Uma Arquitetura de Segurança para Mecanismos de Controle de Acesso Baseados em Serviços *Web*

Ricardo T. Macedo[1], **Bruno A. Mozzaquatro**[1], **Luiz D. Biazus Neto**[1], **Raul C. Nunes**[1]

[1]Centro Tecnológico – Universidade Federal de Santa Maria (UFSM)
Caixa Postal 15.064 – 91.501-970 – Santa Maria – RS – Brasil

{rmacedo,brunomozza,ceretta}@inf.ufsm.br, luizbiazus@mail.ufms.br

Abstract. *In examining efforts to improve the mechanisms for access control, there is the adoption of Web services technology to ensure interoperability across heterogeneous technology domains. However, there are reported concerns about the confidentiality of requests and authorizations that pass through the channel of communication. This paper proposes a security architecture that determines how messages should be formed, transported and processed in order to inhibit attacks on the confidentiality of information managed. It presents a proof of concept with the development of a prototype based on WS-Security, WS-BPEL and WS-Policy, environmental testing and feasibility of use.*

Resumo. *Ao analisar esforços para aperfeiçoamento dos mecanismos de controle de acesso, nota-se a adoção da tecnologia de serviços Web para garantir interoperabilidade entre domínios tecnológicos heterogêneos. No entanto, não são relatadas preocupações quanto à confidencialidade das requisições e autorizações que trafegam pelo canal de comunicação. Este artigo propõe uma arquitetura de segurança que determina como as mensagens devem ser formadas, transportadas e processadas, de modo a inibir ataques contra a confidencialidade da informação gerenciada. Apresenta-se uma prova de conceitos com o desenvolvimento de um protótipo, com base nas especificações WS-Security, WS-BPEL e WS-Policy, ambiente de testes e viabilidade de uso.*

1. Introdução

Modelos de controle de acesso visam gerenciar a revelação de informações e a utilização de recursos digitais, considerando propriedades tais como confidencialidade, integridade e disponibilidade. O gerenciamento é realizado de forma a prevenir a revelação não autorizada dos dados e recursos computacionais (confidencialidade) e inibir modificações não autorizadas da informação (integridade), enquanto assegura que os dados e recursos gerenciados estarão aptos para utilização quando solicitados por entidades autorizadas (disponibilidade) [Lazouski et al. 2010].

A adoção da tecnologia serviços web (Web Services) (WS) [w3c] em mecanismos de controle de acesso proporciona gerenciamento de acesso modular com fraco acoplamento entre domínios tecnológicos heterogêneos [Bertino et al. 2008] e vem sendo empregada por vários autores ([Camy et al. 2005], [Soares et al. 2006], [Han et al. 2009]). Entretanto, preocupações quanto a possíveis ameaças contra a integridade e confidencialidade das requisições e autorizações de acesso (comunicadas através de mensagens *Simple*

Object Access Protocol (SOAP)) não são relatadas. Sem uma Arquitetura de Segurança especificando como as mensagens devem ser formadas, transportadas e processadas de modo seguro, um atacante pode comprometer a confidencialidade dos dados gerenciados ao manipular requisições e autorizações de acesso.

A *Web Service Security Language*, também conhecida como *WS-Secuity* (WSS) [Nadalin et al. 2005], é o padrão de *facto* ao prover segurança na troca de mensagens SOAP [Bertino et al. 2010]. A WSS trata-se de uma linguagem para descrever a segurança em interações fim-a-fim de uma única mensagem [Nadalin et al. 2006], para isso provê mecanismos como *tokens* seguros, assinatura e criptografia, de modo que a mensagem não seja adulterada no meio de transporte. No entanto, a especificação por si só não provê uma solução completa para segurança de serviços *Web*, ela serve como um módulo que pode ser acoplado com outras especificações formando modelos de segurança para aplicações *Web* específicas [Nadalin et al. 2006].

Este trabalho apresenta uma Arquitetura de Segurança para mecanismos de controle de acesso que utilizam a tecnologia WS determinando como as requisições e autorizações de acesso devem ser formadas, transportadas e processadas de modo seguro, tanto para o módulo de controle de acesso, quanto para a aplicação que possui seus dados gerenciados. De forma que as requisições e autorizações de acesso que trafegam pela rede estejam imunes a ataques que comprometam o modelo de gerenciamento empregado no mecanismo baseado em WS, tais como *Tampering* [Bertino et al. 2010] e *XML Signature Element Wrapping* [Gruschka and Iacono 2009]. A organização dos tópicos é apresentada como segue: a seção 2 apresenta modelos de controle de acesso baseados na tecnologia WS, logo após, a seção 3 realiza uma discussão sobre vulnerabilidades associadas à troca de requisições e autorizações de acesso. A seção 4 apresenta a Arquitetura de Segurança, na seção 5 é apresentada a prova de conceitos com a implementação de um protótipo baseado nas especificações WS-Security, WS-BPEL e WS-Policy, enquanto que a seção 6 conclui o trabalho.

2. Modelos de Controle de Acesso Baseados em *Web Services*

Esta seção apresenta modelos de controle de acesso desenvolvidos com a tecnologia WS. Os modelos analisados estão organizados de forma que a 2.1 apresenta o modelo de Controle de Acesso Baseado em Expressões Contextuais, a seção 2.2 o modelo de Controle de Acesso para Sistemas Unidos em Comércio Colaborativo, e a seção 2.3 Uma Arquitetura Semântica de Segurança para serviços *Web*.

2.1. Modelo de Controle de Acesso Baseado em Expressões Contextuais

O modelo de Controle de Acesso Baseado em Expressões Contextuais [Macedo et al. 2010], conhecido como CABEC, foi desenvolvido visando obter flexibilidade e fina granularidade no gerenciamento da informação, considerando o contexto (ctx) contido no ambiente da entidade requisitante ($E1$) e requisitada ($E2$). A abstração desse contexto é armazenada em Expressões Contextuais (EC), codificadas com a *eXtensible Access Control Language* (XACML). Para provar seus conceitos foi realizada uma implementação baseada na tecnologia WS integrada ao portal *Geodesastres* [1]. Uma análise dos passos realizados pelo CABEC ao conceder o acesso é ilustrado na Figura 1 .

