Universidade Federal
do Rio de Janeiro
Escola Politécnica

IMPLEMENTAÇÃO DE AUTENTICAÇÃO FEDERADA EM UMA NUVEM COMUNITÁRIA GEODISTRIBUÍDA

Pedro Henrique Cruz Caminha

Projeto de Graduação apresentado ao Curso de Engenharia de Computação e Informação da Escola Politécnica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro.

Orientador: Luís Henrique Maciel Kosmalski Costa

Co-orientador: Rodrigo de Souza Couto

Rio de Janeiro

Abril de 2016

IMPLEMENTAÇÃO DE AUTENTICAÇÃO FEDERADA EM UMA NUVEM COMUNITÁRIA GEODISTRIBUÍDA

Pedro Henrique Cruz Caminha

PROJETO DE GRADUAÇÃO SUBMETIDO AO CORPO DOCENTE DO CURSO DE ENGENHARIA DE COMPUTAÇÃO E INFORMAÇÃO DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO DE COMPUTAÇÃO E INFORMAÇÃO.

Autor:

Pedro Henrique Cruz Caminha

Orientador:

Prof. Luís Henrique Maciel Kosmalski Costa, Dr.

Co-orientador:

Prof. Rodrigo de Souza Couto, D.Sc.

Examinador:

Prof. Marcelo Gonçalves Rubinstein, D.Sc.

Examinador:

Prof. Miguel Elias Mitre Campista, D.Sc.

Rio de Janeiro

Abril de 2016

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
Escola Politécnica - Departamento de Eletrônica e de Computação
Centro de Tecnologia, bloco H, sala H-217, Cidade Universitária
Rio de Janeiro - RJ CEP 21949-900

Este exemplar é de propriedade da Universidade Federal do Rio de Janeiro, que poderá incluí-lo em base de dados, armazenar em computador, microfilmar ou adotar qualquer forma de arquivamento.

É permitida a menção, reprodução parcial ou integral e a transmissão entre bibliotecas deste trabalho, sem modificação de seu texto, em qualquer meio que esteja ou venha a ser fixado, para pesquisa acadêmica, comentários e citações, desde que sem finalidade comercial e que seja feita a referência bibliográfica completa.

Os conceitos expressos neste trabalho são de responsabilidade do autor.

DEDICATÓRIA

Dedico este trabalho às pessoas que me fazem bem.

AGRADECIMENTO

Agradeço a todas as pessoas que contribuíram com a minha educação e, consequentemente, com este projeto. Mais especificamente, agradeço à minha família, às minhas amigas e amigos, meus professores e professoras e, por último, a todo o povo brasileiro, que investiu em mim de forma tão significativa. Espero que esse seja apenas o início de uma longa história de colaborações em retribuição aos esforços de todas essas pessoas.

RESUMO

A computação em nuvem é uma importante ferramenta para resolver problemas de tecnologia de informação. Uma de suas formas de apresentação é a de Infraestrutura como Serviço (IaaS - Infrastructure as a Service), na qual um provedor fornece poder computacional através da virtualização de recursos computacionais reais, controlados por esse provedor.

Para que seja oferecida uma plataforma de IaaS, é necessário que haja recursos reais disponíveis para a virtualização. Um modelo com o objetivo de reunir os recursos necessários para oferecer IaaS é o compartilhamento de recursos entre diversas instituições. No caso geral, instituições diferentes fazem parte de domínios administrativos diferentes e possuem localizações diferentes. Se houver o compartilhamento de recursos dessas instituições para a implementação de Infraestrutura como Serviço, é criada uma nuvem comunitária e geodistribuída.

O Grupo de Trabalho Plataforma IaaS Distribuída (GT-PID) é um projeto com o objetivo de implementar uma nuvem comunitária e geodistribuída, fornecendo uma Infraestrutura como Serviço voltada para instituições de ensino e pesquisa. A implementação de uma nuvem comunitária e geodistribuída gera desafios pois, além de questões ligadas à implementação de uma nuvem administrativa e geograficamente centralizada, existe a necessidade de garantir transparência aos usuários com relação a essas peculiaridades. Entre os desafios encontrados nessa implementação está o gerenciamento de identidades de usuários oriundos das diversas instituições participantes da nuvem comunitária implantada.

Assim, este trabalho realiza a implementação da autenticação federada no serviço distribuído de infraestrutura proporcionado pelo GT-PID como solução para o gerenciamento de usuários de uma Nuvem Comunitária geodistribuída.
Palavras-Chave: computação em nuvem, Infraestrutura como Serviço, nuvem comunitária, autenticação federada, GT-PID, OpenStack, Shibboleth, CAFe Expresso.

ABSTRACT

Cloud Computing is a valuable tool for solving Information Technology problems, very often in the form of Infrastructure as a Service (IaaS). In this form, a provider offers virtualized resources to its users.

In order to offer virtualized resources, it is necessary to have real resources available for virtualization. A model for IaaS construction is to share resources among partner institutions, resulting in the geographic and administrative distribution of the service. The general case is that different institutions are in different administrative domains and geographic locations. If such institutions share their resources in order to create an IaaS platform, a community and geodistributed cloud is built.

Grupo de Trabalho Plataforma de IaaS Distribuída (from portuguese, Work Group for Distributed IaaS Platform - GT-PID) is a research group focused on implementing a community geodistributed cloud. Building a community and geodistributed cloud creates many challenges, because, other than the issues regarding the creation of a centralized cloud, there is also the need of making the administrative and geographic distribution of the cloud transparent to the users. Among the challenges faced to build such cloud, lies the identity management of users from all the different partner institutions.

The present work offers the federated identity management model as a way to optimize the management in the distributed IaaS built by GT-PID.

Key-words: Cloud Computing, IaaS, Infrastructure as a Service, Community Cloud, federated authentication, GT-PID, OpenStack, Shibboleth, CAFe Expresso.

SIGLAS

CAFe - Comunidade Acadêmica Federada;

CAFe Expresso - Comunidade Acadêmica Federada para Experimentação;

GT-PID - Grupo de Trabalho Plataforma IaaS Distribuída;

IdP - *Identity Provider* (Provedor de Identidade);

RNP - Rede Nacional de Ensino e Pesquisa;

SP - *Service Provider* (Provedor de Serviço);

SSO - *Single Sign On* (Autenticação Única);

VM - *Virtual Machine* (Máquina Virtual);

WAYF - *Where Are You From* (De onde você é?).

Sumário

1 Introdução **1**

1.1 Computação em Nuvem 2

1.1.1 Centralização e distribuição de serviços 4

1.1.2 Nuvens comunitárias 5

1.1.3 Categorias de serviços 6

1.1.4 Virtualização de recursos 7

1.1.5 Controle de acesso 8

1.2 Nuvem comunitária e geodistribuída 9

1.3 Objetivos 10

1.4 Organização do texto 10

2 GT-PID: Grupo de Trabalho Plataforma IaaS Distribuída **12**

2.1 Grupos de Trabalho da RNP 13

2.2 Arquitetura do GT-PID 14

2.2.1 Operações fornecidas 16

2.2.2 O controle de acesso no GT-PID 17

2.2.3 Tecnologias envolvidas 17

3 Autenticação Federada **19**

3.1 Modelos de autenticação 20

3.1.1 Modelo isolado 21

3.1.2 Modelo centralizado 21

3.1.3 Modelo federado 23

3.2 Serviços de descoberta de IdPs 25

3.3 CAFe e CAFe Expresso 26

4 O Orquestrador de Nuvens OpenStack **29**

4.1 Arquitetura modular 30

4.1.1 Horizon 31

4.1.2 Keystone 32

4.2 Ambiente de desenvolvimento 36

4.3 Suporte à autenticação federada 36

5 O Arcabouço de Software Shibboleth **41**

5.1 O padrão SAML2 42

5.2 Metadados, atributos e outras configurações 42

5.3 Where Are You From 43

6 Implantação e Testes **44**

6.1 Autenticação federada sem interface gráfica 44

6.2 Autenticação federada com interface gráfica 48

6.3 Testes 55

7 Conclusões e Trabalhos Futuros **56**

7.1 Contribuições 57

7.2 Trabalhos Futuros 58

Bibliografia **59**

Lista de Figuras

1.1 Utilização de recursos. 2
1.2 Categorias de serviços de computação em nuvem. 6

2.1 Necessidade de recursos de várias instituições. 13
2.2 Arquitetura do GT-PID. 15

3.1 Modelo isolado de gerenciamento de identidade. 21
3.2 Modelo centralizado de gerenciamento de identidade. 22
3.3 Modelo federado de gerenciamento de identidade. 24
3.4 Descoberta de IdP através de credenciais. 26
3.5 Descoberta de IdP através de serviço de WAYF. 27

4.1 Tela de login do GT-PID antes da implantação de autenticação federada. . 32
4.2 Autenticação e autorização de usuário junto ao Keystone. 35
4.3 Pedido de um usuário a um serviço. 36
4.4 Fluxo de autenticação federada suportado pela versão Juno do OpenStack. 39

6.1 Procedimento para autenticação federada sem interface gráfica. 48
6.2 Interface de serviço de WAYF. 49
6.3 Tela de autenticação de IdP. 50
6.4 Procedimento para autenticação federada com interface gráfica. 53
6.5 Tela de autenticação do GT-PID, com botão para autenticação através da CAFe Expresso. 54
6.6 Tela inicial do usuário após autenticação federada. 55

Capítulo 1

Introdução

A forma tradicional da utilização de infraestrutura computacional, de plataformas computacionais e de muitas aplicações em *software* se dá através da compra desses recursos ou de licenças para esses recursos. Uma instituição adquire tais recursos, os instala, utiliza e gerencia. Ao adquirir recursos na forma de sistemas e equipamentos, a instituição assume também a obrigação de gerenciar e manter tais sistemas e equipamentos, o que podem ser operações muito custosas. Não existe a possibilidade de redimensionamento, exceto por nova aquisição e venda ou até mesmo o descarte do equipamento antigo, caso se deseje diminuir a quantidade de equipamentos que precisam de manutenção e gerenciamento. Por outro lado, para garantia de atendimento à demanda de seus usuários, uma instituição deve adquirir com antecedência recursos suficientes para os momentos de utilização mais intensa.

Um outro problema relacionado é que, em grande parte das instituições de ensino e pesquisa, observa-se a utilização de recursos em rajadas. Tipicamente, existem longos períodos de menor utilização, intercalados por picos curtos de alta utilização [1]. Esse fenômeno é ilustrado na Figura 1.1. Então, é possível constatar que, se os picos de demandas dos usuários de uma instituição forem muito maiores do que a média de utilização ou, se o intervalo entre esses picos for muito longo, haverá ociosidade e subutilização dos recursos adquiridos para atender aos picos de demandas. Não é desejável que recursos adquiridos estejam ociosos, especialmente considerando que a aquisição, instalação e manutenção desses equipamentos é onerosa.

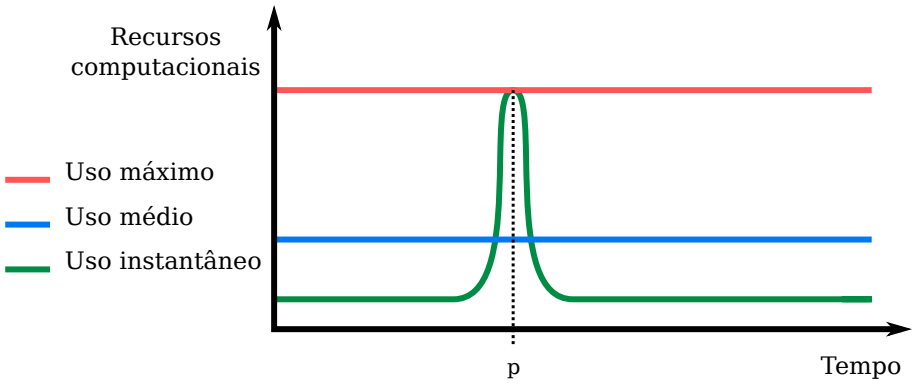

Figura 1.1: Utilização de recursos.

Como terceiro e último problema listado, cada equipamento ou sistema adquirido é limitado às suas características e especificações enquanto produto. Qualquer possibilidade de alterações está atrelada à obtenção de um novo equipamento com novas especificações.

Uma forma de atender aos picos de demanda, fornecer uma grande variedade de equipamentos e sistemas, diminuir custos com gerenciamento e manutenção, e ainda eliminar a ociosidade desses recursos é através da utilização de algum serviço de computação em nuvem. Este tipo de serviço permite "alugar um equipamento" ao invés de adquiri-lo.

1.1 Computação em Nuvem

A computação em nuvem é um modelo para que usuários tenham acesso a serviços compartilhados de computação através de alguma rede, com esforço mínimo por parte dos usuários e do provedor do serviço [2]. A rede utilizada é tipicamente a Internet [3]. Esses serviços podem ter diversas formas: processamento, armazenamento ou até mesmo aplicações, por exemplo. Inicialmente, é possível notar que a computação em nuvem exime instituições da gerência e manutenção de seus recursos computacionais. Além disso, existe um conjunto esperado de características que um serviço de computação em nuvem deve atender [3], que explica como a computação em nuvem resolve os outros problemas expostos para a forma tradicional

de utilização de infraestrutura computacional:

- **Elasticidade:** Recursos são entregues assim que o usuário manifesta sua necessidade por recursos;
- **Cobrança por utilização:** A cobrança se dá pelo tempo e intensidade de utilização;
- **Customização:** A oferta de recursos é variada, de forma que usuários podem escolher por tipos diferentes de recursos que podem ser ajustáveis de acordo com suas demandas;
- **Autonomia de usuários:** Usuários devem ser capazes de utilizar os recursos da nuvem de forma independente e intuitiva.

A elasticidade permite que instituições não necessitem adquirir recursos antecipadamente para atender a picos de demanda: o crescimento da necessidade de recursos pode ser atendido por recursos oferecidos por um serviço de computação em nuvem e, com o decréscimo da demanda, esses recursos deixam de ser utilizados.

Devido à cobrança por utilização, enquanto recursos de um serviço de computação em nuvem forem utilizados, a instituição será cobrada. Porém, quando não há utilização, não há cobrança. Essa característica possibilita à instituição apenas ter custos com recursos que estão, de fato, sendo utilizados.

A customização possibilita que uma instituição utilize equipamentos e sistemas que possuem especificações e características ajustáveis. Quando combinada com a elasticidade, a customização garante que as necessidades de equipamentos e sistemas de uma instituição ao longo do tempo possam ser supridas por um serviço de computação em nuvem.

É possível que uma instituição oriente seus usuários para que utilizem recursos de um serviço de computação em nuvem, ao invés de adquirir equipamentos, licenças e sistemas suficientes para atender a suas diferentes demandas. Dessa forma, a instituição poderá ter menos custos tanto na obtenção de recursos quanto na manutenção de tais recursos.

Outro aspecto importante da computação em nuvem é que, idealmente, recursos oferecidos por um serviço de computação em nuvem estão disponíveis de forma quase imediata, diminuindo o tempo de espera do usuário pela aquisição de equipamentos ou sistemas quando há necessidade de uso.

1.1.1 Centralização e distribuição de serviços

Para que um serviço de computação em nuvem possa ser fornecido, é preciso que existam recursos capazes de prover o poder computacional necessário aos usuários. Esses recursos podem estar localizados em um único centro que agrega todos os recursos disponíveis, ou distribuídos em diversos servidores espalhados geograficamente. Quando um serviço de computação em nuvem é provido com a utilização de recursos que estão distribuídos de maneira geográfica, esse é dito um serviço geo-distribuído [1]. Cada um desses locais que servem como pontos de fornecimento de recursos é chamado de sítio [1].

