação secundária, mais uma vez, notificando o sistema de monitorização.

3.3.3 Antepara de Segurança (*Firewall*)

A segurança é um ponto incontornável na informática. Devido à natureza da solução desenvolvida no âmbito deste projeto, é necessário ter em atenção a instalação e a configuração de uma antepara de segurança no controlador. O *software* **iptables** foi escolhido para efetuar essa implementação.

Todos os equipamentos que se encontram ligados à *Internet* podem ser alvos de ataques informáticos. Tendo esta premissa em conta, é necessário tomar as precauções devidas para que cada máquina seja o menos vulnerável possível. O ponto de entrada de uma rede é o primeiro equipamento suscetível a ataques. Considerando que a solução desenvolvida irá operar precisamente como o ponto de ligação entre a rede interna de um cliente e a *Internet* a sua antepara de segurança deve ser cuidadosamente configurada.

Em primeiro lugar é necessário ter em conta as políticas da antepara de segurança. Estas políticas vão ditar o procedimento por omissão de qualquer pacote que chegue ao dispositivo. Na instalação base do **iptables**, as políticas estão configuradas por omissão como *"ACCEPT"*. Isto significa que todo o tráfego que entre ou saia do equipamento irá ser aceite, não fazendo qualquer tipo de verificação ou diferenciação dos pacotes. Ter todas as políticas de uma antepara de segurança como *"ACCEPT"* denomina-se políticas abertas. A utilização de políticas abertas é uma falha de segurança, pois qualquer acesso exterior benigno ou malicioso é permitido. Considerando este ponto, o primeiro passo na configuração da antepara de segurança da solução é mudar as políticas das tabelas de *INPUT* e *FORWARD* para *"DROP"*. Com esta configuração todos os pacotes serão descartados sem qualquer indicação para o emissor. Isto causa um isolamento do sistema, uma vez que nenhum pacote irá ser aceite pelo equipamento.

A política da tabela OUTPUT deve ser aberta para que seja permitido que o tráfego gerado dentro da rede possa ter acesso exterior. Deve também permitir-se todo o tráfego de fora para dentro que seja relacionado com ligações efetuadas a partir de máquinas internas ou de ligações já estabelecidas. De seguida, deve ativar-se os procedimentos de NAT na interface de saída do controlador. A partir deste momento, juntamente com o encaminhamento ativo e bem definido, qualquer dispositivo dentro da rede que possua um endereço IP e que tenha ligação ao controlador terá acesso ao exterior.

De um modo genérico, todo o tráfego vindo do exterior de forma espontânea, ou seja, tráfego que não tenha sido solicitado por nenhum dispositivo interno à rede, deve ser descartado. No entanto, existem exceções que devem ser consideradas, por exemplo acesso remoto para manutenção ou *pings* para confirmar que a máquina se encontra ativa. A antepara de segurança irá permitir pacotes *ping* do protocolo ICMP ou *Internet Control Message Protocol* e ligações ao porto do servidor ssh da solução. Devido à existência de um servidor de OpenVPN na solução, também devem ser permitidas ligações no porto de escuta deste servidor.

Estas alterações são transversais às interfaces de rede do equipamento. No entanto, há regras que se pretende que apenas afetem a configuração das interfaces de LAN. Pedidos de DHCP devem ser aceites nas interfaces LAN para que possam ser atribuídos endereços IP de forma automática.

Por fim, é necessário adicionar as regras necessárias para satisfazer as necessidades de encaminhamento entre as diferentes interfaces do controlador e entre diferentes redes, como referido na secção 3.3.2, concluindo assim a configuração da antepara de segurança do controlador.

3.3.4 *QoS - Quality of Service*

Introdução ao problema

O tráfego existente na *Internet* é predominantemente gerado por ligações TCP [18], sendo este o tipo de tráfego que mais contribui para situações de congestão nas redes.

Quando é efetuada uma transferência TCP num dos sentidos de uma ligação, os pacotes de *acknowledgement* ocupam uma pequena parte da largura de banda disponível na direção contrária. Quando na direção contrária são iniciadas outras transferências, os pacotes dessas transferências podem ocupar a restante largura de banda existente, causando latência superior ao normal que afeta os pacotes de *acknowledgement*. Este fenómeno é agravado quanto mais o tráfego no retorno se aproximar da capacidade da ligação. Num caso extremo de assimetria, os pacotes de *acknowledgement* das transferências no sentido com maior largura de banda, ocupariam totalmente a ligação no sentido contrário à transferência. Esta situação faz com que a rede seja praticamente inutilizável por parte de aplicações sensíveis a tempo, por exemplo, sessões ssh e chamadas de voz e vídeo.

Em geral, a saturação de qualquer um dos sentidos de tráfego provoca constrangimentos sérios na utilização dos acessos à *Internet*, sendo que em função das larguras de banda de cada um e o padrão de uso dos sistemas ligados ao acesso, um dos sentidos terá maior probabilidade de estar saturado.

Para ilustrar o problema, suponha-se que num determinado instante, numa determinada rede em que a largura de banda de *downstream* é substancialmente superior à largura de banda de *upstream* estão a decorrer, em simultâneo, diversos *downloads* e que estes utilizam na totalidade a largura de banda de entrada disponível, ou seja, ocorre saturação do *downstream*. A saturação do *downstream* provoca uma ligeira latência na rede sem afetar a sua utilização normal de forma drástica. No entanto, se num determinado instante, simultaneamente aos *downloads*, ocorrerem diversos *uploads* concorrentes que saturam o *upstream* ocorre um impacto significativo na rede, provocando uma grande redução na taxa instantânea de *download* e uma latência muito superior à latência da rede quando se encontra em utilização normal.

Este comportamento foi verificado em diferentes dias e em alturas diferentes do dia na plataforma de testes usada durante o trabalho.

Análise do problema

Para análise do problema, foram efetuados testes numa ligação com 55 Mbit/s de *download* e 5 Mbit/s de *upload*. Nestes testes são considerados quatro cenários:

- **Repouso**: Este cenário representa o tráfego gerado pela utilização normal da rede (navegação de páginas *web*, ocasional mensagem de conversação) e observa-se nos primeiros 30 segundos e nos últimos 30 segundos dos testes;

- **Saturação da Entrada**: Onde são efetuados, em simultâneo, oito *downloads* de diferentes distribuições Linux. Este cenário observa-se a partir dos primeiros 30 segundos dos testes durante 120 segundos;
- **Saturação da Saída**: Onde ocorrem, em simultâneo, quatro *uploads* de ficheiros de diferentes dimensões, para serviços de armazenamento na *cloud*. Este cenário pode observar-se a partir dos 90 segundos de teste e tem uma duração de 120 segundos;
- **Saturação Total**: Este cenário é atingido no espaço de tempo em que os cenários **Saturação da Entrada** e **Saturação da Saída** se sobrepõem, iniciando-se 90 segundos após o início dos testes e tendo duração de 60 segundos.

Nota: Irão ser apresentados gráficos referentes a estes testes. Neles a origem do eixo das abcissas não identifica o início do teste mas sim o início da leitura.

Ao analisar as figuras 3.2 a 3.5 verifica-se um padrão no comportamento da latência medida efetuando *pings* a angulosolido.pt. Inicialmente, no cenário de **Repouso**, a latência praticamente não sofre oscilações mantendo-se em cerca de 3ms. Ao entrar no cenário **Saturação da Entrada**, o preenchimento da largura de banda provoca uma ligeira subida nos valores da latência, que passa a oscilar entre os 10ms e os 20ms. Quando é atingido o cenário **Saturação Total**, verifica-se um aumento de cerca de 100 vezes na latência, atingindo valores superiores a 600ms e tendo sido observados picos esporádicos na ordem de 1000ms. Neste cenário, verifica-se também uma queda de cerca de 40% na largura de banda instantânea ocupada pelos *downloads*.

A perda na taxa de transferência de *download* é atribuída à latência adicional sofrida pelos pacotes de *acknowledgement* referentes aos *downloads*, levando a uma reação por parte do protocolo TCP. Esta latência é induzida pela saturação do *upstream* num fenómeno denominado *bufferbloat effect* [4], definido como a latência acrescentada causada pela acumulação excessiva de pacotes resultante de um dimensionamento exagerado do espaço de retenção temporária (*buffer*). O espaço de retenção temporária referido consiste num local onde são armazenados os pacotes que não puderam ser enviados no momento em que chegaram à interface de rede.

Na secção seguinte serão introduzidas e avaliadas soluções já existentes, bem como uma proposta de solução para o caso particular anteriormente demonstrado.

Soluções existentes e formulação de proposta de solução

Em 2005, Jesper Dangaard Brouer propôs, com a sua tese de mestrado [2], uma solução para o problema da saturação do *upstream*. Esta solução consiste no estrangulamento da largura de banda do *downstream* e do *upstream*, para evitar manipulação do tráfego por parte do ISP, redimensionamento do espaço de retenção temporária (*"buffer"*) e aplicação de filas de prioridade que dão preferência a pacotes que sejam sensíveis ao tempo como,

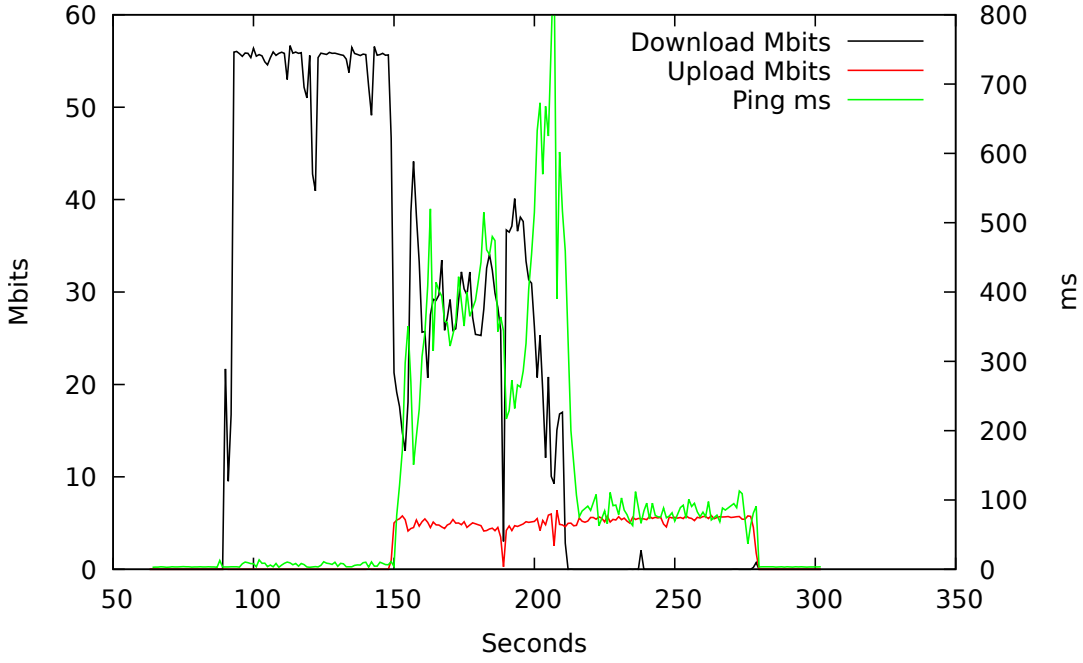

Figura 3.2: Dia 13 de fevereiro de 2018 (hora de almoço)

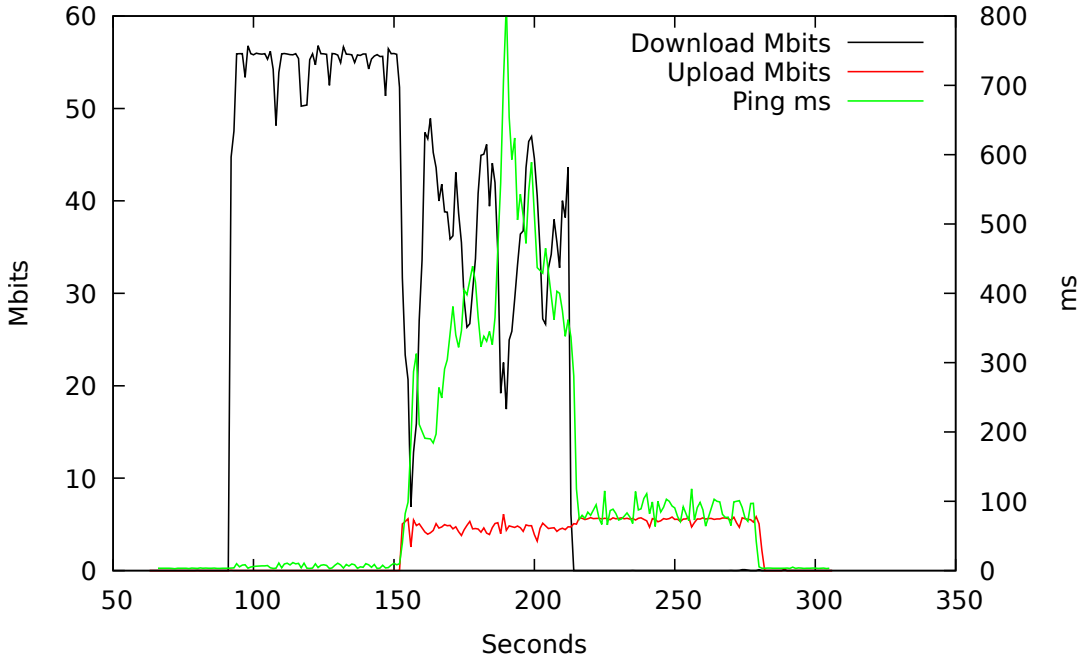

Figura 3.3: Dia 1 de fevereiro de 2018 (meio da tarde)