[1]Portal pertencente ao Instituto Nacional de Pesquisas Espaciais para disponibilizar imagens sobre desastres naturais.

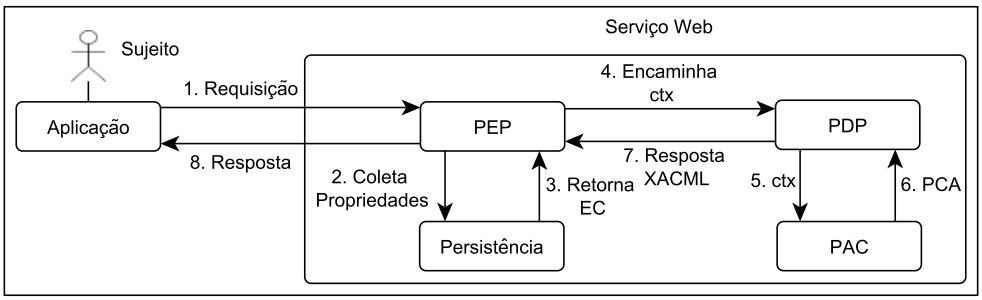

Figura 1. Funcionamento do CABEC durante uma Requisição de acesso.Adaptado de [Macedo et al. 2010]

Observando a Figura 1, nota-se que o CABEC é um WS que possui quatro elementos para conceder o acesso: *Policy Enforcement Point* (PEP), *Policy Decision Point* (PDP), *Policy Access Control* (PAC) e Persistência. Quando um sujeito - S solicita acesso aos dados e recursos da aplicação gerenciada - A, A encaminha as credenciais de S ao elemento PEP (1), este realiza uma busca na camada de persistência de dados formando a $EC \subset ctx \in E1$ e $E2$ (2 e 3), então o PEP encaminha o ctx da requisição para o PDP (4) que confronta com o ctx armazenado em PAC (5 e 6), o resultado desse confronto é retornado ao PEP (7), que por sua vez retorna o resultado para A (8).

2.2. Modelo de Controle de Acesso para Sistemas Unidos em Comércio Colaborativo

O Modelo de Controle de Acesso para Sistemas Unidos em Comércio Colaborativo [Han et al. 2009], é formado por elementos de modelos tradicionais, tais como *Role-Based Access Control* (RBAC) [Ravi et al. 1996] e *Task-Based Authentication Control* (TBAC) [Thomas and Sandhu 1997]. Suas políticas são compostas pelo *Attribute-Based Access Control* (ABAC) [Yuan and Tong 2005] e associadas ao *Automated Trust Negotiation* (ATN) [Winsborough et al. 2000]. O mecanismo para gerenciamento de acesso é organizado como uma *Service-Oriented Architecture* (SOA), composta pelo WS *Gerenciador Central de Regras* (GCR), *Schedule Central da Tarefa* (SCT), com suporte a bancos de dados auxiliares e *Centro Negociativo de Políticas* (CNP). A Figura ilustra a interação entre os WS 2.

A Figura 2 demonstra como ocorre o fluxo de informações entre os WS que compõe a arquitetura do mecanismo ao receber uma requisição externa. Onde uma Organização Colaborativa (OC) invoca uma requisição para o serviço de outra OC (1), O *SOAP Message Handler* traduz a mensagem de solicitação e encontra o serviço solicitado externo (2). Se houver um contrato correspondente, é verificada sua validade (3), se a verificação for bem sucedida, começa a comparação entre a política e a negociação (4), quando a negociação termina, a informação de *feedback* CNP retorna à GCR (5). Se a negociação for bem sucedida, então SCT estabelece instâncias das tarefas requeridas e transfere o risco para um usuário com poderes de supervisão a tarefa de autorizar ou negar o acesso (6).

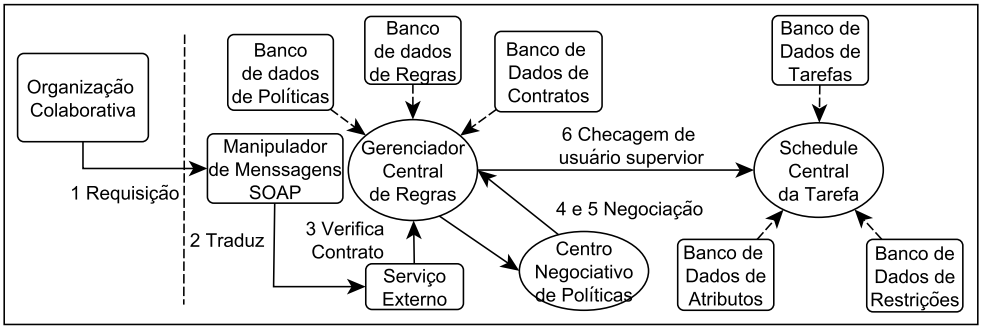

Figura 2. Funcionamento do United Access Control durante uma Requisição de acesso. Adaptado de [Han et al. 2009]

2.3. Uma Arquitetura Semântica de Segurança para serviços *Web*

A Semantic Security Architecture for Web Services [Dürbeck et al. 2010] trata-se de uma solução proposta à necessidade dos órgãos públicos da União Européia ao compartilhar serviços eletronicamente, através do portal *eGov*, sem comprometer o sigilo de dados confidenciais. Para isto, apresenta-se um mecanismo de controle de acesso baseado em um conjunto de WS, que formam uma arquitetura SOA, responsável por conceder ou negar solicitações de acesso, ilustrada na Figura 3.

