INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Departamento de Engenharia de
Eletrónica e Telecomunicações e de Computadores

Benchmarking de uma aproximação de Ticketing as a Service (Taas)

João Nuno Rosa Eleutério Silva

Dissertação de natureza científica realizada para obtenção do grau de
Mestre em Engenharia de Redes de Computadores e Multimédia

Orientador

Professor Adjunto Doutor João Carlos Amaro Ferreira

Júri

Presidente:	Professor Adjunto Doutor Paulo Manuel Trigo Cândido da Silva
Vogais:	Professor Adjunto Mestre Paulo Alexandre Medeiros de Araújo
	Professor Adjunto Doutor João Carlos Amaro Ferreira

Outubro de 2013

Resumo

O trabalho desenvolvido encontra-se integrado no projeto SmartCITIES Cloud Ticketing da empresa Link Consulting SA (Link), no qual se pretende agilizar o processo de desenvolvimento de soluções de bilhética com a passagem da lógica de negócio dos terminais de bilhética para a Nuvem computacional, através da introdução do conceito "thin device".

Atualmente os utentes dos serviços de transporte das grandes cidades necessitam de possuir consigo diversos tipos de cartões para poder usufruir dos vários transportes nas suas deslocações diárias. Com a arquitetura proposta no projeto SmartCITIES será possível aos utilizadores possuírem um único cartão (*smartcard*) contendo títulos de vários operadores, provendo a versatilidade e interoperabilidade neste tipo de serviço.

Esta migração de processos levantou questões ao nível da latência e da segurança das comunicações. Assim, nesta dissertação propôs-se o desenvolvimento de um protótipo de um serviço de comunicação entre os terminais de bilhética presentes nas estruturas de transporte e um serviço central de *backoffice* implementado na Amazon Web Services.

Esta plataforma foi desenvolvida com o objetivo de testar o desempenho das operações de bilhética habituais (e.g., carregamento, validação e venda de bilhetes), analisando-se a latência em função dos diversos níveis de segurança aplicados, rede disponível ou topologia de Nuvem em causa. Foram recolhidas métricas associadas ao round-trip time (RTT) das comunicações efetuadas por este protótipo, por forma a determinar as situações que o conceito de "thin device" se possa aplicar.

Os resultados obtidos apontam no sentido da possibilidade efetiva de migração de sistemas de bilhética eletrónica dotados da arquitetura presente no projeto SmartCITIES Cloud Ticketing para a Nuvem, dependendo a sua aplicabilidade da dimensão e características dos operadores que queiram usufruir deste serviço.

Palavras-chave: bilhética eletrónica, Nuvem, projeto SmartCITIES, latência, interoperabilidade, segurança, migração de serviços, lógica de negócio, "thin devices".

Abstract

This work is part of the SmartCITIES Cloud Ticketing project developed by Link Consulting SA (Link). The project main goal is to streamline the development process of e-ticketing solutions by transferring the business logic located on the ticketing terminals to the Cloud by implementing a new concept – thin device.

Nowadays, big cities transportation services users are required to own several kinds of smartcards to travel across different types of transportation services in their everyday life. The architecture proposed in SmartCITIES project aims to allow users to own a single smartcard containing all the required information to travel across different operator's domains while providing versatility and interoperability.

This process migration raised some issues, namely the latency and security of the communications. With this in mind, in this paper the work was developed in the direction of implementing a prototype of a communication ticketing service (Web Service – SmartSales) serving the communications between the terminals located on the transportation infrastructures and the end central office or backoffice on Amazon Web Services.

This platform was developed to benchmark the normal ticketing operations (e.g., sale, recharge and validation), while providing insights on the latency, according to several levels of security, available network and Cloud topology. Several metrics associated with the round-trip time (RTT) of communications made by the prototype were collected to explore possible situations where the "thin device" concept can be adapted.

The acquired results point towards the possible migration of e-ticketing systems with the SmartCITIES Cloud Ticketing project architecture to the Cloud, depending always on the operator´s characteristics that want to join the project.

Keywords: e-ticketing, Cloud Computing, SmartCITIES project, latency, interoperability, security, service migration, business logic, "thin devices".

Agradecimentos

Ao professor Doutor João Ferreira, por ter assumido a orientação deste trabalho, pela colaboração, incentivo e apoio que me facultou e pela enorme disponibilidade e profissionalismo que sempre demostrou para discutir alternativas e soluções.

Aos profissionais da Link Consulting SA, nomeadamente os engenheiros Gonçalo Cunha e João Silva, pela presença nas muitas reuniões efetuadas e pela paciência demonstrada na transferência de conhecimentos e pareces técnicos complexos.

Ao ISEL – Instituto Superior de Engenharia de Lisboa por ter ultrapassado largamente as espectativas iniciais e por fornecer um ensino de qualidade e excelência. Queria destacar os docentes e Engenheiros: João Ferreira, Manuel Barata, Paulo Trigo, Pedro Ribeiro, Vítor Almeida e Luís Morgado pelos elevados conhecimentos demonstrados, qualidade das aulas e matérias lecionadas, mas sobretudo pela parte humana e pelo sentido de ajuda muitas vezes ultrapassando o carácter de profissional de educação.

Aos meus amigos, especialmente o Nuno Rosa pelas discussões, opiniões e sugestões que comigo partilhou.

À minha família pelo apoio durante toda a minha vida. Uma menção especial para os meus pais e avós que sempre fizeram o possível e o impossível para que eu pudesse prosseguir os meus estudos e que a realização desta etapa da minha vida os encha de orgulho. E claro aos meus irmãos, por me encherem de felicidade e preencherem uma parte essencial da minha vida.

Finalmente à minha adorada companheira que sempre me apoiou nos bons e maus momentos, pela compreensão e paciência durante esta importante etapa das nossas vidas e pelo encorajamento fornecido para abraçar este desafio.

Índice Geral

Índice de Figuras

Lista de Tabelas

Glossário

ACL	Access Control List
AMI	Amazon Machine Images
AMS	Asset Management Systems
ASN.1	Abstract Syntax Notation 1
AWS	Amazon Web Services
CDN	Content Delivery Networks
CLI	Command Line Interface
CM	Customer Media
CPU	Central Processing Unit
CSP	Cloud Services Providers
DDOS	Distributed Denial of Service
DOS	Denial of Service
EBS	Elastic Block Store
Ec2	Elastic Compute Cloud
EEPRO	Electrically Erasable Programmable Read-Only Memory
EIP	Elastic IP
ELB	Elastic Load Balancing
FLC	Fuzzy Logic Controller
HMAC	Hash Message Authentication Code
HOPS	ITSO BackOffice
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure as a Service
IDS	Intrusion Detection Systems
IFM	Interoperable Fare Management
IIN	International Issue Number
IIS7	Internet Information Services 7
IOPS	Input/Output Operations Per Second
IPE	ITSO Product Entities
ISAM	ITSO Secure Application Module
ISMS	ITSO Security Management Service

ISRN	ITSO Shell Environment Identification Number
ITSO	Integrated Transport Smartcard Organization
MD	Message Digest
NAS	Network-Attached Storage
NAT	Network Address Translation
PaaS	Platform as a Service
PAT	Port Address Translation
POST	Point of Service Terminal
QoS	Quality of Service
RDP	Remote Desktop Protocol
REST	Representational State Transfer
RTT	Round-Trip Time
S3	Simple Storage Service
SaaS	Software as a Service
SAM	Security Application Module
SAN	Storage Area Network
SLA	Service Level Agreement
SLB	Server Load Balancing
SOAP	Simple Object Access Protocol
SPI	Stateful Packet Inspection
SSH	Secure Shell
SSS	Security Subsystem
TTR	Transient Ticket Records
VPC	Virtual Private Cloud
VPN	Virtual Private Network
WCF	Windows Communication Foundation
XML	Extensible Markup Language

1. Introdução

Atualmente nas grandes cidades as pessoas movimentam-se em massa para realizar as suas atividades quotidianas (trabalho, abastecimento, educação, saúde, cultura, recreação e lazer), num tempo considerado ideal, de modo confortável e seguro.

Neste contexto de mobilidade urbana os equipamentos de transporte individual ou coletivo de passageiros assumem, cada vez mais, um papel determinante na melhoria da mobilidade nas grandes zonas metropolitanas e urbanas, melhoria essa que visa garantir um melhor acesso das pessoas ao que a cidade oferece, tornando-o mais eficiente em termos socioeconómicos e ambientais.

Esta realidade coloca os operadores de transporte sobre uma crescente pressão para responder às necessidades dos seus clientes e às obrigações impostas pelos reguladores, de forma a oferecer serviços mais convenientes e a preços mais acessíveis.

Assim portanto, qualquer análise do tema não pode abdicar de discutir as questões de ordem tecnológica que envolvem os meios de transporte e que potenciam o conceito de mobilidade. Um dos aspetos que se vem impondo como fundamental para a qualidade global e facilidade de acesso aos serviços é a existência de modernos sistemas de bilhética, baseados em tecnologia eletrónica *contactless*. Este tipo de serviço apresenta-se como uma sólida alternativa ao sistema convencional de validação de títulos de transporte (em papel), ao transferir toda a informação necessária para um dispositivo eletrónico (*smartcard*).

A diferença fundamental na arquitetura entre os sistemas convencionais e os sistemas de bilhética eletrónica prende-se com a introdução de um novo modelo de negócio, onde as funções dos fornecedores de serviços e emissores de títulos são completamente distintos [Caulfield, et al., 2007].O fornecedor de serviços apresenta-se como a entidade que fornece o serviço ao qual o título se destina, enquanto a entidade que fornece o título é a organização que fornece a emissão do mesmo, sendo ainda responsável pela sua credibilidade. A função de recolha e inspeção são exclusivas da entidade fornecedora do serviço, usando equipamentos e infraestruturas fornecidas pela entidade emissora. Este modelo comtempla ainda um terceiro ator como uma instituição financeira, autorizada

ao processamento de pagamentos. O quarto e último ator é o cliente que utiliza e usufrui do título. Um exemplo desta arquitetura é ilustrado na Figura 1.

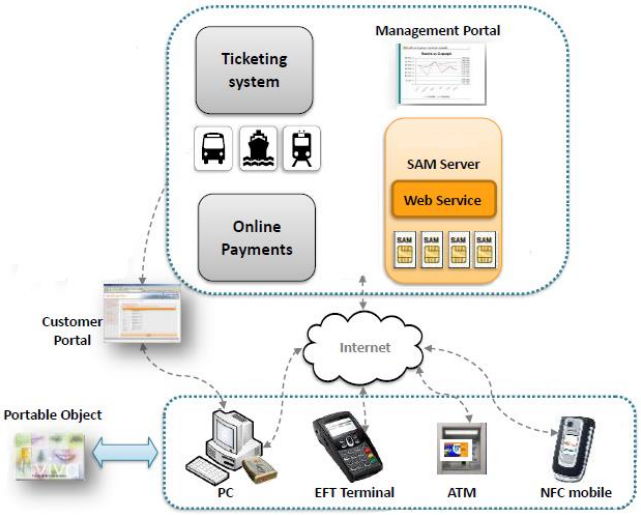

Figura 1 - Arquitetura de um sistema de bilhética eletrónico

A arquitetura descrita compreende um conjunto de tecnologias e normas que satisfazem os requerimentos de cada entidade envolvida no modelo, fornecendo as funcionalidades e conectividades necessárias à realização de tarefas de forma interoperável. O utilizador pode aceder ao sistema quer através de rede fixa ou sem fios. Ambos necessitam de fornecer uma transmissão segura de dados, principalmente nas transações relativas a pagamentos ou a certas operações de bilhética, cujos dados necessitem de permanecer privados. Este acesso é realizado através de um dispositivo eletrónico capaz de armazenar os dados do título eletrónico, podendo ser um *smartcard*, PDA ou dispositivo móvel.

A entidade emissora do título necessita de possuir uma infraestrutura, compreendendo desde servidores, *firewalls*, entre outros, para mitigar e reduzir vulnerabilidades. É necessário um *Web Server* devidamente configurado, para receber ligações seguras, requeridas para que o serviço se encontre disponível, bem como, um servidor dotado de um Security Application Module (**SAM**), armazenando todos os atributos criptográficos. A persistência de dados é garantida por um servidor de dados e o módulo de pagamento é mantido num componente separado devido aos diferentes requisitos das interfaces para os diferentes sistemas bancários [Rehrl, et al., 2007]. O fornecedor do serviço necessitará de um dispositivo móvel, utilizado por um colaborador para

inspeção de títulos, com capacidade de receber a informação do dispositivo eletrónico do utilizador e de interagir com o servidor aplicacional da entidade emissora, para verificar e atualizar os dados relativos aos títulos.

1.1 Projeto SmartCITIES Cloud Ticketing Services

A empresa Link encontra-se a desenvolver um projeto QREN em parceria com o ISEL, no qual este projeto se encontra integrado, onde se está a estudar uma aproximação inovadora na área da bilhética eletrónica. A arquitetura do sistema de bilhética eletrónica proposta no projeto SmartCITIES, está dividida em vários serviços tal como se pode verificar na Figura 2. Cada serviço funciona como se fosse uma camada e pode ser acedido diretamente por dispositivos e/ou por outros serviços.

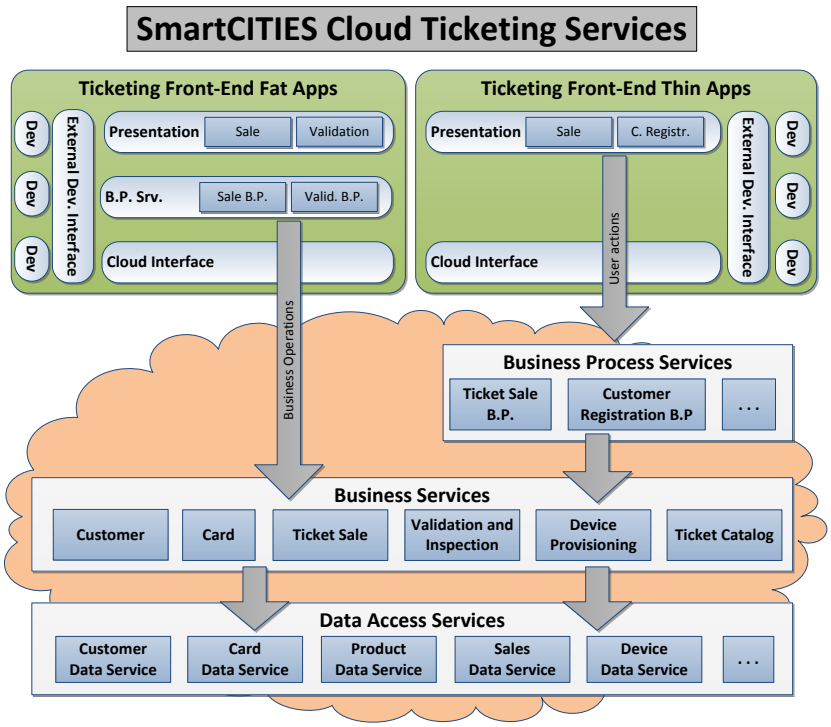

Figura 2 - Arquitetura do projeto SmartCITIES Cloud Ticketing Services

No projeto referido na Figura 2 desenvolveu-se o conceito de "thin device", no qual foi separado o processo da lógica de negócio do terminal para a Nuvem Computacional. Este conceito, contrasta com o tradicional "fat device" onde esta mesma lógica de negócio se encontra nos terminais.

Pretende-se assim possuir uma maior agilidade no processo de desenvolvimento de soluções de bilhética, com o objetivo de facilitar a integração de vários operadores de diferentes transportes, de vários pontos de venda e com um vasto número de dispositivos e, ainda, usufruir-se da elasticidade ao nível dos recursos da infraestrutura central/*backoffice* que um ambiente em Nuvem pode oferecer.

Esta integração é possível utilizando um conjunto de serviços alojados na Nuvem que seriam comuns a todos os operadores e dispositivos. Na Figura 2 é possível observar-se os dois tipos de dispositivos, "fat devices" e "thin devices", e a sua responsabilidade na arquitetura que está dividida em três camadas.

A primeira camada é a de apresentação e está presente nos dois tipos de dispositivos. A segunda camada contém a lógica de negócio e está apenas presente nos dispositivos do tipo "fat devices", interagindo diretamente com a camada *Business Services* que se encontra na Nuvem. Por sua vez, os dispositivos do tipo "thin devices" não têm lógica de negócio local, pois a lógica de negócio que estes dipositivos utilizam encontra-se armazenada na Nuvem e é disponibilizada pelos *Business Process Services*. A última camada permite o acesso a dados, e está disponível através dos *Data Access Services* que estão presente na Nuvem.

Esta arquitetura, pode ser utilizada em simultâneo, tanto por "thin devices" como por "fat devices", o que permite que continuem a ser utilizados dispositivos do tipo "fat device" em condições onde, eventualmente, não possam ser utilizados dispositivos do tipo "thin device".

Os *Data Access Services* permitem que sejam acedidos e manipulados os dados que contêm, por exemplo, as informações sobre os bilhetes vendidos, ou as informações sobre um dado cliente.

Os *Business Services* permitem realizar algumas operações, como listar o catálogo de títulos existentes, carregar um título ou, ainda, inserir os dados de um novo cliente.

Os *Business Process Services* coordenam a utilização entre vários *Business Services*, para implementar casos de utilização. Um exemplo de um caso de utilização pode ser a venda de um título que envolveria os seguintes seis passos:

- Ler o cartão;

- Procurar no catálogo todos os títulos disponíveis;
- Escolher o título que se pretende comprar;
- Efetuar o pagamento**;**
- Carregar o cartão;
- Confirmar e registar a venda.

1.2 Motivação

Pelo atrás exposto, torna-se evidente que a comunicação entre "thin/fat devices" e a infraestrutura central pertencente à entidade emissora de bilhetes, apresenta-se como um ponto fulcral no projeto SmartCITIES. Esta infraestrutura poderá estar sediada numa rede privada nas instalações da entidade emissora, num Centro de Dados alugado ou num ambiente de Nuvem. Estas são normalmente as etapas efetuadas pelas infraestruturas das empresas no que à localização diz respeito. Ao longo dos anos, e impulsionadas pelos crescentes estudos efetuados, tende-se a verificar um crescente aumento das migrações para ambientes Nuvem, devido às muitas vantagens associadas.

De facto, os últimos anos foram profícuos no aparecimento de diversas ideias e tecnologias para a otimização dos processos de Nuvem computacional já existentes, ou mesmo, de outros processos distintos, todos na procura da otimização da estratégia de comunicação neste tipo de metodologia e, de todos os seus elementos derivados como a performance da rede interna dos Cloud Services Providers (**CSP**), a elasticidade computacional dos mesmos ou o armazenamento persistente [Birman, et al., 2009]. Estes factos tornam a migração para este tipo de infraestrutura cada vez mais plausível.

Ao abordar a migração de um qualquer sistema já implementado e funcional para um ambiente com diferentes características como é o caso da Nuvem, é necessário alguma ponderação sobre certos fatores, como a segurança e o desempenho [Amazon Web Services, 2010]. O que no passado estava confinado à segurança e administração da própria entidade passa a ser englobado num ambiente completamente aberto como a Internet, tornando-se assim necessário o estudo prévio duma migração deste tipo. Esta migração traz consigo inerentes vantagens e desvantagens: se por um lado existe uma redução de despesas e complexidade, aumento de escalabilidade e redundância, por

outro existe uma sensação de perda de controlo, sobretudo sobre a segurança [Mather, et al., 2009].

Evidenciou-se então a necessidade de avaliar o enquadramento das diferentes infraestruturas de um sistema de bilhética, as funções que possuem no serviço de bilhética ou o papel que desempenham na comunicação. Tornou-se assim claro que, para a concretização de testes experimentais sobre esta matéria, era necessária uma abordagem de divisão das várias infraestruturas de bilhética (terminais), operações efetuadas, segurança subjacente a cada situação, bem como o respetivo ambiente de rede onde tais infraestruturas se encontram inseridas.

Foram então desenvolvidos objetivos claros para que se possa encontrar uma solução para que a mudança de paradigma apresentada se torne uma realidade, cumprindo todos os requisitos previstos, e recorrendo à utilização das tecnologias emergentes sobre esta matéria.

1.3 Objetivo

O presente trabalho enquadra-se na área da bilhética eletrónica, sendo parte de um projeto real, nomeadamente o projeto SmartCITIES, e pretende desenvolver uma plataforma de testes para analisar e monitorizar o desempenho (Round-Trip Time – **RTT**) das comunicações entre "thin devices" e uma infraestrutura central sediada na Nuvem da Amazon Web Services (**AWS**), consoante a topologia de rede projetada e respetiva segurança subjacente, com o objetivo de inferir sobre as condições de comunicações e segurança onde o conceito do "thin device" se possa aplicar.

Para desenvolver uma plataforma de testes tão realista quanto possível, será implementado um protocolo comunicacional e desenvolvido um *Web Service* baseado nos protocolos utilizados pelas empresas de bilhética atualmente, que disponibilizará um conjunto de serviços, englobando várias operações de bilhética, nomeadamente operações de validação, recarregamento e autenticação.

O *Web Service* será publicado na AWS, onde se construirá uma topologia de rede adequada ao projeto. Ao nível da AWS, será contratado um serviço de Infrastructure as

a Service (**IaaS**), para se poder ter acesso, não apenas ao *Web Service* proposto, mas também, deter o controlo de toda a infraestrutura subjacente. Serão realizados testes em diferentes cenários, correspondendo a vários tipos de topologias de rede, cada qual com as suas características específicas de segurança, que poderão afetar o desempenho do sistema.

A implementação do *Web Service* proposto irá ser realizada num formato o mais genérico possível, por forma a garantir a independência deste com qualquer CSP.

1.4 Organização da Dissertação

Para além deste capítulo de carácter introdutório onde, entre outros, são definidos os objetivos e realizada uma introdução ao projeto SmartCITIES, esta dissertação encontra-se organizada em mais seis capítulos, num total de sete:

Capítulo 2: Projeto SmartCITIES
Neste capítulo é descrito o projeto SmartCITIES e os temas relacionados da Nuvem Computacional, dos sistemas de bilhética eletrónicos e dos serviços e funcionalidades que influenciaram o desenvolvimento deste trabalho.

Capítulo 3: Contribuições para o projeto SmartCITIES
Capítulo onde se ilustra o trabalho de investigação desenvolvido no âmbito da entrega de vários *delivers* para o projeto SmartCITIES, nomeadamente o levantamento das normas e especificações dos sistemas eletrónicos de bilhética, relacionadas com a interoperabilidade e segurança entre entidades bilhéticas.

Capítulo 4: Modelação dos Sistemas de Bilhética Eletrónicos
Neste capítulo é realizada a modulação dos sistemas de bilhética, para se poder criar uma plataforma de teste, que possa simular o problema real dos sistemas. São identificadas as principais características das operações existentes, e definidas as principais classes de terminais de bilhética e as diferentes condições em que podem operar. Como resultado, foi criado um conjunto de serviços modulares que permitem simular as principais operações de bilhética. Assim, neste capítulo foi modelado o

problema consoante as variáveis identificadas, e desenvolvido o *Web Service* (SmartSales) para o efeito.

Capítulo 5: Plataforma de Teste

Neste capítulo é descrita a plataforma de teste desenvolvida na infraestrutura da AWS, onde o *Web Service* SmartSales pudesse ser publicado. São definidos diversos cenários de teste, tendo em conta as topologias de rede e tipos de Nuvem.

Capítulo 6: Resultados

Capítulo onde são apresentados os resultados de desempenho da plataforma de testes e efetuada uma análise comparativa entre os diversos fatores envolvidos nestes, para que se pudesse inferir o desempenho do conceito de 'thin device' aplicado a diferentes cenários.

Capítulo 7: Conclusões

Capítulo de carácter conclusivo sobre o trabalho efetuado e onde se propõe uma possível linha de desenvolvimento para futuros trabalhos.

1.5 Convenção de Escrita

No sentido de facilitar a leitura ao longo da dissertação, são utilizadas as seguintes convenções de escrita:

Esta dissertação foi redigida segundo o novo acordo ortográfico da língua portuguesa e emprega-se no texto normal a fonte Times New Roman;

- Os termos ou expressões sobre os quais serão apenas utilizadas abreviaturas em ocorrências posteriores, surgem na sua primeira ocorrência no texto, referidos por extenso seguidos da abreviação, entre parêntesis;
- Emprega-se texto a negrito para a definição de abreviaturas, como para Network Address Translation (**NAT**);
- Optou-se por não traduzir alguns termos de língua inglesa de utilização generalizada, como hardware, software ou microchip;
- O grafismo em itálico é utilizado para as expressões latinas como *a priori*;

- Surgem igualmente com grafismo em itálico termos ou designações originários da língua inglesa como *hardware*, *software* ou *microchip*;
- As referências bibliográficas são referidas por parêntesis curvos contendo o primeiro autor e ano da publicação, como por exemplo [Mather, et al., 2009].

2. Projeto SmartCITIES

Como foi referido na secção 1.1, o projeto SmartCITIES é um projeto da Link em Parceria com o Instituto Superior de Engenharia de Lisboa (ISEL) a decorrer desde 2012. No âmbito desse projeto decorreram ou, estão a decorrer, os seguintes projetos de Mestrado:

1. **Estudo do Paradigma Computação em Nuvem**, realizado pela aluna Carina Gomes e cujo objetivo foi o desenvolvimento de uma arquitetura para integrar equipamentos de bilhética empregues para, designadamente, vender, validar e fiscalizar títulos de transportes. Para avaliar a arquitetura proposta foi implementado um demonstrador na plataforma Windows Azure. Trabalho concluído.

2. **Android as a Cloud Ticket Validator**, realizado pelo aluno Agostinho da Cunha Baía, cujo objetivo foi, numa plataforma móvel com sistema operativo Android, implementar um validador de baixo custo para a bilhética de transportes coletivos de passageiros. Foi proposta a migração dos terminais de validação tradicionais para uma nova abordagem implementada num *tablet* com o sistema operativo Android, cuja lógica de validação se encontrava alojada numa Nuvem computacional (Amazon Web Services).

3. **Arquitetura para Integração de Dispositivos na Nuvem**, Nuno Maurício, cujo objetivo é a implementação do conceito de "thin device" numa logica de SaaS, usando a plataforma móvel Android para o lado do terminal, para interação com nuvens computacionais (e. g. Windows Azure, Google App Engine ou Amazon Web Services).

E os seguintes projetos finais de curso:

4. **Mobile Ticketing System,** dos alunos Nelson Matias e Nelson Almendra, cujo objetivo é transformar o dispositivo móvel de cada utilizador num bilhete de transporte público. Para tal foi desenvolvida uma aplicação em Android que permite ao utilizador adquirir bilhetes usando códigos de barras nas paragens.

5. **SmartCards**, dos alunos José Matias e João Rebelo, cujo objetivo é virtualizar o acesso de um terminal ao respetivo SAM, através da criação de um sistema de

serviços acessível via *Web,* que permita a comunicação segura sem necessidade dos equipamentos estarem fisicamente ligados.

Face aos trabalhos existentes, o presente trabalho pretende completar o projeto SmartCITIES, desenvolvendo uma plataforma de testes das comunicações dos terminais para a Nuvem, usando os serviços da AWS.

Como grande parte do projeto SmartCITIES apresenta a sua sustentabilidade na Nuvem Computacional, optou-se neste capítulo, para facilidade de compreensão da implementação da plataforma de testes e para realizar um enquadramento a esta temática, por fazer uma separação clara e distinta entre os vários tipos de serviços de Nuvem, respetivos modelos, tecnologias existentes e segurança (riscos/ameaças) associada.

2.1 Nuvem Computacional

O paradigma da computação em Nuvem, originalmente referido em inglês como *Cloud Computing*, encontra-se associado a um modelo de gestão de recursos computacionais. Neste modelo, os recursos são distribuídos para reduzir os custos associados à manutenção de *hardware* e *software* [Deelman, et al., 2008]. O seu crescente sucesso deve-se, em parte ao aparecimento de serviços com características comuns, disponibilizados por empresas tecnológicas líderes de mercado. Contudo, muitos dos conceitos em que esta se baseia (como a virtualização, a computação distribuída e até mesmo a partilha de recursos) não são novos [Mather, et al., 2009].

A Nuvem apresenta interesse comercial, não só, devido à crescente tendência em *outsourcing* de Tecnologias de Informação (**TI**), mas também, como forma de minimizar recursos humanos na gestão/manutenção, garantindo simultaneamente a extensão dos recursos limitados das infraestruturas das TI. Outro aspeto importante que importa considerar é o custo inicial ser mínimo, o que facilita a entrada de fornecedores de serviços que oferecerem novas capacidades, essenciais para satisfazer as necessidades impostas por um mercado amplo e dinâmico.

2.2 Definição e Tipos de Computação em Nuvem

A variedade de tecnologias que integram a Nuvem torna difícil adotar uma definição [Vaquero, et al., 2009]. Adicionalmente, a publicidade ao modelo de negócio sugere uma noção extremamente generalista. Neste âmbito, a noção de Nuvem é referida em quase todos os tipos de solução que facilitem a partilha de recursos computacionais. No entanto, a definição mais comum refere o acesso a recursos/serviços pagos por utilização. Referido como "*pay-per-use*" com elasticidade ilimitada e instantânea.

Existem muitos cenários de utilização de computação em Nuvem, mas todos se parecem focar em determinados aspetos tecnológicos [Mell, et al., 2009], [Vaquero, et al., 2009], [Mather, et al., 2009]. A noção de Nuvem está diretamente relacionada com o fornecimento do serviço oferecido/pago, designadamente: IaaS; Platform as a Service (**PaaS**), Software as a Service (**SaaS**), ou seja, genericamente, Everything as a Service (**EaaS**). Dependendo do tipo de recursos requeridos, destacam-se:

- **Iaas** - Corresponde ao nível mais baixo, responsável por fornecer serviços de suporte de infraestruturas. IaaS refere-se à partilha dos recursos de *hardware* para serviços de execução, geralmente usando a tecnologia de virtualização. Os recursos podem facilmente ser aumentados dependendo das necessidades. Tipicamente, os recursos são máquinas virtuais que têm associadas uma gestão para controlar a criação e manutenção, evitando igualmente o acesso não controlado à informação;
- **Paas** - Apresenta-se como o nível intermédio que oferece serviços de plataforma, e que proporciona um ambiente para alojamento de aplicações específicas. Não existe uma distinção clara entre PaaS e IaaS, sendo por isso frequente a mesma empresa fornecer os dois. PaaS suporta ferramentas para criar interfaces com base em HyperText Markup Language (**HTML**) e JavaScript. Suporta de igual forma, o desenvolvimento de *Web Services* acedidos via protocolos Simple Object Access Protocol (**SOAP**) e Representational State Transfer (**REST**). Adicionalmente, existe suporte para base de dados e outros serviços [Rittinghouse, et al., 2009]. Outros sistemas interagem com a Nuvem através da internet, usando mensagens SOAP,

codificadas em Extensible Markup Language (**XML**), tipicamente transmitidas sobre Hypertext Transfer Protocol (**HTTP**);

- **SaaS** – Apresenta-se como o nível superior que disponibiliza acesso a uma aplicação completa oferecida/paga como serviço [Velte, et al., 2009]. SaaS garante o alojamento completo na Nuvem de aplicações, eliminando a necessidade de instalar e executar a aplicação localmente no computador ou num servidor local, o que reduz os custos de manutenção de *software*. São essencialmente as PME (Pequenas e Médias Empresas) os principais interessados nesta abordagem, já que dificilmente uma PME terá capacidade financeira para efetuar um investimento inicial para aplicações do tipo SAP ou ORACLE. SaaS poderá ser a solução para o acesso a componentes de *software* outrora inalcançáveis.

Atualmente, a Nuvem é vista como uma tendência emergente adotada por uma quantidade crescente de organizações, que tentam obter os benefícios prometidos. Neste cenário, identificam-se vantagens, nomeadamente: (1) redução de *hardware* e consequentemente de custo de manutenção; (2) acesso a partir de qualquer local e (3) flexibilidade e escalabilidade num processo altamente automatizado [Mather, et al., 2009].