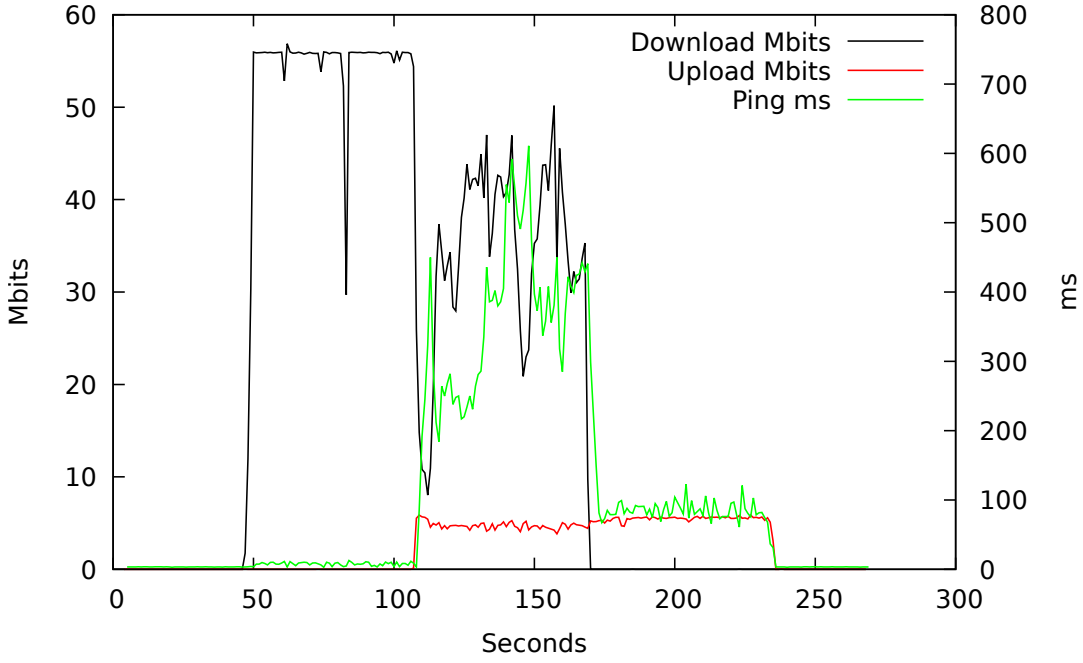

Figura 3.4: Dia 29 de janeiro de 2018 (fim de tarde)

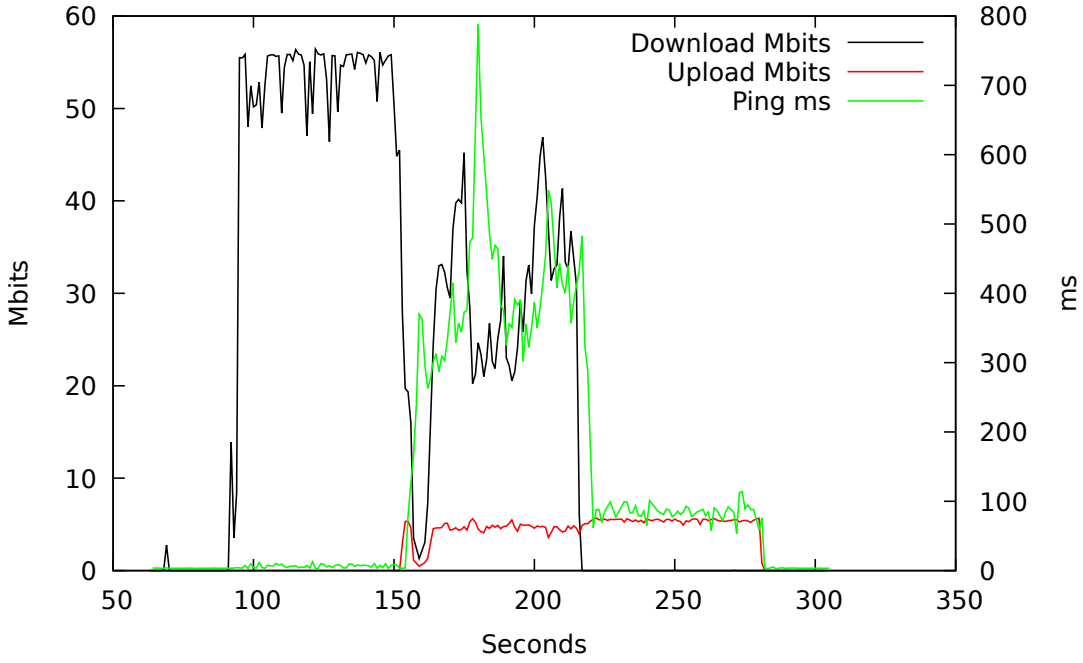

Figura 3.5: Dia 1 de fevereiro de 2018 (meio da manhã)

por exemplo, pacotes de ICMP (*"Internet Control Message Protocol"*) e pacotes de *acknowledgement*.

A instituição de acolhimento, Ângulo Sólido, desenvolveu uma implementação que concretiza o modelo, proposto por Brouer [2], de redução da largura de banda, redução da dimensão do espaço de retenção temporária e utilização de filas de prioridade, obtendo resultados satisfatórios numa ligação de fibra ótica com 55/5 Mbit/s (*download/upload*), como se pode observar na figura 3.6. Utilizando o princípio da redução de largura de banda, as taxas instantâneas utilizáveis de *download* e *upload* são reduzidas em cerca de 20% em relação à largura de banda total disponível. No entanto, o padrão observado anteriormente, o aumento drástico da latência e a grande redução da taxa instantânea de *download*, quando a largura de banda do *upstream* está saturada, não se verifica.

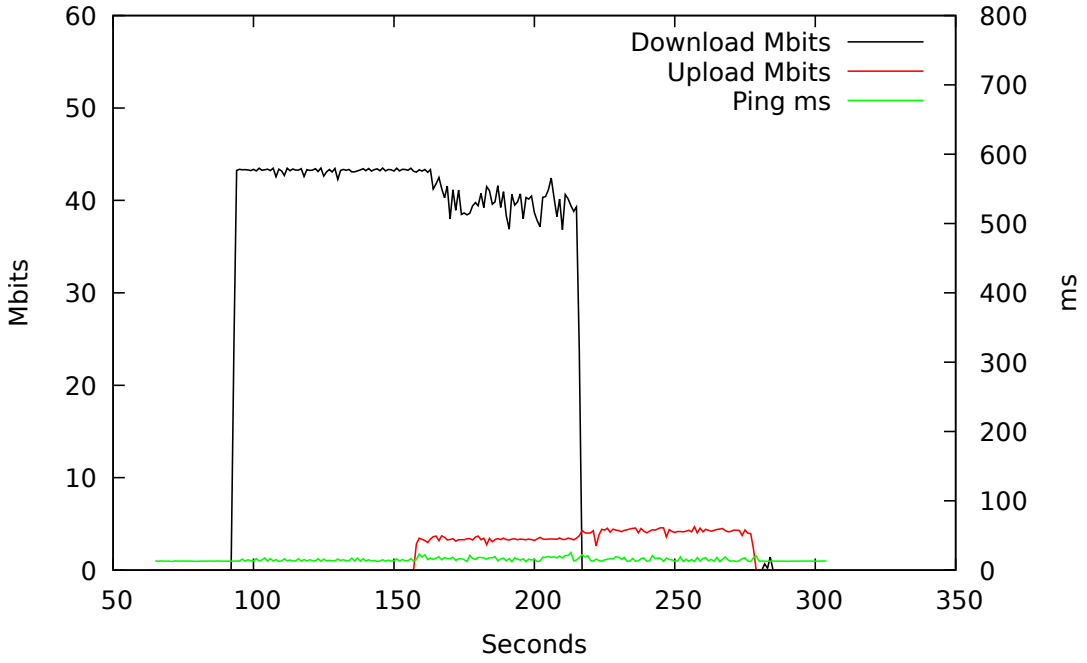

Figura 3.6: Padrão observado utilizando a solução da instituição acolhedora.

Visando o fenómeno do *bufferbloat effect*, considera-se que um dos locais onde este problema se faz sentir com mais intensidade é nas extremidades da rede [17]. Ou seja, é necessário tomar medidas nos extremos para tentar apaziguar o problema do excesso de retenção ao qual chamamos de *bufferbloat*.

Este tópico tem sido bastante discutido na última década, tendo sido criado, inclusive, um sítio dedicado exclusivamente a este problema [4], existindo diferentes soluções para o mesmo. Uma solução ingénua para o problema é a redução do espaço de retenção temporária (*buffer*), visto que uma das causas está no dimensionamento exagerado do mesmo. Na secção seguinte serão demonstrados diversos testes utilizando esta solução.

A utilização de algoritmos de AQM (*"Active Queue Managemment"*) como disciplina de fila no espaço de retenção temporária, contribui para a redução do problema do *buf-*

ferbloat. O princípio de AQM consiste em descartar pacotes de forma inteligente com o intuito de reduzir a congestão no espaço de retenção temporária. Para tal, são utilizados algoritmos desenvolvidos com base neste princípio, como o RED [7](*"Random Early Detection"*) e CoDel [15] (*"Controlled Delay"*: pronunciado "codle").

O algoritmo CoDel apresenta melhorias significativas na latência comparativamente a algoritmos de *"drop tail"*, que descartam pacotes quando o espaço de retenção temporária está cheio. O CoDel apresenta, também, melhorias face ao algoritmo RED [15], justificando a sua utilização no lugar deste algoritmo. Dentro do algoritmo base CoDel, foi desenvolvida uma implementação utilizando princípios de filas justas, designada por sfqCoDel (*"Stocastic Fair Queue CoDel"*). Esta implementação utiliza análise estatística para definir que pacotes devem ser descartados e demonstrou melhorias de desempenho face ao CoDel normal [13].

As figuras 3.7 e 3.8 demonstram a execução do teste apresentado anteriormente mas utilizando como disciplinas de fila o algoritmo CoDel e sfqCoDel, respetivamente. A fim de garantir condições de rede tão semelhantes quanto possível, os testes foram realizados imediatamente um após o outro. Considerado o padrão visível, verifica-se a existência do problema da queda da taxa instantânea de *download*. No entanto, observam-se melhorias ao nível da latência em ambos os casos. Na figura 3.8 podemos verificar uma melhoria da latência, passando de valores médios aproximados de 450ms para 350ms, no momento em que estão a decorrer *downloads* e *uploads* em simultâneo. Podemos então considerar que a utilização de uma disciplina de fila diferente, elegendo um algoritmo direcionado ao problema do *bufferbloat*, apresenta melhorias de desempenho na latência, mas não atinge as taxas de *download* pretendidas.

Tendo em conta a solução apresentada por Brouer [2] e a solução da instituição acolhedora, propõe-se a utilização de uma solução que agregue um algoritmo de AQM e o princípio da utilização de filas com prioridade, colocando-se como hipótese que, com esta combinação, será possível resolver o problema da queda da taxa instantânea de *download* e conter o problema da latência.

Implementação da solução e testes

Uma questão fundamental no desenvolvimento de uma solução que agregue um algoritmo de AQM e filas de prioridade é o dimensionamento do espaço de retenção temporária.

Nota: Os testes apresentados nas figuras 3.2 a 3.5 foram efetuados utilizando um espaço de retenção temporária de tamanho 1000, sendo o valor por omissão do sistema operativo Linux utilizado no desenvolvimento deste projeto.