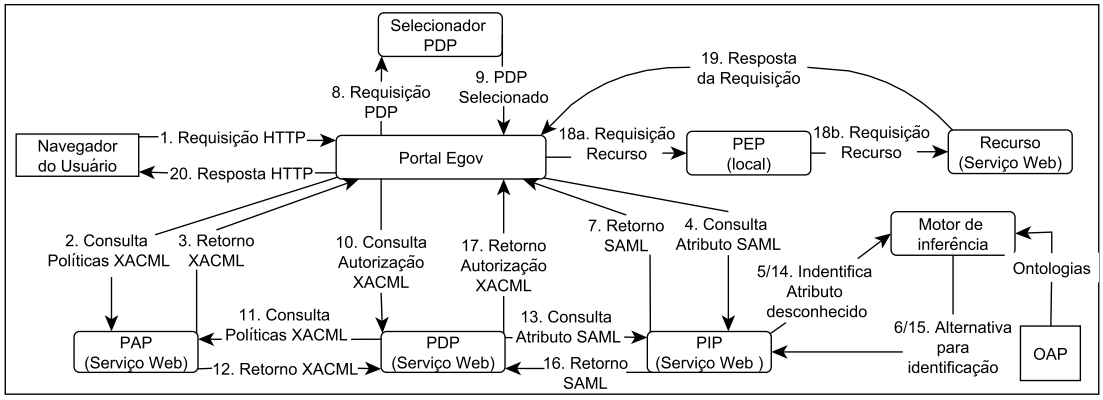

Figura 3. Arquitetura Semântica de Segurança para Serviços *Web* durante uma Requisição de acesso. Adaptado de [Dürbeck et al. 2010]

Na Figura 3, percebe-se um usuário encaminhando uma requisição para acessar o portal *eGov* (1), uma consulta ao WS que contém as políticas de controle de acesso (2), este retorna uma resposta contendo o resultado da consulta para o portal (3), então uma consulta ocorre para verificar se a mensagem foi emitida de um sistema confiável através do WS responsável por armazenar informações sobre a ontologia (4 e 5) uma alternativa para identificação é sugerida pelo motor de inferência (6).

O retorno dessa identificação é encaminhada ao portal *eGov* (7) que encaminha uma requisição ao *PDPSeletor* para selecionar um ponto para decisão de acesso (8 e 9), escolhido o ponto de decisão, é realizada uma consulta para autorização do acesso no ponto de decisão selecionado (10). Este realiza uma consulta nas políticas de acesso

(11 e 12), valida a identificação através de ontologias (13, 14, 15, e 16) para retornar a autorização de acesso (17), uma requisição ao recurso é realizado através do ponto de execução (18.a) ao WS que armazena os recursos (18.b), a resposta da requisição é encaminhada ao portal (19) que comunica o usuário (20).

3. Análise de Vulnerabilidades em Mecanismos desenvolvidos com a tecnologia WS

Conforme analisado anteriormente, a comunicação entre os serviços *Web* pertencentes a mecanismos de controle de acesso é realizada através da troca de requisições e autorizações de acesso através de mensagens SOAP e transportadas pelo *Hyper Text Transfer Protocol* (HTTP). Portanto, como se adota a tecnologia WS para facilitar a integração entre domínios tecnológicos heterogêneos, percebe-se que esse tráfego pode ocorrer através da Internet, o que torna possível a ocorrência de ataques com a manipulação de mensagens SOAP.

Para um melhor esclarecimento de como um atacante pode manipular as mensagens, apresentamos a Figura 4, que ilustra a estrutura de duas mensagens SOAP contendo respectivamente uma requisição e autorização de acesso.

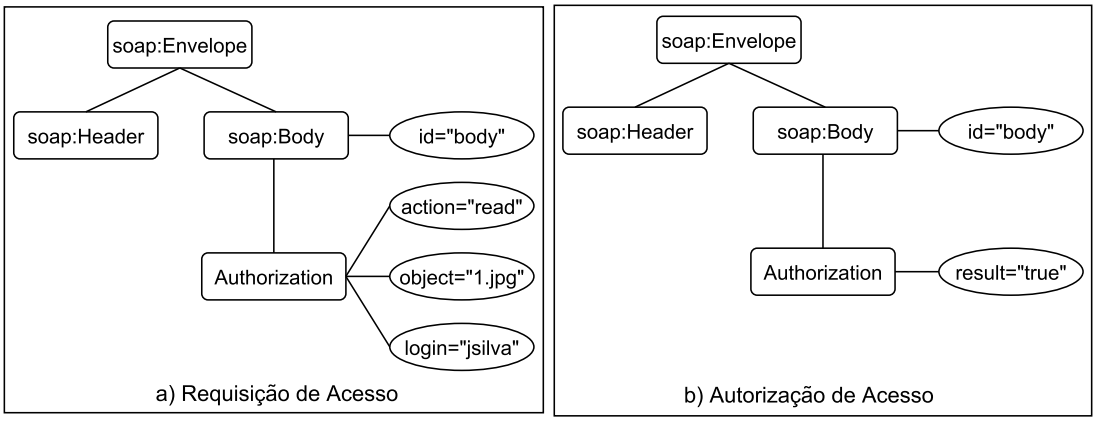

Figura 4. Estrutura de uma Requisição SOAP

Uma requisição de acesso contida em uma mensagem SOAP (Figura 4.a), é formada por um envelope, cabeçalho (*header*) e corpo (*body*), onde incorpora informações sobre o serviço solicitado, tais como nome (*authorization*) e informações referentes a uma solicitação de acesso declarando que o sujeito *jsilva* deseja ler (*read*) o objeto *1.jpg*. Já a autorização de acesso diferencia-se por apresentar a decisão do mecanismo, por exemplo: requisição autorizada (*result = true*).

A seguir, serão detalhadas duas ameaças que colocam em risco a comunicação dos sistemas gerenciados com os mecanismos implementados através de WS. Ataques (*Wrapping*) objetivam inserir elementos maliciosos na estrutura das mensagens (SOAP). Desta forma, ao processar tais elementos o mecanismo de controle de acesso concede ao atacante autorizações equivalentes a de um usuário legítimo que foi devidamente autenticado [Gajek et al. 2009]. Na seção 3.1 será demonstrada como uma mensagem SOAP pode ser manipulada a fim de modificar o resultado final da autorização. Já a seção 3.2,

apresenta a alteração na estrutura de uma mensagem SOAP permitindo que um atacante altere o corpo da mensagem, resultando em uma autorização não desejada.

