DOI: 10.5748/9788599693124-13CONTECSI/PS-4061

SINGLE SIGN-ON: AN INFORMATION SECURITY APPROACH

Emilio Carvalo Dias (FaculdadePitágoras de Uberlândia, Minas Gerais, Brasil) - emiliodias@gmail.com
Rogerio de FreitasRibeiro (FaculdadePitágoras de Uberlândia, Minas Gerais, Brasil) - rogeriofr@gmail.com

This study was created from the analysis of applications that support SSO (Single Sign-On) in corporate environments. The focus of this work is the concepts of authentication and SSO process in addition the aspects of information security. The approach of the SSO concept in corporate environments typically have limitations or most of the time is lost by using to unsatisfactory approaches to user authentication process. The study looking for an architectural proposal that allows the deployment of centralized management of identities, providing mechanisms to improve the user experience. The methodology is guided on a literature review and others works with the same subject. This work provides an analysis about possible improvements on authentication in corporate environment.
Keywords: single sign-on, saml, openid, authentication.

SINGLE SIGN-ON: AN INFORMATION SECURITY APPROACH

O presente trabalho foi elaborado a partir da análise das aplicações com suporte a SSO (Single Sign-On) em ambientes corporativos. São considerados como foco desta análise os conceitos de autenticação e SSO bem como os aspectos relativos à segurança da informação. A abordagem do conceito de SSO em ambientes corporativos normalmente possui limitações ou na maioria das vezes está ausente, o que leva a abordagens pouco satisfatórias nos processos de autenticação do usuário. O estudo objetiva uma proposta arquitetural que possibilite a implantação do gerenciamento de identidades centralizado, provendo mecanismos que melhore a experiência do usuário. A metodologia utilizada pauta-se em uma revisão bibliográfica em obras complacentes sobre o tema. O trabalho em questão proporciona uma análise sobre as possíveis melhorias na autenticação em um ambiente corporativo.

Palavras-chave: single sign-on, saml,openid, autenticação.

1. Introdução

Questões a respeito da segurança da informação estão sempre em discussão e muitas destas estão relacionadas a uma identidade digital. Indagações se uma determinada informação pode ser fornecida para um determinado usuário ou a necessidade de verificação da origem de uma requisição, sempre são levantadas. Como garantir se um usuário é realmente quem ele apresenta ser? Estas questões estão intimamente relacionadas ao gerenciamento de identidades, as quais podem se referir a um indivíduo ou até mesmo um computador. O estabelecimento destas, é pré-requisito fundamental na garantia da autenticidade e não-repúdio, pilares da segurança da informação (REMY, 2004).

Desde a criação da arquitetura TCP/IP (T*ransport Control Protocol / Internet Protocol*), na qual grande parte dos sistemas distribuídos são baseados, a capacidade essencial do controle de identidades é inexistente. Com a ausência de uma camada responsável pelo tratamento de identidades, soluções paliativas e pontuais surgem para resolverem problemas específicos, o que impede uma interoperabilidade maior entre os sistemas atuais (CAMERON, 2005).

Assim, o trabalho tem como tema discorrer sobre o gerenciamento de identidades em ambientes corporativos no que diz respeito à autenticação única, ou como é mais conhecido SSO (Single Sign-On), através de uma pesquisa realizada no Brasil no ano de 2015. Esta pesquisa partiu da seguinte problemática: por que é tão burocrático e complexo a construção de sistemas com suporte a autenticação única em ambientes corporativos e qual o impacto de arquiteturas de software cada vez mais distribuídas e modulares neste modelo?

A justificativa do trabalho é pautada na necessidade de entendimento sobre a autenticação em ambientes corporativos, sendo que atualmente é comum a criação da infraestrutura de autenticação necessária em cada *software*, o que impede um controle centralizado, padronizado e melhor gerenciável. Além disso, com a utilização de sistemas cada vez mais distribuídos e modulares, se torna indispensável uma forma transparente e simples de conectá-los de maneira segura, minimizando custos de integração e possíveis fraudes.

Especificações como *SAML* (*Security Assertion Markup Language*) (OASIS, 2005) e OpenID propõe soluções para mecanismos de SSO. Porém, a falta de conhecimento sobre a existência de tais soluções, leva desenvolvedores a criarem soluções paliativas e incapazes de resolver o problema de forma mais adequada.

É notório a necessidade de uma solução que possibilite uma melhoria na experiência e redução de complexidade na gestão de identidade dos usuários. Outra necessidade é a eliminação da obrigação de novas autenticações em cada um dos sistemas que o usuário utiliza, bem como uma centralização nas políticas de autorização, características que vão de encontro com as subdisciplinas existentes na área de segurança da informação.

O objetivo geral é apresentar uma proposta arquitetural que possibilite a implantação do gerenciamento de identidades centralizado, provendo mecanismos que melhore a experiência do usuário em relação aos seus acessos em sistemas corporativos e que minimize a complexidade de integração entre os mesmos. O trabalho se subdivide em três objetivos específicos: mostrar os padrões atualmente disponíveis para gerenciamento de identidade; propor uma arquitetura que possa ser utilizada na implantação destes sistemas tendo como base os padrões pesquisados; e validar a arquitetura proposta por meio de ferramentas de código aberto.

Foi considerada, para fins de estudos, uma pesquisa bibliográfica em diferentes obras sobre o tema, tais como: teses, dissertações e artigos científicos que discorrem sobre o tema proposto. Adicionalmente foi realizado um levantamento de ferramentas de código aberto disponíveis na comunidade.

Este trabalho está organizado em quatro capítulos, incluindo uma introdução e uma conclusão. O segundo capítulo compreende o referencial teórico mediante aos conceitos fundamentais sobre o tema proposto, tais como SSO e protocolos na sua implementação. O terceiro capítulo faz uma abordagem elencando os processos de uma aplicação web com suporte a SSO utilizando o protocolo SAML.

Assim sendo, o trabalho propõe uma abordagem sob uma ótica de melhorias na autenticação em um ambiente corporativo.

2. Autenticação e Single Sign-On

No contexto da segurança da informação, autenticação é o processo para que um indivíduo ou sistema confirme sua autenticidade. Este processo normalmente esta relacionado ao ato de verificação de identidade (STALLINGS; BROWN, 2014).

Segundo a RFC 2828 (IETF, 2000), a autenticação é um processo de verificação de uma identidade alegada por uma pessoa ou sistema, sendo este processo dividido entre as etapas de identificação e verificação.

Uma prática comum no desenvolvimento de software é a criação do processo de autenticação em cada uma das aplicações. Com o advento de sistemas computacionais cada vez mais distribuídos, é necessário que os usuários gerenciem um número elevado de credenciais, o que gera uma experiência de uso insatisfatória. Além disto, o cenário proporciona o aparecimento de falhas de segurança da informação, dada a necessidade do usuário em ter que gerenciar uma grande quantidade de usuários e senhas que por ventura possuem critérios de formação da senha com uma baixa complexidade. Outro ponto importante que podemos citar seria os custos para desenvolver e manter seguros um módulo individual para cada sistema, para realizar o cadastro de usuário, o *enforcement*[1] da política de senhas definida pela organização, além dos processos de autenticação. A tabela 1 descreve os requisitos comuns para o *enforcement* da política de senhas que precisaria ser desenvolvida em todas as aplicações, caso fosse optado pelo modelo de cada aplicação tratar as questões de autenticação.

Requisito	**Descrição**
Desenvolver a validação do tamanho mínimo da senha	Impedir que o usuário escolha uma senha com o comprimento de caracteres menor do que o aceitável.
Desenvolver a complexidade da senha	Impedir que o usuário escolha senhas consideradas fracas.
Desenvolver a validação da idade mínima para uma nova troca de senha	Impedir que o usuário realize sucessivas trocas de senhas para retirar a senha deseja do histórico da senha permitindo que ele volte a utilizar a mesma senha.
Desenvolver a validação da idade máxima para uma nova troca de	Impedir que o usuário utilize uma mesma senha por tempo maior que o aceitável.

[1] Mecanismos sistêmicos que forcem o usuário a seguir um determinado comportamento ou padrão. No caso do *enforcement* de senhas, trata-se de um módulo que valide se o usuário está construindo uma senha baseado nos critérios pré-configuradas no sistema.

senha	
Desenvolver o mecanismo para bloqueio do reuso de senhas com base no histórico das últimas senhas	Impedir que o usuário utilize sempre as mesmas senhas.
Desenvolver os mecanismos de bloqueios da conta com base em sucessivas falhas de autenticação	Impedir que ataques de força bruta possam descobrir a senha do usuário.

Tabela 1: Requisitos comuns para a melhoria do processo de autenticação com relação a segurança da informação.
Fonte: Autores.

Uma técnica comum para o tratamento desta questão é o SSO, técnica que utiliza uma abordagem que permite ao usuário autenticar-se apenas uma vez e se tornar automaticamente apto a utilizar qualquer sistema desejado. Esta abordagem permite uma melhor experiência de uso, além de centralizar a gestão dos controles de acesso, o que eleva o nível de confiabilidade na segurança da informação dos sistemas envolvidos, além de reduzir os custos de desenvolvimento de módulos individuais para gerenciar os aspectos de segurança em cada uma das aplicações, obviamente quando considerado uma organização que possui desenvolvimento interno ou contrata o desenvolvimento de sistemas específicos para as suas necessidades.

