UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

AUTOMATIZANDO A GERÊNCIA DE INFRAESTRUTURAS DE COMPUTAÇÃO EM NUVEM PARA ENSINO DE COMPUTAÇÃO DISTRIBUÍDA

TRABALHO DE GRADUAÇÃO

Cezar Augusto Contini Bernardi

Santa Maria, RS, Brasil

2016

AUTOMATIZANDO A GERÊNCIA DE INFRAESTRUTURAS DE COMPUTAÇÃO EM NUVEM PARA ENSINO DE COMPUTAÇÃO DISTRIBUÍDA

Cezar Augusto Contini Bernardi

Trabalho de Graduação apresentado ao Curso de Ciência da Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de

Bacharel em Ciência da Computação

Orientador: Prof. Dr. João Vicente Ferreira Lima

Em haver
Santa Maria, RS, Brasil

2016

AGRADECIMENTOS

À minha família por me proporcionar a chance e a liberdade de realizar meus estudos nesta instituição, e todo o suporte no caminho até aqui.

Aos meus amigos e companheiros que estiveram comigo ao longo do curso.

Ao professor Erik Hjelmas, que foi a inspiração para a realização deste projeto.

À banca e ao orientador pelos comentários construtivos que contribuíram para o melhor desenvolvimento do projeto.

RESUMO

Trabalho de Graduação
Curso de Ciência da Computação
Universidade Federal de Santa Maria

AUTOMATIZANDO A GERÊNCIA DE INFRAESTRUTURAS DE COMPUTAÇÃO EM NUVEM PARA ENSINO DE COMPUTAÇÃO DISTRIBUÍDA

AUTOR: CEZAR AUGUSTO CONTINI BERNARDI
ORIENTADOR: JOÃO VICENTE FERREIRA LIMA
Local da Defesa e Data: Santa Maria, 06 de Julho de 2016.

As ferramentas de computação na nuvem têm crescido rapidamente, gerando possibilidades de exploração destas. Mais especificamente, os serviços de IaaS possibilitam a criação de ambientes realistas de desenvolvimento e execução de algoritmos distribuídos e paralelos. Aliando isso aos desafios de se manter um cluster local, esse tipo de serviço se torna interessante para o processo de ensino-aprendizagem do tipo de algoritmos citados. Porém a configuração dessas plataformas para atingir um ambiente ideal requer carga de trabalho, e a repetição desses passos pode ser demorada e tão trabalhosa quanto. Esse projeto implementa uma infraestrutura com máquinas virtuais para Amazon Web Services, utilizando orquestração provida pela plataforma para garantir replicabilidade e facilidade de implantação. Com a ajuda de uma interface em linha de comando também possibilita modificar características das instâncias presentes para diferentes necessidades de aplicação.

Palavras-chave: Computação em Nuvem. Computação Distribuída. Orquestração. Ensino.

ABSTRACT

Bachelor Thesis
Undergraduate Program in Computer Science
Federal University of Santa Maria

MANAGEMENT AUTOMATION OF CLOUD COMPUTING INFRASTRUCTURES FOR DISTRIBUTED COMPUTING TUTORSHIP

AUTHOR: CEZAR AUGUSTO CONTINI BERNARDI
ADVISOR: JOÃO VICENTE FERREIRA LIMA
Defense Place and Date: Santa Maria, July 06th, 2016.

Cloud computing tools have been growing rapidly, creating exploration possibilities for such. More specificaly, IaaS services enable the creation of realistic development and execution environments for parallel and distributed algorithms. Coupling this to the challenges that comes with mantaining a local cluster, this kind of service becomes very interesting to the tutorship process on these algorithms. However, the configuration of such services to get to an ideal environment requires effort and time, also the repetition of the processes. This project implements an Amazon Web Services infrastructure with virtual machines, utilizing the platform orquestration tools to ensure replicability and ease of deployment. With the help of a command line interface, it allows for modifications of the instances to be deployed so that it meets distinct applications.

Keywords: Cloud Computing, Orchestration, Tutorship, Distributed Computing.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE APÊNDICES

LISTA DE ABREVIATURAS E SIGLAS

AMI	Amazon Machine Image
API	Application Programming Interface
AWS	Amazon Web Services
CLI	Command Line Interface
DNS	Domain Name Server
EC2	Elastic Compute Cloud
GPGPU	General Purpose Graphic Processing Unit
IaaS	Infraestrutura como um Serviço
JDK	Java Development Kit
JRE	Java Runtime Environment
JSON	JavaScript Object Notation
MPI	Message Passing Interface
NAT	Network Address Translation
PaaS	Plataforma como um Serviço
RMI	Remote Method Invocation
S3	Simple Storage Service
SaaS	Software como um Serviço
SSH	Secure Shell
VM	Máquinas Virtuais
VPC	Virtual Private Cloud

1 INTRODUÇÃO

Os serviços na nuvem, também conhecidos como Computação em Nuvem, têm expandido ultimamente, proporcionando diversas oportunidades de aplicação, nem sempre de fácil uso. Estes podem ser definidos como a utilização de computadores, servidores e memória disponíveis na internet. Esses serviços podem ser classificados em três principais categorias, sendo elas Infraestrutura como um Serviço (IaaS), Plataforma como um Serviço (PaaS) e Software como um Serviço (SaaS) (SOUSA et al., 2010).

As plataformas de PaaS oferecem um ambiente de desenvolvimento e teste para aplicativos que serão disponibilizados na nuvem. Com isso o desenvolvedor pode se preocupar com o desenvolvimento apenas, enquanto o provedor do serviço trata do gerenciamento, atualização e manutenção da infraestrutura utilizada pelo aplicativo desenvolvido. Já os SaaS, ao invés de oferecer uma plataforma de desenvolvimento, provê aplicativos que podem ser acessados de qualquer lugar. Com isso o usuário passa para o provedor do serviço todas as tarefas de manutenção, atualização, disponibilidade, podendo apenas fazer o uso contínuo do aplicativo. Em IaaS, a plataformas oferecem capacidade de hardware disponibilizadas através de virtualização. Aqui o usuário tem a liberdade de criar infraestruturas virtuais conforme suas necessidades, tirando a necessidade de hardware local e permitindo grande escalabilidade.

Dentre estes modelos, o IaaS garante flexibilidade para se adequar a diferentes requisitos de ensino, se tornando a ferramenta ideal para permitir maior contato com ambientes realistas por parte dos alunos. Ou também acesso a recursos computacionais de difícil acesso de outra maneira, como clusters. Porém, a utilização de tais recursos requer esforço de gerência para implantação e uso sem demais problemas. Consequentemente, este projeto visa a automatização dessa etapa, facilitando o acesso a esses recursos dentro da universidade. Diversas instituições já vêm implementando suas próprias soluções de orquestração, como Stanford University e Massachusetts Institute of Technology (SLAUGHTER, 2014). Sendo assim, este projeto faz o uso da Amazon Web Services, pois esta provê ferramentas de orquestração e gerenciamento. Além disso, possui variados serviços que podem ser acoplados, aumentando o leque de possibilidades na realização do projeto e no processo de ensino.

1.1 Objetivo

Este trabalho objetiva desenvolver um modelo de infraestrutura para Amazon Web Services focado no processo de ensino de programação paralela e distribuída, de forma que possa ser facilmente adaptado a mudanças conforme desejar o educador.

1.1.1 Objetivos Específicos

- Criar um modelo de ambiente de rede genérico e estático para AWS, com instâncias prontas para desenvolvimento.
- Criar uma ferramenta para modificar o modelo conforme a necessidade, integrando escalabilidade.

1.2 Justificativa

O ensino de computação distribuída será beneficiado significativamente, abrindo oportunidades de exploração e experimentação de MPI para programação paralela ou RMI para programação distribuída em um ambiente realista. Um ponto importante é que cada aluno teria acesso a esse ambiente para si só e poderia experimentar mais e compreender melhor o funcionamento dessas técnicas de programação.