3.1. *Tampering*

A interceptação de uma mensagem SOAP entre o serviço *Web* e o mecanismo de controle de acesso possibilita que o atacante manipule uma requisição ou autorização de acesso. Esse tipo de ameaça é denominada *Tampering* [Bertino et al. 2010]. As informações correspondentes à Figura 4, possibilitam ao atacante obter acesso não autorizado através de um mecanismo de controle de acesso baseado em WS. Os elementos que compõem a Figura correspondem ao identificador do sujeito (*jsilva*), recurso (ou objeto) solicitado (*1.jpg*), ação (*read*) e decisão *result = true*. No caso de um atacante capturar o tráfego dessas mensagens ele tem condições de lê-las, modificá-las e reenviá-las para o processamento visando obter acesso não autorizado, ou ainda confundir o modelo de controle de acesso ao alterar uma requisição que trafega na rede, de modo que o módulo gerenciador processe e encaminhe para o requisitante uma autorização incorreta. Para ilustrar como essa ameaça pode prejudicar o gerenciamento da informação, apresenta-se a Figura 5 que ilustra a mensagem original (5.a) e adulterada (5.b).

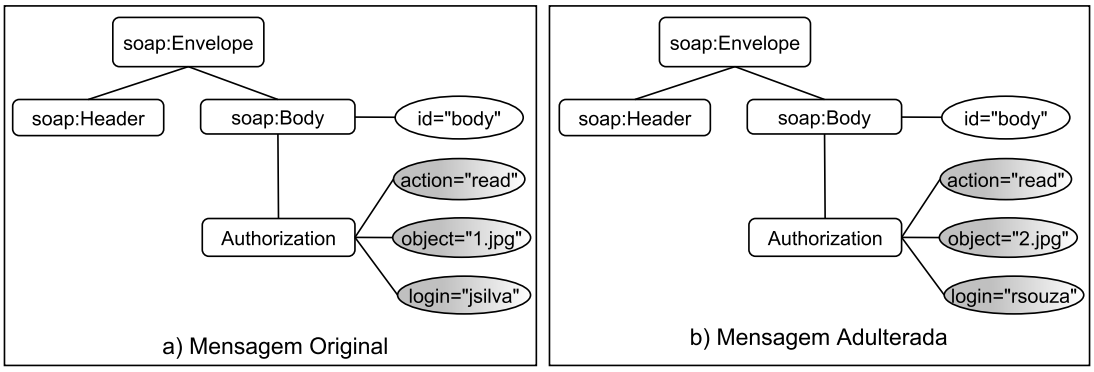

Figura 5. Ataque Tampering

Tendo como base os mecanismos descritos na seção 2, nota-se que todos possuem uma interface entre o mecanismo de controle de acesso e aplicação gerenciada (ou solicitante da autorização) cuja comunicação dá-se por meio de um serviço *Web* através de uma rede de computadores. A Figura 5.a ilustra uma requisição de acesso onde um usuário *jsilva* solicita acesso ao objeto *1.jpg* com permissão de leitura *read*. No entanto, como esse sujeito capturou fluxo HTTP e possui conhecimento que o usuário *rsouza* tem permissão de leitura *read* no objeto *2.jpg* ele manipula sua requisição de acesso de modo a ficar como ilustrado na Figura 5.b burlando o gerenciamento da informação ao comprometer a integridade e confiabilidade da requisição de acesso. Desse modo, nota-se que independente do modelo de gerenciamento da informação adotado, o mecanismo torna-se ineficaz pela manipulação das mensagens SOAP.

3.2. *XML Singnature Wapping Element*

O simples uso dos mecanismos disponíveis pela especificação não provêm integridade e confidencialidade contra ataques *XML Signature Wrapping Element*

[Gruschka and Iacono 2009]. A especificação de segurança WSS provê mecanismos para comunicação baseada em *tokens* [2] seguros, assinatura e criptografia de elementos ou documentos XML. O uso desses mecanismos de segurança inviabiliza ataques como *Tampering*, pois possibilita que elementos como corpo e cabeçalhos da mensagem SOAP tenham a propriedade de não repúdio. No entanto, o ataque *XML Signature Wrapping Element* possibilita que um atacante consiga burlar um serviço *Web* ao anexar ao cabeçalho da mensagem um segundo corpo SOAP contendo a requisição alterada, conforme ilustrado na Figura 6.

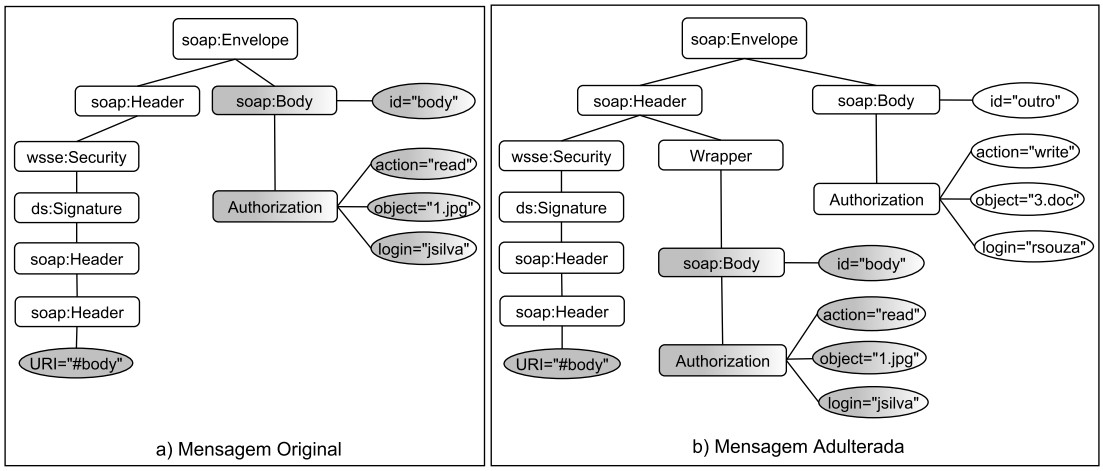

Figura 6. Ataque XML Signature Wrapping

Considera-se que um mecanismo de controle de acesso foi projetado com a especificação de segurança WSS, com o objetivo de prover integridade e confidencialidade as mensagens que trafegam pela rede de computadores. Esse, então, assinou o cabeçalho e criptografou o corpo da mensagem SOAP que contém os atributos das requisições e autorizações de acesso, como pode ser visualizado na Figura 6.a uma mensagem com essas características. Um ataque *XML Signature Wrapping Element* pode comprometer o gerenciamento da informação manipulando as mensagens criptografadas e assinadas, pois essa manipulação dá-se pelo fato que o mecanismo baseado em WS verifica a integridade e confidencialidade dos campos criptografados e não se preocupa com a adição de um novo corpo SOAP anexado à mensagem original, como mostra a Figura 6.b. Dessa forma o mecanismo acaba processando os dois corpos da mensagem SOAP, possibilitando as mesmas técnicas citadas sobre *Tampering*.