O objetivo primário dos testes é perceber de que forma a redução do tamanho do espaço de retenção temporária influencia a latência e que efeito tem na taxa instantânea de *download*. Os testes com diferentes dimensionamentos do espaço de retenção temporária foram efetuados com largura de banda de 4 Mbit/s de *upload*, artificialmente limitada, e

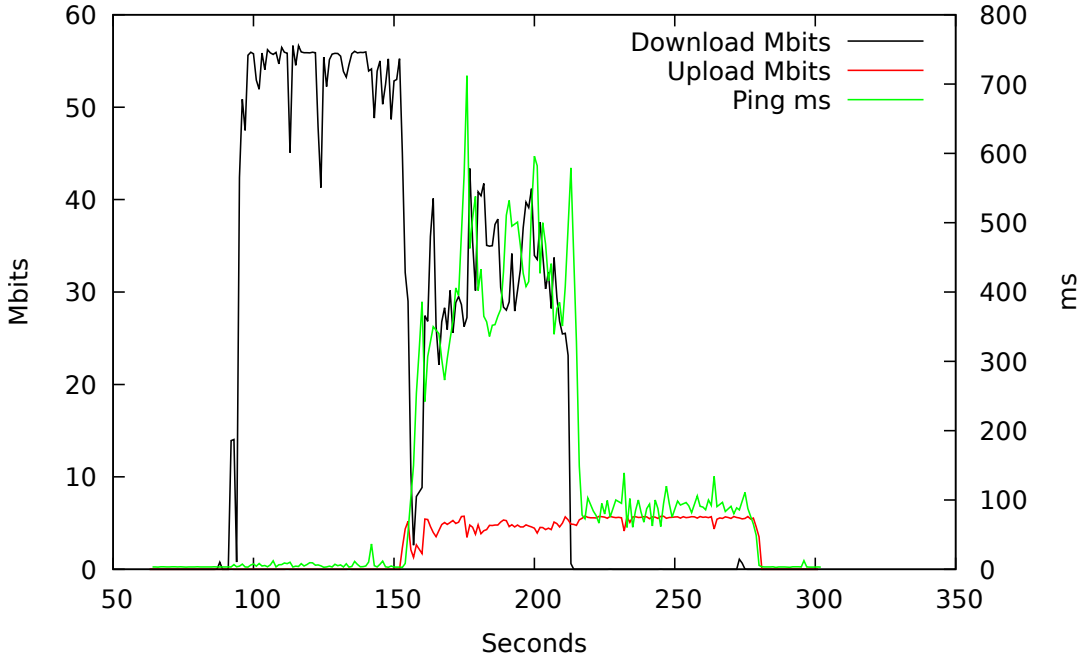

Figura 3.7: Padrão observado utilizando o algoritmo CoDel como disciplina de fila.

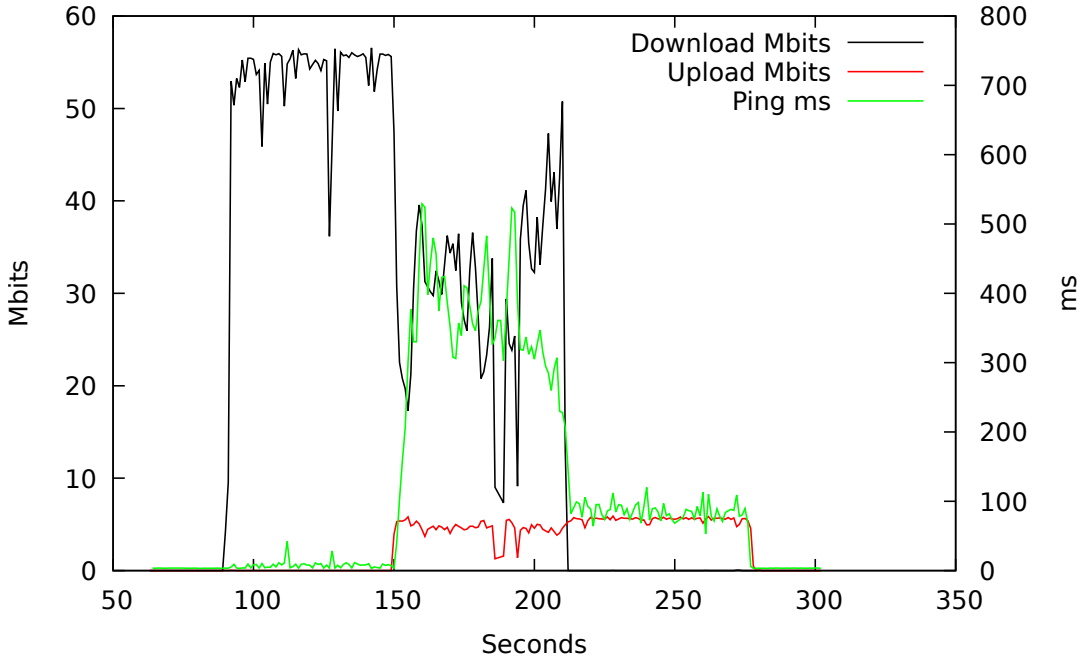

Figura 3.8: Padrão observado utilizando o algoritmo sfqCoDel como disciplina de fila.

55 Mbit/s de *download*. O estrangulamento da largura de banda do *upstream* de 5 para 4 Mbit/s deve-se à necessidade de criar um estreitamento, à saída da rede, para impedir qualquer tipo de modelação de tráfego por parte do ISP.

Como se pode verificar nas figuras 3.9 a 3.13 (note-se a escala do eixo do *ping*), a redução do tamanho do espaço de retenção temporária leva a uma redução da latência. No entanto, esta continua a reagir quando a largura de banda é saturada tanto no *upstream* como no *downstream* e a latência aumenta em linha com o tamanho do espaço de retenção temporária, atingindo valores incomportáveis (figura 3.13). Quanto à taxa de transferência de *download*, pode observar-se que ao dimensionar o espaço de retenção temporária para um valor pequeno (figura 3.9), a redução observada torna-se um problema suportável, visto que não desce drasticamente. No entanto, ao aumentar o tamanho do espaço de retenção temporária verificamos que a situação se agrava atingindo, no pior caso, taxas 80% inferiores à largura de banda total disponível.

Quanto à taxa de entrega de pacotes, podemos observar que o aumento do tamanho do espaço de retenção temporária leva a uma redução da perda de pacotes (figura 3.14). Os valores apresentados referem-se à perda de pacotes num *burst* de tráfego enquanto ambas as larguras de banda do *downstream* e do *upstream* estão saturadas. Foram efetuadas três observações para cada tamanho do espaço de retenção temporária. Os resultados não são surpreendentes na medida em que a um menor espaço de retenção corresponde uma menor taxa de entrega, consequência natural da eliminação voluntária dos pacotes.

Observando os gráficos 3.9 a 3.13, verifica-se que alterando apenas a dimensão do espaço de retenção temporária não soluciona, em simultâneo, o problema da latência e da redução da taxa instantânea de *download*. Apesar de, com tráfego estável, a redução do tamanho do espaço de retenção temporária apresentar melhorias face à latência e à queda da taxa do *download*, o tráfego em *burst* provoca perda de pacotes, o que não é desejável considerando aplicações que sejam sensíveis a perdas, por exemplo aplicações que utilizem VoIP ou resoluções de DNS, podendo causar disrupção de serviço. Pode ainda observar-se, comparando as figuras 3.14 e 3.15, que ao aumentar o tamanho do espaço de retenção temporária, apesar de reduzir a perda de pacotes, provoca um aumento na latência média observada. A variação da latência, por se ter em conta apenas os pacotes entregues com sucesso, pode ser considerada natural, uma vez que a menores filas de espera corresponderão tempos menores. É então necessário procurar uma solução que combine a ideologia de AQM com o princípio das filas de prioridade.

Os algoritmos CoDel e sfqCoDel, por si só, não suportam a implementação de classes, ou seja, não é possível utilizar filas com prioridades diferentes. No entanto, Cake [5] é uma implementação que faz a agregação de um algoritmo denominado HTB (*"Hierarchical Token Bucket"*), que permite a utilização de classes, com o algoritmo de disciplina sfqCoDel, tornando assim possível a utilização das filas com prioridades juntamente com um algoritmo de AQM.

