DANIEL NORIO TAKASU REBELO
LUCAS ARTHUR FELGUEIRAS
LUIZ GUSTAVO DOS SANTOS
VICTOR FRANÇA FERREIRA

COMPUTAÇÃO EM NUVEM: INTRODUÇÃO

São Paulo
2018

DANIEL NORIO TAKASU REBELO
LUCAS ARTHUR FELGUEIRAS
LUIZ GUSTAVO DOS SANTOS
VICTOR FRANÇA FERREIRA

COMPUTAÇÃO EM NUVEM: INTRODUÇÃO

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para a disciplina de Engenharia de Sistemas.

São Paulo
2018

DANIEL NORIO TAKASU REBELO
LUCAS ARTHUR FELGUEIRAS
LUIZ GUSTAVO DOS SANTOS
VICTOR FRANÇA FERREIRA

COMPUTAÇÃO EM NUVEM: INTRODUÇÃO

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para a disciplina de Engenharia de Sistemas.

Área de Concentração:
Engenharia de Sistemas

Orientador:
Profª. Drª. Selma Shin Shimizu Melnikoff

São Paulo
2018

RESUMO

A tecnologia de computação em nuvem é uma tendência para as novas aplicações do mundo. Grandes empresas estão migrando suas estruturas para a nuvem, outras constroem ambientes para abrigar essas novas aplicações. O objetivo do trabalho consiste em compreender melhor como funciona essas tecnologias, sua motivação e as principais opções disponíveis no mercado para uso.

Palavras-Chave – Nuvem, Arquitetura, Computação, Engenharia, Programa.

ABSTRACT

Cloud computing technology is a trend for new applications around the world. Large companies are migrating their structures to the cloud, others build environments to house these new applications. The objective of the work is to better understand how these technologies work, their motivation and the main options available in the market to use.

Keywords – Cloud, Architecture, Computing, Enginnering, Software.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

PARTE I

INTRODUÇÃO

1 OBJETIVO

First to mind when asked what 'the cloud' is, a majority respond it's either an actual cloud, the sky, or something related to weather.

-- Citrix Cloud Survey Guide(**?**)

Com o desenvolvimento da computação, novos programas foram criados, cada vez mais consumindo recursos e sendo hospedados em máquinas pessoais ou dedicadas, porém sem um gerenciamento inteligente de toda a infraestrutura. Além disso, houve a nítida migração de programas **distribuídos** pela internet para programas que **rodem** na internet.

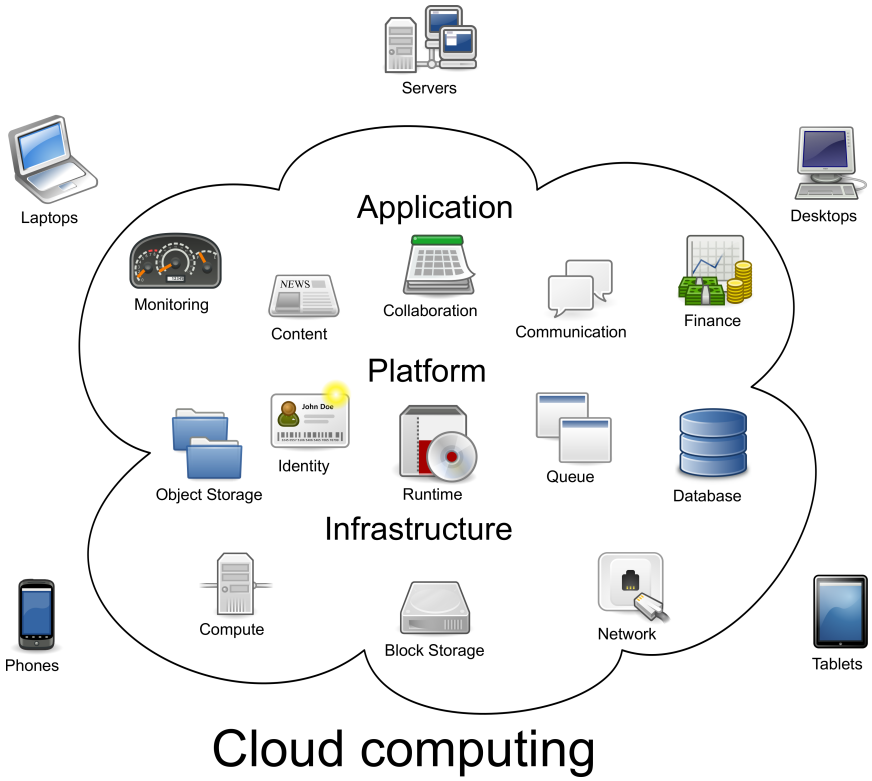

Figura 1: Arquitetura Básica de Computação em Nuvem(**?**)

Essas mudanças de paradigma motivaram a criação e consolidação do que conhecemos hoje como Computação em Nuvem, área fortalecida e necessária no desenvolvimento de programas modernos. Ao longo dos próximos capítulos, haverá a explanação em detalhes de como funciona uma arquitetura em nuvem básica, seus desafios técnicos e as principais soluções existentes no mercado.

2 HISTÓRICO

I don't need a hard disk in my computer if I can get to the server faster... carrying around these non-connected computers is byzantine by comparison.

-- Steve Jobs(**?**)

2.1 Contexto

A computação, em meados dos anos 90, consistia na evolução do computador pessoal e na consolidação da internet pelo mundo todo. A essência das aplicações construídas estava na distribuição de programas pela internet. Com o advento de aplicações web, torna-se necessário hospedar essas aplicações em servidores. Geralmente, era uma máquina física dedicada ao serviço, onde a aplicação ficaria hospedada, recebendo suas requisições.

2.2 Problemas

No contexto da época, esses servidores dedicados falhavam com certa frequência. Em casos de falha, o administrador do sistema precisava colocar uma nova máquina para atender a demanda, enquanto entendia a falha de *software* ocorrida. Porém, toda a provisão de novos servidores para gerar redundância e tolerância a falhas aumentava a probabilidade de máquinas físicas falharem, gerando maiores problemas e dificuldades para manutenção.