4. Arquitetura de Segurança para Mecanismos de Controle de Acesso Baseados em Serviços *Web*

Esta seção apresenta a Arquitetura de Segurança para Mecanismos de Controle de Acesso Baseados em Serviços *Web*. A seção 4.1 apresenta a modelagem arquitetural para prover integridade e confidencialidade as requisições e autorizações a mecanismos de controle de acesso baseados em WS, a seção 4.2 apresenta o algoritmo que realiza o procedimento

[2] É um segmento de texto ou símbolos que podem ser manipulados por um parser (interpretador de tokens).

de envio de uma autorizações e requisições, enquanto que na seção 4.3 é apresentado uma demonstração de validação através de um sistema dedutivo.

4.1. Definições e Conceitos

A Arquitetura de Segurança para Mecanismos de Controle de Acesso Baseados em Serviços *Web* objetiva prover integridade e confidencialidade para requisições e autorizações de acesso, inviabilizando a manipulação das mensagens SOAP que trafegam pelo meio de transmissão. Para solucionar esse desafio, considera-se que somente a utilização dos mecanismos de criptografia e assinatura das mensagens providos pela especificação WSS, apesar de serem capazes de inibir ataques *Tampering*, são insuficientes para tornar as mensagens SOAP imunes à ataques *XML Signature Wrapping*.

Entende-se que para proporcionar a confiança em mecanismos gerenciadores de acesso baseados em WS é necessário identificar padrões no fluxo de mensagens SOAP com base em premissas peculiares ao gerenciamento da informação. Baseado nelas criou-se uma modelagem especificando políticas que descrevam como devem ser formadas, criptografadas e assinadas as requisições e autorizações de acesso. Através da análise da seção 2, considera-se como premissas o formato da requisição e autorização de acesso e a heterogeneidade tecnológica.

A premissa da heterogeneidade tecnológica, origina-se pelo fato que as modelagens que adotam a tecnologia WS, o fazem para prover um controle de acesso com fraco acoplamento entre domínios tecnológicos heterogêneos. Essa premissa afirma que uma aplicação que possui seus dados gerenciados encaminha requisições, encapsuladas no protocolo SOAP, declaradas por usuários as quais são transportadas através do protocolo HTTP ocorrendo na maioria dos casos através da Internet, o que facilita ataques à integridade e confidencialidade das mensagens.

Diz-se que o formato da requisição é uma premissa pela observação de características encontradas, tais como: sempre existe uma solicitação contendo a identificação do usuário, o recurso ou informação gerenciada que o sujeito manifesta uma necessidade de executar uma ação. Enquanto que nas autorizações de acesso encontram-se semelhanças, visto que sempre após o processamento da requisição de acesso é encaminhada uma mensagem ao solicitante do recurso contendo a decisão do mecanismo, que pode estar autorizando ou restringindo o usuário.

Portanto, afirma-se que uma requisição ou autorização de acesso constitui-se de uma mensagem, contendo apenas um corpo SOAP, que pode conter uma requisição ou autorização de acesso e que deve criptografar e assinar os elementos suscetíveis a adulterações, de forma a evitar o ataque *Tampering*. Tendo como base a hipótese criada pelas premissas apresentadas, afirma-se a necessidade de uma arquitetura capaz de identificar se a mensagem sofreu um ataque *XML Signature Wrapping Element* no meio de transmissão.

4.2. Modelagem Arquitetural

A Arquitetura de Segurança para Modelos de Controle de Acesso Baseados em Serviços *Web* é uma modelagem realizada através da hipótese que uma mensagem torna-se imune a ataques *Tampering* através da assinatura e criptografia dos elementos suscetíveis a adulteração, tais como cabeçalho e corpo SOAP. E para torná-la imune a ataques *XML*

Signature Wrapping Element é necessária uma definição prévia de como uma mensagem deve ser formada para ser considerada válida com adição da proteção do canal de comunicação pelo qual o tráfego ocorre. Para realizar essa modelagem, considera-se os indicadores de: Não Repúdio, Processamento e Transporte Seguro, uma ilustração da proposta é apresentada na Figura 7.

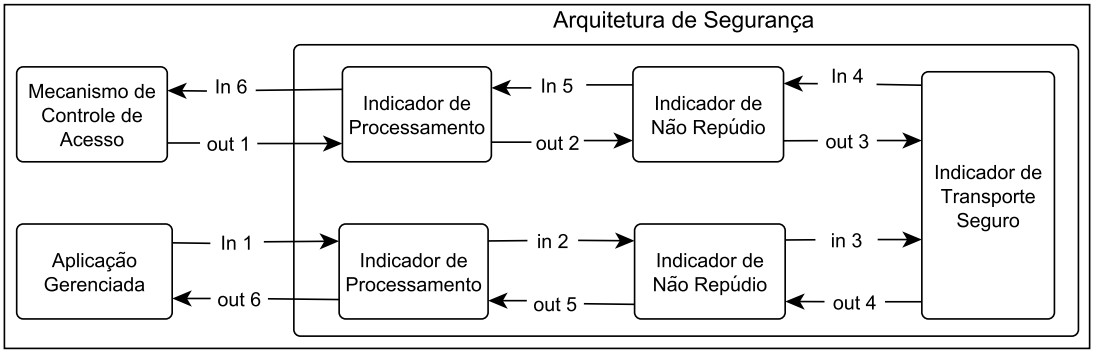

Figura 7. Arquitetura de Segurança para Mecanismos de Controle de Acesso Baseados em Serviços *Web*

Percebe-se que Arquitetura de Segurança para Modelos de Controle de Acesso Baseados em Serviços Web, ilustrada na Figura 7, organiza seus elementos de forma a proporcionar uma comunicação segura entre a aplicação que possui seus recursos gerenciados e o WS do mecanismo de controle de acesso. Nota-se também na Figura 7, o fluxo que deve ocorrer tanto para envio de mensagens (*out*), quanto para inibir as partes comunicantes de processar uma mensagem burlada por um atacante (*out*). O Indicador de Transporte garante que as mensagens foram enviadas por um canal de comunicação seguro, e o Indicador de Não Repúdio garante a não manipulação através do meio de transmissão devido ao uso de criptografia e assinatura de atributos suscetíveis à adulteração, enquanto que o Indicador de Processamento determina o formato padrão de mensagens SOAP em modelos de controle de acesso.