2.3 Papéis

Os papéis são importantes para definir responsabilidades, acesso e perfil para os diferentes atores que fazem parte e estão envolvidos numa solução de Nuvem. Para uma melhor descrição, podem-se classificar os atores dos modelos de acordo com os papéis desempenhados [Marinos, et al., 2009], como ilustra a Figura 3. O fornecedor do serviço é responsável por disponibilizar, gerir e monitorizar a infraestrutura que suporta a solução, deixando o parceiro tecnológico (proprietário da solução, neste projeto a entidade bilhética) e o utilizador final sem qualquer responsabilidade.

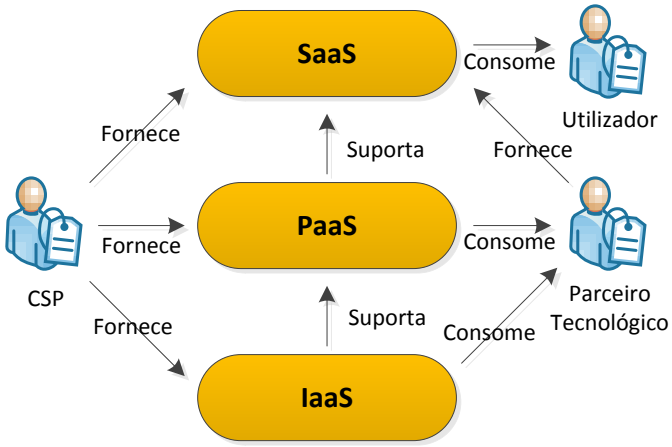

Figura 3 - Atores e papéis na Nuvem

Os parceiros tecnológicos utilizam os recursos disponibilizados para suportarem as aplicações acedidas pelos utilizadores finais (operadores de transporte). Esta organização em papéis ajuda a definir os atores e os seus diferentes interesses. Os atores podem assumir vários papéis ao mesmo tempo de acordo com os seus interesses. No entanto, só o fornecedor é que faculta suporte a todos os modelos de serviços.

2.4 Modelos de Computação em Nuvem

Existem diferentes modelos de instalação de ambientes de Nuvem, designadamente: pública, privada, comunitária e híbrida [Mell, et al., 2009]. A restrição ou abertura de acesso depende do negócio, do tipo de informação e do nível de abstração. Neste âmbito, admite-se que certas empresas desejem controlar o acesso dos utilizadores a determinados recursos. Por isso, coexiste a necessidade de contemplar ambientes mais restritos, onde apenas alguns utilizadores, devidamente autorizados, podem aceder a determinados serviços.

2.4.1 **Nuvem privada**

No modelo de Nuvem privada, a infraestrutura é utilizada exclusivamente por uma organização e posicionada dentro de um ponto de controlo de acessos gerido pelo proprietário, sendo esta Nuvem local ou remota, administrada pela empresa. Neste cenário, são adotadas políticas de acesso aos serviços. As técnicas utilizadas são,

designadamente: (1) gestão das redes; (2) configurações dos fornecedores de serviços e (3) utilização de tecnologias de autenticação. Em [McMilan, 2011], Nuvens privadas não são mais que:

"Private clouds are deployed within the company firewall and managed by this enterprise's organization itself."

2.4.2 **Nuvem Pública**

No modelo de Nuvem pública, a infraestrutura é disponibilizada para o público em geral, sendo acedida naturalmente por qualquer utilizador que conheça a localização do serviço. Neste modelo não podem ser aplicadas restrições de acesso quanto à gestão da rede, e menos ainda, utilizar técnicas para autenticação e autorização. Em [McMilan, 2011], é simplesmente definida como:

"Public clouds are cloud services which exist beyond the company firewall, and they are fully hosted and managed by the cloud provider, not by the customer company."

2.4.3 **Nuvem Comunitária**

No modelo de Nuvem comunitária ocorre a partilha entre diversas empresas de uma Nuvem, sendo esta suportada por uma comunidade específica com interesses comuns, tais como: a missão, os requisitos de segurança, política e considerações sobre flexibilidade. Este modelo pode existir localmente ou remotamente e geralmente é administrado por uma empresa da comunidade ou por terceiros.

2.4.4 **Nuvem Híbrida**

No modelo de Nuvem híbrida, existe uma composição de duas ou mais nuvens, que podem ser privadas, comunitárias ou públicas e que permanecem como entidades únicas, ligadas por uma tecnologia padronizada ou proprietária que permite a portabilidade de dados e de aplicações. Em [McMilan, 2011], refere-se que:

"Hybrid clouds are a combination of public and private clouds together. These clouds would typically be created by the enterprise, and the cloud provider has a part of management responsibilities."

2.5 Plataformas

Uma plataforma integrada de ciclo de vida fornece aos utilizadores todas as condições necessárias para desenvolver aplicações de uma forma colaborativa e ferramentas de controlo das diferentes versões. Em geral, as empresas proporcionam plataformas integradas de ciclo de vida, que incluem os seguintes componentes, nomeadamente:

- Um motor: um núcleo que inclui a lógica, necessária para operar com o fornecedor de serviço;
- Ferramentas de desenvolvimento: diversos meios, que os programadores usam para criarem aplicações;
- Ferramentas de teste: ferramentas especiais, criadas para testar fontes e sistemas de desenvolvimento;
- Ferramentas de terceiros e serviços: *software* e serviços que complementam o ambiente existente com características adicionais;
- Integração de base de dados: o desenvolvimento de aplicações é capaz de armazenar informações na base de dados.

Este tipo de plataformas possui *middleware* próprio e interfaces de programação que são oferecidos para desenvolvimento de aplicações. O nível de abstração é provavelmente menos funcional, mas prevê a utilização de ferramentas ao mais alto nível.

Várias empresas encontram-se a desenvolver plataformas para a Nuvem (Google, Microsoft, Amazon, GoGrid, RackSpace, HP, IBM e outros), as quais competem entre si nos respetivos segmentos de negócio, para permitirem aos clientes o controle e a execução de aplicações de apoio aos respetivos negócios. Google e Microsoft oferecem serviços para a criação de novas aplicações na Nuvem (modelo SaaS) enquanto a Amazon oferece um ambiente para a sua instalação direta (IaaS).

Atualmente existem diversas formas de interação com os vários sistemas de Nuvem. Por exemplo, no caso da Amazon, além de uma API rica em serviços, existe um portal para controlar os recursos necessários às especificações do utilizador. No caso do Windows Azure, para além da API, possui aplicações *Web* para controlar os recursos disponibilizados.

2.6 Segurança na Nuvem

Apesar de ser um tema muito debatido, há certos aspetos por de trás do facto de muitas organizações ainda não se encontrarem totalmente confiantes em mover a sua infraestrutura para a Nuvem. Esta apresenta diversas vulnerabilidades ao nível da segurança [Mcafee Corporation, 2010]. Ao longo deste subcapítulo serão descritos alguns desses problemas tendo em conta: Segurança, Riscos, Ameaças e Mitigações. Sem querer fazer uma descrição exaustiva de cada um destes tópicos, apresenta-se nas próximas secções uma breve descrição, começando pelos fundamentos de segurança computacional.

2.6.1 **Fundamentos de segurança computacional**

Todo o sistema computacional necessita de proteção, porém, é necessário analisar a sensibilidade dos dados que uma aplicação irá manipular, para que a segurança seja corretamente dimensionada [Landwehr, 2001]. Ao longo dos anos, a segurança computacional passou por várias fases, inicialmente pretendia-se prevenir as violações de proteção, após esta fase o objetivo foi detetar e limitar as violações que não podiam ser prevenidas. Posteriormente, o foco foi tolerar os ataques, visando manter o fornecimento dos serviços.

Estamos a caminhar em direção à segurança comercializada como um serviço ou parte deste e neste caso terá que se confiar em quem vende o serviço. Analogamente, à forma como se coloca o dinheiro num banco, é necessário possuir confiança para que o banco proceda de uma forma íntegra.

2.6.2 **Riscos**

As organizações devem avaliar o risco e as opções de segurança antes de mover os seus recursos para o ambiente de Nuvem. É necessário avaliar quais os dados e os serviços que podem ser transferidos para o ambiente externo à organização se a Nuvem for pública. No processo de análise procura-se avaliar os impactos gerados, caso algum requisito de segurança (confidencialidade, integridade ou disponibilidade) seja comprometido. As organizações podem mover integral ou parcialmente os seus processos ou dados para o ambiente de Nuvem. Parte das transações e informações podem ser mantidas dentro do perímetro da organização em ambiente privado [CSA, 2009].

2.6.3 **Ameaças**

Existem vários tipos de ameaças inerentes à computação em Nuvem. Algumas são comuns a qualquer máquina que disponibilize um serviço *on-line*. Antes de resumir os vários tipos de ataques em função dos serviços de segurança na Tabela 1, convém definir o significado destes termos.

Ataque à segurança: Um ataque à segurança apresenta-se como qualquer ação que comprometa a segurança da informação ou do sistema.

Mecanismo de segurança: Caracteriza-se por um mecanismo projetado para detetar, prevenir ou recuperar de um ataque à segurança.

Serviço de segurança: Serviço destinado a melhorar a segurança dos sistemas de processamento de dados e da transferência de informação de uma organização. Os serviços têm como intenção melhorar a resistência a ataques através da utilização de um ou mais mecanismos de segurança, para providenciarem o serviço.

Tabela 1 – Relação entre serviços de segurança e ataques

	Ataques					
Serviços	Divulgação de conteúdo	Análise de tráfego	Dissimulação	Repetição	Modificação de mensagens	Negação de serviço
Autenticação da origem dos dados	----------	----------	X	----------	----------	----------
Controlo de acessos	----------	----------	X	----------	----------	----------
Confidencialidade	X	----------	----------	----------	----------	----------
Confidencialidade dos fluxos de dados	----------	X	----------	----------	----------	----------
Integridade dos dados	----------	----------	----------	X	X	----------
Não Repudiação	----------	----------	----------	----------	----------	----------
Disponibilidade	----------	----------	----------	----------	----------	X

2.6.4 **Formas de Mitigação**

O investimento na segurança e formas de mitigação contra as ameaças às suas respetivas plataformas é elevado e considerado um ponto fundamental para qualquer CSP. Estes implementam todo um conjunto de normas para tornar a segurança da sua plataforma o mais eficaz possível. Embora estes princípios estejam presentes em qualquer modelo ou tipo de Nuvem, é realizada uma abstração destes princípios ao utilizador. Existe, porém, um tipo de ambiente (o utilizado neste trabalho - IaaS) em que é possível ao responsável pela administração, poder manipular e gerir os elementos de segurança da infraestrutura. Nas próximas secções é realizada uma descrição de alguns destes elementos.

2.6.4.1 **Network Address Translation (NAT)**

O grande crescimento dos sistemas de informação por todo o mundo apresentou um grande desafio à escalabilidade da Internet com o esgotamento dos endereços IPv4 disponíveis [Dutcher, 2001]. Antes da introdução do IPv6 foi desenvolvido um mecanismo para suprimir esta necessidade de endereços, nomeadamente o NAT. Este mapeia endereços privados IPv4 em endereços públicos, os quais podem ser encaminhados pela Internet, o tráfego no sentido inverso é novamente traduzido para poder chegar à rede interna. Este exemplo encontra-se representado na Figura 4.

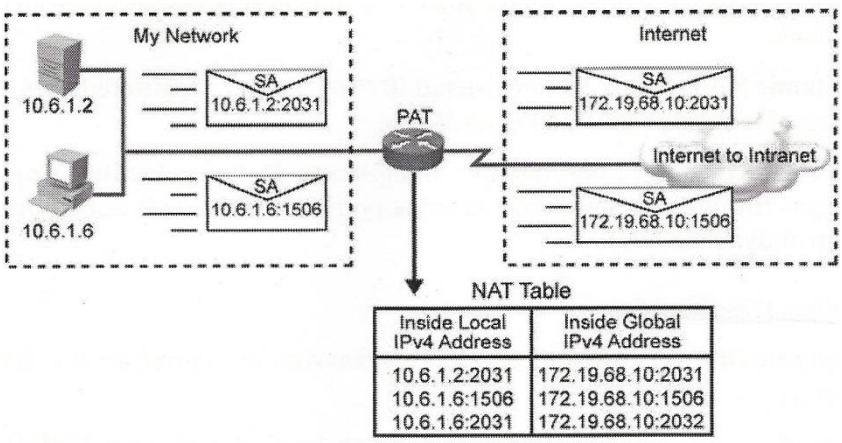

Figura 4 - Exemplo de NAT na forma de Port Address Translation (PAT) em [Cisco Systems, 2007]

Normalmente, o NAT associa duas redes e traduz endereços privados na rede interna em endereços públicos, antes dos pacotes serem encaminhados. Como parte desta funcionalidade, pode-se configurar o NAT para apenas anunciar um endereço (representando a rede interna) para o exterior. Isto efetivamente esconde os endereços da rede interna do exterior, fornecendo alguma segurança adicional.

Qualquer dispositivo entre uma rede interna e uma rede pública (como uma *firewall*, router ou computador) usa NAT definido no RFC 3022 [Cisco Systems, 2007].

Resumindo o NAT oferece os seguintes benefícios:

- Elimina a necessidade de atribuir endereços a todas as máquinas que requeiram acesso externo, poupando-se recursos;
- Conserva endereços através da multiplexagem ao nível das portas. As máquinas internas podem partilhar um único endereço público para todas as comunicações;
- Adiciona segurança na rede, já que as máquinas da rede interna não anunciam os seus endereços internos ou topologia interna ao comunicar com o exterior através de NAT.

2.6.4.2 **Firewalls**

A segurança é uma das principais preocupações para qualquer administrador de sistemas devido ao crescente evoluir do conhecimento dos atacantes, tornando-se, por isso, necessário o conhecimento da implementação de uma *firewall* eficaz que, protegendo o sistema, não interfira no funcionamento do tráfego normal e expectável das organizações. Existem dois principais tipos:

- ***Stateful***: Mantém em registo o estado das ligações que a atravessam. Como tal, consegue analisar se determinados pacotes estão associados com uma ligação já existente;
- ***Stateless***: Estas não possuem a noção de estado e, por isso, examinam cada pacote de forma isolada. Apesar de menos robustas, continuam a ser utilizadas em sistemas sensíveis à carga computacional induzida pelas *stateful*.

As *firewalls* na sua essência são compostas por uma ou mais ACL´s. Estas definem-se como listas ordenadas de regras que concedem a permissão/negação a determinado tipo de tráfego. Embora as ACL sejam utilizadas para filtragem de pacotes, as suas aplicações não se resumem apenas a essa função. Podem também ser utilizadas para manipulação de rotas em protocolos de encaminhamento, validação no acesso a serviços de rede ou manipulação de pacotes (implementação de Quality of Service (**QoS**), aplicação de mecanismos de NAT, ajuste dos cabeçalhos dos protocolos). Existem dois principais tipos de implementações de ACL´s: Cisco e Netfilter. Como neste projeto são utilizadas ACL baseadas em Netfilter, este processo é descrito de seguida, de forma mais detalhada.

2.6.4.3 **Netfilter**

O Netfilter atua dentro do *kernel* do Linux, permitindo aos módulos interagir através de funções predefinidas com o "*stack*" de rede. Uma função previamente registada pode ser, então, executada para inspecionar cada pacote que o transpõe. Este é o elemento principal de um conjunto de ferramentas (IPTtables, Conntrack e NAT) que, juntos, formam a plataforma conhecida como Netfilter. Executa por defeito Stateful Packet Inspection (**SPI**), com recurso ao módulo Conntrack.

O Netfilter define cinco pontos de interceção no processamento de pacotes que passam pelo *kernel*: Prerouting; Input; Foward; Output e Postrouting. A cada um destes pontos está associada uma tabela que possui uma lista de regras. Cada regra representa uma oportunidade para afetar o comportamento de um determinado pacote. As regras são sempre analisadas em sequência. A sequência do fluxo de filtragem pode ser observada na Figura 5.

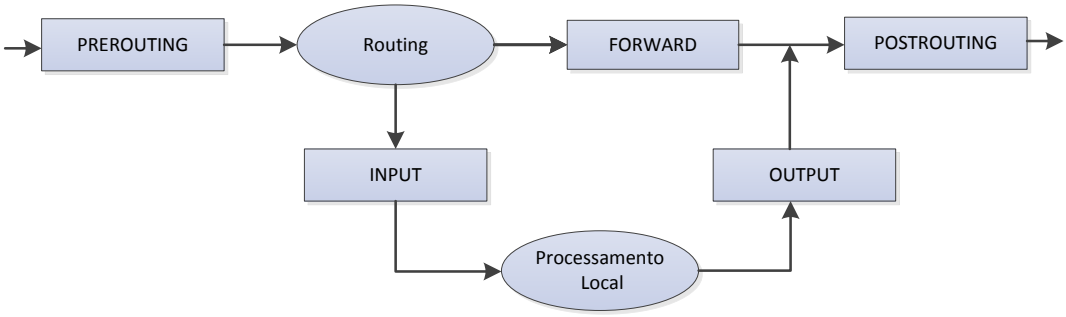

Figura 5 - Sequência do fluxo de filtragem nos vários pontos de interceção

Existem três tabelas com características e funções específicas. A de Mangle é responsável pela alteração do cabeçalho dos pacotes TCP. A tabela Filter é responsável pela filtragem. A terceira tabela, a de NAT, é responsável, como o próprio nome indica, pela tradução de endereços na rede. A Tabela 2 ilustra mais detalhadamente estas relações.

Tabela 2 - Arquitetura e funções do Netfilter

Tipo de tabela	Função	Ponto de interseção	Função do ponto de interseção
Filter	Filtragem	Forward	Filtra pacotes que não são dirigidos à máquina onde se encontra a *firewall*
		Input	Filtra pacotes destinados à máquina da *firewall*
		Output	Filtra pacotes originários na máquina da *firewall*
NAT	NAT	Prerouting	Ativa quando o Nat ocorre antes do routing Usada com o Nat do IP de destino - DNAT
		Postrouting	Ativa quando o Nat ocorrer depois do *routing* Usada com o Nat do IP de origem – SNAT
		Output	Ativa quando pertence a pacotes originários da máquina da *firewall*
Mangle	Modificação do cabeçalho TCP	Prerouting Postrouting Output Input Forward	Modificação dos bits QoS dos pacotes TCP antes de ocorrer *routing*

2.6.4.4 **Módulo Fuzzy: Proteção contra ataques de Denial of Service**

Segundo [Hime, et al., 2012], o módulo Fuzzy permite o mapeamento de pacotes, de acordo com um perfil dinâmico implementado através de um simples Fuzzy Logic Controller (**FLC**). A ideia básica é a de realizar "match", caso sejam concretizados dois parâmetros que definem o intervalo de filtro:

- Quando a percentagem dos pacotes é inferior ao limite inferior do intervalo definido, a regra nunca é realizada;
- Entre o limite inferior e o limite superior, a regra é realizada de acordo com a percentagem dos pacotes;
- Finalmente, quando a percentagem dos pacotes é superior ao limite superior, a regra é realizada com o valor máximo, ou seja 99%.

Levando em conta que a taxa da amostra é variável e aproximadamente 100ms, o autor acredita que o módulo apresenta uma boa resposta, adaptando-se rapidamente à mudança de padrões de tráfego. Por exemplo, se o objetivo for evitar ataques de DOS, poder-se-ia usar a seguinte regra:

```
Iptables –A INPUT –m fuzzy --lower-limit 100 --upper-limit 1000 REJECT
```

Que permitiria a seguinte análise:

- Antes de 100 pacotes por segundo (pps), o filtro está inativo;
- Entre 100 pps e 1000 pps, a percentagem de aceitação de pacotes baixa de 100% (a 100 pps) para 1% (quando são atingidos 1000 pps);
- Acima de 1000 pps, a percentagem de aceitação mantêm-se constante a 1%.

2.7 Características de Ambientes na Nuvem

Neste subcapítulo ilustram-se algumas características intrínsecas de qualquer ambiente na Nuvem, que importa mencionar no contexto do trabalho desenvolvido.

2.7.1 **Desempenho, latência e confiabilidade**

A latência [Napper, et al., 2009], [Minnear, 2011] sempre se apresentou como um fator crítico na Nuvem. Os fatores que contribuem para a latência são a criptografia dos dados e a perda de pacotes quando se opera em redes públicas devido ao congestionamento. A congestão contribui para a latência quando o fluxo de tráfego através da rede é elevado e existem muitos pedidos que necessitem de ser executados ao mesmo tempo. Além disso, o desempenho do sistema é também um fator que deve ser tomado em conta. Por vezes, os CSP´s encontram-se num baixo nível de capacidade, quer por permitir o acesso a muitas máquinas virtuais ou atingindo limites de rendimento superiores às suas ligações de Internet por causa da elevada procura. Isto fere o desempenho do sistema e aumenta a latência do mesmo.

2.7.2 **Portabilidade e interoperabilidade**

As organizações podem necessitar de alterar os CSP´s e já existiram casos em que empresas não puderam transferir os seus dados e aplicações de CSP. Algumas empresas usam plataformas em diferentes Nuvens para aplicações de diferentes tipos, com base nas suas necessidades e nos serviços prestados pelos CSP´s. É necessária uma boa infraestrutura interna da organização para manter um equilíbrio e para lidar com a interoperabilidade entre diferentes plataformas de Nuvem [Foster, et al., 2008]. O risco de serviços externos saírem fora de controlo é muito comum num ambiente de Nuvem híbrido, público e privado. Todos os dados necessitam de possuir uma criptografia adequada, e a gestão das chaves torna-se uma tarefa difícil em situações deste tipo [Kretzschmar, et al., 2010].

2.7.3 **Armazenamento de dados em redes IP**

O armazenamento de dados *on-line* apresenta uma notória popularidade hoje em dia e foi observado que a maioria do armazenamento das empresas estará em rede nos próximos anos, pois permite às empresas manter grandes volumes de dados sem a criação de uma infraestrutura dedicada. Embora existam muitas vantagens de possuir armazenamento de dados *on-line*, existem ameaças à segurança que podem causar a disrupção de dados ou indisponibilidade de dados em horas cruciais. Tais questões são observadas com maior frequência no caso de dados dinâmicos, em comparação com dados estáticos. Dependendo dos vários níveis de operações e de armazenamento previstos, estes dispositivos de rede são classificados em Storage Area Network (**SAN**) e Network-Attached Storage (**NAS**) e, uma vez que estas redes de armazenamento residem em vários servidores, existem múltiplas ameaças ou riscos a eles associados [Mcafee Corporation, 2010]e [Cisco Systems, 2010].

2.7.4 **Virtualização**

A evolução da tecnologia de máquinas virtuais tem permitido a sua ampla adoção em sistemas de produção. Vários aspetos na construção do Hypervison, na sua maioria relacionados com o desempenho na execução de sistemas, foram resolvidos nos últimos anos [Garfinkel, et al., 2003]. Atualmente, a principal utilização de máquinas virtuais no meio corporativo tem sido a consolidação de servidores, procurando a redução de custos em *hardware*, *software* e gestão do parque tecnológico [Newman, et al., 2005], [Ferre, et al., 2006].

2.8 Monitorização

A monitorização incorpora-se no ciclo operacional: Desenvolvimento – Expansão – Monitorização, comum a inúmeras áreas científicas. No entanto, muitas vezes esta monitorização de serviços/equipamentos/sistemas/aplicações/modelos de *software* ou *hardware* é vista com um menor grau de relevância no conjunto global, apesar de ser a fração do ciclo onde a maior parte do tempo é despendido.

Empresas, instituições ou laboratórios no atual contexto económico encontram-se principalmente a operar apenas na parte do ciclo respeitante à monitorização, devido à falta de verbas para realizar expansões de sistemas ou infraestruturas.

A procura por soluções *open source* de monitorização na área das TI foi sempre uma constante, procura essa que ajudou ao melhoramento incremental das aplicações, contribuindo como é claro o elevado grau de conhecimento dos utilizadores das mesmas, evoluindo muitas vezes de utilizadores a desenvolvedores.

A monitorização é realizada atualmente em 4 fases distintas:

- **Fase de *scripting***: Onde são desenvolvidos *scripts* para ir buscar a informação pretendida (Perl e PHP são as linguagens mais utilizadas);
- **Fase de alarmística**: Após a colocação da informação em bases de dados, são configurados os alarmes e a sua visualização, dependendo do pretendido (o Nagios[1] é o software mais utilizado);
- **Fase de construção de elementos de visualização**: Nesta fase são desenvolvidos gráficos dos padrões que se queiram monitorizar (utilização do CPU, tráfego, temperaturas entre outros) novamente através do uso de *scripting* para ir buscar a informação à bases de dados. A visualização é conseguida com o uso de ferramentas como o Cacti[2] e RrdTools;
- **Fase de apresentação**: Onde são incorporados os resultados numa interface *Web*, para indivíduos com menores conhecimentos contribuírem para a monitorização.

[1] http://www.nagios.org/
[2] http://www.cacti.net/

É comum existirem equipas de alguma dimensão focadas apenas na monitorização. Todas estas soluções *open source* são quase sempre baseadas em Linux, aumentando ainda mais a fasquia de conhecimento requerido.

Um dos pressupostos da Nuvem é a diminuição do número de recursos humanos necessários para efetuar a gestão de sistemas. Neste projeto é utilizado o serviço CloudWatch na AWS para monitorizar a infraestrutura realizada e apresentar gráficos dos resultados obtidos. Com este serviço é realizada a abstração da complexidade de executar cada uma das fases descritas, ao usar-se uma interface *Web* e a AWS CLI para a obtenção dos mesmos resultados.

3. Contribuições para o Projeto SmartCITIES

Dentro dos diversos trabalhos em curso, relacionados com o projeto SmartCITIES, este vai abordar os temas associados à interoperabilidade, segurança e comunicações, dos quais se contribui para os seguintes *delivers* do projeto (notação usada corresponde àquela adotada no mesmo):

A5 - Estado da arte dos mecanismos de troca de dados existentes no domínio dos transportes públicos, aplicados à bilhética;

C2 - Aquisição de conhecimento sobre estabelecimento de canais seguros com equipamentos.

Assim, neste capítulo é realizado um levantamento das normas vigentes dos atuais sistemas de bilhética relacionadas com a interoperabilidade entre entidades. É realizada também uma análise das especificações de um protocolo de comunicação (ITSO), que descreve a forma de comunicação entre os canais de bilhética e o bus de serviços central, numa perspetiva funcional e técnica. O protocolo detalha qual o modelo de segurança utilizado, de forma a garantir a confiabilidade e segurança dos dados transmitidos entre os subsistemas.

3.1 Especificações das Comunicações na Bilhética

A arquitetura de um sistema de bilhética eletrónica pode-se subdividir em duas grandes áreas [ISO 14443, 2008]:

- **Front-end** *(*"thin/fat devices"*)*: Um chip RFID (onde se armazenam os dados de um dispositivo eletrónico), uma interface entre este e um terminal de leitura RFID e o próprio leitor RFID;
- **Back-end** (infraestrutura central/*backoffice*): Uma infraestrutura constituída pelas bases de dados, servidores e aplicações necessárias.

A interface de comunicação em *front-end*, entre o terminal e o chip RFID, já se encontra normalizada segundo a norma ISO 14443, a qual define quatro componentes: características físicas, potência da interface rádio, inicialização/anti-colisão e protocolo de transmissão. A maior parte das implementações de sistemas bilhética eletrónica são baseadas nesta norma, amplamente adotada.

Na área de *back-end*, contudo, as soluções podem variar bastante. Segundo [Zanetti, et al., 2011], tem-se verificado um enorme esforço com foco na normalização destes processos. A Figura 6 apresenta as principais normas desenvolvidas desde a interface de leitura até a camada de aplicação.

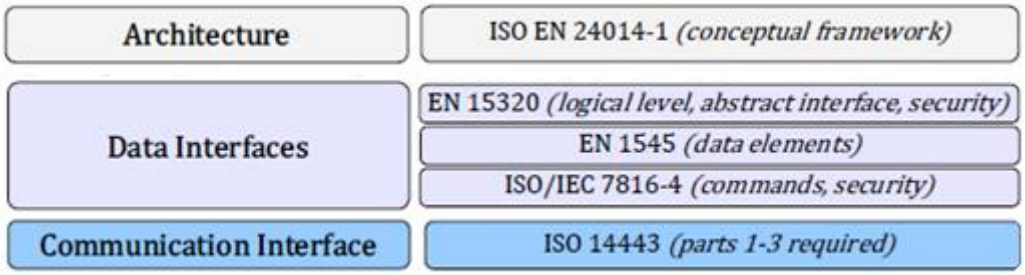

Figura 6 - Normas de suporte à interoperabilidade de sistemas de bilhética eletrónica [Igudim, 2012]

A Europa adotou a norma internacional ISO 24014-1, como um modelo global que define sistemas e processos necessários à gestão da distribuição e uso de produtos de bilhética num ambiente interoperável de transportes públicos [Verity, 2010]. Um passo final para fornecer aos utilizadores de transportes públicos por toda a Europa, maior acessibilidade, facilidade e conforto na utilização dos mesmos, foi dado com a conceção da iniciativa Interoperable Fare Management (**IFM**) [Chantérac, et al., 2009]. O projeto IFM visa a construção com base na norma ISO 24014-1, de um projeto europeu de unificação de toda a rede de transportes públicos na Europa. Dele fazem parte as maiores organizações europeias de bilhética acima descritas. O objetivo é o de fornecer aos utilizadores dispositivos, cuja informação bilhética é interpretada e reconhecida em todo o espaço europeu. Estes dispositivos, além de servirem como títulos válidos na indústria do transporte em diferentes áreas geográficas, poderão ser usados de forma sustentável para serviços como parques de estacionamento por exemplo. Este é um objetivo ambicioso, visto que, atualmente a maioria dos dispositivos eletrónicos existentes encontra-se restringido a projetos/serviços específicos.

Três organizações de bilhética europeias demonstraram que é possível o uso de um único *smartcard* no acesso ao transporte público fornecido pelas respetivas organizações. Numa iniciativa[3] conjunta foram carregadas aplicações da britânica ITSO[4], francesa NaviGO[5] e alemã VDV[6] num *smartcard*, com o objetivo de demonstrar que o transporte entre diferentes redes de transporte público de diferentes países pode ser realizado com recurso a um único dispositivo, representando um importante passo na tecnologia associada aos sistemas de bilhética [Verity, 2010]. O projeto foi coordenado pela ITSO, com a demonstração de interoperabilidade gerida pela RATP[7] (França), participação ativa da VDV e assistência técnica da Nextendis[8] (França) e ESP Systex[9] (Reino Unido). O dispositivo utilizado foi um comum *smartcard* de produção da VDV.

Esta meta de interoperabilidade implica a existência de medidas de segurança e privacidade comuns. A necessidade de segurança é amplamente reconhecida pelas companhias de transporte, já que na falta desta, podem sofrer graves quebras nas suas receitas.

A privacidade, nomeadamente a pertencente ao cliente, não se encontra entre as principais metas dos fornecedores de serviços. Isto deve-se ao facto, dos riscos associados à violação de privacidade serem menores quando comparados com os relativos à segurança. No entanto, a crescente consciencialização dos clientes para a problemática da violação da privacidade, poderá estimular as companhias de transporte a investir em privacidade para permanecerem competitivas.

[3] http://www.ifm-project.eu
[4] http://www.itso.org.uk/
[5] http://www.navigo.fr/
[6] http://www.vdv.de/en/wir_ueber.../wir_ueber_uns.html
[7] http://www.ratp.fr/
[8] http://www.nextendis.com/fr/
[9] http://www.espsystex.co.uk/

3.2 Segurança e Privacidade Associadas às Especificações

As normas ilustradas na Figura 6, representam a estrutura genérica de um sistema de bilhética. Nesta secção é realizada uma análise concisa sobre as medidas de segurança e privacidade especificadas nas mesmas.

3.2.1 **Camada da arquitetura**

3.2.1.1 **ISO EN 24014-1**

Esta norma introduz uma plataforma conceptual para o desenvolvimento de arquiteturas interoperáveis de sistemas de bilhética relativas ao transporte. Descreve a estrutura de uma plataforma interoperável, os atores principais e os principais fluxos de troca de informações. A privacidade é considerada a um nível conceptual, requisitando-se a definição de um esquema de segurança que providencie a proteção de privacidade, além de integridade e confidencialidade entre atores, para assegurar um fluxo seguro de dados no interior de um sistema IFM [ISO 24014-1, 2007]. As medidas relacionas com a segurança, são definidas na respetiva política de segurança. A gestão da segurança é realizada pela entidade responsável pela implementação da política de segurança, englobando todos os atores.