Com um gerenciamento centralizado de autenticação e controles de acesso, uma corporação beneficia-se pelo fato de conseguir de forma centralizada gerir políticas de controle de acesso e controle sobre o que pode ou não ser executado em um sistema.

Sistemas que possuem controle próprio do processo de autenticação e gerenciamento de usuários, podem possuir recursos limitados referentes a políticas para troca e complexidade de senha, e também mecanismos limitados para o armazenamento das credenciais do mesmo.

Um estudo feito com 7 mil contas de usuário pela Purdue University demonstra que quase 4% dos usuários possuíam menos de 3 caracteres em sua senha e que apenas 40% dos usuários utilizavam senhas com no mínimo 8 caracteres (STALLINGS, William, 2014). Fato este que nos remete à necessidade de possuir um sistema que consiga gerenciar uma política de complexidade mais adequada.

Porém, isto de forma descentralizada, remete usuários a um comportamento de uso inadequado, que é o uso da mesma credencial em sistemas diferentes, visto que memorizar senhas complexas para vários sistemas se torna uma tarefa difícil. Esta situação levanta um risco eminente. Pelo fato de alguns sistemas poderem possuir vulnerabilidades no controle de acesso, um atacante poderia obter as credenciais de acesso e utiliza-las em todos os sistemas.

Desta forma um sistema de controle de acesso e SSO centralizados nos permite o controle central das políticas de complexidade e trocas de senha, e melhoria na experiência do usuário, fornecendo ao mesmo a possibilidade de apenas uma credencial. Este controle também diminui os custos de uma corporação, visto que a operação de um sistema centralizado é menos complexa e a eliminação da funcionalidade de controle de acesso repetido nas aplicações deixa de existir.

Um fato que pesa contra mecanismos centrais de controle de acesso e SSO é o fato do usuário de alguma maneira ter suas credenciais roubadas, isso levaria o atacante a ter acesso a todos os sistemas que o usuário possui, de qualquer forma, os benefícios citados acima possuem mais vantagens se comparada a este único problema levantado. Para

minimizar este risco, as corporações precisam garantir um maior nível de segurança para o ambiente centralizado que gerencia as credenciais.

Diferentes arquiteturas são propostas para a implementação de SSO, a abordagem mencionada neste trabalho leva em consideração arquiteturas que possuem em sua essência a utilização de um elemento denominado IdP (*Identity Provider)*, o qual é responsável por gerenciar o acesso centralizado no que diz respeito ao gerenciamento de identidade do usuário, incluindo o processo de autenticação, que é o tema central desta pesquisa.

Em um sistema de gerenciamento de identidade é comum a utilização de quatro elementos no processo de autenticação do usuário, sendo eles:

- Usuário: pessoa que deseja utilizar um recurso protegido;
- Recurso: contém o conteúdo e/ou funcionalidades desejados pelo usuário;
- IdP: responsável pela autenticação do usuário e gerenciamento do ciclo de vida de suas credenciais;
- SP (*Service Provider*): possui o recurso privado em que o usuário esta tentando obter acesso, é chamado também algumas vezes de RP (*Relying Party*).

Com a utilização de um IdP, as aplicações delegam a funcionalidade de autenticação e passam a confiar no processo de autenticação deste elemento. Na figura 1 pode ser visto a diferença arquitetural entre um modelo SSO e uma arquitetura convencional.

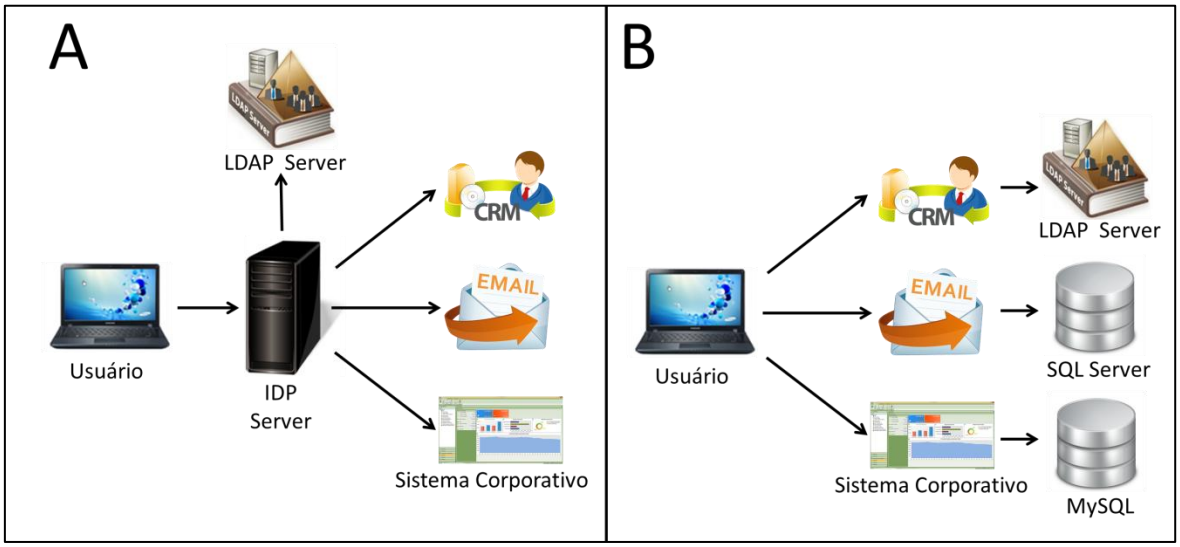

Figura 1: A) Arquitetura SSO; B) Arquitetura convencional.
Fonte: Adaptada pelos autores.

Com a utilização de uma arquitetura SSO, informações sobre identidades e seus atributos são enviados do IdP para o SP, assim algumas vantagens são obtidas. Além da autenticação única, conforme já mencionado, outra característica é que os SPs podem delegar funcionalidades de gerenciamento do ciclo de vida da identidade do usuário e apenas utilizar os serviços desejados nos momentos necessários. Também é importante citar que o IdP pode trabalhar na melhoria dos métodos de autenticação e adicionar novos fluxos necessários, como por exemplo, o segundo fator de autenticação.

Empresas como Google[2], Facebook[3] e Yahoo[4] são exemplos de provedores de identidade. Em aplicações para a Internet atuais é comum a delegação do processo de autenticação para alguns destes provedores.

Em ambientes corporativos é comum o uso de sistemas LDAP (*Lightweight Directory Access Protocol*) (IETF, 2006) onde embora a gestão da identidade possa ser feita de forma centralizada (credenciais únicas) o processo de autenticação ainda é de responsabilidade das aplicações o que impede a criação de um ambiente SSO. Para a implementação de arquiteturas que de fato sigam as características de um ambiente SSO se faz necessário o uso de um IdP.

A implementação e serviços fornecidos por um determinado IdP pode-se dar de diversas formas o que torna necessário a utilização de implementações que sigam alguns protocolos padrões, tais como, OpenID Connect ou SAML, o que permite uma maior interoperabilidade e portabilidade para futuras alterações.

Uma visão geral desta arquitetura com as etapas realizadas é apresentada na figura 2. Esta arquitetura ficará mais compreensível no decorrer do artigo.

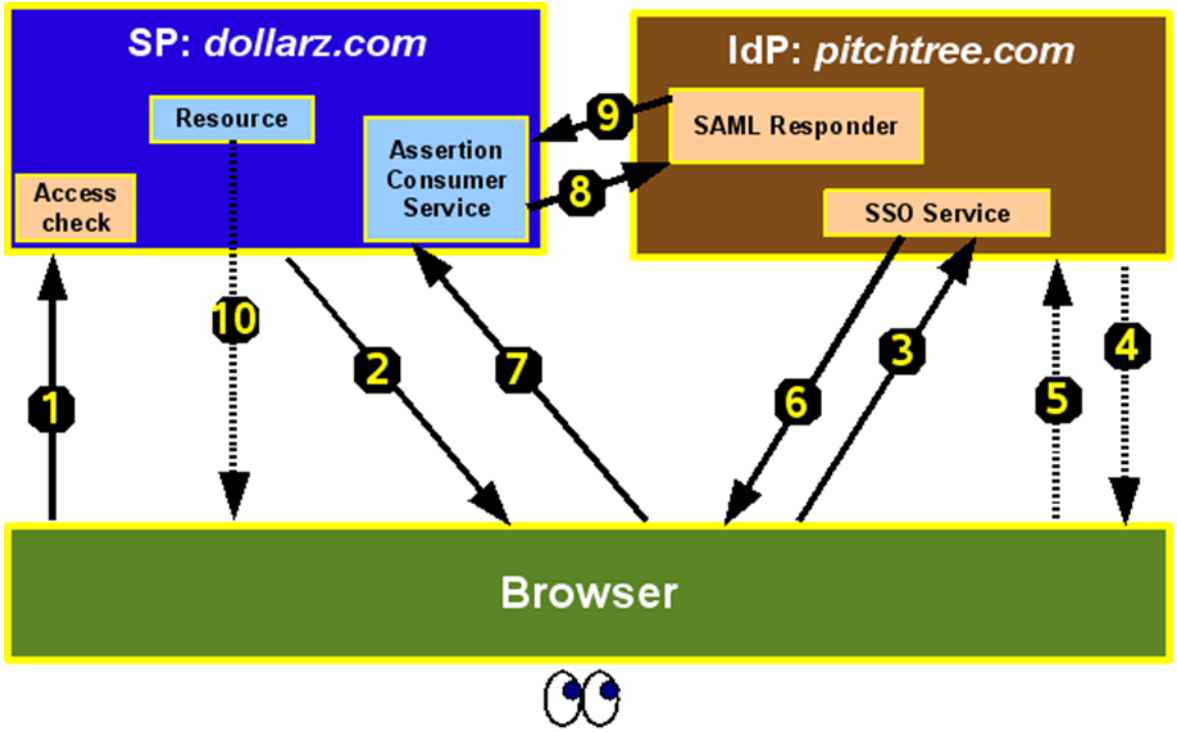

Figura 2: Arquitetura geral de comunicação entre um SP (Recurso Protegido) e um IdP.
Fonte: http://www.xmlgrrl.com/publications/177-maler-fed-id.html

A seguir temos a descrição de cada um dos passos ilustrados na figura 02.