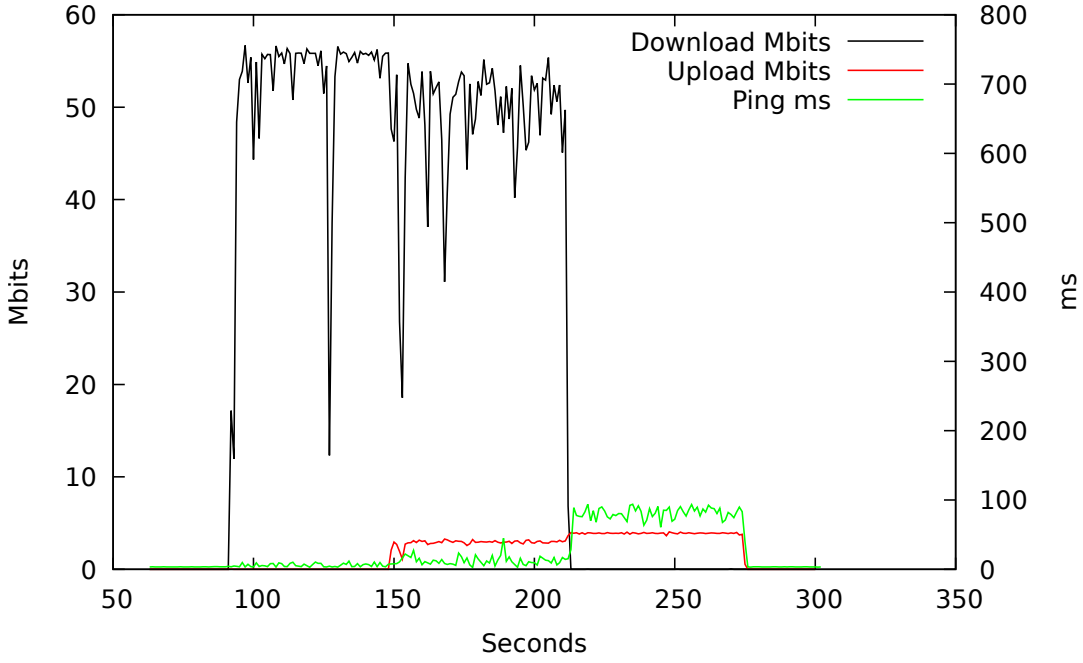

Figura 3.9: Espaço de retenção temporária de tamanho 30

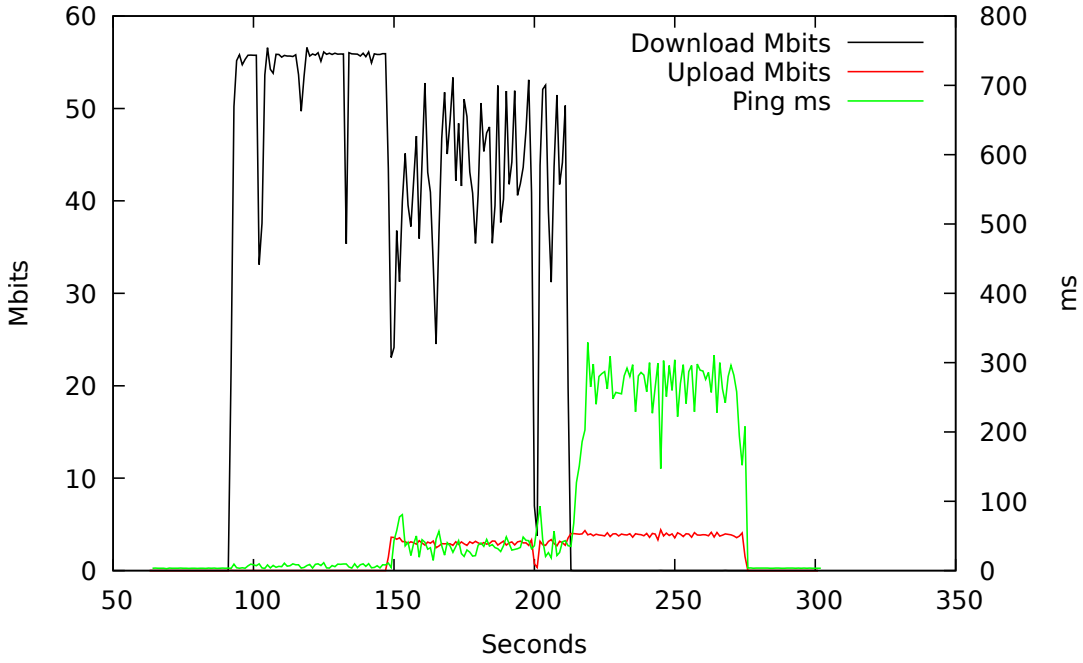

Figura 3.10: Espaço de retenção temporária de tamanho 100

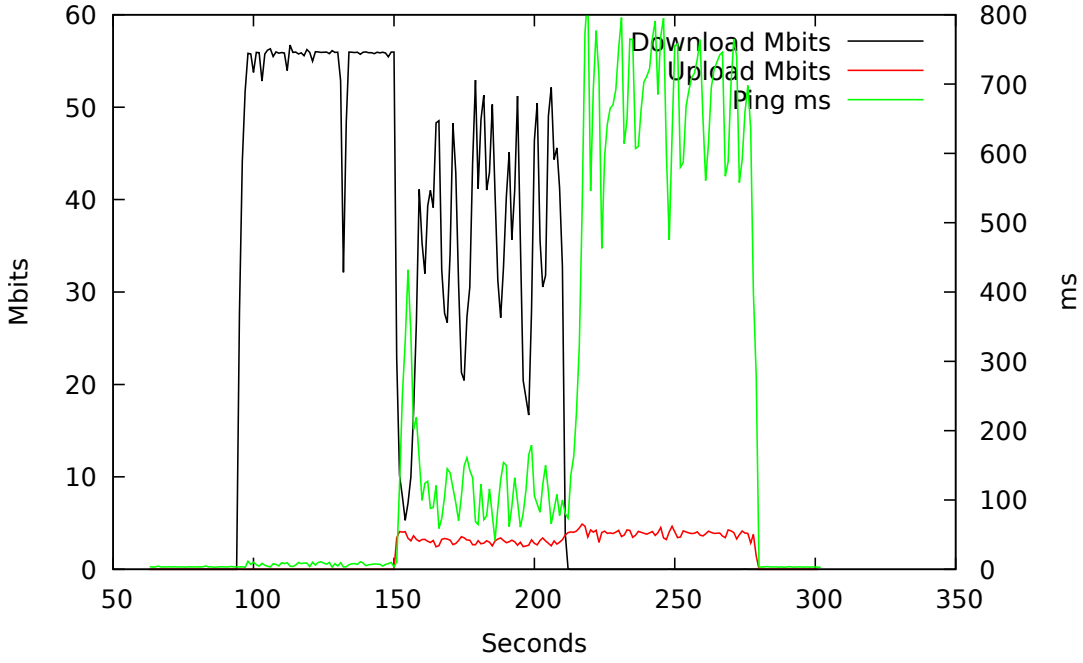

Figura 3.11: Espaço de retenção temporária de tamanho 250

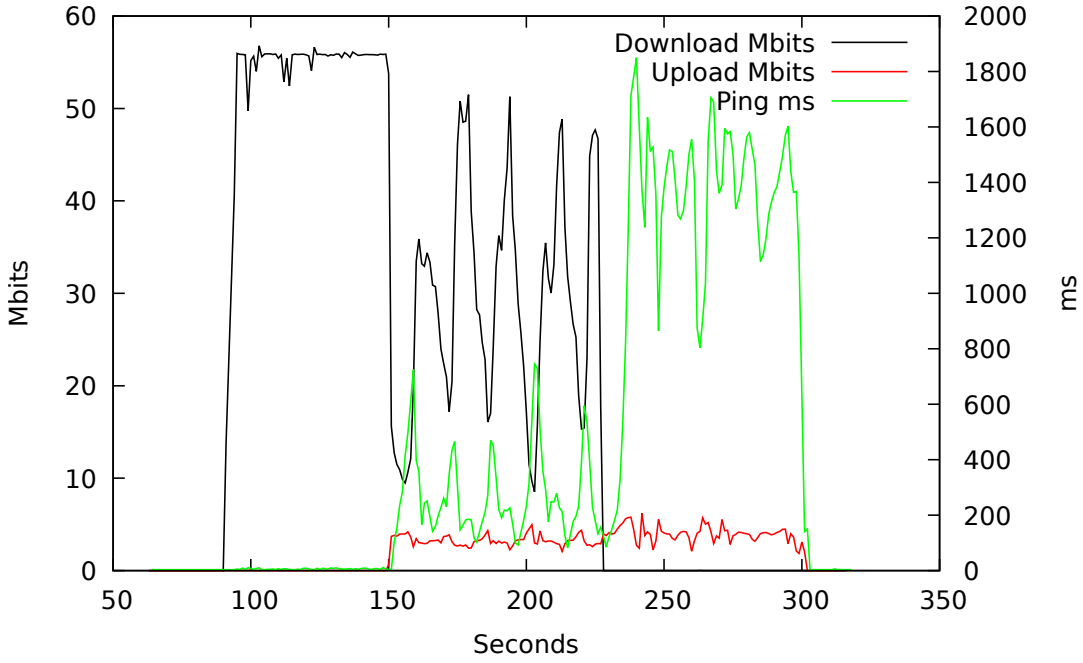

Figura 3.12: Espaço de retenção temporária de tamanho 500

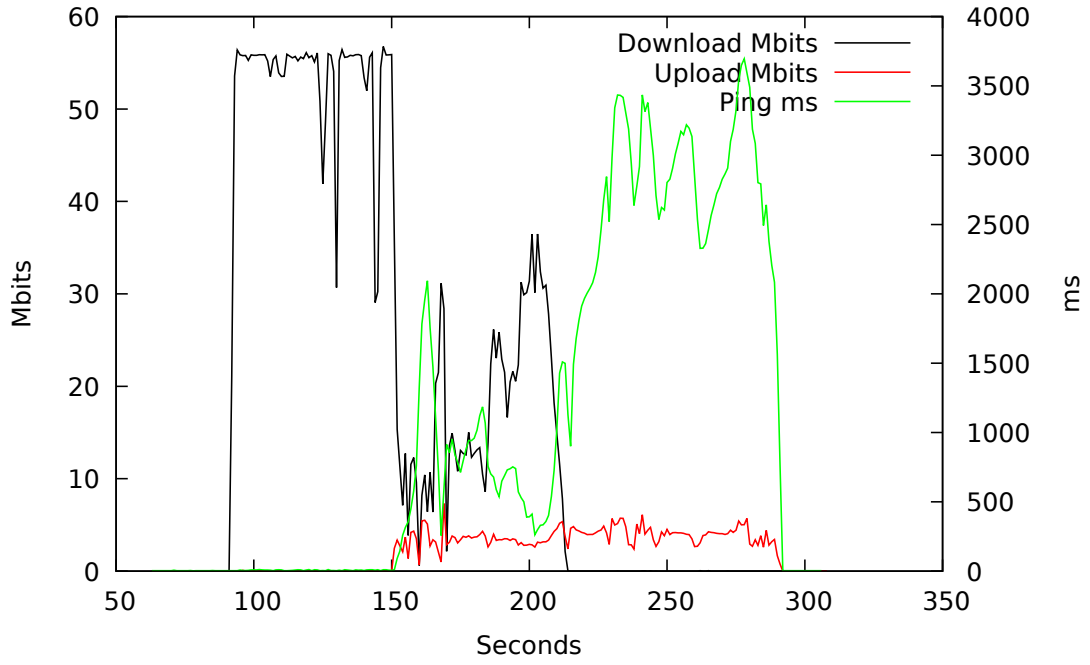

Figura 3.13: Espaço de retenção temporária de tamanho 1000

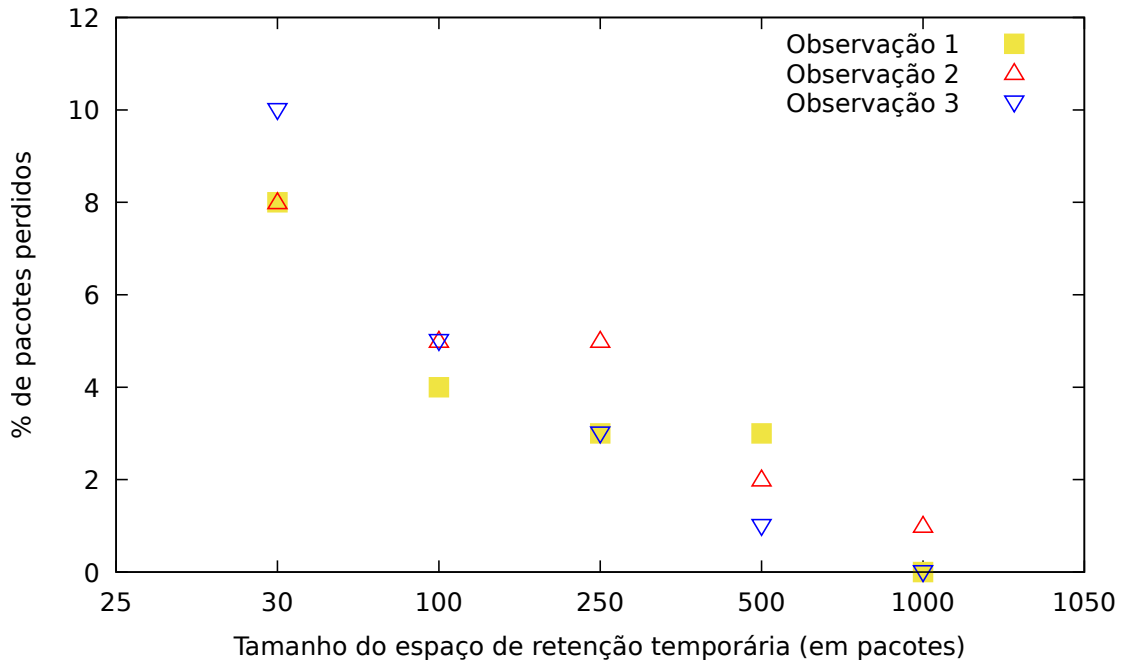

Figura 3.14: Percentagem de perda de pacotes, em *burst* de tráfego relativamente ao tamanho do espaço de retenção temporária.

A figura 3.16 demonstra os resultados obtidos utilizando a largura de banda total disponível, 55/5 Mbit/s (*download/upload*), espaço de retenção temporária por defeito e utilizando o algoritmo Cake como disciplina de fila.

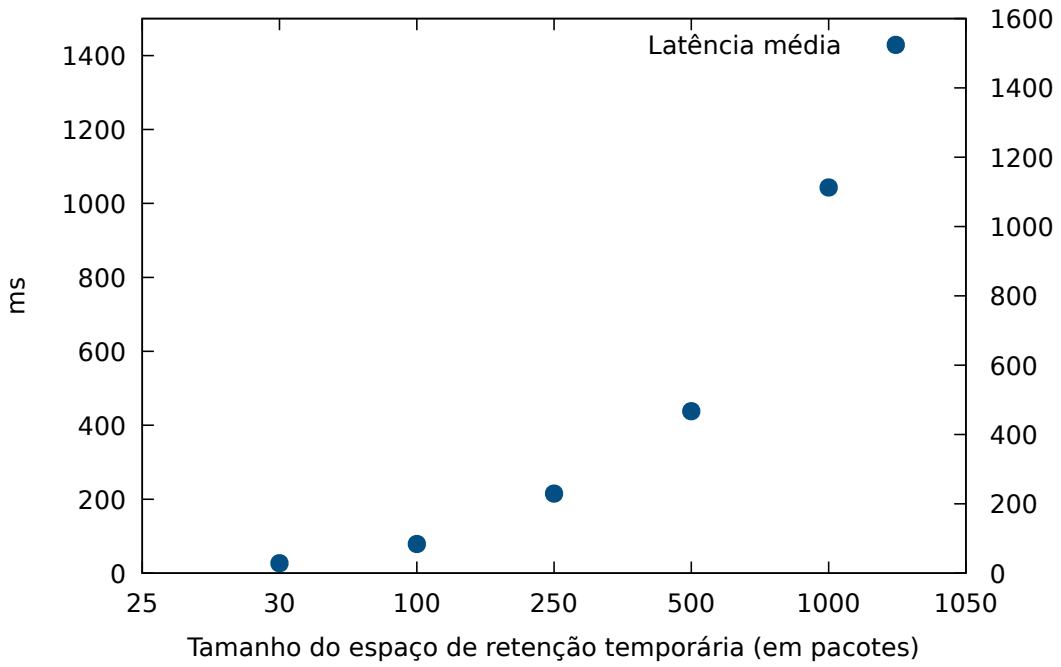

Figura 3.15: Latência média em ms, observada nas figuras 3.9 a 3.13, relativamente ao tamanho do espaço de retenção temporária.

O Cake utiliza o algoritmo o sfqCoDel e combina-o com a implementação de filas de espera com prioridades. A configuração base do Cake contém três filas de prioridade: uma fila de prioridade máxima que dá prioridade a pacotes mais sensíveis a atrasos, nomeadamente pacotes de *acknowledgement*, uma fila de prioridade intermédia, onde estão contemplados pacotes como os pacotes de VOIP ("*Voice Over IP*"), e uma fila para o restante tráfego.

