UNIVERSIDADE FEDERAL DE SANTA CATARINA

UTILIZAÇÃO DO MODELO DE CONTROLE DE ACESSO RADAC (RISK-ADAPTIVE ACCESS CONTROL) EM UM AMBIENTE DE COMPUTAÇÃO EM NUVEM

GUSTAVO ROECKER SCHMITT

Florianópolis - SC

2014/2

UNIVERSIDADE FEDERAL DE SANTA CATARINA

DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

CURSO DE SISTEMAS DE INFORMAÇÃO

UTILIZAÇÃO DO MODELO DE CONTROLE DE ACESSO RADAC (RISK-ADAPTIVE ACCESS CONTROL) EM UM AMBIENTE DE COMPUTAÇÃO EM NUVEM

GUSTAVO ROECKER SCHMITT

Trabalho de conclusão de curso apresentado como parte dos requisitos para obtenção do grau de Bacharel em Sistemas de Informação

Florianópolis - SC
2014/2

Gustavo Roecker Schmitt

Utilização do modelo de controle de acesso RAdAC (Risk-Adaptive Access Control) em um ambiente de computação em nuvem

Trabalho de conclusão de curso apresentado como parte dos requisitos para obtenção do grau de Bacharel em Sistemas de Informação

Orientador(a): Carla Merkle Westphall

Banca examinadora

Daniel Ricardo dos Santos

Jorge Werner

Rafael Weingärtner

RESUMO

O presente trabalho tem como objetivo oferecer uma implementação do modelo RadAc utilizando o XACML, em um ambiente de computação em nuvem, visando calcular o risco de segurança de forma dinâmica.
Na proposta apresentada, o XACML recebe um novo componente responsável por seguir o modelo RadAc e calcular o risco de segurança. O risco de segurança é composto por de três pilares: riscos de contexto, riscos considerando confidencialidade, integridade e disponibilidade, e risco considerando o histórico do sujeito. Um algoritmo que combina as decisões do XACML e RadAc é apresentado para agregar as duas decisões em uma.

Palavras-chave: Controle de acesso, RadAc, XACML, Cloud Computing.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

Com o crescimento da computação em nuvem são levantadas várias questões de segurança envolvendo desde arquivos de usuários a grandes volumes de informações em banco de dados. Dado que o modelo de multi-inquilino possibilita mais de um usuário usar o mesmo software compartilhando a mesma estrutura física e virtual, usuários ficam com medo de expor suas informações por tratar-se de um ambiente dinâmico. A segurança é um fator primordial na nuvem. A migração dos recursos e informações para a nuvem computacional acarreta uma grande massa de dados que pode ser alvo de ataques.

Alguns modelos tradicionais de controle de acesso usados atualmente fornecem o controle de acesso de grão fino. Normalmente, os modelos de controle de acesso tradicionais são estáticos, isto é, tem regras que não mudam durante o acesso dos usuários. Entretanto, ambientes de computação em nuvem são dinâmicos, e os atuais modelos de controle de acesso não têm tanta flexibilidade para suportar um ambiente em nuvem (McGraw, 2009).

Portanto, um mecanismo que controle o acesso de maneira dinâmica resolve o problema da falta de flexibilidade (SANTOS et al., 2013).

1.2 OBJETIVOS

1.2.1 Objetivo geral

O presente trabalho tem por objetivo desenvolver uma estrutura que possibilite o processamento de políticas XACML, em um ambiente distribuído, na nuvem computacional, considerando o modelo RadAc e calcular o risco de forma dinâmica e quantitativa. Uma aplicação de exemplo será implementada para ilustrar o funcionamento da estrutura.

1.2.2 Objetivo específico

Os objetivos específicos deste trabalho são:

- Desenvolver uma estrutura que possibilite avaliações de políticas XACML;
- Implantar o RadAc no XACML;
- Calcular o risco de forma quantitativa;
- Criar um cenário para desenvolver a aplicação de exemplo;
- Desenvolver uma aplicação de exemplo.

1.3 ORGANIZAÇÃO DO TEXTO

O restante do trabalho está organizado da seguinte forma: o capítulo 2 descreve os conceitos básicos da computação em nuvem; o capítulo 3 apresenta os conceitos básicos do controle de acesso, descreve diversos modelos de controle de acesso com enfoque no modelo RadAc e apresenta o XACML com detalhes; o capítulo 4 apresenta a proposta, a ferramenta XACML utilizada, apresenta alguns dos trabalhos relacionados mais expressivos e os resultados encontrados; e por fim, o capítulo 5 apresenta a conclusão e trabalhos futuros.

2 COMPUTAÇÃO EM NUVEM

A ideia de um sistema similar a computação em nuvem é antiga: Em 1966, em um discurso no centenário do MIT, John McCarthy, o inventor do termo Inteligência Artificial, apresentou a ideia onde serviços computacionais seriam oferecidos ao público na forma de computação utilitária, ou seja, os recursos computacionais como um serviço (Parkhill, 1996). Em 1969 J. C. R. Licklider introduziu a ideia de Intergalatic computer network, um pouco familiar com a atual computação em nuvem. (MOHAMED, 2009).

Neste trabalho será adotada a definição de computação em nuvem do NIST (*National Institute of Standards and Thechnology*). De acordo com (NIST, 2011), computação em nuvem é um modelo que proporciona acesso à rede sob demanda a um pool compartilhado de recursos computacionais configuráveis, que pode ser rapidamente provisionado e liberado com o mínimo esforço de gerenciamento ou interação com o provedor de serviços. Este modelo possui cinco características essenciais, três modelos de serviço e quatro da implantação.

A computação em nuvem apresenta cinco características essenciais (NIST, 2011):

- Serviço sob demanda: O usuário configura cada recurso computacional conforme necessidade, sem exigir interação humana com o provedor de serviço;
- Amplo acesso à rede: Os recursos devem ser acessíveis através da rede por meio de mecanismos padronizados que promovem o uso plataformas heterogêneas, como tablets, celulares, notebooks e outros;
- Pool de recursos: Os recursos de computação são agrupados para servir vários usuários, usando o modelo de multi-inquilino (*multi-tenancy*). O modelo multi-inquilino oportuniza o uso de um mesmo software ou hardware, rodando na mesma máquina virtual e física, atendendo vários usuários;
- Elasticidade: Recursos podem ser facilmente provisionados, em alguns casos automaticamente, criando-se assim ilusão de recursos ilimitados;
- Medição do uso de serviços: A nuvem automaticamente controla e aperfeiçoa os recursos, fornecendo métricas apropriadas para os serviços. Tanto consumidor quanto provedor podem monitorar e controlar a utilização de tal serviço, de forma transparente.

2.1 MODELOS DE IMPLANTAÇÃO

Existem quatro modelos para implantar uma nuvem de acordo com (NIST, 2011):

- Nuvem Pública: A infraestrutura é aberta ao público. Os serviços provisionados pela nuvem podem ser vendidos;
- Nuvem Privada: A infraestrutura é provisionada exclusivamente para oferecer os serviços à organização. O gerenciamento pode ser feito pela empresa ou terceirizado. A nuvem pode ser local ou remota;
- Nuvem Comunitária: A infraestrutura é concedida a um grupo de indivíduos/organizações que tenham interesses e preocupações em comum;
- Nuvem Híbrida: É uma combinação de dois ou três modelos de implantação. São entidades únicas ligadas através de alguma tecnologia padrão. Pode viabilizar o balanceamento de carga entre nuvens, ou seja, quando a nuvem privada não consegue atender a demanda, a pública é acionada.

2.2 MODELOS DE SERVIÇO

Os modelos de serviço se referem aos tipos de recursos oferecidos pelo provedor de serviço ao usuário final. (NIST, 2011) cita três classes:

- SaaS (*Software as a Service*): O consumidor usa o software hospedado no prestador de serviços realizando o acesso através de interfaces de programas (APIs) e/ou navegadores de internet (*Web Browser*), sem a necessidade de gerenciar a infraestrutura, como rede, sistema operacional, armazenamento e outros. Exemplos conhecidos são os

serviços de e-mail da Microsoft (Hotmail), Google (Gmail) e Yahoo (Yahoo Mail);

- PaaS (*Plataform as a Service*): Oferece recursos para desenvolvimento de aplicações na infraestrutura da nuvem, desde que utilize as linguagens de programação e ferramentas suportadas pela nuvem. O consumidor não gerencia a rede, sistema operacional ou armazenamento. O cloudBees oferece este modelo que conta com clientes como Netflix, GROUP ADEO e Viridity Energy. (CloudBees, 2014);
- IaaS (*Infraestrutura as a Service*): O consumidor provisiona os recursos, como processamento, armazenamento, rede, e outros recursos computacionais, e, com esses recursos poderá implantar o software que desejar, não sendo necessário gerenciar a estrutura física. A Amazon, principal provedora de infraestrutura como serviço, possui a maior fatia do mercado, seguida pela AT&T, que promete disponibilidade 99,9%.

2.3 PLATAFORMAS DE COMPUTAÇÃO EM NUVEM

2.3.1 OpenStack

O OpenStack é um projeto de código aberto que criou uma plataforma de computação em nuvem para nuvens públicas e privadas (Figura 1). Inicialmente com foco em IaaS (plataforma como serviço), o projeto é dividido em módulos (PEPPLE, 2011):

- *OpenStack Compute*: Responsável por gerenciar e criar máquinas virtuais;
- *OpenStack Object Store*: Cuida da parte de armazenamento, como cópias distribuídas;
- *OpenStack Image Service*: Permite gerenciar serviços para imagens de máquinas virtuais.

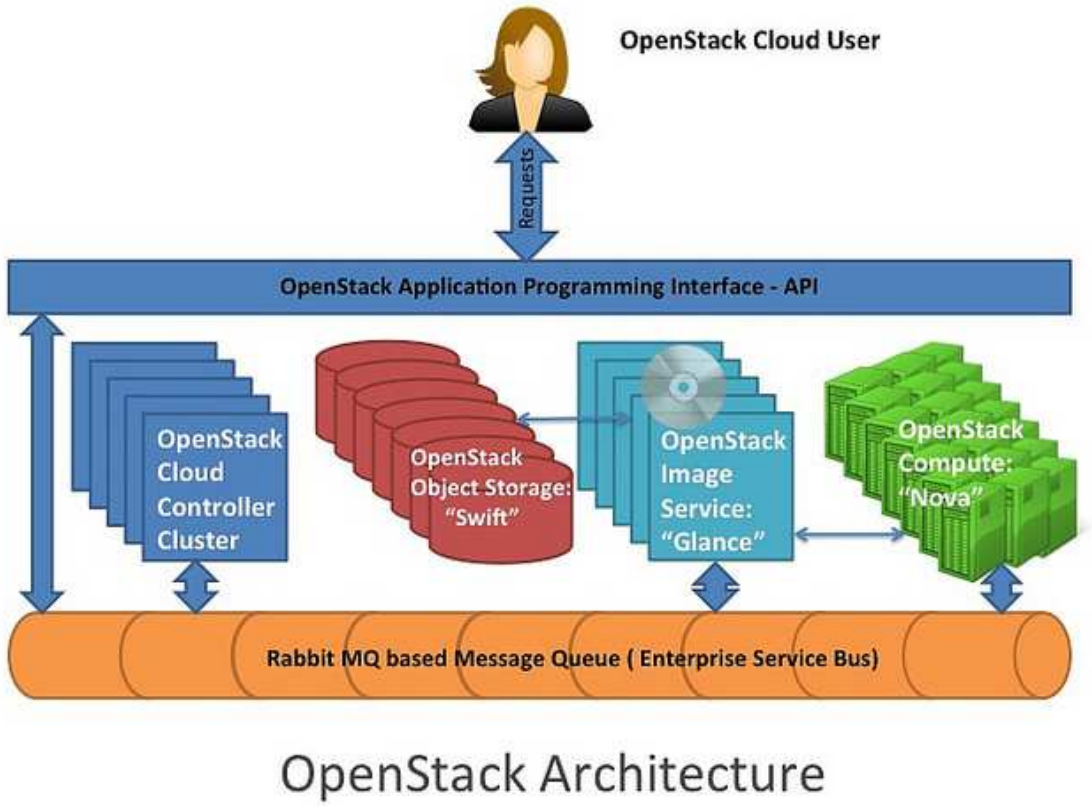

Figura 1 - Arquitetura do OpenStack (OpenStack, 2013).

2.3.2 CloudStack

A CloudStack oferece uma plataforma para gerenciar e provisionar recursos, também é um projeto de código aberto com a vantagem de oferecer o

modelo de implantação para, além de nuvens públicas e privadas, nuvens híbridas. Oferece o modelo de serviço *IaaS*. Tipicamente usada por provedores de serviços ou empresas para implantar, gerenciar e configurar ambientes em nuvem. O CloudStack oferece uma interface amigável para administradores e usuários, onde é possível configurar recursos.