Com a distribuição de um serviço de computação em nuvem, surgem novos desafios, mas também vantagens. Por exemplo, a comunicação entre equipamentos localizados em sítios distintos pode experimentar um maior tempo de latência do que a comunicação entre equipamentos localizados em um mesmo sítio. Por outro lado, a utilização de equipamentos localizados em sítios diferentes diminui as chances de falhas simultâneas [1]. Além disso, uma maior quantidade e distribuição de sítios pode aumentar as chances de haver proximidade entre um usuário qualquer e um desses centros, diminuindo o tempo de comunicação entre centro e usuário e, consequentemente, causando uma redução do tempo de resposta nas operações requisitadas pelo usuário e executadas no centro.

Por conta das características citadas anteriormente e também de outras características, a distribuição de um serviço de computação em nuvem pode ser vantajosa em diversos casos. Desde a impossibilidade de manter um grande centro de dados, a diminuição do tempo de latência médio para os usuários e até a diminuição das chances de falha simultânea. No caso do serviço de computação em nuvem referido neste trabalho, uma das preocupações foi possibilitar a criação de uma nuvem

comunitária, conceito abordado na Seção 1.1.2.

1.1.2 Nuvens comunitárias

A centralização de um serviço de computação em nuvem pode ser do ponto de vista arquitetural, mas também organizacional. Se o serviço de computação em nuvem é completamente gerenciado e mantido sob a responsabilidade de uma única instituição, esse serviço pode ser dito centralizado a nível organizacional. Em contraste, se um serviço é gerenciado e mantido por diversas instituições, esse serviço é dito uma nuvem comunitária [3].

Por haver instituições diferentes relacionadas ao gerenciamento de uma nuvem comunitária, é possível que existam políticas diferentes de gerenciamento entre as instituições e a unificação dessas políticas pode se mostrar complexa. As instituições podem ter normas diferentes a respeito da aquisição de equipamentos, procedimentos distintos para a utilização desses equipamentos, horários de funcionamento diferentes, políticas de segurança diferentes, entre outras peculiaridades que qualquer uma das instituições mantenedoras da nuvem comunitária pode ter. Essas diferenças criam necessidades particulares a cada instituição que devem ser atendidas pelo sistema que provê o serviço de computação em nuvem, mantendo as características importantes ao serviço.

Assim como no caso da distribuição da arquitetura apontado na Seção 1.1.1, é importante que o sistema que implementa o serviço de nuvem seja capaz de tornar transparentes para o usuário os aspectos da distribuição. O usuário deve ser capaz de enxergar apenas um único serviço, exceto quando for necessário. É importante que o usuário seja afetado pelos aspectos positivos da nuvem comunitária e esteja isolado de quaisquer complicações oriundas da distribuição organizacional, que devem ser resolvidas pelo sistema que implementa a nuvem. Este trabalho implementa a adição de uma funcionalidade em uma nuvem comunitária que possibilita o gerenciamento de usuários por parte de cada uma das instituições que compõem a nuvem, em uma implementação que modifica o serviço o mínimo possível do ponto de vista dos usuários.

1.1.3 Categorias de serviços

Os serviços de computação em nuvem são classificados em categorias, de acordo com o tipo de serviço que oferecem. A Figura 1.2 ilustra a divisão dos serviços de computação em nuvem.

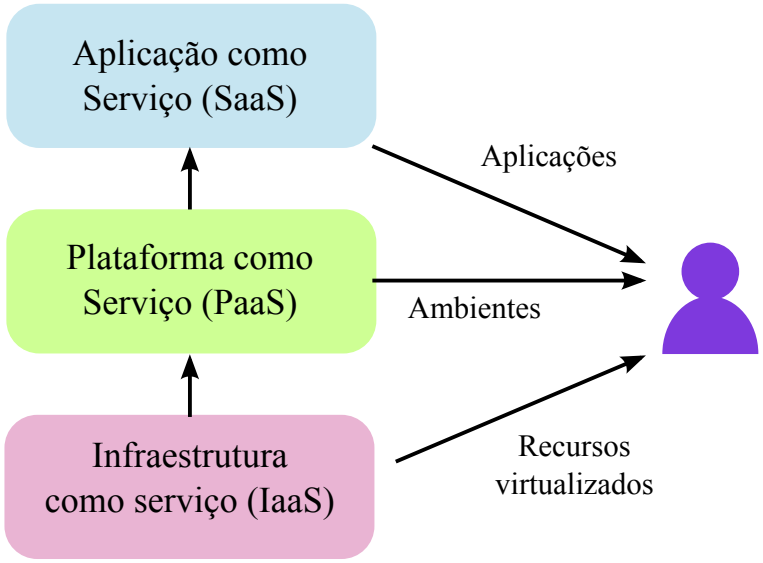

Figura 1.2: Categorias de serviços de computação em nuvem.

Ao realizar a entrega de diferentes tipos de serviço, as categorias possuem também diferentes níveis de abstração:

- **Aplicação como Serviço:** A categoria de nível mais alto de abstração é a chamada categoria de Aplicação como Serviço (*Software as a Service* - SaaS). Nessa categoria, existem aplicações que são oferecidas para o usuário através da Internet ou alguma outra rede, mas que são executadas utilizando recursos localizados em servidores, e não na máquina que o usuário utiliza para acessar o serviço. Esse tipo de serviço se apresenta de muitas formas diferentes, como redes sociais ou editores de arquivos pela Internet [3].

- **Plataforma como Serviço:** Na segunda categoria, intermediária, está a Plataforma como Serviço (*Platform as a Service* - PaaS). Nas PaaS, os usuários têm acesso a ambientes onde podem realizar a instalação e execução de aplicações de seu interesse. Esses serviços podem ser utilizados por desenvolvedores

que desejam executar suas aplicações sem as complicações de gerenciar o ambiente necessário, como aplicações web [3].

- **Infraestrutura como Serviço:** A categoria de nível mais baixo de abstração é a da Infraestrutura como Serviço (*Infraestructure as a Service* - IaaS). Os serviços dessa categoria proveem a seus usuários poder computacional na forma de equipamentos virtualizados, que podem ser acessados através da Internet ou alguma outra rede [3].

Este trabalho é desenvolvido sobre um serviço de IaaS, de forma que as outras categorias estão fora do escopo deste trabalho e não serão analisadas. Algumas noções e conceitos podem ser generalizados para todas as categorias de serviços de computação em nuvem, mas essas discussões não serão abordadas nesse texto.

1.1.4 Virtualização de recursos

Os principais elementos fornecidos por um serviço de IaaS são máquinas virtuais (VMs), discos virtuais e roteadores virtuais. Esses elementos são emulações geradas por equipamentos reais [3]. As emulações são feitas para assegurar as características de elasticidade e customização, desejáveis em um serviço de computação em nuvem. Através da emulação de equipamentos específicos a partir de equipamentos genéricos, um provedor de IaaS pode evitar a obrigação de possuir equipamentos com as exatas especificações dos elementos que se deseja fornecer aos usuários.

É possível, por exemplo, que um fornecedor de serviço de IaaS possua servidores com alto poder computacional e faça com que cada um desses servidores ofereça um número de VMs que se comportem como computadores de menor porte. Um usuário poderia alocar um grande número de VMs, de forma que parecesse ao usuário que o poder computacional oferecido é infinito. A virtualização de recursos é frequentemente utilizada para subdividir ou combinar o poder computacional dos servidores de um serviço de IaaS, a fim de reconstruir o poder computacional demandado por um usuário. Dessa forma, é provida a elasticidade.

Em outro caso de uso, se o equipamento controlado pelo provedor de IaaS é capaz de emular diversos tipos de VMs, a técnica de virtualização permite que a plataforma de IaaS seja capaz de prover a customização do serviço oferecido sem que os equipamentos reais que fornecem tal serviço sejam alterados.

Naturalmente, um serviço de IaaS consegue oferecer poder computacional apenas menor ou igual ao poder dos recursos possuídos pelo provedor dessa plataforma de IaaS. Isso significa que para a implementação de uma plataforma de IaaS é necessário, entre outras coisas, que haja máquinas e discos reais disponíveis para o fornecimento do poder computacional desejado. Existem várias formas de garantir que existam recursos suficientes para a criação de um serviço de IaaS. Este trabalho é parte integrante de uma plataforma de Infraestrutura como Serviço que utiliza o compartilhamento de recursos ociosos entre instituições para fornecer poder computacional a seus usuários.

1.1.5 Controle de acesso

Serviços nem sempre podem ser acessados por quaisquer usuários. Se existirem restrições com relação aos usuários que podem fazer uso de um serviço, é essencial que o provedor do serviço seja capaz de decidir servir ou não algum usuário que tente acessá-lo. Adicionalmente, não deve haver prejuízo aos usuários que têm permissão para a utilização do serviço. Os métodos utilizados para o controle de acesso devem garantir que as políticas de oferta de serviços do provedor de IaaS sejam atendidas ao mesmo tempo em que a experiência dos usuários não seja prejudicada.

No caso em que um serviço de computação em nuvem está totalmente sob responsabilidade de uma única instituição, o controle de acesso depende administrativamente e tecnicamente apenas da instituição que controla essa nuvem. Administrativamente porque a instituição decide quais usuários podem acessar o serviço e o que os usuários precisam fazer para tal; tecnicamente porque a instituição implementa o controle de acesso, diretamente ou através de terceiros. No caso de nuvens comunitárias, formadas por recursos fornecidos por diferentes instituições, os usuários têm acesso de acordo com um modelo onde diversas instituições podem decidir sobre

quem pode utilizar o serviço. Ou seja, há uma administração distribuída no controle de acesso.

Assim, este trabalho implementa para o serviço de IaaS fornecido por uma nuvem comunitária o paradigma da autenticação federada, de forma a proporcionar uma separação técnica no controle ao acesso dos usuários do serviço. No paradigma proposto, a autenticação do usuário é feita por uma entidade diferente daquela que fornece o serviço, podendo haver diversas entidades que fornecem serviços e tantas outras que autentiquem usuários. Uma discussão mais aprofundada sobre a autenticação federada é feita no Capítulo 3.

O objetivo final da separação técnica do controle ao acesso de usuários à nuvem comunitária é possibilitar que cada instituição possa ter controle técnico sobre o gerenciamento de seus usuários, transferindo a responsabilidade de adequação às políticas internas de cada instituição para as próprias instituições, individualmente.

1.2 Nuvem comunitária e geodistribuída

O presente trabalho está inserido no contexto da nuvem comunitária e geodistribuída proposta pelo Grupo de Trabalho - Plataforma IaaS Distribuída (GT-PID), que tem como objetivo criar um serviço de IaaS através do compartilhamento de recursos ociosos de infraestrutura computacional pertencentes a instituições de ensino e pesquisa conveniadas à Rede Nacional de Ensino e Pesquisa (RNP)[1]. A RNP é uma Organização Social de fomento ao desenvolvimento tecnológico e pesquisas na área de tecnologia da informação e é mantida pelo Ministério da Ciência, Tecnologia e Inovação, pelo Ministério da Educação, pelo Ministério da Cultura e pelo Ministério da Saúde [4]. Por conta disso, existe o interesse dessa instituição em organizar um projeto desta natureza, que ofereça serviços de computação em nuvem para as instituições conveniadas.

O GT-PID fornece uma interface gráfica através de uma página web para o gerenciamento e utilização da plataforma de IaaS, sendo possíveis as operações de criação e utilização de VMs, discos virtuais e outros serviços relacionados. A partir da

criação de uma nuvem comunitária, surge o problema da autenticação de usuários oriundos de diferentes instituições. Este trabalho propõe uma implantação de autenticação federada para o serviço de IaaS fornecido pelo GT-PID. A motivação e arquitetura do GT-PID são exploradas no Capítulo 2.

1.3 Objetivos

O objetivo principal deste trabalho é efetuar a implantação de uma sequência de autenticação para usuários do GT-PID de forma que os usuários possam utilizar a interface gráfica para realizar autenticação federada. O resultado esperado é que, ao final do trabalho, as instituições consigam compartilhar seus recursos computacionais através do GT-PID com usuários em quem elas confiem, ao mesmo tempo em que não seja necessário para o GT-PID implementar demandas individuais de instituições com relação ao gerenciamento de seus usuários.

A computação em nuvem e seus serviços de infraestrutura são importantes instrumentos para empresas, universidades, governos e até mesmo pessoas individualmente. Tornar a gerência e utilização dos serviços de infraestrutura mais simples é uma forma de facilitar todos esses agentes. Mais especificamente, este trabalho propõe uma melhoria num serviço planejado por uma organização social para ser utilizado por universidades e financiado com recursos governamentais, influenciando de forma positiva, direta e indiretamente, o relacionamento entre esses agentes e o serviço oferecido.

Um objetivo secundário do trabalho é que, ao longo de sua execução, seja realizada uma documentação da implantação dos procedimentos de autenticação suportados pelos sistemas e plataformas utilizadas. Essa contribuição visa à simplificação de novas implementações que possam ocorrer.

1.4 Organização do texto

Este trabalho está organizado da seguinte forma: o Capítulo 2 posiciona o projeto desenvolvido no contexto de um projeto maior, o GT-PID. O Capítulo 3 detalha

o modelo de gerenciamento de identidades federado, adotado por este projeto. O Capítulo 4 apresenta a arquitetura do sistema utilizado para a orquestração da nuvem, suas funcionalidades de interface gráfica e autenticação. O Capítulo 5 apresenta o arcabouço utilizado para a comunicação entre a plataforma de IaaS e as autoridades de autenticação, enquanto o Capítulo 6 especifica alguns detalhes da implantação realizada neste trabalho. Finalmente, o Capítulo 7 conclui o trabalho e apresenta possíveis trabalhos futuros.

Capítulo 2

GT-PID: Grupo de Trabalho Plataforma IaaS Distribuída

Como mencionado no Capítulo 1, a utilização de infraestrutura computacional por instituições frequentemente se dá em rajadas. Consequentemente, a aquisição de equipamentos suficientes para suprir os picos de demanda gera um esforço que resulta na obtenção de equipamentos que podem permanecer ociosos por consideráveis períodos. Por outro lado, a utilização de serviços de computação em nuvem implica cobrança por utilização e a maior parte das instituições já possui algum tipo de infraestrutura. Isso significa que substituir toda a infraestrutura computacional de uma instituição por serviços de IaaS pode gerar cobranças indesejáveis e desnecessárias. Uma possível solução é adotar um modelo híbrido, onde a instituição adquire recursos suficientes para suprir sua demanda média e, em momentos de alta demanda, utiliza algum serviço de IaaS. Dessa forma, a instituição não precisa arcar com as despesas de recursos que seriam subutilizados e, ao mesmo, desabona-se da cobrança por uso de recursos da nuvem que são satisfatoriamente substituídos por infraestrutura própria. De forma complementar, a instituição pode desfrutar de outras vantagens de utilizar um serviço de Computação em Nuvem, como o acesso a uma variedade de equipamentos virtualizados que, caso contrário, não seria possível.

Adicionalmente, se os picos de utilização de infraestrutura por diferentes instituições acontecerem em momentos distintos, existem grandes chances de, no momento de pico de utilização por uma instituição, existir poder computacional ocioso

em algumas outras instituições [1]. Esse fenômeno é ilustrado na Figura 2.1. É possível, então, criar uma plataforma de IaaS que utilize recursos ociosos da infraestrutura de algumas instituições para suprir necessidades computacionais de outras instituições parceiras. Dessa forma, propõe-se um modelo de compartilhamento: uma instituição utiliza a infraestrutura de outras em seus momentos de pico e, em troca, cede infraestrutura em seus momentos de menor utilização. Esse esquema de compartilhamento forma uma nuvem que é simultaneamente comunitária e geodistribuída. A nuvem é comunitária porque diversas instituições são responsáveis pelo serviço de computação em nuvem; a nuvem é geodistribuída porque as infraestruturas das instituições estão localizadas em suas sedes, que estão espalhadas geograficamente.