Também facilitará o trabalho do educador, o qual terá um ambiente totalmente controlado e replicável para repassar o funcionamento das ferramentas a serem ensinadas.

1.3 Organização do texto

Este trabalho está organizado da seguinte forma: O capítulo 2 apresenta fundamentação, ferramentas e trabalhos relacionados que fazem parte do tema e da proposta de solução do trabalho. O capítulo 3 detalha a implementação do trabalho, apresentando o processo de desenvolvimento da solução e como as ferramentas apresentadas no capítulo 2 foram utilizadas. No capítulo 4 são apresentados os resultados do trabalho: Como utilizar a ferramenta e o que o ambiente implantado disponibiliza para desenvolvimento. E por fim, no capítulo 5, apresentam-se as considerações finais e conclusões do trabalho.

2 FUNDAMENTOS E REVISÃO DE LITERATURA

Este capítulo apresenta as ferramentas utilizadas na realização no projeto, começando por Computação em Nuvem e exemplos dessa na educação, similares ao objetivo primário deste projeto. Em seguida é apresentada a plataforma de IaaS da Amazon, chamada Amazon Web Services, assim como os serviços providos e aqui utilizados. E finalmente, em Trabalho Relacionados, são apresentados casos em que são utilizadas ferramentas de orquestração para tais.

Para a realização deste trabalho foi necessário escolher uma ferramenta de IaaS como plataforma de desenvolvimento, que suportasse escalabilidade e orquestração. A plataforma escolhida foi a Amazon Web Services (AMAZON WEB SERVICES, 2016), por preencher os requisitos, além de disponibilizar concessões a alunos e educadores.

2.1 Computação em Nuvem

Computação em Nuvem pode ser descrita como o uso de recursos computacionais e de memória disponíveis na internet. Existem três principais formas pelas quais esses recursos podem ser disponibilizados, sendo elas Software como um Serviço (SaaS), Plataforma como um Serviço (PaaS) e Infraestrutura como um Serviço (IaaS).

A subseção abaixo apresenta os tipos principais de computação em nuvem, mais especificamente IaaS. Em seguida são apresentados exemplos de uso da nuvem na educação.

2.1.1 Tipos de Serviço

As plataformas de PaaS oferecem um ambiente de desenvolvimento e teste para aplicativos que serão disponibilizados na nuvem. Com isso o desenvolvedor pode se preocupar com o desenvolvimento apenas, enquanto o provedor do serviço trata do gerenciamento, atualização e manutenção da infraestrutura utilizada pelo aplicativo desenvolvido. Exemplos desse serviço são Google App Engine, Cloud Foundry e OpenShift.

Já as plataformas de SaaS proveem aplicativos que podem ser acessados de qualquer lugar, aproveitando a escalabilidade da computação em nuvem para servir grande número de usuários simultâneos. Assim, o usuário passa para o provedor do serviço todas as tarefas de manutenção, atualização e disponibilidade, podendo apenas fazer o uso contínuo do aplicativo.

Exemplos desse modelo são Google Apps for Business e Salesforce.

Em IaaS, a plataformas oferecem capacidade de hardware disponibilizadas através de virtualização. Aqui o usuário tem a liberdade de criar infraestruturas virtuais conforme suas necessidades, removendo a necessidade de hardware local e permitindo escalabilidade, customização e modificações ao longo do tempo.

Essas plataformas oferecem recursos escaláveis que podem ser ajustados sob demanda, tornando IaaS ideal para projetos temporários e experimentais (WHAT IS IAAS, 2015). Mais características incluem automação de tarefas administrativas, facilitando a gerência pós-implantação.

As cobranças são feitas baseadas no uso de recursos, tipicamente por horas, semanas ou meses. Os principais provedores desse modelo atualmente são Amazon Web Services, Microsoft Azure e Google Compute Platform. Dentre essas plataformas, o AWS provê créditos renováveis anualmente para professores e alunos, além de um nível de uso gratuito no primeiro ano, enquanto o Google Compute Platform provê 300 USD em créditos para uso ao longo dos 60 primeiros dias. O Azure não proporciona uma opção semelhante às demais.

2.1.2 Trabalhos Relacionados

Essa subseção apresenta trabalhos que se relacionam de uma entre duas formas com o trabalho aqui realizado. Essas duas formas são: uso de computação em nuvem na educação e orquestração de infraestruturas. Exemplos de ambas são descritos nas subseções a seguir.

2.1.2.1 Computação em Nuvem na Educação

O uso de computação em nuvem também possibilita maior foco no ensino do que em gerenciamento de sistemas por parte dos professores (MIRCEA; ANDREESCU, 2011), aumentando a qualidade final do curso e da instituição em geral. Outro beneficio gerado pelo uso de sistemas na nuvem é a redução de custos para a instituição, reduzindo a necessidade de servidores locais (TOUT; SVERDLIK; LAWVER, 2009).

Aqui são listados casos em que serviços de IaaS são utilizados no processo de ensino.

- System Administration, Gjovik University College por Erik Hjelmas (HJELMAS, 2015) - Faz o uso de OpenStack, disponibilizando um ambiente exclusivo a cada aluno para experimentação. O ambiente provido pelo professor conta com algumas VMs de diferentes sistemas operacionais para que o aluno configure serviços como Active Directory,

servidor DNS e Puppet. O uso da nuvem nesse caso possibilita experimentação em um ambiente similar aos reais, apenas em menor escala.

- Parallel Computing, Stanford University por Alex Aiken e Kunle Olukotun (SLAUGHTER, 2014) - Utiliza diversas infraestruturas AWS para realização de tarefas em threads, STM, MapReduce (Hadoop), OpenMP e GPGPU (CUDA). Cada uma dessas infraestruturas foi construída especificamente para as diferentes tarefas que são realizadas no ensino do conteúdo.

- Introduction to Modeling and Simulation, Massachusetts Institute of Technology por Markus J. Buehler e Jeffrey C. Grossman (TECHNOLOGY, 2011a) - Dado o caráter naturalmente intensivo em computação da disciplina, impossibilitando o uso de computadores pessoais para testes, foi implementado um cluster em nuvem que acomodasse o uso por parte de todos os alunos. Porém, a manutenção de um cluster local se tornou muito expressiva, levando ao surgimento do StarCluster, uma ferramenta para criação de clusters na Amazon EC2, com AMIs personalizadas com os pacotes OpenMPI, ATLAS, NumPy/SciPy e IPython.

Nenhum destes exemplos expõe comparações entre utilização dos serviços na nuvem e recursos locais afim de validar o uso como ferramenta de análise científica de desempenho.

2.1.2.2 Orquestração de IaaS

Existem outras ferramentas para facilitar a implantação de ambientes em EC2. Cada um desses projetos têm um foco diferente dos demais no que se refere a orquestração do ambiente, e normalmente requerem o uso de mais ferramentas auxiliares para o pleno funcionamento.

- Stanford Elliot Slaughter - Baseado em um script Python para automatizar a implantação de diferentes tipos de clusters. Cada um desses tipos contém as ferramentas e ambiente preparado para um conteúdo específico dentre threads, STM, MapReduce (Hadoop), OpenMP e GPGPU (CUDA). A forma como o ambiente é implantado, e quais as ferramentas instaladas, pode ser alterada por meio de modificações no script e nos arquivos de configuração. Este projeto visa a facilitação da criação de uma infraestrutura centralizada e não sua replicação.

- StarCluster (TECHNOLOGY, 2011b) - Provê uma interface em CLI para criação, configuração, gerência e acesso a clusters na Amazon EC2. Este projeto automatiza a implantação em si, deixando de lado a orquestração provida pelo CloudFormation, utilizando AMIs personalizadas para programação com OpenMPI, ATLAS, NumPy/SciPy e IPython, individualmente.