Neste trabalho foi usado a *Cloud* do LRG (Laboratório de Redes e Gerência), localizada nas dependências da UFSC (Universidade Federal de Santa Catarina), disponível para pesquisas acadêmicas na área de gerenciamento e segurança de computação em nuvem.

A tabela 1 mostra a infraestrutura da *Cloud* do LRG, onde é possível verificar que a cloud conta com 8 servidor e mais de 60 *gigabytes* de RAM (*Random Access Memory*) e o espaço de armazenamento é superior a 3 *terabytes* (CloudStack LRG, 2014).

Servidor	**CPU**	**RAM**	**Papel**	**Cluster**
Octacore	Pentium D @ 3.4Ghz	2GB	Storage server	-
Cirrus	Core 2 E6600 @ 2.4Ghz	7GB	Processing server	PV
Stratus	Core 2 quad, Q8200 @2.3Ghz	4GB	Processing server	PV
Hurricane	Xeon E5405 @ 2Ghz	8GB	Processing server	HVM Intel
Twister	Xeon(R) CPU E3-1240 V2 @ 3.40GHz	16GB	Processing server	HVM Intel
Tornado	Xeon(R) CPU E3-1240 V2 @ 3.40GHz	16GB	Processing server	HVM Intel

Nimbus	Phenom 9650 @ 2.3Ghz	4GB	Processing server	HVM AMD
Cumulus	Phenom II 965 @ 3.4Ghz	4GB	Processing server	HVM AMD

Tabela 1 - Infraestrutura da CloudStack LRG. (CloudStack LRG, 2014);

2.3.3 Desafios

O crescimento da computação em nuvem enfrenta diversos obstáculos, como o capital que empresas investiram no passado para aquisição de softwares e licenças, e o fato de que a nuvem pode não dar suporte a sistemas desenvolvidos com tecnologias legadas, como os atuais sistemas bancários, que foram desenvolvidos no século XX. Estes sistemas terão que ser reprojetados para uma linguagem atual, ocasionando custos elevados.

Na mesma medida que a computação em nuvem cresce, o aumento da largura de banda faz-se necessário devido à maior quantidade de dados que será trocada entre a nuvem e os clientes. A Internet é um ambiente hostil e isso se torna crítico quando todos os dados de uma pessoa/organização estão trafegando entre os nós da rede e os servidores em nuvem. Portanto, a segurança é um fator essencial para o crescimento da mesma.

A figura 2 representa que a maior preocupação para a computação em nuvem está na área da segurança, como, por exemplo, as informações sensíveis da organização e vulnerabilidade de ataques.

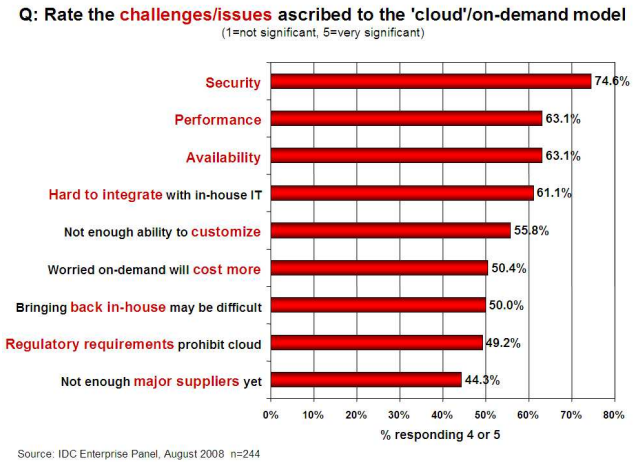

Figura 2 - Classificação dos desafios ainda pendentes na computação em nuvem (IDC, 2012).

3 CONTROLE DE ACESSO

*"...O controle de acesso é responsável por determinar atividades permitidas de usuários legítimos, interpondo-se a cada tentativa de um usuário para acessar um recurso no sistema. A informação dada tecnologia da informação (TI) pode implementar sistemas de controle de acesso em muitos lugares e em diferentes níveis." (*Hu; Ferraiolo; Kuhn, 2006*).*

A principal importância do controle de acesso é garantir acesso para sujeitos autorizados e proteger recursos de acesso não autorizados. Para tal, deve-se controlar todas as operações no sistema a fim de verificar se a mesma é autorizada.

Existem alguns conceitos que são amplamente usados neste contexto:

- Recurso: São objetos que contém informações. Normalmente trata-se de um documento ou diretório.
- Sujeito: São entidades que acessam o recurso. Tipicamente são usuários ou processos. Também é conhecido como Principal;
- Operação: São operações invocadas pelo usuário.
- Monitor de referência: É uma abstração que controla todas as requisições de acesso a objetos feitas por sujeitos.

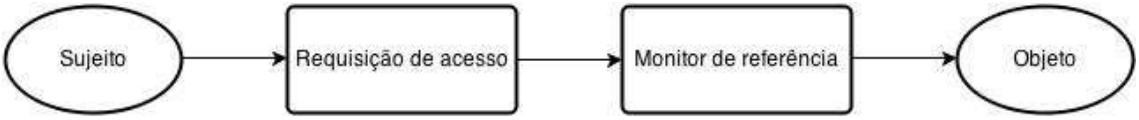

Figura 3 - A base do modelo de controle de acesso (Traduzido de D. Gollmann, 1999).

3.1 GERENCIAMENTO DE IDENTIDADES E DE ACESSO

É comum vermos vários significados atribuídos ao termo identidade. De modo geral, é possível defini-la como uma representação de uma entidade (Cao e Yang, 2010). Porém a definição mais aceita e utilizada é: "*Representação de uma entidade (ou um grupo) sob a forma de um ou mais elementos de informação que permitem que entidades possam ser reconhecidas no contexto de uma aplicação*" (ITU-T, 2009).

Para administrar várias identidades é necessário um sistema de gerenciamento de identidades que (Chadwick, 2009) conceitua como um misto de funções que administram a identidade do usuário para mantê-las seguras e para gerenciar identidades em mundo digital. Ainda mais a fundo no conceito, um serviço de gerência de identidade pode ser definido como "o processo de criação, gerência e utilização de identidades de usuários e a infraestrutura que suporta esse processo" (Chadwick, 2009).

As entidades associadas a gerência de identidade podem ser, de acordo com (ITU-T 2009):

- Provedor de Serviço: Responsável por oferecer serviços a usuários autenticados, como e-mail, e-commerce e outros. O provedor de serviços pode delegar o processo de autenticação de usuários ao

Provedor de Identidade para que este cuide do serviço de gerenciar as identidades.

- Provedor de Identidade: Fornece o serviço de gerenciamento de identidades, responsável por autenticar usuários para que estes usem o provedor de serviço.
- Usuário: É uma entidade que vai utilizar os serviços tanto do provedor de serviço quanto do provedor de identidade e faz uso de uma identidade.

3.2 CONTROLE DE ACESSO DISCRICIONÁRIO

Este modelo restringe o acesso ao objeto baseado na identidade do sujeito e/ou nos grupos a qual este pertence. A principal característica deste modelo é que o proprietário do recurso decide quais grupos e/ou usuários têm acesso e qual permissão podem usufruir. Através de um par entre sujeito e objeto, é possível verificar qual usuário tem permissão a qual objeto (Benantar, 2006).

Para flexibilizar o modelo, é possível permitir que o proprietário forneça suas operações de proprietário para outro usuário. Contudo, este modelo é mais vulnerável a negligência por parte dos usuários, pois o mesmo pode conceder acesso a usuários mal-intencionados.

3.3 CONTROLE DE ACESSO OBRIGATÓRIO

Ao contrário do modelo anterior, onde o acesso é controlado pelo proprietário, no Mandatory Access Control o acesso é controlado por uma classificação, como por exemplo não classificado, classificado, secreto e ultrasecreto.

Normalmente expressa como políticas de segurança, onde cada tentativa de acesso será avaliada através das políticas de segurança. As políticas podem ser criadas por um administrador, que terá a responsabilidade de assegurar que cada sujeito possa acessar somente os recursos especificados na política (Benantar, 2006).

3.4 CONTROLE DE ACESSO BASEADO EM PAPÉIS

Este modelo restringe o acesso do usuário à informação através de papéis, onde um usuário é associado a um papel e permissões são associadas a papéis. Um papel é um conjunto de ações e responsabilidades associados a atividade que o usuário executa no sistema. O modelo é muito utilizado em empresas, pois cada papel representa o cargo do funcionário e suas atribuições na organização, portanto é possível reaproveitar políticas de acesso para um grupo de usuários que têm acessos iguais, gerando um trabalho menor do que nos outros modelos apresentados e, consequentemente aumentando a produtividade do administrador. O NIST classifica os modelos de controle de acesso baseado em papéis em quatro tipos (Sandhu, Ferraiolo e Kuhn, 200):

- Flat RBAC: Usuário são associados a papéis, permissões são associadas a usuário e usuários conseguem permissão sendo membros de papéis.
- Hierarchical RBAC: Adiciona suporte para papéis que podem ser herdados, formando uma hierarquia de papéis.
- Constrained RBAC: Adiciona suporte a separação de obrigações, reduzindo a possibilidade de fraudes
- Symmetric RBAC: Adiciona suporte a revisão de permissões e papéis.

3.5 CONTROLE DE ACESSO BASEADO EM RISCO

O RadAc (*Risk Adaptative Access Control*) foi proposto pela NSA (*National Security Agency*) como uma nova abordagem no âmbito de controle de acesso. O novo paradigma propõe o controle de acesso de uma forma dinâmica, que emula as decisões do mundo real (McGraw, 2009). Atualmente, o fator risco é algo estático nas abordagens, o RadAc surge para suprir esta lacuna. O conceito de necessidade operacional (*Operational need*) refere-se a necessidade real, no momento da requisição, do sujeito ter ou não acesso à informação.

O risco de segurança (*Security risk*) é alguma medida, quantitativa ou qualitativa, dos riscos associados a conceder o acesso a informação. Um exemplo poderia ser caso o usuário não use uma conexão criptografada, o risco de segurança aumenta pois a conexão não é segura. Mais do que comparar atributos, como nos modelos tradicionais, o RadAc calcula o risco de segurança no momento da requisição.

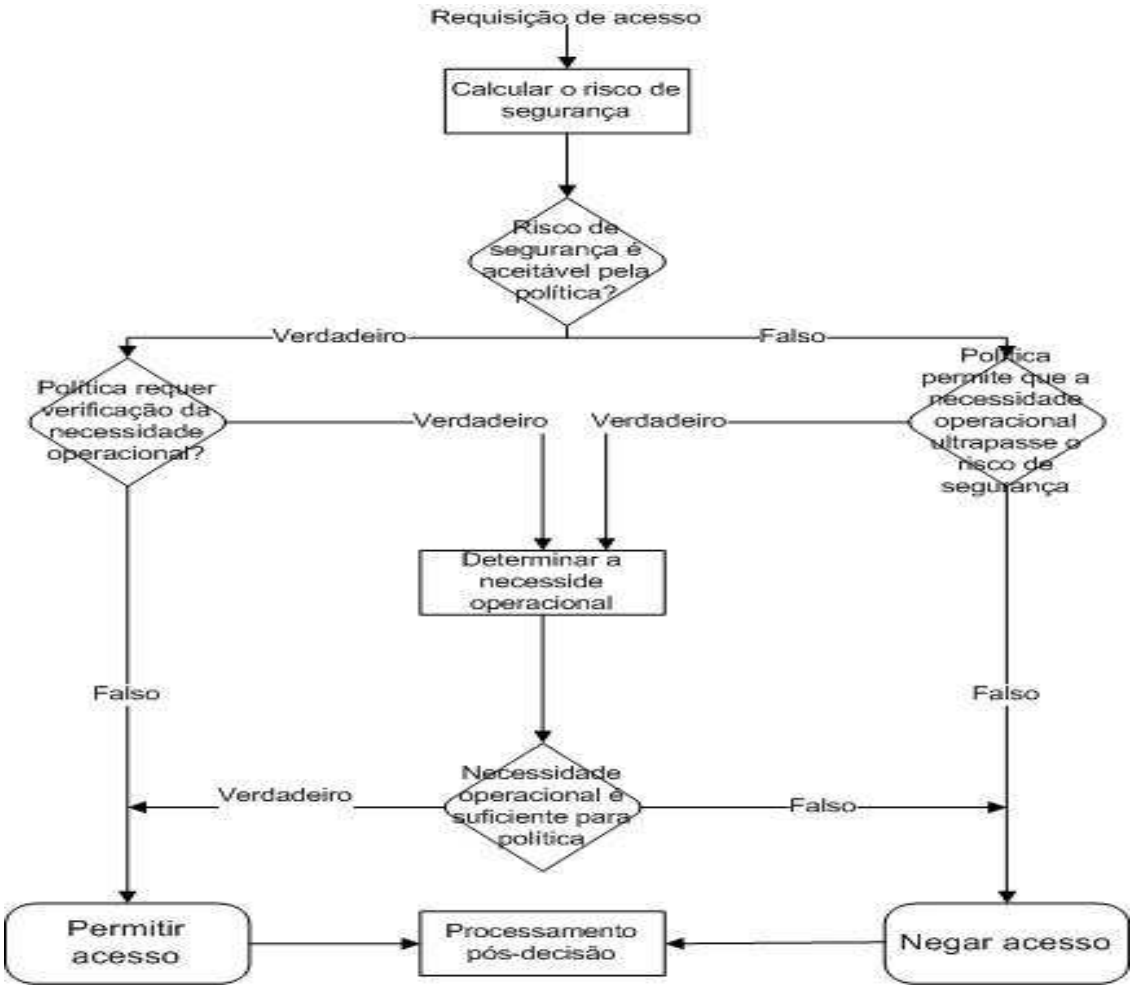

Figura 4 - Fluxograma do RadAc (Adaptado de McGraw, 2009).