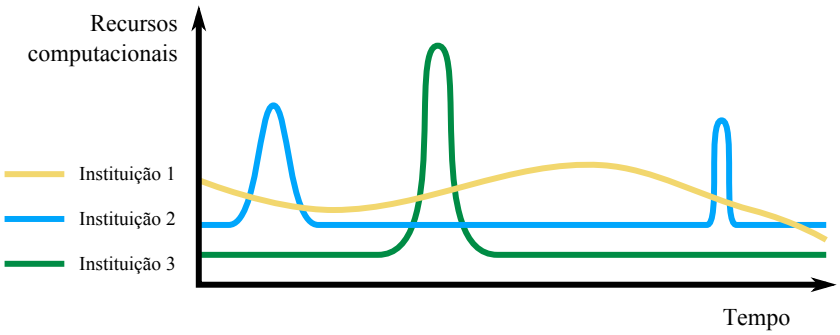

Figura 2.1: Uso de recursos computacionais de várias instituições em função do tempo.

Os desafios relacionados à implementação e implantação de nuvens comunitárias geodistribuídas são diversos [1]. Entre eles, pode-se citar a comunicação entre centros de dados geograficamente distantes, a decisão de sítios para a alocação de recursos, o atendimento às políticas de todas as instituições parceiras e a gerência de usuários e recursos que estão em domínios administrativos diferentes.

2.1 Grupos de Trabalho da RNP

Dentro de suas atribuições, a RNP possui um modelo de incentivo a projetos científicos através da criação de Grupos de Trabalho (GTs) [5]. A comunidade ci-

entífica é incentivada a criar GTs que possam se transformar em serviços oferecidos pela RNP. Como discutido no início deste capítulo, se a utilização de equipamentos por diversas instituições se der em rajadas e se os picos de utilização de cada uma dessas instituições não forem frequentemente concomitantes, a construção de uma nuvem comunitária entre essas instituições pode ser vantajosa. Devido à percepção desse padrão de utilização em universidades e centros de pesquisa clientes da RNP, iniciou-se um GT para a elaboração de uma nuvem comunitária entre essas instituições [1]. Nesse contexto, surgiu o Grupo de Trabalho Plataforma IaaS Distribuída, o GT-PID.

A proposta do GT-PID é construir uma nuvem comunitária visando interligar universidades e centros de pesquisa clientes da RNP, de forma que o poder computacional ocioso de cada um alimente uma plataforma de IaaS para máquinas e discos virtuais. Esse serviço está disponível para todas as instituições parceiras, para que, em momentos de necessidade de uso intensivo de recursos computacionais, as instituições possam utilizar o poder computacional disponibilizado pelo serviço. Em contrapartida, quando em uma dada instituição houver ociosidade de recursos computacionais, esses recursos devem estar à disposição para utilização por outras instituições parceiras. O principal objetivo é criar um ambiente de ajuda mútua, onde cada instituição não precise adquirir mais recursos que sua necessidade média. Dessa maneira, no GT-PID, a instituição contribui com a nuvem e recursos necessários em momentos de computação intensa são providos pelo GT-PID [1].

É importante propor uma arquitetura capaz de implementar a distribuição de recursos computacionais, na qual cada instituição mantenha seus recursos em uma ou mais sedes de sua preferência. A seguir, a Seção 2.2 expõe a arquitetura adotada pelo GT-PID.

2.2 Arquitetura do GT-PID

Para implementar a distribuição geográfica da provisão de IaaS, é utilizado um Controlador central que gerencia os recursos localizados em diversos sítios [1]. A Figura 2.2 ilustra essa arquitetura. Usuários que queiram utilizar máquinas ou

discos virtuais devem primeiro requisitá-los ao Controlador. O Controlador realiza a alocação dos recursos e, então, usuários podem utilizar os recursos alocados. As requisições ao Controlador e a utilização dos recursos fornecidos são realizadas pelos usuários através da Internet.

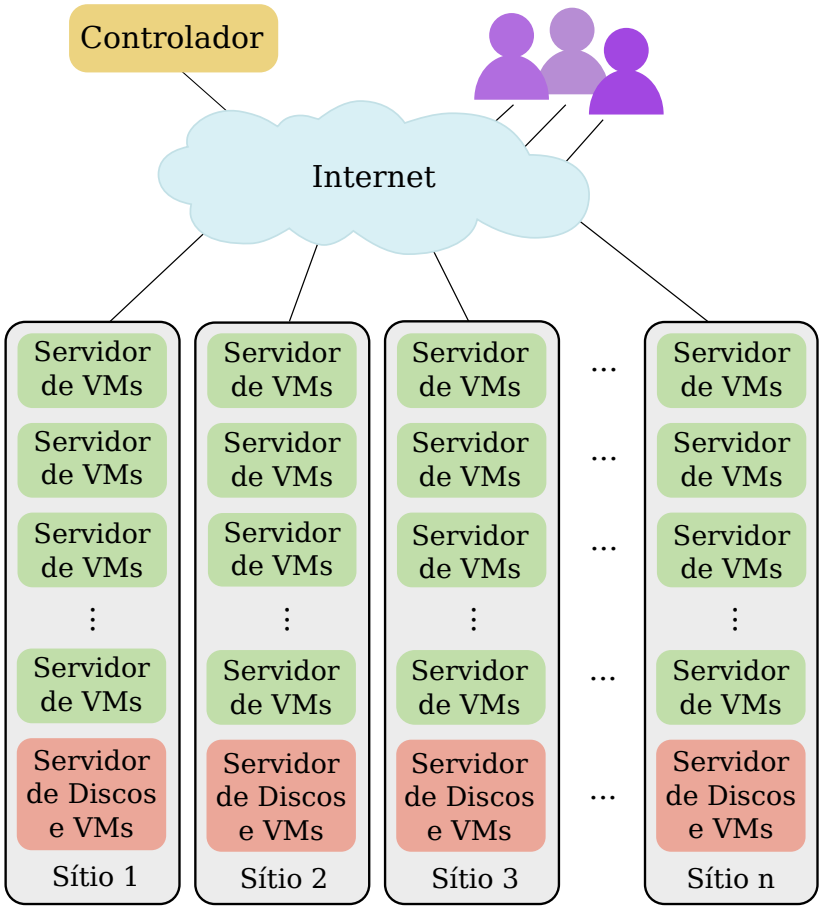

Figura 2.2: Arquitetura do GT-PID.

É importante ressaltar que o GT-PID possui servidores que executam três diferentes papéis, como observado na Figura 2.2 [1]:

- **Controlador:** Recebe chamadas dos usuários, aloca recursos e os entrega aos usuários. Há apenas um Controlador na arquitetura do GT-PID;
- **Servidor de VMs:** Fornece processamento aos usuários na forma de máquinas virtuais. Cada sítio pode hospedar diversos Servidores de VMs;

- **Servidor de VMs e Discos:** Fornece armazenamento aos usuários do GT-PID na forma de discos virtuais e também pode fornecer máquinas virtuais. Cada sítio deve ter exatamente um Servidor de VMs e Discos.

As configurações e modo de operação de cada um dos tipos de servidores são diferentes, visto que suas funções também são diferentes. O Controlador realiza a comunicação dos servidores com o ambiente externo ao GT-PID.

2.2.1 Operações fornecidas

As operações fornecidas pelo GT-PID são oferecidas de acordo com o papel desempenhado por quem está utilizando o sistema. Os papéis disponíveis no GT-PID são: administrador global, administrador local e usuário. Todos os administradores também possuem o papel de usuário e podem executar as operações que são oferecidas a usuários. A divisão da administração entre administrador local e administrador local existe para que seja possível administrar o serviço em nível global e local, permitindo que o administrador local ligado a um sítio gerencie os recursos desse sítio sem interferir em outros sítios.

O administrador global é responsável pela administração do GT-PID como um todo. Por isso, esse papel realiza as operações de criação e gerência de usuários e o estabelecimento de limites para a utilização de recursos pelos usuários [1]. O administrador local, por sua vez, é responsável apenas pela migração de VMs emuladas por um servidor para algum outro servidor localizado no mesmo sítio [1]. As operações fornecidas aos usuários do GT-PID são a instanciação de VMs, o acesso ao console das VMs e o upload de imagens para a inicialização das VMs [1].

Os usuários do GT-PID podem utilizar os serviços providos da maneira que seriam utilizadas em um serviço de IaaS centralizado, exceto pelo detalhe da escolha de sítios. Quando um usuário instancia uma VM no GT-PID, é possível escolher em qual dos sítios disponíveis a máquina será alocada ou até mesmo escolher algum modo automático no qual a escolha de sítios fique a cargo do Controlador [1].

Para que o GT-PID consiga reconhecer os usuários que tentam utilizar seus serviços e saber quais papéis esses usuários desempenham no sistema, existe um mecanismo para controle de acesso, que é o foco deste trabalho e introduzido na Seção a seguir.

2.2.2 O controle de acesso no GT-PID

É necessário que haja controle de acesso em serviços de computação em nuvem e, por consequência, em serviços de IaaS. Isso ocorre porque, em geral, diferentes usuários possuem permissão para executar operações distintas. Consequentemente, os serviços devem ser capazes de conhecer e reconhecer seus usuários de alguma forma.

Quando um serviço é oferecido, a entidade que o oferece usa políticas para gerenciar os usuários e suas permissões. No caso do GT-PID, existem diversas instituições que, simultaneamente, fazem parte da infraestrutura que oferece o serviço. Além disso, os usuários do serviço são vinculados a essas instituições. Cada uma das instituições possui suas próprias políticas para gerenciar usuários vinculados a ela. Assim, é necessário que o GT-PID seja capaz de atender às políticas de cada uma dessas entidades quando usuários utilizarem o serviço. A solução encontrada para isso é a utilização de autenticação federada, que é o foco deste trabalho e introduzida no Capítulo 3. Essa solução permite que cada instituição controle o acesso de seus próprios usuários ao serviço oferecido pelo GT-PID, além de permitir que o GT-PID se integre a serviços que já são oferecidos pela instituição e pela RNP.

2.2.3 Tecnologias envolvidas

Para a realização do projeto GT-PID, foram utilizadas diversas tecnologias e sistemas. Algumas dessas tecnologias e sistemas não fazem parte do escopo deste trabalho, pois não possuem impacto na autenticação de usuários do GT-PID. É possível citar como principal exemplo o Open vSwitch, que possibilita a comunicação de máquinas virtuais de sítios diferentes. Também são de extrema importância tecnologias de virtualização de processamento e de disco.

Para este trabalho foi utilizado como plataforma de nuvem e núcleo do GT-PID, o OpenStack [6] e, como arcabouço de autenticação federada, foi utilizado o Shibboleth [7]. O OpenStack e o Shibboleth utilizam internamente outras tecnologias que também serão abordadas nesse trabalho, como HTTP (*Hypertext Transfer Protocol*), a arquitetura REST (*Representational State Transfer*) [8] e a linguagem Python [9].

O OpenStack será abordado no Capítulo 4 e o Shibboleth no Capítulo 5. Nesses capítulos são explicadas as importâncias de cada um desses sistemas para o presente trabalho e alguns aspectos de seus mecanismos.

Capítulo 3

Autenticação Federada

Se um serviço demanda controle de acesso de seus possíveis usuários, é preciso que esse serviço seja capaz de conhecer e reconhecer os usuários que devem ter acesso garantido. Para que um serviço conheça um usuário do mundo real, é necessário que esse usuário seja representado no serviço por uma identidade [10]. Além disso, para que um serviço reconheça um usuário do mundo real, é fundamental que o serviço suporte algum procedimento de autenticação.

Um procedimento de autenticação verifica se um usuário que tente acessar o serviço através de uma determinada identidade é, de fato, o usuário representado por aquela identidade. Ou seja, o procedimento de autenticação é capaz de reconhecer um usuário real e relacioná-lo à sua identidade. Esse procedimento é executado por uma autoridade autenticadora, que possui as informações de identidade dos usuários capazes de acessar determinado serviço e também mecanismos para associar essas identidades a usuários reais.

Geralmente o procedimento de autenticação acontece por meio de credenciais pertencentes ao usuário que são entregues à autoridade autenticadora. A autoridade autenticadora busca em uma base de dados interna por alguma identidade que corresponda àquelas credenciais. Se as credenciais forem encontradas, a autoridade autenticadora considera que a identidade utilizada pelo usuário é a identidade que o representa e o usuário é quem informou ser [10].

Se, ao final do procedimento, ficar confirmado que a identidade corresponde a esse usuário, devem ser verificadas e garantidas as permissões de acesso do usuário ao serviço. Ou seja, devem ser decididas quais operações do serviço podem ser executadas por aquele usuário. Esse último procedimento é chamado de autorização [10].

Além do procedimento de autenticação, uma autoridade autenticadora também precisa realizar todas as operações relacionadas à gerência das identidades existentes, como cadastrar novos usuários ou apagar o cadastro de usuários que não devem mais ter acesso aos sistemas. Existem diferentes maneiras de realizar esse gerenciamento. Algumas formas de gerenciamento de identidade são apresentadas a seguir.

3.1 Modelos de autenticação

Os três tipos mais conhecidos de arquitetura para gerenciamento de identidades são: o modelo isolado, o modelo centralizado e o modelo federado. Eles conferem, em ordem crescente, maior flexibilidade para a autenticação. Em contrapartida, aumentam a complexidade dos sistemas que os implementam.

Os modelos centralizado e federado possuem dois tipos diferentes de entidades. O Provedor de Identidade (IdP - *Identity Provider*) é uma entidade que gerencia as informações de identidade dos usuários de um conjunto de serviços. O IdP é capaz de autenticar usuários, verificando se esses usuários são realmente quem dizem ser. Um Provedor de Serviço (SP - *Service Provider*) é uma entidade que fornece um serviço qualquer a seus usuários. Dependendo do tipo de modelo em que os SPs estão inseridos, eles são capazes de confiar em um ou mais IdPs. No modelo isolado, a mesma entidade concentra as atribuições de IdP e SP.

O modelo de autenticação centralizada e o modelo federado possibilitam que os usuários possuam apenas uma identidade e credencial para acessar todos os serviços abrangidos, uma funcionalidade chamada *Single Sign On* (SSO) [10]. O resultado final é uma simplicidade de utilização para os usuários, que só precisam realizar um procedimento de cadastro e necessitam apenas de um conjunto de credenciais para todos os serviços fornecidos por todos os SPs abrangidos. As arquiteturas de cada

modelo são abordadas nas subseções seguintes.

3.1.1 Modelo isolado

No modelo isolado, o provedor de um serviço qualquer realiza a gerência das identidades de usuários que acessam o serviço provido [10]. Nesse modelo, o próprio provedor deve arcar com todos os custos e complicações referentes a criar identidades, autenticar e autorizar os usuários do serviço provido. Um usuário que acesse diversos serviços deverá ter uma identidade e credencial para cada um desses serviços. A Figura 3.1 ilustra o procedimento de autenticação para esse modelo.

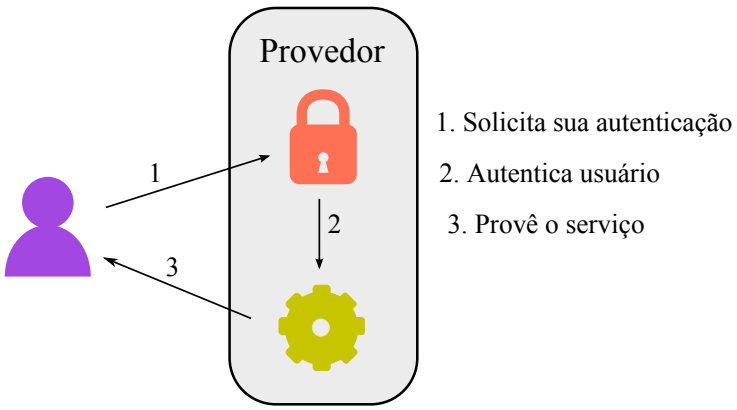

Figura 3.1: Modelo isolado de gerenciamento de identidade.