De imediato são identificadas grandes diferenças face aos testes anteriormente apresentados. A latência exibe valores dentro dos valores referidos nos requisitos, entre aproximadamente 5 e 20 milissegundos, à excepção de alguns picos esporádicos em que atinge, no máximo, 149 milissegundos (segundo 190 na figura 3.16). Estes valores apresentam uma melhoria de cerca de 95% relativamente à latência observada nos primeiros testes. Quanto às taxas instantâneas de *download* e *upload*, verifica-se que na maior parte do teste a largura de banda manteve-se com uma utilização bastante satisfatória, com quedas ligeiras no *download*. Considerando que, o caso ótimo descreve-se como sendo a utilização total da largura de banda disponível, as oscilações na taxa instantânea de *download* são toleráveis, visto que a maior queda atingiu ainda assim 80% da taxa instantânea máxima.

A implementação do algoritmo Cake, como fila de disciplina, demonstra ser uma solução suficiente para resolver o problema da latência, cumprindo o requisito R12. Esta solução apazigua também o problema da queda da taxa instantânea do *download*, sustendo-a em valores próximos da utilização ótima da largura de banda disponível, satisfazendo assim o requisito R13. Pode então concluir-se que a implementação deste algoritmo

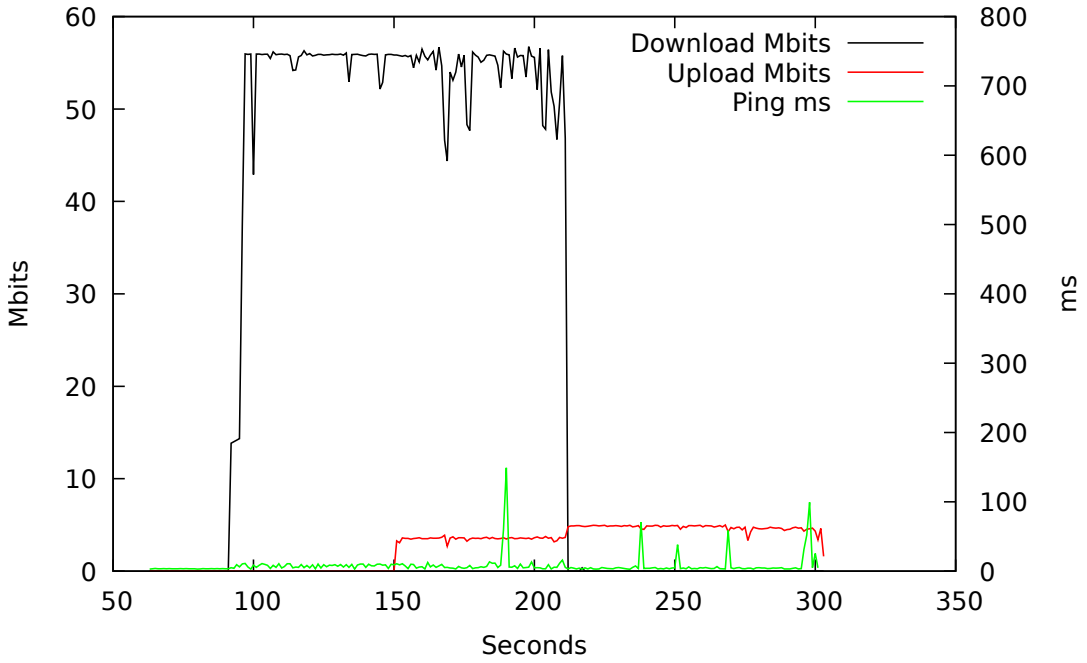

Figura 3.16: Algoritmo Cake, dia 6 de março de 2018 (meio da tarde)

apresenta-se como uma solução para o problema do *bufferbloat*, garantindo gestão do tráfego com qualidade de serviço.

3.3.5 *DHCP*

Numa rede de computadores é necessário atribuir endereços IP para que os dispositivos alojados nessa rede possam comunicar com outros dispositivos e utilizar os serviços da mesma. A atribuição dos endereços pode ser efetuada de forma estática ou de forma dinâmica, utilizando DHCP (*"Dynamic Host Configuration Protocol"*). Quando um dispositivo entra numa rede, efetua um *dhcp discovery* em difusão pela rede (*broadcast*), em busca de um servidor dessa rede. O servidor ao receber o *dhcp discovery* reserva um endereço IP na sua gama de endereços disponíveis e responde com uma oferta de uma *lease* (*dhcp offer*). Após receber uma oferta por parte do servidor DHCP, o cliente envia para o servidor um pedido dhcp (*dhcp request*) aceitando a oferta que lhe foi feita. Por fim o servidor efetua uma resposta de confirmação ao pedido (*dhcp acknowledgement*) enviando um conjunto de variáveis referentes à *lease* que lhe foi atribuída. A partir deste momento é esperado que o cliente efetue as suas configurações utilizando os valores que lhe foram transmitidos pelo servidor.

A atribuição estática de endereços IP pode ser feita através de configuração manual, diretamente no dispositivo, ou através dos servidores DHCP. A configuração estática é efetuada através do endereço MAC (*"Media Access Control"*), ou seja, o servidor DHCP atribui sempre o mesmo endereço IP a um dispositivo com um determinado endereço

MAC.

No caso particular das configurações disponibilizadas pela solução desenvolvida ao longo do projeto descrito neste documento, a atribuição de endereços IP depende da configuração do controlador. A instituição acolhedora fornece uma solução utilizando um servidor DHCP externo ao controlador de rede. Este servidor externo gere os endereços IP da rede primária, atribuindo-os com base no endereço MAC das máquinas. Esta medida restringe o acesso à rede primária a dispositivos registados no servidor. Qualquer outra rede que não a primária é gerida pelo próprio controlador, por exemplo a rede LAN secundária, a rede sem fios (WLAN) do controlador ou a rede destinada à utilização de VPN como descrito na secção 3.3.1. Caso não exista um servidor DHCP externo na rede do cliente, todas as redes locais passam a ser responsabilidade do servidor DHCP residente no controlador. O servidor ***isc-dhcp*** foi o elegido para desempenhar a função de servidor DHCP.

3.3.6 *DNS*

Cada sítio na *Internet* pode ser identificado por um endereço IP público e um *hostname*. Esses endereços podem ser simples de memorizar, como 8.8.8.8, ou mais complicados, como 2a00:1450:4003:802::200e, enquanto que os *hostnames* são curtos e fáceis de captar, por exemplo google.pt. Os seres Humanos têm muito mais facilidade em decorar conjuntos de palavras ou mnemónicas contrariamente a conjuntos algo extensos de números e letras sem sentido aparente. Visto que os computadores identificam a localização de um determinado sítio ou serviço pelo endereço IP é necessário um "dicionário"que faça a tradução de *hostnames*, facilmente reconhecíveis, para endereços IP, e vice-versa. O DNS (*"Domain Name Service"*) é uma base de dados implementada de forma distribuída e em hierarquia [9] que efetua esta tradução. Neste projeto, foi eleito o *software* **bind** para desempenhar as funções de servidor de nomes. O *software* foi configurado com as configurações base pois desta forma desempenha as funções desejadas.

Na solução desenvolvida ao longo deste projeto, é necessário ter em consideração outras questões no que toca à resolução de nomes. A instituição acolhedora utiliza um servidor IPBrick [8] integrado nas redes dos seus clientes. Este servidor desempenha diversas funções, de entre as quais, resolução de DNS interno. A instalação e configuração deste servidor não está contemplada no âmbito deste projeto. Na presença de um servidor IPBrick na rede do cliente, a rede primária utiliza-o como servidor de DNS. As restantes redes locais do cliente utilizam o controlador como servidor de DNS.

3.3.7 *VPN*

As redes locais de uma instituição podem alojar serviços que são internos a essa rede. Alguns desses serviços podem ser acedidos a partir da *Internet*, no entanto, há serviços que

não devem ser acessíveis a partir de uma rede externa. A utilização de uma VPN permite implementar numa instituição uma política que permite trabalhar remotamente, não pondo em causa a segurança e confidencialidade do trabalho. Desta forma, os funcionários desse cliente podem aceder aos serviços da sua rede local como se estivessem a aceder fisicamente a um ponto de acesso dessa rede. A potencial dispersão geográfica dos escritórios de uma instituição representa outro motivo para a implementação de uma VPN. Tipicamente essa dispersão constitui uma necessidade de interligar, de forma lógica, os diferentes escritórios da instituição por forma a criar uma rede global que permite comunicação entre dispositivos de forma segura e confidencial. A utilização de VPN induz uma abstração geográfica e os dispositivos operam como se estivessem na mesma rede LAN.

Nesta solução, o *software* **openvpn** é o utilizado para desempenhar a função de servidor VPN interno ao controlador desenvolvido ao longo do projeto. O acesso à VPN é exclusivo a utilizadores que estejam num determinado grupo no domínio da instituição, como referido na secção 3.3.9.

3.3.8 Pontos de Acesso e *Roaming*

Pontos de acesso de uma rede são definidos como os locais onde é possível ligar um dispositivo a uma rede. Estes pontos de acesso são tipicamente representados por tomadas de rede, onde se pode ligar um cabo *Ethernet*, ou pontos de acesso sem fios.

Nas diferentes configurações possíveis para a solução desenvolvida neste projeto, os pontos de acesso são tidos em conta consoante as funções da LAN onde são colocados. A rede primária, ou "segura", apenas dispõe de pontos de acesso físicos, tomadas de rede ao longo da instituição interligadas por um comutador que também está ligado ao controlador. No caso da rede secundária, ou "não segura", a solução tem prevista a utilização de pontos de acesso sem fios, criando uma rede WLAN (*"Wireless Local Area Network"*) para dispositivos convidados. Isto permite que dispositivos móveis possam utilizar uma rede sem fios, nas instalações da instituição em questão, com acesso à *Internet* sem pôr em causa a segurança e confidencialidade da rede primária. A solução desenvolvida prevê a possível utilização de uma terceira rede local para implementação de uma LAN e WLAN com o propósito de disponibilizar o serviço VPN com cobertura total das instalações da instituição cliente.

Disposição de Pontos de Acesso

Atingir uma cobertura total de sinal das instalações da instituição cliente é o objetivo ótimo da implementação de uma WLAN para dispositivos convidados. No entanto, é necessário ter em consideração diversas variáveis externas que põem em causa essa cobertura. A disposição das instalações é o maior desafio neste problema, uma vez que todo o planeamento de colocação de pontos de acesso sem fios depende dessa disposição.

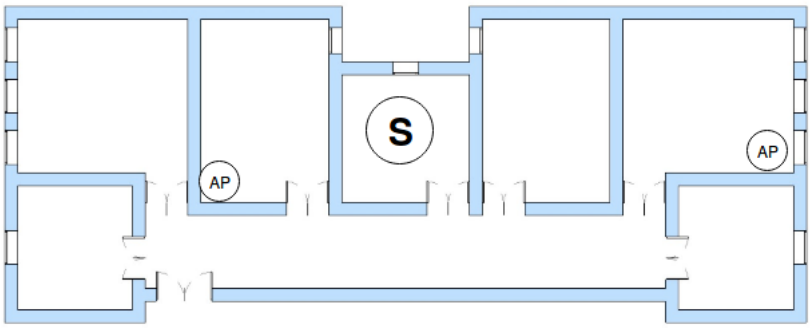

Figura 3.17: Planta exemplo de um escritório.

Outros desafios da colocação dos pontos de acesso sem fios são referentes às características físicas das ondas magnéticas. Algumas dessas características são mais relevantes para a colocação dos pontos de acesso, nomeadamente, o facto de o sinal ter dificuldades em trespassar paredes. Em acréscimo, não se deve configurar dois pontos de acesso para transmitirem no mesmo canal rádio, caso os seus sinais se sobreponham.

Suponha-se, como exemplo, que é necessário instalar pontos de acesso sem fios para cobrir um escritório que tem uma disposição como a representada na figura 3.17. Instalar apenas um ponto de acesso na divisão representada pelo símbolo **"S"** poderia não ser suficiente para atingir cobertura total do escritório. As divisões nas pontas direita e esquerda do escritório ficariam com pouco ou nenhum sinal. Uma disposição possível para atingir cobertura de sinal total, encontra-se exemplificada pelos símbolos **"AP"**. O ponto de acesso mais à esquerda é colocado numa das divisões interiores para dar melhor cobertura à divisão marcada por **"S"**. O ponto de acesso mais à direita é colocado o mais perto da ponta do escritório para não existir muita sobreposição de sinal. Mesmo colocando os pontos de acesso distantes um do outro haverá alguma sobreposição de sinal, logo, cada ponto de acesso deve transmitir em canais de rádio com uma distância mínima de 5 canais, evitando desta forma a sobreposição das frequências de transmissão.