Um cenário muito utilizado como exemplo é o hospital (KANDALA; SANDHU; BHAMIDIPATI, 2011). O médico tem todas as informações relativas aos pacientes internados no hospital, enquanto que o enfermeiro apenas possui informações que os médicos disponibilizam. Numa situação atípica, o médico está indisponível, então um paciente tem um problema, e para medicá-lo, o enfermeiro precisa consultar a base de dados e verificar se o paciente é alérgico a determinado medicamento. Nos modelos de controle de acesso atuais, o enfermeiro não teria esse acesso a essa informação a não ser que alguém concedesse. O RadAc calcularia o risco de segurança de liberar essa

informação ao enfermeiro e compararia a necessidade de uso e, talvez, o enfermeiro tivesse acesso à informação.

A figura 5 ilustra os possíveis fatores que o RadAc pode considerar para avaliar uma requisição, como características do usuário, componentes de tecnologia e outros.

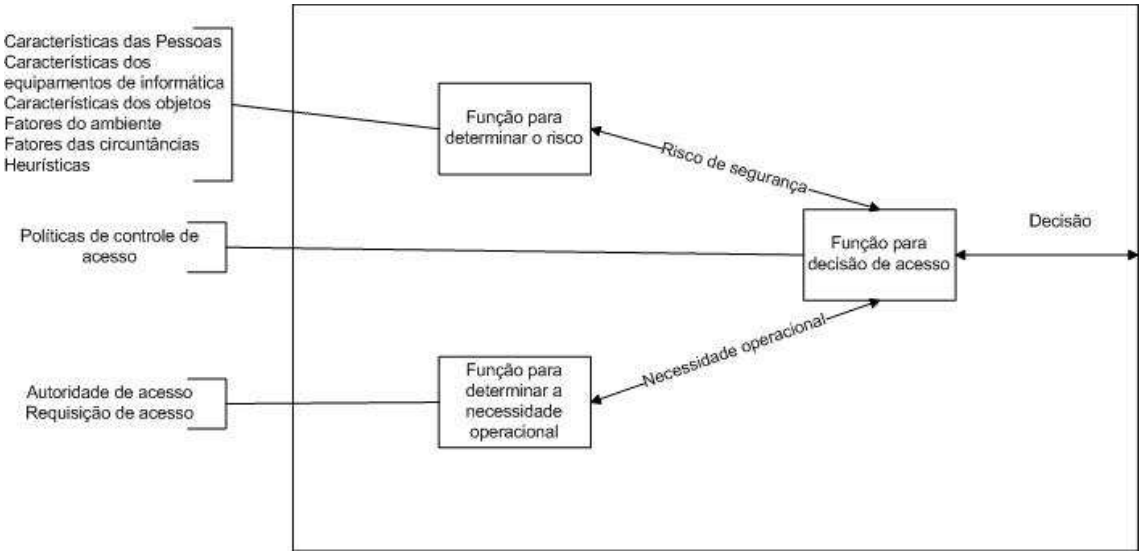

Figura 5 - Estrutura geral do RadAc (Adaptado de McGraw. R, 2009).

3.6 XACML

O XACML (*eXtensible Access Control Markup Language*) é um padrão, criado pela OASIS (*Advancing Open Stadards for the Information Society*), para definir políticas de controle de acesso, ambas desenvolvidas em XML (*eXtensible Markup Language*). A linguagem é declarativa, usada para descrever situações genéricas (OASIS, 2011).

Os requisitos básicos, de acordo com a especificação da (OASIS, 2005), para criar a linguagem para expressar políticas de segurança, são:

- Fornecer um método para a combinação de regras e políticas individuais para um conjunto de critérios simples que se aplica a um pedido de decisão particular.
- Fornecer um método para definição flexível do processo pelo qual as regras e políticas são combinadas.
- Fornecer um método para lidar com múltiplos sujeitos.
- Fornecer um método para basear uma decisão de autorização nos atributos do sujeito e do recurso.
- Fornecer um método para lidar com valores múltiplos de atributos.
- Fornecer um método para basear uma decisão de autorização que possibilite o uso do recurso como atributo.
- Fornecer um conjunto de operadores lógicos e matemáticos em atributos do sujeito, recursos e ambiente.
- Fornecer um método para o tratamento de um conjunto distribuído de componentes de política, abstraindo o método para localizar, recuperar e autenticar os componentes de política.
- Fornecer um método para identificar rapidamente a política que se aplica a uma determinada ação, com base nos valores dos atributos dos sujeitos, recursos e ação.
- Fornecer um método para definir um conjunto de ações que devem ser realizadas em conjunto com a aplicação da política.

Diversas linguagens foram propostas, mas o XACML apresenta algumas vantagens, como: uma linguagem que foi testada por uma comunidade de usuários e desenvolvedores, gerando menor probabilidade de ocorrerem erros;

é possível usar em qualquer ambiente, se adapta a diversas situações, visto que é uma linguagem genérica; ser distribuída por natureza; trata-se de um padrão.

O XACML é, basicamente, dividido em dois componentes. O componente que realiza a lógica para realizar uma avaliação e retornar uma resposta. E outro, usado dentro da aplicação, para fornecer requisições ao anterior. Estes dois principais componentes são, respectivamente:

- *Policy Decision Point* (PDP): O componente que realiza a decisão de controle de acesso, com base nas políticas e informações recuperadas;
- *Policy Enforcement Point* (PEP) O ponto de acesso do usuário ao sistema. É o ponto que protege um recurso sensível, recebendo a requisição de acesso e enviando-a ao PDP.

A figura 6 ilustra o processo de funcionamento do XACML. O primeiro passo é receber uma requisição no PEP. O PEP envia uma requisição de acesso ao PDP. O PDP faz o processo de verificação e retorna uma resposta. O PEP então retorna à aplicação uma resposta, garantindo ou negando o acesso.

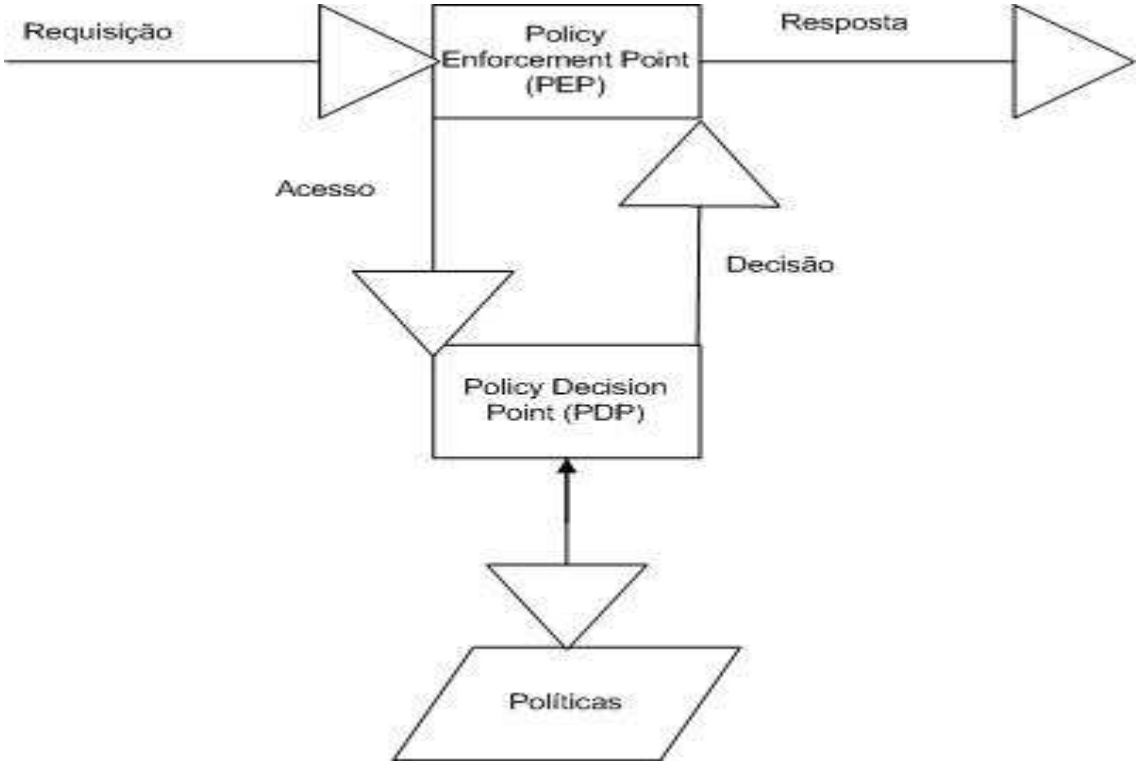

Figura 6 - Fluxo do XACML

O elemento principal do XACML é, conforme a Figura 7, uma *Policy* ou *PolicySet*. Um *PolicySet* é um repositório que pode conter zero ou mais *Policies*, um *Target* e zero ou mais *Obligations*.

Uma *Policy* representa uma política de controle de acesso que é uma combinação de várias regras, estas regras são chamadas de *Rules* pelo XACML. Uma *Rule* é uma regra e será avaliada individualmente.

Cada *Policy* tem um alvo, este é chamado de *Target* pelo XACML, que é uma combinação de sujeito, ação e recurso (*subject*, *action* e *resource*). *Subject* é usada para identificar quem está realizando a ação. Recurso é o que está sendo acessado. Define-se ação como sendo a própria ação em questão, como uma tentativa de leitura ou escrita.

Como um *PolicySet* pode conter várias *Policies* e uma *Policy* pode conter muitas *Rules*, que podem ser avaliadas com decisões de acesso diferentes, é necessário que haja um modo de agrupar essas decisões em um resultado único. Para isso o XACML oferece algoritmos de combinação, tanto o algoritmo de combinação de regras como de políticas são definidos da seguinte forma:

- *Deny-Overrides*: Caso o resultado da avaliação de uma política ou regra seja “*Deny*” o resultado da avaliação será “*Deny*”;
- *Permit-Overrides*: Do mesmo modo, caso o resultado da avaliação de apenas uma política ou regra seja “*Permit*” o resultado da avaliação será “*Permit*”;
- *First-applicable*: O resultado da avaliação será o mesmo da primeira política ou regra encontrada.
- *Only-one-applicable*: O resultado será o mesmo da regra ou política, caso apenas uma seja encontrada. Se mais de uma for encontrada, o resultado será "*Indeterminate*". Caso nenhuma seja encontrada será "*Not-Applicable*";

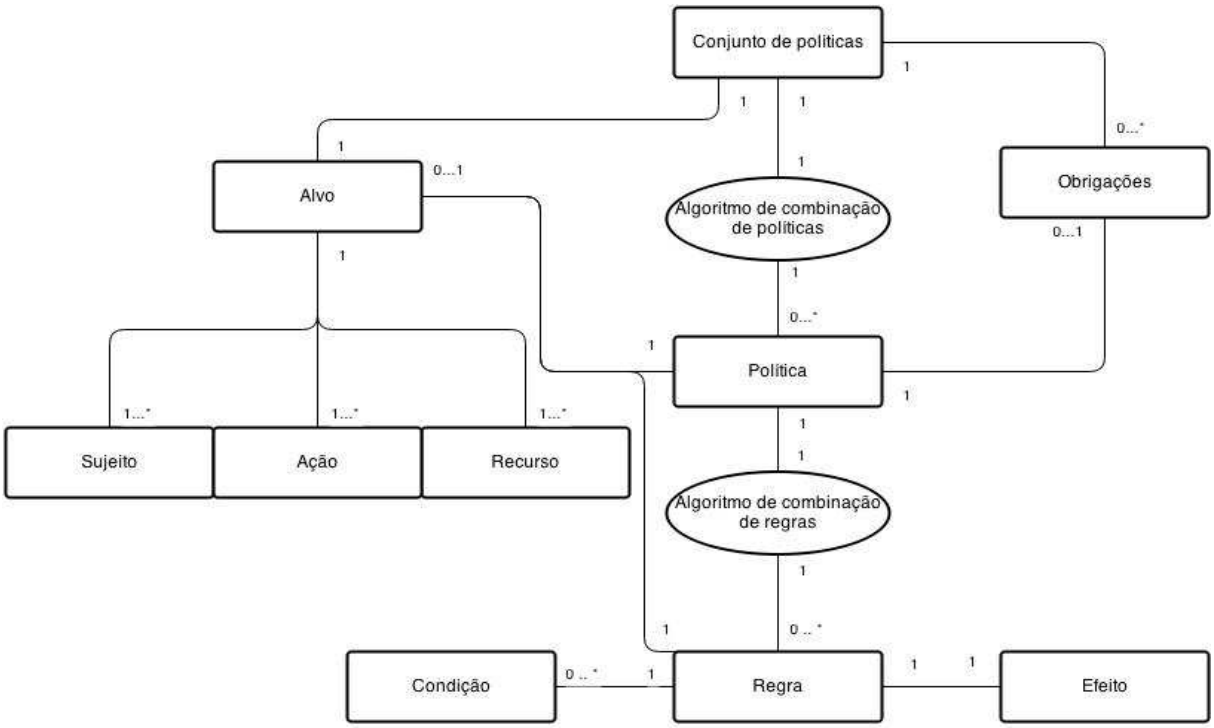

Figura 7 - Diagrama do XACML (IBM, 2004)

Exemplo 1: O usuário Gustavo quer ler o arquivo igfxdev.dll. Para tanto, o primeiro passo deve ser criar uma requisição para o PEP. Uma requisição é formada por atributos de tipos e categorias conhecidas, como no exemplo, o atributo usuário é Gustavo, do tipo *String*, categorizado como Sujeito.