2.2 Amazon Web Services

A plataforma de IaaS da Amazon foi escolhida por ter infraestrutura física própria e prover acesso a educadores e alunos, além de escalabilidade e ferramentas que aprimoram a replicabilidade do trabalho. Para replicabilidade, será feito o uso do Amazon CloudFormation (AMAZON CLOUDFORMATION, 2016), o qual recebe um arquivo JSON (JavaScript Object Notation) descrevendo a pilha a ser inicializada e a implanta. A pilha a ser implantada utilizará os serviços de Elastic Compute Cloud (EC2)(AMAZON ELASTIC COMPUTE CLOUD, 2016) e Virtual Private Cloud(AMAZON VIRTUAL PRIVATE CLOUD, 2016) para melhor organização e acesso. Esses serviços são naturalmente integrados, possibilitando a elaboração de infraestruturas adequadas a necessidade com grande facilidade. Conta com diversas regiões pelo mundo onde sua infraestrutura está instalada (Figura 2.1), aumentando a área de acesso com baixa taxa de atraso.

Assim como a maioria dos serviços disponíveis para nuvem, o AWS tem diferentes valores associados aos seus recursos, incluindo um nível de uso gratuito, com limitações (EC2 INSTANCE PRICING, 2016). Os serviços disponíveis, classificados por categoria estão representados na Figura 2.2. As subseções abaixo apresentam os serviços utilizados no projeto e suas funções.

2.2.1 Amazon Elastic Compute Cloud - EC2

Amazon Elastic Compute Cloud (EC2) faz parte da solução de IaaS da Amazon, provendo recursos de computação em nuvem. O serviço de EC2 conta com diversos tipos de instâncias que melhor se adequam a diferentes propósitos dentro da computação. Esses tipos são:

- Uso Geral
  - T2 - São instâncias com menor custo. Tem capacidade de intermitência, contando

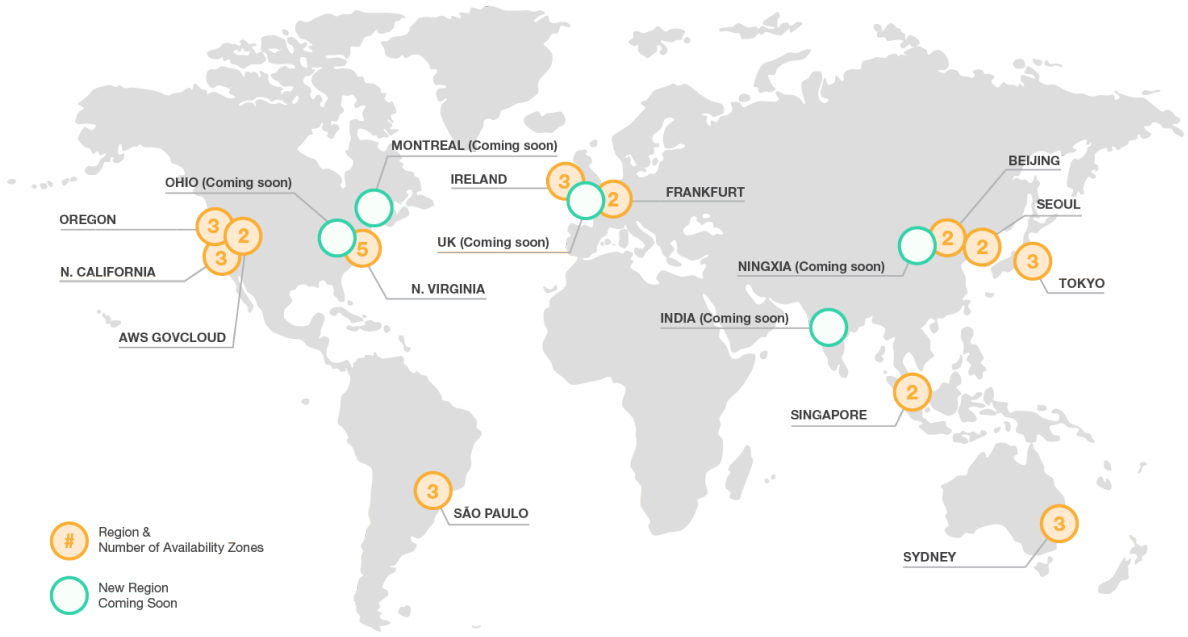

Figura 2.1: *Infraestrutura Geral da Amazon Web Services (INFRAESTRUTURA GLOBAL, 2016)*

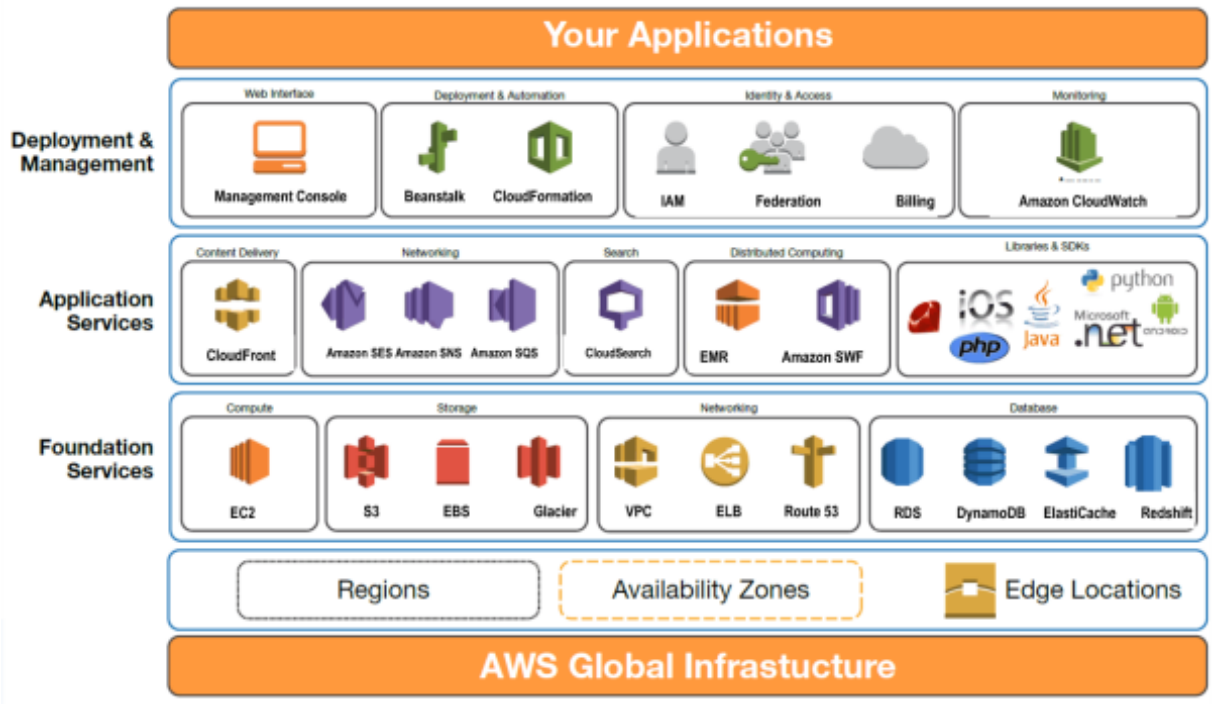

Figura 2.2: *Infraestrutura de serviços AWS (TOWNSEND, 2013)*

com processadores Intel Xeon com até 3.3 GHz em frequência de turbo, com 1 ou 2 vCPUs com 0.5 a 8 GB de RAM. Seus principais casos de uso são ambientes de desenvolvimento, servidores de compilação, repositórios de código, sites e aplicações web de baixo tráfego, micro serviços, experimentos iniciais de produtos e pequenos bancos de dados. Utiliza Amazon EBS (Elastic Block Store) como dispositivo de armazenamento.