A norma estabelece que a privacidade de um utilizador deve ser protegida, como exigido pelas leis aplicáveis, estabelecendo as seguintes regras:

- Apenas dados pessoais relevantes à operação do sistema IFM, possam ser exigidos ao utilizador;
- A divulgação detalhada do consumo de serviços numa fatura deverá ser uma opção ao critério do cliente;
- Um ator IFM não pode divulgar informações dos seus clientes a terceiros, sem a específica autorização do cliente;

- Num sistema IFM, os dados relativos aos clientes, trocados entre estes e o proprietário de um produto, devem ser apenas o número de identificação do contrato. Apenas entre o parceiro contratual do cliente e este poderão ser trocados, em simultâneo, o número de identificação do contrato e o nome do cliente.

3.2.2 **Camada das interfaces de dados**

3.2.2.1 **EN 15320**

Esta norma define a estrutura lógica dos dados residentes no cartão, especifica uma interface abstrata de interação entre o cartão e o terminal (logicamente apresenta duas interfaces: a interface dos dados do cartão e a interface dos grupos de dados) e considera a segurança através da especificação de um Security Subsystem (**SSS**). Este é dividido quanto ao sistema de gestão da segurança do cartão e no relativo aos grupos de dados, para corresponder às duas interfaces lógicas. As operações relativas à segurança são definidas em perfis (perfis de cartões e de grupos de dados), ilustrados na Figura 7.

A privacidade é considerada apenas indiretamente nesta norma, através da descrição de grupos de dados contendo informação relacionada com esta (grupos de dados do titular do cartão). Se estes dados se encontrarem presentes, os mecanismos de controlo de acesso em conjunto com a devida encriptação, devem ser implementados para fornecer a privacidade necessária.

A divisão em duas interfaces lógicas fornece uma implementação flexível de esquemas de controlo de acessos. Operações relacionadas com a segurança podem ser definidas nos respetivos perfis e executadas quando for necessário assegurar a execução adequada das mesmas. Este mecanismo pode ser usado para o processamento de dados relacionados com a privacidade, fornecendo proteção contra a imprópria identificação de clientes. Esta norma no entanto, não procura visar questões relacionadas com a privacidade dos clientes, focando-se apenas nas questões relativas à segurança.

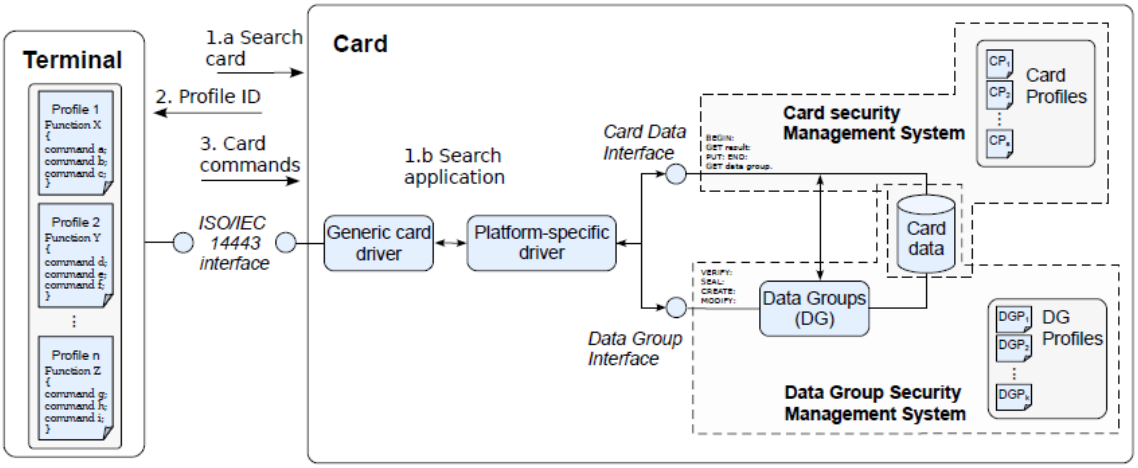

Figura 7 - Interação entre um terminal e cartão, baseada nos processos descritos na norma EN 15320 [Igudim, 2012]

3.2.2.2 **EN 1545-1**

A estrutura dos elementos de dados residentes no cartão é considerada e expressa de acordo com a Abstract Syntax Notation 1 (**ASN.1**). A informação relacionada com privacidade encontra-se nos elementos de dados presentes na Tabela 3. Estes dados podem ser protegidos aplicando encriptação e mecanismos de controlo de acessos definidos num nível lógico nos respetivos perfis do SSS, abrangendo assim as questões de privacidade como a exposição dos identificadores pessoais ou dos objetos.

Tabela 3 - Campos relacionados com privacidade presentes na norma EN1545-1

Campo relativo à privacidade	Descrição
Data de nascimento	—
Nome	—
Local de nascimento	—
Número de cliente	Número de referência do cliente
ID do dispositivo	Pode ser relacionado a um cliente específico
Endereço de correio eletrónico	—
Número de telefone	—
Código postal	—
ID do perfil de cliente	Por exemplo: Estudante, Militar.
Dados de utilizador	Informação adicional sobre um utilizador

3.2.2.3 **EN 1545-2**

As estruturas de dados presentes no cartão são adicionalmente especificadas, de acordo com os requisitos de um sistema de gestão de bilhética interoperável na área dos transportes. Esta parte da especificação foca-se apenas, nas questões funcionais de um sistema de transporte.

3.2.2.4 **ISO/IEC 7816-4**

Esta especificação considera as questões relacionadas com a troca de comandos, bem como a recuperação de estruturas de dados e objetos contidos no cartão. A segurança e privacidade são concretizadas com a especificação de métodos para uma segura troca de mensagens e de uma arquitetura de segurança que defina direitos de acesso a ficheiros e dados no cartão. Os métodos de acesso aos algoritmos presentes nos cartões também são considerados [ISO 7816-4, 2005].

3.2.3 **Camada da interface de comunicações**

3.2.3.1 **ISO 14443**

As primeiras três partes são necessárias ao estabelecimento de uma ligação entre o cartão e terminal. A quarta parte é opcional e usualmente usada para cartões com uma elevada capacidade de processamento. A norma não considera nenhumas questões relacionadas com segurança e privacidade, focando-se na funcionalidade [ISO 14443, 2008].

3.2.4 **Síntese**

Sumariamente, as normas constituintes de um sistema genérico de bilhética consideram primariamente a segurança dos ativos das companhias de transporte e a adequada manutenção das funcionalidades do sistema. As questões de privacidade dos clientes são abordadas como uma extensão da segurança, sem serem tomadas medidas

especificas na sua resolução. A Tabela 4 sintetiza as medidas de segurança e privacidade presentes em cada norma.

Tabela 4 - Medidas de segurança e privacidade presentes em cada norma

Norma	Segurança	Privacidade
ISO EN 24014-1	- Definição de uma política de segurança; - Gestão da segurança.	- Requisitos insuficientes, direcionados ao cumprimento da regulação.
EN 15320	- Security Subsystem (SSS); - Operações de segurança definidas em perfis.	- Grupos de dados de privacidade; - Proteção através de controlo de acessos e encriptação.
EN 1545	- Definição de campos de segurança.	Definição de campos de privacidade
ISO/IEC 7816-4	- Troca de dados segura; - Arquitetura de segurança através de controlo de acessos.	- Mecanismos de segurança aplicam-se a dados críticos de privacidade.
ISO 14443 (1-3)	Não se encontra considerada	Não se encontra considerada

3.3 Smartcards

O uso de *smartcards* como meio alternativo de pagamento de serviços de transporte, emergiu como uma opção perfeitamente viável para a maioria dos operadores de bilhética. Estes são essencialmente cartões de plástico, com o mesmo tamanho de um cartão bancário, incorporando um *microchip* eletrónico. Este executa todas as funções de armazenamento, processamento e escrita de dados. Segundo [Blythe, 2004] os *microchips* podem dividir-se em duas classes: (a) aqueles constituídos apenas por memória, para armazenamento de informação, com um razoável nível de segurança; (b) e os que para além desta, possuem um processador controlado por um sistema operativo com a capacidade de processamento de dados e execução local de pequenos programas. A área de armazenamento, denominada de Electrically Erasable Programmable Read-Only Memory (**EEPROM**), quando devidamente protegida com restrições de segurança, pode suportar a atualização do seu conteúdo. Os *smartcard* mais recentes possuem ainda processadores dedicados a desempenhar funções criptográficas, exigidas por diversos sistemas de segurança.

Um *smartcard* é, portanto, caracterizado pela habilidade de ler, escrever e armazenar dados dentro de um ambiente extremamente seguro, dependendo do *microchip* que possui. Segundo [Blythe, 1999], estas características transformam-no num título reutilizável que pode possuir direitos de subscrição, crédito, detalhes do titular e acesso a um extenso conjunto de serviços e aplicações (dependendo da complexidade da configuração do *microchip* e sistema operativo). A segurança associada a este tipo de dispositivo permite que os dados não possam ser alterados ou copiados por terceiros sem acesso autorizado, contribuindo assim para a longevidade deste tipo de dispositivo eletrónico e conferindo-lhe a propriedade de armazenar títulos de elevado (títulos anuais de transportes terrestres ou aéreos) e baixo (títulos diários de transportes rodoviários/metropolitanos) valor económico.

Do ponto de vista da indústria dos transportes públicos, os *smartcards* podem fornecer importantes dados de mercado e pesquisa sobre quando e onde estes são usados pelos seus clientes, concedendo às organizações a capacidade de desenvolver e readaptar serviços para melhor suprimir as necessidades dos mesmos. [Bagchi, 2003] demonstra o grande potencial que estes dados podem conter e os benefícios dai provenientes. As autoridades locais e as empresas de transportes públicos beneficiariam de um melhor controlo sobre os orçamentos de viagens concessionadas e os operadores dos dados necessários para uma distribuição mais efetiva dos recursos necessários. Também oferecem uma poupança do ponto de vista dos operadores, através da redução de recursos para tratamento de receitas, redução de fraudes e um melhor controlo de proveitos. A tecnologia inerente aos dispositivos eletrónicos vai desempenhar um papel significativo na área dos transportes públicos num futuro próximo.

3.4 Protocolos Integrados de Comunicação (ITSO)

No âmbito da comunicação entre os *smartcards* e a infraestrutura central/*backoffice*, o mercado da bilhética evoluiu no sentido da unificação das diversas normas e de fornecer sistemas *middlewares* genéricos interoperáveis, que possam ser usados por diferentes fornecedores. Como resultado, foram criadas várias especificações adotadas em diferentes países, sendo as principais: Calypso Network Systems (Bélgica, Canadá, China, França, Israel, Itália, Portugal), Integrated Transport Smartcard Organization (Reino Unido) ou VDV Core Application (Alemanha). Estas foram desenvolvidas sobre a norma ISO 14443, adotando características suficientemente genéricas, para criar interoperabilidade entre fornecedores, entre os vários sistemas de bilhética. Na secção seguinte é realizada uma análise mais detalhada a uma destas especificações.

Em resposta às preocupações sobre a fragmentação no uso de *smartcards* nos transportes públicos do Reino Unido, e como tal inviabilizando os benefícios que tal sistema pudesse oferecer, foi formada a Integrated Transport Smartcard Organization (**ITSO**) em 1999. A ITSO agregou as principais entidades da indústria de transportes no Reino Unido, para construir uma especificação interoperável de implementação de smartcards e ambiente de suporte, na área dos transportes [Blythe, 2004]. Esta especificação é essencialmente composta pelas estruturas Customer Media (**CM**), ITSO Product Entities (**IPE**), Point of Service Terminals (**POST**), BackOffice (**HOPS**), Arquitetura de Segurança, Comunicações e ITSO Data Messages [ITSO Technical Specification 1000-0, 2010]. Todos os componentes da sua infraestrutura, bem como o seu modo de funcionamento são descritos nas próximas secções.

3.4.1 **Customer Media (CM)**

O termo CM refere-se à plataforma eletrónica usada pelos utilizadores para armazenar os produtos de bilhética. Atualmente, é usado um *smartcard*, podendo no futuro mudar, visto que a especificação é genérica neste termo. O CM deve ser autónomo em termos de energia e possuir uma interface de dados compatível com a norma ISO 14443.

A interface CM-to-POST é uma parte crítica para a interoperabilidade, contendo vários atributos chave: mecanismos de transferência de energia, físicos, mecanismos de

transferência de dados, arquiteturas de dados, segurança e acesso a dados e análise de tempos de transação.

Uma das funções mais importantes de um CM é o mapeamento das estruturas de dados lógicos para o dispositivo eletrónico físico, em alguns casos este mapeamento é realizado ao nível físico do bit e byte, enquanto noutros, possui a função de armazenar elementos lógicos. Em [ITSO Technical Specification 1000-10, 2010] são definidas duas grandes classes de dispositivos:

- Plataformas contendo uma aplicação ITSO completa;
- Plataformas contendo uma aplicação ITSO compacta.

Enquanto as primeiras podem conter vários produtos de bilhética concorrentemente, as segundas conseguem apenas armazenar um produto num dado intervalo de tempo.

A segurança dos dados no CM é um fator critico para o sucesso das operações na arquitetura ITSO, como tal, vários mecanismos para assegurar esta segurança são usados, entre os quais:

- O uso de mecanismos "anti-tear", no sentido de proteger a integridade dos dados em situações em que o dispositivo é removido do alcance da frequência rádio do POST, antes de se completar a transferência completa de dados;
- Garantia de integridade dos dados através de selos gerados por um ISAM;
- Uso de chaves de acesso, para controlar as permissões de escrita no CM e, como tal, proteger a integridade dos dados;
- Derivação de um número, único entre todos os CM, no sentido de se gerarem selos e chaves únicas.

O tempo que demora a efetuar uma transação é geralmente importante por várias razões. O mais óbvio será o aumento de tempo de espera dos utilizadores. Tempos de transação longos têm demonstrado em estudos, o aumento do risco dos CM serem prematuramente removidos antes da atualização de dados estar completa. Devido ao impacto no sistema e na respetiva confiabilidade provocado pelo tempo excessivo de operação, [ITSO Technical Specification 1000-10, 2010] define que, para cada tipo de operação, devem ser definidos tempos máximos de transação através de testes de desempenho.

3.4.2 **Arquitetura do CM**

Esta arquitetura permite o carregamento de vários IPE em vários tipos de CM, e o seu uso em todos os POSTs, num ambiente ITSO.

3.4.2.1 **CM ITSO Shells**

O termo CM é usado em detrimento do termo *smartcard*, já que no futuro, este termo pode abranger dispositivos como telefones móveis ou PDA´s. A arquitetura ITSO suporta a separação lógica entre os produtos e o dispositivo onde se encontra uma aplicação ITSO. Como tal, no contexto dos *smarcards*, a especificação adiciona uma terceira camada, a camada de produtos ao tradicional modelo de duas camadas, a Figura 8 ilustra este modelo.

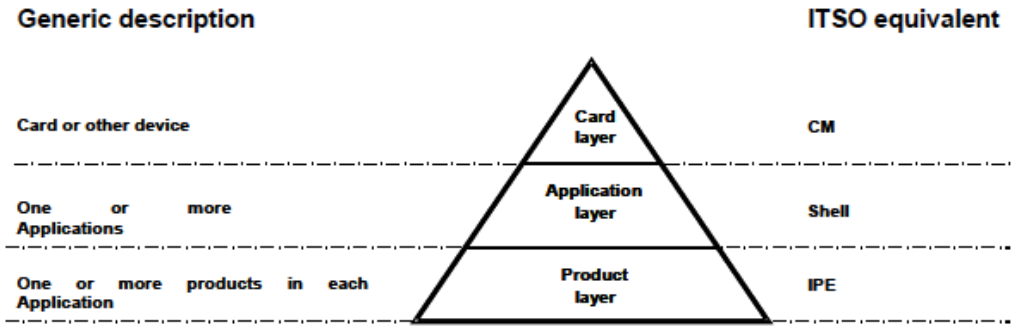

Figura 8 - O modelo ITSO de 3 camadas [ITSO Technical Specification 1000-0, 2010]

Isto significa que uma aplicação ITSO[10] (ITSO Shell) pode suportar múltiplos produtos ou, um produto ITSO pode ser integrado numa aplicação exterior. Esta flexibilidade maximiza a interoperabilidade que pode ser atingida.

A especificação ITSO também prevê a possibilidade dos operadores poderem escolher de entre múltiplos dispositivos, quais irão distribuir os seus produtos. Estes dispositivos vão desde CM´s capazes de conter apenas uma ITSO Shell compacta com um produto, a CM´s providos de uma ITSO Shell completa com múltiplos tipos de produtos. Isto encontra-se ilustrado na Figura 9.

[10] Renomeado para Shell para evitar confusão com o modelo tradicional.

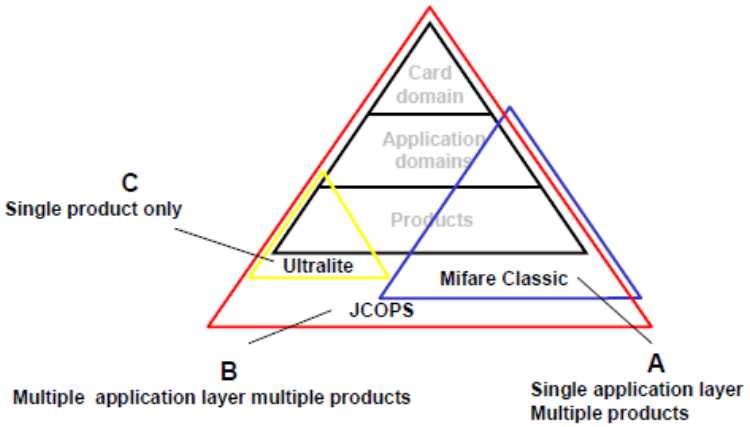

Figura 9 - Exemplo dos vários tipos de *smartcards* [ITSO TS 1000-0, 2010]

3.4.2.2 Grupos de Dados

Todos os dados contidos numa ITSO Shell encontram-se contidos em grupos de dados. Estes contêm: um identificador do proprietário e do tipo de grupo, um conjunto alojando o conteúdo, um identificador da instância e, por fim um selo que fornece um meio de verificação da autenticidade de todos os dados. Esta estrutura genérica, encontra-se ilustrada na Figura 10.

Figura 10 – Grupos de Dados [ITSO TS 1000-0, 2010]

3.4.2.3 Produtos contidos nas ITSO Shells

Como já foi referido uma ITSO Shell pode conter vários produtos. Cada produto possui um identificador que descreve o proprietário, o tipo, subtipo e validade. Os identificadores estão listados numa "diretoria". Esta é utilizada por uma aplicação POST, no sentido de se verificar se algum produto contido na mesma é aceite pelo POST [ITSO Technical Specification 1000-10, 2010]. Todos os produtos e "diretorias" contêm selos, os quais fornecem um meio para que uma aplicação POST possa verificar a sua autenticidade. Estes selos estão ligados ao CM através de métodos criptográficos. Esta relação entre identificadores, produtos e "diretorias" é ilustrado na Figura 11.

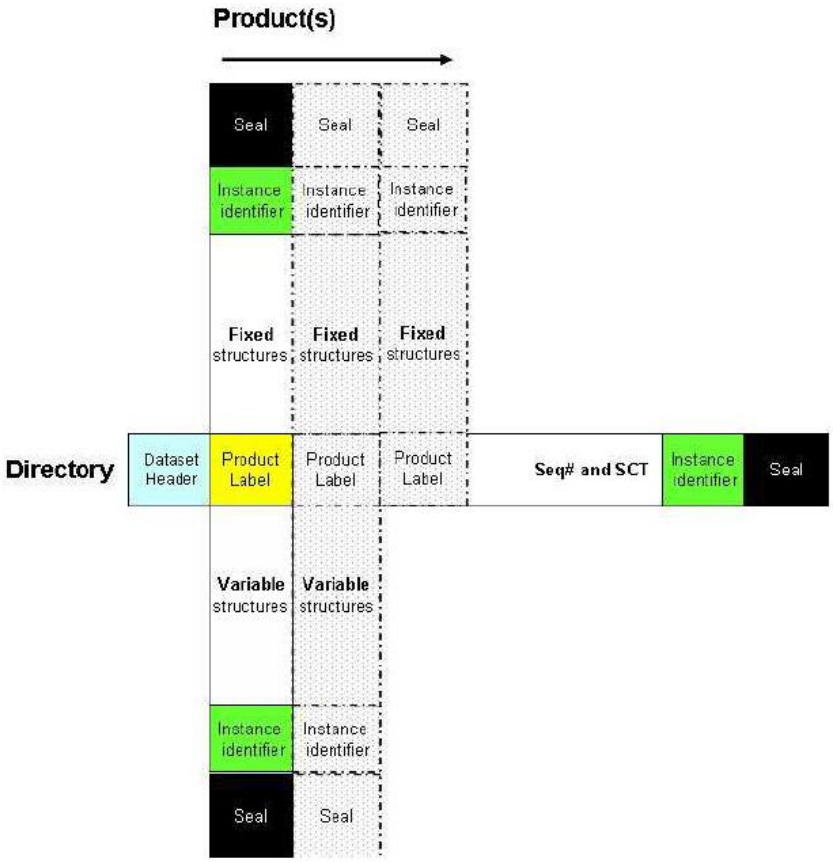

Figura 11 - Estrutura da "diretoria" e a sua relação com os Produtos [ITSO Technical Specification 1000-0, 2010]

3.4.3 **Dados dos Produtos contidos numa ITSO Shell**

Em [ITSO Technical Specification 1000-5, 2010] são definidas duas grandes classes de dados: IPE´s e Transient Ticket Records (**TTR**). As IPE´s são usadas para armazenar produtos, tais como, títulos de transporte, títulos de concessão, direitos de viagem ou vouchers por exemplo. Esta classe é usada quando um produto é vendido antecipadamente, ou quando é necessário que o produto seja armazenado durante um grande intervalo de tempo, como acontece por exemplo nos produtos sazonais. Existem vários tipos de IPE´s, cada qual definido por um código TYP. Os TTR são usados quando se vende um título apenas no âmbito de uma viagem, ou seja, armazenam títulos temporários e registos de eventos específicos.

3.4.4 **Point of Service Terminal (POST)**

[ITSO Technical Specification 1000-0, 2010] refere que qualquer POST numa estrutura ITSO pode ser visto como um terminal que permite efetuar transações com um CM. As entidades que interagem com estes são:

- Agentes de venda de produtos: Venda e recarregamento de instâncias IPE;
- Operadores: Uso e validação de instâncias IPE;
- Clientes: Operações de aquisição;
- Agentes de venda de Shell´s: Venda e recarregamento de uma ITSO Shell num CM.

Os POST fornecem os mecanismos de acesso aos dispositivos e garantem a interoperabilidade, ao interagiram com vários tipos de dispositivos de bilhética de diferentes operadores ou com produtos de múltiplos proprietários. [ITSO Technical Specification 1000-3, 2010] afirma que todos os POST contêm os seguintes atributos: capacidade de ler e escrever dados em *smartcard*s compatíveis com a especificação ITSO, troca de dados periódicos com um sistema HOPS e inclusão de um ISAM único entre todos os POST. Refere-se também que, para fornecer um elevado nível de interoperabilidade, devem verificar-se os requisitos funcionais:

- Meio de comunicação:
    - Validação e deteção da aplicação ITSO;
    - Validação da "diretoria";
    - Seleção de produtos;
    - Tempos de transação;
    - Implementação de mecanismos "anti-tear".
- Manipulação dos IPE´s
    - Processamento das instâncias IPE;
    - Eliminação de instâncias IPE;
    - Atualização de registos cíclicos;
    - Renovação automática;
    - Processamento de STR;
    - Anulamento de transações bilhéticas;

    - Impressão de títulos e recibos.
- Geração e processamento de mensagens
- Processos de configuração

3.4.5 **ITSO BackOffice (HOPS)**

A entidade conceptual de HOPS definida em [ITSO Technical Specification 1000-4, 2010], refere-se ao sistema central de processamento e armazenamento de dados. São apenas definidos os aspetos que possam afetar a interoperabilidade entre HOPS, permitindo assim alguma flexibilidade nas suas implementações. Os requisitos funcionais apresentados são:

- Gestão de comunicações;
- Armazenamento de todos os dados recebidos e transmitidos com a finalidade de posteriormente se proceder à sua recuperação ou auditoria;
- Gestão quer de contas IPE, quer de aplicações ITSO;
- Gestão de ativos (POST´s, HOPS´s, ISAM´s, HSAM´s), incluindo o seu estado localização e segurança. Qualquer HOPS terá que possuir um dispositivo de segurança – HSAM, que na sua forma mais simples pode ser equiparado a um único ISAM;
- Serviços:
    - Auditoria;
    - Cumprimento de regras;
    - Monitorização de parâmetros de segurança;
    - Cópias de segurança;
    - Arquivamento.

3.4.6 **Arquitetura de Segurança**

A arquitetura de segurança definida em [ITSO Technical Specification 1000-7, 2010], foi implementada de forma a permitir a todos os membros ITSO e utilizadores de *smartcards* a confiança de que apenas produtos genuínos possam ser admitidas pelos respetivos operadores. As transações geradas a partir de qualquer reclamação ou repartição de receitas, se necessário, podem ser autenticadas e auditadas pelo proprietário do produto ou ITSO Shell, em conformidade com o utilizador do *smartcard*. Foi também desenhada para garantir a interoperabilidade de uma forma equitativa, oferecendo aos membros a possibilidade de usar várias plataformas de *smartcards* de variada capacidade/custo. Os produtos e transações, quando criados, são selados, podendo ser autenticados sobre o seu estado de alteração e autenticidade por qualquer membro do projeto ITSO.

A ITSO desenvolveu um SSS, implementando segurança em toda a estrutura da arquitetura, desde da plataforma CM, ao POST e por fim o HOPS. O SSS pode ser utilizado por qualquer entidade pertencente ao projeto ITSO, independentemente da sua função. Esta arquitetura engloba vários elementos: numeração única dos produtos, transações e ITSO Shells, esquema de numeração interoperável com outros sectores de negócio, produtos com selos vitalícios para elevar a confiança dos operadores, processos de transação com garantia de entrega e privacidade do titular do CM.

3.4.6.1 **Numeração**

Todos os produtos são numerados de forma única, isto assegura o apuramento de responsabilidades e a possibilidade de auditorias num ambiente interoperável. A base do sistema é o International Issue Number (**IIN**), registado de acordo com as normas ISO, isto assegura a unicidade mundial, caso se pretenda interoperabilidade com projetos semelhantes ou outros setores de negócio. A hierarquia desta numeração é ilustrada na Figura 12.

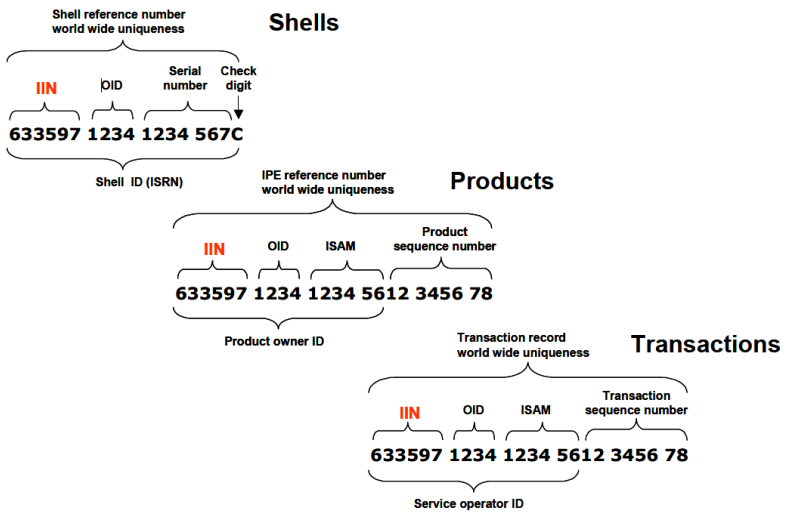

Figura 12 - Hierarquia na numeração ITSO [ITSO TS 1000-7, 2010]

3.4.6.2 **Chaves e selos**

Cada produto possui no mínimo um selo, o que fornece uma garantia de autenticidade criptográfica. Este selo é vitalício e adicionado na conceção do produto. O proprietário controla os membros (agentes) com permissões para criar e adicionar selos aos produtos, e os operadores que podem aceitar e verificar os produtos ao instruir o ITSO Security Management Service (**ISMS**). A autenticação do selo oferece aos operadores a confiança de que o produto é genuíno e inalterado.

As chaves criadas no processo de aceder às várias plataformas CM ou para autenticar a "diretoria", são distribuídas de forma semelhante, mas à disposição do proprietário da ITSO Shell. Todos os produtos e elementos da "diretoria" são agrupados em grupos de dados criptograficamente selados. Os selos apenas podem ser gerados pelo SSS.

3.4.6.3 **Perfis**

Os perfis dos produtos são mantidos nos POST´s, podendo ser retirados de qualquer HOPS. Este mecanismo permite aos proprietários dos produtos e operadores a flexibilidade de assegurar de que cada POST é apto a:

- Atuar como agente, apenas de certos produtos;
- Produção de um número máximo de títulos antes de um *reset*;

- Permissões de adicionar STR aos produtos.

3.4.6.4 **Lossless Transaction Records**

Um registo de uma transação é numerado, envolto e selado numa estrutura de dados. Cada registo transporta a identidade do dispositivo e um número de sequência, incrementado por uma unidade para cada registo selado. O HOPS ao receber a estrutura, verifica o selo para autenticar a mensagem. É mantido um cabeçalho para cada grupo de registo de transação criado, e retido até se receber um criptograma para o apagar. Grupos ativos indicam que determinado registo de transação deve ser retransmitido até a entidade recetora conseguir aceitar o grupo. Este mecanismo permite detetar a existência de registos duplicados, alterados ou fora de sequência.

3.4.6.5 **Privacidade do titular do CM**

Nas transações em que esteja presente o ITSO Shell Environment Identification Number (**ISRN**), este é encriptado de forma a garantir que apenas o proprietário da Shell possa associar a transação a um determinado utilizador.

3.4.6.6 **Funcionalidades Recentes**

O ISMS pode distribuir configurações e código que permitam ao ambiente de segurança ser atualizado caso:

- Sejam adicionados novos métodos de acesso ao CM;
- Introdução de novos modelos IPE.

Este mecanismo é executado através de mensagens seguras, evitando o *tampering* (malicioso ou acidental), e ainda que a correspetiva informação não possa ser interpretada fora do ambiente seguro.

3.4.7 **O Subsistema de Segurança (ISAM)**

Para que seja implementada uma arquitetura de segurança, todos os membros do projeto devem implementar um SSS com as seguintes características: possuir chaves secretas, contadores de sequências que não possam ser modificáveis por POST´s e transportar configurações e dados de forma interoperável.

Um SSS é fornecido sobre forma de um ITSO Secure Application Module (**ISAM**), construído a partir de um *smartcard* programável com memória extensível. A inclusão de um ISAM é obrigatória para todos os POST´s. Este é controlado por uma aplicação POST ou HOPS, implementando toda a segurança ao administrar: CM, produtos, mensagens ISMS e mensagens HOPS. As funções e respetivas funcionalidades de um ISAM são descritas na Tabela 5.