A tradução para XML do exemplo 1 é ilustrada na figura 8. O PDP então verificará se há uma política que pode ser aplicada a essa requisição, onde o sujeito é chamado Gustavo, o recurso http://dll.com.br/igfxdev.dll e a ação é ler.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Request xmlns="urn:oasis:names:tc:xacml:1.0:context" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:context cs-xacml-schema-context-01.xsd">
  - <Subject>
    - <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id" DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>Gustavo</AttributeValue>
      </Attribute>
    </Subject>
  - <Resource>
    - <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id" DataType="http://www.w3.org/2001/XMLSchema#anyURI">
        <AttributeValue>http://dll.com.br/igfxdev.dll</AttributeValue>
      </Attribute>
    </Resource>
  - <Action>
    - <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id" DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>read</AttributeValue>
      </Attribute>
    </Action>
  </Request>
```

Figura 8 - Requisição em XACML.

A figura 9 mostra uma política possível de ser aplicada à requisição de exemplo.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Policy xmlns="urn:oasis:names:tc:xacml:1.0:policy" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:policy cs-xacml-schema-policy-01.xsd" PolicyId="urn:oasis:names:tc:xacml:1.0:conformance-test:IIA1:policy"
  RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides">
  <Description>Politica exemplo 1.</Description>
  - <Target>
    - <Subjects>
        <AnySubject />
      </Subjects>
    - <Resources>
        <AnyResource />
      </Resources>
    - <Actions>
        <AnyAction />
      </Actions>
    </Target>
  - <Rule RuleId="urn:oasis:names:tc:xacml:1.0:conformance-test:IIA1:rule" Effect="Permit">
      <Description>Gustavo pode ler o arquivo igfxdev.dll</Description>
    - <Target>
      - <Subjects>
        - <Subject>
          - <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
              <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Gustavo</AttributeValue>
              <SubjectAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
                DataType="http://www.w3.org/2001/XMLSchema#string" />
            </SubjectMatch>
          </Subject>
        </Subjects>
      - <Resources>
        - <Resource>
          - <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
              <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#anyURI">http://dll.com.br/igfxdev.dll</AttributeValue>
              <ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
                DataType="http://www.w3.org/2001/XMLSchema#anyURI" />
            </ResourceMatch>
          </Resource>
        </Resources>
      - <Action>
        - <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">read</AttributeValue>
            <ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id" DataType="http://www.w3.org/2001/XMLSchema#string" />
          </ActionMatch>
        </Action>
      </Target>
    </Rule>
  </Policy>
```

Figura 9 - Política escrita em XACML.