2.3 Primórdios

Em 2006, a Amazon, percebendo a oportunidade de diversificar seu negócio atuando com provisionamento de infraestrutura, gerenciando máquinas ociosas de acordo com os horários possíveis e cuidando de toda a manutenção trabalhosa pelo lado do desenvolvedor, formulou e lançou a primeira versão do Amazon Elastic Compute Cloud (EC2), o primeiro produto voltado para a computação em nuvem, com o propósito de atender essa necessidade encontrada e vender infraestrutura para o usuário final.

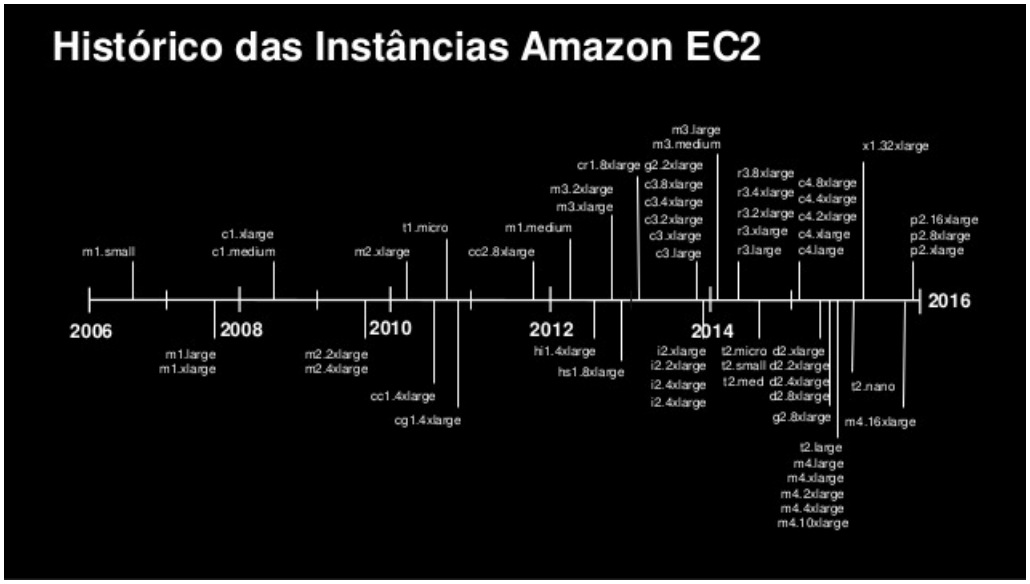

Figura 2: Histórico do Amazon EC2(**?**)

A postagem original no site da Amazon anunciava o serviço (em inglês)(**?**):

> Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale computing easier for developers. Just as Amazon Simple Storage Service (Amazon S3) enables storage in the cloud, Amazon EC2 enables "compute" in the cloud. Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change. Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you actually use.

2.4 Impactos

Essencialmente, a Amazon comercializou o conceito de infraestrutura como serviço (*IaaS*). A revolução iniciada na época permitiu que muitas aplicações hospedadas em servidores físicos fossem migrados para a infraestrutura da Amazon. A cobrança realizada por hora de instância virtual carregada era bem inferior aos custos (tanto reais quanto de trabalho) de manter uma infraestrutura física equivalente, fora o fato de que a ociosidade foi drasticamente reduzida, pois o gerenciamento inteligente do EC2 permitia que uma mesma máquina atendesse dois serviços em horários distintos.

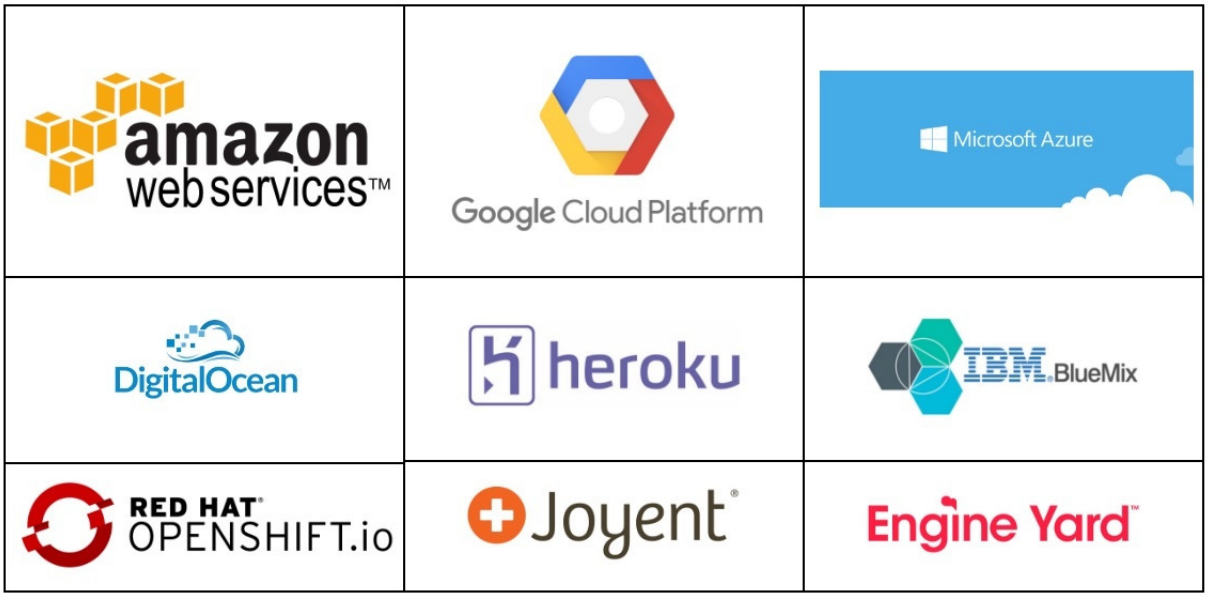

Figura 3: Algumas soluções de Computação em Nuvem(**?**)

Com o trunfo do EC2, outras soluções ganharam força, merecendo destaque para: **Salesforce Heroku**, **Google Cloud** e **Microsoft Azure**, que serão exploradas adiante.

PARTE II

ASPECTOS CONCEITUAIS

3 CONCEITOS BÁSICOS

Este capítulo introduz as definições e os conceitos básicos quando se trata de Cloud Computing.

3.1 Definição de Cloud Computing

Computação em Nuvem (do Inglês *Cloud Computing*) é segundo a NIST (*National Institute of Standards and Technology*) um modelo para prover um acesso de rede a um grupo compartilhado (*shared pool*) de recursos computacionais (recursos como capacidade de rede, servidores, armazenamento, aplicações e serviços) de forma ubíqua, prática e sob demanda.