Tabela 5 - Funcionalidades de um ISAM

Função	Funcionalidade
Administração do CM	O ISAM interage com o CM através de um aplicação POST para garantir: • O acesso a todas as plataformas específicas pela ITSO; • Estabelecimento de uma sessão apropriada a cada plataforma; • Integração da aplicação ITSO como parte da gestão de uma sessão;
Administração de Produtos	O ISAM interage com uma aplicação POST para garantir: • Autenticidade de produtos e "diretorias"; • Qualquer modificação relativa a produtos é apenas permitida caso esteja identificada pelo ID do POST do operador de serviço; • Qualquer produto, é concebido com o OID do proprietário, o número de sequência e selado,
Administração de mensagens ISMS	O ISAM interage com o ISMS através de uma aplicação POST. Estas mensagens são transmitidas ao ISAM quando este necessita de: • Criar ou modificar chaves, usadas para aceder ao CM/"grupo de dados"; • Modificar chaves usadas para gerar informações de transações; • Adicionar ou modificar novas funções ao SSS. Desativar o SSS;
Administração de mensagens ISAM em POST	Serviços fornecidos pelo ISAM: • Assegurar a identificação única dos registos de transação, pelo ID do POST e um número de sequência; • Assegurar que os cabeçalhos dos grupos são corretamente geridos; • Armazenar uma Hot/Action List; • Assegura que as mensagens de Hot/Action List recebidas possam ser autenticadas; • Confirma mudanças à configuração interna, instruídas pelo HOPS; • Confere segurança a todas as mensagens que não façam parte do sistema de registo de transações.
Administração de mensagens	Um ISAM instalado num HOPS interage com um POST ou outro HOPS através de uma aplicação HOPS, e assegura: • Geração de mensagens criptográficas enviadas aos POST´s;

ISAM em HOPS	• Mensagens gerais e de Hot/Action List enviadas a POST´s possam ser seladas; • Autenticação de mensagens enviadas de POST´s; • A que mensagens enviadas a outros HOPS possam ser seladas; • Autenticação de mensagens enviadas por outros HOPS´s; • Geração de mensagens seguras de criação e eliminação de produtos.

3.4.8 **Comunicações**

Em [ITSO Technical Specification 1000-9, 2010] definem-se os dados e os mecanismos de comunicação que permitem o desenvolvimento de um sistema robusto, seguro e interoperável. Os requisitos apresentados são mínimos de forma a permitir alguma flexibilidade aos fornecedores de equipamentos e operadores, por forma a se utilizar as infraestruturas e tecnologias existentes.

Um dos requisitos fundamentais definidos é a garantia de entrega de dados ao nível da aplicação. Isto requer a retenção das mensagens por parte do transmissor, até receber a confirmação das mesmas. Se tal não acontecer num dado intervalo de tempo, as mensagens são retransmitidas.

O conteúdo das mensagens é definido em [ITSO Technical Specification 1000-6, 2010]. O XML é o formato padrão de transmissão [ITSO Technical Specification 1000-9, 2010], embora não seja obrigatório nas comunicações realizadas entre POST e HOPS. Os HOPS estão interligados numa rede ponto a ponto, usando uma Virtual Private Network (**VPN**). É necessária a implementação de uma estrutura de comunicações separada entre os HOPS´s e o ISMS. Existem quatro classes de mensagens aplicacionais [ITSO Technical Specification 1000-9, 2010]:

- Classe 0: Mensagens de *acknowledgements* positivos;
- Classe 1: Mensagens de grupo POST-to-HOPS;
- Classe 2: Mensagens genéricas;
- Classe 3: Mensagens de segurança ISAM.

Os mecanismos de comunicação foram projetados para controlar a transmissão segura de mensagens "*end-to-end*", mesmo quando estas necessitam de transpor camadas

específicas de comunicação. Estes mecanismos também protegem mensagens encaminhadas por HOPS intermédios.

A confidencialidade dos dados transmitidos entre HOPS é assegurada pelo uso de uma VPN. A VPN fornece a encriptação padrão de todos os dados. Os dados transmitidos entre HOPS que não necessitem do uso de VPN, a sua confidencialidade é assegurada pelo proprietário do projeto ou operador. Para as operações normais de troca de dados entre HOPS´s e POST´s, a confidencialidade também é assegurada pelo operador (através de uso de HTTPS por exemplo).

A integridade dos dados é assegurada pelo uso de um selo em cada estrutura de dados, constituinte de cada mensagem aplicacional.

A autenticidade dos dados transmitidos é assegurada pelo uso de um número de sequência, uma chave e selo em cada estrutura de dados constituintes de cada mensagem aplicacional.

A não-repudiação dos dados transmitidos é assegurada por um Timestamp, DealerID e Selo em cada estrutura de dados constituinte de cada mensagem aplicacional.

A arquitetura de comunicação permite a troca segura de chaves e outros dados relativos à segurança entre o ISMS, os Asset Management Systems (**AMS**) e ISAM´s

3.4.9 **Mensagens de Dados ITSO**

As mensagens de dados encontram-se definidas em [ITSO Technical Specification 1000-6, 2010], enquanto as relativas com a segurança estão presentes em [ITSO Technical Specification 1000-8, 2010]. O Tabela 6 apresenta alguns exemplos.

Tabela 6 - Tipos de mensagens ITSO [ITSO TS 1000-6, 2010]

Tipo de Mensagem	**Grupo**
Transaction Record data	ITSO shell, IPE Administration, Card issuer messages
Transaction Record data	Stored Travel Rights
Other Message Data	POST to HOPS queries
Other Message Data	HOPS to HOPS messages
HOPS to POST /HOPS messages	Message control
HOPS to POST /HOPS messages	HOPS Response to POST queries

4. Modelação dos Sistemas de Bilhética Eletrónicos

Neste capítulo é modelado o problema dos sistemas de bilhética eletrónica, de modo a estruturar um *Web Service* contendo um conjunto de serviços/métodos modulares, que possam representar as principais operações de bilhética, usando a infraestrutura da Amazon para criar um ambiente semelhante à aproximação proposta no projeto SmartCITIES, TaaS ou seja uma solução de serviços modulares de bilhética implementados numa Nuvem computacional. O *Web Service* criado (SmartSales) é publicado na infraestrutura desenvolvida na AWS descrita no próximo capítulo.

Os terminais de bilhética (POST´s) são analisados através das variáveis identificadas: operacionalidade do equipamento (terminais estáticos ou móveis), a sua função (validação, compra ou recarregamento), ambiente geográfico de utilização ou características de rede onde se podem encontrar inseridos.

Foi desenvolvido um protótipo de um *Web Service* (SmartSales) que disponibiliza diversos serviços para emular as três operações fundamentais identificadas neste modelo (validação, compra/recarregamento, autenticação), através de métodos desenvolvidos para o efeito.

Os modelos foram concebidos em conjunto com especialistas da empresa Link, a partir de discussões a respeito das características mais importantes dos mesmos, e de quais as relações estritamente necessárias. Baseado nas conclusões obtidas, foi possível construir um protótipo fiável no que respeita às comunicações efetuadas por um serviço de bilhética.

Este protótipo pretende apenas focar-se na comunicação entre os terminais e um serviço central, abstraindo-se de todos os pressupostos inerentes à comunicação entre o terminal e o CM (*smartcard*), ou seja entre outros, é assumido que o utilizador se encontra devidamente autenticado perante o terminal.

4.1 Modelização do Problema

4.1.1 **Primeira fase**

Com o objetivo de atender às necessidades referidas, procedeu-se numa primeira fase à análise dos vários terminais envolvidos no processo e às características específicas dos mesmos. Tornou-se clara a necessidade de divisão destes em várias classes, tendo em conta: as operações efetuadas; a mobilidade; os requisitos temporais ou ambiente onde se inserem. Assim realizou-se a divisão em 3 classes de equipamentos:

- **Terminais de Venda/Recarregamento**: Esta classe de terminais enquadra-se nos terminais estáticos, onde se processam as operações de venda e recarregamento de títulos de transporte. Embora estas operações sejam mais complexas, ou seja, envolvam uma grande troca de parâmetros e métodos associados, não são tão usadas como uma operação de validação. Para o exemplo de um título de transporte anual, segundo a Link, por cada venda existem cerca de 12 recarregamentos. Apresentam-se como terminais cuja localização se encontra normalmente em estações com uma razoável capacidade de acesso à rede e, por não interferirem diretamente no acesso dos clientes ao serviço de transporte, pode-se considerar que os requisitos temporais, apesar de importantes, não se apresentam como um fator critico;
- **Terminais de Validação Estáticos**: Englobaram-se nesta classe, os equipamentos que desempenham operações de validação presentes no acesso dos clientes ao serviço de transporte. Segundo a Link, este tipo de equipamento apesar de efetuar uma operação relativamente simples, a mesma é realizada inúmeras vezes, cerca de 30 a 40 vezes por recarregamento. Isto, aliado ao seu posicionamento interferir com o acesso ao serviço de transporte, torna-os muito sensíveis a requisitos temporais, pelo que deveram apresentar valores inferiores a 300 milissegundos por validação. As características de rede associadas diferem do tipo de operadora. Para o caso de uma operadora de transportes rodoviários, a sua localização encontra-se dentro do meio de transporte o que os torna acessíveis apenas por 3G ou 4G. No caso de uma operadora de transportes

metropolitanos, a sua localização é dentro de estações onde o acesso se efetua normalmente por Ethernet (802.3) ou Wi-Fi (802.11);

- **Terminais de Validação Móveis**: Este tipo de terminais corresponde aos PDA´s usados pelos agentes de fiscalização, cuja utilização e função, requerem que estes sejam móveis. O grau de utilização é esporádico e por ser um equipamento móvel, a ligação à rede efetua-se por Wi-Fi quando a fiscalização é realizada nas estações, ou por 3G e 4G quando fora destas. Dada a natureza da operação de fiscalização e da não interrupção do fluxo no serviço de transporte durante a realização desta operação, os requisitos temporais associados são relativamente baixos.

Na Tabela 7 exibe-se o resumo das principais características identificadas em cada uma das classes analisadas.

Tabela 7 – Principais características de cada classe de terminal

Classe de Terminal	Operações	Mobilidade	Requisitos Temporais	Carga de Transções	Tipo de Rede
Venda /Recarregamento	Venda e Recarregamento	Não	Moderados	Baixa a Moderada	802.3 e 802.11
Validação Estático	Validação	Não	Elevados	Elevada	Todos
Validação Móvel	Validação	Sim	Baixos	Baixa	3G e 4G

De referir que neste trabalho, apenas é analisada a operação de recarregamento da classe de terminais de venda/recarregamento, divido à redundância entre as duas operações. Também foi identificada a operação de autenticação, que segundo a Link é executada por cada classe de terminais no inicio/final do dia ao serviço central de *backoffice*.

4.1.2 **Segunda fase**

Numa segunda fase desta modelização, ponderou-se sobre a forma correta da realização dos testes de desempenho.

Numa operação de recarregamento, o interesse está em testar o desempenho de cada método individualmente e não o tempo total da operação. Isto porque a operação não é contínua, um utilizador ao efetuar um recarregamento faz várias interações com o terminal, e o terminal pode estar a processar as frações de segundo enquanto o utilizador pondera sobre a sua decisão. Assim, apenas convém analisar o desempenho dos vários métodos que possibilitam a apresentação da informação ao utilizador

Já nas operações de validação e autenticação é possível analisar o seu desempenho numa forma global, uma vez que apenas existe uma interação. Assim considerou-se analisar as referidas operações consoante o cenário desenvolvido no quinto capítulo, as características de rede envolvidas, *hardware* dos servidores e a carga de pedidos através dos métodos:

1) Operação Recarregamento – método GetCatalog;
2) Operação Recarregamento – método TransactionConfirm;
3) Operação Recarregamento – método TransactionDo;
4) Operação Validação;
5) Operação de Autenticação.

4.2 Protótipo Desenvolvido

Modelado o problema, foi desenvolvido um *Web Service* baseado no Omaha Client-Server Protocol[11] V3, ilustrado na Figura 13, disponibilizando os métodos necessários para os três tipos de operações bilhéticas distintas referidas anteriormente, nomeadamente, a autenticação, recarregamento e validação num contexto de uma "*thin app*" (visto ser em função das suas características, muito mais exigente em termos de desempenho que uma "*fat app*"), com o apoio de uma estrutura de dados desenvolvida no contexto de uma entidade emissora de títulos de transporte.

[11] https://code.google.com/p/omaha/wiki/ServerProtocol

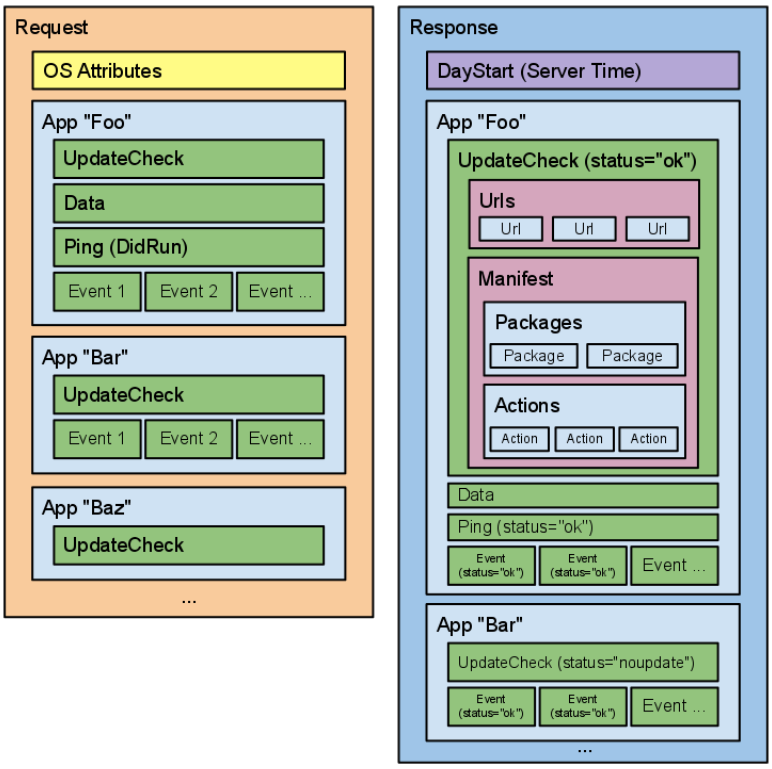

Figura 13 - Protocolo Omaha em [Google, 2011]

O *Web Service* SmartSales contém uma interface na qual são declarados todos os contratos, ou seja, as assinaturas dos métodos. Estes métodos são implementados numa outra classe, que pode ser considerada o ponto central do Web Service, pois é nesta que são recebidos os pedidos e devolvidas as respostas. Quando cada método é invocado são guardados dinamicamente todos os elementos contidos no XML do pedido numa classe respetiva. Uma operação bilhética pode ser constituída por um ou por um conjunto de vários métodos. De uma forma genérica o fluxo no *Web Service* consiste em: Após a invocação de um método e a informação contida nos elementos do pedido ser guardada, este comunica com uma classe que contém a lógica de negócio (Worker), cuja função é receber e enviar dados e aplicar a lógica de negócio. Ao receber os dados do método específico chama a camada de acesso aos dados (DAL) que executa *stored procedures* à base de dados. Ao receber a informação da base de dados o "Worker" pode então aplicar a lógica de negócio ao aplicar um parâmetro definido aos dados recebidos (por exemplo se for necessário efetuar um desconto, aplica uma redução de preço aos dados recebidos) e envia a informação de volta ao método correspondente que com esta informação gera o XML de resposta.

Descrição sumária do serviço SmartSales:

- O terminal (POST) envia pedidos ao servidor através de uma mensagem XML, contida num envelope SOAP submetida através de um POST HTTPS. A resposta é enviada através do corpo da mensagem XML;
- Cada pedido HTTPS pode enviar múltiplas ações, requisitadas pela aplicação cliente "*thin app*". O servidor responde com o estado e outras informações relevantes para cada ação. A resposta é organizada numa estrutura semelhante ao pedido XML.

O *Web Service* foi desenvolvido na *framework* Windows Communication Foundation (**WCF**), através da linguagem C#, com o intuito de ser publicado no Internet Information Services 7 (**IIS7**) presente nos *Web Servers* dos diferentes cenários descritos no quinto capítulo.

Os métodos desenvolvidos de suporte às operações consideradas, estão descritos em detalhe no Apêndice 9.3 e foram resumidamente os seguintes:

1) *SessionOpen*;
2) *SessionClose*;
3) *GetCatalog*;
4) *TransactionConfirm*;
5) *TransactionDo*;
6) *TransactionValidation*.

Como já foi referido, é importante analisar as comunicações em função de cada tipo de operação envolvida, isto porque, a cada operação correspondem diferentes cenários de uso com condições específicas.

Estas operações são descritas de seguida mais pormenorizadamente, através de diagramas de sequência para atribuir ênfase à sequência lógica em que as mensagens são trocadas, e também com o objetivo de se ilustrar a colaboração dinâmica entre os vários objetos e atores do sistema. Através destes diagramas também é possível perceber a sequência de mensagens enviadas entre os objetos ao longo do tempo.

4.2.1 **Operação de Autenticação**

Esta operação é invocada todos os dias automaticamente por cada terminal ao serviço central de *backoffice*, onde no início do dia tenta abrir uma sessão, e no final do respetivo dia executa o seu fecho. Se a autenticação for bem-sucedida, é retornada uma variável de sessão que persiste durante todo o dia, até se executar o fecho da mesma. A forma de autenticação que se utilizou foi a *basic access authentication*, que consiste em enviar através de HTTPS um utilizador e palavra-chave, previamente inseridos na base de dados do serviço central. Estas interações são descritas na Figura 14.

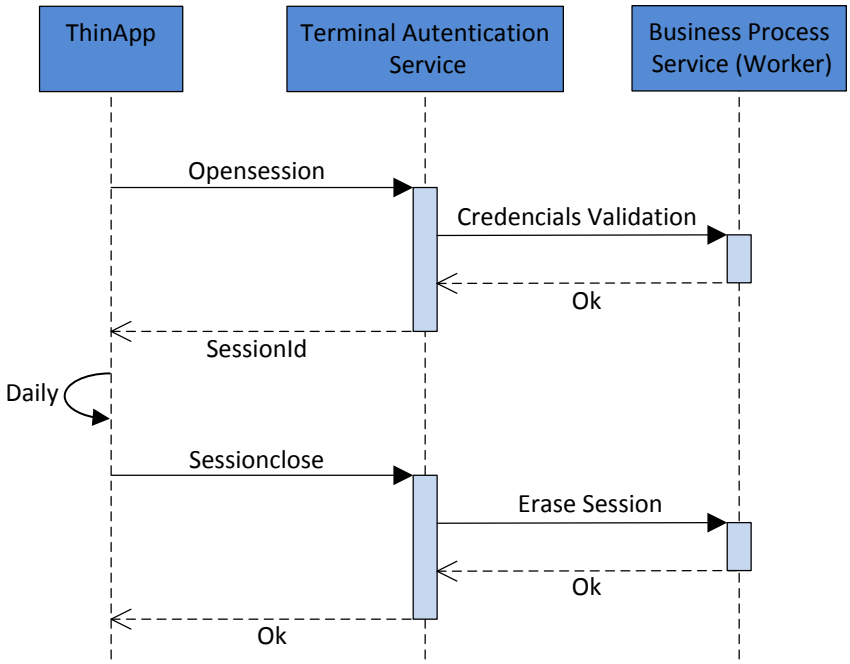

Figura 14 - Diagrama de sequência de uma operação de autenticação de um terminal

4.2.2 **Operação de Validação**

O caso de uso deste tipo de operação é relativamente simples, o utilizador aproxima-se do terminal, o título de transporte é detetado e é acionado o método *TransactionValidation* com o intuito de validar o título. Em caso afirmativo a porta do validador permite a passagem do utilizador.

Para acelerar a operação de validação, uma vez que esta apresenta um tempo crítico de execução, o método *TransactionValidation* não efetua a alteração (remoção da respetiva viagem) na base de dados quando o contrato é do tipo recarregável/1 viagem.

Os terminais vão armazenando os dados dos vários títulos validados, para que em determinados períodos com pouco movimento, sejam despoletadas para a Nuvem as atualizações à base de dados. O diagrama de sequência desta operação encontra-se representado na Figura 15.

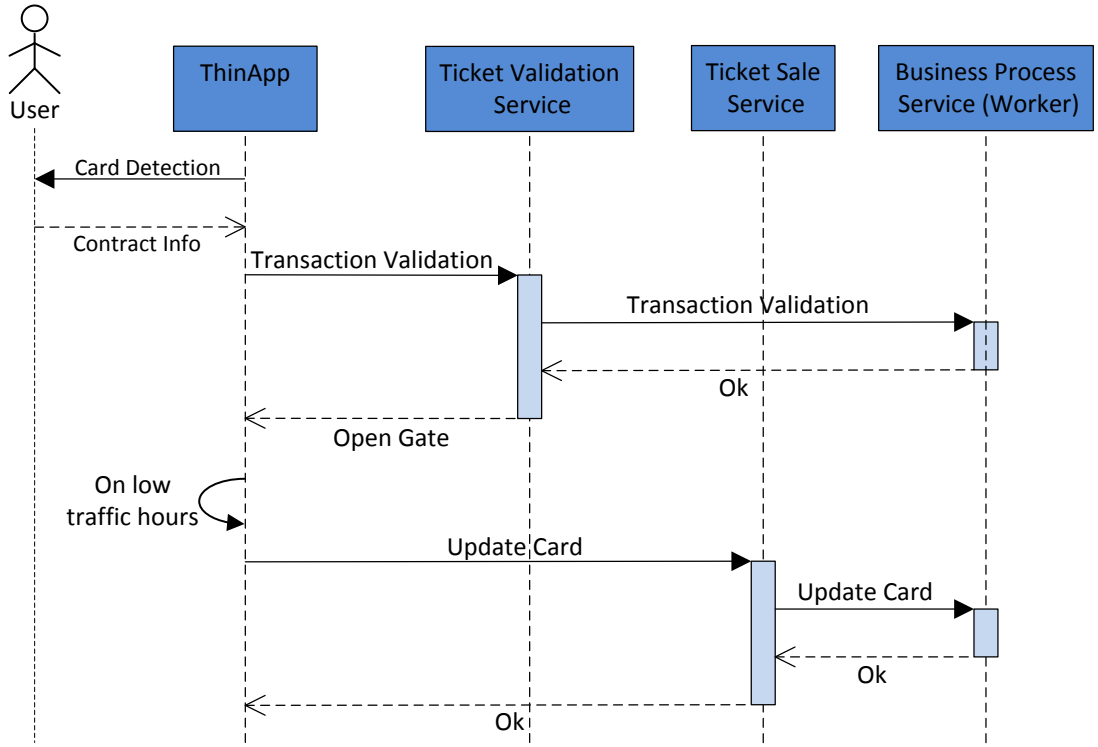

Figura 15 - Diagrama de sequência de uma operação de validação

4.2.3 **Operação de recarregamento**

Ao contrário da operação de validação, esta operação requer que o utilizador interaja com o terminal para que se inicie o processo de recarregamento. Após a apresentação do título por parte do utilizador, o terminal envia a informação presente no título ao serviço de Catálogo na Nuvem, que consoante o tipo de informação recebida (tipo de contrato), apresenta o catálogo correspondente. Depois da seleção do produto é invocado o método *TransactionConfirm* a fim de se verificar se o agente de venda tem autorização

(pode encontrar-se *blacklisted*) para efetuar a venda do produto escolhido. Em caso afirmativo e após pagamento, o método *TransactionDo* atualiza na base de dados o contrato com o produto selecionado. O diagrama de sequência desta operação encontra-se representado na Figura 16.

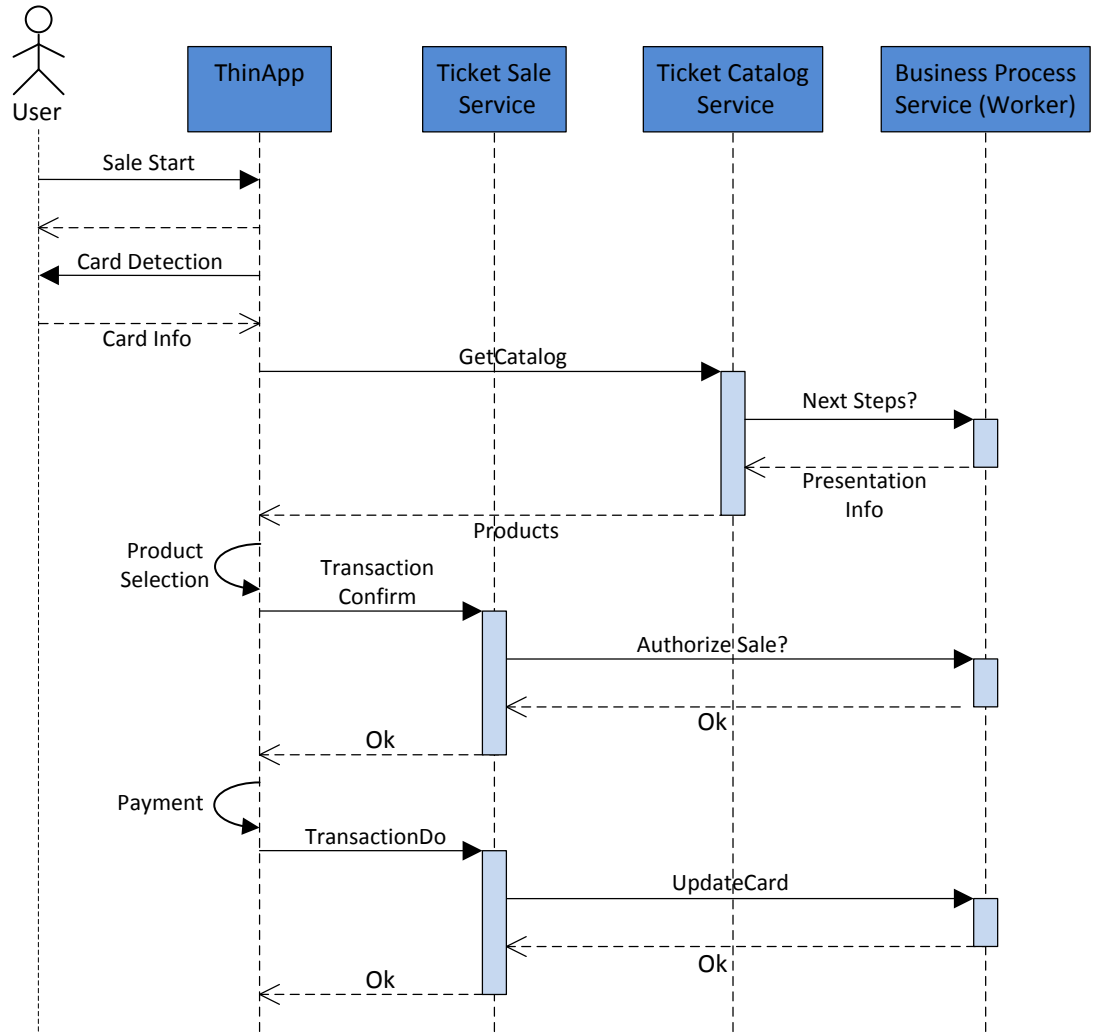

Figura 16 - Diagrama de sequência de uma operação de recarregamento

4.3 Modelo de Dados

O *Web Service* implementado teve como suporte à sua estrutura de dados um Catálogo suportado no Modelo Relacional, consistindo num repositório, gerido pelo Sistema de Gestão de Bases de Dados Microsoft SQL Server, contendo informação relativa a:

- Base de dados;
- Modelo lógico de dados da base de dados;
- Utilizadores da base de dados e seus direitos de acesso;
- Toda a restante informação necessária à definição e organização dos dados dentro da base de dados como:
    - Definição das tabelas;
    - Definição dos atributos das tabelas (colunas e seus tipos);
    - Mecanismo de chave para cada tabela;
    - Regras de integridade para as tabelas.

A informação relativa ao processo de autenticação dos terminais, nomeadamente as palavras-chave, presente na base de dados encontra-se armazenada de forma segura, para evitar ataques do tipo *information disclosure*. Esta forma de segurança é implementada recorrendo a um método disponibilizado pela Microsoft, que executa o *hashing* da palavra-chave inserida, e transforma o Message Digest (**MD**) obtido num inteiro. O valor correspondente à palavra-chave guardado na base de dados é este inteiro obtido. O diagrama Entidade-Associação referente ao modelo de dados é ilustrado no Apêndice 9.4.

5. Plataforma de Teste (Smartsales)

Neste capítulo foi implementada uma infraestrutura na Nuvem da AWS que servisse de suporte aos *Web Services* descritos no capítulo anterior. Usufruindo-se do tipo de Nuvem escolhido (IaaS), desenvolveram-se vários cenários com diversas arquiteturas, correspondendo a cada cenário uma topologia de rede com características específicas.

Ao utilizar-se uma plataforma computacional exterior às fronteiras de uma organização ou indivíduo, como neste caso o Amazon Ec2, é realizado um acordo com o respetivo fornecedor do serviço sobre as responsabilidades de cada parte interveniente, como seja o QoS ou segurança associada ao chamado "*last-mile"*.

É então necessário o profundo conhecimento dos recursos controlados e geridos pelo CSP, para poder-se decidir sobre o que ainda necessita de ser implementado. Em [Amazon Web Services, 2010] verifica-se que a AWS garante a percentagem de 99.98% de QoS em relação à disponibilidade dos seus Centros de Dados, e controla inúmeras medidas de segurança, as quais não vale a pena mencionar, por ser uma área bastante vasta. Assim, no próximo subcapítulo, realiza-se apenas menção aos parâmetros passíveis de controlo, e também uma análise às tecnologias patentes na plataforma AWS, na qual a infraestrutura de rede bilhética proposta é realizada.

5.1 Amazon Web Services

Das diversas soluções disponíveis de Nuvens, optou-se pelo desenvolvimento na Nuvem da Amazon Web Services, para completar outras aproximações desenvolvidas por outros projetos em curso no ISEL.

A AWS apresenta-se como líder de mercado no segmento IaaS, embora o seu foco inicial como empresa não tenha começado nesta área [Amazon Web Services, 2010]. Fornece um conjunto de serviços embutidos na sua plataforma computacional, construída e refinada ao longo dos anos, encontrando-se disponível para quem tenha acesso à Internet. A infraestrutura sobre a qual a AWS instalou todos os seus serviços apresenta como princípios fundamentais: a elasticidade da mesma e a segurança a esta

associada [Amazon Web Services, 2010]. O acesso aos serviços disponíveis na infraestrutura AWS é realizado através de uma *Web* API REST/SOAP, ou através de uma linha de comandos em Linux (AWS CLI). Este acesso requer autenticação para cada solicitação realizada à API de controlo para que somente os utilizadores autenticados possam aceder e gerir este serviço. As solicitações são assinadas com uma assinatura de Hash Message Authentication Code (**HMAC**) criada a partir da solicitação e da chave privada do utilizador.

Embora a AWS forneça uma enorme quantidade de serviços, nas próximas secções foi realizada referência apenas àqueles utilizados no desenvolvimento deste trabalho.

5.1.1 **Amazon Simple Storage Service (S3)**

O serviço Amazon Simple Storage Service (**S3**) permite o armazenamento flexível de dados na Internet. Pode ser visto como o serviço que armazena os dados com os quais todos os outros serviços trabalham. Apresenta três conceitos básicos: *buckets*, objetos e chaves. Um *bucket* é um simples contentor para os objetos armazenados no S3. Pode-se realizar a analogia deste, como sendo uma pasta ou uma diretoria num sistema de ficheiros, mas que pode ser acedida por um URL. Por exemplo, se o nome do objeto for *fotos/img.jpg* e estiver contido no "*bucket* joao*"*, este objeto pode ser acedido através do URL – *http://joao.s3.amazonaws.com/fotos/img.jpg*.

Um objeto representa o ficheiro que se pretende armazenar. Cada objeto é composto de duas entidades: dados e meta dados. Os dados representam o que se pretende guardar como documentos, fotografias, imagens de máquinas virtuais, vídeos, etc. Os meta dados descrevem o objeto associado através de pares chave-valor, gerados pelo utilizador na criação do objeto no S3.

Cada objeto armazenado num *bucket* é identificado através de uma chave única. Cada objeto tem associado apenas uma chave. O uso conjunto do *bucket* e chave, permite inequivocamente a identificação de cada objeto. No exemplo descrito anteriormente, o nome do *bucket* é joao e a chave *fotos/img.jpg*.

5.1.1.1 **Conceitos de segurança associados ao S3**

Como cada *bucket* e objeto gerado no S3 pertence à conta do utilizador onde foram gerados, é necessário conceder permissões aos utilizadores/clientes que pretendam aceder aos mesmos. Ao efetuar o armazenamento de dados neste serviço, deve-se ter em considerações os seguintes aspetos [Amazon Web Services, 2011]:

- Autenticação: Quando se envia um pedido ao serviço S3, este deve ser autenticado. Para o efeito é fornecido a cada utilizador uma chave e identificador únicos;
- Autorização: Para um utilizador aceder a determinado recurso, deve deter permissões para tal. Existe uma Access Control List (**ACL**) associada a cada objeto que identifica as permissões para o mesmo;
- Integridade: Cada pedido S3 deve ser assinado digitalmente com uma chave secreta. A assinatura digital é enviada em conjunto com o pedido;
- Encriptação: Pode-se aceder ao serviço S3 através do protocolo HTTPS, para assegurar que os dados são transmitidos por uma ligação encriptada;
- Não-Repudiação: A cada pedido S3 é associado um campo, que contêm o tempo e a data do mesmo, servindo como prova da transação.