01 – O usuário em seu *browser* (Navegador de Internet) tenta acessar uma determinada área protegida do site http://dollarz.com. Assumimos neste ponto que o usuário ainda não está autenticado no sistema, desta forma ainda não existe uma sessão criada.

[2] Google: www..google.com
[3] Facebook: www.facebook.com
[4] Yahoo: www.yahoo.com

02 – O site dollarz.com envia de volta um formulário HTML para *browser* do usuário. Este formulário contém uma requisição de Autenticação no formato SAML solicitando algumas informações.
03 – O *browser*, executa algumas ações e auto submete uma requisição via POST para o serviço de SSO hospedado em http://pichtree.com.
04 – Caso o serviço de SSO não tenha uma informação válida de uma autenticação recente deste usuário, é retornado ao *browser* do usuário uma requisição de autenticação.
05 – O usuário realiza o envio das credenciais de acesso corretas. Nesta situação o processo de autenticação ocorrerá com sucesso.
06 – O serviço de SSO pichtree.com gera um conjunto de informações no formato SAML descrevendo os detalhes da autenticação do usuário.
07 – O browser do usuário envia as informações geradas pelo serviço de SSO. Como existe uma relação de confiança pré-configurada entre o serviço de SSO pichtree.com com o recurso protegido dollarz.com o processo continua.
08 – O recurso protegido dollarz.com consulta o serviço de SSO solicitando uma confirmação da autenticação.
09 – O serviço de SSO pichtree.com responde com a confirmação desejada.
10 – Neste momento o site dollarz.com concede o acesso ao recurso protegido.

2.1 SAML

SAML é uma linguagem de marcação mantida pela OASIS[5] que tem como princípio o fornecimento de asserções (*Assertions*) para autenticação, autorização e informações relacionados a identidade (SIRIWARDENA, 2014).

SAML é uma linguagem declarativa baseada em XML constituída de basicamente de quatro elementos (ROSEMBERG; REMY, 2004), sendo eles:

- *Assertions*: Um *schema* XML constituído de definições para autenticação, autorização, atributos e que pode ser estendido.
- *Protocol*: Elementos *Request* e *Response* para encapsular *Assertions*.
- *Bindings*: Regras para utilização de *Assertions* sobre determinados protocolos e frameworks de mensagens, como por exemplo SOAP.
- *Profiles*: É o agrupamento de *Assertions*, *Protocols* e *Bindings* para um determinado caso de uso. Como exemplo pode-se citar o SAML 2.0 Web Browser Single Sign-On, que será utilizado neste trabalho para implementação do ambiente SSO.

SAML permite a implementação de ambientes federados[6] como especificado pelo padrão WS-Trust (LAWRENCE; KALER, 2007) e a criação de ambientes SSO, sejam eles baseados em *browser* ou em ambientes STS (*Security Token Service*). Este trabalho demonstra a implementação de um ambiente SSO com SAML a partir da utilização da ferramenta WSO2 Identity Server.

[5] OASIS: Consórcio sem fins lucrativos que impulsiona o desenvolvimento, convergência e adopção de normas abertas para a sociedade da informação global.
[6] Federação: gerenciamento de identidades federadas possui um conjunto comum de políticas, práticas e protocolos em vigor para gerenciar a identidade e confiança em usuários de TI e dispositivos em corporações. SSO é um subconjunto do gerenciamento de identidade federado.

2.2 OpenID Connect

Como já discutido anteriormente, desde o advento da Web, a cada nova aplicação que usuários desejam utilizar, se faz necessário a criação de novas credenciais. Este processo, além de burocrático, possui diversos inconvenientes de segurança.

OpenID (OIDF, 2014) foi criado com o intuito de habilitar a federação de identidades na WEB, permitindo que usuários possuam uma única credencial e utilizem-na através de diferentes aplicações. Isto elimina a necessidade de criação de diversas credenciais pelo usuário além de retirar a necessidade de cada aplicação construir seu próprio processo de autenticação (BOYD, 2012). OpenID Connect é a próxima geração do protocolo OpenID e é baseado sobre o protocolo OAuth (IETF, 2012). Atualmente, grandes empresas como Google, Yahoo e Amazon[7] utilizam este protocolo.

Diferentemente de abordagens como SAML que precisam de configurações prévias entre um IdP e um SP o OpenID trabalha com um padrão de federação aberta na qual um SP pode confiar a delegação para qualquer servidor de OpenID já existente. Apesar de ser uma abordagem mais flexível a confiabilidade e privacidade podem apresentar limitações se comparadas com SAML.

A especificação OpenID (OIDF, 2014) recomenda a utilização de um canal de comunicação segura como o TLS/SSL para que as mensagens trafegadas durante o processo de autenticação não sofram algum tipo de ataque como por exemplo "*man in the middle*"[8] (STALLINGS; BROWN, 2014). É importante citar que apesar do protocolo fornecer garantias contra diversos tipos de ataque, uma má implementação ainda pode trazer riscos de segurança para o usuário.

3. Implementação de uma aplicação Web com suporte a Single Sign-On

Na implementação realizada neste trabalho a ferramenta WSO2 Identity Server foi utilizada como um servidor de identidade (IdP) para que seja possível o desenvolvimento do ambiente SSO. A aplicação desenvolvida com suporte ao protocolo SAML irá delegar sua autenticação ao WSO2 Identity Server, demonstrando a capacidade de "terceirizar" o processo de autenticação ao IdP. É importante citar que arquiteturalmente a aplicação executa o papel de um SP.

Em uma abordagem convencional, no momento que um usuário tenta acessar uma aplicação é apresentado uma tela de autenticação onde o mesmo pode fornecer suas credenciais. Neste cenário, esta interface e toda a implementação do processo de autenticação estão desenvolvidos internamente na própria aplicação, o que impede o fato de SSO.

Na aplicação apresentada neste trabalho, ao receber uma requisição e perceber que o usuário não esta autenticado a aplicação irá redirecioná-lo ao IdP para que ele possa fornecer suas credenciais. Após a validação, o IdP redireciona o usuário de volta à aplicação com o resultado da autenticação. Como resultado da autenticação, o IdP pode também enviar atributos do usuário para que a aplicação faça uso dos mesmos.

[7] Amazon: www.amazon.com

[8] Ataque que objetiva realizar a interceptação de dados em uma rede de computadores por um atacante que inicialmente não teria acesso a comunicação realizada pela origem e destino.

3.1. WSO2 Identity Server

O WSO2 Identity Server é um servidor de código aberto (*Open Source*) que provê avançados recursos para o gerenciamento de identidade e acesso para aplicações Web corporativas, APIs (*Application Program Interface*) e tem como objetivo ser um servidor de fácil gerenciamento e configuração.

Ele provê serviços para a implementação de arquiteturas com suporte a SSO, delegação de autorização com XACML (*eXtensible Access Control Markup Language*) e gerenciamento do ciclo de vida do usuário baseado em SCIM (*System for Cross-domain Identity Management*). Dentre os principais recursos, pode-se citar:

- Gerenciamento de identidade
  - APIs para integração com qualquer aplicação por meio de SOAP (*Simple Object Access Protocol*);
  - Autenticação com múltiplos fatores;
  - Single Sign-On com OpenID, SAML e Kerberos;
  - Autorização com Oauth[9].
- Aprovisionamento de usuários
  - Suporte a SCIM;
  - Suporte a várias bases de dados como LDAP e JDBC (*Java Database Connectivity*).
- Autorização
  - XACML.
- Gerenciamento e monitoramento

O WSO2 Identity Server pode ser obtido em seu endereço oficial, a versão utilizada neste trabalho é a "4.5.0". Desenvolvido em linguagem Java, ele é uma aplicação baseada na especificação OSGi[10]. Na figura 3 pode ser visto a tela inicial de sua interface administrativa.