Comparando a requisição com a política, o PDP gera uma resposta. Esta resposta está na figura 10, onde o acesso é garantido.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Response xmlns="urn:oasis:names:tc:xacml:1.0:context" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:context cs-xacml-schema-context-01.xsd">
  - <Result>
      <Decision>Permit</Decision>
    - <Status>
        <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok" />
      </Status>
    </Result>
  </Response>
```

Figura 10 - Resposta escrita em XACML.

4 UTILIZAÇÃO DO RADAC EM UM AMBIENTE DE COMPUTAÇÃO EM NUVEM

4.1 PROPOSTA

A infraestrutura proposta é ilustrada na figura 11, são 3 máquinas virtuais, a primeira roda uma aplicação de exemplo, a segunda o PEP e a terceira o PDP. O usuário tenta acessar um recurso qualquer, como um documento. A aplicação faz uma requisição ao PEP que cria uma requisição para o PDP. O PDP então avalia a requisição e define um resultado, o RadAc calculará o risco de segurança em conceder acesso e a necessidade operacional. Com o resultado do XACML e do RadAc é utilizado um algoritmo de agregação para gerar o resultado final.

As requisições de permissão serão intermediadas pelo XACML e todo o ambiente estará dentro de uma nuvem computacional.

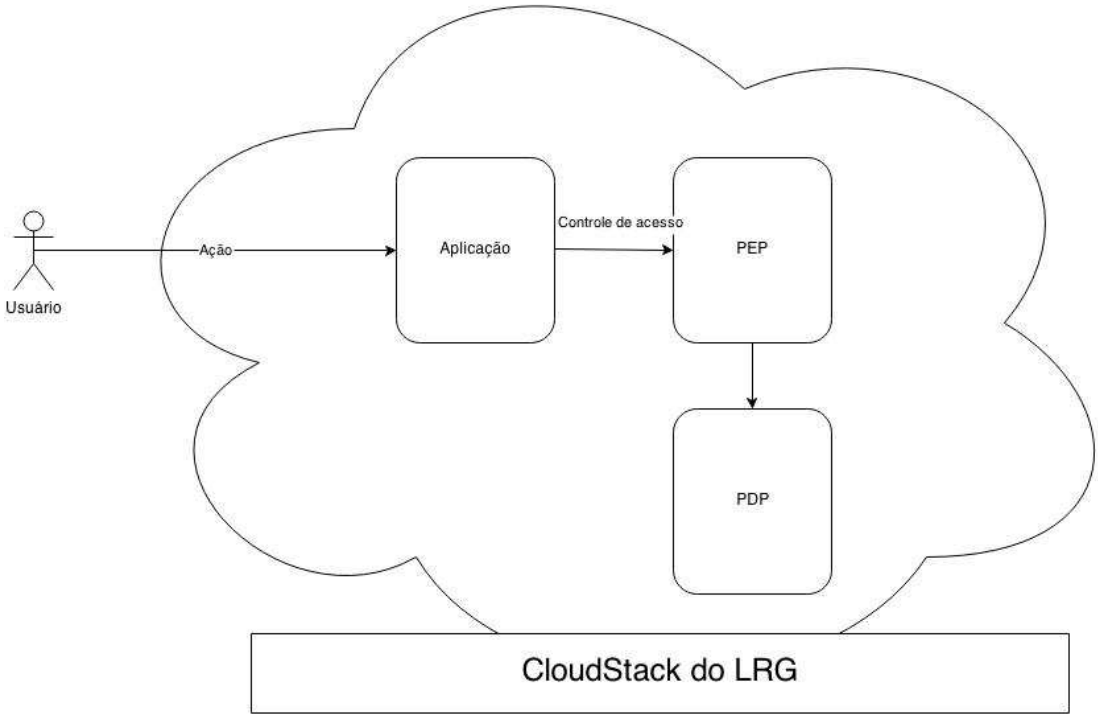

Figura 11 - Diagrama geral da proposta

Uma das principais dificuldades na implementação do modelo RadAc consiste em encontrar uma forma de calcular o risco. SANTOS et al., (2013) e (Marinho, 2014) trazem modelos que quantificam o risco para utilizar no modelo RadAc. Ambos utilizam a abordagem que leva em consideração fatores do contexto e confidencialidade, integridade e disponibilidade. Além disso, Marinho utiliza também o histórico do usuário.

Para calcular os valores relativos ao contexto, ambos utilizam o trabalho de (BRITTON; BROWN, 2007), onde os fatores de risco são classificados e atribuídos pesos. Este método apresenta um modelo para o cálculo de risco do

RadAc, através de 27 métricas divididas em 6 grupos, onde cada métrica possui um risco associado. Tanto os pesos quanto as métricas são atribuídos por especialistas. A tabela pode ser vista no apêndice A.

Os valores de confidencialidade, integridade e disponibilidade são calculados a partir das do trabalho de (SARIPALLI; WALTERS, 2010). O risco é calculado a partir da seguinte fórmula:

Risco = Probabilidade * Impacto

A probabilidade é a quantidade de tentativas de violação de acesso divido pela quantidade total de acessos (SARIPALLI; WALTERS, 2010).

O impacto é calculado a partir do dano que seu acesso pode vir a ocasionar. (SARIPALLI; WALTERS, 2010) sugere os seguintes valores: impacto baixo (1-5), impacto moderado (6-10) e impacto alto (11-15).

O risco relativo ao histórico do sujeito, contemplado no modelo de (Marinho, 2014), sugere que exista uma pontuação para cada usuário. Esta pontuação é compreendida numa escala de 0 a 10, de modo que este valor é decrementado quando ações positivas são feitas e incrementado quando ações negativas são realizadas. Portanto, quanto maior o valor do histórico, maior o valor do risco total.

Com base nestes trabalhos quantificaremos o risco através da seguinte formula:

Risco Total = peso1 * risco do contexto + peso2 * risco confidencialidade, integridade e disponibilidade + peso3 * risco considerando o histórico

Os pesos serão carregados através do banco de dados, oferecendo uma maior flexibilidade para a estrutura.

Para incluir o RadAc e possibilitar o cálculo do risco, incluiremos um novo componente ao Heras-af:

- *Dynamic Decision Point*: Responsável por analisar e quantificar o risco, este componente diretamente ligado ao PDP para fornecer uma resposta o mais rápido possível.

Incluindo o novo componente ao Heras-af, a sua arquitetura ficará conforme a figura 12.

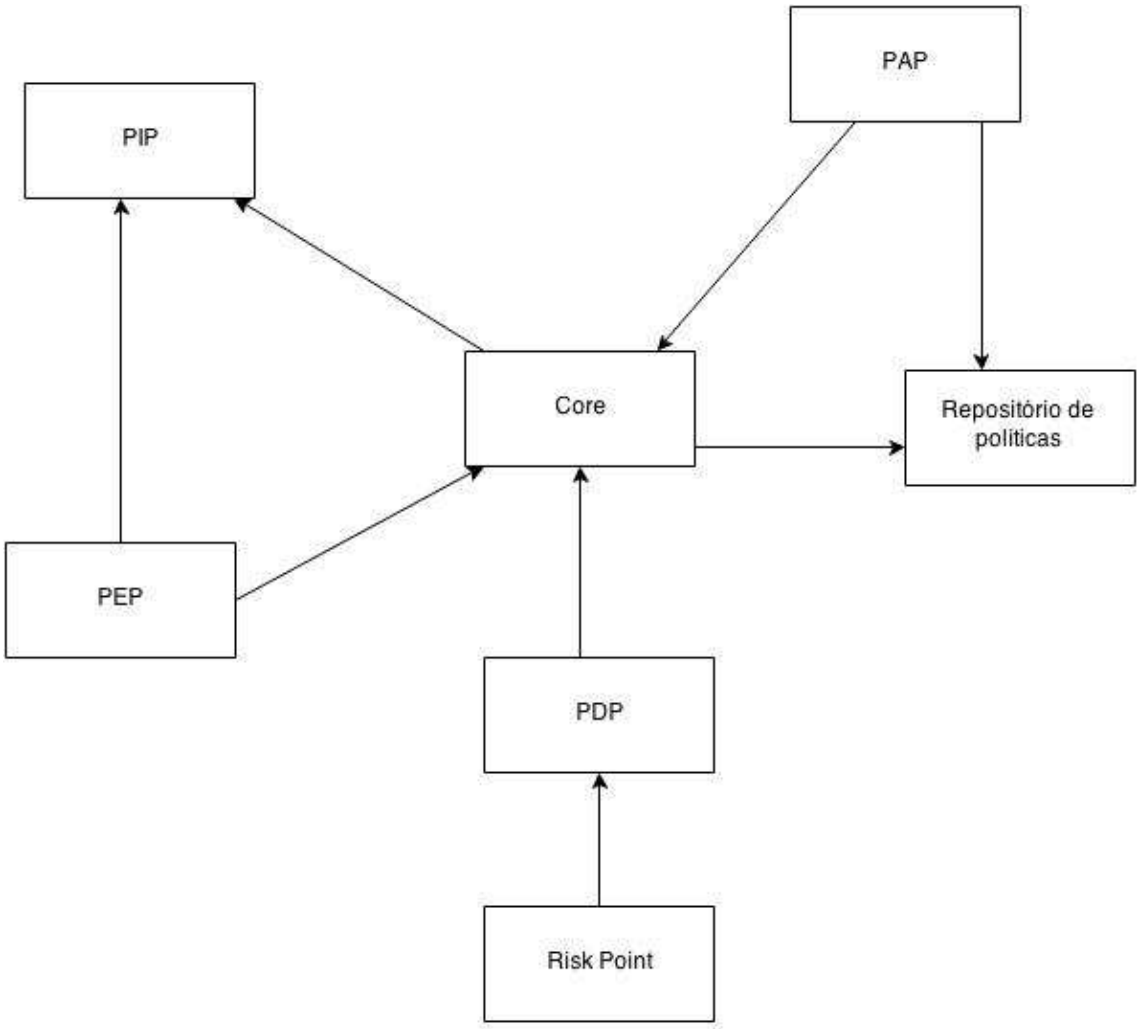

Figura 12 - Proposta de extensão do XACML.

4.2 FERRAMENTAS XACML

A OASIS define um modelo teórico para o XACML e, portanto, é necessário implementá-lo. Há poucas implementações e, dentre elas, um número menor ainda que se comporte como a OASIS prevê, em seu documento de especificação (OASIS, 2005). Neste trabalho procuramos por implementações na linguagem Java, que é a linguagem mais comum para este tipo de sistema. A tabela 2 mostra as implementações estudadas e as compara.

Entidade	Versão	Open-source	Ativa	Conformidade	PDP	PEP	Documentação	JAVADOC
Balana (BALANA, 2012)	3.0	-	X	X	X	X	-	-
HERAS (HERAS, 2012)	2.0	X	X	X	X	X	X	X
Enterprise (Enterprise, 2014)	2.0	X	O	-	X	X	-	-
SUN (SUN, 2006)	2.0	X	O	X	X	X	X	X
JBOSS	2.0	X	X	X	X	X	-	-

VIEWDS	3.0	O	X	-	-	-	-	-
NextLabs (NextLabs, 2012)	3.0	O	X	-	-	-	-	-
XEngine (Xengine, 2012)	2.0	X	O	-	X		X	-
XACML LIGHT (XACML LIGHT, 2012)	2.0	X	O	-	-	-	-	-

Tabela 2 - Implementações e suas características, onde “X” o atributo está presente, “O” não está e "-" não foi possível avaliar.

Legenda:

- Versão: Qual a versão do XACML, publicado pela OASIS, a implementação reproduz?
- Open-source: A implementação é livre, ou seja, respeita as normas de (Free Software Foundation, 2013).
- Ativa: Há desenvolvedores interagindo com o projeto?
- Conformidade: Há um conjunto de testes que valide os requisitos da OASIS?
- PDP: Há um Policy Decision Point na implementação?
- PEP: Há um Policy Enforcement Point na implementação?
- Documentação: Há algum manual, diagramas ou outro tipo de documento que auxilia o desenvolvimento?

- JAVADOC: Há javadoc para a implementação?