  - M4 - Tem hardware mais robusto que as instâncias T2, utilizando processadores Intel Xeon® E5-2676 v3 2,4 GHz (arquitetura Haswell), com 2 a 40 vCPUs e 8 a 160 GB de RAM. Seu armazenamento em EBS é otimizado, com taxas de transferência de 450 a 4000 Mbps.
  - M3 - Contam com processadores Intel Xeon E5-2670 v2 (Ivy Bridge) ou Xeon E5-2670 (Sandy Bridge) 2.6 GHz, com 1 a 8 vCPUs, 3.75 a 30 GB de RAM e armazenamento em SSD, seu diferencial das demais instâncias dessa categoria. Seus casos de uso mais indicados são bancos de dados de pequeno e médio porte, tarefas de processamento de dados que exigem memória adicional, grupos de armazenamento em cache e servidores de backend para SAP, Microsoft SharePoint, computação em cluster e outras aplicações empresariais.

- Otimizadas para computação

  - C4 - Utilizam processadores Intel Xeon E5-2666 v3 (Haswell) de alta frequência otimizados especificamente para o EC2, com 2 a 36 vCPUs e 3.75 a 60 GB de RAM. O armazenamento EBS é otimizado, com taxas de transferência entre 500 e 4000 Mbps. Esta instância, assim como a C3, tem suporte aprimorado a rede, aumentando a quantidade de pacotes por segundo que podem ser enviados e recebidos.
  - C3 - Com processadores Intel Xeon E5-2680 v2 (Ivy Bridge) de 2 a 32 vCPUs, 3,75 a 60 GB de RAM e armazenamento em SSD, seus casos de uso principais são frotas de frontend de alto desempenho, servidores da web, processamento em lotes, dados analíticos distribuídos, aplicativos científicos e de engenharia de alto desempenho, veiculação de anúncios, jogos MMO e codificação de vídeo.

- Otimizadas para memória

  - R3 - O foco deste tipo de instância é a memória RAM, contando com valores de

15.25 a 244 GB, tem também armazenamento em SSD, tornando-a ideais para bancos de dados de alto desempenho, caches de memória distribuídos, análises em memória, montagem e análise de genomas, implementações maiores de SAP, Microsoft SharePoint e outros aplicativos empresariais.

- GPU
  - G2 - Conta com 1 ou 4 GPUs NVidia, com 1.536 núcleos CUDA e 4 GB de VRAM. Seu propósito é streaming de aplicações 3D, aprendizagem de máquina, codificação de vídeo e outras cargas de trabalho de gráficos ou computação de GPU no lado do servidor. Para isso conta com processador Intel Xeon E5-2670 (Sandy Bridge) com 8 ou 32 vCPUs, 15 ou 60 GB de RAM e armazenamento em SSD.
- Otimizadas para armazenamento
  - I2 - Com foco em E/S, possui de 1 a 8 SSDs com 800 GB cada, processadores Intel Xeon E5-2670 v2 (Ivy Bridge) com 4 a 32 vCPUs e 30.5 a 244 GB de RAM. O suporte a rede também é aprimorado. Seus casos de uso incluem bancos de dados NoSQL, como o Cassandra e o MongoDB, bancos de dados transacionais escaláveis, armazém de dados, Hadoop e sistemas de arquivo em cluster.
  - D2 - Instâncias de armazenamento denso, contam com processadores Intel Xeon E5-2676v3 (Haswell) com 4 a 36 vCPUs, 30.5 a 244 GB de RAM e de 3 a 20 HDDs de 2 TB. Essas instâncias são aplicáveis a armazéns de dados para processamento paralelo massivo (MPP), computação distribuída de MapReduce e Hadoop, sistemas de arquivo distribuídos, sistemas de arquivos de rede e aplicações de processamento de logs ou dados.

Cada tipo e sub-tipo de instância tem um valor de utilização por hora diferente, crescendo consideravelmente ao se utilizar recursos melhores.

Além dos tipos de instância, estão disponíveis, na forma de AMIs (AMAZON MACHINE IMAGES, 2016), diversos tipos de imagens de máquinas virtuais, baseados em diferentes sistemas operacionais. Cada uma das AMIs têm características que as tornam melhores para determinadas tarefas computacionais, geralmente diferenciadas por recursos pré-instalados e configurados.

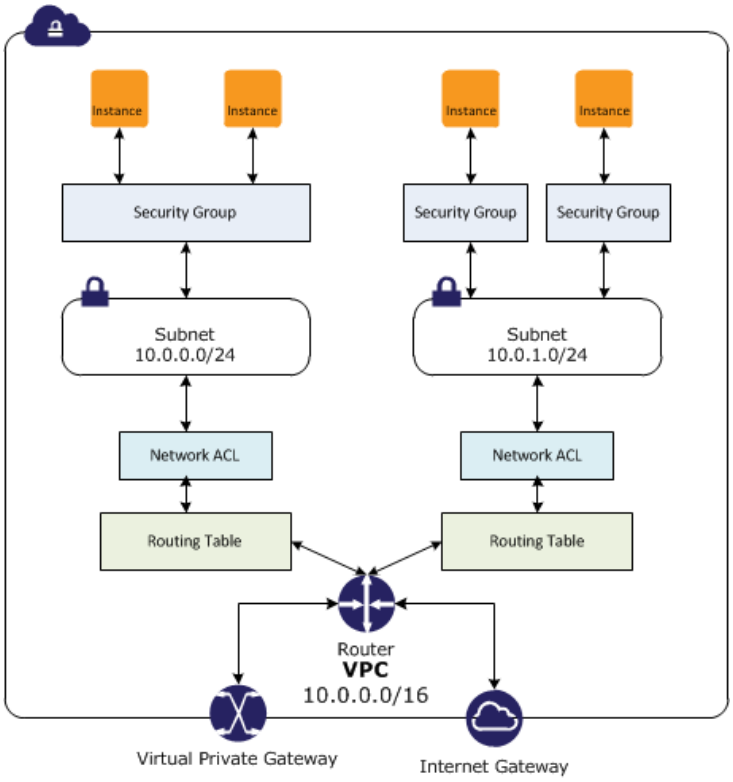

Figura 2.3: *Modelo de VPC com sub-redes, grupos de segurança, ACL, tabelas de roteamento, internet gateway e VPG (TONELLI, 2013)*

2.2.2 Amazon Virtual Private Cloud - VPC

O VPC é um recurso do AWS feito para possibilitar a divisão de outros recursos, principalmente EC2, em sub-redes. Com isso, é possível definir um ponto único de acesso aos demais recursos, aumentando a organização, escalabilidade e segurança da infraestrutura.

O VPC permite a personalização e divisão de endereços IP, levando a possibilidade de separação ou agrupamento de serviços. Também permite a comunicação com serviços em diferentes VPCs.

Também suporta a implantação de grupos de segurança, tabelas de roteamento e listas de controle de acesso, possibilitando a formação de uma infraestrutura robusta de rede, conforme Figura 2.3.

2.2.3 Amazon CloudFormation

Oferece um modo simples de implantar e gerenciar um grupo de recursos do AWS, de forma previsível e replicável. O uso dessa ferramenta pode ser feita de duas formas principais, sendo elas:

- Declaração estruturada em JSON - Através de um arquivo de texto estruturado é possível descrever toda a infraestrutura a ser criada, com parâmetros variáveis ou estáticos para cada recurso da pilha.

- CloudFormation Designer - Uma interface gráfica que disponibiliza o drag-and-drop dos recursos afim de facilitar ainda mais a criação e visualização de uma pilha de recursos AWS.

Esse recurso inclui um grupo de *helper scripts* (cfn-init, cfn-signal, cfn-get-metadata e cfn-hup) que são baseados no cloud-init (DEPLOYING APPLICATIONS ON AMAZON EC2 WITH AWS CLOUDFORMATION, 2016). Esses scripts podem ser chamados de dentro do modelo CloudFormation para instalar, configurar e atualizar aplicações nas instâncias EC2.