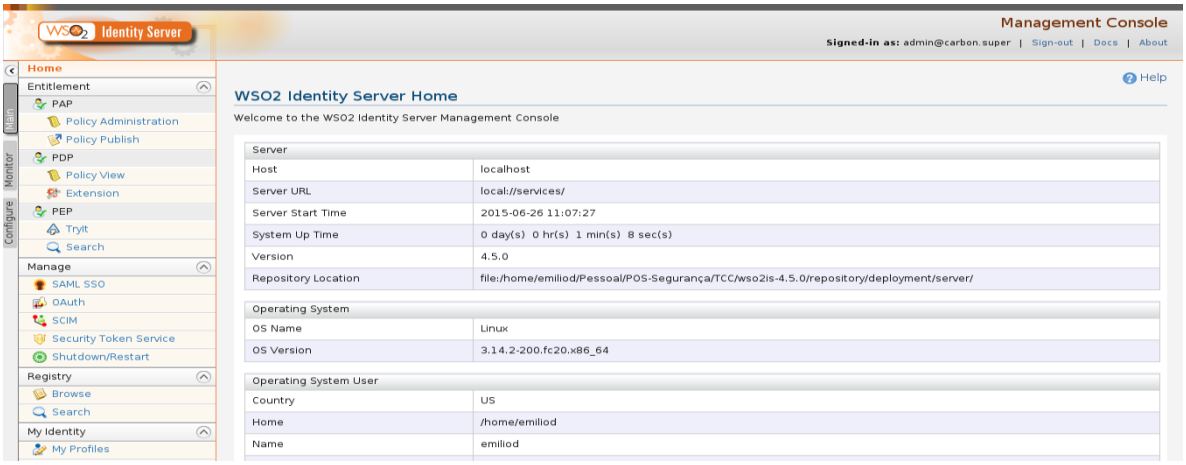

Figura 3: Interface de administração WSO2 Identity Server.
Fonte: Autores.

[9]OAuth: protocolo aberto que permite a autorização segura em um método simples e padrão de aplicações web, desktop e móveis. OAuth foi criado com o intuito de delegar o processo de autorização a aplicações terceiras sem a necessidade de o usuário compartilhar suas credenciais.
[10] OSGI: A especificação OSGi descreve um sistema modular e uma plataforma de serviços para a linguagem de programação Java que implementa um modelo de componente completo e dinâmico, algo que não existe em ambientes Java / VM autônomos.

Após efetuado o *download* e executado o processo de descompactação do arquivo, pode ser visto uma estrutura de diretórios semelhante a listagem presente na figura 4.

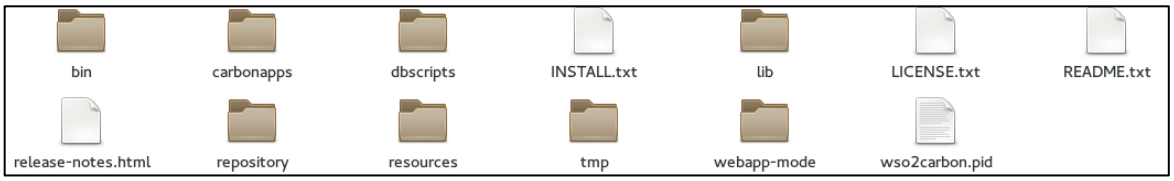

Figura 4: Arquivos e diretórios do WSO2 Identity Server.
Fonte: Autores.

Para a execução do servidor, considerando que esta esteja sendo executada em um ambiente com o sistema operacional Linux, deve-se acessar o diretório "bin" e realizar a execução do *script* "./wso2server.sh". Caso a execução seja feita sob uma máquina Windows, o *script* executado deve ser o "wso2server.bat".

Na execução do *script*"wso2server.sh" ou "wso2server.bat" deve-se garantir que o servidor utilizado tenha uma máquina virtual Java instalada e configurada corretamente além de garantir a existência da variável de ambiente "JAVA_HOME" referenciando corretamente o diretório de instalação da JVM (*Java Virtual Machine*).

Um grande conveniente no WSO2 Identity Server é a facilidade em sua configuração inicial, como discutido nesta sessão, por ser uma aplicação desenvolvida em linguagem Java, não se faz necessário à instalação de arquivos binários, o que facilita a configuração de ambientes para o desenvolvimento de aplicações e até mesmo o aprovisionamento de ambientes de produção, não sendo necessário grande esforço de uma equipe.

3.2. SAML 2.0 Web Browser Single Sign-On

Como já discutido anteriormente, SAML é um padrão baseado em linguagem XML que provê mecanismos para autenticação e autorização entre SPs e um IdP. Também foi discutido a flexibilidade que o mesmo possui em relação a adição de novos cenários de uso, podendo o mesmo ser utilizado para aplicações WEB ou APIs por exemplo.

Neste trabalho foi utilizado o Web Browser Single Sign-On, um profile especificado na versão 2 de SAML e que envolve a troca de informações entre um IdP, SP e o WEB browser do usuário final. Na figura 4, pode-se ver um diagrama de sequência para um caso onde o usuário acesse uma aplicação (*Service Provider*) e o mesmo é redirecionado ao IdP por não ter realizado uma autenticação prévia.

O fluxo demonstrado na figura 4, permite a implementação de aplicações corporativas WEB, na qual uma corporação deve possuir um IdP devidamente configurado e configurado suas aplicações para delegar sua autenticação a ele.

Apesar de executar o fluxo para o processo de autenticação, é importante notar que a base de dados dos usuários fica a critério da implementação e configurações do IdP.

O SAML fornece uma interface padrão para a comunicação entre um SP e um IdP, desta forma caso uma corporação queira realizar a troca de um IdP por outro, a mesma não sofrerá impactos, o que torna a portabilidade muito mais eficiente em relação a usar soluções proprietárias.

Pelo fato de SAML apenas controlar o fluxo de mensagens para a execução do processo de autenticação, o IdP pode ter uma flexibilidade em sua implantação e utilizar a base de usuários já existente em uma corporação. Caso a empresa já possua sistemas que

controlam o aprovisionamento e o armazenamento de dados referentes aos acessos do usuário, o IdP pode fazer uso destes dados.

Para ilustrar melhor, caso uma empresa já tenha uma base LDAP em produção, o IdP controla o fluxo de autenticação junto ao SP utilizando SAML, mas quando o usuário efetuar sua autenticação o IdP delega a este LDAP o processo de validação de suas credenciais. A figura 5, apresenta um diagrama de sequencia que descreve este processo. Desta forma, pode-se ter um impacto menor em uma implantação de um cenário SSO.

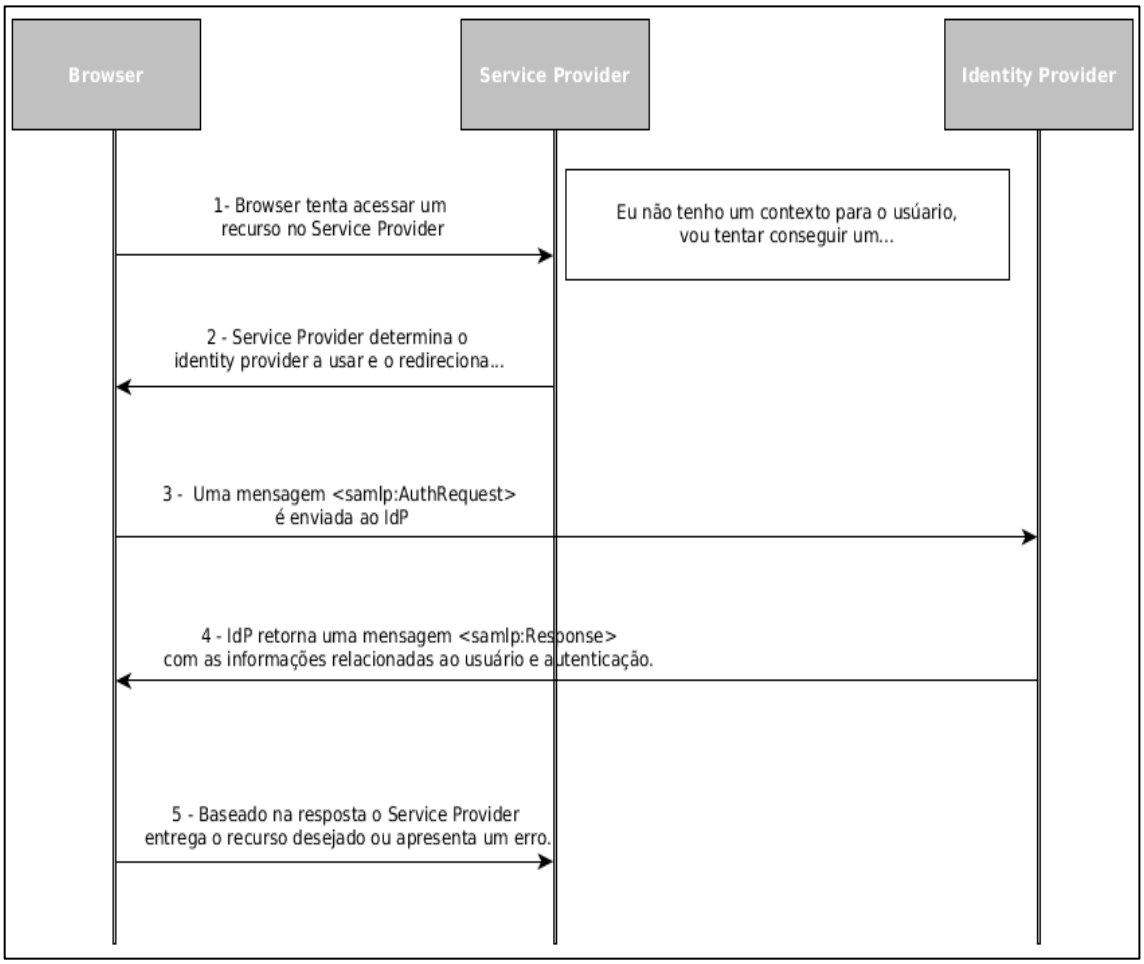

Figura 5: Diagrama de sequencia com as interações.
Fonte: Autores.