Tal acesso deve ser rapidamente provisionado e lançado com o mínimo de gerenciamento e interação com o provedor de serviços por parte da aplicação.

O modelo de computação em nuvem deve possuir algumas características básicas, que estão descritas na secção seguinte.

3.2 Características básicas de Cloud Computing

A NIST define 5 características essenciais do modelo de Cloud Computing:

- Serviço sobre demanda: Um consumidor pode provisionar unilateralmente capacidades computacionais, como tempo de servidor e armazenamento de rede, conforme for necessário, sem qualquer interação humana com o provedor de serviço;
- Agrupamento de recursos: Os recursos de computação do provedor são agrupados para atender a vários consumidores usando um modelo "multi inquilino", com diferentes recursos físicos e virtuais dinamicamente atribuídos e reatribuídos de acordo com a demanda do consumidor. Existe uma sensação de independência de loca-

lização pois o cliente geralmente não tem controle ou conhecimento sobre a localização exata dos recursos fornecidos, mas pode ser capaz de especificar a localização em um nível abstração (por exemplo, país, estado ou datacenter). Exemplos de recursos incluem armazenamento, processamento, memória e largura de banda de rede.

- Amplo acesso à rede: Os recursos estão disponíveis na rede e são acessados por meio de mecanismos que promovam o uso por plataformas heterogêneas por clientes em diversos dispositivos (por exemplo, telefones celulares, tablets, laptops e estações de trabalho). Esta característica promove o conceito de computação ubíqua, isto é, em toda parte, onipresente.

- Elasticidade rápida: Os recursos podem ser provisionados e liberados elasticamente, em alguns casos automaticamente, proporcionando uma escalabilidade crescente ou descrescente conforme a demanda. Os recursos disponíveis normalmente aparentam ser ilimitados para o consumidor, podendo ser requisitidados em qualquer quantidade e a qualquer momento.

- Serviço mensurável: Os sistemas em nuvem controlam e otimizam automaticamente o uso de recursos, aproveitando-se de uma capacidade de medição em um nível de abstração apropriado ao tipo de serviço (por exemplo, armazenamento, processamento, largura de banda e contas de usuário ativas). O uso de recursos pode ser monitorado, controlado e reportado, gerando transparência tanto para o fornecedor e consumidor do serviço utilizado.

4 MODELOS PARA CLOUD COMPUTING

Este capítulo introduz brevemente os modelos de serviço de cloud computing que podem ser adotados por um provedor e os modelos de *deployment*, nas secções seguintes.

4.1 Modelos de serviços

A teoria por trás dos serviços de computação em nuvem abrange três elementos principais: software, plataforma e infraestrutura. Temos os seguintes modelos de serviço:

- **SaaS, Software as a Service** ("Software como um serviço")

  Neste modelo é oferecido ao consumidor o uso de aplicações de um provedor que rodam sobre uma infraestrutura em nuvem. Estas aplicações são acessíveis a partir de vários dispositivos clientes por meio de uma interface simples, como um navegador da web, ou uma interface de por meio de um programa (mobile ou desktop).

  O consumidor não gerencia ou controla a infraestrutura de nuvem por trás da aplicação, incluindo rede, servidores, sistemas operacionais, armazenamento ou capacidade da aplicação individual. São disponibilizadas apenas configurações do aplicativo específicas para aquele usuário individual.

  Alguns exemplos de SaaS são serviços de *webmail*, *streaming* de vídeos, conversão de arquivos e trabalho colaborativo com arquivos. Este modelo não será mais detalhado neste trabalho.

- **PaaS, Platform as a Service** ("Plataforma como um serviço")

  A capacidade fornecida ao consumidor é de implantar na nuvem aplicações criadas por meio de linguagens de programação, bibliotecas, serviços e ferramentas suportadas pelo provedor. Tais aplicações podem ser criadas pelo próprio consumidor, ou consumidas por este.

Assim como no modelo de SaaS, o consumidor geralmente não gerencia ou têm controle sobre a infraestrutura de nuvem por trás, incluindo rede, servidores, sistemas operacionais, ou armazenamento. Porém o consumidor as tem controle sobre os aplicativos implantados e possivelmente sobre definições de configuração para o ambiente de hospedagem do aplicativo.

Como exemplos de provedores no mercado temos *IBM Bluemix*, *Heroku*, e *Windows Azure Cloud*.

- **IaaS, Infrastructure as a Service** ("Infraestrutura como um serviço")

  A capacidade oferecida ao consumidor é provisionar processamento, armazenamento, redes e outros recursos fundamentais de computação onde o consumidor é capaz de implantar e executar software arbitrário, incluindo-se sistemas e aplicações. O consumidor não gerencia nem controla a infraestrutura de nuvem por trás, mas tem controle sobre sistemas operacionais, armazenamento e aplicativos implantados. Possivelmente possui também um controle limitado de componentes de rede (por exemplo, firewalls de host). Geralmente acompanha serviços de máquinas virtualizadas.

  Alguns exemplos de provedores no mercado são *Amazon Web Services*, *Microsoft Azure*, *Google Cloud* e *VMware Cloud on AWS*.

4.2 Modelos de implantação

Os modelos de implantação (*deployment models*) a seguir delimitam as formas possíveis de tarifação dos serviços, as possíveis localizações da infraestrutura física e o público de escopo. São eles:

- **Nuvem privada**: A infraestrutura de nuvem é disponibilizada para uso exclusivo por uma única organização, compreendendo vários consumidores (*business units*). Pode ser propriedade e gerenciada pela própria organização, um terceiro ou alguma combinação de ambos, e pode existir dentro ou fora das instalações da organização.

- **Nuvem comunitária**: A infraestrutura em nuvem é de uso exclusivo por uma comunidade de consumidores de organizações que compartilham mesmos interesses (por exemplo, missão, requisitos de segurança, políticas e considerações de conformidade). Pode ser de propriedade e administrada por uma ou mais organizações da

comunidade, um terceiro, ou alguma combinação de ambos, e pode existir dentro ou fora das instalações.

- **Nuvem pública**: A infraestrutura de nuvem é para o uso aberto pelo público em geral. Pode ser propriedade ou gerenciada por uma organização comercial, acadêmica ou governamental, ou alguma combinação destes. Existe dentro das instalações do provedor de nuvem.