Acompanhando o fluxo da Figura 7, uma aplicação ao requisitar acesso encaminha os atributos da requisição ao Indicador de Processamento (*in* 1) responsável por codificar no formato SOAP determinado para as mensagens, após o término da construção do envelope, este é encaminhado ao Indicador de Não Repúdio (*in* 2) que criptografa e assina os elementos suscetíveis a adulteração. Ao final dessa etapa a mensagem é transportada através de um Indicador de Transporte Seguro (*in* 3). O elemento Indicador de Não Repúdio (*in* 4), através do canal de comunicação, verifica a integridade dos campos criptografados e assinados, e logo após (*in* 5) o Indicador de Processamento verifica a estrutura do envelope SOAP para detectar ameaças *XML Signature Wrapping Element*. Caso a mensagem possua a estrutura determinada nas políticas de segurança, então ela está apta para o processamento (*in* 6). Na presença da anomalia no formato da mensagem, a mesma é descartada; o processo inverso (*out*) se repete para realizar a autorização.

5. Prova de Conceitos

Nessa seção três aspectos relacionados à prova de conceitos são apresentados, de forma que, a seção 5.1 apresenta detalhes técnicos relacionados ao desenvolvimento do

protótipo, na seção 5.2 é demonstrada a integração da Arquitetura de Segurança com o gerenciamento de uma rede *Wireless*, enquanto que em 5.3 é apresentada a viabilidade de uso.

5.1. Desenvolvimento do Protótipo

Envolvendo a Arquitetura de Segurança foi implementado um protótipo visando comprovar sua adaptação em mecanismos de controle de acesso desenvolvidos com a tecnologia de serviços *Web*, bem como a inviabilização de ataques *Tampering* e *XML Signature Wrapping Element*. Para isto, os Indicadores de Não Repúdio e Processamento foram implementados com base nas especificações *WS-Policy*, *WS-Policy Attachment*, WSS e WS-BPEL respectivamente, visto que a combinação dessas tecnologias permite o desenvolvimento de um processo de negociação seguro entre partes comunicantes [da Silva Böger et al. 2007] [da Silva Böger et al. 2008] [Lohn and Wangham 2009]. Enquanto que no Indicador de Transporte Seguro, utilizou-se o *Secure Shell* (SSH), por possibilitar segurança fim a fim no canal de comunicação.

A implementação do Indicador de Processamento foi realizada através da criação de políticas de segurança, com base na especificação WS-Policy e WS-Policy Attachment, e anexadas à interface WSDL do serviço *Web* do mecanismo de controle de acesso e ao processo *Processamento* BPEL. O processo *Processamento* garante que apenas mensagem SOAP que possuem apenas um elemento SOAP - contendo envelope, corpo e serviços - com três argumentos de entrada (requisições) e um de saída (autorizações) estarão aptos para o processamento, inibindo dessa forma ataques *XML Signature Wrapping Element*. Nesta etapa foi utilizado o *framework* de código aberto *Java Pattern Oriented Framework* (JT)[3] por possibilitar a manipulação de processos BPEL.

A escolha da especificação WSS para o Indicador de Não Repúdio, justifica-se devido a mensagem agregar os padrões *XML-Signature* e *XML-Encryptation* que correspondem aos mecanismos de assinatura e criptografia citados na arquitetura, além de possibilitar a utilização de *tokens* seguros. Para o desenvolvimento do Indicador de Não Repúdio foi adotado o *framework Web Services Interoperability Technologies* (WSIT)[4], enquanto que o servidor *Web* utilizado foi o *Glassfish*[5], com essa abordagem inibe-se os ataques *Tampering*.

O Indicador de Transporte Seguro foi implementado com SSH por motivos relacionados à topologia de rede utilizada, pois se fez necessária a criação de um túnel que possibilitasse que o servidor responsável por disponibilizar o serviço *Web* do mecanismo de controle de acesso fosse capaz de comunicar-se com a aplicação gerenciada. No entanto como a Arquitetura de Segurança não define uma tecnologia a ser empregada, apenas salienta o uso do conceito de comunicação segura, a implementação poderia ser realizada com canais de comunicação seguros providos pelo WSS como *Secure Socket Layer* (SSL) e *Transport Layer Security* (TLS) [Nadalin et al. 2006], ou ainda com uma rede virtual privada.

[3]http://java.dzone.com/announcements/jt-java-pattern-oriented-frame-0
[4]https://wsit.dev.java.net/
[5]https://open-esb.dev.java.net/

5.2. Ambiente de Testes

O ambiente de testes trata-se de uma rede *Wireless* gerenciada por um servidor *proxy Squid*, cujo módulo *rewriter* foi modificado de modo que ao invés de consultar ACLs para verificar o acesso, realize uma requisição ao serviço *Web* de um mecanismo de controle de acesso. Os servidores utilizados para a realização dos testes possuem a seguinte configuração: processador *Intel (R) Core(TM)2 Quad CPU Q8200 @2.33GHz* com 1 GB de memória *RAM*. As modificações no serviço *Web* do mecanismo foram realizadas de modo que este possua os Indicadores de Não Repúdio, Processamento e Transporte Seguro. Uma ilustração da integração da Arquitetura de Segurança no ambiente de testes é apresentada na Figura 8.