5.1.2 **Amazon Elastic Compute Cloud (Ec2)**

O Amazon Elastic Compute Cloud (**Ec2**) é o principal serviço utilizado para a realização deste trabalho. Pode ser visto como o espaço virtual representativo da Nuvem. Fisicamente este espaço está associado ao serviço S3, ou seja a Centros de Dados espalhados pelo mundo em diferentes regiões geográficas.

É um *Web Service*, que permite distribuir uma mesma aplicação por N instâncias diferentes, permitindo assim o correto balanceamento e redundância em caso de falha [Amazon Web Services, 2012]. A aplicação é distribuída no conceito de servidor virtual (instância), independente do sistema operativo. Uma imagem de uma instância é armazenada no Amazon S3 e poderá ser multiplicada por N, conforme as necessidades.

A cada instância lançada no Ec2 é atribuído automaticamente um IP público e outro privado, diretamente mapeados através de Network Address Translation (**NAT**).

O endereço público atribuído é diferente de cada vez que se lança uma instância. Isto representa um problema no caso de se usar qualquer mapeamento dinâmico de DNS para interligar um nome domínio a um IP, visto que tal processo pode demorar 24 horas até que a mudança seja propagada através da Internet. Para resolver esta situação, pode-se adquirir um Elastic IP (**EIP**) que representa um IP estático associado à conta e que pode ser usado em qualquer instância.

Neste tipo de serviço, o conjunto de instâncias lançadas funciona num espaço considerado "público" para a Internet, ou seja, qualquer instância possui um endereço IP encaminhável na Internet. Isto, representa um problema de segurança quando se pretenda implementar uma infraestrutura de rede, já que, apesar de se poderem implementar restrições de acesso através de Grupos de Segurança e ACL´s, as instâncias podem ser alvo de ataques elaborados no sentido de revelar a topologia e serviços disponibilizados, como por exemplo *port-scanning*, ou mesmo ataques que visem a perturbação da qualidade dos serviços como ataques de *denial of dervice* (**DOS**).

De seguida, são referidos os principais conceitos associados ao Ec2, para que melhor se perceba o funcionamento deste serviço [Amazon Web Services, 2010].

5.1.2.1 **Availability zones**

A Amazon possui múltiplos Centros de Dados em localizações geográficas distintas. O primeiro grau de separação realizado é por regiões, correspondendo normalmente estas a continentes. Normalmente apenas existe uma região por continente, excetuando o caso da América do Norte que possui três regiões distintas. Cada região contém várias *Availability Zones* correspondendo a Centros de Dados onde todo o *hardware* que suporta a infraestrutura está localizado.

Cada zona é completamente autónoma e isolada das restantes na mesma região. Esta arquitetura permite que se possam lançar instâncias em diferentes regiões, dependendo dos clientes alvo que se prende alcançar. Permite também aos administradores lançarem instâncias a disponibilizar o mesmo serviço em diferentes zonas, conseguindo-se assim redundância contra possíveis falhas dentro de uma zona, como também balanceamento de carga. Na Figura 17 ilustra-se esta separação realizada pela AWS.

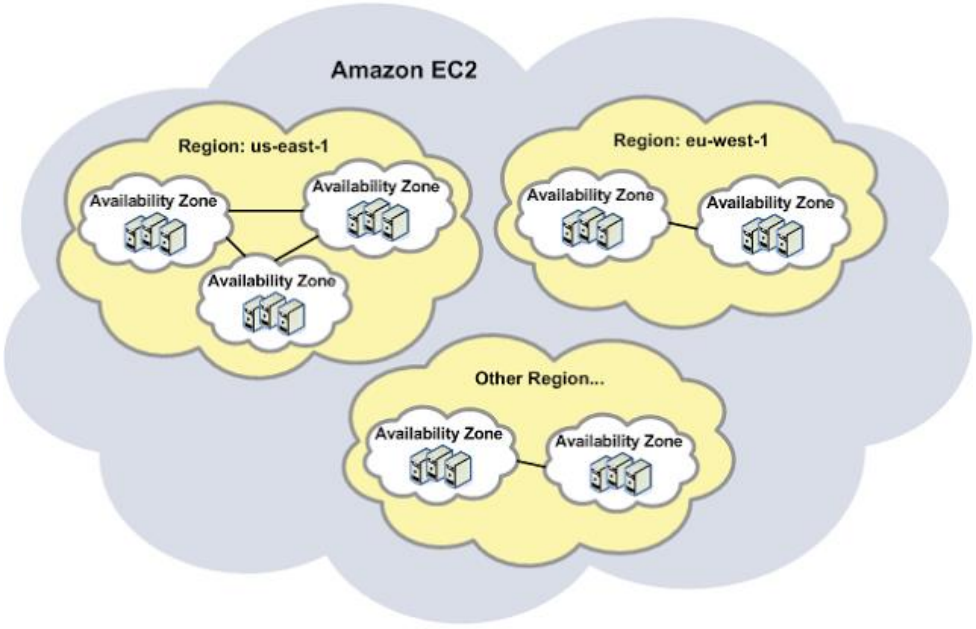

Figura 17 - Exemplo de algumas regiões e respetivas Availability Zones [Amazon Web Services, 2010]

5.1.2.2 **Amazon machine images (AMI)**

O termo Amazon Machine Images (**AMI**), designa uma imagem encriptada de uma máquina contendo toda a informação necessária ao arranque de instâncias, encontrando-se armazenadas no serviço S3. Por exemplo, uma AMI pode conter um Linux com o Apache instalado ou um Windows com o SQL Server. Pode-se optar por duas formas distintas quando se pretende usar uma AMI. Pode-se localizar uma pré-existente disponibilizada pela Amazon, ou pode-se criar uma nova AMI (apenas disponível para Linux e UNIX), efetuando o seu registo e correspetivo *upload* para o serviço S3. A Figura 18 ilustra este fluxo.

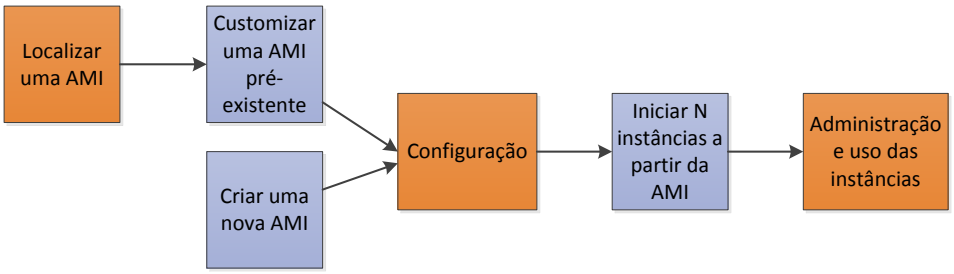

Figura 18 - Fluxo de iniciação de uma AMI

5.1.2.3 **Elastic block store**

O Elastic Block Store (**EBS**) permite a criação de discos que podem ser encaixados como dispositivos de bloco numa instância (em execução ou não). A qualquer momento podem ser criadas *snapshots* dos EBS, onde cada, representa o estado do disco num dado instante de tempo. Pode-se assim, armazenar os ficheiros e dados para lá do tempo de vida de uma instância. Quando for necessário o seu uso, pode-se montar novamente o disco EBS numa qualquer instância.

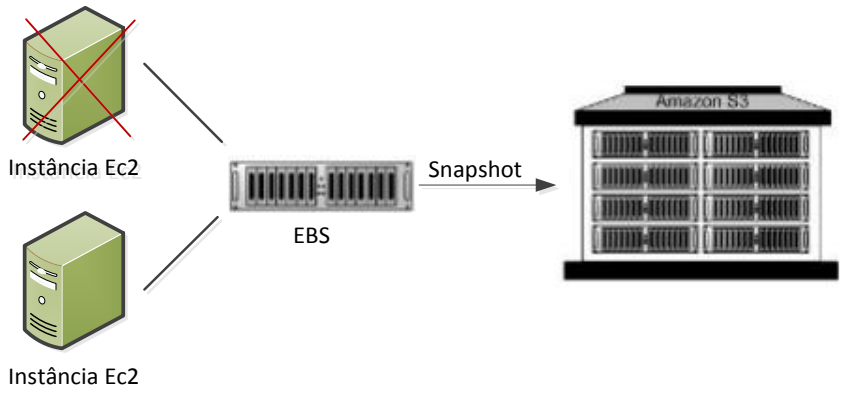

Figura 19 - Exemplo da vantagem do EBS em caso de falha de uma instância

Esta abordagem apresenta diversas vantagens. Em caso de falha de alguma instância, o disco EBS que contém todos os dados, desmonta-se automaticamente, é retirada uma *snapshot* e colocada no S3. Pode-se depois lançar uma nova instância a partir da *snapshot* guardada, recuperando-se assim de uma forma extremamente rápida de um erro. Este mesmo caso é ilustrado na Figura 19.

5.1.2.4 **Grupos de Segurança**

Qualquer instância lançada dentro do Ec2 pertence a um Grupo de Segurança. Podem-se associar diferentes instâncias a diferentes grupos. Estes não são mais que ACL´s, especificando o tipo de tráfego que poderá entrar ou sair da instância ou instâncias a que esteja associado. A Figura 20 ilustra um exemplo da aplicação destes. Os Grupos de Segurança pertencentes ao âmbito do Ec2 são bastante simples, podendo possuir configurações mais complexas, mas apenas quando criados no âmbito de um VPC.

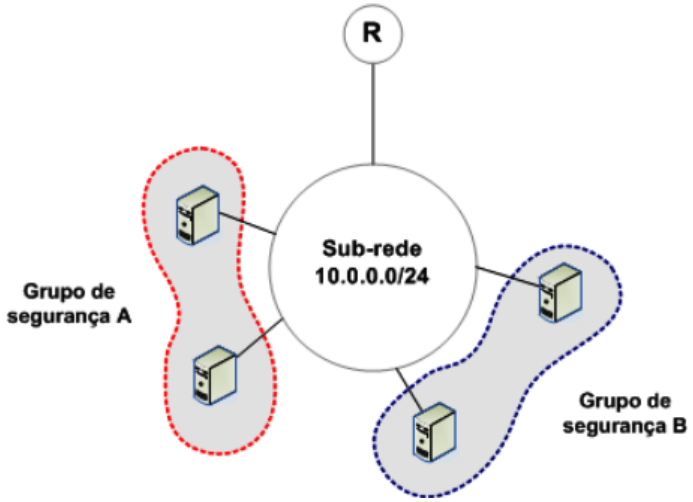

Figura 20 - Exemplo de Grupos de Segurança

Apesar de a maior parte dos serviços, como balanceamento de carga, monitorização ou acesso a serviços de Content Delivery Networks (**CDN**), estarem disponíveis no Ec2, o mesmo não permite a realização de uma infraestrutura de rede complexa, como a separação lógica de servidores em diferentes segmentos de rede – "*subnets*" e respetiva definição e configuração de tabelas de encaminhamento.

Estas opções avançadas são implementadas pelo serviço Virtual Private Cloud (**VPC**), descrito na próxima secção.

5.1.3 **Amazon Virtual Private Cloud (VPC)**

O Amazon VPC permite o uso de uma secção privada e isolada da Nuvem da AWS onde se podem executar recursos AWS numa rede virtual definida pelo utilizador [Amazon Web Services, 2012]. Com o Amazon VPC, pode-se definir uma topologia de rede virtual, que se assemelhe a uma rede tradicional idêntica à que se pode encontrar em Centros de Dados privados, assegurando um elevado grau de segurança como ilustra a Figura 21. É permitido o controlo total sobre um ambiente de rede virtual, incluindo a seleção de intervalos de endereços IP, criação de *subnets* e configuração de tabelas de encaminhamento e *gateways* de rede [Tavis, et al., 2012].

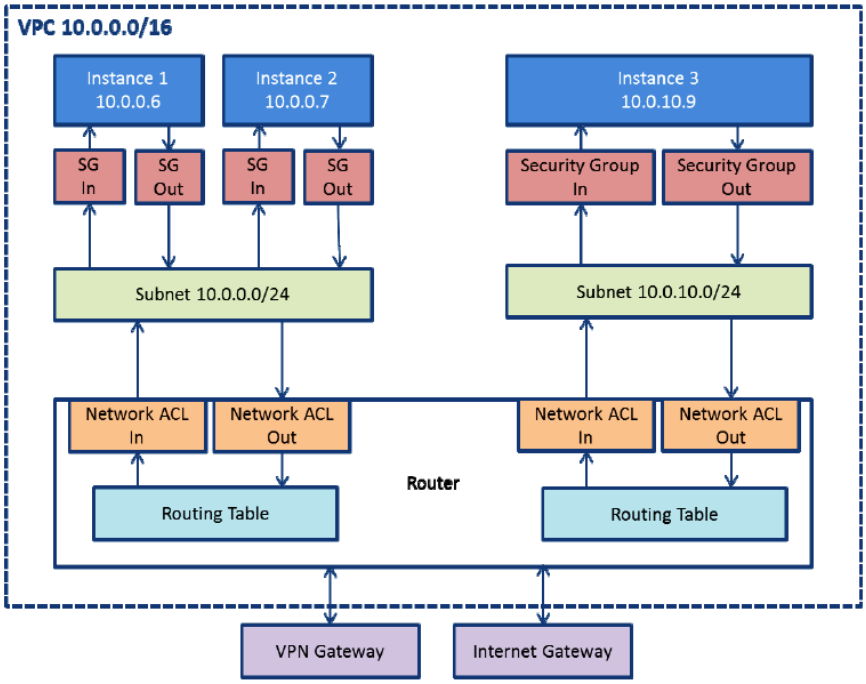

Figura 21 - Visão geral do processo de segurança da AWS numa VPC em [Amazon Web Services, 2012]

5.1.3.1 **Elastic Load Balacing**

O Elastic Load Balancing[12] (**ELB**) distribui automaticamente o tráfego de entrada dos aplicativos em várias instâncias do Amazon EC2. Permite uma maior tolerância a falhas nos aplicativos, fornecendo a capacidade de equilíbrio de carga necessária em resposta ao tráfego de entrada dos aplicativos. O ELB deteta instâncias com problemas de integridade dentro de um conjunto, e redireciona automaticamente o tráfego para instâncias íntegras, até que as instâncias com problemas sejam restauradas.

5.1.4 **Amazon CloudFront**

O Amazon CloudFront[13] é um *Web Service*, desenhado com o objetivo de replicar os dados requeridos em locais geograficamente distintos, para que estes transitem o mais rapidamente possível ao cliente final. O facto de a Amazon dispor de uma rede global, permite que o conteúdo esteja geograficamente mais próximo do utilizador final, evitando assim questões de desempenho.

[12] http://aws.amazon.com/pt/elasticloadbalancing
[13] http://aws.amazon.com/cloudfront

5.1.5 **Amazon CloudWatch**

O Amazon CloudWatch[14] é um *Web Service* que fornece a monitorização dos recursos da Nuvem da AWS, como por exemplo o EC2. Fornece visibilidade sobre a utilização de recursos, desempenho operacional e padrões de utilização geral, incluindo métricas como a utilização de Central Processing Unit (**CPU**), leituras e gravações do disco e tráfego de rede.

5.2 Cenários de Teste

Os vários cenários desenvolvidos, serviram de plataforma de publicação do serviço Smartsales desenvolvido no quarto capítulo. Realizando a analogia com a especificação ITSO, foi implementado um *backoffice* central contendo vários HOPS, no sentido de se realizarem testes à comunicação entre este e um terminal cliente (POST).

Cada modelo de Nuvem (pública, privado, híbrida) detém características específicas, diferindo na complexidade associada, segurança ou desempenho. A principal diferença entre os diferentes modelos, baseia-se no posicionamento da infraestrutura, ou seja, se esta se encontra antes/depois da *firewall* da organização no caso do modelo de Nuvem privada e pública, respetivamente. No caso do modelo híbrido, a infraestrutura possui recursos em ambos os setores.

A abordagem seguida foi a de implementar em cada cenário um diferente modelo de Nuvem (pública, privada e híbrida), com o objetivo de se possuir uma diversidade de topologias de rede e modelos, pensando-se conseguir com o proposto, uma maior variância de resultados para análise.

Todos os cenários foram implementados na região europeia da AWS (Dublin, Irlanda) em duas *Availability Zon*es distintas, usando-se AMI´s de Windows Server 2008 R2 para as instâncias dos servidores de base de dados e *Web*, e AMI´s de Linux Ubuntu Server 12.04 LTS para implementação da *firewall* organizacional, NatBox e balanceamento de carga interno.

[14] http://aws.amazon.com/pt/cloudwatch

A Amazon AWS disponibiliza diversos tipos de instâncias, encontrando-se o seu desempenho associado ao custo por hora de utilização, visto ser um sistema *"pay-per-use"*. No sentido de aproveitar a elasticidade fornecida pela Amazon AWS, na fase da construção das arquiteturas de rede, utilizaram-se as instâncias padrão (*m1.small*), aumentando-se a capacidade de acordo o tipo de serviços e desempenho prestado na fase de testes, culminando com a implementação de *Auto-Scaling*. Estes detalhes serão descritos mais pormenorizadamente no sexto capítulo.

5.2.1 **Primeiro cenário**

Este cenário corresponde a um modelo de Nuvem pública fazendo uso do Ec2, onde se implementou numa primeira fase, um *Web Server* e um servidor de base de dados diretamente na *Availability Zone A* da Ec2 como ilustrado na Figura 22. Cada servidor possui um IP público e um privado, sendo o privado apenas para comunicação entre o servidor e qualquer máquina presente na infraestrutura da AWS. Não existe qualquer tipo de balanceamento de carga nem controlo sobre o *gateway*.

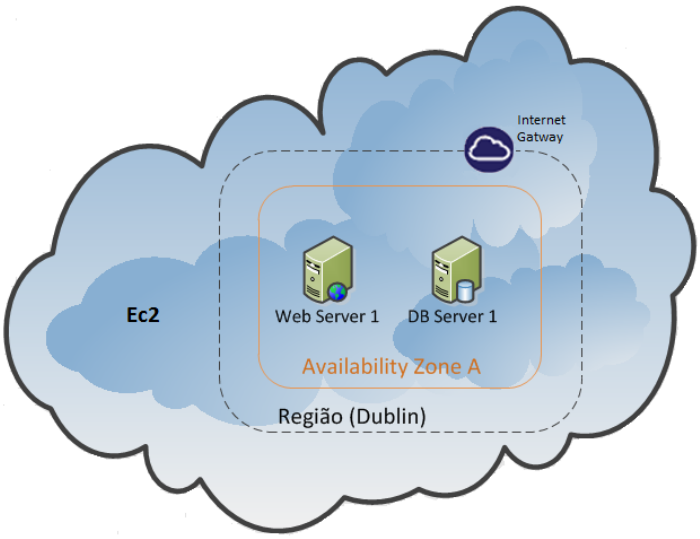

Figura 22 - Primeira fase do primeiro cenário proposto

Numa segunda fase, foram acrescentados mais servidores para adicionar redundância à infraestrutura e implementou-se o serviço de balanceamento de carga disponibilizado pela AWS (ELB), como ilustra a Figura 23. O ELB além de distribui automaticamente o tráfego de entrada para as várias instâncias do EC2, também permite detetar instâncias

com problemas de integridade, ao efetuar pedidos ICMP periodicamente a cada instância. Foi atribuído um endereço IP público ao ELB, e configurado para que, qualquer pedido destinado à sua porta 443 fosse redirecionado para os dois *Web Servers* através de balanceamento de carga em *weighted round robin*. De referir, que para todos os diagramas posteriormente apresentados é realizada a abstração da entidade Ec2, continuando a mesma sempre presente.

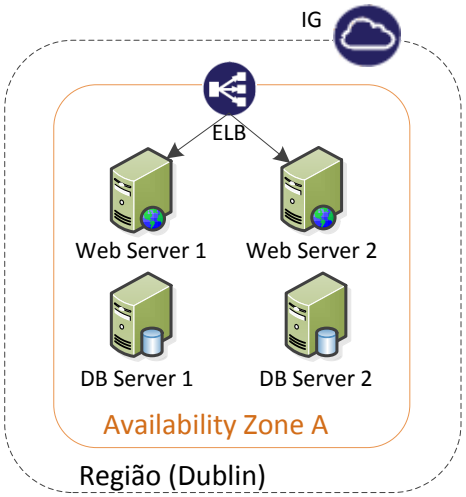

Figura 23 - Segunda fase do primeiro cenário proposto

No entanto esta topologia ainda estava dependente da disponibilidade da *Availability Zone* A. Se este Centro de Dados ficasse indisponível, toda a infraestrutura ficaria indisponível. Para acrescentar redundância ao nível dos Centros de Dados, implementou-se numa última fase a topologia presente na Figura 24.

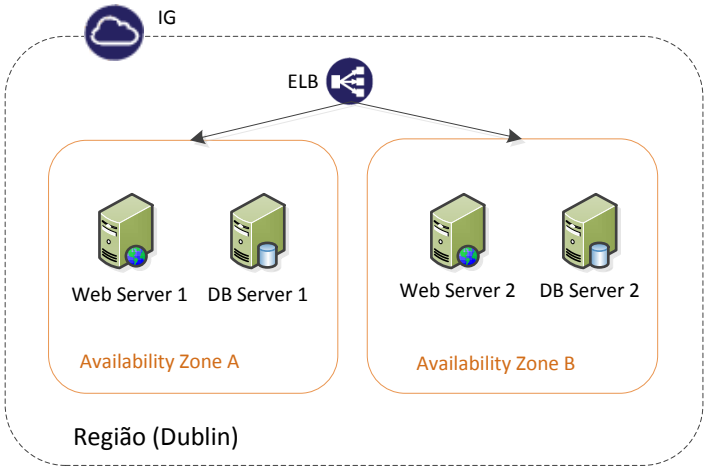

Figura 24 - Última fase do primeiro cenário proposto

Em qualquer das topologias apresentadas, a noção de *firewall* organizacional não está implementada, ficando-se dependente da filtragem efetuada pelo CSP. Qualquer das máquinas possui um endereço IP público, encontrando-se assim expostas para a Internet. Apesar de se conseguir um sistema robusto ao nível da redundância, o balanceamento de carga não é ideal e a segurança implementada é mínima, ficando-se nesta matéria completamente dependente do CSP.

5.2.2 **Segundo cenário**

Para a realização do segundo cenário, recorreu-se ao serviço VPC para estender as funcionalidades do Ec2, criando-se um segmento de Nuvem privado no Ec2. Neste segmento, foram definidas duas áreas, uma considerada pública/exterior à organização e outra privada/interna.

As máquinas pertencentes às *subnets* 10.0.1.0/24 e 10.0.11.0/24, apesar de se encontrarem numa *subnet* privada como é representado na Figura 25, dispõem de endereços privados pertencentes à respetiva *subnet*, e de endereços públicos, recorrendo-se ao uso dos EIP.

A AWS recorre a uma variante de NAT (Static NAT) [Cisco Systems, 2007] para a implementação dos EIP, realizando o mapeamento no *router*/*internet-gateway* que define a VPC. A título de exemplo, ilustram-se duas situações:

- **Comunicação no sentido da Internet para um *Web Server***: Ao receber este tráfego, o *router* verifica o endereço de destino, consulta a tabela de NAT, que mapeia o endereço publico do *Web Server* com o seu respetivo privado e envia para este;
- **Comunicação no sentido de um *Web Server* para a Internet**: O *Web Server* envia do seu endereço privado para o seu *default-gateway* (*router*), que ao receber, verifica na tabela de NAT o respetivo mapeamento, e envia para a Internet como endereço de origem o endereço público do respetivo mapeamento.

Portanto, apesar das máquinas presentes nas *subnets* 10.0.1.0/24 e 10.0.11.0/24 se encontrarem numa rede privada, a *firewall* por onde passa o tráfego proveniente do

exterior bem como o mapeamento efetuado por NAT é controlado exclusivamente pelo CSP, modelando esta infraestrutura numa Nuvem híbrida.

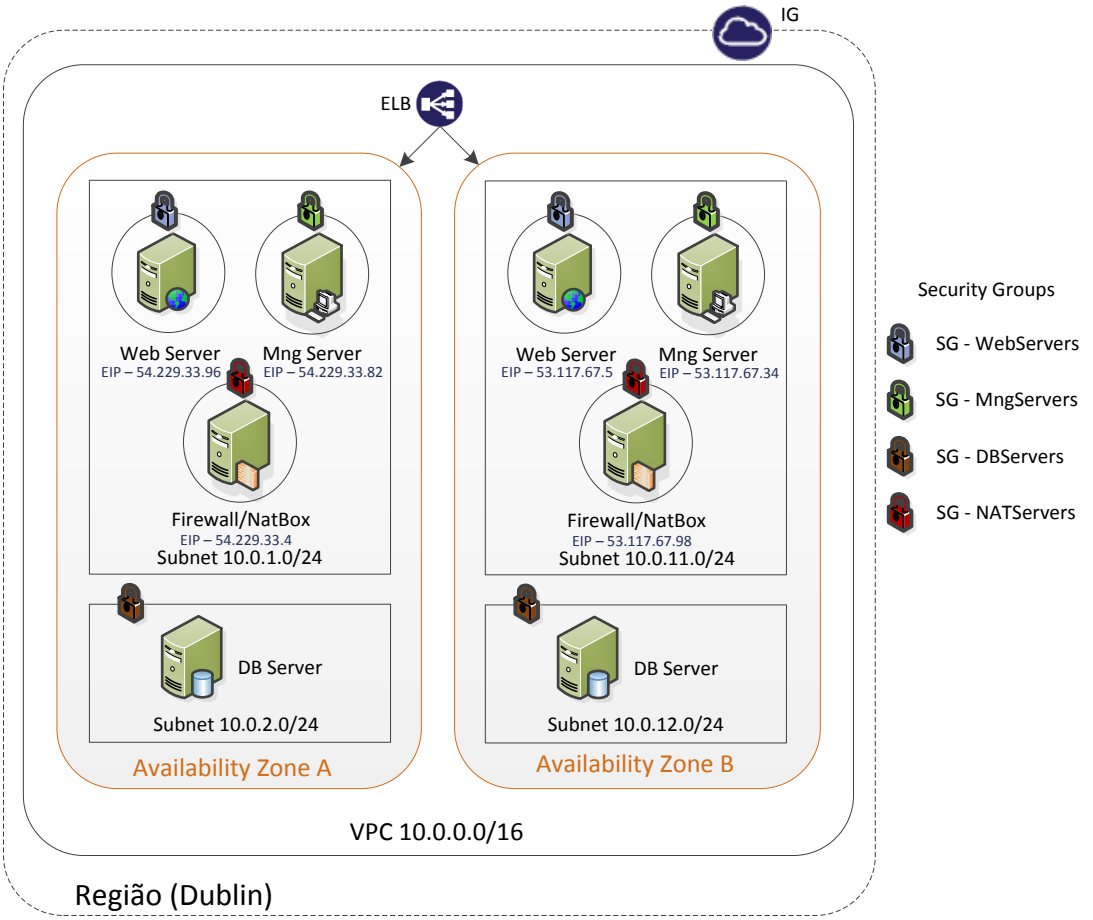

Figura 25 - Diagrama do segundo cenário proposto

Qualquer *Web Server* deverá apenas possuir abertas as portas 80 e 443, representando estas o serviço que efetua, devendo o acesso exterior, por questões de gestão, ser efetuado através de outra máquina. Já os servidores de dados, encontram-se isolados do exterior propositadamente, devido a questões de segurança da arquitetura, a gestão neste caso também terá de ser efetuada com recurso a outra máquina.

Assim, foi necessária a adição à infraestrutura, dum servidor de gestão para se aceder por Secure Shell (**SSH**), ou Remote Desktop Protocol (**RDP**) a qualquer máquina, quando se pretenda efetuar operações de gestão e manutenção das mesmas. O acesso a este servidor, é possível apenas para tráfego proveniente da organização, ou seja do bloco de endereços públicos que se adquiriu.

A restrição das comunicações é realizada através da implementação de Grupos de Segurança para cada tipo de servidores ou de segmento de rede. Apesar de não se conseguir implementar ACL muito específicas ou complexas através deste sistema, é de realçar que possui algumas características úteis, como por exemplo a vantagem de se poder indicar como origem ou destino de tráfego o identificador do Grupo de Segurança, em que determinada máquina se encontra inserida em oposição ao endereço IP. A representação desta lógica é ilustrada na Tabela 8.

Tabela 8 – Grupos de segurança definidos e regras associadas

Grupo de Segurança	Entrada			Saída		
	Protocolo	**Porta**	**Origem**	**Protocolo**	**Porta**	**Destino**
MngServers	TCP	22(SSH)	85.245.149.158/31	TCP	22(SSH)	10.0.0.0/16
	TCP	3389(RDP)	85.245.149.158/31	TCP	3389(RDP)	10.0.0.0/16
WebServers	TCP	80(HTTP)	0.0.0.0/0	TCP	80(HTTP)	0.0.0.0/0
	TCP	443(HTTPS)	0.0.0.0/0	TCP	443(HTTPS)	0.0.0.0/0
	TCP	22(SSH)	SG-MngServers	TCP	1433(MS SQL)	SG-DBServers
	TCP	3389(RDP)	SG-MngServers	TCP	1433(MS SQL)	SG-DBServers
DBServers	TCP	1433(MS SQL)	SG-WebServers			
	TCP	22(SSH)	SG-MngServers			
	TCP	3389(RDP)	SG-MngServers			
NATServers	TCP	80(HTTP)	SG-DBServers	TCP	80(HTTP)	0.0.0.0/0
	TCP	443(HTTPS)	SG-DBServers	TCP	443(HTTPS)	0.0.0.0/0
	TCP	22(SSH)	SG-MngServers			
	TCP	3389(RDP)	SG-MngServers			

Apesar dos servidores de dados apenas receberem tráfego proveniente dos *Web Servers*, necessitam de comunicar com a Internet para efetuarem atualizações de *software*. Impôs-se então, a necessidade de implementar um servidor a realizar o papel de NatBox e de *firewall* organizacional. Foi atribuído um EIP a este servidor e instalado o *software* Netfilter que permitiu a realização destas funções. O NAT implementado foi por Masquerede (correspondendo a NAT *Overload* em [Cisco Systems, 2007]), e a especificação da *firewall* encontra-se presente no Apêndice 9.1.1.

A arquitetura presente neste cenário, herda todas as vantagens da evolução da redundância do primeiro cenário e adquire importantes aspetos relativos à segurança da

infraestrutura. No entanto, ainda existe um certo grau de dependência do CSP no que à segurança diz respeito. O balanceamento de carga permanece inalterável, apesar de ainda não ser o ideal.

5.2.3 **Terceiro cenário**

No último cenário proposto, retirou-se proveito de todas as funcionalidades do serviço VPC, eliminando qualquer dependência de administração e configuração da infraestrutura por parte do CSP em termos de segurança. É modelada uma infraestrutura de Nuvem privada, com todas as máquinas inseridas numa rede organizacional privada como demonstra o esquema da Figura 26.