2.2.4 Amazon Simple Storage Service - S3

É um serviço de armazenamento da Amazon, baseado em buckets. Oferece uma interface web service para armazenar e recuperar dados. Para isso, permissões de acesso são necessárias. Estas podem ser públicas, para determinados serviços ou usuários. Esse serviço está bem acoplado aos demais serviços da Amazon, disponibilizando modelos de permissão que abrangem VPCs e tabelas de roteamento. Isso torna a configuração de acesso ao serviço bem organizada em relação a infraestrutura.

2.2.5 Amazon Web Services Educate

Existem hoje duas maneiras de se conseguir créditos para serem utilizados em recursos do AWS, sendo uma para professores e outra para alunos (AMAZON WEB SERVICES EDUCATE, 2016). Há ainda a divisão entre professores e alunos de instituições associadas ou não ao programa da Amazon. Dentre outros benefícios como cursos e conteúdo para ensino de AWS, professores de instituições associadas recebem 200 USD em créditos, e de instituições não associadas, 75 USD. Já os alunos de instituições associadas ou não recebem 100 USD e 35 USD em créditos, respectivamente. Todas as opções de créditos são renováveis anualmente no atual modelo.

2.3 Sumário

Este capítulo descreveu as ferramentas e recursos necessários para a realização deste projeto. Ao final do desenvolvimento, estará disponível uma ferramenta que abrirá portas para exploração de possibilidades no ensino de algoritmos paralelos e distribuídos através de uma infraestrutura de rede e computação escalável em AWS.

3 DESENVOLVIMENTO

Este capítulo apresenta as atividades realizadas com o objetivo de desenvolver uma infraestrutura customizável para Amazon Web Services, focando no ensino de computação paralela e distribuída. Serão apresentados tópicos de organização da rede e integração de recursos AWS, assim como desafios de automatização encontrados. Em seguida serão apresentadas as opções de customização dessa infraestrutura.

A metodologia do projeto concentrou-se em construir a infraestrutura começando dos recursos mais gerais e indo em direção aos mais específicos, devido a dependências para correto funcionamento. De forma geral, o desenvolvimento seguiu esta ordem:

1. Planejamento de Implementação
2. Ambiente de Rede
   (a) Sub-redes
   (b) NAT (Network Address Translation)
   (c) Tabelas de Roteamento
3. Instâncias
   (a) Instalação e configuração de bibliotecas e pacotes
4. Nomeação dos Hosts
   (a) Configuração acesso SSH a demais hosts
5. Acesso ao S3 (Simple Storage Service)
6. Automação de Configurações da Infraestrutura

3.1 Planejamento de Implementação

Esta seção apresenta os requisitos para elaboração da infraestrutura focada nas disciplinas de Computação Distribuída e Programação Paralela, seguido dos casos de uso da ferramenta final.

3.1.1 Levantamento de Requisitos

Com o auxílio dos professores responsáveis foi realizado um levantamento de requisitos para implementação no projeto de orquestração. As disciplinas abrangidas diretamente são Programação Paralela e Sistemas Distribuídos, porém, o objetivo do projeto é que esse modelo possa ser aproveitado para ensino de demais disciplinas.

Em Programação Paralela, as bibliotecas requisitadas foram MPI (Message Passing Interface) e OpenMPC. Para instalação destas, os pacotes necessários, disponíveis nos repositórios padrões, são openmpi, openmpi-devel, gcc e gcc-c++.

Na disciplina de Sistemas Distribuídos, faz-se o uso de RMI (Remote Method Invocation), biblioteca em Java. Para fazer o uso desta ferramenta os pacotes necessários são java e java-devel.

3.1.2 Casos de Uso

Os casos de uso podem ser divididos em dois grupos, sendo eles: alunos e professores. Esta seção apresenta suas funcionalidades.

3.1.3 Professores

Utilizando o sistema de automação implementado, cada professor tem a liberdade de criar infraestruturas com ferramentas específicas para cada tarefa a ser repassada, ou uma infraestrutura única contendo todas as ferramentas necessárias ao andamento do ensino.

3.1.4 Alunos

Utiliza o console AWS para implantação da infraestrutura provida pelo seu professor. Através do uso de SSH, o usuário aluno acessa a infraestrutura e se beneficia da orquestração, acessando diretamente um ambiente pronto para desenvolvimento. Um manual de utilização é apresentado no Apêndice B.

3.2 Ambiente de rede

A metodologia de desenvolvimento do projeto foi incremental, primeiramente focando em criar um ambiente base onde as instâncias serão posicionadas. Esse ambiente foi feito

em volta de um VPC. Primeiramente, estabeleceu-se uma rede para endereços privados sob a qual todas as instâncias estarão alocadas. Decidiu-se pelo uso da rede 192.168.0.0/24 por ser um espaço de rede comumente utilizado para o propósito de redes privadas, além de fornecer endereços suficientes para a andamento do projeto.

Esta rede foi subdividida em duas, sendo as sub-redes 192.168.0.0/28 para uso das instâncias de computação e 192.168.240.0/28 para a instância com acesso externo. A instância colocada sob a sub-rede pública além de funcionar como *bastion* de acesso a rede privada com as instâncias de computação, também funciona como NAT (Networking Address Translation) para citadas instâncias. Tal medida foi tomada para facilitar o controle sobre endereçamento e organização dentro do VPC.

Como boa prática de segurança, foram associados grupos de segurança as sub-redes, tornando disponível o acesso a redes externas somente pela instância NAT, também apenas permitindo o acesso SSH via esta e demais nodos da sub-rede. Além dos grupos de segurança, tabelas de roteamento foram implantadas para direcionamento do tráfego de rede aos respectivos gateways. Do ponto de vista da subrede pública, esse gateway é um recurso especial da Amazon que permite o acesso a internet, chamado Internet Gateway, e do ponto de vista da sub-rede privada o gateway é a própria instância NAT. Uma visão geral da infraestrutura é mostrada na Figura 3.1.

3.3 Instâncias

Para realização do projeto foram utilizadas instâncias do tipo *t2.micro*, as quais se encaixam no plano de uso livre da AWS. Esse tipo pode ser alterado no momento da implantação do modelo, com a ressalva de que os servidores AWS da América do Sul não suportam o tipo *g2*, com GPUs. Porém, essas instâncias podem ser utilizadas se a região escolhida for outra, como *us-east*.

Para configuração de arquivos e instalação de pacotes necessários foram utilizados as ferramentas disponibilizadas pelo recurso Init do CloudFormation. Dentre as opções disponíveis, foram utilizadas as seguintes:

- Packages - Provê suporte aos gerenciadores de instalação apt, msi, python, rpm, rubygems e yum.
- Files - Suporte para criação de arquivos e diretórios, assim como seus conteúdos.