Nesse modelo, cada provedor de serviço possui independência para gerir as identidades de seus usuários, mas possui também a responsabilidade de fazer isso. Não é possível que um provedor serviço se abstenha dessa tarefa, a não ser que abdique de controlar o acesso de seus usuários.

O modelo isolado de autenticação é tradicionalmente utilizado na autenticação de usuários em PCs compartilhados por diversas pessoas, por exemplo.

3.1.2 Modelo centralizado

No modelo de gerenciamento centralizado, um único IdP realiza o gerenciamento de identidade para os serviços conveniados, oferecidos por diversos SPs [10]. Todos

os SPs ficam isolados do gerenciamento de identidade de seus usuários. Usuários que queiram acessar um serviço oferecido por um SP devem primeiro ser autenticados pelo IdP, num procedimento chamado de autenticação centralizada. O IdP informa ao SP que o usuário foi autenticado e o SP então oferece o serviço ao usuário. O procedimento de autenticação nesse modelo é ilustrado na Figura 3.2. Um exemplo de sistema que segue o modelo centralizado de autenticação é o LDAP (*Lightweight Directory Access Protocol*) [11], que concentra todos os dados de identidade em um servidor e controla o acesso de usuários a diversos serviços diferentes.

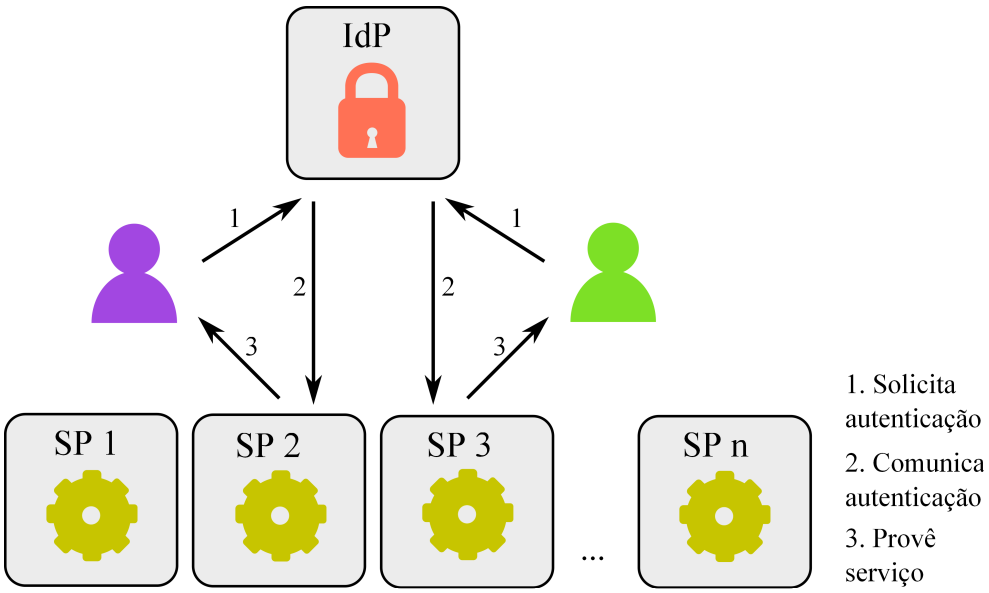

Figura 3.2: Modelo centralizado de gerenciamento de identidade.

Existem duas formas comuns de implementação da interação de usuários com a autenticação centralizada. Na primeira delas, o SP coleta as credenciais dos usuários e as repassa ao IdP. Na segunda, qualquer tentativa de acesso ao serviço por um usuário não autenticado é interceptada e redirecionada para o IdP, que realiza a autenticação e redireciona o usuário já autenticado para o SP.

A autorização de um usuário pode ser feita tanto pelo IdP quanto pelo SP. Como visto anteriormente, a autorização consiste em verificar quais operações um usuário tem permissão de realizar em um determinado serviço. No caso em que o IdP realiza

a autorização dos usuários, o IdP precisa enviar ao SP uma lista com as permissões de cada usuário autenticado; no caso em que o SP realiza a autorização dos usuários, o IdP precisa enviar ao SP uma identificação de cada usuário autenticado, para que o SP verifique as permissões de acesso desses usuários.

A identificação do usuário emitida pelo IdP para o SP se dá através de atributos previamente combinados entre o IdP e o SP, de modo que o SP seja capaz de relacionar o usuário autenticado às operações permitidas para esse usuário. Em alguns casos de uso, o SP precisa reconhecer completamente o usuário, pois existe algum nível de personalização do serviço oferecido. Nesse caso, os atributos enviados pelo IdP devem ser capazes de identificar unicamente o usuário, para aquele SP.

O modelo centralizado permite que usuários possuam apenas um conjunto de credenciais para acessar diversos serviços, possibilitando a funcionalidade de *Single Sign-On*. Uma desvantagem desse modelo é que apenas um IdP deve realizar o gerenciamento de diversos serviços. Isso cria a obrigação de um único IdP se conformar às necessidades de todos os usuários, de todos os serviços abrangidos. Os SPs, por sua vez, se veem desobrigados de realizar o gerenciamento de identidades de seus usuários.

3.1.3 Modelo federado

O modelo federado de autenticação, por sua vez, pode possuir mais de um IdP que confiam e são confiados por mais de um SP, formando um ambiente chamado de federação [10]. Os IdPs e SPs podem fazer parte de domínios administrativos diferentes, que definem regras de confiança que garantem a operacionalidade da federação. Quando um usuário deseja acessar um serviço oferecido por um SP, esse usuário deve primeiro ser autenticado por algum dos IdPs que fazem parte da federação, de forma análoga ao procedimento que ocorre no modelo centralizado. O procedimento de autenticação executado em uma federação é chamado de autenticação federada e está ilustrado na Figura 3.3. A principal diferença entre os procedimentos de autenticação federada e a centralizada é que, como há diversos IdPs em uma federação, é necessário um passo com o objetivo de descobrir qual

IdP deve executar a autenticação do usuário. Esse passo será discutido com mais detalhes na Seção 3.2.

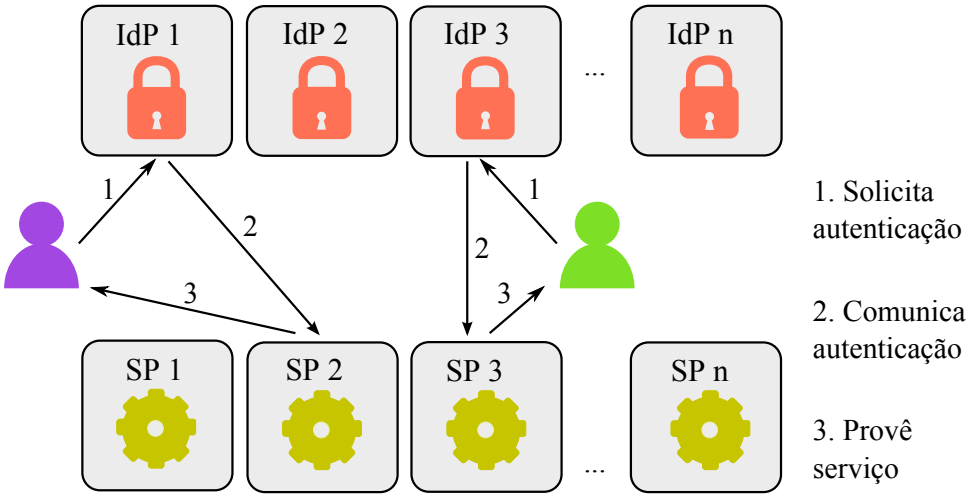

Figura 3.3: Modelo federado de gerenciamento de identidade.

Analogamente ao caso centralizado, existem duas possibilidades de interação do usuário com a autenticação federada: na primeira, o usuário envia suas credenciais para o SP quando tenta acessar o serviço e elas são repassadas para o IdP que realizará a autenticação; na segunda, o usuário tenta acessar um serviço fornecido por um SP e é redirecionado para um IdP de forma automática, realiza o procedimento de autenticação e, após o sucesso da autenticação, o usuário é redirecionado ao SP. Em ambos os casos, um SP deve receber as informações de identidade armazenadas pelo IdP que forem necessárias para esse SP [10].

Após o sucesso da autenticação junto a um IdP, o usuário poderá acessar o serviço fornecido pelo SP [10]. Assim como no modelo centralizado, existe a questão da autorização de usuário: se ela for executada pelo IdP, o IdP deve enviar ao SP as informações de permissão do usuário; se ela for executada pelo SP, o IdP deve enviar ao SP informações que identifiquem aquele usuário.

O modelo federado, assim como o modelo centralizado, permite que os SPs não tenham os encargos do gerenciamento de identidades. Do ponto de vista dos usuários,

em contraste com o modelo centralizado, o modelo federado ajuda os usuários a encontrarem em um IdP as políticas de gerenciamento de identidades que atendam às suas preferências ou necessidades. Além do mais, se esses usuários forem vinculados a alguma instituição, esse modelo proporciona às instituições independência para que realizem o gerenciamento de identidades de seus usuários, pois abre a possibilidade para que essas instituições controlem um ou mais IdPs, garantindo que as políticas de identificação de usuário lhes sejam adequadas. Por último, os usuários também podem desfrutar da funcionalidade de *Single Sign-On*.

Uma federação conhecida no meio acadêmico é a Eduroam [12], que possibilita que um usuário da infraestrutura de Internet de uma instituição de ensino seja autenticado por outra instituição de ensino. Quando um usuário vinculado a uma instituição de ensino tenta acessar uma rede pertencente a outra instituição de ensino, a instituição que controla a rede pede que um processo de autenticação do usuário seja executado pela instituição à qual o usuário está vinculado.

3.2 Serviços de descoberta de IdPs

Quando um usuário tenta acessar um serviço provido por um SP que está inserido em uma federação, é necessário que esse usuário seja autenticado por um, e apenas um, dos IdPs pertencentes a essa federação. É imprescindível, então, saber para qual IdP o usuário deve ser direcionado para autenticação. Existem duas formas para que isso aconteça, cada uma delas referente a uma possível forma de interação do usuário com a autenticação federada, mencionadas na Seção 3.1.3.

Caso o usuário insira suas credenciais diretamente no SP para que sejam repassadas ao IdP, é possível que as credenciais possuam alguma informação a respeito do IdP que deve realizar aquela autenticação. Por exemplo, se as credenciais forem email e senha, o domínio do email pode indicar o IdP a ser utilizado. A Figura 3.4 ilustra o procedimento.

Na situação em que o usuário tenta acessar um serviço fornecido por um SP e é redirecionado para um IdP, ocorre primeiro um redirecionamento para um serviço

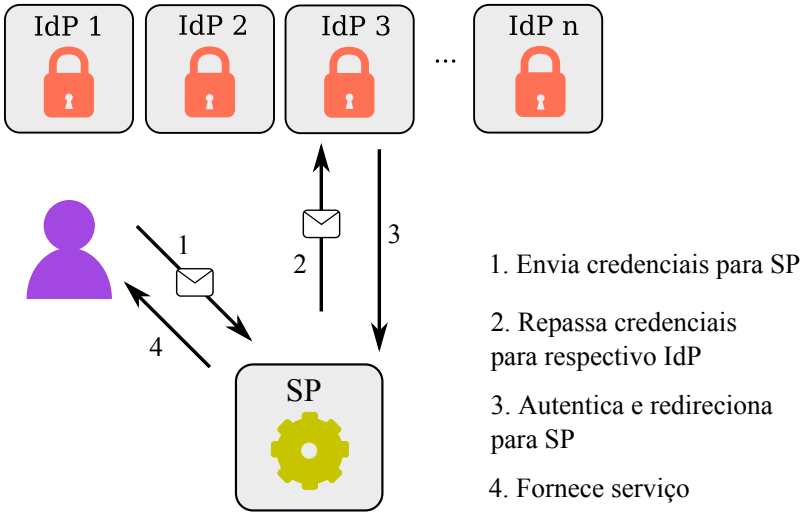

Figura 3.4: Descoberta de IdP através de credenciais.

de *IdP proxy* [10]. Esse serviço lista para o usuário todos os IdPs disponíveis e o usuário pode escolher qual deles deve ser utilizado. Após a escolha, o usuário é redirecionado para o IdP no qual a autenticação deve ser realizada. Ao final da autenticação junto ao IdP, o serviço de *IdP proxy* redireciona o usuário de volta para o SP juntamente com as informações do usuário liberadas pelo IdP. A Figura 3.5 ilustra o procedimento.

O GT-PID utiliza uma implementação para um serviço de *IdP proxy* chamado *Where Are You From* (WAYF) [13]. Na Seção 5.3, a implementação do serviço de WAYF utilizada será abordada em mais detalhes.

3.3 CAFe e CAFe Expresso

A RNP fornece uma federação de identidades para as instituições interessadas. O serviço é oferecido por meio de uma plataforma chamada Comunidade Acadêmica Federada (CAFe) [14]. Através de plataforma CAFe, instituições podem cadastrar seus IdPs e SPs. Depois de cadastrados, os IdPs e SPs passam a servir os usuários da comunidade acadêmica, com a CAFe garantindo a autenticidade e confiabilidade de cada um desses provedores. A CAFe oferece também um serviço de *IdP Proxy* para a descoberta de Provedores de Identidade, contribuindo para o procedimento

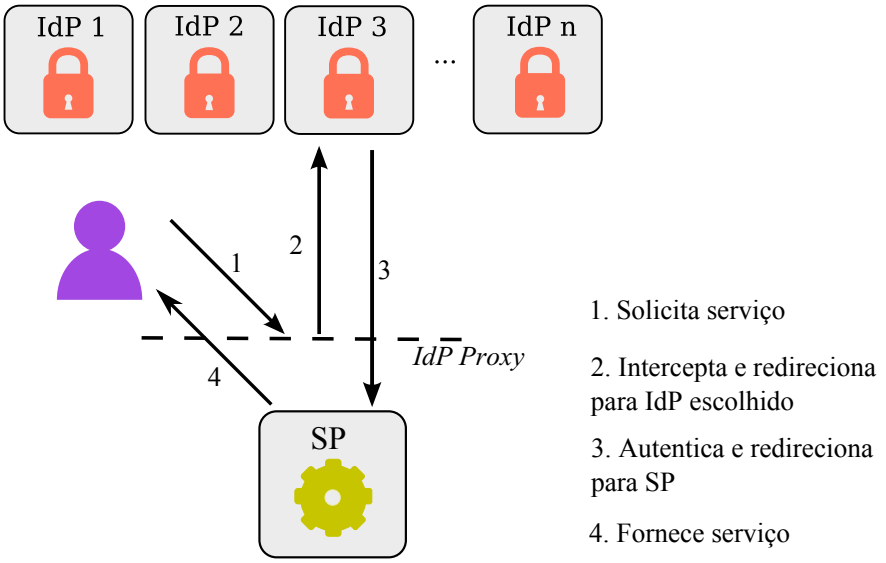

Figura 3.5: Descoberta de IdP através de serviço de WAYF.

de autenticação federada dos usuários.

Como mencionado na Seção 1.2, a RNP é voltada para a criação de infraestrutura de tecnologias de informação para instituições de ensino e pesquisa. Dada a natureza das atividades dessas instituições, novos serviços experimentais são criados com muita frequência. Esses novos serviços podem criar instabilidade nas federações que participam. Para facilitar a experimentação de serviços ligados à CAFe sem trazer instabilidade à sua federação de produção, a RNP oferece também uma federação de identidades experimental, chamada CAFe Expresso [15]. A CAFe Expresso fornece uma infraestrutura de IdPs e SPs para que pesquisadores consigam realizar experimentos tanto de serviços genéricos quanto de gerenciamento federado de identidades. Assim como a CAFe, a CAFe Expresso também possui um serviço para a descoberta de Provedores de Identidade, que é discutido na Seção 5.3.