Atualmente, poucas implementações da versão 3.0 estão funcionando. Contudo, a versão 2.0, que é a mais utilizada no momento, está ativa desde 2005. Empresas como Paypal, Datev e Sweedish National Health Service fazem uso do sistema da WSO2 (BALANA, 2012). A Orange, empresa de telecomunicação da Europa, por outro lado, utiliza o Heras-af. Outras organizações permanecem anônimas, pois por se tratar de uma questão de segurança, expor a versão para um possível invasor, pode facilitar ataques.

Neste trabalho usaremos o Heras-AF, um projeto open-source, desenvolvido na *University of Applied Science Rapperswil*, está sendo utilizado por diversas empresas desde 2006.

Dentre as implementações estudadas foi a que cumpriu todos os requisitos, como uma implementação na linguagem Java, documentada e com JAVADOC. Outra qualidade é que a comunidade que implementa esta ferramenta está ativa e oferece suporte através de fóruns e e-mail. Além de possuir código fonte formatado, identado e sem estruturas longas e

desnecessárias.

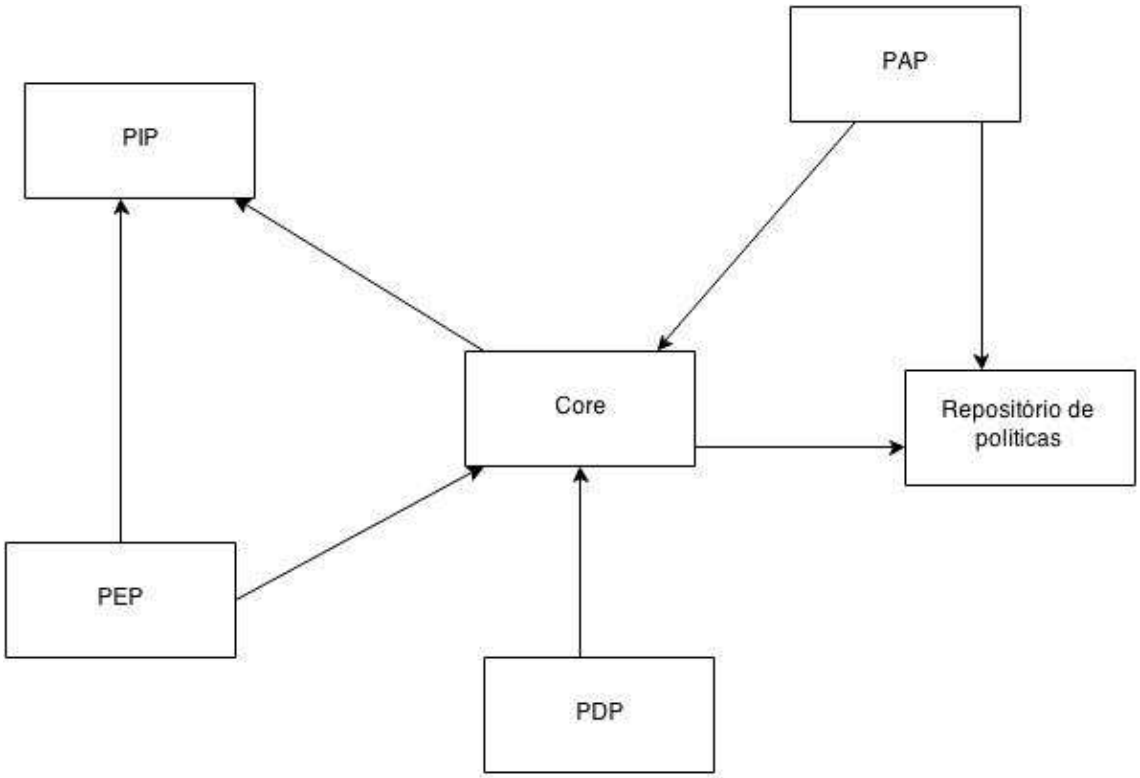

Figura 13 - Arquitetura do HERAS. (Heras-af arquitetura, 2010)

4.3 TRABALHOS RELACIONADOS

(Santos et al., 2013) e (MARINHO, 2014) mostraram um modelo expressivo para o controle de acesso dinâmico baseado em risco para computação em nuvem, definindo sua arquitetura e realizando testes.

(Santos et al., 2013) apresenta uma arquitetura construída a partir de uma extensão do XACML, fazendo o uso de políticas de risco. Além da arquitetura, (Santos et al., 2013) oferece uma forma de quantificar o risco e valida sua proposta através de uma implementação.

(Marinho, 2014) também oferece uma abordagem que possibilita o cálculo do risco, porém utiliza ontologias. O cálculo do risco de adapta conforme o número de fatores, não necessitando que todos sejam conhecidos. O trabalho é validado com um protótipo e são feitos testes com a quantidade de fatores disponíveis.

O trabalho de (Fall et al., 2011) tem como proposta a utilização do RadAc com alguns fatores de risco e a quantificação é feita utilizado com técnicas de aprendizado de máquina, porém nenhuma implementação ou simulação é realizada.

4.4RESULTADOS EXPERIMENTAIS

Uma aplicação deve ser feita para demonstrar os resultados. Para tanto, foi necessário montar uma estrutura para desenvolver a aplicação.

A estrutura utilizou a linguagem de programação Java na versão 8. Para facilitar o desenvolvimento e agregar funcionalidades, sem a necessidade de desenvolve-las, o Spring 4 foi utilizado. Dentre estas funcionalidades, podemos citar o controle de transação, que é feito na camada de persistência; a possível internacionalização de mensagens, permitindo tornar a aplicação multi-idioma; injeção de dependências; a divisão de camadas do sistema. Para a parte de visualização foi usado JSP 2.1, HTML 5 e CSS 3.

A divisão de camadas, conforme figura 14, começa na comunicação do browser com o controlador. O usuário clica ou digita uma URL (*Uniforme Resource Locator*) e a requisição é passada para o controlador.

O controlador é responsável por chamar o método que realiza a regra de negócio daquela URL (*Uniforme Resource Locator*) e envia a página HTML (*HyperText Markup Language*) para o *Browser*.

A camada de regra de negócio implementa as regras relativas ao domínio da aplicação.

A camada de entidade representa as abstrações feitas do mundo real que têm relação com a aplicação.

A camada de persistência separa a entidade das regras de acesso ao banco de dados, como conexão com o banco ou executar comandos SQL (*Structured Query Language*).

Por fim, o banco de dados salva as entidades.

Figura 14 - Estrutura da aplicação

Para tanto, visando viabilizar a distribuiçãodo XACML (PED e PDP) conforme figura 11, três máquinas virtuais foram criadas: Aplicação, PEP e PDP.

A aplicação hospedará um sistema para demonstrar o funcionamento deste trabalho. A visão geral é ilustrada na Figura 15 e melhor descrita nos próximos três parágrafos.

A aplicação terá como cenário o armazenamento de documentos eletrônicos, onde será possível listar os documentos e visualizar o mesmo,caso tenha permissão. Ao tentar visualizar um documento, a aplicação deverá

solicitar ao controle de acesso se usuário tem privilégios para acessá-lo. A tela principal da aplicação pode ser visto no apêndice D.

O PEP é responsável por transformar a requisição da aplicação em uma requisição XACML, enviar para o PDP e devolver uma resposta para a aplicação.

O PDP verifica, através de um atributo em uma tabela do banco de dados, se desejamos levar em consideraçãoo RadAc na decisão. Se não desejarmos, o PDP segue o fluxo normal do XACML. Se desejamos levar em consideração o RadAc, o PDP chama o DDP (*Dynamic Decision Point*) que segue o fluxo da Figura 4. Após seguir o fluxo, o resultado da política do XACML e do RadAc passam pelo algoritmo de combinação que possibilita agregar os resultados e transformar no resultado final.

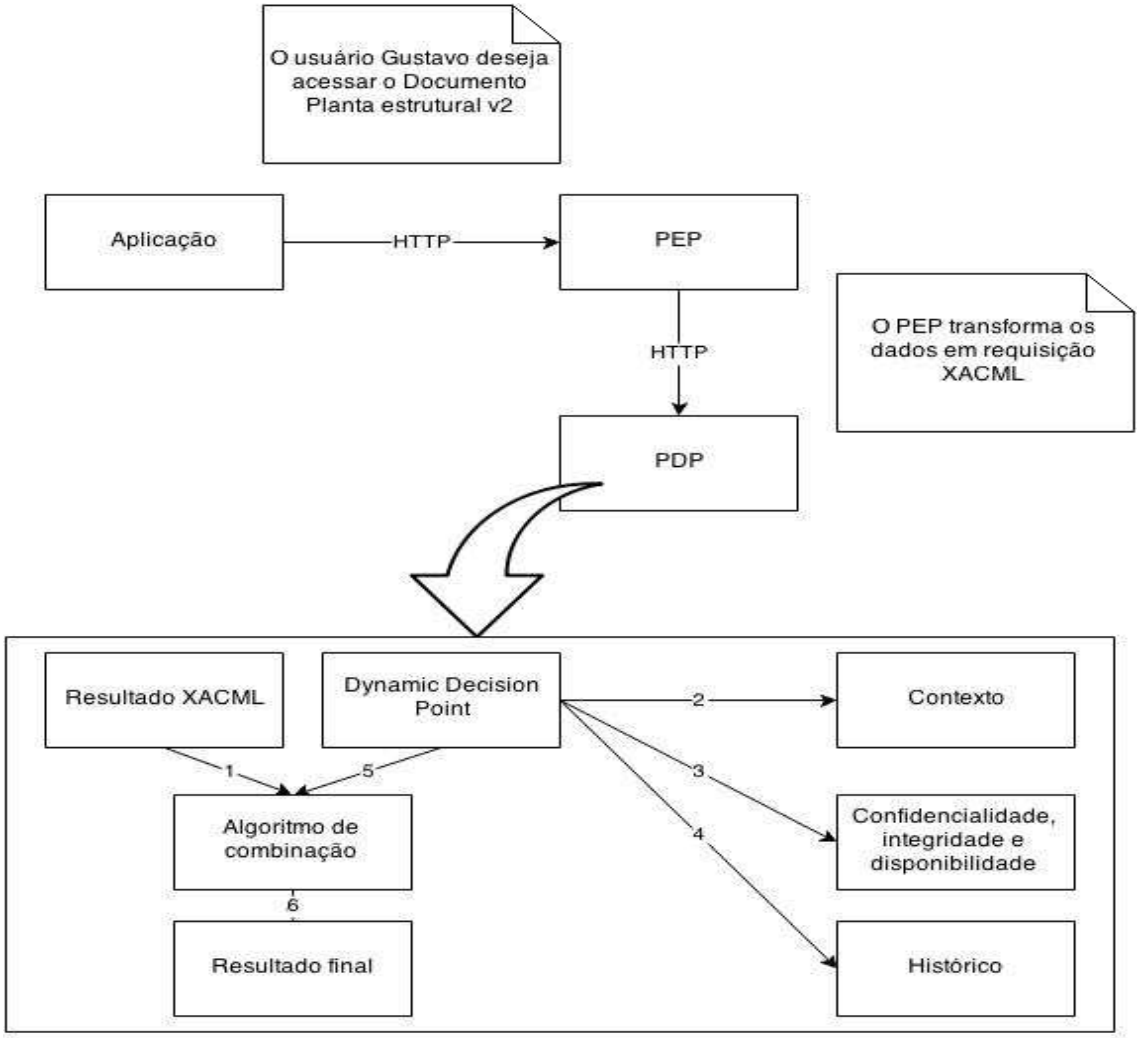

Figura 15 - Visão geral da implementação proposta

Para implantar o RadAc no XACML foi criado o DDP (*Dynamic Decision Point*), responsável por chamar o RadAc e agregar o resultado do XACML com o do RadAc. Estas responsabilidades são atribuídas através do método “avaliar”, ilustrado na figura 16. O método tem como parâmetro formal a decisão do XACML que posteriormente será utilizado no algoritmo de combinação. O DDP faz a chamada do método “avaliarRisco” do RadAc, que retorna a decisão do RadAc. Então, as decisões do RadAc e XACML são submetidas ao método “agregar” da classe “Algoritmo de Combinação”, onde a decisão final é retornada.