Além disso, caso o IdP queira adicionar algum segundo fator de autenticação ou alguma outra validação com o usuário, ele pode fazê-lo sem que seja gerado impacto no fluxo SAML, a única coisa que deve ser mantida é o autômato de comunicação entre o IdP e SP.

3.3. Aplicação com suporte nativo a SAML

Aplicações que já são desenvolvidas com o intuito de fazer parte de um ambiente SSO normalmente possuem suporte nativo a delegação de autenticação o que fornece uma maneira mais natural na criação deste ambiente. Tais aplicações podem fazer uso de protocolos como OpenID Connect e SAML ou optarem por implementações proprietárias. Esta sessão tem como objetivo demonstrar a implementação de uma aplicação WEB

desenvolvida na linguagem Java, executada sobre um servidor Tomcat e que tenha suporte nativo ao protocolo SAML.

Na tentativa de acesso a esta aplicação, o usuário será redirecionado ao WSO2 Identity Server com o propósito de autenticação, após autenticado o mesmo será redirecionado novamente à aplicação e poderá fazer uso da mesma.

O processo de redirecionamento e autenticação é feito de forma transparente para o usuário e nenhum tipo de complexidade a mais é exigida do mesmo. Na figura 6, é demonstrado o diagrama de sequência que o usuário percorre até estar apto a utilizar a aplicação. Em sequencia está descrito os passos que correm durante a autenticação.

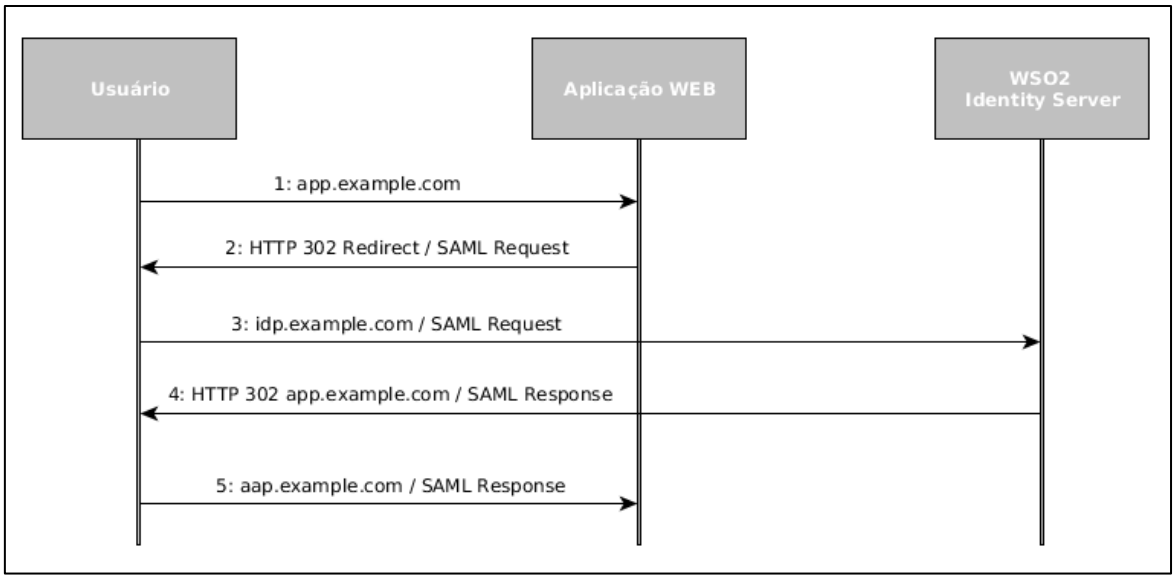

Figura 6: Fluxo para autenticação com SAML Web Browser SSO.
Fonte: Autores

No passo 1 o usuário acessa o endereço da aplicação. A aplicação verifica se o usuário já está autenticado. Caso não esteja autenticado, ele é redirecionado utilizando o método HTTP 302 Redirect (IETF, 1999).

No passo 2 o usuário é redirecionado com uso do HTTP 302 Redirect com um *payload*[11] SAML, requisitando uma autenticação junto ao WSO2 Identity Server.

No passo 3 é apresentado uma interface de autenticação para que o usuário possa fornecer suas credencias, esta interface é apresentada na figura 7.

[11] Payload: refere-se as informações enviadas e recebidas em uma transmissão de dados. Também é referido como os dados reais ou corpo de uma mensagem. O payload desconsidera informações referidas aos metadados de uma transmissão, como por exemplo, as informações referentes aos cabeçalhos dos protocolos TCP/IP e HTTP

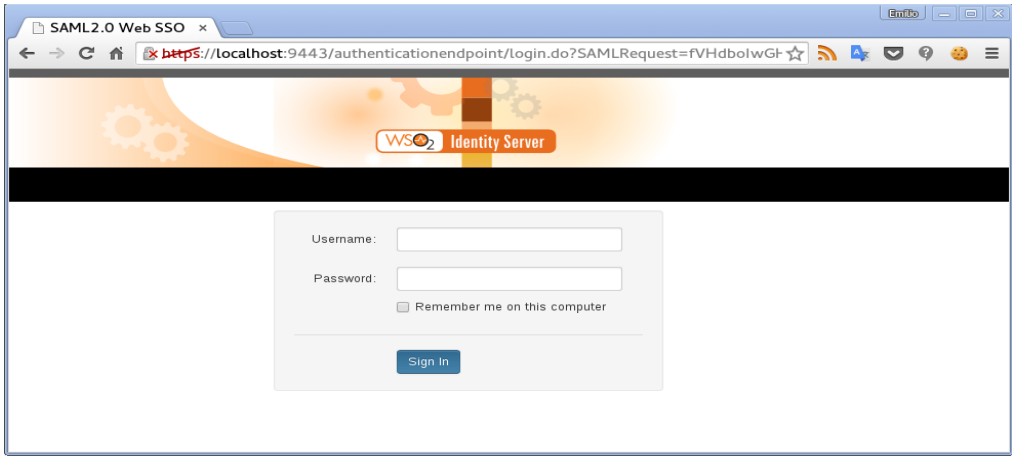

Figura 7: Tela de autenticação do WSO2 Identity Server para o SSO.
Fonte: Autores.

No passo 4 após a autenticação o WSO2 Identity Server envia um HTTP 302 Redirect com a resposta SAML para que o usuário possa acessar a aplicação.

No passo 5 finalmente o usuário acessa aplicação com a resposta SAML enviada pelo WSO2 Identity Server. A resposta é validada pela aplicação e em caso de sucesso a aplicação é acessada pelo usuário, conforme pode ser visto na figura 8.

Figura 8: Tela da aplicação após processo de autenticação pelo usuário.
Fonte: Autores.

O protocolo HTTP é baseado em um modelo *stateless*[12] *(IETF, 1999)*, com isto, se faz necessário alguma forma para que o usuário possa manter seu estado autenticado. Na atualidade, existem basicamente duas formas de atender este requisito, são elas: *Token based*[13] e *Cookie based*[14] (SIRIWARDENA, 2014).

[12] Stateless: refere-se a um protocolo sem estado. Protocolos de comunicação sem estado tratam cada pedido como uma transação independente que não está relacionada a qualquer pedido anterior. O servidor não tem a necessidade de reter informações de sessão ou de status sobre cada usuário.
[13] Token based: um token é utilizado para autenticar o usuário em cada requisição ao servidor e informações sobre a autenticação não são armazenadas em memória.
[14] Cookie based: após uma autenticação válida, o servidor cria um ID e o associa a sessão, armazendo-o em um cookie no browser do usuário.

A implementação de SAML Web Browser SSO utilizada neste trabalho é baseado em um modelo de autenticação baseado em *cookies*[15], assim sendo, o usuário mantém um *cookie* estabelecido com o WSO2 Identity Server e outro *cookie* com a aplicação, conforme demonstrado na figura 9.