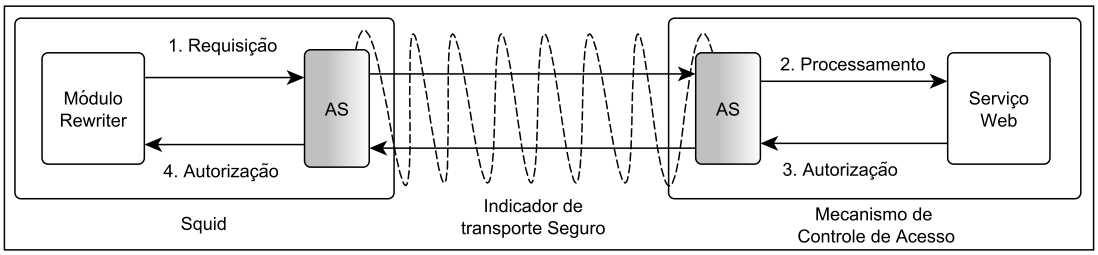

Figura 8. Integração entre o mecanismo de controle de acesso e Squid.

Nota-se que a Figura 8 apresenta o mecanismo de controle de acesso integrado ao servidor *Squid* através da Arquitetura de Segurança (AS). Percebe-se que o tráfego das mensagens percorre o túnel com criptografia fim a fim provido pelo *Secure Shell* (SSH), representando o Indicador de Transporte seguro. O Indicador de Não Repúdio está agregado a *Web Services Description Language* (WSDL) através da especificação WS-Security, inibindo modificações devido a assinatura e criptografia. Já o Indicador de Processamento também é incorporado a WSDL com a especificação WS-BPEL, determinando que as mensagens a serem processadas devem possuir somente um elemento *envelope*, *body*, *header* e apenas um serviço por mensagem SOAP.

No ambiente apresentado, foram realizados ataques contra a integridade e confidencialidade das requisições (destinadas ao mecanismo de controle de acesso) e as autorizações (delegadas aos usuários do servidor *proxy*) visando burlar a segurança proporcionada pelo modelo de segurança. Para realizar esses ataques utilizou-se o *sniffer tcpdump*[6] de modo que este capturasse as mensagens SOAP no meio de transmissão, de forma que ao analisar o fluxo de mensagens SOAP obtenha-se informações sobre como burlar credenciais para obter acesso não autorizado. Para simular o envio de mensagens SOAP adulteradas, utilizou-se o *software Web SoapUI*[7], que permite a realização de testes de performance, avaliação e processamento de requisições.

5.3. Viabilidade de Uso

Para demonstrar a viabilidade de uso da proposta, foram realizados três cenários de testes para avaliar a performance do mecanismo de controle de acesso após a agregação da

[6]http://www.tcpdump.org/
[7]http://www.soapui.org/

Arquitetura de Segurança para Mecanismos de Controle de Acesso Baseados em Serviços *Web*, sendo que durante a realização dos testes foram capturadas as mensagens SOAP no canal de transporte. A Figura 9 apresenta os resultados obtidos.

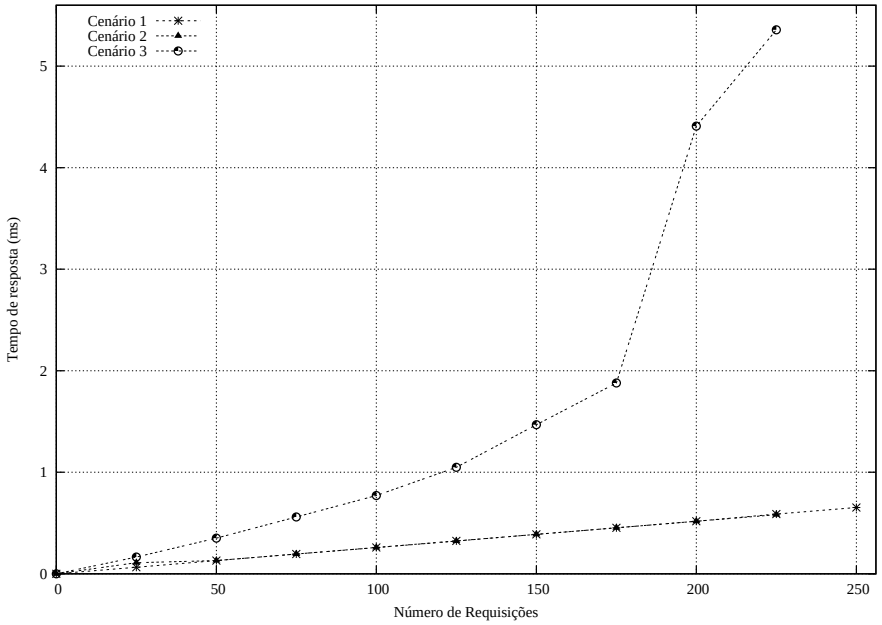

Figura 9. Tempos de resposta obtidos em três cenários diferentes.

O primeiro cenário aborda o modelo baseado em WS sem a agregação da Arquitetura de Segurança, de forma a obter o tempo de resposta médio de requisições e autorizações do ambiente. No entanto, nesse cenário não foram utilizados mecanismos de segurança, permitindo a manipulação das mensagens SOAP. Já o segundo cenário considera somente o Indicador de Transporte Seguro, gerado para avaliar o custo computacional agregado ao indicador utilizado no cenário. Com o acréscimo deste indicador, as mensagens SOAP que trafegavam pelo canal de comunicação do cenário 2 não eram possíveis de serem analisadas. Já o terceiro cenário abrange a Arquitetura de Segurança com os três indicadores: Indicador de Não Repúdio, Indicador de Processamento e Indicador de Transporte Seguro. Portanto, com a adição de todos os indicadores pertencentes na arquitetura, foi possível verificar a impossibilidade de execução de ataques *Tampering*, devido à criptografia e assinatura das mensagens SOAP.

Com base na Figura 9, nota-se que a performance do mecanismo baseado em um WS sem a utilização da arquitetura de segurança, apresenta o tempo mínimo para o processamento das requisições correspondente em média de 0,35625 milissegundos em um teste de *stress* correspondentes a 250 requisições. Com a adição da propriedade de segurança correspondente ao canal de transmissão esse tempo médio de resposta não possui alteração enquanto que a utilização completa das propriedades de segurança pertencentes à arquitetura o valor mediano é elevado 1,8796 milissegundos.