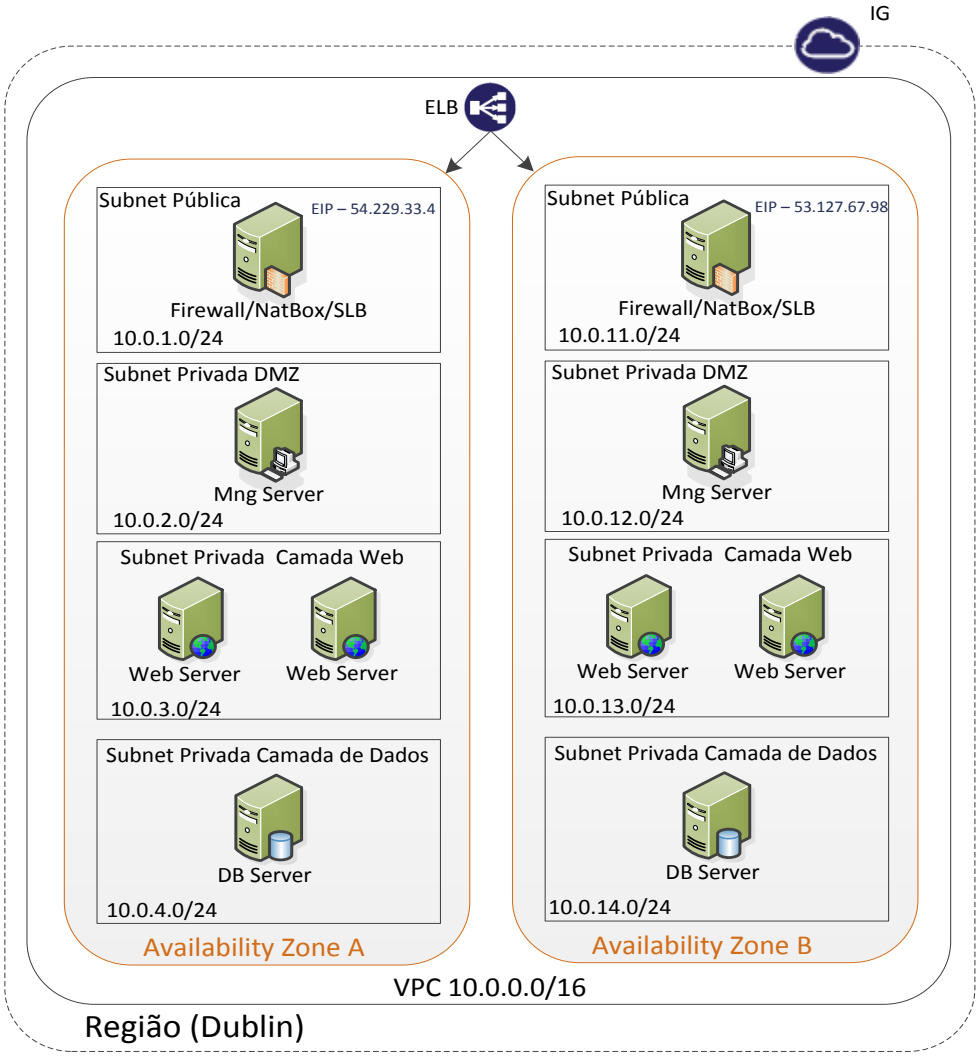

Figura 26 - Diagrama do terceiro cenário proposto

Neste modelo, a única máquina exposta para o exterior é o servidor que realiza o papel de *firewall*, NatBox, Server Load Balancing (**SLB**) e *gateway* da rede organizacional privada (posteriormente, será referido apenas como NatBox). Configuraram-se as tabelas de encaminhamento para que o *gateway* de qualquer *subnet* fosse a NatBox, assim como uma rota entre o *router/gateway* fornecido pela AWS e NatBox.

Implementou-se através do *software* Netfilter uma *stateful firewall*. Esta cria um poderoso sistema, que se recorda das ligações já efetuadas, evitando ataques do tipo *stealth scans*, que trazem "flags" especiais para técnicas de *port scanning*. Com a adição do módulo fuzzy, minimizaram-se ataques do tipo DOS. Foram realizados também os mapeamentos de NAT, necessários para a comunicação das máquinas presentes na rede privada com o exterior, estas configurações encontram-se presentes no Apêndice 9.1.2.

Foi também através de NAT que se realizou um segundo nível de balanceamento de carga entre os dois *Web Servers* de cada *Availability Zone*. Assim, todo o tráfego exterior ou interior à rede organizacional privada terá de passar neste servidor. Para melhor se perceber os fluxos do tráfego e os respetivos sentidos, origens e destinos dos mesmos que serão intercetados pela NatBox, esquematizou-se na Figura 27, os vários cenários possíveis numa *firewall* tradicional, servindo a Tabela 9 como complemento à compreensão dos mesmos. De referir que uma das diferenças de um sistema de *firewall* convencional, e o implementado na Nuvem, é a NatBox possuir apenas uma interface, o que provoca uma mudança na abordagem de implementação de algumas regras. Em vez de se considerar as interfaces de entrada e saída para distinguir o tráfego, foi considerado o endereço de origem ou de destino.

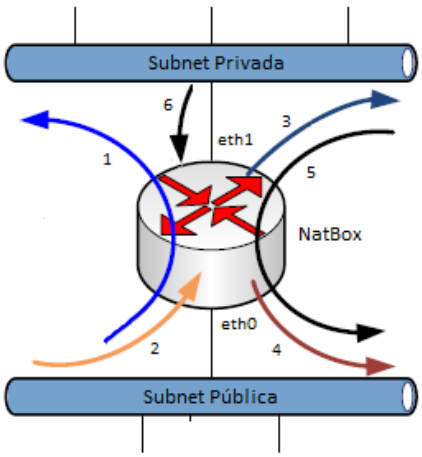

Figura 27 - Fluxos dos vários tipos de tráfego numa NatBox tradicional

Tabela 9 - Especificação dos fluxos de tráfego na NatBox

Origem	Destino	Entrada	Saída	P. Interseção	Número
Subnet Pública	Subnet Privada	Eth0	Eth1	Forward	1
Subnet Pública	NatBox	Eth0	--	Input	2
NatBox	Subnet Privada	--	Eth1	Output	3
NatBox	Subnet Pública	--	Eth0	Output	4
Subnet Privada	Subnet Pública	Eth1	Eth0	Forward	5
Subnet Privada	NatBox	Eth1	--	Input	6

A implementação de Grupos de Segurança deixou de ser necessária, já que existe total controlo do tráfego na NatBox. Apesar disto, a AWS requer a associação de um Grupo de Segurança a cada *subnet*, pelo que se implementou um *SG-Default* sem qualquer restrição de tráfego em cada *subnet*. Como não representa qualquer impacto positivo/negativo para a infraestrutura, não é representado na Figura 26.

Esta última arquitetura proposta, apresenta uma elevada segurança, além de um completo controlo sobre a administração e gestão desta. Foi adicionado um segundo nível de balanceamento de carga, herdando a redundância dos cenários anteriores uma vez que já era consideravelmente razoável.

Apesar das vantagens adquiridas, não se pode deixar de ponderar o *overhead* introduzido, quer pela introdução de um ponto de interceção de tráfego adicional quer pela realização de NAT. Estas questões serão abordadas no capítulo de testes à plataforma realizada.

5.2.4 **Implementação de *Auto-Scaling***

O *Auto-Scaling* permite dimensionar automaticamente a capacidade de uma infraestrutura na AWS, podendo este dimensionamento ser realizado com o aumento dos recursos computacionais ou com a sua diminuição, de acordo com as condições que se definirem.

Esta funcionalidade representa a elasticidade de que se pode usufrui em qualquer Nuvem, aumentando o número de instâncias (servidores) em serviço, quando existe um pico na sua utilização. Existe uma relação simbiótica entre o *Auto-Scaling* e o

CloudWatch. É através deste último, que se definiram alarmes na monitorização que permitiram o despoletar de cada grau (para cima ou para baixo) de elasticidade. Entre as diferentes métricas que se podem monitorizar em tempo real, encontram-se entre outros:

- Utilização de CPU;
- Utilização de memória;
- Tráfego de entrada ou de saída da rede;
- Utilização do disco;
- Número de pedidos recebidos por segundo.

Numa segunda fase de implementação da infraestrutura na Nuvem, foi adicionada esta funcionalidade aos cenários descritos anteriormente. Como qualquer configuração mais avançada na AWS, a simples interface Web que disponibiliza não permite a sua realização. Foi através da ferramenta AWS Comand Line Interface (**CLI**) que se implementou a configuração descrita na Tabela 13 do Apêndice 9.2.

Em suma, escolheu-se como métrica a utilização do CPU em percentagem (visto ser um bom indicador da capacidade de processar pedidos pelos servidores), e definiu-se os limiares de 80 e 20 por cento de utilização, para o respetivo aumento ou decréscimo da infraestrutura.

5.3 Considerações

Apesar de inicialmente ter sido considerada a hipótese da utilização de *hardware* de bilhética, o mesmo não se concretizou, dificultando a implementação do Subsistema de Segurança (SSS) já que o mesmo comtempla o uso de tecnologias e *hardware*, aos quais não foi viável ter acesso. Isto, prendeu-se com o facto de estarem a decorrer em paralelo, trabalhos no mesmo projeto onde se estuda a interação de *smartcards* com os terminais e a onde a alocação dos recursos/*hardware* fazia mais sentido.

Idealmente num SSS, os terminais (POSTS) efetuam troca de dados periódicos com um sistema de *backoffice* central (HOPS) e requerem a inclusão de um ISAM único entre todos os POSTS. A inexistência de um terminal de bilhética, para o efeito de testes da plataforma/serviço, levou a que a função desempenhada por um POST seja realizada

por uma máquina (computador) normal, não sendo possível a integração de um ISAM nesta.

Também do lado do *backoffice*, qualquer HOPS terá que possuir um dispositivo de segurança (HSAM), que na sua forma mais simples pode ser equiparado a um único ISAM. A impossibilidade de contratar a AWS para a implementação de um HSAM na sua infraestrutura física, considerando que se trata dum projeto de baixa dimensão, também contribuiu para a dificuldade da implementação integral do SSS.

Apresentam-se assim as plataformas descritas nas secções anteriores, implementadas de uma forma robusta, diversificada e segura, de acordo com os recursos disponíveis.

6. Resultados

Neste capítulo, são apresentados os resultados experimentais obtidos dos diversos testes realizados. Estes, não sendo de natureza exaustiva na descrição do desempenho da infraestrutura realizada (visto existirem múltiplas variáveis configuráveis no CSP, como Input/Output Operations Per Second (**IOPS**), variadíssimos tipos de instâncias e especificações, hardware dedicado ou largura de banda dedicada por exemplo, tudo dependendo do orçamento disponível), pretendem sobretudo responder de uma forma geral à questão que se levantou no início do projeto, sobre a possibilidade de migração de sistemas de bilhética para a Nuvem.

6.1 Plataforma de Testes

A plataforma de testes consistiu nas infraestruturas implementadas na Nuvem da AWS descritas no quinto capítulo, onde se publicou nos *Web Server*s o *Web Service* realizado no quarto capítulo e onde foi criado o modelo de dados nos servidores de base de dados.

A simulação de um terminal é realizada numa máquina pessoal, através do *software* SoapUI. Este permite a invocação dos métodos presentes no *Web Service*, através da análise do ficheiro WSDL do mesmo.

Através da utilização deste *software*, conseguiu-se realizar a abstração de um cliente SOAP que consumisse o *Web Service*, não tendo sido necessário desenvolvimento de um cliente para o efeito. É também através do SoapUI, que são retirados os valores dos testes realizados. Os valores obtidos indicam o RTT de cada método, sendo através destes que se afere o desempenho do sistema.

Os testes foram realizados ajustando o número de pedidos em concorrência, executados por segundo, aos *Web Services* dos vários cenários desenvolvidos. Estes são realizados em cada tipo de rede possível onde os terminais possam estar inseridos e em função do tipo de instância dos servidores.

6.2 Resultados

Para não tornar a avaliação demasiado extensa, foi considerado para análise apenas o método de validação, visto estar associado à operação com maior grau de requisitos temporais. Os resultados para os restantes métodos, encontram-se ilustrados no Apêndice 9.5.

Os resultados foram analisados em duas fases, a primeira onde se realizam testes aos cenários desenvolvidos no quinto capítulo sem a funcionalidade de *Auto-Scaling*, e numa segunda fase com a implementação desta funcionalidade.

Cada valor presente na Tabela 10 e na Tabela 11, corresponde à média de cinco testes efetuados para cada campo, representando o RTT por pedido efetuado.

6.2.1 **Primeira fase**

Na primeira fase efetuaram-se testes a partir do mesmo local, obtendo-se os valores representados na Tabela 10.

Tabela 10 – Resultados em milissegundos dos diferentes cenários sem *Auto-Scaling*, consoante o tipo de instância, rede e carga de pedidos efetuados por segundo ao método de Validação

Dimensão	**Instância m1.Small**				**Instância m1.Large**			
Rede	**Ethernet (802.3)**	**WiFi (802.11G)**	**WCDMA-HSUPA/3G**	**4G - LTE**	**Ethernet (802.3)**	**WiFi (802.11G)**	**WCDMA-HSUPA/3G**	**4G - LTE**
Primeiro Cenário								
1 pps	68,6	67,4	381,8	127,5	68,4	68	373,1	128,5
100 pps	96,6	95,5	428,8	216,7	93,6	91,7	401,4	212,2
200 pps	800,8	800,7	1548,3	1005,3	611,5	739,5	1415,1	1073,4
Segundo Cenário								
1 pps	72,8	71,8	390,48	121,5	70,5	69,8	385,4	117,4
100 pps	95,1	99,1	430,1	214,6	93,1	94,9	394,6	196,7
200 pps	811,6	788	1483,9	1111,5	696,3	704	1329	1033,9
Terceiro Cenário								
1 pps	70,8	69,2	376,6	129,5	70,2	67,3	368,2	116,3
100 pps	92,7	91,3	405,5	189,6	90,9	89,8	392,1	179,9
200 pps	150,7	154,4	688	373,5	137,1	147,3	605,8	368,3
300 pps	613,1	638,5	1210	1285,7	523,7	599,6	1040,8	1086,6

Ao analisar os resultados obtidos, verificou-se que o RTT médio apresentava um valor acima do esperado consoante se ia aumentando a carga do teste.

Num ambiente de produção, milhares de pedidos são realizados por diversos terminais em diferentes localizações (podendo considerar-se que existem no limite 50 terminais por local/estação no caso em que o transporte não é rodoviário, sendo que neste último apenas existe um terminal por veículo), colocando a carga de transações no local que recebe os pedidos e não onde os mesmos são efetuados.

Concluiu-se que para retratar um cenário o mais real quanto possível, ter-se-ia de efetuar testes de diferentes máquinas e em locais distintos, para estes parâmetros não afetarem o seu desempenho.

Verificou-se também que os dois primeiros cenários, apenas conseguiam servir no máximo cerca de 200 pedidos por segundo, enquanto que o terceiro, por apresentar mais um nível de balanceamento de carga e como tal mais capacidade de processamento, conseguia servir 300 pedidos. Isto indicou claramente que os servidores estavam com a sua capacidade de processar pedidos esgotada.

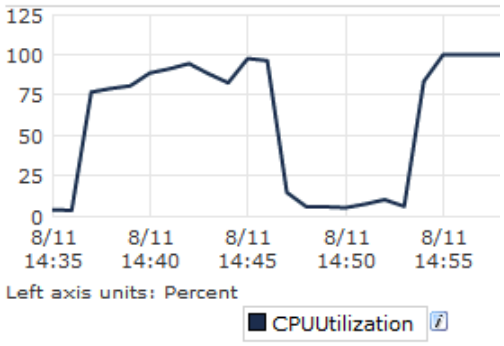

Figura 28 – Média de utilização do CPU de ambos os servidores presentes no 2º cenário

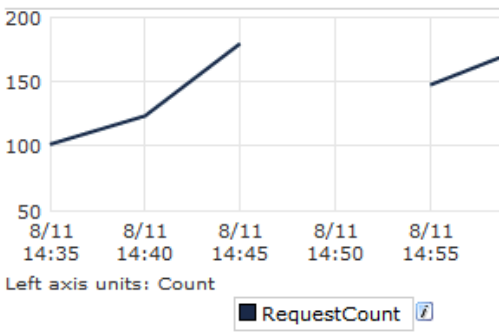

Figura 29 – Número de pedidos por segundo no balanceador de carga

Para confirmar, foi realizada a monitorização desta situação ao efetuar dois testes de carga a 200 pps com um intervalo de 10 minutos e confirmado como ilustra a Figura 28 e a Figura 29, que a utilização do CPU dos *Web Servers* do segundo cenário encontrava-se com valores próximos dos 100%, o que além de não permitir que processassem uma maior quantidade de pedidos, os que conseguiam processar apresentavam um RTT elevado.

6.2.2 **Segunda fase**

Numa segunda fase, executaram-se os mesmos testes em diferentes locais e consequentemente por diferentes máquinas, efetuando-se 100 pedidos por local.

Usufruiu-se do *Auto-Scaling* implementado e os resultados presentes na Tabela 11 corresponderam às expectativas.

Tabela 11 - Resultados em milissegundos dos diferentes cenários com *Auto-Scaling*, consoante o tipo de instância, rede e carga de pedidos efetuados por segundo ao método de Validação

Dimensão	**Instância m1.Small**				**Instância m1.Large**			
Rede	**Ethernet (802.3)**	**WiFi (802.11G)**	**WCDMA-HSUPA/3G**	**4G - LTE**	**Ethernet (802.3)**	**WiFi (802.11G)**	**WCDMA-HSUPA/3G**	**4G - LTE**
Primeiro Cenário								
1 pps	69,6	71,3	393,3	130,5	70,1	68,2	386,1	134,5
100 pps	95,3	96,1	448,5	211,8	94,6	97,2	437,7	205,2
200 pps	94,5	96,9	435,3	212,4	94	93,5	437,3	208,8
300 pps	92	92,6	441,7	207,5	97,2	96,6	435,6	201
800 pps	93,1	94	434,8	217	93,4	99,2	431,6	204,6
Segundo Cenário								
1 pps	68,4	70,2	400,3	129,6	70,4	67,8	385,3	131,4
100 pps	93,1	95,9	440,6	211,1	94,5	94,2	428,4	196,7
200 pps	91,6	93,1	438,5	206,2	95,2	96,4	436,9	208,6
300 pps	96,8	95,7	442	202,8	95,1	91,3	429,8	189,8
800 pps	95,2	99,1	435,7	213,3	94,7	92,2	427,5	202,1
Terceiro Cenário								
1 pps	69,1	68,4	394,3	120,3	67,2	68	382,5	120,9
100 pps	92,7	92	410,9	199,2	91	90,5	398,5	191,6
200 pps	89,5	93,7	415,6	190	88,4	88,4	402,5	188,5
300 pps	92,5	90,9	410	194,1	89,1	87.5	389,6	192,5
800 pps	90,1	88,4	407,2	189,2	92,7	89,2	394,1	187,7

Ou seja, independentemente da quantidade de pedidos a serem processados pela infraestrutura, esta, devido à elasticidade fornecida adaptou-se ao iniciar mais instâncias de *Web Servers* e como tal, o tempo de processar os 100 primeiros pedidos é exatamente igual ao tempo de processar 100 pedidos quando já se encontrava a receber 700.

Nos valores obtidos pensa-se ainda existir interferência das condições de testes (visto não se conseguir replicar o ambiente ideal ou seja, cada pedido ser invocado de um local diferente), porque se, para o teste de carga a 800 pps, os primeiros 100 pedidos apresentam a mesma latência que os últimos 100, então, num cenário de produção, a latência de cada pedido independentemente dos pedidos que a infraestrutura já esteja a processar, terá o valor representado no teste de carga a 1 pps.

Os resultados foram bastante positivos, mesmo considerando que num ambiente de produção as bases de dados terão maior dimensão, e as operações apresentarão um maior grau de complexidade, pensa-se que não ultrapassarão os 300 milissegundos, apresentados como requisito.

6.3 Considerações

Neste subcapítulo efetuou-se uma análise à infraestrutura implementada consoante os resultados obtidos, e inferiu-se sobre os mesmos na ótica dos operadores. Assim, nas próximas secções, estas considerações encontram-se separadas por tipos de operadores de transporte, cenário, tipos de instâncias, tipo de rede e por fim a aplicabilidade da melhor solução aos diferentes tipos de operadores.

6.3.1 **Tipos de operadores de transporte**

Antes de começar a analisar os resultados, convém definir de uma forma clara que os operadores podem inserir-se em várias categorias de transportes distintos, possuindo cada categoria diferentes características operacionais que determinam a localização dos terminais. Interessa então, considerar os diferentes tipos de transportes visto que os mesmos podem certamente condicionar a aquisição do modelo e conceito de “thin

device". A Tabela 12 resume as características de rede destes, em função da localização dos terminais de validação.

Tabela 12 – Características dos vários tipos de transporte

Tipo de Transporte	Posicionamento dos terminais de validação	Rede acessível
Metropolitano	Interior	Ethernet/Wifi
Rodoviário	Exterior	3G/4G
Ferroviário	Interior	Ethernet/Wifi
Aviação	Interior	Ethernet/Wifi
Fluvial	Interior	Ethernet/Wifi

Da análise à Tabela 12, constatou-se que se podia agrupar os diferentes tipos de transporte em dois grupos, para efeitos de estudo da aplicabilidade do modelo em função do tipo de rede disponível: um contendo transportes rodoviários e outro contendo todos os outros operadores.

6.3.2 **Comparação de cenários**

Avaliando os resultados obtidos, constatou-se que o desempenho dos primeiros 2 cenários foi muito semelhante, tendo a vantagem do segundo apresentar mais segurança. A análise do primeiro cenário apenas é equacionada para ilustrar o desenvolvimento progressivo da infraestrutura, já que a sua topologia é impraticável para qualquer empresa devido à segurança da mesma.

Comparando o segundo, relativamente ao terceiro cenário, constatou-se que este último apresentou melhores resultados devido ao nível adicional de balanceamento de carga. Neste, quando é ativado o *Auto-Scaling*, são iniciadas duas instâncias por *Availability Zone* enquanto que no segundo cenário é apenas lançada uma. O resultado é um maior poder de processamento por cada nível de elasticidade, ajustando rapidamente a necessidade de processamento ao número de pedidos efetuados. Isto demonstra-se na Figura 30 e Figura 31, respetivamente.

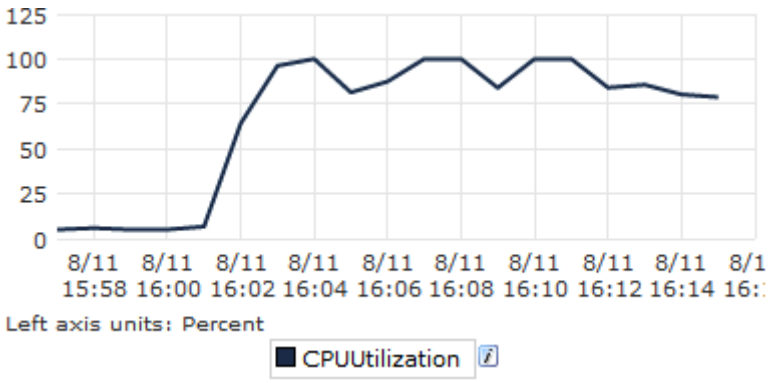

Figura 30 – Utilização média do CPU das instâncias relativas ao 2º cenário para 300 pps

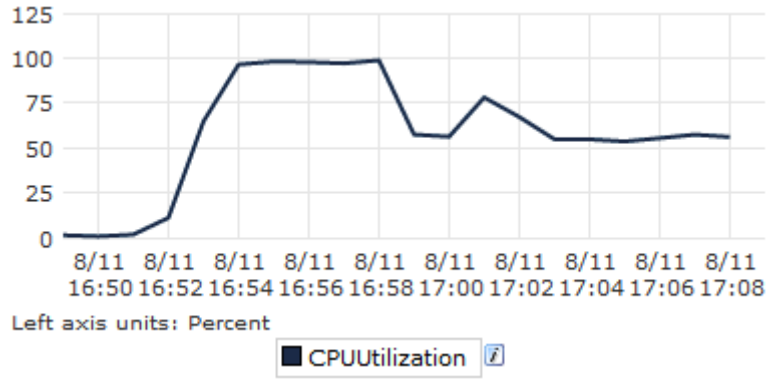

Figura 31 - Utilização média do CPU das instâncias relativas ao 3º cenário para 300 pps

Como o terceiro cenário, além de apresentar melhores resultados é também o que fornece um maior grau de segurança controlável pela entidade bilhética, por isso será a partir deste que se realizarão todas as próximas considerações.

6.3.3 **Tipos de instâncias**

Não se verificou uma melhoria considerável pelo aumento de *hardware* das instâncias dos *Web Servers*, devido ao *Auto-Scaling* realizado. A ténue melhoria de resultados deveu-se ao facto de que para o mesmo número de pedidos concorrentes que o serviço esteja a servir: (a) se o tipo de instância for *m1.Large*, são lançadas X instâncias; (b) se o tipo de instâncias for *m1.Small*, são lançadas $X * 2$. Ou seja, a capacidade computacional acaba por ser igual, apenas existe uma maior variação no caso em que as instâncias são do tipo *m1.Small*, assim, se existisse uma grande afluência de pedidos

que exigisse dois graus de elasticidade para as *m1.Small* significaria que se fossem instâncias *m1.Large* apenas seria necessário um grau de elasticidade.

A adoção do tipo de instância poderá depender do custo de cada solução e do valor que se pretenda investir. Se o custo por hora das instâncias com mais capacidade fosse o dobro, ou menos relativamente às de menor capacidade então compensaria sempre adotar as de maior capacidade.

A realidade é que o custo quadruplica em função do tamanho, não compensando na maioria dos casos o investimento. Acrescentando ainda o facto de as de menor capacidade conseguirem uma maior granularidade de processamento e como tal um maior ajuste de processamento efetivo, relativamente ao necessário. Assim considerar-se-á os resultados relativos às instâncias *m1.Small*.

6.3.4 **Tipo de rede**

O tipo de rede em questão define o tipo de operador (rodoviário/outro), como é referido anteriormente. Se para os operadores rodoviários, os resultados que fornecem informação relevante estão associados às redes 3G e 4G, já para os outros tipos de operadores, são os resultados relativos às redes Ethernet e WiFi que fornecem os dados relevantes. A análise destes em função do tipo de operador é realizada na próxima secção.

6.3.5 **Aplicabilidade da melhor solução aos diferentes tipos de operadores**

Esta análise foi efetuada com os requisitos estabelecidos pela Link, de que o método de validação num terminal deve apresentar valores inferiores a 300ms, e analisando os resultados referentes ao terceiro cenário para o tipo de instância *m1.Small*, pode-se concluir que, para operadores cujo o foco sejam transportes metropolitanos, fluviais, aviação ou ferroviários, a mudança de paradigma apresentada poderá ser concretizada sem prejuízo da efetividade/disponibilidade do serviço prestado.

Já para os operadores de transportes rodoviários, esta mudança precisará de ser ponderada em função do tipo de rede que o operador possui ou implementará. Se a rede for 3G, a disponibilidade do serviço será afetada, não cumprindo os requisitos apresentados. Se por contrário, o operador possuir ou esteja a pensar em adquirir uma infraestrutura que suporte 4G, então poderá usufruir da nova arquitetura proposta no projeto SmartCITIES Cloud Ticketing.

7. Conclusões

O trabalho foi dividido em duas fases:

1. Fase inicial, a qual decorreu de Setembro a Dezembro de 2012, onde foi realizado o levantamento do estado da arte relacionado com o projeto SmartCITIES, dos temas da Nuvem computacional e dos sistemas de bilhética, os quais serviram de base para a escrita dos meus capítulos 2 e 3. De igual forma contribuí para a documentação de suporte do projeto SmartCITIES. O conhecimento adquirido permitiu definir melhor o objetivo do meu trabalho e a estratégia para o desenvolvimento da plataforma de teste, apresentada no capítulo 5.
2. Numa segunda fase foi desenvolvida a plataforma e obtidos os resultados, a qual passou por três fases: (a) modelização do problema e respetivo desenvolvimento na linguagem C#, a qual originou o capítulo 4; (b) implementação das infraestruturas necessárias ao projeto na Nuvem da AWS, o qual deu origem ao capítulo 5; e (c) a obtenção de resultados, capítulo 6.

Embora a implementação tenha decorrido de forma fluída ao longo do projeto, em grande parte devido ao apoio prestado pelos especialistas da Link, a obtenção dos resultados revelou-se algo "*tricky*", devido à flutuação destes com o local onde se efetuavam os testes, das características da máquina a emular os vários terminais ou número de máquinas e efetuar esta emulação.

A implementação dos vários cenários e das condições de teste, foi revisitada várias vezes, culminando numa fase final com a iteração em paralelo destas duas fases, com o objetivo de obter resultados o mais realistas e credíveis quanto possível.

O trabalho desenvolvido e simulações realizadas, apontam no sentido da possibilidade efetiva de migração de sistemas de bilhética eletrónica dotados da arquitetura presente no projeto SmartCITIES Cloud Ticketing para a Nuvem, dependendo a sua aplicabilidade, da dimensão e características dos operadores que queiram usufruir deste serviço.

Do trabalho desenvolvido foram submetidos dois artigos para as conferências:

1. João E. Silva e João C. Ferreira, *Cloud Ticketing Comunication Testing Platform*. Submetido para a International Conference on Advances in Computing, Control and Networking (ACCN), a realizar de 07-08 Dezembro, 2013 no Dubai;
2. João E. Silva e João C. Ferreira, A*mazon Smartsales Ticketing System*. Submetido para a Conference on Electronics, Telecommunications and Computers (CETC 2013), a realizar de 5 a 6 Dezembro em Lisboa.

7.1 Trabalho Futuro

O desenvolvimento exponencial das tecnologias da informação, tem permitido às empresas de bilhética revolucionar a forma como armazenam os dados relativos aos títulos de transporte [Blythe, Improving public transport ticketing through smart cards, 2004]. Entre as vantagens de tal abordagem, encontra-se a interoperabilidade entre operadores de transporte, geridos por uma única entidade bilhética. Para atingir esta meta de interoperabilidade será seguramente necessária a centralização das infraestruturas de *backoffice* pertencentes a cada operador.

A arquitetura do projeto SmartCITIES potencia isto mesmo, ao permitir a diferentes operadores em ambientes geográficos distintos ou dentro da mesma área geográfica, usar uma infraestrutura bilhética partilhada.

A Nuvem surge então como o sistema ideal para esta centralização, permitindo a migração de infraestruturas de bilhética de operadores, que implementem uma arquitetura multicamada.

Esta opção de migração, poderá ser tomada pelos operadores que integrem uma especificação comum (ITSO/Calypso/VDV). Cada operador independentemente da sua localização geográfica, poderá possuir na Nuvem um serviço ou infraestrutura que sirva os seus interesses, dependendo do modelo que escolha.

Este trabalho concentrou-se na avaliação do desempenho referente a um modelo, onde diversos serviços comuns a todos os operadores são alojados na Nuvem numa única VPC, exclusivamente administrada por uma entidade bilhética.

Pelo conhecimento adquirido ao longo deste trabalho, pensa-se no entanto que, possam existir outras formas de implementação da arquitetura proposta pelo projeto SmartCITIES na Nuvem, que poderão fazer sentido investigar, entre as quais destaco:

1. Um modelo, onde cada operador possui o seu próprio serviço numa VPC exclusivamente sua, partilhando apenas o acesso a determinados dados, como é sugerido na Figura 32.

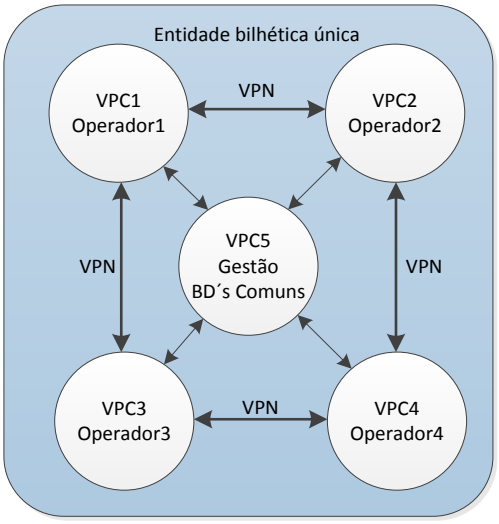

Figura 32 - Modelo de vários operadores na Nuvem com os serviços separados em VPC´s distintas

O acesso aos dados poderá realizar-se de duas formas, dependendo da confiança e acordos realizados entre os operadores. Os operadores poderão disponibilizar na sua infraestrutura, bases de dados apenas de leitura com a informação estritamente necessária à validação de passageiros, ou poderá ser implementada uma VPC de gestão, onde se localiza uma base de dados central com a informação estritamente necessária, sincronizada com as bases de dados de cada um dos operadores.