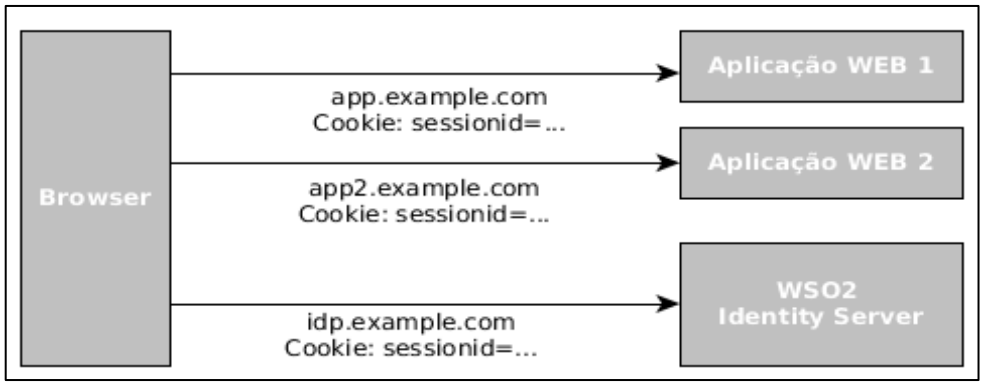

Figura 9: Controle de sessão baseado em cookie.
Fonte: Autores.

Soluções baseadas em *cookie* possuem um inconveniente relacionado a escalabilidade da aplicação. Caso exista a necessidade de adicionar nós computacionais por questões relacionadas à necessidade de alta disponibilidade ou por motivos de balanceamento de carga (aumento de usuários concorrentes acessando a aplicação), se faz necessário algum mecanismo para que se permita essa escalabilidade horizontal.

A primeira alternativa é a utilização de uma sessão baseada em Token. Atualmente pode-se utilizar o formato JWT (JSON WEB Token) para o armazenamento dos dados referentes à sessão do usuário em formato de Token (SIRIWARDENA, 2014).

Outra possibilidade é manter a utilização de cookies e abordar a centralização dos dados da sessão, desta maneira utiliza-se um elemento central como um *Memcached*[16] para armazenamento dos dados ao invés de um armazenamento em memória local.

Uma terceira possibilidade é a utilização de *sticky session,* onde o balanceador de carga pode manter o usuário conectado em apenas um nó computacional, não sendo necessário a utilização de *tokens* ou a centralização dos dados contidos em memória. Esta é a abordagem menos recomendada devido a uma possível má distribuição de carga entre os nós.

Com o controle baseado em *cookie*, quando um usuário tenta acessar uma outra aplicação que delegue sua autenticação para o mesmo IdP, não será necessário que ele forneça novamente as credencias, pois, uma sessão já foi estabelecida anteriormente, este cenário pode ser visto na figura 10.

[15] Cookies: dados enviados pelo servidor e armazenados no browser do usuário enquanto o mesmo visita um determinado web site. Em toda requisição feita pelo usuário estes dados são enviados para que o servidor possa recuperá-los e possa tomar algum tipo de ação.
[16] Memcached: Sistema de *chache* de objetos em memória distribuído, livre, de código aberto e de alta performance.

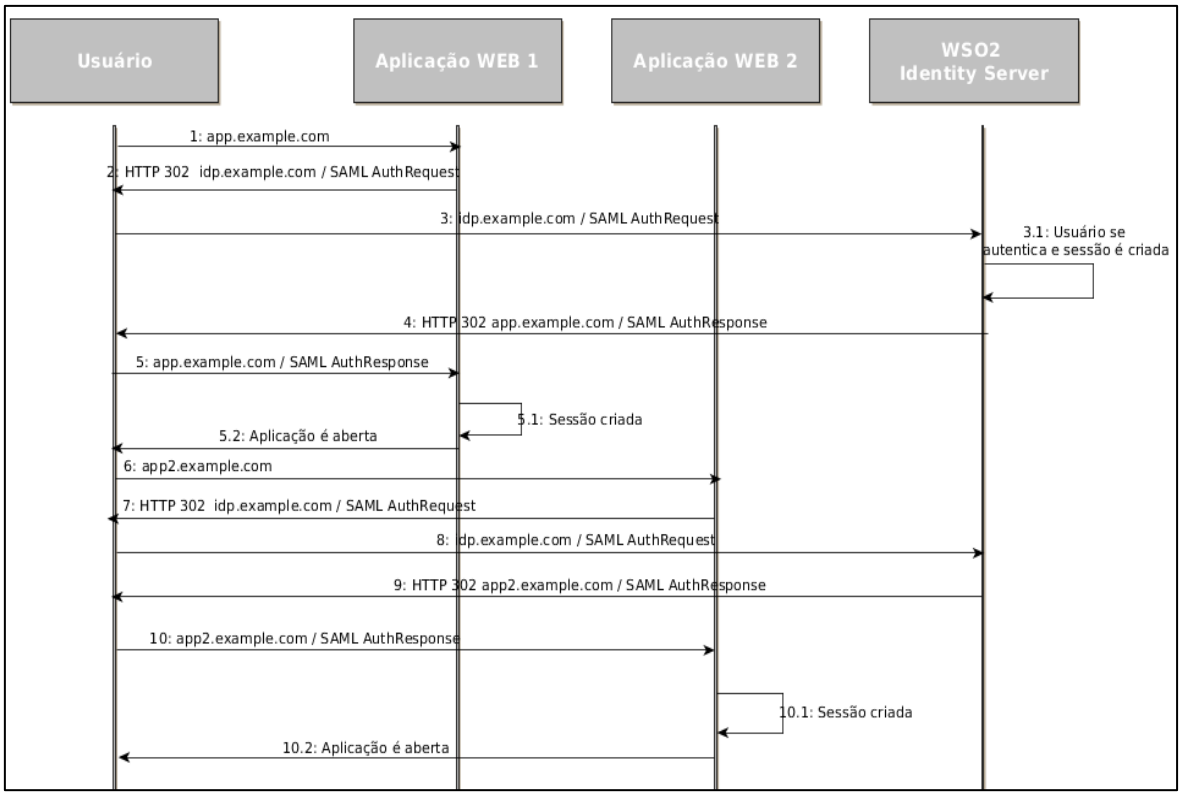

Figura 10: Fluxo de acesso a duas aplicações e demonstração do Single Sign-On.
Fonte: Autores.

Conforme discutido anteriormente, o processo de autenticação utilizando SAML é baseado em uma série de redirecionamentos HTTP e mensagens, nas quais o IdP e SP trocam informações para validar a identidade do usuário. As sessões abaixo detalham cada uma das mensagens utilizadas e seus principais atributos.

3.3.1. Authentication Request

Quando uma requisição de acesso chega à aplicação, a mesma verifica se o usuário já possui uma sessão ativa ou não. Neste trabalho, foi utilizado o *framework* Spring Security[17] com o intuito de auxiliar e abstrair algumas complexidades no desenvolvimento da aplicação. Conforme pode ser visto na figura 11, um filtro HTTP é configurado na aplicação com o intuito de interceptar todas as requisições que chegam à aplicação.

```
<!-- Spring Security Filter(s) -->
<filter>
    <filter-name>springSecurityFilterChain</filter-name>
    <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
</filter>

<filter-mapping>
    <filter-name>springSecurityFilterChain</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

Figura 11: Arquivo web.xml da aplicação com a configuração do filtro HTTP.
Fonte: Autores.

[17] Spring Security: Framework desenvolvido pela Spring que permite a criação de aplicações Java de forma segura. www.spring.io

Desta maneira, caso não haja uma sessão ativa do usuário, o Spring Security executa a criação de uma mensagem SAML e redireciona o mesmo para o IdP. Na *tag* "*url-pattern*" é possível visualizar que todas as URLs (<url-pattern>/*</urlpattern>) acessadas na aplicação serão interceptadas pelo framework Spring Security.

Na figura 12 pode-se ver a requisição que a aplicação envia ao IdP. Na mensagem enviada são encontradas informações relativas ao IdP e a própria aplicação. A tag "*issuer*" faz indicação à identidade da aplicação que esta requisitando a autenticação, desta forma o IdP pode verificar se esta aplicação faz parte de uma federação ou não, podendo assim aceitar ou negar pedidos de autenticação de origens desconhecidas. Esta característica é uma das principais diferenças em relação ao OpenID Connect apresentado na sessão anterior.

```
<saml2p:AuthnRequest xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol"
    AssertionConsumerServiceURL=
    "http://localhost:8180/appwebsso/saml/SSO/alias/appwebsso"
    Destination="https://localhost:9443/samlsso" ForceAuthn="false"
    ID="a1f0ig9e11jg3029488h3a2bigi0igi" IsPassive="false"
    IssueInstant="2015-06-26T21:49:17.771Z"
    ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
    Version="2.0">
    <saml2:Issuer
        xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">appwebsso</saml2:Issuer>
</saml2p:AuthnRequest>
```

Figura 12: SAML Authentication Request.
Fonte: Autores.

Na mensagem apresentada na figura 12, pode-se encontrar a identificação "appwebsso", este nome é utilizado para identificar a aplicação junto ao IdP. Nas configurações do IdP é necessário fazer o cadastro desta aplicação para que o processo de autenticação funcione corretamente.

Além da identificação da aplicação configurada a partir do campo "*issuer",* esta interface permite a configuração da "*Assertion Consumer URL*". Esta URL é utilizada pelo IdP para enviar a mensagem de resposta com a confirmação da autenticação.

3.3.2. Authentication Response

Após o redirecionamento e a execução do processo de autenticação, o IdP realiza a montagem da resposta baseada nas configurações da aplicação e nos dados do usuário. Na figura 13, pode-se ver um exemplo de mensagem de resposta.