6. Conclusões

Nota-se que modelos de controle de acesso visam agregar integridade, confidencialidade e disponibilidade no gerenciamento de recursos e informações e que a implementação desses mecanismos tendem a ocorrer com base na tecnologia WS, de forma a proporcionar

interoperabilidade entre domínios tecnológicos heterogêneos. No entanto, não são relatadas preocupações com a integridade e confidencialidade na troca de mensagens SOAP que trafegam na rede informando requisições e autorizações.

Este trabalho apresentou uma Arquitetura de Segurança para Mecanismos de Controle de Acesso baseados em Serviços *Web* projetada para inibir ataques *Tampering* e *XML Wrapping Signature Element*, impedindo que atacantes manipulem autorizações e requisições de acesso para obter acesso não autorizado em mecanismos de controle de acesso baseados em WS. Para isto, foram propostos os Indicadores de Não Repúdio, Processamento e Transporte Seguro, de modo a proporcionar confidencialidade e integridade ao fluxo de mensagens SOAP no meio de transmissão.

Para provar esses conceitos foi realizada uma integração entre a Arquitetura de Segurança e um mecanismo de controle de acesso baseado em WS, responsável por gerenciar as requisições de um servidor *proxy Squid*. Desta forma a cada requisição que o servidor *proxy* realizaria para consultar suas ACLs, será direcionada para o mecanismo baseado em WS. Com os testes realizados compreende-se que a arquitetura possui um tempo aceitável de processamento, visto que esta pode ser implementada parcialmente em mecanismo de controle de acesso.

Ao comparar os resultados do Indicador de Transporte Seguro com requisições desprotegidas, percebe-se que esse indicador possui custo computacional baixo e dificulta ataques *Tampering* e *XML Signature Wrapping Element*. No entanto, ao analisar o resultado com todos os indicadores nota-se que o custo computacional é maior, podendo ser amenizado com a utilização de *tickets* de acesso[8], todavia além da inibição dos ataques mencionados proporciona um grau mais elevado de segurança para o mecanismo de controle de acesso.

Referências

Bertino, E., Martino, L., Paci, F., and Squicciarini, A. (2010). *Security for Web Services and Service-Oriented Architectures*. Springer Publishing Company, Incorporated.

Bertino, E., Thuraisingham, B., Gertz, M., and Damiani, M. L. (2008). Security and privacy for geospatial data: concepts and research directions. In *SPRINGL '08: Proceedings of the SIGSPATIAL ACM GIS 2008 International Workshop on Security and Privacy in GIS and LBS*, pages 6–19, New York, NY, USA. ACM.

Camy, A., Westphall, C., and Righi, R. (2005). Aplicação do modelo uconabc em sistemas de comércio eletrônico b2b. *5th Brazilian Symposium on Information Security and Computing Systems (SBSeg)*.

Champion, M., Haas, D. H. C. H., and Booth, D. (2004). Web services architecture working group. *Disponível em: http://www.w3.org/2002/ws/arch/*.

da Silva Böger, D., Wangham, M. S., and da Silva Fraga, J. (2007). Implementação de um modelo de transposição de autenticação para serviços web. *VII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - VII SBSeg*.

[8]Dessa forma a aplicação gerenciada verifica a validade do *ticket* de autorização antes de realizar uma requisição ao serviço *Web* do mecanismo de controle de acesso, diminuindo o fluxo de mensagens SOAP e agregando segurança nas requisições e autorizações realizadas.

da Silva Böger, D., Wangham, M. S., Fraga, J., and Mafra, P. (2008). Um modelo de composição de políticas de qualidade de proteção para serviços web compostos. *VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - VIII SBSeg*.

Dürbeck, S., Fritsch, C., Pernul, G., and Schillinger, R. (2010). A semantic security architecture for web services - the access-egov solution. In *Proc. of the 5th International Conference on Availability, Reliability and Security (ARES 2010)*. IEEE Computer Society, Krakow, Poland.

Gajek, S., Jensen, M., Liao, L., and Schwenk, J. (2009). Analysis of signature wrapping attacks and countermeasures. In *Web Services, 2009. ICWS 2009. IEEE International Conference on*, pages 575–582.

Gruschka, N. and Iacono, L. L. (2009). Vulnerable cloud: Soap message security validation revisited. In *ICWS*, pages 625–631.

Han, R.-F., Wang, H.-X., Xiao, Q., Jing, X.-P., and Li, H. (2009). A united access control model for systems in collaborative commerce. *Journal of Networks*, 4(4):279–289.

Lazouski, A., Martinelli, F., and Mori, P. (2010). Usage control in computer security: A survey. *Computer Science Review*, 4(2):81 – 99.

Lohn, J. R. and Wangham, M. S. (2009). Desenvolvimento de uma infra-estrutura para composição dinâmica e segura de serviços web. *IX Simpósio Brasileiro em Segurança da Informação e de Sistemas computacionais - IX SBSeg*.

Macedo, R., Nunes, R., and Bandeira, J. (2010). Modelo de controle de acesso baseado em expressões contextuais. *Conferência Latino Americana de Informática CLEI*.

Nadalin, A., Goodner, M., Gudgin, M., Barbir, A., Granqvist, H., and eds. (2005). Ws-trust 1.3 oasis standard (oasis). *Disponível em: http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.html acessado em 01 de Julho de 2010*.

Nadalin, A., Kaler, C., Hallam-Baker, P., Monzillo, R., and eds. (2006). Web services security: Soap message security 1.0 (ws-security, 2004) errata in 2006 (oasis standard). *Disponível em: http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf acessado em 01 de Julho de 2010*.

Ravi, Edward, Hal, and Charles (1996). *Role-Based Access Control Models*, volume 29. IEEE Press.

Soares, G., Nunes, R., and Amaral, E. (2006). Um modelo de controle de acesso baseado em contexto para autorizações a informações médicas. *Conferência Latino Americana de Informática CLEI*.

Thomas, R. and Sandhu, R. (1997). Task-based authentication control (tbac): A family of models for active an enterprise-oriented authentication management. *IFIP 97, pp. 11-13*.

Winsborough, W. H., Seamons, K. E., and Jones, V. E. (2000). Automated trust negotiation. *Proceedings of 2000 DARPA Information Survivability Conference and Exposition. IEEE press, pp. 88-102*.

Yuan, E. and Tong, J. (2005). Attributed based access control (abac) for web services. *ICWS 05*.