A CAFe atua como plataforma de gerenciamento federado de identidades para os usuários de instituições de ensino e pesquisa, que são o público-alvo do GT-PID. Naturalmente, deseja-se inserir o GT-PID na federação proporcionada pela CAFe. Porém, para que um serviço seja integrado à CAFe, ele deve primeiro ser integrado à CAFe Expresso. Dessa forma, neste trabalho utiliza-se a plataforma

CAFe Expresso.

Capítulo 4

O Orquestrador de Nuvens OpenStack

O OpenStack é um software de código aberto capaz de criar serviços de computação em nuvem na forma de IaaS. Sua principal função é criar um sistema capaz de gerenciar o fornecimento de grandes volumes físicos de recursos de computação, armazenamento e conectividade de forma virtualizada, de acordo com configurações estabelecidas por administradores e requisitos de usuários [16]. Em outras palavras, o OpenStack é um sistema orquestrador de recursos, capaz de gerenciar a virtualização de máquinas, discos e ferramentas de rede disponíveis em um determinado servidor, para que usuários possam acessá-los como uma plataforma de IaaS.

O OpenStack utiliza a licença Apache 2.0. Essa licença permite a outras pessoas e instituições utilizar, modificar e redistribuir o código do OpenStack. A principal condição para isso é que, em caso de redistribuição, seja claro para quem recebe o código modificado quais foram as modificações inclusas no código [17].

Existe uma entidade oficialmente responsável pelo desenvolvimento, distribuição e adoção do OpenStack, chamada *OpenStack Foundation* (Fundação OpenStack) [18]. A fundação atualmente possui mais de 28 mil membros individuais e é financiada por empresas, que devem contribuir financeiramente para serem membros da fundação. Existem diversas empresas que contribuem de alguma forma para o projeto OpenStack, inclusive empresas brasileiras [19]. Isso pode ser interpretado como um indicativo da grande importância desse software.

A comunidade de desenvolvimento do OpenStack é separada em diversos grupos que trabalham em partes do código e funcionalidades específicas. Existe, por exemplo, um grupo voltado para a segurança, que avalia o impacto na segurança de determinadas mudanças no código e recebe comunicações de possíveis vulnerabilidades. Esses grupos são muito importantes para o desenvolvimento do sistema e como suporte à utilização, implantação e modificação do sistema.

Periodicamente, a comunidade do OpenStack se reúne em um OpenStack Summit, que é uma conferência de cinco dias. Nessas conferências, um calendário é definido para os próximos passos de desenvolvimento do software e são estabelecidos os parâmetros para a próxima versão do OpenStack. Isso significa que existem frequentes atualizações de versão do OpenStack, na mesma frequência em que ocorrem os OpenStack Summits, que são realizados a cada seis meses [20]. Este trabalho utiliza a versão Juno do OpenStack, que foi lançada em outubro de 2014 e é a décima versão do OpenStack [21].

4.1 Arquitetura modular

O OpenStack é dividido em módulos. Esses módulos representam divisões lógicas do ponto de vista das funcionalidades, da arquitetura e do desenvolvimento. Cada módulo fornece serviços a usuários, sistemas externos ou a outros módulos do OpenStack [22]. A divisão em módulos facilita o gerenciamento do sistema como um todo, utilizando a estratégia chamada de "dividir para conquistar". Isso acontece porque os módulos também promovem uma divisão do código do OpenStack como um todo. Dessa maneira, tanto as funcionalidades implementadas quanto o código que implementa essas funcionalidades são divididos em pedaços mais simples de conceber, entender, manter e modificar.

Cada módulo fornece um conjunto de serviços e funcionalidades. Desse modo, a instalação de todos os módulos não é obrigatória, porque alguns serviços são desnecessários para determinadas aplicações. No caso específico do GT-PID, os servidores de VM, os servidores de discos e o controlador recebem conjuntos diferentes de módulos, uma vez que as funções de cada uma dessas máquinas são diferentes. A

arquitetura em módulos deixa aberta uma grande possibilidade de personalizações.

A comunicação entre os módulos pode se dar através de bibliotecas Python ou de chamadas a APIs (*Application Programming Interfaces*). Quando um módulo é instalado para ser o servidor de uma determinada API, é necessário que os outros módulos sejam informados a respeito das URLs (*Uniform Resource Locators*) nas quais as APIs estão sendo oferecidas. Essas URLs são chamadas de pontos de acesso e, quando acessadas através de métodos do protocolo HTTP, respondem com os serviços correspondentes utilizando a arquitetura REST.

As funcionalidades oferecidas por um módulo podem ser diretamente relacionadas ao gerenciamento de recursos fornecidos pela IaaS criada pelo OpenStack ou podem ser funcionalidades periféricas, que possibilitam a utilização avançada do ambiente criado. Por exemplo, uma funcionalidade diretamente relacionada ao gerenciamento de recursos fornecidos pela IaaS é a instanciação de VMs. Por outro lado, uma funcionalidade periférica é a limitação de recursos para usuários. No caso específico deste trabalho, foram alterados o módulo Horizon, responsável pela interface gráfica e o módulo Keystone, que fornece aos outros módulos o serviço de autenticação.

4.1.1 Horizon

A interface gráfica do OpenStack é fornecida pelo módulo Horizon. Esse módulo foi escrito na linguagem Python, utilizando o arcabouço de software Django [23]. Esse arcabouço facilita o desenvolvimento de projetos ao produzir de forma automática o código para funcionalidades comuns em aplicações *web*, como a autenticação de usuários ou a representação de objetos em bancos de dados.

O módulo Horizon funciona como um servidor de páginas web que são acessadas pelos usuários para acessar os outros serviços do OpenStack. O módulo, então, se comunica com os outros módulos do OpenStack para que sirvam os usuários.

É possível realizar algum nível de personalização na aparência das páginas web servidas pelo módulo. A Figura 4.1 mostra a página de autenticação gerada pelo Horizon, modificada com a imagem de exibição do GT-PID.

Figura 4.1: Tela de login do GT-PID antes da implantação de autenticação federada.

Em primeira análise, pode-se dizer que o Horizon tem páginas *web* dinâmicas que recebem informações vindas de outros módulos do OpenStack ou realizam comandos nesses módulos.

Por utilizar o arcabouço Django, o Horizon resolve a autenticação de usuários de forma desacoplada. Do ponto de vista do Horizon, o código gerado pelo arcabouço realiza a autenticação. Porém, em segundo plano, esse código executa chamadas ao Keystone, módulo do OpenStack responsável pela autenticação. O procedimento realizado pelo módulo Keystone é abordado na Seção 4.1.2 e também na Seção 6.2.

4.1.2 Keystone

O Keystone é o módulo responsável pelo gerenciamento de identidade dos usuários. Esse módulo gerencia e controla todas as operações de identificação e autorização de usuários, como também de catálogo das operações permitidas por usuários e dos pontos de acesso das APIs que executam essas operações [22].

4.1.2.1 Conceitos e definições

Para compreender o funcionamento do Keystone, é preciso compreender determinados conceitos, listados nesta seção. Esses conceitos são utilizados mais à frente neste trabalho, quando forem discutidos os serviços oferecidos pelo Keystone. Todos esses conceitos foram retirados do manual de instalação do OpenStack versão Juno [22] ou dos manuais de configuração de autenticação federada no Keystone [24]:

- **Usuário:** O conceito de usuário utilizado pelo OpenStack e pelo Keystone é equivalente ao definido como identidade no Capítulo 3, sendo uma representação interna para entidades existentes no mundo real. Como na documentação oficial a nomenclatura é *user*, ela será usada ao longo do trabalho.

- **Projeto:** O conceito de projeto é utilizado para criar uma separação lógica entre recursos utilizados por diferentes usuários ou grupos de usuários. No OpenStack é chamado em inglês de *tenant* ou, em outros casos, de *project*, dependendo da versão de interface de software instalada.

- **Ficha de acesso:** É uma sequência de bits assinada pelo Keystone que é entregue a um usuário real após uma autenticação de sucesso. Funciona como uma confirmação de que o portador foi autenticado. Em inglês, esse conceito é denominado *token*.

- **Provedor de Identidade:** Uma representação interna dos IdPs externos. Se faz necessário para que o OpenStack seja capaz de dar tratamentos diferentes a IdPs diferentes. É chamado de *identity_provider* pelo sistema OpenStack.

- **Mapeamento:** Pode ser um procedimento que encontra uma equivalência entre usuários externos e usuários internos ao Keystone, como também o nome da estrutura de dados que guarda informações para possibilitar tal procedimento. Originalmente chamada de *mapping*.

4.1.2.2 Autenticação

Com o Keystone, existem várias formas diferentes de autenticar um usuário. Uma delas é através do uso de credenciais: um usuário previamente cadastrado entrega suas credenciais para o Keystone, que verifica em seu banco de dados se aquelas

credenciais são válidas. Em caso afirmativo, o Keystone emite uma ficha de acesso (*token*) para o usuário. Uma ficha de acesso, enquanto for válida, serve para o usuário portador utilizar como uma garantia de que o mesmo já foi autenticado pelo Keystone. Nesse caso, a autenticação por meio de credenciais segue o modelo isolado de autenticação [10], porque o OpenStack oferece um serviço e realiza a autenticação de seus próprios usuários simultaneamente. Quando um usuário solicita um serviço a um módulo do OpenStack, ele envia ao módulo sua ficha de acesso junto à solicitação. Dessa forma, o usuário pode confirmar ao módulo que já foi autenticado pelo Keystone. Para garantir que usuários mal-intencionados não utilizem fichas de acesso inválidas, os módulos verificam junto ao Keystone a validade das fichas apresentadas por usuários a cada nova solicitação.

A ficha de acesso gerada pelo Keystone após uma autenticação ainda não permite ao usuário o acesso à maioria dos serviços oferecidos pelo OpenStack. Como dito na Seção 4.1.2.1, as operações relativas a IaaS do OpenStack, como alocar e utilizar máquinas virtuais e discos virtuais, são oferecidas com base em projetos. Para que o OpenStack saiba a qual projeto vincular as operações que um usuário executa, a ficha de acesso utilizada para executar tais operações precisa estar ligada a algum projeto. A ficha de acesso gerada inicialmente pelo procedimento de autenticação não está vinculada a nenhum projeto. Por isso, essa ficha de acesso é dita uma ficha de acesso sem escopo ou, na nomenclatura original do OpenStack, *unscoped token*. Para que o usuário tenha acesso aos serviços relativos a IaaS que tem permissão de acessar, o usuário deve obter uma ficha de acesso que esteja vinculada a algum projeto, denominada ficha de acesso com escopo (em inglês, *scoped token*).

Utilizando a ficha de acesso sem escopo obtida no procedimento de autenticação por credenciais, o usuário pode obter do Keystone uma lista de projetos aos quais possui permissão de acesso. Então, esse usuário pode realizar uma nova chamada ao Keystone requisitando a emissão de uma ficha de acesso com escopo, que esteja vinculada a um dos projetos contidos na lista obtida anteriormente. Logicamente, o usuário deve utilizar a ficha de acesso sem escopo para realizar esse pedido, garantindo ao Keystone sua identidade sem a necessidade de uma nova autenticação. O procedimento é ilustrado na Figura 4.2 Lembrando das operações discutidas no

Capítulo 3, é interessante notar que a ficha de acesso sem escopo e a ficha de acesso com escopo são os resultados das operações de autenticação e autorização, respectivamente.

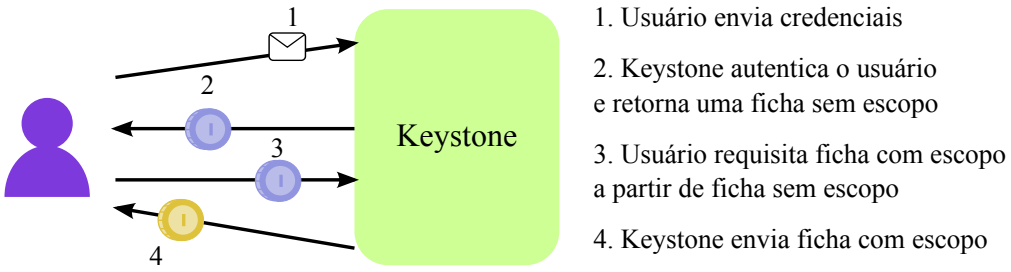

Figura 4.2: Autenticação e autorização de usuário junto ao Keystone.

Para que um usuário consiga acessar um serviço do OpenStack ligado a algum projeto, o usuário deve informar ao serviço uma ficha de acesso com escopo para esse projeto. O objetivo dessa ficha de acesso é garantir ao serviço que o usuário foi autenticado e autorizado para esse projeto. O serviço envia a ficha de acesso ao Keystone para que a validade dessa ficha seja avaliada. O Keystone executa duas verificações. Primeiro, confere se a ficha de acesso é válida e se existe registro da expedição de tal ficha de acesso; segundo, verifica se o escopo da ficha de acesso confere ao usuário a permissão de acesso ao serviço requisitado. Uma vez que ocorra a validação da ficha de acesso, o módulo poderá fornecer o serviço ao usuário. O procedimento é ilustrado na Figura 4.3.

No caso de um usuário utilizar a interface gráfica para requisitar sua autenticação, o procedimento de autorização é transparente. O usuário insere suas credenciais na interface oferecida e depois é conduzido para uma página contendo as operações que são permitidas para seu projeto principal. Também há a opção de alteração de projeto, se possível. Não há contato do usuário com os diferentes tipos de ficha de acesso e nem necessidade de procedimentos relativos à autorização por parte do usuário. O módulo Horizon isola o usuário de tais procedimentos.

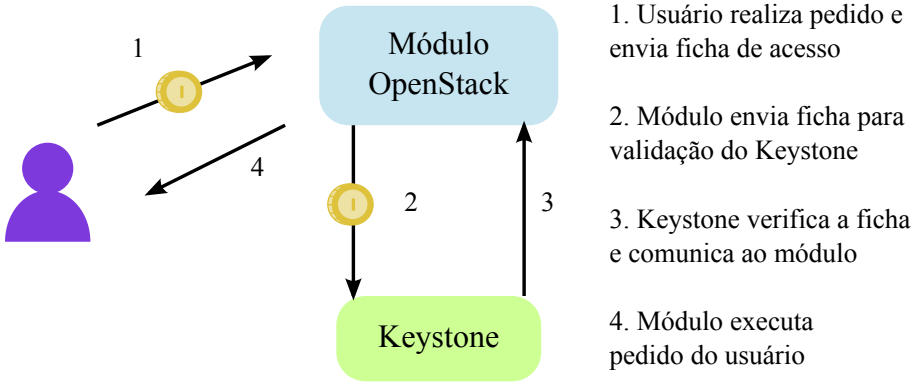

Figura 4.3: Pedido de um usuário a um serviço.

4.2 Ambiente de desenvolvimento

Existe para o OpenStack um ambiente previamente configurado para desenvolvimento, denominado DevStack [25]. A implantação realizada neste trabalho utiliza o DevStack, pois esse ambiente possui instalação e configuração simplificadas, em relação ao ambiente de produção do OpenStack. Isso possibilita maior agilidade no processo de desenvolvimento através do isolamento de problemas relevantes para instâncias OpenStack de produção, mas desprezíveis em um ambiente de desenvolvimento.

Com o DevStack, é possível em alguns minutos instalar ou reinstalar completamente um ambiente de desenvolvimento do OpenStack que esteja sendo executado em algum computador. Além disso, através de um arquivo de configuração unificado, é possível manter um rigoroso controle das alterações nas configurações de cada módulo do OpenStack.