```java
public class DDP {

    @Autowired
    private RadAc radac;

    public boolean avaliar(boolean xacmlDecision) throws NaoHaRegistroException {
        boolean decisaoFinal = false;
        boolean radacDecision = radac.avaliarRisco();

        decisaoFinal = AlgoritmoDeCombinacao.agregar(xacmlDecision, radacDecision, decisaoFinal);

        return decisaoFinal;
    }
```

Figura 16 - Principal método do DDP.

Para utilizar o DDP é necessário fazer três configurações. Na primeira deve-se ativar ou desativar a possibilidade de calcular o risco. É possível definir o controle de acesso apenas com o XACML ou utilizar o XACML e o RadAc. É possível definir através do banco dedados na tabela “config”; A segunda é escolher qual o algoritmo de combinação utilizar; Na última configuração são escolhidos os valores de cada peso para calcular o risco. Os valores são definidos na tabela “peso”.

Regra	XACML	Risco	Decisão final
Deny overrides	DENY	PERMIT	DENY
Permit overrides	DENY	PERMIT	PERMIT
ABAC Precedence	DENY	PERMIT	DENY
Risk Precedence	DENY	PERMIT	PERMIT

Figura 17 - Algoritmo de combinação de Santos et al. (2013).

O RadAc, conforme diagrama de classe mostrado na figura 18, possui dentre outros métodos, o principal, “*avaliarRisco*” que enviando todas as informações necessárias, simula o fluxograma da figura 4. Este método é chamado pelo DDP.

O método “*calcularSecurityRisk*” faz o cálculo total do risco de segurança, buscando o peso de cada pilar no banco de dados e chamando os

métodos“*calcularRiscoContexto*”, “*calcularConfidencialidadeIntegridadeDisponibilidade*” e “*calcularHistorico*”. Com todos os três pilares calculados e seus pesos, basta colocar na fórmula abaixo.

Risco Total = peso1 * risco do contexto + peso2 * risco confidencialidade, integridade e disponibilidade + peso3 * risco considerando o histórico

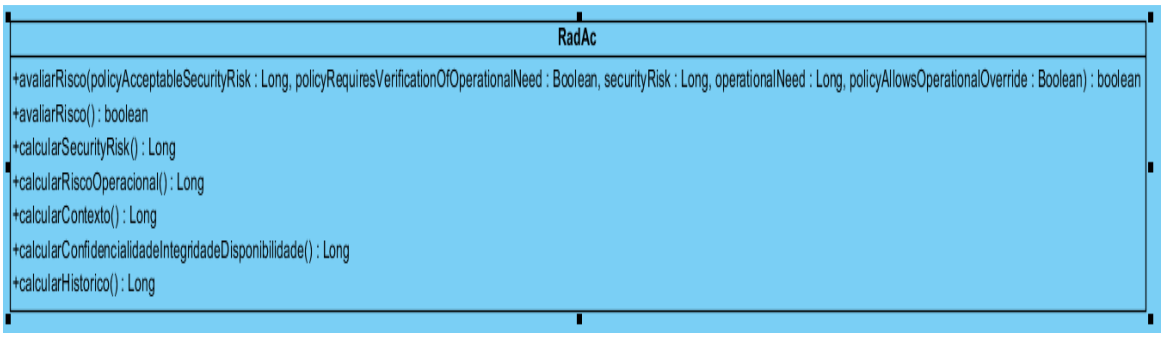

Figura 18 - Diagrama de classe proposta para o RadAc.

Para avaliar o desempenho deste trabalho, foram feitas medições relativas a três pontos de referência, o DDP, a aplicação e o navegador utilizado pelo usuário. A tabela 3 apresenta os valores mínimos, médias e máximos dos experimentos realizados.

Ponto de Referência	Mínimo (s)	Média (s)	Máximo (s)
DDP	0,442	0,465	0,524
Aplicação	0,840	0,897	1,129
Navegador	0.970	1,060	1,570

Tabela 3 - Desempenho

Nota-se que o tempo para o cálculo no DDP é pequeno, porém a demora maior é encontrada entre a Aplicação e o DDP, onde o maior

processamento é feito. Contudo, cerca de 1 segundo é um tempo viável para abrir uma página.

4.4.1 CASO DE USO

Para criar o caso de uso, criamos uma política que adeque-se ao cenário proposta. Esta política pode ser vista no apêndice B. A política permite acesso ao sujeito “Gustavo”, no recurso “Documento Estrutural”, poder fazer as ações de “Visualizar” e “Listar”. Para a requisição, utilizaremos a requisição que está no apêndice C. Portanto, a resposta do XACML será “*Permit*”.

No DDP configuramos para calcular o risco e utilizar o algoritmo de combinação “*Deny Overrides*”. Os pesos foram 50% para o contexto, 30% para confidencialidade, integridade e disponibilidade e 20% para o histórico. Portanto, teremos a fórmula conforme abaixo.

Risco Total = 50% * risco do contexto + 30% * risco confidencialidade, integridade e disponibilidade + 20% * risco considerando o histórico

Para completar a fórmula, devemos preencher os outros três riscos. Para calcular o risco do contexto foi utilizado os dados conforme a figura 17.

Fator de Risco	Peso	Atributo	Valor
Characteristics of Requester	16,666670,		
Role	2,777780,	Teantheader	7
Rank	2,777778,	E3	8
Clearance Level	2,777778,	Secret	4
Access Level	2,777778,	No	10
Previous Violations	2,777778,	No	0
Education Level	2,777778,	Phd	2
Characteristics of IT Components	16,666670,		
Machine Type	2,380952,	PDA	10
Application	2,380952,	Database	4
Connection Type	2,380952,	Wireless	8
Authentication Type	2,380952,	User/Password	7
Network	2,380952,	Internet	9
QoP/Encryption Level	2,380952,	WEP	5
Distance from requester to source	2,380952,	10000km	8
Heuristics	16,666670,		
Risk Knowledge	8,333333,	None	10
Trust Level	8,333333,	Low	9
Situational Factors	16,666670,		
Specific Mission Role	3,333333,	SuportTeam	2
Time Sensitivity of Information	3,333333,	Needed Now	2
Transaction Type	3,333333,	Query	2
Auditable or Non-auditable	3,333333,	Auditable	2
Audience Size	3,333333,	SinglePerson	2
Environmental Factors	16,666670,		
Current Location	8,333333,	Unknown	10
Operational Environment Threat Level	8,333333,	Severe	10
Characteristics of Information Requested	16,666670,		
Classification Level	3,333333,	TopSecret	10
Encryption Level	3,333333,	PHI	1
Network Classification Level	3,333333,	TWICS	9
Permission Level	3,333333,	ReadOnly	9
Perishable/Non-Perishable	3,333333,	NonPerishable	9

Figura 19 - Risco do contexto (Marinho, 2014).

Após fazer os cálculos, o valor encontrado é 701.

O risco relativo a confidencialidade, integridade e disponibilidade será considerado 250. O risco relativo ao histórico, consideraremos 600.

Portanto, o risco total será conforme abaixo.

Risco Total = 50% * 701 + 30% * 250 + 20% * 600

Risco Total = 545,5

Portanto, o risco de segurança é de 54. O fluxograma do RadAc, apresentado na figura 4, necessita de cinco atributos para ser calculado. Neste caso de uso, os valores para os atributos serão:

- Risco aceitável: 60
- Política requer verificação da necessidade operacional: Sim
- Risco de segurança: 54
- Necessidade operacional: 60
- Política permite que a necessidade operacional sobrescreva o risco de segurança: Sim

O RadAc verifica se o risco aceitável é maior que o risco de segurança, ou seja 60 é maior que 54. O segundo passo é verificar a política requer a verificação da necessidade operacional, neste caso sim. Portanto, o terceiro passo compara a necessidade operacional com o risco de segurança, neste caso a comparação é entre 60 e 52. Logo, a avaliação do RadAc resulta em "*Permit*".

Com as duas respostas, do XACML e RadAc, o DDP utiliza o algoritmo de combinação configurado que, neste caso, é o "*Deny Overrides*". Portanto, o resultado final será "*Permit*", ou seja, concederá acesso ao usuário.

5 CONCLUSÃO E TRABALHOS FUTUROS

Neste trabalho foi apresentado uma estrutura flexível, em conjunto com uma extensão do XACML, para avaliação de políticas de controle de acesso utilizando o RadAc.

Uma vez que o RadAc é um modelo complexo, torna-se oneroso e confuso o desenvolvimento de um protótipo em virtude dos diversos fatores associados ao seu fluxo. Contudo, este trabalho é um esforço para o desenvolvimento de um controle de acesso utilizando o RadAc.

Para utlização do RadAc, o risco de segurança do modelo foi calculado através do risco de contexto, confidencialidade, integridade e disponibilidade e histórico do contexto. Já o cenário desenvolvido foi o controle de documentos, onde o acesso a documentos é feito através de uma aplicação Web.

Ao final da implementação, os objetivos foram atingidos. Foi possível a criação da estrutura, a extensão do XACML, incluindo o RadAc, o cálculo do risco, a criação de um cenário a o desenvolvimento de uma aplicação de exemplo. Com a aplicação desenvolvida, foi possível realizar medições de desempenho, verificando o tempo de resposta da aplicação. Com a adição da estrutura neste trabalho, o tempo de execução, como esperado, foi maior, mas com desempenho aceitável. A utilização do algoritmo de combinação permite uma maneira customizável para a agregação entre os valores do XACML e do RadAc, de forma simples e transparente.

Com a aplicação, é possível criar diversos estudos relativos ao controle de acesso dinâmico utilizando o RadAc. Podendo verificar a efetividade do modelo de controle de acesso, assim como testes relativos ao cálculo de risco e seus fatores. Como a quantidade de fatores associados ao risco é grande, faz-se necessário uma maneira para distribuir os pesos para os fatores não conhecidos. Existem diversas maneiras, mas, por questão de tempo, não foi possível implementar.

A principal contribuição deste trabalho está no fomento ao RadAc e sua evolução, pois até o momento um número pequeno de implementações surgiram no meio acadêmico, e dentre estas, poucas possuem flexibilidade e possibilidade de estender para outros cenários.

Durante este trabalho, foram encontradas possibilidades para desenvolver trabalhos futuros. Dentre estes, podemos citar:

- O estudo da necessidade operacional e a definição de uma maneira de calculá-la de forma quantitativa;
- A utilização de métricas e fatores adequadas conforme o escopo da aplicação;
- A definição de fatores em tempo de execução.
- Utilizar técnicas de inteligência artificial para melhorar os fatores;
- Melhorar o desempenho;
- Realização de testes pertinentes ao cálculo do risco, visando refinar ainda mais o cálculo.

6 REFERÊNCIAS

[BALANA] BALANA the open-source XACML 3.0 implementation. (2012). Disponível em: < http://xacmlinfo.org/2012/08/16/balana-the-open-source-xacml-3-0-implementation/>. Acessado 01 de Setembro de 2014.

[BRITTON, D.; BROWN, I.] A security risk measurement for the RAdACmodel. [S.l.: s.n.], 2007.

[Cao and Yang 2010] Cao, Y. and Yang, L. (2010). A Survey of Identity Management

[Chadwick, D.] Federated identity management. (2009). Foundations of Security Analysis and Design V. Springer.

[CloudBees] CloudBees Customers (2014). Disponível em: <http://www.cloudbees.com/customers>. Acessado em 5 de Setembro de 2014.

[CloudStack LRG] CloudStack LRG (2014). Disponível em: <https://wiki.lrg.ufsc.br/mediaWiki/index.php/Cloud>. Acessado em 3 de novembro de 2014.

[D. Gollmann] DIETER GOLLMAN. Computer security. Chichester: Wiley, 1999. 320 p.

[ENTERPRISE] Enterprise Java XACML implementation. (2012). Disponível em: https://code.google.com/p/enterprise-java-xacml//>. Acessado 01 de Setembro de 2014.

[FALL, D. et al. 2011] Toward Quantified Risk-Adaptive Access Control for Multi-tenant Cloud Computing. In: Proceedings of the 6th Joint Workshop on Information Security (JWIS2011), 2011.

[HERAS] Heras-af XACML (2012). Disponível em: <http://www.herasaf.org/heras-af-xacml.html>. Acessado em 1 de Setembro de 2014.

[IDC] International Data Corporation. (2012). Disponível em: <http://blogs.idc.com/ie/?p=210 >. Acessado em 15 de janeiro de 2013.

[IBM] XML Security: Control information access with XACML the objectives, architecture, and basic concepts of eXtensible Access Control Markup Language. (2004). Disponível em: <http://www.ibm.com/developerworks/xml/library/x-xacml/ >. Acessado em 12 de dezembro de 2012.

[ITU-T 2009] ITU-T (2009). NGN Identity Management Framework. Recommendation Y.2720. Technical report. Disponível em: <http://www.itu.int/itut/recommendations/rec.aspx?id=9574>. Acessado em 12 de Outubro de 2012.

[KANDALA, S.; SANDHU, R.; BHAMIDIPATI, V.], "An Attribute Based Framework for Risk-Adaptive Access Control Models," Availability, Reliability and Security (ARES), 2011 Sixth International Conference on , vol., no., pp.236,241, 22-26 Ago 2011

[MARINHO, Roberto.] UM MODELO PARA AVALIAÇÃO DINÂMICA DE RISCO UTILIZANDO ONTOLOGIA. 2014. 110 f. Dissertação (Mestrado) - Curso de Ciências da Computação, Departamento de Informática e Estatística, Universidade Federal de Santa Catarina, Florianópolis, 2014.

[M. Benantar.] Access Control Systems: Security, Identity Management and Trust Models. Springer, New York, NY, 2006.

[MCGRAW, R.] Risk Adaptive Access Control. 2009. National Security Agency.

[MOHAMED, A.] A history of Cloud Computing. (2009). Disponível em: <http://www.computerweekly.com/feature/A-history-of-cloud-computing>. Acessado em 12 de Outubro de 2012.

[NextLabs] XACML-based information control plataform. (2012). Disponível em: <http://www.nextlabs.com/html/?q=xacml-based>. Acessado em 30 de Setembro de 2013.

[NIST] The NIST definitions of Cloud Computing. (2011). Disponível em: <http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf >. Acessado em 11 de Outubro de 2012.

[OASIS] About Us. Disponível em: <https://www.oasis-open.org/org>. Acessado em 12 maio 2011.

[OASIS] Specifications and standards. (2005). Disponível em: <http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf >. Acessado 29 de Março de 2013.

[OpenStack] OpenStack architecture. (2013). Disponível em: <http://openvswitch.org/openstack/ >. Acessado em 07 de maio de 2013.

[PARKHILL, D.] Douglas F. Parkhill. (1966). the Challenge of the Computer Utility. Addison-Wesley Educational Publishers Inc., US.

[PEPPLE, K] Deploying OpenStack. Creating open source clouds. (2011). Primeira Edição. O'Reilly Media. pp. 2-4. Califórnia, Estados Unidos

[SANDHU, R.; FERRAIOLO, D.; KUHN, R.] The nist model for role-basedAccess control: towards a unified standard. In: Proceedings of the fifth ACMWorkshop on Role-based access control. New York, NY, USA: ACM, 2000. (RBAC ’00), p. 47–63. ISBN 1-58113-259-X.

[SANTOS, D. R.; WESTPHALL, C.M.; WESTPHALL, C.B.] RISK-BASED DYNAMIC ACCESS CONTROL FOR A HIGHLY SCALABLE CLOUD FEDERATION". IN: SECURWARE 2013 - THE SEVENTH INTERNATIONAL CONFERENCE ON EMERGING SECURITY INFORMATION, SYSTEMS AND TECHNOLOGIES, 7TH EDITION, BARCELONA. PROCEEDINGS IARIA: XPS PRESS, 2013.

[SARIPALLI, P.; WALTERS, B.] QUIRC: A Quantitative Impact and Risk Assessment Framework for Cloud Security. In: 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD), 3 rd edition, USA. Proceedings... IEEE: IEEE Press, 2010. pp. 280-288. doi: 10.1109/CLOUD.2010.22.

[SUN] Sun's XACML Implementation (2012). Disponível em: <http://sunxacml.sourceforge.net/>. Acessado em 15 de janeiro de 2014.

[V. C. Hu, D. F. Ferraiolo, and D. R. Kuhn] "Assessment of Access Control Systems," National Institute of Standards and Technology (NIST), 2006.
[Heras-af arquitetura] HERAS-AF XACML. (2010). Disponível em: <http://www.herasaf.org/heras-af-xacml.html>. Acessado em 02 de novembro de 2014.

[XACML LIGHT] XACML Light reference. (2012). Disponível em: <http://xacmllight.sourceforge.net/>. Acessado em 01 de setembro de 2014.

APÊNDICES

APÊNDICE A -Tabela para cálculo do contexto

Fator de Risco	Peso
Characteristics of Requester	16.66667
Role	2.777778
Rank	2.777778
Clearance Level	2.777778
Access Level	2.777778
Previous Violations	2.777778
Education Level	2.777778
Characteristics of IT Components	16.66667
Machine Type	2.380952
Application	2.380952
Connection Type	2.380952
Authentication Type	2.380952
Network	2.380952
QoP/Encryption Level	2.380952
Distance from requester to source	2.380952
Heuristics	16.66667
Risk Knowledge	8.333333
Trust Level	8.333333
Situational Factors	16.66667
Specific Mission Role	3.333333
Time Sensitivity of Information	3.333333
Transaction Type	3.333333
Auditable or Non-auditable	3.333333
Audience Size	3.333333
Environmental Factors	16.66667
Current Location	8.333333
Operational Environment Threat Level	8.333333
Characteristics of Information Requested	16.66667
Classification Level	3.333333
Encryption Level	3.333333
Network Classification Level	3.333333
Permission Level	3.333333
Perishable/ Non-Perishable	3.333333

(BRITTON, D.; BROWN, I., 2007)

APÊNDICE B– Política de controle de acesso utilizado nos casos de uso