- **Nuvem híbrida**: A infraestrutura de nuvem é uma composição de duas ou mais infraestruturas de nuvens distintas (privadas, comunitárias ou públicas) que permanecem como entidades únicas, mas unidas por tecnologia padronizada ou proprietária permitindo portabilidade de dados e aplicações (por exemplo, *cloud bursting* para balanceamento de carga entre nuvens).

5 ARQUITETURA

Este capítulo apresenta uma arquitetura de referência para cloud computing na primeira secção. Na secção final apresenta boas práticas e métodos para arquitetar uma aplicação que deve ser colocada em um ambiente de nuvem.

5.1 Arquitetura de referência

O objetivo da arquitetura de referência apresentada a seguir é de providenciar uma taxonomia simples e sem ambiguidade para os três modelos de serviço:

- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

Tal arquitetura deve disponibilizar uma visão unificada das cinco características essenciais da NIST:

- Serviço sobre demanda
- Amplo acesso à rede
- Agrupamento de recursos
- Elasticidade rápida (escalabilidade)
- Serviço mensurável

Após pesquisar informações disponibilizadas por provedores, instituições de pesquisa e de consultoria na área achamos interessante a seguinte arquitetura de referência:

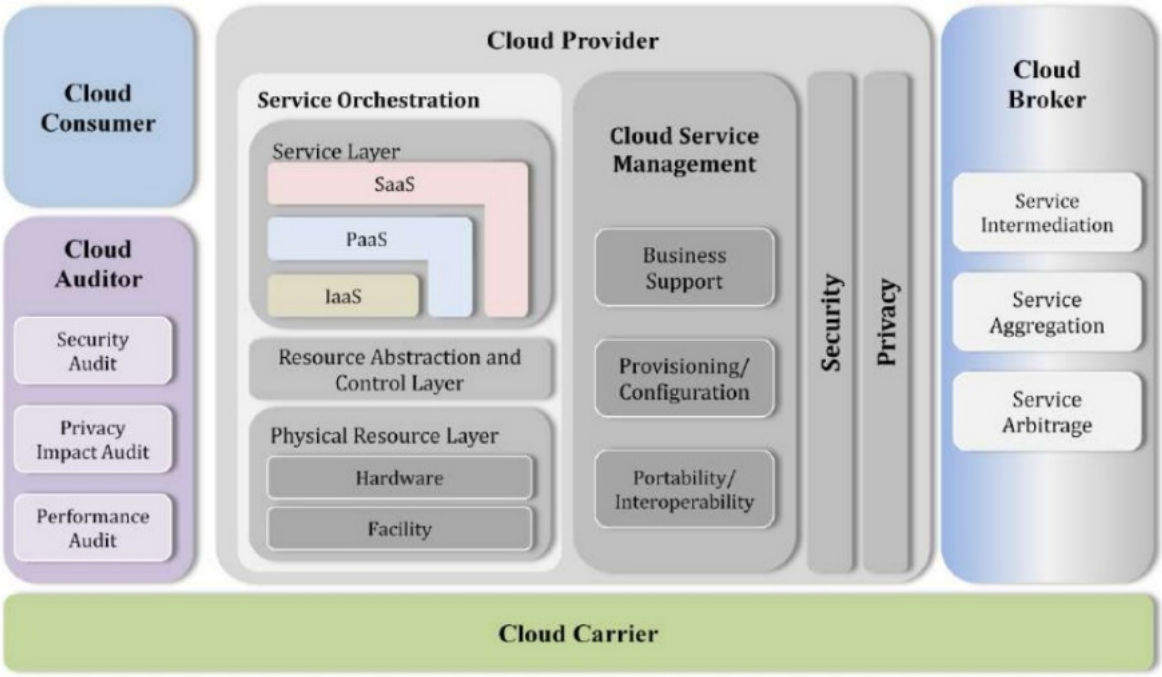

Figura 4: Arquitetura de referência para Cloud Computing

5.1.1 Cloud Consumer

Representa classes de consumidores para cada modelo possível de serviço.

No caso de SaaS são diversos departamentos operacionais de uma empresa, consumindo serviços de ERP, CRM, vendas, finanças, RH, redes sociais, webmail etc.

No caso de PaaS são os setores de análise de dados e desenvolvimento, que consomem serviços de BI, banco de dados, *deployment* de aplicações, integração de sistemas e desenvolvimento e testes.

Por fim, os consumidores de IaaS são os setores da empresa responsável pela infraestrutura e IT, que consomem serviços de backup e recuperação, gerenciamento de serviços, armazenamento, hospedagem de plataformas etc.

5.1.2 Cloud Auditor

Cloud Auditor é o componente responsável por relatar e registrar as ações relativas a segurança, privacidade e performance da nuvem, tanto para fins de transparência para os consumidores quanto de regulamentação governamental para rastrear atividades suspeitas e irregulares. Pode ser tanto um órgão governamental ou uma entidade fiscalizadora global na Internet.

5.1.3 Cloud Provider

Cloud Provider é o provedor de nuvem como a *Amazon Web Services*, *Microsoft Azure*, *Netflix*, *Youtube*, dentre outros. Dependendo das categorias de modelo de serviço em que o provedor se encaixa, ele poderá ter camadas de serviço de SaaS, PaaS ou IaaS.

As principais atividades do Cloud Provider são a implantação de serviços, a orquestração de serviços, o gerenciamento de serviços, a segurança e por fim, a privacidade. Deve gerenciar e abstrair toda lógica de controle e a camada física de recursos de hardware, bem como as instalações e locais físicos.

5.1.4 Cloud Broker

O Cloud Broker é responsável por uma espécie de serviço de corretagem. Este ator no ecossistema de nuvem é responsável por oferecer serviços de proteção jurídica durante a negociação de contratos com provedores de nuvem.

Oferece também serviços de diminuição de custos com o provedor, agregando diversos serviços a fim de obter o melhor custo-benefício.

Por fim, pode oferecer serviços diferenciados, como um serviço adicional de segurança, para proteger o consumidor de vazamento de dados na nuvem.

5.1.5 Cloud Carrier

O Cloud Carrier são todas entidades responsáveis por disponibilizar a conexão e transporte de serviços na nuvem entre consumidores e provedores (Companhias de redes, telecomunicações, dispositivos de acesso e IoT).