Roaming

Com cobertura total, numa WLAN onde há mais do que um ponto de acesso, a transição entre dois pontos de acesso (*roaming*) deve ser imperceptível pelo utilizador quando este se desloca fisicamente. Para tal, o dispositivo deve preservar o seu endereço IP, independentemente do ponto de acesso a que está associado. A qualidade e força do sinal de um ponto de acesso degrada-se incrementalmente à medida que o dispositivo móvel se afasta do mesmo e, ao entrar no alcance de um outro ponto de acesso, irá receber um *beacon* desse novo ponto de acesso. Tipicamente, dentro da mesma instituição, os pontos de acesso terão o mesmo SSID (*"Service Set ID"*), se assim for, o dispositivo irá desassociar-se do ponto de acesso antigo e associar-se ao ponto de acesso novo sem perder

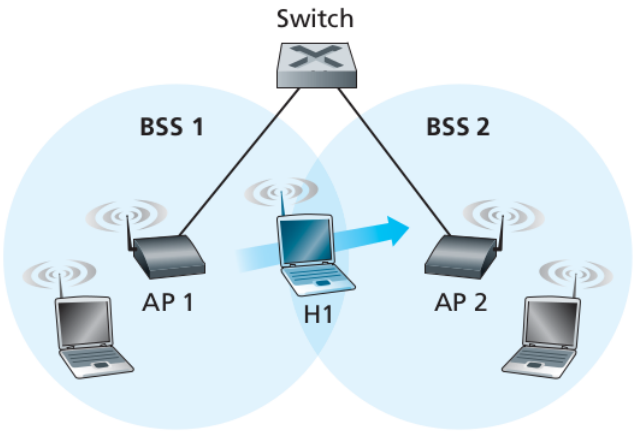

Figura 3.18: Mobilidade de um dispositivo (retirado de [9])

as ligações estabelecidas e sem necessitar de nova autenticação e renovação do endereço IP, efetuando a transição de forma transparente para o utilizador. Quando o dispositivo se associa a um novo ponto de acesso, os encaminhadores da rede ainda não estão conscientes da mudança. Uma forma de atualizar os encaminhadores é efetuar um *ethernet broadcast* a partir do novo ponto de acesso, utilizando o endereço MAC (*"Media Access Control"*) do dispositivo móvel como endereço origem [9].

3.3.9 Autenticação em Rede

Uma rede local de uma instituição pode albergar diversos serviços, diversos dispositivos e uma grande quantidade de utilizadores frequentes. Os dispositivos desta rede local podem ser partilhados por diversos utilizadores dessa instituição, por exemplo computadores de uma sala comum numa universidade. A complexidade da gestão, destes computadores partilhados, cresce com o número de utilizadores existentes, sendo necessário criar utilizadores em cada máquina para permitir o seu uso, tornando incomportável a manutenção destes computadores num cenário de maior escala.

Como referido anteriormente, a instituição acolhedora utiliza um servidor IPBrick que desempenha diversas funcionalidades na rede do cliente. Este servidor permite também efetuar autenticação dos utilizadores no domínio, centralizando a criação dos utilizadores no servidor.

O controlador utiliza o servidor IPBrick para efetuar a autenticação dos utilizadores no serviço de VPN. Desta forma os utilizadores podem usufruir do serviço de VPN utilizando as suas credenciais do domínio para autenticação. O ficheiro de configuração "pam_winbind.conf" permite especificar os grupos aos quais é permitido efetuar autenticação. Desta forma, o serviço VPN fica limitado apenas aos utilizadores de um determinado grupo.

Neste projeto foi utilizado o *software* **winbind**, que efetua a tradução de grupos de domínios *Windows* para grupos *Unix*, efetua a ligação ao servidor IPBrick e permite a

autenticação dos utilizadores.

3.3.10 Automatização

O processo de instalação e configuração do controlador de rede apresentado neste documento é longo e minucioso. Se todo o processo for efetuado manualmente, a probabilidade de erro torna-se muito elevada devido à complexidade e profundidade dos procedimentos. A automação do processo atua em prol da solução, permitindo escalar no número de clientes, no entanto, a implementação de uma solução com procedimentos automáticos traz desafios devido à heterogeneidade de configurações possíveis para diferentes clientes.

A primitiva utilizada para implementar um procedimento de instalação automática, consiste em tratar a infraestrutura de instalação como um serviço onde, dependendo dos dados de entrada, o resultado é diferente. O programa principal de instalação e todos os ficheiros necessários para a configuração completa do controlador estão desenhados como um esqueleto. Em todos os ficheiros, os valores que dependem do cliente onde irá ser instalado o controlador, estão marcados com palavras chave que serão posteriormente substituídas pelo valor correspondente no cliente em questão (Listagem 3.2).

Listagem 3.2: Exemplo de palavras chave num ficheiro

```
auth_algs=3
beacon_int=100
channel=6
country_code=PT
driver=nl80211
hw_mode=g
interface=<WIRELESS_IFACE>
ssid=<SSID>
```

Desta forma é possível reduzir a complexidade da instalação a apenas alguns passos. O passo de maior importância reside no preenchimento correto do ficheiro que contém todas as variáveis necessárias para a instalação e configuração do controlador, visto que todo o procedimento depende desse ficheiro. O procedimento de instalação e configuração é idempotente. Se uma instalação for executada diversas vezes com o mesmo ficheiro de configuração o seu resultado final não será diferente.

A automação do procedimento de instalação e configuração permite que a solução escale em número de clientes, obtendo assim as características necessárias que permitem colocar no mercado uma solução chave na mão, flexível para cada cliente e baseada em código aberto.

Capítulo 4

Avaliação

Neste capítulo pretende-se testar a instalação e configuração do controlador, verificando o bom funcionamento dos diferentes componentes do mesmo. Em primeiro lugar será apresentada uma bateria de testes a efetuar e uma descrição do resultado esperado desses testes. Em segundo lugar são apresentados os resultados dos testes descritos na bateria. Por fim, é descrita a entrada em produção do controlador.

4.1 Bateria de Testes

O conjunto de testes necessários para verificação e validação do controlador consiste na realização de diversas tarefas que contemplam todo o processo de instalação e configuração do mesmo. Em primeiro lugar é necessário confirmar que o dispositivo arranca corretamente o sistema operativo. De seguida efetua-se o processo de instalação e configuração do *software* necessário seguindo as instruções expressas na base de conhecimento do projeto. Para concluir esta tarefa com sucesso é necessário chegar ao final das instruções sem que tenha sido necessário intervenção da pessoa que faz a instalação, exceto para preparação do ambiente de instalação e diálogos gerados pelo processo de instalação, por exemplo pedidos de *password*.

Após a instalação do controlador os primeiros testes a serem efetuados são relacionados com funções cruciais do equipamento:

- Verificação das interfaces de rede;
- Teste de ligação à *Internet*;
- Teste de tradução de endereços (NAT);
- Teste de resolução de nomes (DNS).

A verificação das interfaces de rede é efetuada após reiniciar o controlador, consultando o ficheiro */etc/networking/interfaces*, utilizando o comando *ifconfig* e comparando a

informação providenciada pelo ficheiro com a informação providenciada pelo comando. A informação retornada pelo comando *ifconfig* tem de ser coerente com o ficheiro de configuração das interfaces para que o teste seja bem sucedido. O teste de ligação à *Internet* efetua-se executando o comando *ping* para testar a conectividade a uma máquina externa à rede da instituição. Se o comando retornar valores de tempo de ida e volta (rtt), o controlador tem acesso à *Internet* e o teste é bem sucedido.

Para testar a execução de NAT, devem ser efetuados *pings* a uma máquina externa à rede, a partir de uma máquina interna. Durante a execução dos *pings*, deve iniciar-se uma captura de pacotes com o comando *tcpdump*. Deve então verificar-se os endereços IP de origem e destino dos pacotes referentes aos *pings* efetuados.

O teste para verificar a resolução de nomes consiste em executar o comando *host* e tentar resolver o nome de uma máquina externa, por exemplo google.pt. Se existir um servidor IPBrick na rede, outra parte do teste consiste em tentar resolver um nome que apenas o servidor reconhece.

Deve efetuar-se uma verificação das regras da antepara de segurança, por exemplo verificar o resultado da execução do comando *iptables-save* e o conteúdo do ficheiro */etc/iptables/rules.v4*. Seguem-se testes para verificar que os servidores de DHCP e VPN estão a funcionar corretamente. No caso do servidor DHCP, o teste consiste em ligar um dispositivo na rede, verificar se o dispositivo obtém endereço IP por pedido DHCP e consultar o ficheiro que regista as *leases* entregues pelo servidor. O teste ao servidor VPN, consiste em ligar a VPN e verificar que a interface lógica criada no cliente está corretamente configurada.

Os testes referentes à ligação ao servidor IPBrick consistem em executar dois comandos: *wbinfo -u* e *wbinfo -g*. Os comandos executam pedidos para obter, respetivamente, a lista de utilizadores e a lista dos grupos existentes no servidor IPBrick.

O controlador deve também ser testado utilizando diferentes configurações para assegurar que as diferentes hipóteses são suportadas. No caso de uma configuração com duas ligações WAN, devem ser efetuados testes específicos para esta configuração (sem nenhuma ordem em particular):

1. Arranque com falha da ligação principal;
2. Arranque com falha da ligação secundária;
3. Falta da ligação primária durante funcionamento normal;
4. Falta da ligação secundária durante funcionamento normal;
5. Arranque com ambas as ligações em falta.

No teste descrito em 1, espera-se que o dispositivo arranque sem ligação à *Internet* sendo que ao fim de um ciclo de verificações do mecanismo de tolerância a faltas, todo o

tráfego do dispositivo que é encaminhado pela ligação primária seja redirecionado para a ligação secundária, recuperando assim a ligação ao exterior. Na situação descrita em 2, espera-se que o dispositivo arranque normalmente e que após um ciclo de verificação, o tráfego de servidores que esteja a ser encaminhado pela ligação secundária seja redirecionado para a primária. No caso dos testes 3 e 4, o teste consiste em efetuar um *ping* a uma máquina que esteja fora da rede e desligar o cabo por onde o *ping* esteja a ser efetuado. Em 3 o *ping* pode ser efetuado a partir de qualquer máquina que esteja dentro da rede. No teste 4, o *ping* tem de ser efetuado a partir de uma máquina cujo tráfego esteja a ser direcionado para a ligação secundária. Nos testes 3 e 4 espera-se que os pacotes de *ping* sejam perdidos a partir de o momento em que o cabo é desligado até que o mecanismo de tolerância a faltas encaminhe o tráfego para a ligação que está ativa. No teste descrito em 5, espera-se que o controlador arranque sem ligação à *Internet* mas que todos os serviços internos da rede continuem o seu normal funcionamento.

Serão também efetuados testes para obter uma estimativa do atraso induzido pelo controlador na propagação de pacotes, analisando observações obtidas através de *pings* e capturas de pacotes com o comando *tcpdump*.

4.2 Resultados

Para realizar os testes, foi utilizado o controlador que se encontra em operação nas instalações da instituição acolhedora. Foi também configurado um controlador de testes com duas ligações externas, sendo que as ligações são providenciadas por duas redes diferentes nas instalações da instituição acolhedora, simulando duas ligações independentes.

O processo de instalação do controlador cumpre os requisitos estipulados. O procedimento completo leva cerca de 10 a 15 minutos a concluir e apenas requer intervenção do responsável pela instalação para a preparação do ambiente e testes de verificação pós instalação, estando de acordo com os objetivos especificados, cumprindo o requisito R14. A solução desenvolvida suporta diversas configurações sendo que a instalação é automática, independentemente da configuração utilizada. Em caso de substituição basta refazer a instalação utilizando o ficheiro de configuração do cliente em questão, sem requerer mais intervenção.

Para que a verificação das interfaces de rede seja bem sucedida, o resultado de executar o comando *ifconfig* tem de refletir o conteúdo do ficheiro */etc/networking/interfaces*. Pode verificar-se que o resultado da execução do comando *ifconfig* (Listagem 4.1) reflete o conteúdo do ficheiro das interfaces (Listagem 4.2), pelo que é seguro afirmar que a configuração foi bem efetuada.