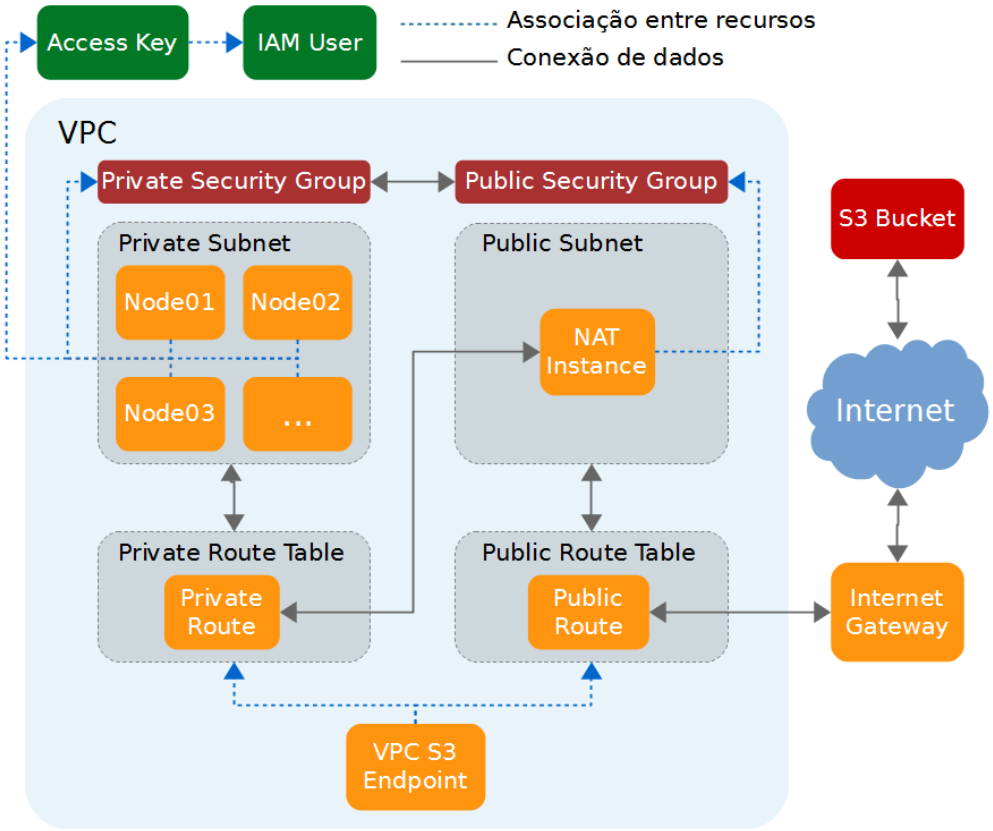

Figura 3.1: *Representação simplificada da pilha de recursos*

- Commands - Disponibiliza a execução de comandos *bash*, no caso de instâncias Linux.

Essas opção são sempre executadas pelo CloudFormation na ordem descrita, evitando a edição de pacotes ou arquivos ainda inexistentes.

3.3.1 Instalação de Pacotes

A imagem utilizada nas VMs já possui repositórios yum configurados, disponibilizando a instalação dos pacotes openmpi, openmpi-devel, java-devel, ant, gcc e gcc-c++. A listagem de pacotes para instalação foi feita segundo a Figura 3.2.

3.3.1.1 OpenMPI

A configuração desse pacote começa com a exportação de variáveis de ambiente, indicando os *wrappers* de compilação e execução. Para isso foi utilizado o recurso de execução de comandos do Init, conforme Figura 3.3.

Para sincronização dos executáveis compilados, está disponível nas máquinas virtuais a

```
"AWS::CloudFormation::Init" : {
  "setup" : {
    "packages" : {
      "yum" : {
        "openmpi"          : [],
        "openmpi-devel"        : [],
        "gcc"              : [],
        "gcc-c++"          : [],
        "java-1.8.0-openjdk-devel"   : [],
        "ant"              : []
      }
    }
  }
}
```

Figura 3.2: Listagem de pacotes a serem instalados pelo Init

```
"AWS::CloudFormation::Init" : {
  "configure" : {
    "commands" : {
      "exportOpenMPIBin" : {
        "command" : "echo \"export PATH=/usr/lib64/openmpi/bin:$PATH\" >> /
    home/ec2-user/.bashrc"
      },
      "exportOpenMPILib" : {
        "command" : "echo \"export LD_LIBRARY_PATH=/usr/lib64/openmpi/lib:
    $LD_LIBRARY_PATH\" >> /home/ec2-user/.bashrc"
      }
    }
  }
}
```

Figura 3.3: Exportação do PATH do OpenMPI

```
"exportJavaHome" : {
  "command" : "echo \"export JAVA_HOME=/usr/lib/jvm/java-openjdk/\" >> /
    home/ec2-user/.bashrc"
}
```

Figura 3.4: Exportação do PATH do JDK

ferramenta rsync ou scp, conforme preferir o usuário.

3.3.1.2 Java RMI

Para habilitar o ambiente para a compilação de códigos em Java foi instalado o pacote java-devel, que provê o Java Development Kit (JDK). Também foi instalada a ferramenta ant, uma ferramenta de compilação Java, caso seja necessária.

Através do Init foi feita a exportação da variável de ambiente JAVA_HOME para indicar o caminho do JDK ao invés do JRE previamente indicado, conforme Figura 3.4.

3.3.2 Acesso SSH

O primeiro desafio encontrado nesta etapa foi a simplificação da configuração do acesso SSH entre as instâncias, necessário a execução de algoritmos distribuídos com MPI. Por padrão, instâncias EC2 já têm a chave pública referente ao *key pair* configurado previamente a aplicação da pilha, porém, a chave privada está apenas disponível em arquivo PPK e é utilizada da máquina do usuário para acesso as instância pública. Isto não permite que as instâncias tenham acesso entre si via SSH, então foi necessário requerer a chave privada como *string* no momento de implantação, salvá-las nas VMs e reorganizar a estrutura do arquivo com um script python (Figura 3.5a), pois as quebras de linha são perdidas e são necessárias para a correta interpretação da chave.

Há ainda um script (Figura 3.5b) que escaneia e armazena os fingerprints de cada um dos hosts presentes na rede. Isso é feito para que não seja necessário dar permissão de acesso manualmente aos demais hosts em cada um deles, como é necessário para execuções de OpenMPI.

Outra modificação aplicada foi a edição do arquivo `/etc/hosts` para adicionar *alias* aos endereços das instâncias da sub-rede, conforme Figura 3.6.

```
import re
input = open("/home/ec2-user/.ssh/id_rsa.in").read()
bgnEnd = re.findall('-----[A-Z\s]+-----',input)
content = re.findall('-\s(.+?)\s-',input)
lines = content[0].split(" ")
outputFile = open("/home/ec2-user/.ssh/id_rsa", "w")
outputFile.write(bgnEnd[0] + '\n')
[outputFile.write(line + '\n') for line in lines]
outputFile.write(bgnEnd[1])
outputFile.close()
```

(a) sshParser.py

```
import re
from subprocess import check_output
input = open("/etc/hosts").read()
hosts = re.findall('\s([a-z0-9]+[0-9])\n', input)
ips = re.findall("\n(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})\s", input)
output = list()
[output.append(check_output(["ssh-keyscan", "-H", ip])) for ip in ips]
[output.append(check_output(["ssh-keyscan", "-H", host])) for host in hosts
    ]
outputFile = open("/home/ec2-user/.ssh/known_hosts", "w")
[outputFile.write(line + '\n') for line in output]
outputFile.close()
```

(b) sshHostScan.py

Figura 3.5: Scripts de automatização

```
"AWS::CloudFormation::Init" : {
  "config" : {
    "files" : {
      "/etc/hosts" : {
        "content" : {
          "Fn::Join" : ["\n", [
          "127.0.0.1   localhost   localhost",
          {"Fn::FindInMap":["IpAddressConfig", "Node01", "IP" ]},"  node01
    node01",
          {"Fn::FindInMap":["IpAddressConfig", "Node02", "IP" ]},"  node02
    node02",
          {"Fn::FindInMap":["IpAddressConfig", "Node03", "IP" ]},"  node03
    node03",
          {"Fn::FindInMap":["IpAddressConfig", "Node04", "IP" ]},"  node04
    node04",
          {"Fn::FindInMap":["IpAddressConfig", "Node05", "IP" ]},"  node05
    node05"]]
        }
      }
    }
  }
}
```

Figura 3.6: Edição do arquivo /etc/hosts para 5 nodos via CloudFormation

```
"VPCEndpointS3" : {
  "Type" : "AWS::EC2::VPCEndpoint",
  "Properties" : {
    "ServiceName"    : "com.amazonaws.sa-east-1.s3",
    "VpcId"          : { "Ref": "VPC" },
    "RouteTableIds" : [ { "Ref": "PrivateRouteTable"  }, { "Ref": "
   PublicRouteTable"  } ]
  }
}
```