PARTE III

ASPECTOS TÉCNICOS

6 ESTRUTURA

6.1 Hypervisor

6.2 Orquestrador

6.3 Armazenamento

6.4 Rede

7 CONSIDERAÇÃO TÉCNICAS SOBRE REQUISITOS

Neste capítulo apresentamos algumas considerações técnicas para cumprir com êxito três principais requisitos não funcionais claros para Cloud Computing: disponibilidade, performance e segurança.

7.1 Disponibilidade

Assume-se que a nuvem sempre estará disponível, 24 horas por dia, 7 dias por semana. Porém nada é a prova de falhas. The cloud is assumed to be always available. But everything can fail. A virtual machine, for example, is hosted on a physical machine that can fail. The virtual network is less likely to fail, but it too is fallible. It behooves the architect of a system to plan for failure. The service-level agreement that Amazon provides for its EC2 cloud service provides a 99.95 percent guarantee of service. There are two ways of looking at that number: (1) That is a high number. You as an architect do not need to worry about failure. (2) That number indicates that the service may be unavailable for .05 percent of the time. You as an architect need to plan for that .05 percent. Netflix is a company that streams videos to home television sets, and its reliability is an important business asset. N etflix also hosts much of its operation on Amazon EC2. On April 2 1 , 20 1 1 , Amazon EC2 suffered a four-day sporadic outage. Netflix customers, however, were unaware of any problem. Some of the things that N etflix did to promote availability that served them well during that period were reported in their tech blog. We discussed their Simian Army in Chapter 10. Some of the other things they did were applications of availability tactics that we discussed in Chapter 5 . • Stateless services. Netflix services are designed such that any service instance can serve any request in a timely fashion, so if a server fails, requests can be routed to another service instance. This is an application of the spare tactic, because the other service instance acts as a spare. • Data stored across zones. Amazon provides what they call ”availability zones,”which are distinct data

centers. Netflix ensured that there were multiple redundant hot copies of the data spread across zones. Failures were retried in another zone, or a hot standby was invoked. This is an example of the active redundancy tactic. • Graceful degradation. The general principles for dealing with failure are applications of the degradation or the retnoval from service tactic: • Fail fast: Set aggressive timeouts such that failing components don't make the entire system crawl to a halt. • Fallbacks: Each feature is designed to degrade or fall back to a lower quality representation. • Feature removal: If a feature is noncritical, then if it is slow it may be removed from any g1ven page.

PARTE IV

SOLUÇÕES DO MERCADO

8 AMAZON WEB SERVICES

8.1 Introdução

9 GOOGLE CLOUD

9.1 Introdução

O Google Cloud é uma das soluções mais completas atualmente no mercado, com diversas possibilidades e serviços disponíveis aos usuários, de acordo com suas necessidades. Possui toda o conjunto de hospedagem, desde uma aplicação pronta até toda a parte de infraestrutura de máquinas para escalabilidade.

Figura 5: Serviços do Google Cloud(**?**)

O Google Cloud possui quatro serviços essenciais de nuvem para aplicações, sendo eles: **App Engine**, **Compute Engine**, **Kubernetes Engine** e **Cloud Functions**. Além disso, possui toda a infraestrutura de gerenciamento de rede, banco de dados, *big data*, *machine learning*, entre outros.

9.2 Funcionamento básico

No contexto do documento, serão destacados os quatro serviços já citados anteriormente:

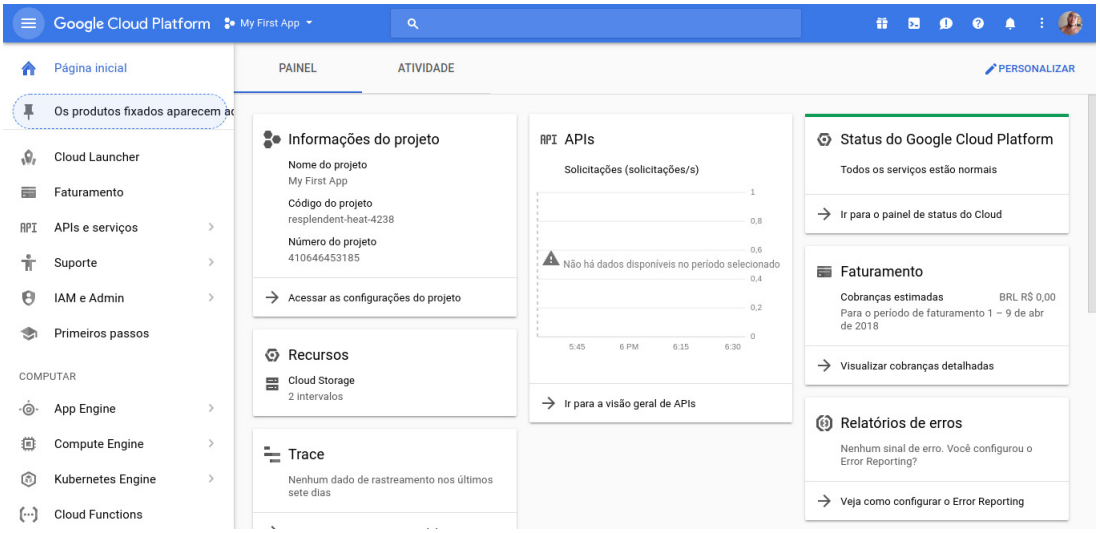

Figura 6: Exemplo do *Dashboard* do Google Cloud com seus recursos

9.2.1 Cloud Functions

As **Cloud Functions** têm como principal funcionalidade atender pequenos trechos de código, gerando *endpoints* prontos para o usuário final utilziar. O produto atende em especial pessoas que não necessitam de grandes aplicações para solucionarem seus problemas, onde um microserviço seria algo elevado demais para o contexto.

O funcionamento básico dele consiste em implementar soluções simples no ambiente de microserviços sobre monólitos, explicado da seguinte forma pelo Google(**?**):

> A agilidade do desenvolvedor vem de sistemas de construção compostos por pequenas unidades de funcionalidade independentes que executam uma única coisa muito bem. O Cloud Functions permite criar e implantar serviços no nível de uma única função, e não no nível de aplicativos, contêineres ou VMs inteiros.

Contextualizando, é possível subir simples funções e deixar o gerenciamento da escalabilidade com o próprio Google Cloud, facilitando a vida do desenvolvedor. Uma solução similar é implementada pela suíte de soluções da Amazon Web Services: as funções *lambda*.