Neste modelo, cada operador possui a sua própria lógica de negócio isolada dos restantes (apesar de se encontrar isolada, assenta sobre acordos efetuados entre operadores), sendo a camada de dados a única partilhada. De referir que as bases de dados nunca seriam completamente partilhadas, mas cada operador partilhava numa base de dados "read only" o que deseja-se, efetuando-se a sincronização das bases de dados de cada operador, dentro das respetivas VPC´s periodicamente.

Abaixo ilustra-se um possível exemplo de um caso de uso, utilizando ambas as operações de venda e validação:

- **Um passageiro, com um título adquirido no operador B, pretende usufruir dos serviços de transporte pertencentes ao operador A:** No momento de aquisição de um título multioperador no operador B, é contactada a lógica de negócio do operador em questão, que através de pré-acordos já estabelecidos sabe o valor a imputar. Após a venda, é efetuada a sincronização com a base de dados partilhada para colocar a informação do título vendido. Quando o passageiro pretender usar o título no operador A, este operador acede à base de dados partilhada, onde encontra a informação relativa à validade do título.

2. Um modelo com o mesmo princípio que o anterior, ou seja, o de manter cada operador com o seu próprio serviço, partilhando apenas os dados, mas agora agrupando os operadores pertencentes à mesma zona geográfica, ou com o mesmo conjunto de interesses comerciais num submodelo, como se sugere na Figura 33.

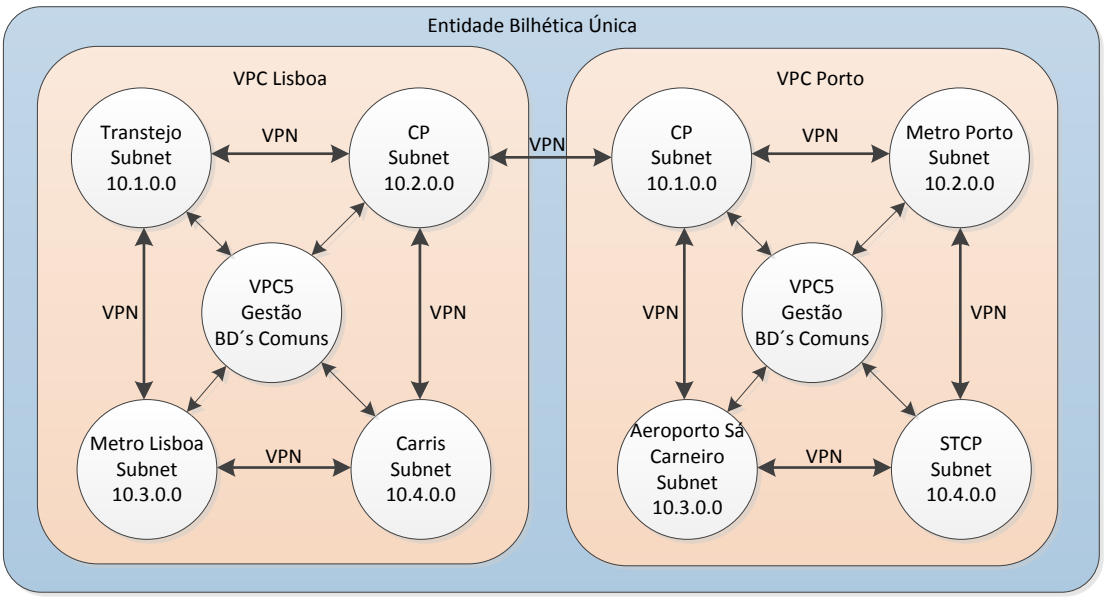

Figura 33 - Modelo assente numa Nuvem Comunitária

Enquanto o modelo anterior consistia numa Nuvem privada, onde a estrutura de cada operador se encontrava posicionada dentro de um ponto de controlo de acessos, gerido pelo proprietário, o modelo ilustrado na Figura 33 assenta no modelo de

Nuvem Comunitária onde ocorre a partilha entre diversas empresas de uma Nuvem, sendo esta suportada por uma comunidade específica com interesses comuns.

Este tipo de modelo poderá ser útil a empresas a operar em diversa áreas geográficas (a CP no exemplo referido), que por questões de lógica de negócio ou devido a acordos municipais, poderão beneficiar deste tipo de implementação.

No entanto, esta interoperabilidade relativa a estes dois modelos, levanta questões relacionadas com a privacidade e segurança dos dados de cada operador. Existe a possibilidade de operadores acederem a dados restritos de operadores concorrentes. Poderá surgir a necessidade, de serem impostos requisitos de segurança e políticas pelos operadores, deixando a administração destes a carga da entidade bilhética. A validação destes modelos e segurança associada, poderá ser uma interessante tarefa a perseguir através do desenvolvimento dos respetivos cenários e implementação de testes para obtenção de resultados.

Relativamente ao serviço SmartSales, pretendeu funcionar apenas como um protótipo de testes, faltando por exemplo entre outros, associar uma data ao método *SessionOpen* para que a sessão termine automaticamente após 24 horas.

Não foi considerada uma operação de venda, e como tal, o método *TransactionDo* não considera uma venda. Poderá desenvolver-se este método, para que além de funcionar como recarregamento, possa também funcionar como uma venda, criando contratos na base de dados.

Enfim, pensa-se que foram criadas bases sólidas, para que trabalhos futuros apenas focados na implementação do Serviço de Bilhética se possam apoiar no projeto desenvolvido. Entre os que poderão beneficiar da estrutura realizada, encontra-se o desenvolvimento da segurança (através da implementação das normas WS-*) associada ao *Web Service*, em detrimento da segurança da plataforma na AWS.

7.2 Considerações Finais

A área de computação em Nuvem, especialmente Iaas, tem sofrido uma contínua e crescente evolução ao longo dos últimos anos, com uma tendência crescente para expansão dos domínios aos quais esta pode ser aplicada. Esta evolução pode mesmo ser considerada exponencial, propriedade comum a muitas outras áreas dentro das TI. De facto, durante a realização deste trabalho foram constantemente sendo adicionados novos serviços, funcionalidades e melhorias à plataforma utilizada (AWS).

Este trabalho serviu para estudar o estado da arte do atual paradigma que envolve a computação em Nuvem, e de alguma forma, aferir a sua aplicabilidade aos modernos sistemas de bilhética. Dos resultados obtidos e do caminho percorrido, pensa-se ter-se contribuído com algumas importantes métricas para a parte comunicacional do projeto SmartCITIES, apresentando-se ainda algumas sugestões para uma futura investigação, com vista ao melhoramento do modelo, tendo em vista a migração para a Nuvem das infraestruturas centrais de uma entidade bilhética.

8. Referências Bibliográficas

Amazon Web Services. (2010, October). *AWS Migration Scenarios: Web Application Architecture.* Retrieved from http://d36cz9buwru1tt.cloudfront.net/CloudMigration-scenario-wep-app.pdf

Amazon Web Services. (2010, December). *AWS Overview Whitepaper.* Retrieved from http://awsmedia.s3.amazonaws.com/pdf/AWS_Security_Whitepaper.pdf

Amazon Web Services. (2011, March). *Security Whitepaper.* Retrieved from http://awsmedia.s3.amazonaws.com/pdf/AWS_Security_Whitepaper.pdf

Amazon Web Services. (2012, October). *Amazon Virtual Private Cloud Connectivity Options.* Retrieved from http://aws.amazon.com/pt/whitepapers/

Bagchi, M. (2003). *Use of Smartcard Data from Bus Systems for Travel Behaviour Analysis and Implications for Marketing.* Westminster: PhD Thesis.

Birman, K., Chockler, G., & Renesse, V. (2009). Toward a Cloud Computing Reserch Agenda. *SIGACT News, 40*(2), 68-80.

Blythe, T. (1999). The Smart Card Solution for Public Service Across Europe. *Proceedings of Smart Cards in Public Services, AIC Conference.* Brussels.

Blythe, T. (2004). Improving public transport ticketing through smart cards. *Proceedings of the Institution of Civil Engineers*, 47-54.

Caulfield, B., & O'Mahoney, M. (2007). An examination of the public transport information requirements of users. *IEEE Trans. on Intelligent Transportation Systems, 8*(1), 21-30.

Chantérac, G., & Graindorge, L. (2009). *Focus Paper on Privacy in Transport IFM Applications.* IFM Project.

Cisco Systems. (2007). *Interconnecting Cisco Networking Devices Part 1.* Cisco Press.

Cisco Systems. (2010). *Security Considerations White Paper for Cisco Smart Storage*. Retrieved from Cisco: http://www.cisco.com/en/US/docs/storage/nass/csbcdp/smart_storage/white_paper/Security_Considerations_OL-23025.pdf

CSA. (2009, December). *Security Guidance for Critical Areas of Focus in Cloud Computing*. Retrieved from Cloud Security Alliance: https://cloudsecurityalliance.org/csaguide.pdf

Deelman, E., Singh, G., Livny, M., Berriman, B., & Good, J. (2008). The Cost of Doing Science on the Cloud: the Montage Example. *Proccedings of the 2008 ACM/IEEE Conference on Supercomputing* (pp. 1-12). New Jersey: IEEE Press.

Dutcher, B. (2001). *The NAT Handbook: Implementing and Managing Network Address Translation.* United States: Wiley & Sons.

Ferre, R., Pomeroy, D., Wahlstrom, M., & Watts, D. (2006). *Virtualization of the IBM System X3950 Server.* IBM Redbooks.

Foster, I., Zhao, Y., Raicu, I., & Lu, S. (2008). Cloud Computing and Grid Computing 360-Degree Compared. *Grid Computing Environments Workshop.* Austin.

Garfinkel, T., & Rosenblum, M. (2003). A Virtual Machine Introspection Based Architecture for Intrusion Detection. *Proceedings of the 2003 Network and Distributed System Security Symposium.* San Diego.

Google. (2011). *Omaha Client-Server Protocol V3*. Retrieved from https://code.google.com/p/omaha/wiki/ServerProtocol

Hime, J., Ingber, L., Petraglia, A., & Rembold, M. (2012). *Stochastic Global Optimization and Its Applications with Fuzzy Adaptive Simulated Annealing: Intelligent Systems.* Springer.

Igudim, P. (2012). *Privacy Analysis of E-ticketing Systems.* Technical University of Dresden.

ISO 14443. (2008). *Standards family. Identication cards - Contactless integrated circuit cards - Proximity cards. Part 1, 2, 3 and 4.* International Organization for Standartization.

ISO 24014-1. (2007). *Interoperable Fare Management System, Part 1: Architecture.* International Organization for Standartization.

ISO 7816-4. (2005). *Identication cards - Integrated circuit cards, Part 4: Organization, security and commands for interchange.* International Organization for Standartization.

ITSO Technical Specification 1000-0. (2010). *Interoperable Public Transport Ticketing using contactless smart customer media, Part 0: Concepts and Contents.*

ITSO Technical Specification 1000-10. (2010). *Interoperable Public Transport Ticketing using contactless smart customer media, Part 10: Customer-Media-Definitions.*

ITSO Technical Specification 1000-3. (2010). *Interoperable Public Transport Ticketing using contactless smart customer media, Part 3: Terminals.*

ITSO Technical Specification 1000-4. (2010). *Interoperable Public Transport Ticketing using contactless smart customer media, Part 4: HOPS.*

ITSO Technical Specification 1000-5. (2010). *Interoperable Public Transport Ticketing using contactless smart customer media, Part 5: Customer Media Data Record Definitions.*

ITSO Technical Specification 1000-6. (2010). *Interoperable Public Transport Ticketing using contactless smart customer media, Part 6: Message Data.*

ITSO Technical Specification 1000-7. (2010). *Interoperable Public Transport Ticketing using contactless smart customer media, Part 7: ITSO Security Subsystem.*

ITSO Technical Specification 1000-8. (2010). *Interoperable Public Transport Ticketing using contactless smart customer media, Part 8: ISAM Detailed Operation.*

ITSO Technical Specification 1000-9. (2010). *Interoperable Public Transport Ticketing using contactless smart customer media, Part 9: Communications.*

Kretzschmar, M., & Hanigk, S. (2010). Security Management Interoperability Challenges for Collaborative Clouds. *Systems and Virtualization Management (SVM), 2010 4th International DMTF Academic Alliance Workshop*, (pp. 43-49). Niagara Falls.

Landwehr, C. (2001). Computer Security. *International Journal of Information Security, 1*(1), 3-13.

Marinos, A., & Briscoe, G. (2009). Community Cloud Computing. In *Cloud Computing* (Vol. Volume 5931 of Lecture Notes in Computer Science, pp. 472-484). Springer.

Mather, T., Kumaraswamy, S., & Latif, S. (2009). *Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance (Theory in Practice).* United States: O´Reilly.

Mcafee Corporation. (2010). *Database Security in Virtualization and Cloud Computing Environment: The Three Key Technology Challenges in Protecting Sensitive Data in Modern IT Architectures.*

McMilan, B. (2011). Cloud Computing: What it can do for your business. *Entrepreneurial Practise Review, 2*(1), 67-71.

Mell, P., & Grance, T. (2009). *The Nist Definition of Cloud Computing.* National Institute of Standarts and Technology.

Minnear, R. (2011). *Latency: The Achilles Heel of Cloud Computing*. Retrieved from cloud Computing Journal: http://cloudcomputing.sys-con.com/node/1745523

Napper, J., & Bientinesi, P. (2009). Can Cloud Computing Reach the Top 500? *UCHPC-Maw`09: Proceedings of the Combined Workshops on Unconventional High Performance Computing Workshops Plus Memory Access Workshop*, (pp. 17-20). New York.

Newman, M., Wiberg, C., & BrasWell, B. (2005). *Server Consolidation with Vmware ESX Server.* IBM Redbooks.

Rehrl, K., Bruntsch, S., & Mentz, J. (2007). Assisting multimodal travelers: Design and prototypical implementation of a personal travel companion. *IEEE Trans. on Intelligent Transportation Systems, 8*(1), 31-42.

Rittinghouse, J., & Ransome, J. (2009). *Cloud Computing: Implementation, Management, and Security.* United States: CRC Press.

Tavis, M., & Fitzsimons, P. (2012). *Web Application Hosting in the AWS Cloud.* Amazon Team.

Vaquero, M., Rodero-Merino, L., Caceres, J., & Lindner, M. (2009). A Break in the Clouds: Towards a Cloud Definition. *ACM SIGCOMM Computer Communication Review, 39*(1), 50-55.

Velte, T., Velte, A., & Elsenpeter, R. (2009). *Cloud Computing, A Practical Approach.* United States: Mcgraw-Hill.

Verity, G. (2010). *Interoperable Management Project, Deliverable 7.3, version 6.*

Zanetti, D., Sachs, P., & Capkun, S. (2011). The Practicality of UHF RFID Fingerprinting: How Real is the RFID Tracking Problem? In S. Fischer-Hbner, & N. Hopper, *Privacy Enhancing Technologies* (Vol. 6794 of Lecture in Computer Science, pp. 97-116). Berlin: Springer.

9. Apêndices

Sob a forma de apêndices apresentam-se entre outros as configurações das *firewalls*, as configurações de *Auto-Scaling*, resultados completos dos testes efetuados bem como exemplos dos pedidos e respostas dos vários métodos desenvolvidos.

9.1 Configuração das *Firewalls*

As *firewalls* foram realizadas com recurso à linguagem de Shell Script Linux, criando *scripts* para a sua implementação. Apresenta-se em primeiro lugar a configuração comum a ambas, seguido das configurações individuais das *firewalls* presentes no segundo e terceiro cenário do quinto capítulo.

```
#!/bin/bash

###Load of module which allows stateful packet inspection
modprobe ip_conntrack

### other network protection
# enable syn cookies (prevent against the common 'syn flood attack')
# ignore all ICMP ECHO and TIMESTAMP requests sent to it via broadcast/multicast
# log packets with impossible addresses to kernel log
# disable logging of bogus responses to broadcast frames
# don't accept packets with SRR option
# enable internal network machines to communicate with the outside world (because of NAT)

echo 1 > /proc/sys/net/ipv4/tcp_syncookies
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
echo 1 > /proc/sys/net/ipv4/conf/all/log_martians
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route
echo 1 > /proc/sys/net/ipv4/ip_forward

### Delete all rules in the chains
Iptables –t filter –F
Iptables –t nat –F
Iptables –t mangle –F

###Delete user created chains
Iptables –t filter –X
Iptables –t nat –X
Iptables –t mangle –X
```

```
### Reset byte counter
Iptables –t filter –Z
Iptables –t nat –Z
Iptables –t mangle –Z

### Default policy
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

### INPUT logging chain
iptables -N LOGGINGINPUT
iptables -A LOGGING –p tcp -m limit --limit 2/min -j LOG --log-prefix "log_tcp_IN"
iptables -A LOGGING –p udp -m limit --limit 2/min -j LOG --log-prefix "log_udp_IN"
iptables -A LOGGING –p icmp -m limit --limit 2/min -j LOG --log-prefix "log_icmp_IN"
iptables -A LOGGING -j REJECT --reject-with icmp-port-unreachable

### OUTPUT logging chain
iptables -N LOGGINGINPUT
iptables -A LOGGING –p tcp -m limit --limit 2/min -j LOG --log-prefix "log_tcp_OUT"
iptables -A LOGGING –p udp -m limit --limit 2/min -j LOG --log-prefix "log_udp_OUT"
iptables -A LOGGING –p icmp -m limit --limit 2/min -j LOG --log-prefix "log_icmp_OUT"
iptables -A LOGGING -j REJECT --reject-with icmp-port-unreachable

### FORWARD logging chain
iptables -N LOGGINGINPUT
iptables -A LOGGING –p tcp -m limit --limit 2/min -j LOG --log-prefix "log_tcp_FW"
iptables -A LOGGING –p udp -m limit --limit 2/min -j LOG --log-prefix "log_udp_FW"
iptables -A LOGGING –p icmp -m limit --limit 2/min -j LOG --log-prefix "log_icmp_FW"
iptables -A LOGGING -j REJECT --reject-with icmp-port-unreachable

### Prevention against IP spoofing
# Logs the “Bad Guys” if they are on the list
iptables -A INPUT -m recent --rcheck --seconds 60 -m limit --limit 10/second -j LOG --log-prefix
"BG"
# if the BG were last seen on the list at least 60s ago, it updates the list and drops the packets
iptables -A INPUT -m recent --update --seconds 60 -j DROP
# creates a list which records source address (mark as Bad Guys) and drops the spoofed packets
iptables -A INPUT -i eth0 -s 10.0.0.0/8 -m recent --set -j DROP
iptables -A INPUT -i eth0 –s 127.0.0.0/8 -m recent --set -j DROP

### drop silently well-known virus/port scanning attempts
iptables -A INPUT  -i eth0 -m multiport -p tcp --dports 53,113,135,137,139,445 -j DROP
iptables -A INPUT  -i eth0 -m multiport -p udp --dports 53,113,135,137,139,445 -j DROP
iptables -A INPUT  -i eth0 -p udp --dport 1026 -j DROP
iptables -A INPUT  -i eth0 -p tcp --dport 1026 -j DROP

### prevent DOS attacks and promote QoS with Fuzzy module
iptables -A INPUT  -m conntrack --ctstate NEW -m fuzzy --lower-limit 200 --upper-limit 1000 -j
ACCEPT

### accept everything from loopback (some internal applications may need to communicate
###trough the TCP/IP stack
iptables -A INPUT  -i lo -j ACCEPT
```

9.1.1 **Configuração da *firewall* do segundo cenário**

A *firewall* do segundo cenário tem apenas a função de realizar NAT do tráfego proveniente da *subnet* privada onde se encontram os servidores de base de dados para a Internet. Em relação ao tráfego no sentido oposto é apenas necessário considerar o tráfego relacionado com algum pedido efetuado, já que não é suposto serem iniciadas ligações de fora para dentro.

```
### Default policy
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

### Overloading NAT for the internal Database subnets
iptables -t  nat -A POSTROUTING -s 10.0.2.0/24  -j MASQUERADE
iptables -t  nat -A POSTROUTING -s 10.0.12.0/24  -j MASQUERADE

###Only connections to software update are allowed to the outside/ssh and rdp responses
#creation of Outbound chain
iptables -N OUTBOUND
iptables -A OUTBOUND -s 10.0.2.0/24 -m multiport -p tcp --dport 80, 443 -m conntrack --ctstate
NEW,ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTBOUND -s 10.0.2.0/24 -m multiport -p tcp --dport 22, 3389 -m conntrack –ctstate
ESTABLISHED,RELATED -j ACCEPT

### Internet IN (only packets belonging to established and related connections are allowed)
# Creation of Inbound chain
iptables -N INBOUND
iptables -A INBOUND -m conntrack --ctstate INVALID -j DROP
iptables -A INBOUND -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
# allow rdp and ssh access only from the management server
iptables -A INBOUND -s 10.0.1.109 -p tcp --dport 22 -j ACCEPT
iptables -A INBOUND -s 10.0.1.109 -p tcp --dport 3389 -j ACCEPT
iptables -A INBOUND -s 10.0.11.109 -p tcp --dport 22 -j ACCEPT
iptables -A INBOUND -s 10.0.11.109 -p tcp --dport 3389 -j ACCEPT

###Main CHAINS###
#INPUT
iptables -A INPUT -j INBOUND
iptables -A INPUT -j LOGGINGINPUT
#OUTPUT
iptables -A OUTPUT -j OUTBOUND
iptables -A OUTPUT -j LOGGINGOUTPUT
#FORWARD
iptables -A FORWARD -s ! 10.0.2.0/24 -j INBOUND
iptables -A FORWARD -s 10.0.2.0/24 -j OUTBOUND
# Communication between Web Servers and DB Servers
iptables -A FORWARD -s 10.0.1.101 -p tcp --dport 1433 -j ACCEPT
iptables -A FORWARD -s 10.0.11.101 -p tcp --dport 1433 -j ACCEPT
iptables -A FORWARD -d 10.0.1.101 -p tcp --sport 1433 -j ACCEPT
iptables -A FORWARD -d 10.0.11.101 -p tcp --sport 1433 -j ACCEPT
iptables -A FORWARD -j LOGGGINGFORWARD
```

9.1.2 **Configuração da *firewall* do terceiro cenário**

A *firewall* deste cenário executa diversas funções, entre elas: balanceamento de carga entre os *Web Servers*, mapeamentos de NAT entre a rede interna e externa (por NAT Overload [Cisco, 2007] / NAT Masquerade[15]) e atua como *gateway* da rede interna. As configurações apresentadas restringem-se apenas à *firewall* presente na NatBox da *Availability Zone* A, devido à sua redundância com a da *Availability Zone* B.

```
### Default policy
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP

### Overloading NAT for the internal subnets
iptables -t  nat -A POSTROUTING -s 10.0.0.0/16   -j MASQUERADE

### Redirecting ssh/rdp connections made to NatBox (only one with a public IP) to Mng Server
iptables -t nat -A PREROUTING -m multiport -p tcp --dport 22, 3389 -s 85.245.149.158/31 -d
54.229.33.4 -j DNAT --to-destination 10.0.2.109

###Load Balancing
# Marking each 2 packets, the first with mark 1 and the second with mark 2
iptables -t mangle -A PREROUTING -s ! 10.0.0.0/16  -m multiport -p tcp --dport 80, 443 -m
conntrack --ctstate NEW -m statistic nth --every 2 –packet 0 –j CONNMARK --set-mark 1
iptables -t mangle -A PREROUTING -s ! 10.0.0.0/16  -m multiport -p tcp --dport 80, 443 -m
conntrack --ctstate NEW -m statistic nth --every 2 –packet 1 –j CONNMARK --set-mark 2
# Depending on the mark, send to a diferent Web Server
iptables -t nat -A PREROUTING -m connmark -mark 1 -j DNAT --to-destination 10.0.3.101
iptables -t nat -A PREROUTING -m connmark -mark 2 -j DNAT --to-destination 10.0.3.102

### Creation of InboundInput chain
iptables -N INBOUNDINPUT
iptables -A INBOUNDINPUT -m conntrack --ctstate INVALID -j DROP
iptables -A INBOUNDINPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
iptables -A INBOUNDINPUT -j LOGGINGINPUT

###Creation of InboundForward chain
iptables -N INBOUNDFORWARD
iptables -A INBOUNDFORWARD -m conntrack --ctstate INVALID -j DROP
iptables -A INBOUNDFORWARD -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
#New connections from outside are accepted (these will be redirected to the Web Servers)
iptables -A INBOUNDFORWARD -s ! 10.0.0.0/16  -m multiport -p tcp --dport 80, 443 -j ACCEPT
# New ssh/rdp connections to the Mng Server are accepted
iptables -A INBOUNDFORWARD -s 85.245.149.158/31-d 10.0.2.109  -p tcp --dport 22 -j ACCEPT
iptables -A INBOUNDFORWARD -s 85.245.149.158/31 -d 10.0.2.109 -p tcp --dport 3389 -j ACCEPT
# Give access of the DB subnet to the Web Servers subnet (no need to do the opposite, since the
# response will be considerated ESTABLISHED/RELATED
iptables -A INBOUNDFORWARD -s 10.0.3.0/24 -d 10.0.4.0/24 -p tcp --dport 1433 -j ACCEPT
iptables -A INBOUNDFORWARD -j LOGGINGFORWARD
```

[15] http://www.netfilter.org/

```
###Creation of OutboundForward chain
iptables -N OUTBOUNDFORWARD
# Connections to outside are allowed if they belong to existing connections (ex:WB responses)
iptables -A OUTBOUNDFORWARD ss 10.0.0.0/16 -m conntrack --ctstate ESTABLISHED,RELATED -j
ACCEPT
# New http connections to the outside are allowed for software updates
iptables -A OUTBOUNDFORWARD -s 10.0.0.0/16 -m multiport --dport 80, 443 -j ACCEPT
iptables -A OUTBOUNDFORWARD -j LOGGINGFORWARD

### Connections made to the NatBox
iptables -A INPUT -j INBOUNDINPUT
iptables -A INPUT -j LOGGINGINPUT

### Connections made from the NatBox
iptables -A OUTPUT -j LOGGINGOUTPUT
```

9.2 Configurações do *Auto-Scaling*

Tabela 13 – Ações realizadas para a implementação do *Auto-Scaling* e respetivas descrições e comandos

Ação	Descrição	Comando
Configuração de iniciação	• Nome único para a configuração; • AMI a usar como *template* para a iniciação de novas instâncias; • Tipo de instância a lançar; • Grupo de Segurança a usar para a instância.	as-create-launch-config myLC --image-id ami-97a2bee3 --instance-type m1.small --region eu-west-1 --group sg-8c2fcae3
Grupo elástico	Define as propriedades do grupo de servidores • Nome único para o grupo; • VPC, zonas e *subnets* onde serão iniciados os servidores; • Mínimo e máximo de servidores ativos (convém o mínimo nunca ser inferior a 1 para o serviço não ficar inacessível); • Balanceador de carga a utilizar; • Número de segundos que a AWS espera após um evento de elasticidade até despoletar outro evento – *Grace-Period*.	as-create-auto-scaling-group mySG --launch-configuration myLC --availability-zones eu-west-1c --min-size 1 --max-size 6 --load-balancers myELB --vpc-zone-identifier subnet-0e2f5f66 --health-check-type ELB --grace-period 300 --region eu-west-1
Políticas de "scale-up" e "scale-down"	Onde se define a ação a realizar para determinado grupo • Nome do grupo a associar a política e o seu respetivo nome; • Número de instâncias a ajustar; • Tipo de ajustamento; • Tempo de prevenção para prevenir que a AWS inicie instâncias de uma forma demasiado acelerada.	as-put-scaling-policy --auto-scaling-group mySG --name scale-up --adjustment 1 --type ChangeInCapacity --cooldown 300 --region eu-west-1 as-put-scaling-policy --auto-scaling-group mySG --name scale-dn "--adjustment=-1" --type ChangeInCapacity --cooldown 300 --region eu-west-1

Alarmes	Realizam a ligação dos eventos monitorizados pelo CloudWatch com as políticas • Nome para os alarmes; • Descrição da monitorização do alarme; • Métrica Monitorizada; • *Namespace* do alarme (depende da métrica: Se a métrica fossem pedidos por segundo o *namespace* seria AWS/ELB); • Tipo de estatística utilizada pela métrica (Média/Percentagem/etc); • Intervalo de tempo em que o alarme é ativado; • Limiar da estatística; • Operador de comparação; • Dimensão em que o alarme é válido (grupo de instancias, balanceador de carga ou apenas uma instancia em particular); • Número de períodos onde a métricas quer se avalia retornou um valor acima ou abaixo do limiar escolhido.	aws cloudwatch put-metric-alarm --alarm-name sample-scale-up --alarm-description "Scale up at 80% load" --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average --period 60 --threshold 80 --comparison-operator GreaterThanThreshold --dimensions name=AutoScalingGroupName,value=mySG --evaluation-periods 3 --unit Percent --region eu-west-1 --alarm-actions autoScalingGroupName/mySG:policyName/scale-up
		aws cloudwatch put-metric-alarm --alarm-name sample-scale-dn --alarm-description "Scale down at 20% load" --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average --period 60 --threshold 20 --comparison-operator LessThanThreshold --dimensions name=AutoScalingGroupName,value=mySG --evaluation-periods 3 --unit Percent --alarm-actions autoScalingGroupName/mySG:policyName/scale-dn

9.3 Estrutura do Serviço Desenvolvido

Nesta secção é descrito o aspeto estrutural do serviço proposto, como os elementos principais, atributos envolvidos e a sua correspondente descrição bem como os métodos implementados, parâmetros trocados no fluxo de informação e exemplos XML de pedidos e respostas correspondentes a cada método.

9.3.1 **Elementos principais**

<*Request*>

Este é o envelope do pedido. Os seus atributos contêm os dados globais que descrevem a máquina/terminal ou a instância cliente.

Atributo	**Descrição**
protocol	Versão do protocolo.
terminalId	Identificador do cliente/terminal.
requestId	Um GUI aleatório gerado para cada pedido. Permite identificar pedidos duplicados enviados múltiplas ocasiões pelo stack de rede ou por proxys.
sessionId	Um GUI aleatório gerado para cada sessão, identifica a sessão pertencente a cada cliente/terminal e permite que não existam sessões duplicadas.

<*Response*>

Este é o envelope da resposta.

Atributo	**Descrição**
protocol	Versão do protocolo.

<*os*>

Este elemento encontra-se contido no envelope do pedido e descreve o sistema operativo do cliente. O servidor na resposta não fornece este elemento.

Atributo	**Descrição**
platform	Sistema operativo do cliente/terminal.
version	Versão do sistema operativo.

<app>

Também contido no envelope do pedido, indica o identificador da aplicação e a linguagem suportada.

Atributo	**Descrição**
appId	Identificador da aplicação.
lang	Linguagem suportada. Tem que ser enviado no formato BCP 47.