(a) VPC Endpoint

```
"IAMUser" : {
  "Type" : "AWS::IAM::User",
  "Properties" : {
    "Path" : "/"
  }
},
"AcessKey" : {
  "Type" : "AWS::IAM::AccessKey",
  "Properties" : {
    "UserName" : { "Ref" : "IAMUser" }
  }
}
```

(b) Usuário IAM e Chave de Acesso

```
"/home/ec2-user/.aws/credentials" : {
  "content" : {
    "Fn::Join" : ["", [
      "[default]\n",
      "aws_access_key_id=", { "Ref" : "AcessKey" }, "\n",
      "aws_secret_access_key=", { "Fn::GetAtt" : [ "AcessKey", "
   SecretAccessKey" ] }
    ]]
  }
}
```

(c) AWS Credentials

Figura 3.7: Configuração de acesso ao S3

3.3.3 Upload de Arquivos

Com o intuito de integrar essa parte do processo de desenvolvimento, foi criado um VPC Endpoint (Figura 3.7a) que cria um ponto de acesso ao S3. Para usufruir dos serviços, porém, foi necessária a criação de um usuário IAM e chaves de acesso ao serviço (Figura 3.7b), assim como a configuração (Figura 3.7c) destas nas instâncias.

O acesso pode ser feito via API da Amazon, pré instalada na AMI, ou via GET/wget.

A automatização da criação do bucket em si enfrenta o porém de que estes devem ter nomes únicos dentre todos os buckets do S3.

3.4 Automação de Configurações da Infraestrutura

Através do uso de scripts em python, criou-se uma interface em linha de comando para fazer modificações em partes específicas da infraestrutura. As possíveis modificações são: quantidade de nodos, pacotes a serem automaticamente instalados em cada nodo e scripts a serem executados no instanciamento. O objetivo deste modelo de automação é gerar uma infraestrutura que pode ser implantada e utilizada individualmente por qualquer pessoa.

A quantidade de nodos é limitada a 124, isso se dá por conta das configurações de sub-rede. Como o AWS limita a rede de cada VPC em 24 bits, e duas sub-redes (pública e privada) são necessárias na infraestrutura criada, a maior sub-rede possível tem 25 bits. Com isso, totalizam-se 128 endereços, sendo 1 de broadcast, 1 de rede e os 2 primeiros endereços válidos reservados pela Amazon.

A instalação automática de pacotes está limitada aos repositórios disponíveis na instância. Contudo, isto pode ser contornado fazendo-se uso dos scripts customizados que a automação permite para instalar novos repositórios e posteriormente, pacotes.

As configurações podem ser modificadas utilizando-se o assistente de configuração ou através de argumentos. A utilização dessas opções está descrita no Apêndice A.

3.5 Sumário

Este capítulo apresentou os passos e desafios encontrados no desenvolvimento de uma infraestrutura customizável de IaaS em AWS, assim como a automação das modificações possíveis. Também apresentou os casos de uso para a ferramenta de edição e da infraestrutura em si dentro do contexto de ensino.

4 RESULTADOS

Este projeto possibilita a customização e criação de uma pilha de recursos para AWS, com seu modelo padrão pronto para desenvolvimento paralelo e distribuído com OpenMP, MPI e RMI. Os recursos de desenvolvimento instalados podem ser modificados conforme a necessidade de cada professor. Com o uso de scripts é possível instalar e configurar recursos que não estão disponíveis por padrão, aumentando a flexibilidade da infraestrutura.

O acesso a essa pilha de recursos se dá por SSH, primeiramente acessando a instância bastion, que tem endereço IP público e a partir dela, também por SSH, pode-se acessar às demais instâncias destinadas ao desenvolvimento, utilizando o prefixo *node* acompanhado por número identificador de dois dígitos, por exemplo *ssh node01*.

Uma vez acessada uma instância de computação, pode-se acessar às demais da mesma forma descrita acima. O ambiente também já está pronto para compilação e execução de códigos MPI, RMI e OpenMP, sem interface gráfica. A transferência de dados da máquina do usuário para a instância pode ser feita através do upload de arquivos para um bucket da Amazon S3, o qual precisa ser criado por cada usuário, e então fazendo o download na instância utilizando *wget* ou comandos do Amazon CLI.

Os comandos importantes para acesso ao S3 são *aws s3 cp s3://nome-do-bucket/arquivo-remoto arquivo-local*, para download de um arquivo único, ou *aws s3 sync s3://nome-do-bucket pasta-local* para sincronizar todo o bucket para um diretório local. O processo de upload de arquivos da instância para o S3 pode ser feito invertendo a posição dos caminhos locais e remotos em ambos os comandos. Todas as configurações e permissões de acesso para realizar esse processo são automaticamente realizadas ao se lançar a pilha de recursos.

Uma versão preliminar da infraestrutura foi testada na disciplina de Programação Paralela, com ajuda da professora Andrea Charão. O ponto mais crítico levantado nos testes se refere a necessidade de cartão de crédito para criação da conta no AWS, o que é uma característica inerente aos serviços de IaaS públicos. Além disso, os testes de desempenho dos códigos em paralelo mostrou maiores variações comparado a execuções locais, o que é esperado de ambientes altamente virtualizados. Tirando-se esses aspectos específicos ou imutáveis, as experiências de implantação e utilização foram positivas por parte dos alunos e professora.

A ferramenta de orquestração aqui criada se diferencia das demais ferramentas citadas na seção 2.1.2.2 pois o foco é na replicabilidade e flexibilidade. A ferramenta desenvolvida para

Stanford foca em um uma interface intermediária entre o usuário e a implantação, facilitando o processo de criação de uma infraestrutura única e personalizada pelo seu utilizador. De forma semelhante, a ferramenta desenvolvida pelo MIT foca na orquestração da implantação de uma única infraestrutura centralizada, contando com AMIs próprias, dando mais confiabilidade e retirando flexibilidade.

5 CONCLUSÃO

Esse projeto teve como propósito criar um ambiente facilmente implantável, escalável e replicável, que simule ambiente reais de desenvolvimento, para uso no processo de ensino de conteúdos de computação paralela e distribuída.

A etapa de desenvolvimento se deu através do planejamento da infraestrutura de rede de forma escalável e simples. Assim como pelo levantamento de quais são as ferramentas necessárias dentro do ambiente para execução dos códigos alvo para então realizar a instalação automatizada das bibliotecas e aplicativos. Também foi feito levantamento de que opções seriam mais úteis na customização do modelo.

Ao fim desta etapa, foi atingido o objetivo de criação de um ambiente replicável, facilmente implantável e customizável. As customizações possíveis são a quantidade de nodos de computação, os pacotes automaticamente instalados nesses e execução de scripts. Escrever scripts para execução no momento de instanciamento é uma tarefa mais trabalhosa, mas essa é uma opção bastante flexível que gera mais possibilidades de customização da instâncias.

Como trabalhos futuros, mais opções podem ser adicionadas as configurações da infraestrutura, ou modificações nesta para possíveis diferentes casos de ensino.

REFERÊNCIAS

AMAZON CloudFormation. Disponível em: <https://aws.amazon.com/pt/cloudformation/>, acessado em Março de 2016.

AMAZON Elastic Compute Cloud. Disponível em: <https://aws.amazon.com/pt/ec2/instance-types/>, acessado em Abril de 2016.

AMAZON Machine Images. Disponível em: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html>, acessado em Abril de 2016.

AMAZON Virtual Private Cloud. Disponível em: <https://aws.amazon.com/vpc/>, acessado em Abril de 2016.

AMAZON Web Services Educate. Disponível em: <https://aws.amazon.com/pt/education/awseducate/>, acessado em Março de 2016.