9.2.2 App Engine

O **App Engine** é um serviço intermediário, permitindo que aplicações completas sejam hospedadas em seu serviço, porém com os recursos necessários apenas e sem grandes preocupações em gerenciamento da escalabilidade.

Seu funcionamento consiste em hospedar aplicações completas no estilo *PaaS* (*Platform as a Service*), deixando de lado a parte de gerenciamento de infraestrutura para o Google Cloud. O depoimento abaixo, de Stefan Hauk, Desenvolvedor da empresa de games Rovio, ilustra bem o funcionamento do serviço(**?**):

> Nossos jogos da Web tendem a ficar famosos imediatamente, então não temos a opção de escalonamento ao longo do tempo. O Google App Engine simplifica esse processo, já que ele pode iniciar instantaneamente quantos servidores forem necessários.

A vantagem do serviço está no escalonamento automático simples, o que facilita o trabalho do desenvolvedor. Alem disso, há uma integração tranquila com as principais linguagens de programação usadas, atendendo os mais diversos problemas de maneira simples. A solução é bem parecida com a implementada por padrão pelo Heroku.

9.2.3 Kubernetes Engine

O **Kubernetes Engine** é uma solução mais robusta que permite o uso de *containers*, dando maior poder ao desenvolvedor para escalar sua aplicação em máquinas virtuais. A solução adotada pelo Google usa seu próprio orquestrador de *containers*, o Kubernetes, definido pelo Google (em inglês)(**?**):

> Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications. It groups containers that make up an application into logical units for easy management and discovery.

Seu funcionamento consiste em gerenciar *containers* de acordo com as necessidades dos usuários, onde as aplicações colocadas em produção se encontram dentro deles. Essa ferramenta permite maior controle do desenvolvedor para determinar quantos e quais *containers* serão escalados, para atender a demanda necessária.

9.2.4 Compute Engine

Por fim, o **Compute Engine** é a solução mais poderosa fornecida pelo Google, onde é disponibilizada uma máquina completa para o usuário subir suas aplicações. A solução

é da categoria *IaaS* (*Infrastructure as a Service*), provisionando infraestruturas de acordo com a necessidade do desenvolvedor.

Seu funcionamento se assemelha ao EC2 da Amazon Web Services, onde a essência está em fornecer máquinas completas ao usuário final, com os mais diversos sistemas operacionais (tanto Linux quanto Windows). A escalabilidade é possível, expandindo recursos da máquina de acordo com as necessidades, mas todo o trabalho fica ao encargo do desenvolvedor.

9.3 Outros serviços

Além dos serviços básicos de aplicações, merecem destaque outros serviços:

- Armazenamento: Bigtable (NoSQL), Cloud SQL (MySQL e PostgreSQL).
- Rede: Rede VPC (Gerenciamento de IP), Balanceamento de Carga.
- *Stackdriver*: Monitoração, Depuração e Registros das Aplicações.
- Big Data: BigQuery (Análise de dados massivos), Dataflow (*Pipeline* para processamento de dados), IoT Core (Mensagens em tempo real).

9.4 Conclusão

O Google Cloud possui uma suíte bem completa de produtos, para os mais diversos públicos e problemas a serem solucionados, tanto soluções auto-escaláveis quanto máquinas completas para aplicações mais complexas, tudo com a infraestrutura do Google e a preços mais acessíveis do que concorrentes como o Heroku.

10 HEROKU

10.1 Introdução

O Heroku é uma plataforma de computação em nuvem conhecida no mercado, com a possibilidade de subir aplicações nas linguagens Ruby, Node.js, Java, Python, Clojure, Scala, Go e PHP. O grande destaque do Heroku está na facilidade em subir uma aplicação com facilidade e de maneira gratuita, o que possibilita testar e validar ideias básicas antes de escalar de fato.

A estrutura básica do Heroku funciona com o uso de *dynos*, que servem tanto para hospedar sua aplicação principal quanto máquinas auxiliares para serviços externos e/ou paralelos. Porém, recentemente o Heroku disponibilizou também a estrutura de *containers* em Docker (tecnologia de gerenciamento de *containers*) (assim como seus principais concorrentes), gerando maior flexibilidade para o serviço.

10.2 Funcionamento básico

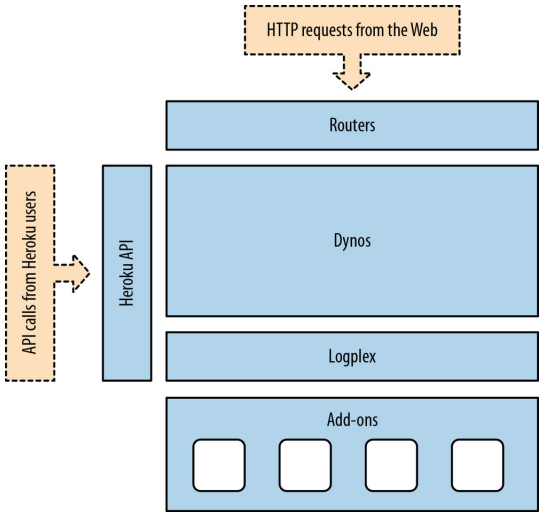

Figura 7: Arquitetura Básica do Heroku(?)

O funcionamento do Heroku consiste no uso de *dynos* para hospedar as aplicações e nos *routers* para tratar e encaminhas as requisições dos usuários. Além disso, o próprio Heroku disponibiliza extensões para gerenciar sua aplicação, como por exemplo o gerenciador de IP estático, ou o banco de dados necessário para a aplicação.

Um *dyno* é um *container* pronto, com 512MB de memória RAM, responsável por abrigar uma ou mais instâncias de sua aplicação, permitindo escalabilidade e tolerância de erros. Além disso, os *dynos* permitem rodas tarefas de sua aplicação à parte, como filas, requisições assíncronas, entre outros.

Já os *routers* são responsáveis por gerenciar os acessos dos usuários às aplicações correspondentes, dado que não é padrão do Heroku estruturar IP estático para cada *dyno*. Ou seja, a estrutura é responsável por fazer sua aplicação funcionar corretamente, sem acessar aplicações alheias.