4.3 Suporte à autenticação federada

A versão Juno do OpenStack, utilizada no presente trabalho, suporta algumas funções de autenticação federada. O módulo Keystone, e apenas ele, possui funcionalidades capazes de configurá-lo como um SP ou como um IdP. Quando configurado como um SP, o Keystone consegue aceitar a autenticação federada provida por um

ou mais IdPs externos; quando configurado como um IdP, o Keystone pode autenticar usuários a pedido de outras instâncias do Keystone [24]. O GT-PID se comporta como um SP inserido em uma federação, logo, a instância interna do Keystone deve estar configurada como um SP.

Quando configurado como um SP, o módulo Keystone é capaz de gerar uma ficha de acesso sem escopo para um usuário autenticado por algum IdP. Como visto na Seção 3.1, para que haja autenticação federada, um SP deve ser capaz de confiar em um ou mais IdPs para que esses IdPs autentiquem os usuários que utilizarão os serviços oferecidos por esse SP. Dessa forma, um dos pré-requisitos é que o IdP da autenticação do usuário seja conhecido pelo sistema e que tenha a confiança do Keystone. Para tal, o IdP precisa ser registrado como um *identity_provider* no Keystone. É necessário que haja também algum arcabouço de software que realize a comunicação entre o Keystone e o IdP. Esse arcabouço deve ser capaz de enviar o usuário para autenticação pelo IdP e depois retornar o usuário para o Keystone, juntamente com outras informações de identidade oriundas do IdP. O Capítulo 5 entra em mais detalhes sobre o arcabouço utilizado neste trabalho.

Outro pré-requisito para que haja a autenticação federada de um usuário é que a identidade do usuário já exista na base de dados do Keystone. Utilizando os conceitos apresentados na Seção 4.1.2.1, um *user* representando o usuário real autenticado já deve existir na base de dados do Keystone. Esse fator é necessário porque o serviço proporcionado pelo OpenStack requer algum nível de persistência: um usuário realiza operações no serviço e deseja que os resultados dessas operações sejam acessíveis quando o mesmo usuário voltar a acessar o serviço.

O Keystone aceita a autenticação vinda de um IdP de confiança, mas realiza localmente a autorização do usuário. A estrutura de *mapping* define regras para que um usuário que realizou autenticação federada seja identificado como um *user* existente na base de dados e receba as permissões devidas. Para isso, é necessário que o IdP utilizado na autenticação do usuário envie algum tipo de informação ao Keystone que seja capaz de identificar de forma única um objeto *user* da base de dados local.

Na versão Juno do OpenStack, não existe suporte à autenticação federada de usuários através da interface gráfica. Um usuário precisa utilizar comandos através das APIs para obter uma ficha de acesso sem escopo, transformá-la em uma ficha de acesso com escopo e depois continuar usando os serviços do OpenStack através de chamadas de API. Durante todo o procedimento de autenticação federada suportada pelo Keystone, a única interface gráfica gerada pelo GT-PID para o usuário é a visualização da ficha de acesso. Ainda assim, na implantação deste trabalho, será utilizado um navegador para a realização da implantação da autenticação federada suportada pelo Keystone. Isso ocorre porque essa autenticação exige a utilização das APIs através de métodos HTTP, que são suportados pelos navegadores, e também porque os procedimentos de autenticação dos IdPs utilizados devem ser realizados através de páginas web, tornando navegadores um meio apropriado para a autenticação.

O serviço de autenticação federada proporcionado pelo Keystone é oferecido na forma de API, em URLs que devem ser protegidas pelo arcabouço que realiza a comunicação entre o Keystone e o IdP escolhido para autenticação. Para cada IdP disponível para a autenticação há uma URL diferente, fornecida pelo Keystone.

O procedimento completo de autenticação encontra-se na Figura 4.4. Inicialmente, o usuário tenta acessar a URL do serviço de autenticação federada do Keystone, mas é interceptado pelo arcabouço (1). O arcabouço de comunicação entre Keystone e IdP redireciona o usuário para o IdP escolhido para autenticação (2). O usuário realiza sua autenticação junto ao IdP (3) e o IdP redireciona o usuário e retorna seus atributos para o arcabouço (4), que insere esses atributos no Keystone (5). O Keystone utiliza os atributos para mapear o usuário reconhecido pelo IdP em um usuário interno (6) e cria uma ficha de acesso sem escopo para esse usuário (7). Ao final, o Keystone exibe para o usuário a ficha de acesso criada.

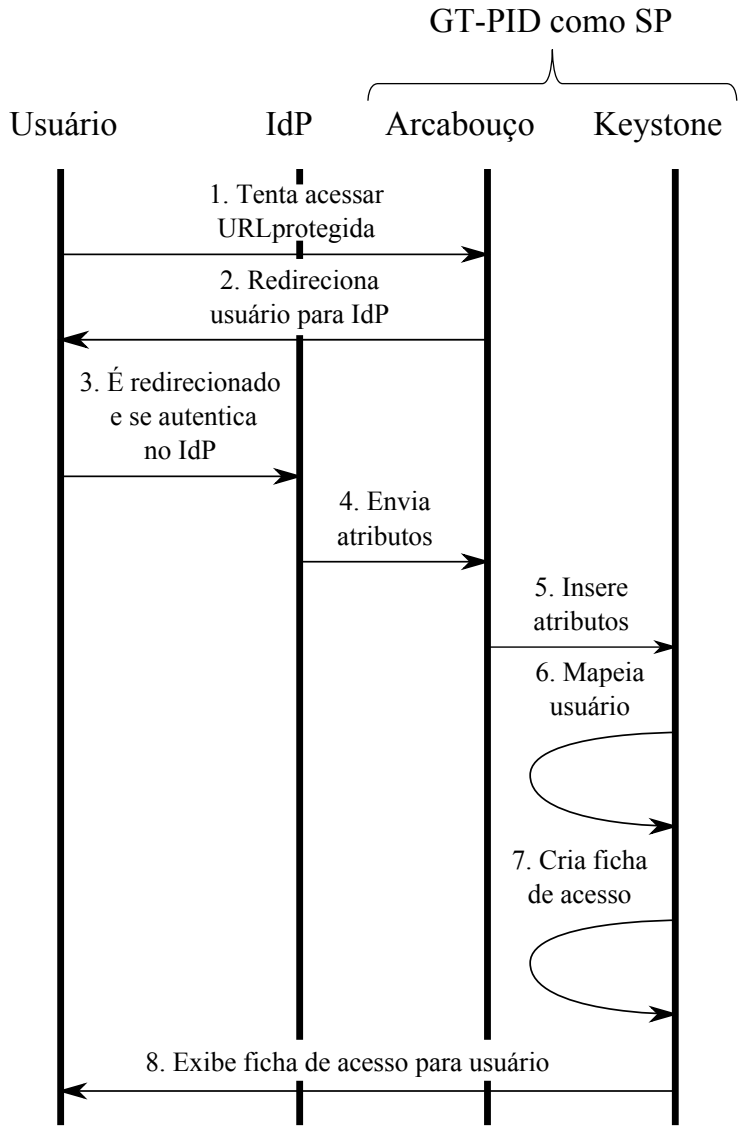

Figura 4.4: Fluxo de autenticação federada suportado pela versão Juno do OpenStack.

Para uma experiência completa de usuário, é necessário no GT-PID que um usuário possa realizar um procedimento de autenticação federada utilizando a interface gráfica, sem que seja necessário acessar outras interfaces do sistema. Neste projeto, existem configurações e modificações nos códigos dos módulos Horizon e Keystone que possibilitam esse cenário. O Capítulo 6 relata a configuração da autenticação federada suportada e a subsequente implementação da interface gráfica para autenticação federada.

Para a troca de mensagens entre Keystone e IdP, existem vários arcabouços de software suportados pelo Keystone. No GT-PID utiliza-se o Shibboleth [7], recomendado pela CAFe e pela CAFe Expresso, que são federações nas quais o GT-PID está inserido. Mais informações sobre o Shibboleth estão no Capítulo 5.

Capítulo 5

O Arcabouço de Software Shibboleth

Quando um usuário deseja utilizar um serviço fornecido por um SP em uma federação, é necessário que esse usuário se autentique em um IdP pertencente a essa federação. Porém, é necessário também que o SP seja informado de que a autenticação ocorreu e, em alguns casos, o SP precisa também de atributos do usuário autenticado. O Shibboleth é um arcabouço de software que proporciona essa comunicação entre SPs e IdPs [7].

O Shibboleth deve estar instalado e configurado tanto nos SPs como também nos IdPs de uma federação. Quando um usuário não autenticado tenta acessar um serviço oferecido por um SP, o Shibboleth instanciado no SP redireciona o usuário para um IdP configurado. Esse IdP realiza o procedimento de autenticação e o Shibboleth instanciado no IdP redireciona o usuário para o SP novamente, enviando para o SP original uma mensagem contendo atributos previamente configurados nas instâncias do Shibboleth do SP e do IdP.

O Shibboleth possui versões diferentes para utilização nos IdPs e nos SPs. O GT-PID possui instalada a versão SP do Shibboleth, uma vez que o GT-PID é modelado como um serviço oferecido em uma federação e os IdPs, como mencionado na Seção 3.3, são fornecidos por um serviço externo. A infraestrutura da CAFe Expresso fornece ao GT-PID os IdPs para autenticação de seus usuários. As instâncias de Shibboleth executadas pelos IdPs pertencentes à CAFe Expresso estão fora do

escopo deste trabalho, uma vez observadas as configurações locais necessárias para a comunicação com os mesmos.

5.1 O padrão SAML2

Para que uma comunicação aconteça, é necessário que haja uma linguagem entendida e acordada por todas as partes envolvidas em tal comunicação. O Shibboleth utiliza a linguagem *Security Assertion Markup Language 2.0* (SAML2) [26] para a comunicação entre seus pares. O SAML2 é uma linguagem aberta, desenvolvida pela OASIS (*Organization for the Advancement of Structured Information Standards*), com o objetivo de padronizar comunicações relativas à identificação de usuários. Ela está definido desde 2005 e possui muitas implementações disponíveis.

O padrão SAML2 funciona baseado na troca de mensagens que representam asserções a respeito de autenticações, atributos ou autorizações de determinados indivíduos. Uma asserção é uma mensagem que contém dados de identidade ou pedidos para a troca desses dados. Para que a comunicação funcione, é necessário que haja alguma relação de confiança entre as entidades que trocam as mensagens [27]. Esse problema é resolvido no Shibboleth através de algumas configurações, que são descritas a seguir.

5.2 Metadados, atributos e outras configurações

Para que duas instâncias diferentes do Shibboleth consigam se comunicar, é necessário que haja uma relação de confiança e o estabelecimento de um canal seguro entre essas instâncias. Além disso, o padrão SAML2 também exige o estabelecimento de identificadores, pontos de acesso e certificados entre as entidades de uma federação [28]. Cada entidade que participa de uma federação deve, então, produzir um arquivo de metadados sobre si mesma. Esses metadados são um registro das configurações daquela entidade e carregam as informações que devem ser previamente conhecidas por uma outra entidade para que elas possam se comunicar [28]. Um SP consome metadados a respeito de IdPs e um IdP consome metadados a respeito de SPs. Isso significa que cada provedor deve ter acesso ao arquivo de metadados refe-

rente às entidades que executam o papel oposto ao seu em uma federação [29]. Na implementação feita pelo Shibboleth, existem duas formas possíveis de ter acesso aos metadados de uma outra entidade: em uma, é necessário possuir o arquivo da outra entidade na máquina onde o Shibboleth está sendo executado; na outra, o arquivo é disponibilizado pela outra entidade em uma URL e, a intervalos configurados, a instância Shibboleth local busca o arquivo de metadados da instância Shibboleth externa.

O GT-PID é modelado como um SP na federação CAFe Expresso. Portanto, os IdPs da CAFe Expresso devem ter acesso ao arquivo de metadados do Shibboleth instalado no GT-PID. Além disso, o Shibboleth instalado no GT-PID deve ter acesso aos arquivos de metadados do Shibboleth dos IdPs da CAFe Expresso. Devido às possibilidades implementadas pelo Shibboleth, o GT-PID acessa os metadados da CAFe Expresso disponibilizadas em uma URL.

5.3 Where Are You From

A Seção 3.2 descreve a necessidade de serviços de descoberta de IdPs para a realização de procedimentos federados de autenticação. Um protocolo de descoberta de provedores de identidade suportado pelo Shibboleth é o *Where Are You From* (WAYF) [27]. Depois de solicitar um procedimento de autenticação federada para um determinado serviço, o usuário é redirecionado para o serviço de WAYF. O serviço de WAYF lista ao usuário todos os IdPs capazes de realizar autenticação para aquele serviço. O usuário então escolhe qual IdP deve prosseguir com a autenticação. O serviço WAYF redireciona o usuário para o IdP que irá realizar a autenticação e, a partir desse momento, atua de forma transparente, redirecionando o usuário para o SP após sua autenticação pelo IdP e repassando as mensagens do IdP para o SP.

O WAYF é utilizado pela CAFe e pela CAFe Expresso e foi utilizado na implantação relatada neste trabalho. O próximo capítulo fornece mais detalhes da implantação.

Capítulo 6

Implantação e Testes

Como afirmado na Seção 4.3, a versão do OpenStack utilizada já possui um suporte limitado à autenticação federada, que permite o reconhecimento da autenticação federada pelo Keystone, mas não permite ao usuário autenticado acesso à interface gráfica. A interface gráfica é implementada pelo módulo Horizon, que confia no gerenciamento de identidades provido pelo módulo Keystone. Isso faz da implantação da autenticação federada sem interface gráfica uma dependência para a autenticação federada com interface gráfica. É adotada, então, uma divisão do procedimento de implantação em duas partes: a primeira, a configuração da autenticação federada sem interface gráfica e a segunda, a implementação da interface gráfica.

6.1 Autenticação federada sem interface gráfica

A versão Juno do OpenStack possui suporte para inserção do Keystone em uma federação, como relatado na Seção 4.3. Dessa forma, implantar a autenticação federada sem a utilização da interface gráfica é uma questão de configuração. Ainda assim, é uma configuração que requer planejamento, pois existe uma grande quantidade de parâmetros a serem definidos.

O objetivo em criar um planejamento para a implantação é diminuir o tempo gasto com a implantação, através da identificação de dependências e de fatores críticos. Além disso, o planejamento utilizado divide o trabalho em etapas que possuem resultados observáveis, criando pontos de avaliação do trabalho realizado em cada

etapa. A identificação de dependências é utilizada para garantir que as tarefas sejam executadas em uma sequência eficiente. A identificação de fatores críticos tem como principal ponto a eliminação ou, pelo menos, a atenuação de possíveis gargalos. Por fim, a criação de pontos de avaliação do trabalho serve para solucionar rapidamente erros de execução nos procedimentos, a fim de evitar uma propagação de tais erros. Tal processo de desenvolvimento é uma modificação do processo proposto no tutorial de inclusão do Keystone em uma federação fornecido pelo OpenStack [24].

Um dos fatores críticos que podem ser observados é que a federação CAFe Expresso é gerenciada por uma equipe externa ao GT-PID, incluindo as operações de inclusão de novos SPs. Isso significa que a execução de quaisquer testes junto à CAFe Expresso precisa ser requisitada a pessoas que não fazem parte do GT-PID. A possível demora nos contatos com essas pessoas pode aumentar em muito o tempo para a finalização da implementação. Para agilizar testes de configuração da autenticação federada e reduzir as chances de um grande número de contatos desnecessários com a equipe responsável pelo gerenciamento da CAFe Expresso, é utilizada uma plataforma de testes para autenticação federada com o Shibboleth. Essa plataforma chama-se TestShib [30]. Ao contrário da CAFe Expresso, o TestShib não possui um serviço de WAYF e disponibiliza apenas um IdP para testes. Esse fator por um lado é positivo, pois torna os testes mais simples e, por outro lado, é negativo, pois torna o ambiente de testes menos próximo do ambiente real da CAFe Expresso.