Listagem 4.1: Excerto do resultado do comando ifconfig

```
~$ ifconfig
```

```
br0       Link encap:Ethernet  HWaddr 00:0d:b9:46:1d:08
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask
             :255.255.255.0
          inet6 addr: fe80::20d:b9ff:fe46:1d08/64 Scope:
             Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric
             :1
          RX packets:208227540 errors:0 dropped:0 overruns
             :0 frame:0
          TX packets:315839758 errors:0 dropped:0 overruns
             :0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:164819268239 (164.8 GB)  TX bytes
             :454246716055 (454.2 GB)

enp1s0    Link encap:Ethernet  HWaddr 00:0d:b9:46:1d:08
          inet6 addr: fe80::20d:b9ff:fe46:1d08/64 Scope:
             Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500
              Metric:1
          RX packets:250737512 errors:0 dropped:0 overruns
             :0 frame:0
          TX packets:368515913 errors:0 dropped:0 overruns
             :0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:170491976297 (170.4 GB)  TX bytes
             :455817165836 (455.8 GB)
          Memory:fe500000-fe51ffff

(......)
```

Listagem 4.2: Excerto do ficheiro das interfaces

```
~$ cat /etc/networking/interfaces
auto lo
iface lo inet loopback

auto br0
iface br0 inet static
        address 192.168.1.1
        network 192.168.1.0
        netmask 255.255.255.0
        broadcast 192.168.1.255
        dns-nameservers 192.168.1.7
        bridge_ports enp1s0
        bridge_stp off
```

```
(....)
```

O teste de ligação à *Internet* consiste em executar um *ping* a uma máquina externa à rede. Foram efetuados testes de ligação à *Internet* a partir do controlador e a partir de ambas as redes internas. O teste foi bem sucedido em todos os casos, confirmando a existência de ligação à *Internet*, ficando o requisito R4 cumprido. A Listagem 4.3 apenas apresenta um dos testes.

Listagem 4.3: Teste de ping efetuado

```
~$ ping google.pt
PING google.pt (172.217.168.163) 56(84) bytes of data.
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163):
   icmp_seq=1 ttl=56 time=17.4 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163):
   icmp_seq=2 ttl=56 time=17.5 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163):
   icmp_seq=3 ttl=56 time=17.3 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163):
   icmp_seq=4 ttl=56 time=19.1 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163):
   icmp_seq=5 ttl=56 time=19.0 ms
--- google.pt ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time
   3003ms
rtt min/avg/max/mdev = 17.340/17.873/19.180/0.768 ms
```

Como esperado, ao executar o comando *tcpdump* para capturar pacotes que passam pela interface WAN, os pacotes respetivos aos *pings* efetuados a partir de um dispositivo hospedado na rede LAN apresentam o IP público como origem e destino. Observando as Listagens 4.4 e 4.5, o endereço IP 192.168.1.169 é o endereço IP do computador hospedado na rede LAN, o endereço IP_CONTROLADOR representa o endereço IP público do controlador e o endereço mad07s10-in-f3.1e100.net é o endereço da máquina à qual os *pings* foram efetuados.

Pode observar-se que os pacotes que são enviados (*"echo request"*) a partir da máquina interna, chegam à interface LAN do controlador tendo como endereço origem o endereço IP dessa máquina e como endereço destino o endereço da máquina externa à qual foram efetuados os *pings* (Listagem 4.4). Quando os pacotes são enviados a partir da interface de rede WAN do controlador, o endereço de origem dos pacotes passa a ser o endereço do controlador (Listagem 4.5). Na direção contrária, quando as respostas (*"echo reply"*) aos pacotes enviados chegam à interface WAN, o seu endereço origem é o da máquina externa à qual os *pings* foram efetuados e o endereço destino é o endereço público do

controlador (Listagem 4.5). Posteriormente quando esses pacotes saem pela interface LAN do controlador, o seu endereço destino foi substituído pelo endereço da máquina interna (Listagem 4.4). Este comportamento identifica o bom funcionamento do NAT, cumprindo o requisito R5.

Listagem 4.4: Captura do comando tcpdump na interface LAN

```
10:55:49.731837 IP (tos 0x0, ttl 64, id 25099, offset 0,
   flags [DF], proto ICMP (1), length 84) 192.168.1.169 >
   mad07s10-in-f3.1e100.net: ICMP echo request, id 17616,
   seq 9, length 64

10:55:49.742091 IP (tos 0x0, ttl 56, id 0, offset 0, flags
   [none], proto ICMP (1), length 84) mad07s10-in-f3.1e100.
   net > 192.168.1.169: ICMP echo reply, id 17616, seq 9,
   length 64

10:55:50.733852 IP (tos 0x0, ttl 64, id 25140, offset 0,
   flags [DF], proto ICMP (1), length 84) 192.168.1.169 >
   mad07s10-in-f3.1e100.net: ICMP echo request, id 17616,
   seq 10, length 64

10:55:50.745978 IP (tos 0x0, ttl 56, id 0, offset 0, flags
   [none], proto ICMP (1), length 84) mad07s10-in-f3.1e100.
   net > 192.168.1.169: ICMP echo reply, id 17616, seq 10,
   length 64

10:55:51.735876 IP (tos 0x0, ttl 64, id 25229, offset 0,
   flags [DF], proto ICMP (1), length 84) 192.168.1.169 >
   mad07s10-in-f3.1e100.net: ICMP echo request, id 17616,
   seq 11, length 64

10:55:51.746248 IP (tos 0x0, ttl 56, id 0, offset 0, flags
   [none], proto ICMP (1), length 84) mad07s10-in-f3.1e100.
   net > 192.168.1.169: ICMP echo reply, id 17616, seq 11,
   length 64

10:55:52.737401 IP (tos 0x0, ttl 64, id 25427, offset 0,
   flags [DF], proto ICMP (1), length 84) 192.168.1.169 >
   mad07s10-in-f3.1e100.net: ICMP echo request, id 17616,
   seq 12, length 64

10:55:52.749409 IP (tos 0x0, ttl 56, id 0, offset 0, flags
   [none], proto ICMP (1), length 84) mad07s10-in-f3.1e100.
   net > 192.168.1.169: ICMP echo reply, id 17616, seq 12,
   length 64
```

Listagem 4.5: Captura do comando tcpdump na interface WAN

```
10:55:49.732246 IP (tos 0x0, ttl 63, id 25099, offset 0,
   flags [DF], proto ICMP (1), length 84) IP_CONTROLADOR >
   mad07s10-in-f3.1e100.net: ICMP echo request, id 17616,
   seq 9, length 64

10:55:49.742044 IP (tos 0x0, ttl 57, id 0, offset 0, flags
   [none], proto ICMP (1), length 84) mad07s10-in-f3.1e100.
   net > IP_CONTROLADOR: ICMP echo reply, id 17616, seq 9,
   length 64

10:55:50.735859 IP (tos 0x0, ttl 63, id 25140, offset 0,
   flags [DF], proto ICMP (1), length 84) IP_CONTROLADOR >
   mad07s10-in-f3.1e100.net: ICMP echo request, id 17616,
   seq 10, length 64

10:55:50.745937 IP (tos 0x0, ttl 57, id 0, offset 0, flags
   [none], proto ICMP (1), length 84) mad07s10-in-f3.1e100.
   net > IP_CONTROLADOR: ICMP echo reply, id 17616, seq 10,
    length 64

10:55:51.736542 IP (tos 0x0, ttl 63, id 25229, offset 0,
   flags [DF], proto ICMP (1), length 84) IP_CONTROLADOR >
   mad07s10-in-f3.1e100.net: ICMP echo request, id 17616,
   seq 11, length 64

10:55:51.746206 IP (tos 0x0, ttl 57, id 0, offset 0, flags
   [none], proto ICMP (1), length 84) mad07s10-in-f3.1e100.
   net > IP_CONTROLADOR: ICMP echo reply, id 17616, seq 11,
    length 64

10:55:52.739518 IP (tos 0x0, ttl 63, id 25427, offset 0,
   flags [DF], proto ICMP (1), length 84) IP_CONTROLADOR >
   mad07s10-in-f3.1e100.net: ICMP echo request, id 17616,
   seq 12, length 64

10:55:52.749364 IP (tos 0x0, ttl 57, id 0, offset 0, flags
   [none], proto ICMP (1), length 84) mad07s10-in-f3.1e100.
   net > IP_CONTROLADOR: ICMP echo reply, id 17616, seq 12,
    length 64
```

O comando *host* executa resolução de DNS. O sucesso na execução deste comando confirma o bom funcionamento do servidor DNS local. Existindo um servidor IPBrick na rede, o controlador tem de conseguir resolver o nome do domínio local. Como podemos

verificar na Listagem 4.6, o controlador resolveu tanto o nome de um domínio externo, google.pt, como um nome do domínio local, ipbrick.asolido.lan, confirmando o bom funcionamento dos servidores DNS, contemplando o requisto R6.

Listagem 4.6: Exemplos do comando host

```
~$ host google.pt
google.pt has address 172.217.168.163
google.pt has IPv6 address 2a00:1450:4006:805::2003
google.pt mail is handled by 40 alt3.aspmx.l.google.com.
google.pt mail is handled by 50 alt4.aspmx.l.google.com.
google.pt mail is handled by 10 aspmx.l.google.com.
google.pt mail is handled by 20 alt1.aspmx.l.google.com.
google.pt mail is handled by 30 alt2.aspmx.l.google.com.

~$ host ipbrick.asolido.lan
ipbrick.asolido.lan has address 192.168.1.7
```

Ao executar o comando *iptables-save*, observa-se uma representação das regras da antepara de segurança atualmente em execução, como demonstrado na Listagem 4.7. As políticas de *INPUT*, *FORWARD* e *OUTPUT* da antepara de segurança estão de acordo com o pretendido, estando configuradas com *"DROP"*, *"DROP"* e *"ACCEPT"* respetivamente. Observa-se a regra que permite que ligações já estabelecidas ou relacionadas sejam aceites e pode ainda observar-se regras para permitir comunicação em *loopback*, para permitir respostas a *pings* e para permitir ligações *ssh*. Verifica-se então que a antepara de segurança está corretamente configurada. Com isto podemos verificar que o requisito R7 está cumprido.

Listagem 4.7: Excerto do resultado do comando iptables-save

```
(....)
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
:LIMIT_INDIVIDUAL - [0:0]
:LIMIT_OVERALL - [0:0]
-A INPUT -i lo -j ACCEPT
-A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
(....)
```

Para testar o funcionamento do serviço de DHCP, foram ligados dois dispositivos à rede secundária do controlador. Ambos os dispositivos receberam endereço IP dinamicamente, como especificado no requisito R9. Consultando o ficheiro que contém as *leases* entregues

pelo servidor DHCP, como demonstrado na Listagem 4.8, pôde confirmar-se que os endereços MAC e os endereços IP correspondiam aos endereços dos dispositivos que foram ligados. Pode então afirmar-se que o serviço de DHCP está funcional no controlador. Não foram efetuados testes na rede primária pois o DHCP nessa rede é responsabilidade do servidor IPBrick.

Listagem 4.8: Ficheiro de leases do servidor DHCP

```
lease 192.168.5.99 {
  starts 5 2018/05/11 10:16:57;
  ends 5 2018/05/11 10:26:57;
  cltt 5 2018/05/11 10:16:57;
  binding state active;
  next binding state free;
  rewind binding state free;
  hardware ethernet 9c:b6:d0:8b:04:35;
  client-hostname "tiago-XPS";
}
lease 192.168.5.91 {
  starts 5 2018/05/11 10:17:41;
  ends 5 2018/05/11 10:27:41;
  cltt 5 2018/05/11 10:17:41;
  binding state active;
  next binding state free;
  rewind binding state free;
  hardware ethernet c0:ee:fb:58:d4:a7;
  uid "\001\300\356\373X\324\247";
  set vendor-class-identifier = "android-dhcp-6.0.1";
  client-hostname "android-efdbdf100fb09ee0";
}
```

Quando um computador se liga à VPN disponibilizada pelo controlador, é gerada uma interface lógica denominada *"tap"* no cliente de VPN. Essa interface pode ser verificada utilizando o comando *iptables* no computador que está ligado à VPN, satisfazendo o requisito R11.

Foi efetuada uma ligação VPN a partir do exterior da rede, estando geograficamente distante da localização física da rede interna e utilizando um ISP diferente para efetuar o teste. Como se pode verificar na Listagem 4.9 a interface lógica foi criada e foi-lhe atribuída um endereço IP da rede interna.