```
<?xml version="1.0" encoding="UTF-8"?>
<saml2p:Response
    Destination="http://localhost:8180/appwebsso/saml/SSO/alias/appwebsso"
    ID="mkkkgnpegdgcgonogmobfeeaioognekkjdbjjplm"
    InResponseTo="ajh0c40g0a31b781cg3d71ee3ef101"
    IssueInstant="2015-06-26T22:10:26.381Z" Version="2.0"
    xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol">
    <saml2:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity"
        xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
        https://localhost:9443/samlsso</saml2:Issuer>
    <saml2p:Status>
        <saml2p:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success" />
    </saml2p:Status>

    ...

</saml2p:Response>
```

Figura 13: SAML Authentication Response.
Fonte: Autores.

Dentre algumas *tags* é importante analisar as tags "I*ssuer*" e "*Status*". A *tag* "*Issuer*" nos remete a identificação do IdP. A aplicação utiliza esta identificação para confirmar se a resposta esta vindo de um servidor confiável de tal forma que ela possa aceitar ou não a mensagem. Já na *tag* "*Status*" podemos encontrar a confirmação de sucesso, que define que o usuário efetuou o processo de autenticação com sucesso.

Apesar de não demonstrado na mensagem de exemplo, a mensagem SAML de resposta contém também informações referentes à validade da mensagem, desta forma a aplicação poderá desconsiderar mensagens com um tempo de validade expirado.

3.3.3. Attribute Statement

Ao se configurar uma aplicação no Idp, conforme discutido anteriormente, é possível adicionar atributos do usuário como parte da resposta enviada a aplicação após o processo de autenticação. Estas informações podem ser utilizadas internamente na aplicação para validação de perfis de acesso ou até mesmo para apresentar informações em tela para o usuário. Como parte da resposta SAML, ao verificar que a aplicação possui atributos configurados, o IdP adiciona os atributos do usuário e os envia a aplicação. Na figura 14, podemos ver um exemplo de atributos que podem ser enviados, como nome de usuário, perfis e e-mail.

```
<saml2:AttributeStatement>
    <saml2:Attribute Name="http://wso2.org/claims/role">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="xs:string">admin,Internal/everyone</saml2:AttributeValue>
    </saml2:Attribute>
    <saml2:Attribute Name="http://wso2.org/claims/emailaddress">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="xs:string">admin@wso2.com</saml2:AttributeValue>
    </saml2:Attribute>
    <saml2:Attribute Name="http://wso2.org/claims/fullname">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="xs:string">admin</saml2:AttributeValue>
    </saml2:Attribute>
</saml2:AttributeStatement>
```

Figura 14: SAML *Authentication Response* com os atributos de usuário.
Fonte: Autores.

Como discutido anteriormente, as informações referentes ao usuário são armazenadas em uma base de dados externa como por exemplo um LDAP, desta maneira o IdP apenas faz uma consulta nesta base e monta suas respostas baseadas no resultado destas.

3.3.4. Single Logout

Durante este trabalho foi discutido diversos conceitos e mecanismos de implementação em torno de SSO. De forma resumida, SSO provê um mecanismo para que o usuário se autentique uma única vez e possa fazer uso dos sistemas pertencentes a um domínio federado.

Porém, existe uma outra funcionalidade tão importante quanto e que não foi levantada durante o trabalho, sendo ela, a capacidade de efetuar o *logout* do usuário nos sistemas que o mesmo utilizou durante uma sessão. O *logout* de uma forma pragmática é o processo no qual uma aplicação executa a invalidação de uma sessão de usuário após a saída do mesmo do sistema. A invalidação da sessão, é um aspecto importante para a segurança da informação, pois uma sessão ativa pode ser utilizada por um atacante para ter acesso indevido a um sistema.

Assim como é importante a propagação da sessão do usuário no processo de autenticação, se faz necessário a invalidação da mesma em todas as aplicações que o usuário utilizou durante a validade da mesma. SAML Web Browser SSO Profile provê um processo para que a sessão do usuário seja invalidada em um determinado domínio.

Em uma aplicação convencional, ao executar o *logout* a mesma irá invalidar de forma local aquela sessão. A forma mais comum é apagar um registro de identificação relacionado aquela sessão, o que normalmente esta associado a um *cookie* do usuário.

Utilizando SSO, precisamos de alguma forma de notificar todas as aplicações para que elas invalidem a sessão de um determinado usuário. Quando o usuário efetua o *logout* em uma aplicação, além de ser invalidado a sua sessão referente aquela aplicação a mesma envia um pedido de *logout* ao IdP para que o mesmo possa notificar as outras aplicações envolvidas.

Na figura 15, pode-se ver um pedido de *logout* onde a aplicação envia ao IdP um pedido para que o mesmo invalide a sessão do usuário. É importante notar que *tag* "*SessionIndex*" contém a identificação da sessão do usuário, desta forma quando o IdP recebe essa conexão ele sabe quais aplicações ele deve notificar.

```
<?xml version="1.0" encoding="UTF-8"?>
<saml2p:LogoutRequest xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol"
    Destination="https://localhost:9443/samlsso"
    ID="a4dbbggj5icb4i7g4lbh0jj71g02ghe"
    IssueInstant="2015-06-26T22:19:35.320Z" Version="2.0">

    <saml2:Issuer
        xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">appwebsso</saml2:Issuer>
    <saml2:NameID xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"
        Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">admin</saml2:NameID>
    <saml2p:SessionIndex>a0d36c18-42f9-4ffd-8bb8-4e798fde995f</saml2p:SessionIndex>

</saml2p:LogoutRequest>
```

Figura 15: SAML *Logout Request*.
Fonte: Autores.

Após o pedido de *logout*, o IdP envia a mesma requisição de *logout* para as outras aplicações que o usuário fez uso e após esta notificação ele faz a notificação para a aplicação que fez a requisição inicial, conforme a figura 16.

```
<?xml version="1.0" encoding="UTF-8"?>
<saml2p:LogoutResponse ID="omalmihandjjilkkdkpgeikohhkgffcoibalgffg"
    InResponseTo="a4dbbggj5icb4i7g41bh0jj71g02ghe"
    IssueInstant="2015-06-26T22:19:35.455Z"
    Version="2.0"
    xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol">

    <saml2:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity"
        xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
            https://localhost:9443/samlsso</saml2:Issuer>
    <saml2p:Status>
        <saml2p:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success" />
    </saml2p:Status>