O procedimento definido para a configuração da autenticação federada no Keystone é uma adaptação e instanciação do processo definido no manual de instruções de inclusão do Keystone em uma federação [24]:

1. Instalação e configuração de uma instância de desenvolvimento do OpenStack através do DevStack em um único servidor, como Controlador da arquitetura GT-PID;

2. Instalação local do Shibboleth no controlador e configuração de chaves e certificados;

3. Configuração do IdP fornecido pelo TestShib no Shibboleth local;

4. Geração de metadados do Shibboleth local, criando conceitualmente um SP;
5. Configuração do SP recém-criado no IdP fornecido pelo TestShib;
6. Configuração, no SP, da leitura dos atributos enviados pelo IdP fornecido pelo TestShib;
7. Migração do banco de dados do Keystone para a versão federada;
8. Criação de objetos no Keystone para a realização de autenticação federada junto ao TestShib;
9. Teste do funcionamento da autenticação junto ao IdP fornecido pelo TestShib;
10. Configuração do Shibboleth local para comunicação com os IdPs da CAFe Expresso;
11. Inclusão do SP GT-PID na federação CAFe Expresso através do envio dos metadados à equipe de gerenciamento da CAFe Expresso;
12. Criação de objetos no Keystone para a autenticação federada junto à CAFe Expresso;
13. Teste de funcionamento da autenticação junto aos IdPs da CAFe Expresso.

Em concordância com as preocupações mencionadas anteriormente neste texto, essa sequência específica observa a questão das dependências entre as etapas: em quase todos os casos, é necessário que todas as etapas anteriores tenham sido concluídas para que a etapa seguinte possa ser executada [24]. Adicionalmente, para cada etapa é possível realizar um teste que garante que a etapa foi concluída com sucesso. O objetivo é criar uma situação em que pode-se assumir que falhas em uma determinada etapa são oriundas de erros cometidos naquela mesma etapa, e não em etapas anteriores.

A conclusão dos testes de funcionamento da autenticação junto aos IdPs da CAFe Expresso garante que o Keystone esteja inserido na federação CAFe Expresso. A implantação é considerada concluída quando um usuário é capaz de realizar com sucesso o procedimento apresentado na Figura 6.1. Inicialmente, o usuário tenta,

através de um navegador, acessar uma URL do GT-PID que é ponto de acesso para um serviço do Keystone capaz de emitir fichas de acesso (1). Essa URL é protegida pelo Shibboleth, logo, o usuário é interceptado e redirecionado para o serviço de WAYF da CAFe Expresso (2). Em seguida, o usuário escolhe um dos IdPs listados pelo serviço de WAYF (3), numa tela que pode ser observada na Figura 6.2. Após a escolha, o usuário é redirecionado para esse IdP (4), junto ao qual executa o procedimento do IdP para autenticação (5). A tela de autenticação de um dos IdPs da CAFe Expresso pode ser vista na Figura 6.3. O IdP então envia atributos do usuário para o Shibboleth instalado no GT-PID (6) e esses atributos são inseridos no Keystone (7). O Keystone realiza o mapeamento dos atributos enviados em um usuário local (8) e cria uma ficha de acesso sem escopo (9) para esse usuário. Finalmente, o Keystone exibe a ficha de acesso para o usuário (10), concluindo o procedimento. A ficha é exibida ao usuário através do navegador usado no início do procedimento.

É possível comparar a Figura 6.1 com a Figura 4.4 e perceber que no GT-PID existe a participação de um serviço de WAYF que não está presente na descrição da sequência de autenticação federada originalmente suportada pelo Keystone. Tal participação é possível porque o Shibboleth isola o Keystone do serviço de WAYF. As mensagens são trocadas entre Keystone e IdP de forma dependente do Shibboleth, que executa todas as operações necessárias para a adição de um serviço WAYF. A tela do serviço de WAYF utilizado se encontra na Figura 6.2.

É interessante observar que a ficha de acesso gerada ao final do procedimento não está relacionada a nenhum projeto, ou seja, é uma ficha de acesso sem escopo. Um usuário que queira utilizar alguma função de IaaS proporcionada pelo OpenStack deve primeiro vincular a ficha de acesso a algum projeto, obtendo uma ficha de acesso com escopo. Somente depois desse procedimento é possível utilizar a ficha de acesso para acessar os serviços de IaaS dos respectivos módulos do OpenStack, através de APIs [24].

Para que o módulo de interface gráfica Horizon seja capaz de reconhecer a ficha de acesso emitida pelo Keystone, é preciso que o código original da versão Juno do

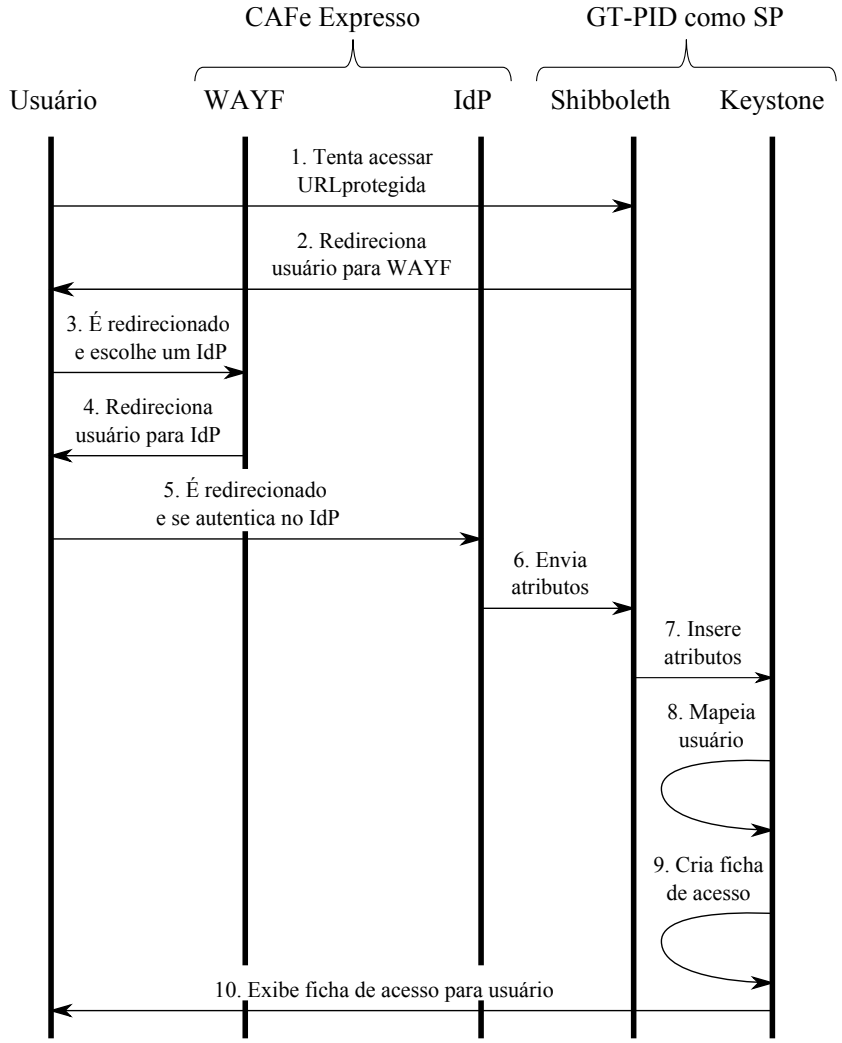

Figura 6.1: Procedimento para autenticação federada sem interface gráfica.

OpenStack seja alterado. A Seção 6.2 explica o funcionamento do GT-PID para que o módulo Horizon seja capaz de reconhecer usuários autenticados pelo procedimento de autenticação federada.

6.2 Autenticação federada com interface gráfica

A Seção 4.1.2 explica o funcionamento da autenticação isolada para um usuário utilizando a interface gráfica. Em uma análise de mais baixo nível, existe um código fornecido pela plataforma Django [23] que é responsável pela obtenção e validação das fichas de acesso. O Django fornece uma abstração de autenticação para o Ho-

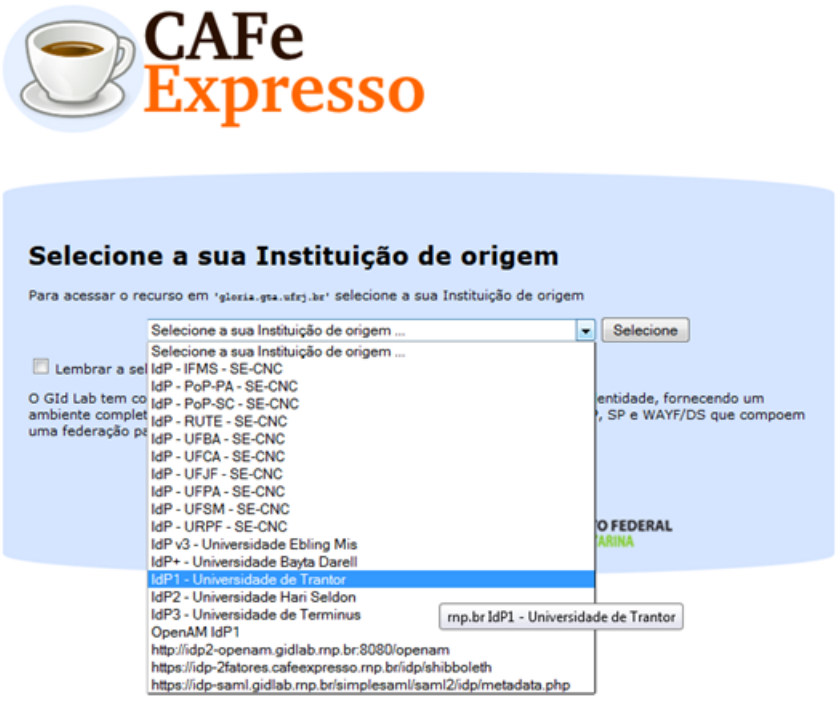

Figura 6.2: Interface de serviço de WAYF.

rizon que realiza chamadas ao Keystone. O código da plataforma Django recebe as credenciais do usuário e utiliza o Keystone para obter uma ficha de acesso com escopo. Além disso, a cada vez que um usuário executa algum pedido ao Horizon, existe um código da plataforma Django que intercepta esse pedido e valida a ficha de acesso junto ao Keystone. O usuário é tão isolado dos procedimentos internos de autenticação que até mesmo as páginas web geradas pelo Horizon não possuem nenhum tipo de ficha de acesso.

A partir da análise do código relativo ao procedimento de autenticação centralizada no módulo Horizon, pode-se perceber que o Horizon armazena a ficha de acesso com escopo obtida no procedimento de autenticação, executado pelo Django, em uma estrutura interna de representação de usuários chamada *User*, juntamente com outras informações do usuário autenticado. O código exige que a criação de um objeto *User* esteja atrelada a uma ficha de acesso. É possível ao Horizon acessar serviços de outros módulos quando um objeto do tipo *User* é instanciado, porque a

Figura 6.3: Tela de autenticação de IdP.

ficha de acesso armazenada no objeto pode ser utilizada na execução das chamadas a esses serviços. Após sua instanciação, o objeto *User* é utilizado pelo Horizon na criação de cada página web dinâmica que será exibida para um usuário, validando junto ao Keystone a ficha de acesso em cada criação. A validação é realizada através de uma abstração oferecida pelo Django, o que significa que essa abstração deve ter acesso ao objeto *User* uma vez que o objeto tenha sido instanciado.

Para que a abstração provida pelo Django funcione em seu modo normal de operação, o módulo Horizon precisa receber as credenciais do usuário e usá-las para realizar sua autenticação junto ao Keystone local. Esse procedimento não é compatível com o modelo federado de autenticação, já que a autenticação federada deve ser realizada por algum IdP, e não por um componente interno ao SP. Para a implantação do modelo federado de autenticação, é desejável que o usuário consiga ser autenticado sem a participação da abstração, mas que a ficha de acesso gerada pela autenticação esteja disponível para cada validação. Ou seja, o procedimento federado obtém uma ficha de acesso com escopo e instancia um objeto do tipo *User* utilizando essa ficha.

Recapitulando a Seção 6.1, ao final da autenticação federada, oferecida pelo Keystone, é gerada e exibida ao usuário uma ficha de acesso sem escopo. A solução implementada neste trabalho consiste no Horizon receber essa ficha de acesso sem escopo e realizar chamadas ao Keystone a fim de obter uma ficha de acesso com escopo. Após isso, um objeto do tipo *User* é criado a partir dessa ficha de acesso com escopo e a página inicial do usuário é exibida. Para tal, o Keystone precisa ser capaz de enviar a ficha sem escopo para o Horizon, que deve, por sua vez, ser capaz de receber essa ficha. Como as arquiteturas do Horizon e do Keystone facilitam a interação entre módulos através do protocolo HTTP, é utilizado um POST HTTP para o envio da ficha de acesso sem escopo do Keystone para o Horizon.

Quando um usuário chega à última etapa do procedimento de autenticação federada relatado na Seção 6.1, o módulo Keystone, ao invés de exibir ao usuário uma página com a ficha de acesso sem escopo, redireciona o usuário para o Horizon ao mesmo tempo em que envia para o Horizon a ficha de acesso sem escopo. Esse comportamento por parte do Keystone é obtido através da alteração da página de exibição da ficha de acesso sem escopo. No código implementado neste trabalho, a ficha é transformada em um formulário HTML de submissão automática. Quando a página contendo o formulário é carregada pelo navegador do usuário, o formulário é automaticamente submetido ao Horizon.

O Horizon deve estar preparado para receber um POST contendo uma ficha de acesso sem escopo e, no método para tratamento desse POST, utilizar essa ficha de acesso para obter junto ao Keystone uma ficha de acesso com escopo. Depois disso, o Horizon cria uma instância do objeto *User* utilizando a ficha de acesso com escopo e exibe a página inicial do usuário como resposta ao POST recebido.

Para que a experiência do usuário seja completa, e o mesmo sinta que a autenticação federada está completamente integrada à interface gráfica fornecida pelo Horizon, um botão é adicionado à página de autenticação original gerada pelo Horizon. Esse botão é um link para a URL de autenticação federada fornecida pelo Keystone e protegida pelo Shibboleth. A tela de autenticação implementada pode ser observada na Figura 6.5, mais adiante. Quando pressionado pelo usuário, esse

botão o leva para o início da sequência de autenticação federada.