Listagem 4.9: Interfaces de rede de um computador alojado na rede interna utilizando VPN

```
enx106530960cf5 Link encap:Ethernet  HWaddr 10:65:30:96:0c:
   f5
```

```
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame
             :0
          TX packets:0 errors:0 dropped:0 overruns:0
             carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:32183 errors:0 dropped:0 overruns:0
             frame:0
          TX packets:32183 errors:0 dropped:0 overruns:0
             carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2851294 (2.8 MB)  TX bytes:2851294 (2.8
             MB)

tap0      Link encap:Ethernet  HWaddr 02:27:50:e0:37:f0
          inet addr:192.168.1.204  Bcast:192.168.1.255
             Mask:255.255.255.0
          inet6 addr: fe80::27:50ff:fee0:37f0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric
             :1
          RX packets:30 errors:0 dropped:0 overruns:0 frame
             :0
          TX packets:100 errors:0 dropped:0 overruns:0
             carrier:0
          collisions:0 txqueuelen:100
          RX bytes:5972 (5.9 KB)  TX bytes:14834 (14.8 KB)

wlp2s0    Link encap:Ethernet  HWaddr 9c:b6:d0:8b:04:35
          inet addr:10.101.226.191  Bcast:10.101.231.255
             Mask:255.255.248.0
          inet6 addr: fe80::567e:542b:5019:4757/64 Scope:
             Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric
             :1
          RX packets:2113106 errors:0 dropped:0 overruns:0
             frame:0
          TX packets:878452 errors:0 dropped:0 overruns:0
             carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2819652667 (2.8 GB)  TX bytes:105426027
```

```
(105.4 MB)
```

Em relação à autenticação de rede e comunicação com o servidor IPBrick, executar o comando *wbinfo* permite obter informação acerca do estado da ligação e acerca de utilizadores e grupos existentes nesse servidor. Como demonstrado na Listagem 4.10, a execução dos comandos retorna a lista de utilizadores e grupos existentes no servidor IPBrick. Pode então afirmar-se que os mecanismos de autenticação em rede estão funcionais e que o requisito R10 foi cumprido.

Listagem 4.10: Resultados do comando wbinfo

```
~$ wbinfo -u
administrator
gustavo
jcorreia
mario
ncegonho
pessoa
tester_01
testvdi
testvpn
tiago
webservices
joinuser
mferri
sferreira
ipbtest
tcarrondo

~$ wbinfo -g
domain guests
print operators
backup operators
replicator
domain computers
domain admins
shell
vdi
developers
asolido
vpn
testgroup
```

Quanto à tolerância a faltas de ligação à *Internet*, os testes foram efetuados no controlador de testes que foi configurado com duas ligações WAN. As faltas foram simuladas desligando os cabos de rede das interfaces de rede.

A Listagem 4.11 demonstra o teste de falta na ligação primária. Como se pode observar,

quando o cabo é desligado, simulando uma falta, a ligação à *Internet* é perdida. Quando o mecanismo deteta a falta, redireciona todo o tráfego para a segunda ligação WAN, recuperando a ligação à *Internet*. O teste em que é simulada a falta da ligação secundária teve resultados idênticos. Em relação aos testes em que o controlador arranca com falta, todos obtiveram os resultados esperados, demonstrando o cumprimento do requisito R8.

Listagem 4.11: Teste a falta de Internet

```
~$ ping -D google.pt
PING google.pt (172.217.168.163) 56(84) bytes of data.

(.......)

[1528278236.732471] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=16 ttl=55 time=11.3 ms
[1528278237.734434] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=17 ttl=55 time=11.4 ms
[1528278238.735507] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=18 ttl=55 time=10.7 ms
[1528278239.736713] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=19 ttl=55 time=10.8 ms
[1528278240.739323] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=20 ttl=55 time=12.1 ms
[1528278241.740917] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=21 ttl=55 time=12.0 ms
[1528278242.741945] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=22 ttl=55 time=11.5 ms
[1528278243.744207] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=23 ttl=55 time=11.9 ms
[1528278244.746060] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=24 ttl=55 time=12.4 ms
[1528278291.741938] From 192.168.1.186 icmp_seq=67
   Destination Host Unreachable
[1528278291.745636] From 192.168.1.186 icmp_seq=68
   Destination Host Unreachable
[1528278291.745708] From 192.168.1.186 icmp_seq=69
   Destination Host Unreachable

(.......)

[
[1528278361.338082] From 192.168.1.186 icmp_seq=136
   Destination Host Unreachable
[1528278361.338219] From 192.168.1.186 icmp_seq=137
   Destination Host Unreachable
[1528278361.338277] From 192.168.1.186 icmp_seq=138
```

```
   Destination Host Unreachable
[1528278364.354252] From 192.168.1.186 icmp_seq=139
   Destination Host Unreachable
[1528278364.354395] From 192.168.1.186 icmp_seq=140
   Destination Host Unreachable
[1528278364.766115] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=142 ttl=55 time=10.6 ms
[1528278365.765889] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=143 ttl=55 time=12.2 ms
[1528278366.767520] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=144 ttl=55 time=12.1 ms
[1528278367.768453] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=145 ttl=55 time=11.5 ms
[1528278368.770830] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=146 ttl=55 time=12.0 ms
[1528278369.773600] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=147 ttl=55 time=13.2 ms
[1528278370.774769] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=148 ttl=55 time=12.6 ms
[1528278371.776413] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=149 ttl=55 time=12.0 ms
[1528278372.777133] 64 bytes from mad07s10-in-f3.1e100.net
   (172.217.168.163): icmp_seq=150 ttl=55 time=11.3 ms

(.......)

--- google.pt ping statistics ---
160 packets transmitted, 43 received, +74 errors, 73%
   packet loss, time 161085ms
rtt min/avg/max/mdev = 10.670/12.121/13.929/0.890 ms, pipe
   3
```

Atraso Induzido pelo Controlador

A estimação do atraso induzido pelo controlador, na propagação de pacotes entre as redes externa e interna foi realizada utilizando duas ferramentas. Foi capturada da execução do comando *ping* respetivamente iniciado num computador da rede interna e no controlador ao mesmo destinatário. Os comandos foram executados simultaneamente, tendo-se registado um acréscimo de 1.1ms no tempo médio de ida e volta apresentado pelo computador localizado na rede interna.

A segunda aproximação consistiu na observação dos mesmos pacotes nas interfaces de entrada e saída do controlador utilizando o comando *tcpdump* (Listagens 4.4 e 4.5) numa nova execução do comando *ping* de um computador na rede interna a uma máquina externa. Neste caso, registou-se uma diferença de tempo de 0.64ms entre a saída e a entrada dos

pacotes no controlador.

Considera-se que o valor calculado através da informação obtida pelas capturas do comando *tcpdump* é uma boa estimativa do atraso real induzido pelo controlador. O atraso estimado representa 5.5% da latência observada entre o computador interno e a máquina externa, sendo um valor considerado aceitável para um equipamento com estas características.

4.3 Colocação em Produção

No dia 19 de fevereiro foi colocado em funcionamento um protótipo do controlador de redes desenvolvido neste projeto. Este protótipo foi instalado num cliente da instituição acolhedora e esteve em funcionamento durante cerca de dois meses sem falhas. Posteriormente foi substituído por um controlador com uma versão final que se encontra em operação desde então, também sem falhas. Até à data o controlador instalado no cliente encontra-se em atividade há mais de 23 dias, já transmitiu cerca de 20GB, recebeu cerca de 123GB e atribuiu 64 *leases* DHCP diferentes.

A rede local das instalações da instituição acolhedora também usufrui de um controlador de redes com uma versão final. Inicialmente foi feita a instalação de uma versão protótipo, que esteve em execução mais de dois meses sem falhas. Posteriormente foram efetuadas instalações com versões mais recentes. De momento, o controlador mais recente encontra-se em operação há mais de 9 dias e já transmitiu cerca de 265GB e recebeu cerca de 550GB de informação. O servidor DHCP interno ao controlador já atribuiu 48 *leases* diferentes a dispositivos.

Capítulo 5

Conclusões e Trabalho Futuro

Os componentes que constituem uma rede de computadores são heterogéneos e desempenham papéis diferentes. Isto leva a uma complexidade acrescida na sua gestão e manutenção. Agregar diversos componentes da rede num único controlador de rede permite alguma centralização de serviços, facilitando a gestão. As versões comerciais dos controladores de rede têm um custo elevado e utilizam *software* proprietário. A necessidade de criar uma solução de custo moderado baseada em código aberto que permita reduzir a complexidade da gestão e manutenção de uma rede, criando uma alternativa à predominância do mercado por parte de equipamentos proprietários levou ao desenvolvimento deste projeto.

Este projeto teve como objetivo o desenvolvimento de uma solução de controlador de redes chave na mão, de custo moderado e baseada em tecnologias de código aberto, respondendo assim às necessidades identificadas pela Ângulo Sólido, que presta serviços informáticos na área de administração de sistemas e é especializada em tecnologias de código aberto.

Prentendeu-se com esta solução disponibilizar aos clientes um controlador de redes com uma instalação e configuração automáticas, com custo moderado, atendendo às suas necessidades específicas e não necessitando de um perito para efetuar a sua instalação.

O desenvolvimento da solução iniciou-se com uma pesquisa e eleição dos componentes físicos, filtrando diferentes opções encontradas no mercado por determinadas características relevantes ao âmbito do projeto e pela seleção da distribuição Linux a utilizar.

Foram desenvolvidos *scripts* para que a instalação e a configuração dos pacotes de *software* necessários sejam efetuadas de forma automática. Esta instalação é específica de cada cliente pois os *scripts* de instalação são alimentados por um ficheiro com informação referente à rede do cliente em questão.

Para a avaliação dos resultados, foram realizadas diversas instalações, com configurações diferentes e em ambiente de testes. Foram ainda instalados dois controladores em produção, um nas instalações de um cliente e outro nas instalações da instituição acolhedora. Ambos estiveram em execução sem problemas e já foram substituídos por versões mais recentes,

perfazendo um total de seis instalações em ambiente real. Até à data, os controladores encontram-se em funcionamento sem incidentes a reportar.

Numa perspectiva futura, seria interessante abordar a centralização da gestão dos pontos de acesso sem fios. De momento, cada ponto de acesso tem de ser configurado manualmente, o que para empresas pequenas não se revela um problema. Tendo em conta o tipo de clientes da instituição acolhedora, a configuração individual de pontos de acesso sem fios não representa um acréscimo considerável na complexidade do procedimento de instalação do controlador. No entanto, num cliente com instalações de maior dimensão, a configuração manual de um número superior de pontos de acesso sem fios eleva a probabilidade de erro e acrescenta complexidade de gestão.

Bibliografia

[1] Massimo Bernaschi, Filippo Cacace, Giulio Iannello, Massimo Vellucci, and Luca Vollero. Opencapwap: An open source capwap implementation for the management and configuration of wifi hot-spots. *Computer Networks*, 53(2):217–230, 2009.

[2] Jesper Dangaard Brouer. Optimization of tcp/ip traffic across shared adsl. *Master's thesis, University of Copenhagen*, 2005.

[3] Cisco. Cisco 8540 wireless controller. `https://www.cisco.com/c/en/us/products/collateral/wireless/8540-wireless-controller/datasheet-c78-734258.pdf`, 2017. Document ID: 3909a78d-be28-488c-885a-7905820abd09 (consultado em: 16/10/2017).

[4] Bufferbloat community. Bufferbloat site. `https://www.bufferbloat.net/projects/`. (consultado em: 27/02/2018).

[5] Bufferbloat community. Bufferbloat site. `https://www.bufferbloat.net/projects/codel/wiki/Cake/`. (consultado em: 27/02/2018).

[6] CZ.NIC. Turris omnia. `https://omnia.turris.cz/en/`. (consultado em: 23/04/2018).

[7] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on networking*, 1(4):397–413, 1993.

[8] IPBbrick. Ipbrick. `https://www.ipbrick.com/`. (consultado em: 26/04/2018).

[9] James F Kurose. *Computer networking: A top-down approach featuring the internet, 3/E*. Pearson Education India, 2005.

[10] Canonical Ltd. Ubuntu long term support. `https://wiki.ubuntu.com/LTS`. (consultado em: 25/10/2017).

[11] Canonical Ltd. Ubuntu recommended minimum requirements. `https://help.ubuntu.com/lts/serverguide/preparing-to-install.html`. (consultado em: 25/10/2017).

[12] MikroTik. Mikrotik. `https://mikrotik.com/`. (consultado em: 23/04/2018).

[13] V Preethi Rao, Mohit P Tahiliani, and Udaya Kumar K Shenoy. Analysis of sfqcodel for active queue management. In *Applications of Digital Information and Web Technologies (ICADIWT), 2014 Fifth International Conference on the*, pages 262–267. IEEE, 2014.

[14] RedHat. Centos recommended minimum requirements. `https://wiki.centos.org/About/Product`. (consultado em: 25/10/2017).

[15] Tanvi Sharma. Controlling queue delay (codel) to counter the bufferbloat problem in internet. *Int. J. Curr. Eng. Technol*, 4(3):2210–2215, 2014.

[16] Dan Siemon. Queueing in the linux network stack. `https://www.coverfire.com/articles/queueing-in-the-linux-network-stack/`. (consultado em: 28/02/2018).

[17] CACM Staff. Bufferbloat: What's wrong with the internet? *Communications of the ACM*, 55(2):40–47, 2012.

[18] Kevin Thompson, Gregory J Miller, and Rick Wilder. Wide-area internet traffic patterns and characteristics. *IEEE network*, 11(6):10–23, 1997.

[19] Vodafone. Vodafone. `https://www.vodafone.pt/main/particulares/tv-net-voz/internet/`. (consultado em: 07/05/2018).