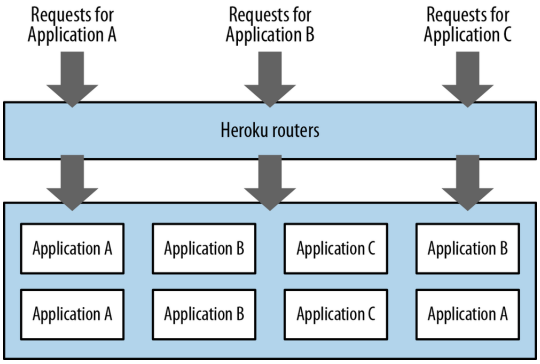

Figura 8: Ligação entre *router* e *dynos* do Heroku(**?**)

Graças a essa estrutura, é possível, com facilidade, gerenciar versionamento e escalar a aplicação de maneira horizontal, com novos *dynos* gerenciados pelo *router*, dando uma popularidade considerável ao Heroku, em especial nas aplicações onde velocidade de validar hipóteses é o principal foco.

10.3 Tarifação

O Heroku possui um diferencial em relação à tarifação: possui um plano básico gratuito que possibilita testar aplicações de maneira fácil e sem dificuldades de expansão. Essencialmente, a cobrança do Heroku ocorre via uso de seus *dynos*-hora. Além disso, há a cobrança pelas extensões usadas dentro da aplicação, onde, no geral, há um plano gratuito de testes ou até para projetos pequenos funcionarem com tranquilidade sem necessidades de escalabilidade.

Para qualquer usuário do Heroku, é disponibilizado um pacote de 750 *dynos*-hora durante um mês, ou seja, uma aplicação gratuita usando um simples *dyno* pode durar tranquilamente. Além disso, o Heroku de uso gratuito limita o tempo em que um *dyno* fica ligado de maneira ociosa: passando trinta minutos desde a última requisição feita pelo cliente (e repassada pelo *router*), o *dyno* é derrubado, sendo religado após uma nova requisição.

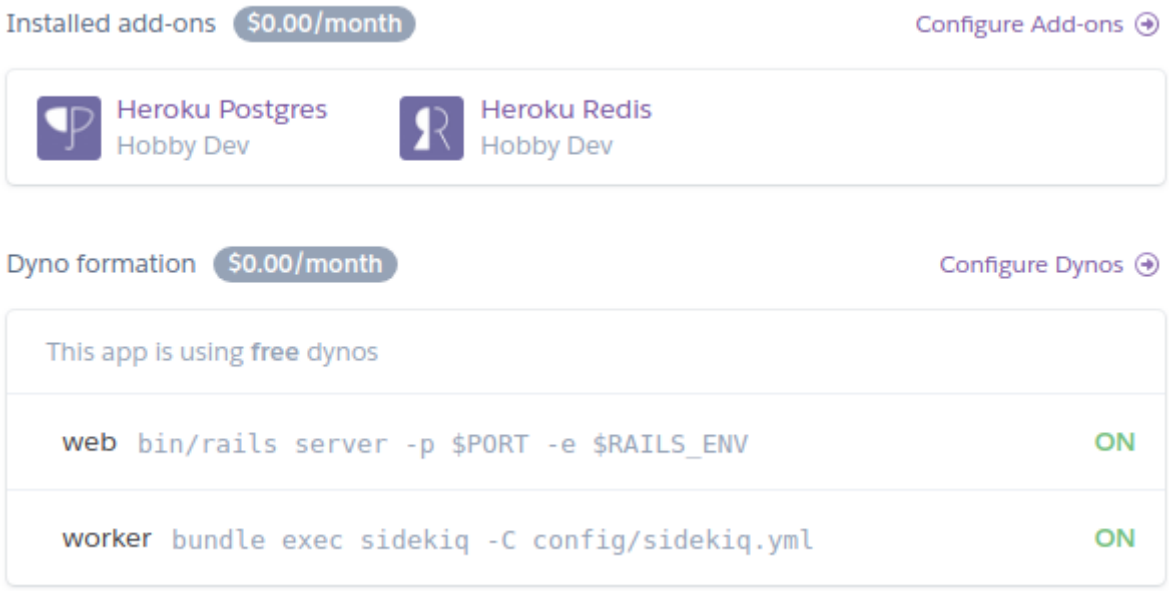

Figura 9: Exemplo do *Dashboard* do Heroku com seus recursos

Além da cobrança do uso dos *dynos*, há também a cobrança para cada extensão utilizada por sua aplicação, o que de fato encarece o custo final para escalar uma aplicação. Por exemplo, o banco de dados relacional padrão do Heroku é gratuito até certo limite de dados, passando desse limite, há a extensão do plano que permite, além de mais dados armazenados, aumenta o fluxo possível de acesso ao banco.

10.4 *Containers*

Recentemente, o Heroku possibilitou em seu catálogo de soluções o uso de estruturas isoladas customizadas: os *containers*. *Containers*, segundo o site do Docker(**?**) (em inglês):

> A container image is a lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings.

Ou seja, são imagens autossuficientes e virtualizadas que permitem rodar aplicações prontas nos mais diversos dispositivos, nos moldes da virtualização.

O Heroku possibilitou, em 2017, uma maneira eficiente de receber *containers* em seu serviço, com o **Heroku Container Registry**(**?**). Esse serviço facilita colocar em produção as máquinas isoladas contruídas com o uso do Docker, aumentando as possibilidades de operação com a plataforma. A tarifação do serviço segue o mesmo padrão do Heroku tradicional, dado o fato de que *dynos* são uma espécie de *containers*.

10.5 Conclusão

O Heroku é uma solução que se destaca pela velocidade e facilidade de colocar uma ideia em produção, de maneira gratuita, com possibilidades eficientes de escalabilidade. Porém, financeiramente, não é a melhor solução, com uma tarifação elevada para escalar a aplicação. Além disso, sua estrutura atende apenas aplicações prontas, não permitindo outras estruturas avançadas como o uso de *containers*, sendo incorporado ao catálogo de produtos recentemente.

11 AZURE

11.1 Introdução

A Azure é uma plataforma de computação em nuvem, agindo como provedor *SaaS, PaaS* e *Iaas*. Tem como proprietária a empresa americana *Microsoft*, e disponibiliza serviços como *builds*, testes, *deploys* e gerenciamento de aplicações que estão hospedadas em *data centers* administrados pela própria empresa.

Atualmente possui mais de 600 ferramentas para os mais diversos usos na área da computação, como serviços de armazenamento, soluções para desenvolvimento *Mobile*, administração de dados, messageria, CDN e etc. Além disso, seus *data centers* que constituem 36 regiões ao redor do globo, provendo estabilidade e escalabilidade sem problemas de latência.