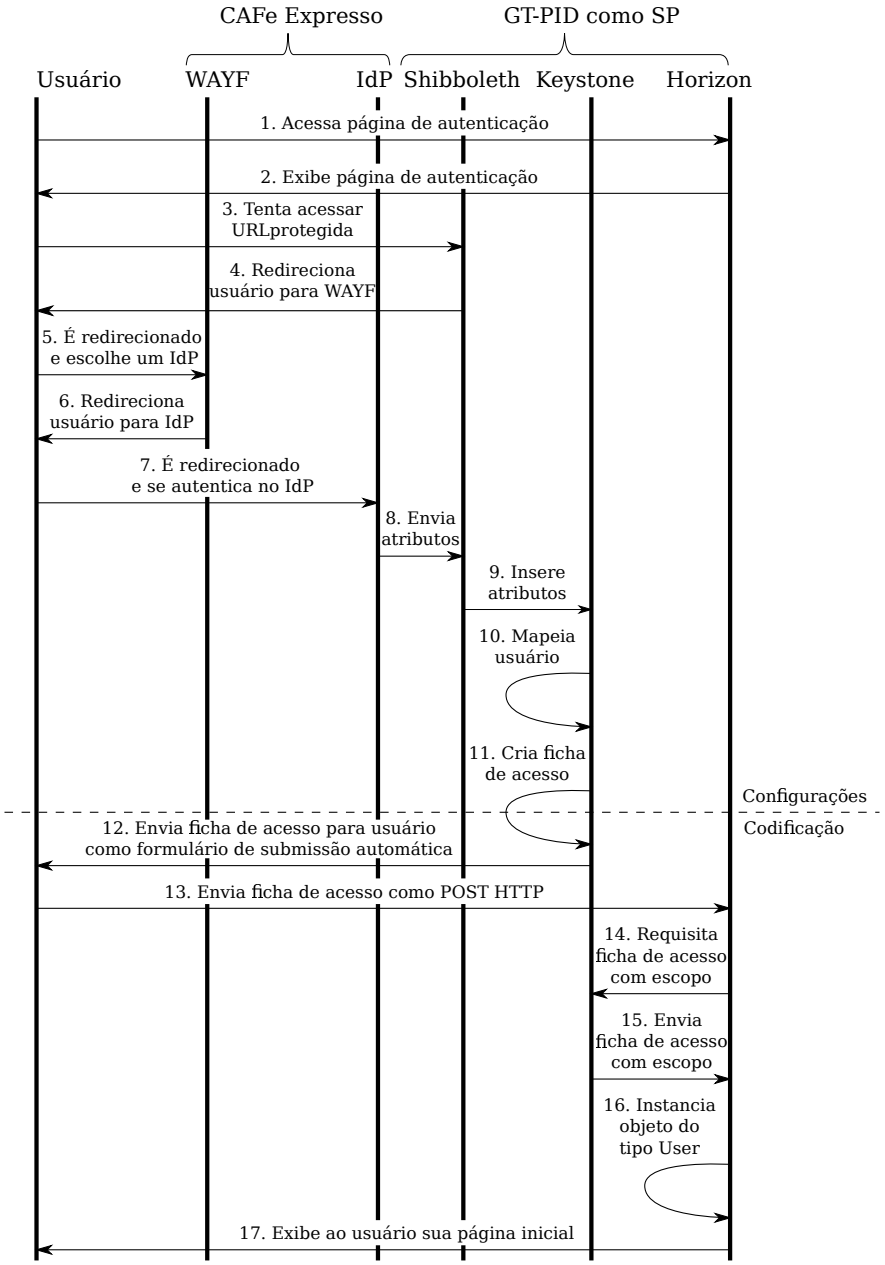

Figura 6.4: Procedimento para autenticação federada com interface gráfica.

A versão final completa do algoritmo é mostrada na Figura 6.4. Comparando esta figura com a Figura 6.1, pode-se ver que existem um primeiro e um segundo passos na versão do procedimento com interface gráfica que são inexistentes na versão sem interface gráfica. A tela de autenticação com botão para autenticação federada pode ser vista na Figura 6.5. Depois, os próximos nove passos são idênticos. A partir daí, o Keystone, ao invés de exibir a ficha de acesso sem escopo para o usuário, a

envia como um formulário HTML de submissão automática (12), que é submetido pelo navegador do usuário diretamente para uma URL controlada pelo Horizon (13). O Horizon recebe a ficha de acesso sem escopo e a utiliza para receber uma ficha de acesso com escopo (14, 15). De posse da ficha de acesso com escopo, o Horizon consegue instanciar um objeto do tipo *User* (16) e exibe ao usuário sua página inicial (17). A Figura 6.6 mostra a página inicial do usuário no GT-PID, gerada e exibida na última etapa do procedimento de autenticação federada. Esse procedimento garante que o Horizon será capaz de funcionar exatamente da mesma forma que funcionaria caso o usuário executasse um procedimento de autenticação isolado.

Figura 6.5: Tela de autenticação do GT-PID, com botão para autenticação através da CAFe Expresso.

Em termos de sequência de implantação, uma possibilidade é primeiro implantar a autenticação federada sem interface gráfica como descrito na Seção 6.1 e depois implementar o código de forma a seguir a própria sequência do algoritmo. Ao

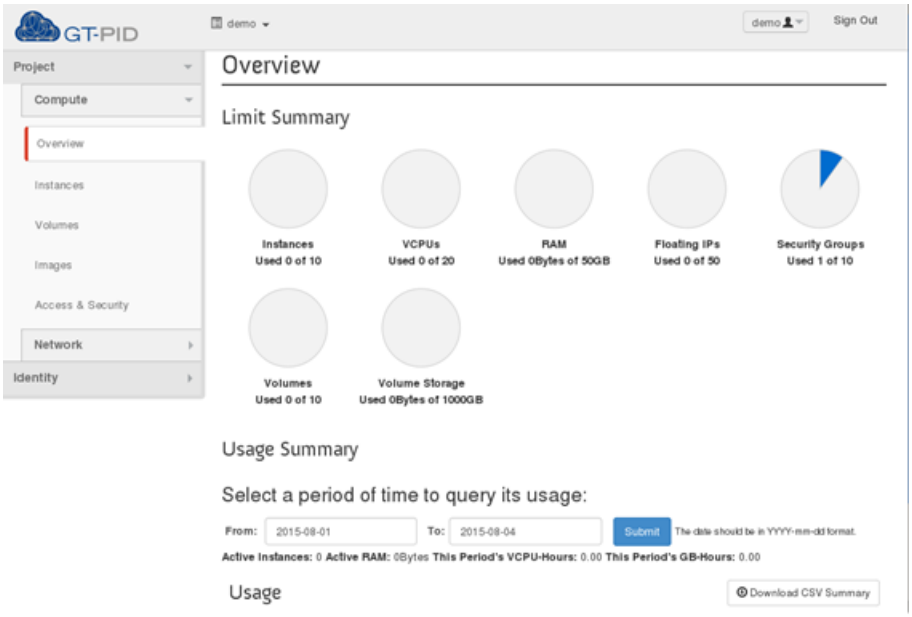

Figura 6.6: Tela inicial do usuário após autenticação federada.

final da implementação de cada etapa do algoritmo, é possível realizar testes que confirmem o sucesso da implementação da etapa, impedindo a propagação de erros para a implementação de etapas seguintes. O êxito dos testes com a última etapa do algoritmo confirma a implantação e é verificável através de uma autenticação de sucesso.

6.3 Testes

Cada fase da implantação possui resultados claros que podem ser facilmente observados. A observação de cada um desses resultados confirma a execução correta da etapa contemplada. Após a implantação completa, é desejável garantir que um usuário qualquer familiarizado com a utilização de autenticação em outros sistemas seja capaz de realizar o procedimento implantado.

Quando a implantação foi concluída, possíveis usuários do sistema foram convidados a realizar a autenticação federada oferecida pelo sistema. O sucesso das operações por esses usuários marcou o fim da fase de testes, considerada exitosa.

Capítulo 7

Conclusões e Trabalhos Futuros

O fornecimento de recursos computacionais para instituições através de plataformas IaaS ainda possui espaço para diversas propostas de melhorias. O modelo de compartilhamento de recursos entre instituições é uma dessas propostas e traz consigo a construção de uma nuvem comunitária e geodistribuída, que deve ser mantida e gerenciada por diversas instituições. O GT-PID é uma implementação desse modelo de compartilhamento entre instituições.

Utilizando como núcleo o orquestrador de nuvens OpenStack, o GT-PID cria uma nuvem computacional comunitária entre instituições de ensino e pesquisa ligadas à RNP, realizando o compartilhamento da infraestrutura cedida pelas próprias instituições. A infraestrutura pertencente a cada instituição permanece em locais decididos por tais instituições. Esse fato confere à nuvem criada um caráter geodistribuído.

Como o GT-PID é um esforço conjunto de várias instituições e seus usuários são sempre ligados a uma dessas instituições, o GT-PID deve submeter seus usuários às políticas de sua respectiva instituição. Portanto, no que tange ao gerenciamento de usuários, a solução implementada por este trabalho foi a autenticação federada, que permite que cada instituição tenha controle direto sobre o gerenciamento identidades de seus usuários.

A existência da federação CAFe, uma federação oferecida pela RNP que reúne universidades e centros de pesquisa, motivou a inclusão do GT-PID nessa mesma

federação. O procedimento inicial é a inclusão do GT-PID na CAFe Expresso, um ambiente de testes para a CAFe. Este trabalho foi focado na inclusão do GT-PID na CAFe Expresso.

É de extrema importância que os usuários familiarizados com plataformas comuns de IaaS não sintam algum estranhamento ao utilizar um novo serviço de IaaS. Em virtude disso, este projeto buscou criar uma interface gráfica para o procedimento de autenticação adaptada à interface gráfica já existente na plataforma de IaaS utilizada, que é de ampla utilização.

7.1 Contribuições

Este trabalho demonstrou que é possível que uma plataforma de nuvem IaaS comunitária e geodistribuída seja parte de uma federação como um SP e que os procedimentos de autenticação sejam realizados por entidades externas ao serviço provido. Também foi demonstrado que é possível, a partir de modificações na versão Juno orquestrador de nuvens OpenStack, criar uma sequência de autenticação federada que permita a utilização da interface gráfica provida pelo módulo Horizon.

A sequência de implantação proposta e implementada demonstrou também a possibilidade da inserção dessa nuvem em um modelo centralizado de gerência de identidades, uma vez que os testes realizados com o TestShib [30] representam justamente um modelo de autenticação centralizado.

O trabalho abriu as portas para a instalação de IdPs controlados pelas instituições integrantes do GT-PID, bastando que haja concordância prévia sobre os atributos passados pelos IdPs ao serviço, para que identifiquem os possíveis usuários do sistema. Adicionalmente, o GT-PID também precisa da criação de representações de usuários em seu banco de dados, para que haja a persistência das operações realizadas pelos usuários.

7.2 Trabalhos Futuros

Como principal trabalho futuro é possível apontar a inclusão do GT-PID na CAFe, que é a federação fornecida pela RNP em um ambiente de uso para produção. Dessa maneira, o GT-PID poderá ser utilizado de forma plena por usuários de universidades e instituições de pesquisa ligadas à RNP.

Outro trabalho futuro relevante é a obtenção e avaliação de métricas de utilização, que podem sanar possíveis dúvidas sobre o impacto da utilização da interface gráfica na experiência dos usuários. Em termos de implementação e implantação, é interessante que o GT-PID seja capaz de aceitar usuários oriundos de IdPs sem que esses usuários estejam previamente no banco de dados interno. Para que isso aconteça, o GT-PID deve ser capaz de criar automaticamente objetos representando usuários todas as vezes em que um IdP emitir uma autenticação e o usuário não for reconhecido pelo GT-PID durante a fase de mapeamento. Uma última preocupação é estudar o funcionamento da implantação realizada neste trabalho com a instanciação do GT-PID em um ambiente de produção, sem a participação da plataforma de desenvolvimento DevStack.

É esperado que este trabalho contribua para a construção de uma nuvem comunitária acessível para as universidades de todo o Brasil que seja simultaneamente confiável, eficiente e de fácil utilização.

Referências Bibliográficas

[1] COUTO, R. S., SCIAMMARELLA, T., BARRETO, H. F. S. S. M., *et al.*, "GT-PID: Uma Nuvem IaaS Universitária Geograficamente Distribuída". In: *Quinta Conferencia de Directores de Tecnología de Información - TICAL 2015*, pp. 1–19, Redclara, 2015.

[2] MELL, P., GRANCE, T., *The NIST Definition of Cloud Computing*, Report, 09 2011.

[3] VOORSLUYS, W., BROBERG, J., BUYYA, R., *Introduction to Cloud Computing*, New York, USA, Wiley Press, pp. 1–44, 2011.

[4] "RNP Quem somos", `https://www.rnp.br/institucional/quem-somos`, Acessado em: 22/01/2016.

[5] "Grupos de Trabalho - RNP", https://www.rnp.br/pesquisa-e-desenvolvimento/grupos-trabalho, Acessado em: 13/03/2016.

[6] "OpensStack - Open Source Cloud Computing Software", `https://www.openstack.org/`, Acessado em: 22/01/2016.

[7] "Shibboleth", `https://shibboleth.net/`, Acessado em: 15/02/2016.

[8] FIELDING, R. T., *Architectural Styles and the Design of Network-based Software Architectures*. Ph.D. dissertation, University of California, Irvine, 2000.

[9] "Python", https://www.python.org/, Acessado em: 19/03/2016.

[10] FELICIANO, G., AGOSTINHO, L., GUIMARÃES, E., *et al.*, "Gerência de identidades federadas em nuvens: Enfoque na utilização de soluções abertas". In: *Minicursos do XI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, pp. 182–231, Sociedade Brasileira de Computação, 2011.

[11] HOWES, T. A., *The Lightweight Directory Access Protocol: X.500 Lite*, Report, 01 1995.

[12] MILINOVIĆ, M., RAUSCHENBACH, J., WINTER, S., *et al.*, *Deliverable DS5.1.1: eduroam Service Definition and Implementation Plan*, Report, 01 2008.

[13] "Where Are You From", `https://wayf.switch.ch/`, Acessado em: 18/02/2016.

[14] MOREIRA, E. Q., FOSCARINI, É. D., JUNIOR, G. C. D. S., *et al.*, *Federação CAFe: Implantação do Provedor de Identidade*. Rio de Janeiro, Brasil, Escola Superior de Redes, 2011.

[15] SOUZA, M. C. D., MELLO, E. R. D., WANGHAN, M. S., "CAFe Expresso: Comunidade Acadêmica Federada para Experimentação usando Framework Shibboleth". In: *XIV Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - SBSeg 2014*, pp. 405–414, Sociedade Brasileira de Computação, 2014.

[16] "Software - Open Source Cloud Computing Software", `http://www.openstack.org/software/`, Acessado em: 22/01/2016.

[17] "Apache License, Version 2.0", `http://www.apache.org/licenses/LICENSE-2.0`, Acessado em: 15/02/2016.

[18] "The OpenStack Foundation", `https://www.openstack.org/foundation/`, Acessado em: 15/02/2016.

[19] "Companies - Open Source Cloud Computing Software", `http://www.openstack.org/foundation/companies/`, Acessado em: 22/01/2016.

[20] "OpenStack Summit", `https://www.openstack.org/summit/`, Acessado em: 15/02/2016.

[21] "OpenStack Juno", `https://www.openstack.org/software/juno/`, Acessado em: 15/02/2016.

[22] OpenStack Foundation, "OpenStack Installation Guide", http://docs.openstack.org/juno/install-guide/install/apt/content/ch_preface.html, Dezembro 2015, Acessado em: 16/02/2016.

[23] "Django - The web framework for perfectionists with deadlines", https://www.djangoproject.com/, Acessado em: 16/02/2016.

[24] "Configuring Keystone for Federation", http://docs.openstack.org/developer/keystone/configure_federation.html, Acessado em: 23/02/2016.

[25] "DevStack - an OpenStack Community Production", http://docs.openstack.org/developer/devstack/, Acessado em: 24/02/2016.

[26] Security Services Technical Committee, *Authentication Context for the OASIS Security Assertion Markup Language (SAML) V2.0*. OASIS Open, 2005.

[27] CARMODY, S., ERDOS, M., HAZELTON, K., *et al.*, "Shibboleth Architecture: Protocols and Profiles", http://shibboleth.internet2.edu/shibboleth-documents.html, 9 2005.

[28] CANTOR, S., MOREH, J., PHILPOTT, R., *et al.*, "Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0 - Errata Composite", http://www.oasis-open.org/committees/documents.php?wg_abbrev=security, 2 2007.

[29] "Metadata - Shibboleth Concepts", https://wiki.shibboleth.net/confluence/display/CONCEPT/Metadata, Acessado em: 19/02/2016.

[30] "TestShib", http://www.testshib.org/, Acessado em: 23/02/2016.