</saml2p:LogoutResponse>
```

Figura 16: SAML Logout Response.
Fonte: Autores.

Além do IdP, as aplicações também precisam de uma URL capaz de aceitar mensagens do tipo "LogoutRequest". O Spring Security já disponibiliza esta funcionalidade e ao receber este tipo de mensagem efetua o *logout* automático da aplicação.

3.4. Habilitando suporte a Integridade e Confidencialidade

Um conceito amplamente difundido na segurança da informação é a tríade CID (confidencialidade, integridade e disponibilidade). (STALLINGS; BROWN, 2014).

Integridade é o termo relacionado a defesa contra modificação ou destruição inapropriada da informação sem que os envolvidos no processo não tenham condições de identificar tal ação. Quando a informação passa por um processo de alteração por uma pessoa não autorizada, este dado não é mais considerado como íntegro.

A confidencialidade é o meio que se fornece para que a informação não seja acessada por pessoas indevidas, permitindo assim um maior nível de privacidade.

Dentre as possíveis vulnerabilidades relacionadas à integridade de mensagens SAML, pode-se citar a elevação de privilégios[18]. Quando um usuário faz a sua autenticação, uma resposta SAML é enviada ao seu browser e o mesmo pode fazer alterações na mesma sem nenhuma dificuldade. Este fato acontece porque as respostas SAML apenas são codificadas utilizando o padrão de codificação BASE64[19], com isto, um usuário mal-intencionado pode fazer a decodificação e inserir dados para que se ganhe níveis elevados de acesso.

Na figura 17 pode ser visto uma mensagem capturada por um interceptador de tráfego com o intuito de verificação das mensagens trocadas entre o browser do usuário e o servidor. Na imagem é possível visualizar o "SAMLResponse", o mesmo encontra-se codificado em BASE64 e utilizando uma ferramenta de conversão é possível obter a mensagem SAML em formato XML, alterá-la e enviá-la ao servidor.

[18] Elevação de privilégios: permite a um usuário mal-intencionado adquirir permissões de autorização além daquelas concedidas inicialmente. Um invasor poderia por exemplo, promover seu acesso de "somente leitura" para "leitura e gravação" em algum determinado recurso do sistema.
[19] BASE64: Método para codificação de dados para transferência de dados na Internet.

Figura 17: Mensagem SAML interceptada em formato BASE64.
Fonte: Autores.

Para evitar esta vulnerabilidade, é recomendado a assinatura das mensagens trafegadas. O WSO2 Identity Server e o framework Spring Security fornecem mecanismos para que esta configuração seja aplicada. Utilizando um PKI (*Public Key Infrastrcture*) basta habilitar esta configuração no IdP e na aplicação para que a mesma trafegue suas mensagens de forma mais segura e atenda as preocupações básicas relacionadas a segurança da informação. Na figura 18 pode ser visto a configuração habilitada no IdP.

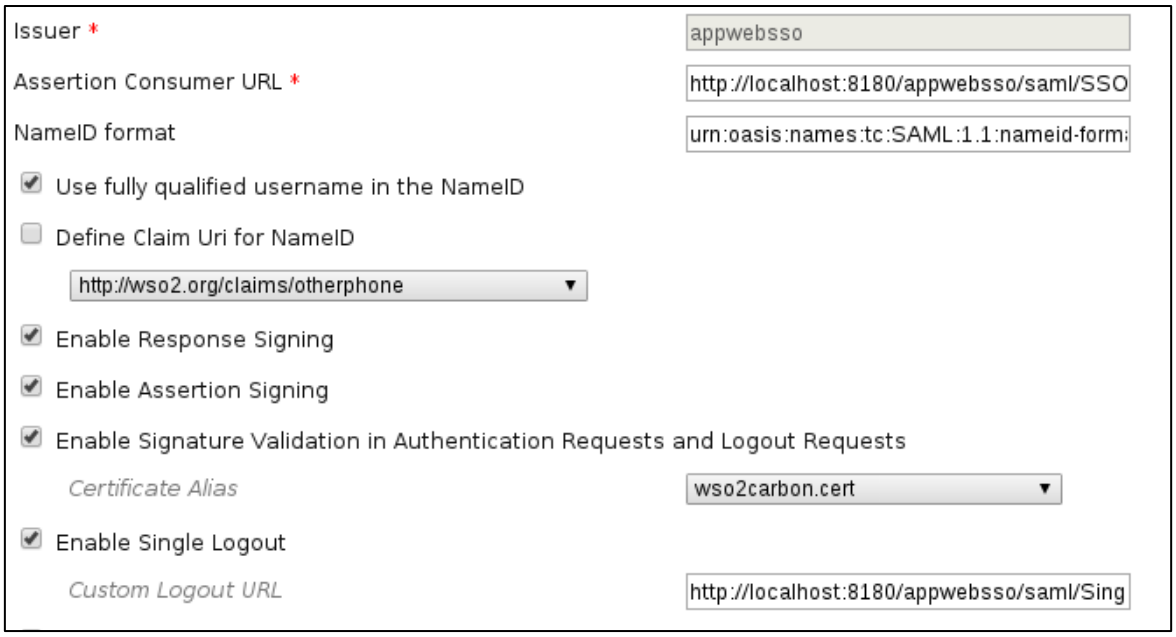

Figura 18: Configuração de certificados habilitados no IdP.
Fonte: Autores.

Na aplicação, o arquivo "metada_sp.xml" é responsável pelas configurações de certificado. Na imagem 19 pode ser verificado um trecho desta configuração.

```
<md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
    entityID="appwebsso">
    <md:SPSSODescriptor AuthnRequestsSigned="false"
        WantAssertionsSigned="false" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
        <md:KeyDescriptor use="signing">
            <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
                <ds:X509Data>
                    <ds:X509Certificate>MIIDUjCCAjqgAwIBAgIEUOLIQTANBgkqhkiG9w0BAQUFADBrMQswCQYDVQQG
                        CBMHVXVzaW1hYTERMA8GA1UEBxMISGVsc2lua2kxGDAWBgNVBAoTD1JNNSBTb2Z0d2FyZSBPeTEM
                        MAoGA1UECwwDUiZEMQ8wDQYDVQQDEwZhcG9sbG8wHhcNMTMwMTAxMTEyODAxWhcNMjIxMjMwMTEy
                        ODAxWjBrMQswCQYDVQQGEwJGSTEQMA4GA1UECBMHVXVzaW1hYTERMA8GA1UEBxMISGVsc2lua2kx
                        GDAWBgNVBAoTD1JNNSBTb2Z0d2FyZSBPeTEMMAoGA1UECwwDUiZEMQ8wDQYDVQQDEwZhcG9sbG8w
                        ggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCXqPOwqL2Ai1haeTj0alwsLafhrDtUt00E
                        5xc7kdD7PISRA270ZmpYMB4W24Uk2QkuwaBp6dI/yRdUvPfOT45YZrqIxMe2451PAQWtEKWF5Z13
                        FOJ4/lB71TtrzyH94RnqSHXFfvRN8EY/rzuEzrpZrHdtNs9LRyLqcRTXMMO4z7QghBuxh3K5gu7K
                        qxpHx6No83WNZj4B3gvWLRWv05nbXh/F9YMeQClTX1iBNAhLQxWhwXMKB4u1iPQ/KSaal3R26pON
                        UUmu1qVtU1quQozSTPD8HvsDqGG19v2+/N3uf5dRYtvEPfwXN3wIY+/R93vBA6lnl5nTctZIRsyg
                        8rxj3Jhxi2Vb/MJn7XzeVHHLzA1sV5hwl/2PLnaL2h9WyG9QwBbwtmkMEqUt/dgixKb1Rvby/tBu
                        RogWgPONNSACiW+Z5o8UdAOqNMZQozD/i1gOjBXoF0F5OksjQN7xoQZLj9xXefxCFQ69FPcFDeEW
                        bHwSoBy5hLPNALaEUoa5zPDwlixwRjFQTc5XXaRpgIjy/2gsL8+Y5QRhyXnLqgO67BlLYW/GuHE=
                    </ds:X509Certificate>
                </ds:X509Data>
            </ds:KeyInfo>
        </md:KeyDescriptor>

        ...

    </md:SPSSODescriptor>
</md:EntityDescriptor>
```

Figura 19: Configuração do arquivo “metada_sp.xml” com suporte a assinatura das mensagens SAML.
Fonte: Autores.

Após a configuração dos certificados na aplicação e no IdP, as mensagens trafegadas entre ambos passam a possuir mecanismos de integridade e confidencialidade, o que aumenta o nível de segurança da aplicação.

4. Conclusão e trabalhos futuros

Neste trabalho foi apresentado uma arquitetura SSO baseada em elementos como IdP e SP na qual a aplicação delega seu processo de autenticação a partir do protocolo SAML para o IdP, para que o mesmo gerencie este processo de autenticação. Foi possível verificar que a utilização de protocolos padrões de mercado e ferramentas *Open Source* fornecem os mecanismos necessários para a implementação de um ambiente SSO e que esta abordagem oferece diversos benefícios se comparado com abordagens tradicionais, como autenticação única e melhor gerenciamento de políticas de acesso.

É importante salientar que as utilizações de ferramentas já desenvolvidas pela comunidade ajudam na validação de possíveis lacunas de implementação, visto que tais já foram utilizadas, assim se encontram em um nível mais maduro e menos suscetíveis a erros de implementação, o que poderia levar a vulnerabilidades.

Para trabalhos futuros, sugere-se o desenvolvimento de um *proxy* reverso que atue como um intermediador entre as aplicações e o usuário. O trabalho desenvolvido nesta pesquisa leva em consideração aplicações que são desenvolvidas internamente em uma corporação e que desta forma possuem a possibilidade de delegar o seu processo de autenticação.

Em ambientes corporativos, é muito comum a compra de softwares que em grande parte não possuem a funcionalidade de delegação de autenticação, o que impede a criação de um ambiente SSO. Desta forma, utilizando um proxy reverso com suporte a SAML o mesmo será responsável pela interceptação e execução do processo de autenticação junto a aplicação.

5. Referências

BOYD, Ryan. Getting Started with OAuth 2.0. O'Reilly. Sebastopol. 2012.

IETF. The OAuth 2.0 Authorization Framework. Disponível em: http://tools.ietf.org/html/rfc6749. Acesso em: 07 abr. 2015

IETF. Hypertext Transfer Protocol -- HTTP/1.1. Disponível em: http://tools.ietf.org/html/rfc2616. Acesso em: 07 abr. 2015

IETF. Internet Security Glossary. Disponível em: https://www.ietf.org/rfc/rfc2828.txt. Acesso em: 07 abr. 2015

IETF. Lightweight Directory Access Protocol (LDAP): The Protocol. Disponível em: https://tools.ietf.org/html/rfc4511. Acesso em: 07 abr. 2015

LAWRENCE, Kelvin; KALER, Chris. WS-Trust 1.3. Disponível em: http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.pdf. Acesso em: 07 abr. 2015.

OASIS. Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0. Disponível em: http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf. Acesso em: 07 abr. 2015

OIDF. OpenID Connect Core 1.0. Disponível em: http://openid.net/specs/openid-connect-core-1_0.html. Acesso em: 07 abr. 2015.

ROSEMBERG, Jothy; REMY, David. Securing Web Services with WS-Security: Demystifying WS-Security, WS-Policy, SAML, XML Signature, and XML Encryption. SAMS. Indianapolis. 20 04.

SIRIWARDENA, Prabath. Advanced API Security: Securing APIs with OAuth 2.0, OpenID Connect, JWS and JWE. Apress. New York. 2014.

STALLINGS, William; BROWN, Lawrie. Segurança de Computadores: Princípios e Práticas. 2.ed. Campus. Rio de Janeiro. 2014.