Figura 10: Regiões disponibilizadas pela Azure

11.2 Vantagens e diferenciais

O primeiro atributo positivo a considerar em relação aos serviços de nuvem da Azure é a confiabilidade da infraestura do provedor. Seu Contrato de Nível de Serviço (em inglês, *SLA*) estima uma porcentagem de disponibilibade de 99,95%, o que equivale a pouco mais de quatro horas anuais de tempo de *downtime*. Essa é uma conquista que até mesmo uma solução *on premises* simplesmente não consegue atingir de forma consistente. No entanto, hospedar um aplicativo ou armazenar dados na nuvem geralmente está associado a uma alta disponibilidade - certamente um dos maiores apelos de *cloud computing*. Porém a alta disponibilidade dos Serviços de Nuvem do Azure da *Microsoft* continua a impressionar, inspirando muitas vezes *CIOs* a migrar pelo menos uma parte de suas soluções mais pesadas de dados *on-line*.

Além disso, uma das maiores preocupações das empresas ao migrar para a nuvem é a segurança. A Azure segue o modelo de segurança padrão, que consiste em "detectar, avaliar, diagnosticar, estabilizar e fechar". Esse modelo, alinhado com outras políticas rígidas de segurança, rendeu à plataforma uma séria de certificações de *compliance*, a estabelecendo como líder em segurança de *IaaS*. Fornece proteção em vários níveis e serviços simples e fáceis de usar.

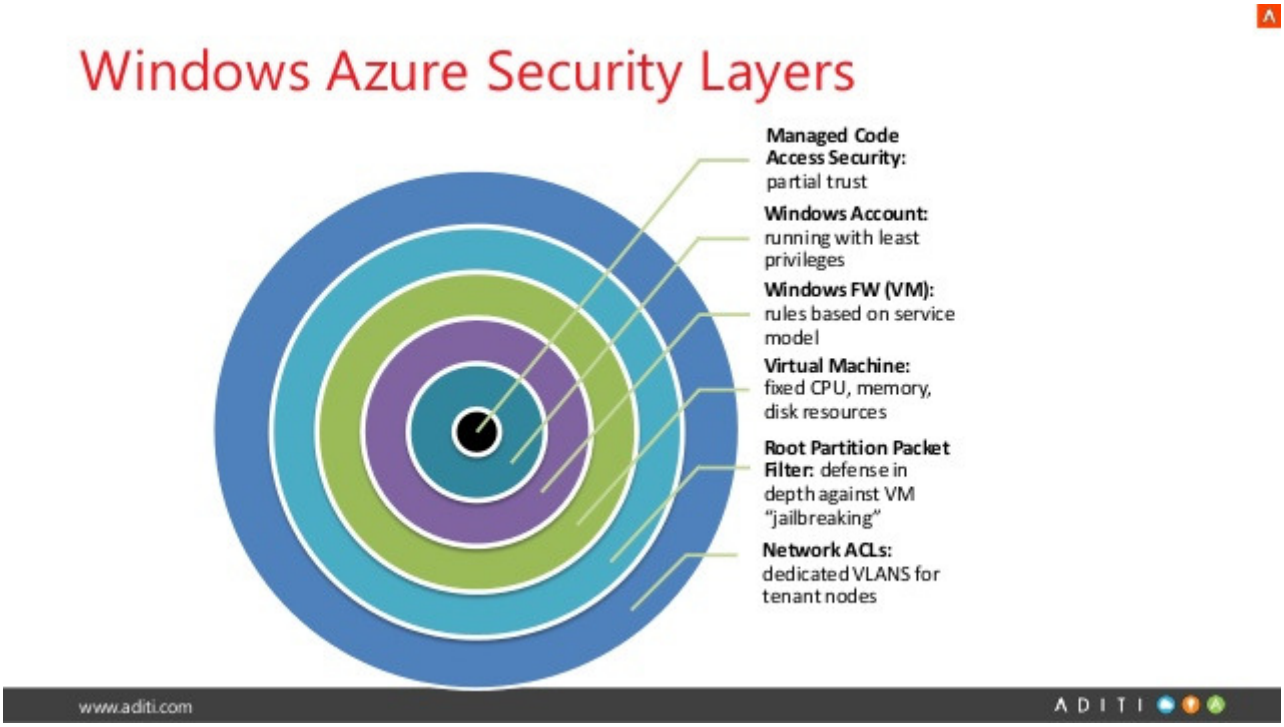

Figura 11: Camadas de Segurança da Azure

Finalmente, os serviços de nuvem da Azure também oferecem a escalabilidade necessária para o crescimento de qualquer solução, fornecendo maior poder computacional conforme necessário. Seja uma pequena empresa, ou uma empresa com a necessidade de um grande *data warehouse*, os serviços de nuvem do Azure têm armazenamento e capacidade computacional para lidar com praticamente qualquer tipo de demanda.

11.3 Desvantagens

Ao contrário das plataformas *SaaS*, nas quais o usuário final está consumindo informações (por exemplo, o *Office 365*), a Azure migra o poder computacional de dentro de sua empresa *(datacenter)* ou escritório para a nuvem. Como ocorre com a maioria dos provedores de serviços em nuvem, o Azure precisa ser gerenciado e mantido cuidadosamente, o que inclui aplicação de *patches* e o monitoramento dos servidores.

Ao contrário dos servidores locais, o Azure exige experiência para garantir que todas as partes móveis trabalhem juntas de forma eficiente. Um erro comum dos administradores de negócios que não estão totalmente envolvidos em quão bem (ou mal) seus servidores de nuvem estão operando é o de fazer o chamado *over-provisioning* dos serviços em nuvem. Embora seja um erro corriqueiro, essa falta de conhecimento dos administradores pode custar às empresas milhares de dólares por ano.

11.4 Conclusão

No geral, a Azure é uma excelente plataforma de *cloud computing*, ficando atrás de seus concorrentes apenas em alguns poucos pontos. Possui serviços robustos e bem consolidados, bastante segurança, e boa disponibilidade e escalabilidade. Além disso, torna-se quase que uma escolha obrigatória para empresas que já utilizam ou tem familiaridade com ambientes *Microsoft*, devido às inúmeras facilidades extras fornecidas (integrações e possíveis cortes de gastos com licenças).