9.3.2 **Método SessionOpen**

Tabela 14 – Estrutura e exemplos do método SessionOpen

<table>
<tr><td>Descrição:</td><td colspan="3">Gera uma sessão no modelo de dados e autentica o cliente/terminal.</td></tr>
<tr><td>Método http:</td><td colspan="3">POST</td></tr>
<tr><td>Tipo http:</td><td colspan="3">Https</td></tr>
<tr><td>Requere Autenticação:</td><td colspan="3">Sim</td></tr>
<tr><td rowspan="7">Formato do Request Body:</td><td>Campo</td><td>Atributo</td><td>Descrição</td></tr>
<tr><td>authentication</td><td>type</td><td>Método de autenticação:
• “password” (default).</td></tr>
<tr><td>authentication</td><td>username</td><td>Nome de utilizador. Aconselha-se o uso do número de série do terminal para uma autenticação anónima.</td></tr>
<tr><td>authentication</td><td>password</td><td>Password de autenticaçãoenviada em texto simples.</td></tr>
<tr><td>metadata (opcional)</td><td></td><td>Informação adicional sobre o cliente enviada, sobre a qual o serviço responde com dados customizados para cada cliente/terminal.</td></tr>
<tr><td>metadata/entry</td><td>key</td><td>Tipo de metados:
• “IMEI”: International mobile equipment identity;
• “IMSIS”: International mobile subscriber identity.</td></tr>
<tr><td>metadata/entry</td><td>value</td><td>O valor correspondente á chave inserida.</td></tr>
<tr><td>Exemplo:</td><td colspan="3"><pre>
<soap:Header/>
  <soap:Body>
    <tem:sessionOpen>
      <tem:request>
        <smar:protocol>1.0</smar:protocol>
        <smar:terminalId>60173703</smar:terminalId>
        <smar:requestId>C8F6EDF3-B623-4ee6-B2DA-1D08A0B4C665</smar:requestId>
        <smar:os>
          <smar1:platform>telium</smar1:platform>
          <smar1:version>2.0</smar1:version>
        </smar:os>
        <smar:app>
          <smar1:appId>pt.link.sc.smartpos</smar1:appId>
          <smar1:lang>pt</smar1:lang>
        </smar:app>
        <smar:authentication>
          <smar:type>password</smar:type>
          <smar:login>
            <smar:username>joao</smar:username>
            <smar:password>1qaz2wsX</smar:password>
</pre></td></tr>
</table>

<table>
<tr><td></td><td>
</smar:login>
</smar:authentication>
<!--Optional:-->
<smar:metadata>
<smar:entry>
<!--Zero or more repetitions:-->
<smar1:Entry>
<smar1:key>IMEI</smar1:key>
<smar1:value>35395803-121326-9</smar1:value>
<smar1:key>IMSIS</smar1:key>
<smar1:value>404685505601234</smar1:value>
</smar1:Entry>
</smar:entry>
</smar:metadata>
</tem:request>
</tem:sessionOpen>
</soap:Body>
</soap:Envelope>
</td></tr>
<tr><td>Formato do Response Body:</td><td>
<table>
<tr><th>Campo</th><th>Atributo</th><th>Descrição</th></tr>
<tr><td>sessionId</td><td></td><td>Identificador de sessão do cliente/terminal. Em caso de falha na autenticação, identifica o tipo de erro ocorrido.</td></tr>
<tr><td>Metadata (opcional)</td><td></td><td>Informação adicional sobre o cliente.</td></tr>
<tr><td>metadata/entry</td><td>key</td><td>Tipo de informação do cliente/terminal.</td></tr>
<tr><td>metadata/entry</td><td>value</td><td>Descrição da informação.</td></tr>
</table>
</td></tr>
<tr><td>Exemplo:</td><td>
<s:Header>
<a:Action
s:mustUnderstand="1">http://tempuri.org/IServiceSmartSales/sessionOpenResponse</a:Acti
on>
</s:Header>
<s:Body>
<sessionOpenResponse xmlns="http://tempuri.org/">
<sessionOpenResult
xmlns:b="http://schemas.datacontract.org/2004/07/SmartSales.Session.Data"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
<b:sessionId>FF762C67-6EF8-46D2-803F-50655419CBC3</b:sessionId>
<b:metadata>
<b:entry
xmlns:c="http://schemas.datacontract.org/2004/07/SmartSales.Generic.Data">
<c:Entry>
<c:key>AgentName</c:key>
<c:value>mr.agente1</c:value>
</c:Entry>
<c:Entry>
<c:key>AgentPhone</c:key>
<c:value>962523534</c:value>
</c:Entry>
<c:Entry>
<c:key>AgentLocation</c:key>
<c:value>Lisboa</c:value>
</c:Entry>
<c:Entry>
<c:key>PromotionalMessage</c:key>
<c:value>mensagem de promocao 1</c:value>
</c:Entry>
</b:entry>
</b:metadata>
</sessionOpenResult>
</sessionOpenResponse>
</s:Body>
</s:Envelope>
</td></tr>
<tr><td>Notas:</td><td>Este método de autenticação não cria uma nova sessão caso a mesma já exista. De referir que as passwords são armazenadas na base de dados como um número</td></tr>
</table>

	inteiro resultante do hash da password. Suporta a adição de outros tipos de autenticação, atualmente o username e password são enviados em claro, sendo realizado o hash da password, transformado em inteiro e comparado com a base de dados. A não colocação da password explicitamente na base de dados fornece uma camada adicional de segurança.

9.3.3 **Método SessionClose**

Tabela 15 - Estrutura e exemplos do método SessionClose

Descrição:	Termina uma sessão no sistema.		
Método http:	POST		
Tipo http:	Https		
Requere Autenticação:	Sim		
Formato do Request Body:	---		
Exemplo:	<soap:Header/> <soap:Body> <tem:sessionClose> <tem:request> <smar:protocol>1.0</smar:protocol> <smar:terminalId>60173703</smar:terminalId> <smar:requestId>C8F6EDF3-B623-4ee6-B2DA-1D08A0B4C665</smar:requestId> <smar:sessionId>FF762C67-6EF8-46D2-803F-50655419CBC3</smar:sessionId> <smar:os> <smar1:platform>telium</smar1:platform> <smar1:version>2.0</smar1:version> </smar:os> <smar:app> <smar1:appId>pt.link.sc.smartpos</smar1:appId> <smar1:lang>pt</smar1:lang> </smar:app> </tem:request> </tem:sessionClose> </soap:Body> </soap:Envelope>		
Formato do Response Body:	**Campo**	**Atributo**	**Descrição**
		status	O estado da ação da aplicação.
Exemplo:	<s:Header> <a:Action s:mustUnderstand="1">http://tempuri.org/IServiceSmartSales/sessionCloseResponse</a:Action> </s:Header> <s:Body> <sessionCloseResponse xmlns="http://tempuri.org/">		

	<sessionCloseResult xmlns:b="http://schemas.datacontract.org/2004/07/SmartSales.Session.Data" xmlns:i="http://www.w3.org/2001/XMLSchema-instance"> <b:status>OK</b:status> </sessionCloseResult> </sessionCloseResponse> </s:Body> </s:Envelope>
Notas:	Os requisitos são de que uma sessão seja aberta na primeira interação diária do cliente/terminal com o serviço SmartSales. Programaticamente o terminal deve chamar o método SessionClose ao final do dia, sendo esta hora, especifica do operador em questão.

9.3.4 **Método GetCatalog**

Tabela 16 - Estrutura e exemplos do método GetCatalog

Descrição:	Devolve os produtos no catálogo do sistema associados ao tipo de contrato do utilizador.		
Método http:	POST		
Tipo http:	Https		
Requere Autenticação:	Sim		
Formato do Request Body:	**Campo**	**Atributo**	**Descrição**
		contractid	Identificador do contrato do utilizador.
Exemplo:	<tem:getCatalog> <tem:request> <smar:protocol>1.0</smar:protocol> <smar:terminalId>60173703</smar:terminalId> <smar:requestId>C8F6EDF3-B623-4ee6-B2DA-1D08A0B4C665</smar:requestId> <smar:sessionId>ff762c67-6ef8-46d2-803f-50655419cbc3</smar:sessionId> <smar:os> <smar1:platform>telium</smar1:platform> <smar1:version>2.0</smar1:version> </smar:os> <smar:app> <smar1:appId>pt.link.sc.smartpos</smar1:appId> <smar1:lang>pt</smar1:lang> </smar:app> <smar:userContractId>6666</smar:userContractId> </tem:request> </tem:getCatalog>		
Formato do Response Body:	**Campo**	**Atributo**	**Descrição**
	ProductData	productId	Identificador do produto do catálogo.
		description	Descrição do produto: • "Mensal"; • "7Dias"; • "Diário".
Exemplo:	<getCatalogResponse xmlns="http://tempuri.org/"> <getCatalogResult xmlns:b="http://schemas.datacontract.org/2004/07/SmartSales.Product.Data" xmlns:i="http://www.w3.org/2001/XMLSchema-instance"> <b:CatalogListProducts>		

	<b:ProductData> <b:productId>4</b:productId> <b:description>7Dias</b:description> </b:ProductData> <b:ProductData> <b:productId>5</b:productId> <b:description>1Dia</b:description> </b:ProductData> </b:CatalogListProducts> </getCatalogResult> </getCatalogResponse>

9.3.5 **Método TransactionConfirm**

Tabela 17 - Estrutura e exemplos do método TransactionConfirm

Descrição:	Confirma se o cliente/terminal está autorizado a vender o tipo de produto requerido ou se está blacklisted e assim impossibilitado de vender o tipo de produto selecionado.		
Método http:	POST		
Tipo http:	Https		
Requere Autenticação:	Sim		
Formato do Request Body:	**Campo**	**Atributo**	**Descrição**
		productId	Identificador do produto escolhido pelo utilizador.
Exemplo:	<tem:transactionConfirm> <tem:request> <smar:protocol>1.0</smar:protocol> <smar:terminalId>60173703</smar:terminalId> <smar:requestId>6B6642A5-53B1-46FE-9ABF-C908A08F2088</smar:requestId> <smar:sessionId>ff762c67-6ef8-46d2-803f-50655419cbc3</smar:sessionId> <smar:os> <smar1:platform>telium</smar1:platform> <smar1:version>2.0</smar1:version> </smar:os> <smar:app> <smar1:appId>pt.link.sc.smartpos</smar1:appId> <smar1:lang>pt</smar1:lang> </smar:app> <smar:productId>5</smar:productId> </tem:request> </tem:transactionConfirm>		
Formato do Response Body:	**Campo**	**Atributo**	**Descrição**
		status	O estado da ação da aplicação: • OK; • NOK.
Exemplo:	<transactionConfirmResponse xmlns="http://tempuri.org/"> <transactionConfirmResult xmlns:b="http://schemas.datacontract.org/2004/07/SmartSales.Transaction.Data" xmlns:i="http://www.w3.org/2001/XMLSchema-instance"> <b:status>OK</b:status> </transactionConfirmResult> </transactionConfirmResponse>		

9.3.6 **Método TransactionDo**

Tabela 18 - Estrutura e exemplos do método TransactionDo

Descrição:	Insere na base de dados o produto escolhido e associa-o ao contrato do utilizador.		
Método http:	POST		
Tipo http:	Https		
Requere Autenticação:	Sim		
Formato do Request Body:	**Campo**	**Atributo**	**Descrição**
		productId	Identificador do produto escolhido pelo utilizador.
		UserContractId	Identificador do contrato do utilizador.
Exemplo:	<tem:transactionDo> <tem:request> <smar:protocol>1.0</smar:protocol> <smar:terminalId>60173703</smar:terminalId> <smar:requestId>C8F6EDF3-B623-4ee6-B2DA-1D08A0B4C665</smar:requestId> <smar:sessionId>ff762c67-6ef8-46d2-803f-50655419cbc3</smar:sessionId> <smar:os> <smar1:platform>telium</smar1:platform> <smar1:version>2.0</smar1:version> </smar:os> <smar:app> <smar1:appId>pt.link.sc.smartpos</smar1:appId> <smar1:lang>pt</smar1:lang> </smar:app> <smar:productId>5</smar:productId> <smar:userContractId>6666</smar:userContractId> </tem:request> </tem:transactionDo>		
Formato do Response Body:	**Campo**	**Atributo**	**Descrição**
		status	O estado da ação da aplicação: • OK; • NOK.
Exemplo:	<transactionDoResponse xmlns="http://tempuri.org/"> <transactionDoResult xmlns:b="http://schemas.datacontract.org/2004/07/SmartSales.Transaction.Data" xmlns:i="http://www.w3.org/2001/XMLSchema-instance"> <b:status>OK</b:status> </transactionDoResult> </transactionDoResponse>		

9.3.7 **Método TransactionValidation**

Tabela 19 - Estrutura e exemplos do método TransacionValidation

Descrição:	Valida se determinado utilizador possui acesso ao serviço disponibilizado pela entidade de transporte.
Método http:	POST
Tipo http:	Https
Requere Autenticação:	Sim

Formato do Request Body:	**Campo**	**Atributo**	**Descrição**
		UserContractId	Identificador do contrato do utilizador.
Exemplo:	<tem:transactionValidation> <tem:request> <smar:protocol>1.0</smar:protocol> <smar:terminalId>60173703</smar:terminalId> <smar:requestId>C8F6EDF3-B623-4ee6-B2DA-1D08A0B4C665</smar:requestId> <smar:sessionId>ff762c67-6ef8-46d2-803f-50655419cbc3</smar:sessionId> <smar:os> <smar1:platform>telium</smar1:platform> <smar1:version>2.0</smar1:version> </smar:os> <smar:app> <smar1:appId>pt.link.sc.smartpos</smar1:appId> <smar1:lang>pt</smar1:lang> </smar:app> <smar:userContractId>6666</smar:userContractId> </tem:request> </tem:transactionValidation>		

Formato do Response Body:	**Campo**	**Atributo**	**Descrição**
		status	O estado da ação da aplicação: • OK; • NOK.
Exemplo:	<transactionValidationResponse xmlns="http://tempuri.org/"> <transactionValidationResult xmlns:b="http://schemas.datacontract.org/2004/07/SmartSales.Transaction.Data" xmlns:i="http://www.w3.org/2001/XMLSchema-instance"> <b:status>OK</b:status> </transactionValidationResult> </transactionValidationResponse>		

9.4 Modelo de Dados

O diagrama Entidade-Associação referente ao modelo de dados realizado encontra-se ilustrado abaixo na Figura 34.

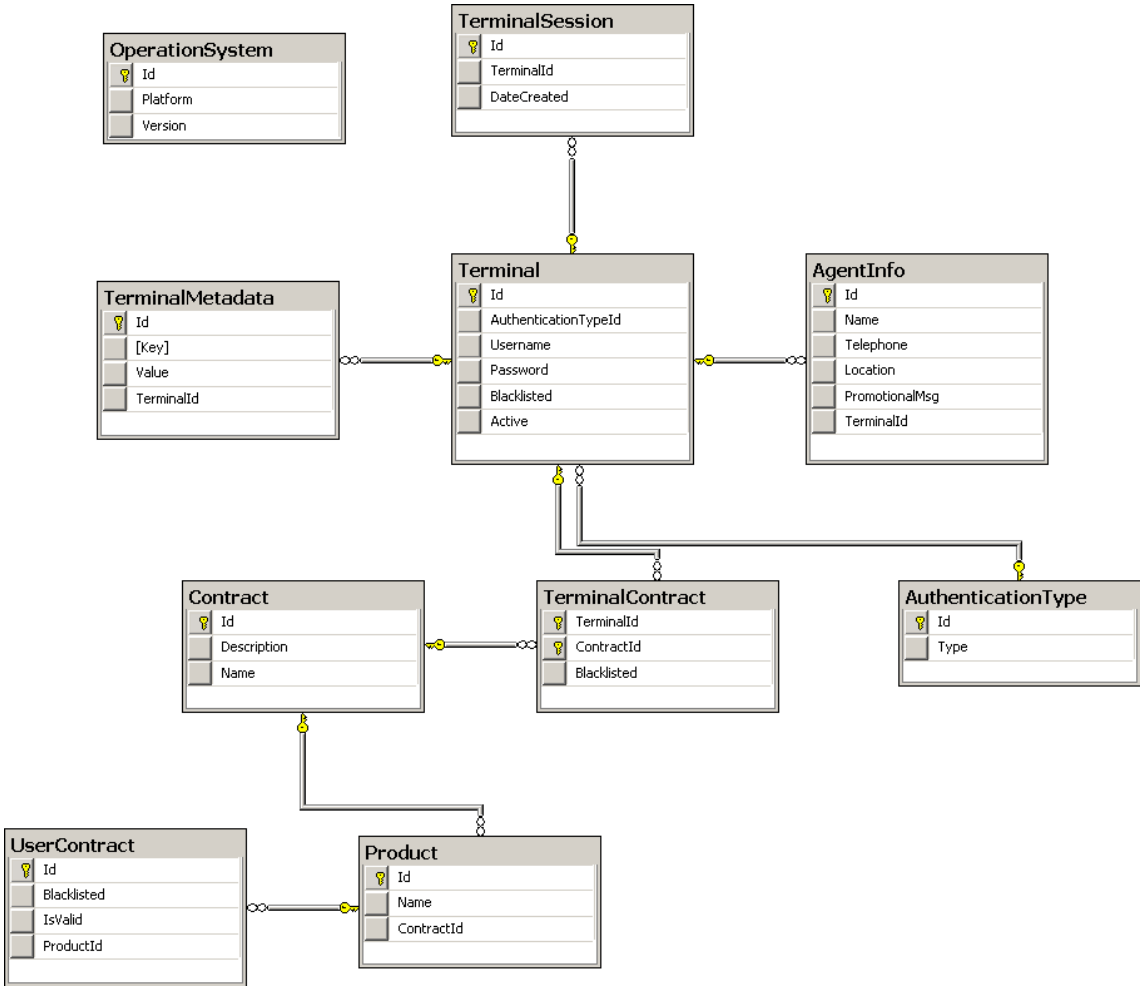

Figura 34 – Modelo Entidade-Associação

9.5 Resultados

9.5.1 **Resultados da 1ª Fase de todos os métodos relativamente ao 1º cenário**

Tabela 20 - Resultados da 1ª Fase de todos os métodos relativamente ao 1 cenário

Tipo de Rede	**Rede Ethernet (802.3)**						**Rede WiFi (802.11G)**					
Tipo de Instância	**m1.Small**			**m1.Large**			**m1.Small**			**m1.Large**		
Pedidos por Segundo	1pps	100pps	200pps	1pps	100pps	200pps	1pps	100pps	200pps	1pps	100pps	200pps
Método *GetCatalog*	80,8	99,9	828,9	78,2	98,0	702,4	81,6	101,2	876,8	79	100,2	818,6
Método *TransactionConfirm*	66,6	93,7	784,9	66,8	88,2	620,6	67,8	93,4	805,1	66,4	89,7	718,6
Método *TransactionDo*	67	92,0	801,0	67,8	90,2	604,9	69,8	92,7	794,8	69,6	90,7	719,1
Método *TransactionValidation*	68,6	96,6	800,8	68,4	93,6	611,5	67,4	95,5	800,7	68	91,7	739,5
Método *SessionOpen*	82,2	106,2	975,9	81,2	99,4	892,9	80,6	113,7	992,4	80,6	108,5	908,1
Tipo de Rede	**Rede WCDMA-HSUPA/3G**						**Rede 4G – LTE**					
Tipo de Instância	**m1.Small**			**m1.Large**			**m1.Small**			**m1.Large**		
Pedidos por Segundo	1pps	100pps	200pps	1pps	100pps	200pps	1pps	100pps	200pps	1pps	100pps	200pps
Método *GetCatalog*	405	577,7	1893,3	398,3	567,3	1615,2	159,3	239,6	1276,7	153,1	228,5	1239,7
Método *TransactionConfirm*	352,1	402,9	1713,4	361,4	392,6	1614,5	148,6	211,7	1196,7	132,8	205,6	1052,2
Método *TransactionDo*	361,4	391,3	1648,5	358,2	386,9	1520,4	138,5	205,8	1105,4	122,8	200,7	1153,5
Método *TransactionValidation*	381,8	428,8	1548,3	373,1	401,4	1415,1	127,5	216,7	1005,3	128,5	212,2	1073,4
Método *SessionOpen*	438,5	634,5	2126,1	434,9	605,7	1911,2	142,9	226,4	1297,8	132,7	218,7	1207,9

9.5.2 **Resultados da 1ª Fase de todos os métodos relativamente ao 2º cenário**

Tabela 21 - Resultados da 1ª Fase de todos os métodos relativamente ao 2 cenário

Tipo de Rede	**Rede Ethernet (802.3)**						**Rede WiFi (802.11G)**					
Tipo de Instância	**m1.Small**			**m1.Large**			**m1.Small**			**m1.Large**		
Pedidos por Segundo	1pps	100pps	200pps	1pps	100pps	200pps	1pps	100pps	200pps	1pps	100pps	200pps
Método *GetCatalog*	82,8	90,6	797,8	79,6	89,3	706,1	82,8	92,3	802,9	81,2	91,6	727,5
Método *TransactionConfirm*	67	91,7	802,7	68,4	89,5	702,3	66	98,7	827,3	68,2	94,7	691,5
Método *TransactionDo*	65,2	98,5	859	65	97,6	705,7	63,2	103,9	821,5	63,8	100,1	668,9
Método *TransactionValidation*	72,8	95,1	811,6	70,5	93,1	696,3	71,8	99,1	788	69,8	94,9	704
Método *SessionOpen*	78,8	136,8	1010,7	80,4	134,4	907,1	82,6	141	917,1	80,2	139,8	929,7
Tipo de Rede	**Rede WCDMA-HSUPA/3G**						**Rede 4G – LTE**					
Tipo de Instância	**m1.Small**			**m1.Large**			**m1.Small**			**m1.Large**		
Pedidos por Segundo	1pps	100pps	200pps	1pps	100pps	200pps	1pps	100pps	200pps	1pps	100pps	200pps
Método *GetCatalog*	379,5	556,6	1879,4	382	496,7	1619,5	151,8	241,4	1146,7	147,3	233,5	1100,4
Método *TransactionConfirm*	330,5	403,2	1683	324,8	387,7	1488,5	124,4	203,3	1046,8	122,9	201,8	964,6
Método *TransactionDo*	358,2	404,2	1660,1	367,4	389,7	1447,8	127,5	211,9	1108,1	118,6	208,1	1077,2
Método *TransactionValidation*	390,4	430,1	1483,9	385,4	394,6	1329	121,5	214,6	1111,5	117,4	196,7	1033,9
Método *SessionOpen*	447,3	637	2105,3	425,5	621,2	1886,3	138,8	242,2	1243,9	131,9	233,8	1204,3

9.5.3 **Resultados da 1ª Fase de todos os métodos relativamente ao 3º cenário**

Tabela 22 - Resultados da 1ª Fase de todos os métodos relativamente ao 3 cenário

Tipo de Rede	**Rede Ethernet (802.3)**								**Rede WiFi (802.11G)**							
Tipo de Instância	**m1.Small**				**m1.Large**				**m1.Small**				**m1.Large**			
Pedidos por Segundo	1pps	100pps	200pps	300pps	1pps	100pps	200pps	300pps	1pps	100pps	200pps	300pps	1pps	100pps	200pps	300pps
Método *GetCatalog*	82,2	90,4	162,5	647,0	81,2	89,3	153,8	548,9	81,6	90,0	161,7	632,3	81	90,4	156,2	565,6
Método *TransactionConfirm*	69	86,4	157,6	612,3	67,8	84,9	142,8	549,0	68,2	89,0	159,8	590,6	66,8	87,8	142,6	533,3
Método *TransactionDo*	69,6	92	150,2	683,1	68,6	89,8	142,6	618,5	69,4	89,9	158,1	636,4	66,6	87,9	149,2	539,9
Método *TransactionValidation*	70,8	92,7	150,7	613,1	70,2	90,9	137,1	523,7	69,2	91,3	154,4	638,5	67,3	89,8	147,3	599,6
Método *SessionOpen*	78,8	133,3	205,1	799,3	77,4	125,8	197,5	707,6	81	136,8	220,5	784,1	79,6	130,4	203,1	683,9
Tipo de Rede	**Rede WCDMA-HSUPA/3G**								**Rede 4G - LTE**							
Tipo de Instância	**m1.Small**				**m1.Large**				**m1.Small**				**m1.Large**			
Pedidos por Segundo	1pps	100pps	200pps	300pps	1pps	100pps	200pps	300pps	1pps	100pps	200pps	300pps	1pps	100pps	200pps	300pps
Método *GetCatalog*	376,5	447,4	827,0	1385,0	361,4	415,4	780,8	1211,9	147,6	206,7	429,6	1395,6	142,5	201,3	394,8	1306,2
Método *TransactionConfirm*	348,7	340,2	680,1	1193,7	331,8	313,8	636,4	982,6	132,8	187,3	386,9	1253,7	127,5	179,6	364,3	1185,6
Método *TransactionDo*	366,5	368,6	611,5	1162,5	362	337,8	560,4	988	128,5	194,6	396,3	1284,8	120,8	173,5	373,5	1153,8
Método *TransactionValidation*	376,6	405,5	688,0	1210	368,2	392,1	605,8	1040,8	129,5	189,6	373,5	1285,7	116,3	179,9	368,3	1086,6
Método *SessionOpen*	447,5	527,3	1076,1	1574,5	442,1	492,5	883,7	1309	138,4	212,4	401,6	1486,5	128,5	192,2	392,2	1374,5

9.5.4 **Resultados da 2ª Fase de todos os métodos relativamente ao 1º cenário**

Tabela 23 - Resultados da 2ª Fase de todos os métodos relativamente ao 1º cenário

Tipo de Rede	**Rede Ethernet (802.3)**										**Rede WiFi (802.11G)**									
Tipo de Instância	**m1.Small**					**m1.Large**					**m1.Small**					**m1.Large**				
Pedidos por Segundo	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps
Método *GetCatalog*	72,7	102	99,4	104,7	98	72,3	100,5	104,7	102,9	97	74,6	102,1	101	102,5	102,4	73,8	101,6	102,3	104,1	102,6
Método *TransactionConfirm*	71,1	95,8	95,3	92,6	93,9	69,4	93,8	94,1	89,3	101,1	70,4	93,4	92,5	92,9	91,9	68,1	88,9	93,4	92,7	97,6
Método *TransactionDo*	70,5	91,1	95,4	95,5	92,1	71,2	93,6	95,3	92,1	90,3	69,5	93,7	95,4	93,1	92,5	69	92,7	94	95,5	95,2
Método *TransactionValidation*	69,6	95,3	94,5	92	93,1	70,1	94,6	94	97,2	93,4	71,3	96,1	96,9	92,6	94	68,2	97,2	93,5	96,6	99,2
Método *SessionOpen*	74	103,7	102,5	104,6	101,8	73,6	108,4	106,2	108,3	109,9	78,6	111,7	108,5	109,3	104,6	77,2	106,5	106	107,1	110,3
Tipo de Rede	**Rede WCDMA-HSUPA/3G**										**Rede 4G – LTE**									
Tipo de Instância	**m1.Small**					**m1.Large**					**m1.Small**					**m1.Large**				
Pedidos por Segundo	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps
Método *GetCatalog*	403,1	577,7	564,7	577,2	601,8	394,5	563,2	570,4	550,8	581,1	144,6	231,6	240,6	239,6	226,9	140,6	227,4	235,6	225,7	229
Método *TransactionConfirm*	377,5	395,8	403,6	394,1	408,4	380,1	392,8	411,3	401,5	403,9	134,5	204,2	213	201,5	217,3	131,5	204,3	207,8	209,5	203,3
Método *TransactionDo*	390,2	423,3	421,9	411,5	422,9	390,4	401,6	420	429,7	417,7	144,7	215,4	211,5	217,5	220	129,6	202,5	205	211,7	210,9
Método *TransactionValidation*	393,3	414,2	425,3	421,7	414,8	386,1	437,7	437,3	435,6	431,6	130,5	211,8	212,4	207,5	217	134,5	205,2	208,8	201	204,6
Método *SessionOpen*	427,4	599,4	663,9	639,2	655,5	416,3	612	633,8	609,6	647,5	164,6	246,3	250,3	248,6	249,3	170,9	249,5	236,4	235,5	243,8

9.5.5 **Resultados da 2ª Fase de todos os métodos relativamente ao 2º cenário**

Tabela 24 – Resultados da 2ª Fase de todos os métodos relativamente ao 2º cenário

Tipo de Rede	**Rede Ethernet (802.3)**										**Rede WiFi (802.11G)**									
Tipo de Instância	**m1.Small**					**m1.Large**					**m1.Small**					**m1.Large**				
Pedidos por Segundo	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps
Método *GetCatalog*	76,4	97,5	101,4	100	104,6	74,5	99,2	98	102,5	101,8	78,2	96,1	104,2	104,6	102,5	77,2	97,6	98,9	101,6	100,5
Método *TransactionConfirm*	73,5	90,4	92	90,1	94,2	69,9	92,5	89,7	91,5	92,9	73,7	97,5	102	96,8	99	74	95,7	93,5	97,1	93,9
Método *TransactionDo*	68,9	94,9	93,7	96,2	94,5	70,3	95,6	91,6	96,4	95,2	72	97,4	93,5	97,4	93,8	72,7	94,1	89,5	92,5	96,2
Método *TransactionValidation*	68,4	93,1	91,6	96,8	95,2	70,4	94,5	95,2	95,1	94,7	70,2	95,9	93,1	95,7	99,1	67,8	94,2	96,4	91,3	92,2
Método *SessionOpen*	78,3	114,6	121,5	110,7	112	81	106,7	111,7	109,5	108	83,4	104,7	114,4	125,1	117,9	86	108,4	106,8	113,9	120
Tipo de Rede	**Rede WCDMA-HSUPA/3G**										**Rede 4G - LTE**									
Tipo de Instância	**m1.Small**					**m1.Large**					**m1.Small**					**m1.Large**				
Pedidos por Segundo	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps
Método *GetCatalog*	423,8	568,6	576,5	558,9	563,6	414,7	559,3	543,2	556,9	542,3	142,8	221,9	231,6	227,5	218,3	138,5	225,4	227	219,6	219,3
Método *TransactionConfirm*	411,5	415,5	422,1	428,6	419,5	398,6	423,1	432,7	424,6	432,6	124,9	209,7	218,6	215,4	219,7	122,8	204,7	206,6	203,7	210,8
Método *TransactionDo*	384,2	429	422,7	418,2	407,6	403,1	438,5	447,2	421,8	413,9	134,1	216,4	219,4	219	211,1	131,5	201,9	198,7	205,7	189,1
Método *TransactionValidation*	400,3	410,6	428,5	422	415,7	385,3	428,4	436,9	429,8	427,5	129,6	211,1	206,2	202,8	213,3	131,4	196,7	208,6	189,8	202,1
Método *SessionOpen*	433,8	612,7	641,1	653,6	639	421,9	602,1	623,6	628,6	631,7	157,5	241,4	243,6	252,1	246,8	152,6	231,2	241,6	236,3	228,1

9.5.6 **Resultados da 2ª Fase de todos os métodos relativamente ao 3º cenário**

Tabela 25 - Resultados da 2ª Fase de todos os métodos relativamente ao 3º cenário

Tipo de Rede	**Rede Ethernet (802.3)**										**Rede WiFi (802.11G)**									
Tipo de Instância	**m1.Small**					**m1.Large**					**m1.Small**					**m1.Large**				
Pedidos por Segundo	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps
Método *GetCatalog*	72,6	94,7	97,5	98,4	94,2	72,7	96,3	93,6	91	92,9	74,7	95,2	98,4	95,2	95,1	75,5	96,9	94,6	95,4	94,8
Método *TransactionConfirm*	68	90,5	91,6	89,3	92,4	70,1	88,6	89,2	86,9	90	71,4	92,4	93,9	94	93,2	70,4	88,3	90,5	90,9	87,4
Método *TransactionDo*	70,9	89,6	93,1	92,8	88,9	67,3	93,5	91,8	87,4	89,8	68,3	89,7	90,7	89,6	92,6	72,5	87,3	89	88,5	89,3
Método *TransactionValidation*	69,1	92,7	89,5	92,5	90,1	67,2	91	88,4	89,1	92,7	68,4	92	93,7	90,9	88,4	68	90,5	88,4	87,5	89,2
Método *SessionOpen*	75,6	108,4	101,5	104,3	107,8	73,9	101,6	96,8	97,3	103,2	77,6	107,3	103,1	102	107,4	77,4	97,4	99,4	100,3	98
Tipo de Rede	**Rede WCDMA-HSUPA/3G**										**Rede 4G - LTE**									
Tipo de Instância	**m1.Small**					**m1.Large**					**m1.Small**					**m1.Large**				
Pedidos por Segundo	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps	1pps	100pps	200pps	300pps	800pps
Método *GetCatalog*	418,7	460	448,5	449,3	437,3	405,7	407,4	417	412,5	410,8	140,7	218,6	206,7	210,3	218	138,7	197,7	205,5	196,3	199
Método *TransactionConfirm*	366,4	404,6	421,6	418,9	419,6	365	375,8	384,8	379,4	391,7	114,7	201,8	194,6	185,3	193,7	116	185,7	189	177,3	186,2
Método *TransactionDo*	368,1	411,6	430,6	406,8	415,7	358,6	373,1	365	371,8	387,2	117	186,6	202,6	194,6	184,8	120,7	185,7	194,3	190	195,5
Método *TransactionValidation*	394,3	410,9	415,6	410	407,2	382,5	398,5	402,5	389,6	394,1	120,3	199,2	190	194,1	189,2	120,9	191,6	188,5	192,5	187,7
Método *SessionOpen*	426,7	543,7	541,7	550,3	537,6	430,3	498,3	501,7	518,7	516,9	148,4	239,7	222,5	218,7	226,9	148	220,6	215,2	217,6	215,9