AMAZON Web Services. Disponível em: <https://aws.amazon.com/pt/>, acessado em Março de 2016.

DEPLOYING Applications on Amazon EC2 with AWS CloudFormation. Disponível em: <http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/deploying.applications.html#deployment-walkthrough-basic-server>, acessado em Abril de 2016.

EC2 Instance Pricing. Disponível em: <https://aws.amazon.com/pt/ec2/pricing/>, acessado em Abril de 2016.

HJELMAS, E. **Lecture notes in System Administration**. [S.l.]: Gjovik University College, 2015.

INFRAESTRUTURA Global. Disponível em: <https://aws.amazon.com/pt/about-aws/global-infrastructure/>, acessado em Março de 2016.

MIRCEA, M.; ANDREESCU, A. I. Using cloud computing in higher education: a strategy to improve agility in the current financial crisis. **Communications of the IBIMA**, [S.l.], 2011.

SLAUGHTER, E. **CS-149 Setup**. Disponível em: <https://bitbucket.org/elliottslaughter/cs149-setup>, acessado em Março de 2016.

SOUSA, F. R. et al. Gerenciamento de dados em nuvem: conceitos, sistemas e desafios. **Topicos em sistemas colaborativos, interativos, multimidia, web e bancos de dados, Sociedade Brasileira de Computacao**, [S.l.], p.105, 2010.

TECHNOLOGY, M. I. of. **StarCluster at MIT**. Disponível em: <http://star.mit.edu/cluster/about.html>, acessado em Abril de 2016.

TECHNOLOGY, M. I. of. **StarCluster**. Disponível em: <http://star.mit.edu/cluster/docs/latest/overview.html>, acessado em Abril de 2016.

TONELLI, R. **AWS**: introduction to virtual private cloud security. Disponível em: <http://blog.celingest.com/en/2013/04/19/aws-virtual-private-cloud-vpc-security/>, acessado em Maio de 2016.

TOUT, S.; SVERDLIK, W.; LAWVER, G. Cloud computing and its security in higher education. **Proc ISECON, v26 (Washington DC)**, [S.l.], v.2314, 2009.

TOWNSEND, J. **Getting Started With Amazon Web Services**. Disponível em: <http://blog.clearpathsg.com/blog/bid/304931/Getting-Started-with-Amazon-Web-Services-AWS-Part-1/>, acessado em Maio de 2016.

WHAT is IaaS. Disponível em: <http://searchcloudcomputing.techtarget.com/definition/Infrastructure-as-a-Service-IaaS>, acessado em Junho de 2016.

APÊNDICES

APÊNDICE A – Manual de Configuração

Repositório

https://github.com/hizeph/AWS-CloudFormation-Editor

Utilização

Todas as ações são realizadas através da execução do arquivo *setup.py* e requerem Python 3+ com os pacotes *argparse* e *json* (ambos podem ser instalados com *pip* ou *easy_install*).

Executando *setup.py* sem argumentos leva a um assistente com opções explícitas de ações possíveis.

Os argumentos opcionais implementados para o *setup* são:

-h, --help
: Mostra os argumentos disponíveis.

-o, --output OUTPUT
: Lê o arquivo de configuração, gera e salva a infraestrutura com o nome dado.

-f, --file FILE
: Indica um arquivo de configuração diferente do padrão.

-n, --nodes NODES
: Indica o número de nodos da infraestrutura, no máximo 124. Esta opção sobrescreve o arquivo de configuração.

O arquivo de configuração segue o modelo a seguir e está na pasta raiz do projeto, com o nome *config*.

```
{
	"n_nodes" : 3,
	"packages" :
	[
		"gcc",
		"gcc-c++"
	],
	"custom_scripts" :
	[
		{ "name" : "custom.py", "interpreter" : "python" },
		{ "name" : "custom.sh", "interpreter" : "bash" }
	]
}
```

APÊNDICE B – Manual de Implantantação e Uso

Requisitos

- Conta na Amazon Web Services - https://aws.amazon.com/pt/
    - Criar Key Pair
- Arquivo com a definição da infraestrutura
- Cliente SSH
    - Windows:
        - PuTTY - https://the.earth.li/~sgtatham/putty/latest/x86/putty.exe
        - Conversor de chaves *.pem* para *.ppk*
            - PuTTYgen - ttps://the.earth.li/~sgtatham/putty/latest/x86/puttygen.exe
                - Conversions -> Import key -> Save private key (sem passphrase)
    - Linux:
        - OpenSSH - Disponível em repositórios

Criando Key Pair

1. Login no console AWS (https://aws.amazon.com/pt/).
2. Selecionar região desejada no canto superior direito.
3. Selecionar EC2 no painel inicial do console AWS.
4. No menu à esquerda, sob Rede e Segurança (Network & Security) selecionar Key Pair.
5. Criar novo Key Pair - por motivos de praticidade futura recomendo nomea-lo ‘aws-sp’, sem aspas.
6. (PuTTY) Converter a chave *.pem* para *.ppk*.
7. Importante: Manter arquivo *.pem*.

Passos para Implantação da Infraestrutura

1. Login no console AWS (https://aws.amazon.com/pt/).
2. Selecionar região desejada no canto superior direito.
3. Acessar o CloudFormation através do menu Serviços (Services)-> Ferramentas de Gerência (Management Tools) -> CloudFormation.
4. Criar nova Pilha (Create new Stack).
5. Selecionar arquivo JSON com a definição da infraestrutura, ou S3 template URL, dependendo do que tiver sido disponibilizado.
6. Dê o nome que desejar ao stack.
7. No campo KeyName selecione o mesmo nome com que salvou a chave anteriormente.
8. No campo PrivateKey, copie e cole todo o conteúdo (inclusive cabeçalhos e rodapé) do arquivo *.pem* salvo anteriormente e avançe.
9. Na página de Revisão (Review) marque a opção no final da tela dando permissão para criar recursos e conclua o processo.
10. Aguarde a criação de todos os recursos.

Acessando a Infraestrutura

1. Acesse a página do EC2, seguido pela opção Instâncias no menu à esquerda.
2. Haverá uma instância chamada NAT, essa instância é o ponto de acesso as VMs e tem um endereço IP público para acesso ssh, guarde este endereço.
3. Acesse a instância NAT via SSH com PuTTY ou terminal, conforme sua preferência, utilizando 'ec2-user' como nome de usuário.
4. Uma vez acessada a instância NAT, as demais instâncias, destinadas a computação, podem ser acessadas utilizando ssh.
    a. O *alias* de acesso as demais instâncias segue o padrão: node01, node02 … node10.

Notas Finais

IMPORTANTE: Desligar as instâncias sempre que parar de utilizá-las. O não desligamento das instâncias pode resultar em cobranças por uso (Total máximo gratuito de 750h/mês dividido entre todas as instâncias).

Enviando arquivos para as Instâncias

- Utilizando o recurso S3 do AWS, crie um bucket com um nome fácil de lembrar e digitar. Faça upload de arquivos para dentro deste bucket.
- Utilizando a opção 'Propriedades' ao selecionar o arquivo *upado*, dê permissões para 'Qualquer usuário AWS autenticado'.
- A partir da instância em que desejar o arquivo, utilize o comando:
  $aws s3 cp s3://nome-do-bucket/arquivo-remoto ./arquivo-local

  Ou para sincronizar todo o bucket:
  $aws s3 sync s3://nome-do-bucket diretorio-local

  Esse último comando mantém as modificações feitas no bucket atualizadas no diretorio local. As mudanças feitas localmente podem ser enviadas ao bucket utilizando o comando:
  $aws s3 sync diretorio-local s3://nome-do-bucket

  Mais informações sobre o S3 podem ser encontradas em:
http://docs.aws.amazon.com/cli/latest/userguide/using-s3-commands.html