```
<Rule
      RuleId="urn:oasis:names:tc:xacml:2.0:conformance-test:IIA1:rule"
      Effect="Permit">
    <Description>
    </Description>
    <Target>
        <Subjects>
            <Subject>
                <SubjectMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                          DataType="http://www.w3.org/2001/XMLSchema#string">Gustavo</AttributeValue>
                    <SubjectAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
                          DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </SubjectMatch>
            </Subject>
        </Subjects>
        <Resources>
            <Resource>
                <ResourceMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                          DataType="http://www.w3.org/2001/XMLSchema#string">Documento Estrutural</AttributeValue>
                    <ResourceAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
                          DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ResourceMatch>
            </Resource>
        </Resources>
        <Actions>
            <Action>
                <ActionMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                          DataType="http://www.w3.org/2001/XMLSchema#string">Visualizar</AttributeValue>
                    <ActionAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
                          DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
            <Action>
                <ActionMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                          DataType="http://www.w3.org/2001/XMLSchema#string">Listar</AttributeValue>
                    <ActionAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
                          DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
        </Actions>
    </Target>
</Rule>
```

APÊNDICE C– Requisição de controle de acesso utilizado nos casos de uso

```
<Request
      xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:context:schema:os
        access_control-xacml-2.0-context-schema-os.xsd">
    <Subject>
        <Attribute
              AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
              DataType="http://www.w3.org/2001/XMLSchema#string">
            <AttributeValue>Gustavo</AttributeValue>
        </Attribute>
    </Subject>
    <Resource>
        <Attribute
              AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
              DataType="http://www.w3.org/2001/XMLSchema#string">
            <AttributeValue>Documento Estrutural</AttributeValue>
        </Attribute>
    </Resource>
    <Action>
        <Attribute
              AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
              DataType="http://www.w3.org/2001/XMLSchema#string">
            <AttributeValue>Visualizar</AttributeValue>
        </Attribute>
    </Action>
    <Environment/>
</Request>
```

APÊNDICE D– Listagem de documentos da aplicação

Titulo	Versão do documento	Código	Autor
Documento Estrutural	2A	1001-EN-DE	Gustavo
Planta baixa	2A	1001-EY-FE	Gustavo
Planta alta	2A	1001-EP-UN	Gustavo

ANEXO – Artigo

Utilização do modelo de controle de acesso RADAC (Risk-adaptive access control) em um ambiente de computação em nuvem

Gustavo R. Schmitt[1], Carla M. Westphall[2]

[1]Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – 88.040-900 – Florianópolis – SC – Brasil

gustavorschmitt@grad.ufsc.br, carlamw@inf.ufsc.br

Abstract. *This meta-paper describes the style to be used in articles and short papers for SBC conferences. For papers in English, you should add just an abstract while for the papers in Portuguese, we also ask for an abstract in Portuguese (“resumo”). In both cases, abstracts should not have more than 10 lines and must be in the first page of the paper.*

Resumo. *O presente trabalho tem como objetivo oferecer uma implementação do modelo RadAc (Risk-Adaptive Acess Control) utilizando o XACML, em ambiente de computação em nuvem, visando calcular o risco de segurança de forma dinâmica. Na proposta apresentada, o XACML recebe um novo componente responsável por seguir o modelo RadAc e calcular o risco de segurança. O risco de segurança é composto três pilares: riscos de contexto, riscos considerando confidencialidade, integridade e disponibilidade e risco considerando o histórico do sujeito. Um algoritmo que combina as decisões do XACML e RadAc é apresentado para agregar as duas decisões em uma.*

1. INTRODUÇÃO

Com o crescimento da computação em nuvem são levantadas várias questões de segurança envolvendo desde arquivos de usuários a grandes volumes de informações em banco de dados. Dado que o modelo de multi-inquilino possibilita mais de um usuário usar o mesmo software compartilhando a mesma estrutura física e virtual, usuários ficam com medo de expor suas informações por tratar-se de um ambiente dinâmico. A segurança é um fator primordial na nuvem. Com ferramentas que proporcionam mais segurança haverá ainda mais usuários transferindo seus dados para a mesma. A migração dos recursos e informações para a nuvem computacional acarreta uma grande massa de dados que pode ser alvo de ataques.

Alguns modelos tradicionais de controle de acesso usados atualmente fornecem o controle de acesso de grão fino. Normalmente, os modelos de controle de acesso tradicionais são estáticos, isto é, tem regras que não mudam durante o acesso dos usuários. Entretanto, ambientes de computação em nuvem são dinâmicos, e os atuais modelos de controle de acesso não têm tanta flexibilidade para suportar um ambiente em nuvem (McGraw. R, 2009). Portanto, um mecanismo que controle o acesso de maneira dinâmica resolve o problema da falta de flexibilidade SANTOS et al., (2013).

2. RADAC

O RadAc (Risk Adaptative Access Control) foi proposto pela NSA (National Security Agency) como uma nova abordagem no âmbito de controle de acesso. O novo paradigma propõe o controle de acesso de uma forma dinâmica, que emula as decisões do mundo real (McGraw. R, 2009). Atualmente, o fator risco é algo estático nas abordagens, o RadAc surge para suprir esta lacuna. O conceito de necessidade operacional (Operational need) refere-se a necessidade real, no momento da requisição, do sujeito ter ou não acesso à informação.

O risco de segurança (Security risk) é alguma medida, quantitativa ou qualitativa, dos riscos associados a conceder o acesso a informação. Um exemplo poderia ser caso o usuário não use uma conexão criptografada, o risco de segurança aumenta pois a conexão pode não ser segura. Mais do que comparar atributos, como nos modelos tradicionais, o RadAc calcula o risco de segurança no momento da requisição.

A figura 1 ilustra os possíveis fatores que o RadAc pode considerar para avaliar uma requisição, como características do usuário, componentes de tecnologia e outros.

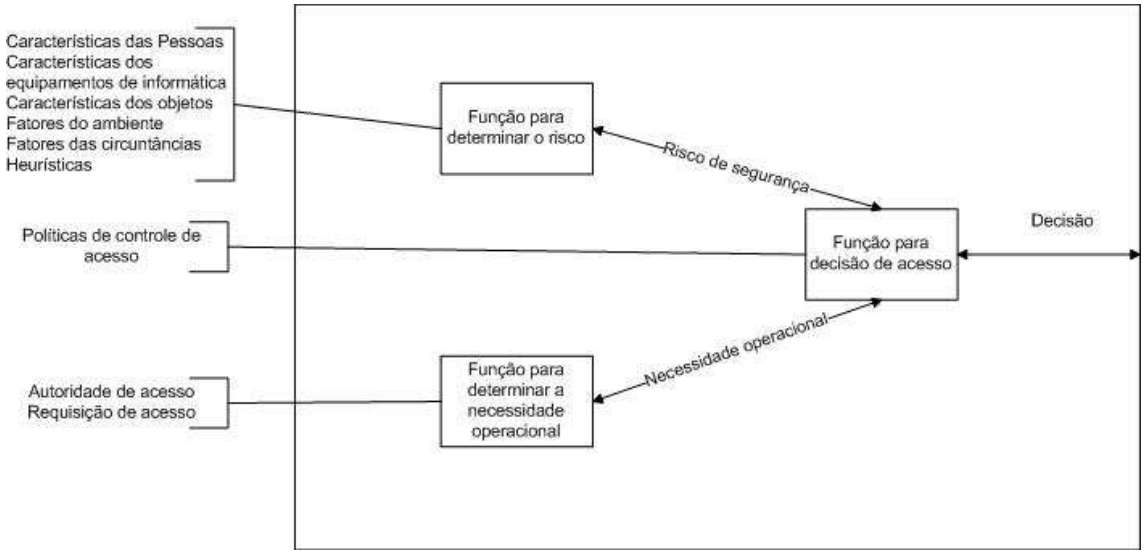

Figura 1. A estrutura do RadAc

3. XACML

O XACML (eXtensible Access Control Markup Language) é um padrão, criado pela OASIS (Advancing Open Stadards for the Information Society), para definir políticas de controle de acesso, ambas desenvolvidas em XML (eXtensible Markup Language). A linguagem é declarativa, usada para descrever situações genéricas (OASIS, 2011).

Diversas linguagens foram propostas, mas o XACML apresenta algumas vantagens, como: uma linguagem que foi testada por uma comunidade de usuários e desenvolvedores, gerando menor probabilidade de ocorrerem erros; é possível usar em qualquer ambiente, se adapta a diversas situações, visto que é uma linguagem genérica; ser distribuída por natureza; trata-se de um padrão. O XACML é, basicamente, dividido em dois componentes. O componente que realiza a lógica para realizar uma avaliação e retornar uma resposta. E outro, usado dentro da aplicação, para fornecer requisições ao anterior. Estes dois principais componentes são, respectivamente:

- Policy Decision Point (PDP): O componente que realiza a decisão de controle de acesso, com base nas políticas e informações recuperadas;

- Policy Enforcement Point (PEP) O ponto de acesso do usuário ao sistema. É o ponto que protege um recurso sensível, recebendo a requisição de acesso e enviando-a ao PDP.

A figura 2 ilustra o processo de funcionamento do XACML. O primeiro passo é receber uma requisição no PEP. O PEP envia uma requisição de acesso ao PDP. O PDP faz o processo de verificação e retorna uma resposta. O PEP então retorna à aplicação uma resposta, garantindo ou negando o acesso.

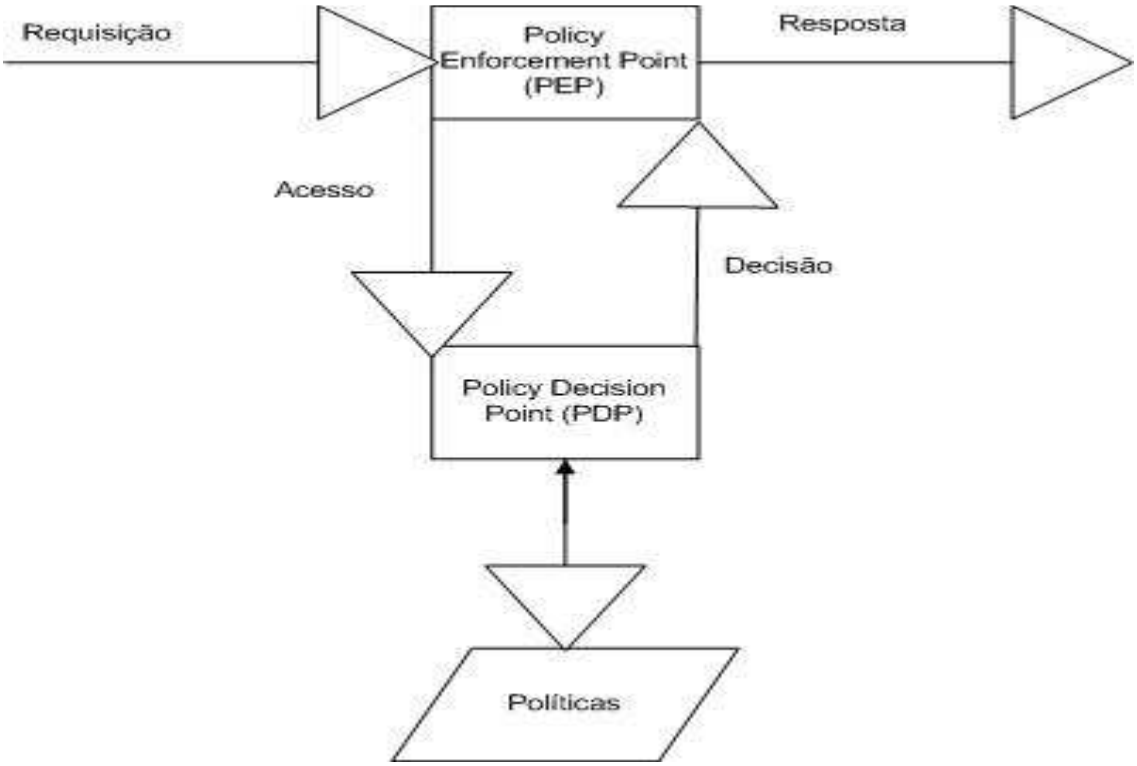

Figura 2. Fluxo do XACML.

4. PROPOSTA

A infraestrutura proposta é ilustrada na figura abaixo, são 3 máquinas virtuais, a primeira roda uma aplicação de exemplo, a segunda o PEP e a terceira o PDP. O usuário tenta acessar um recurso qualquer, como um documento. A aplicação faz uma requisição ao PEP que cria uma requisição para o PDP. O PDP então avalia a requisição e define um resultado, o RadAc calculará o risco de segurança em conceder acesso e a necessidade operacional.

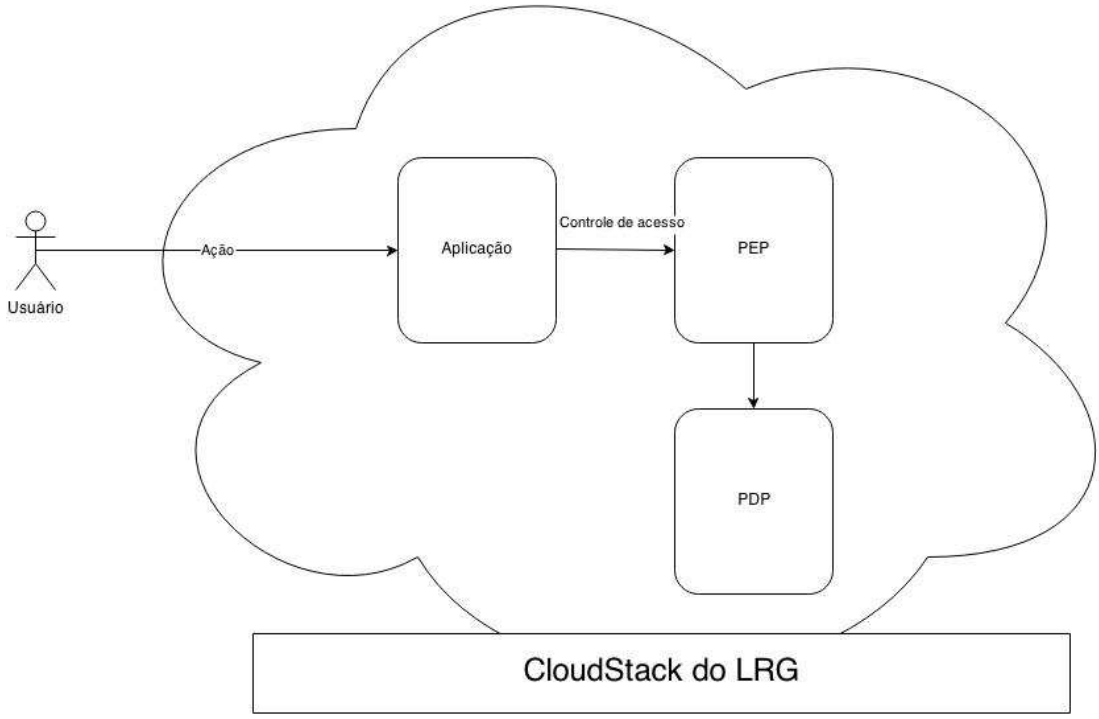

Figura 3. Diagrama geral da proposta

Com o resultado do XACML e do RadAc é utilizado um algoritmo de agregação para gerar o resultado final.

Uma das principais dificuldades na implementação do modelo RadAc consiste em encontrar uma forma de calcular o risco. SANTOS et al., (2013) e (Marinho, 2014) trazem modelos que quantificam o risco para utilizar no modelo RadAc. Ambos utilizam a abordagem que leva em consideração fatores do contexto e confidencialidade, integridade e disponibilidade. Além disso, Marinho utiliza também o histórico do usuário.

Para calcular os valores relativos ao contexto, ambos utilizam o trabalho de (BRITTON, D.; BROWN, I., 2007), onde os fatores de risco são classificados e atribuídos pesos.Este método apresenta um modelo para o cálculo de risco do RadAc, através de 27 métricas divididas em 6 grupos, onde cada métrica possui um risco associado. Tanto os pesos quanto as métricas são atribuídos por especialistas. A tabela pode ser vista no apêndice A.

Os valores de confidencialidade, integridade e disponibilidade são calculados a partir das do trabalho de (SARIPALLI, P.; WALTERS, B ,2010). O risco é calculado a partir da seguinte fórmula:

Risco = Probabilidade * Impacto

A probabilidade é a quantidade de tentativas de violação de acesso divido pela quantidade total de acessos (SARIPALLI, P.; WALTERS, B ,2010).

O impacto é calculado a partir do dano que seu acesso pode vir a ocasionar. (SARIPALLI, P.; WALTERS, B ,2010) sugere os seguintes valores: impacto baixo (1-5), impacto moderado (6-10) e impacto alto (11-15).

O risco relativo ao histórico do sujeito, contemplado no modelo de (Marinho, 2014), sugere que exista uma pontuação para cada usuário. Esta pontuação é compreendida numa escala de 0 a 10, de modo que este valor é decrementado quando ações positivas são feitas e incrementado quando ações negativas são realizadas. Portanto, quanto maior o valor do histórico, maior o valor do risco total.

Com base nestes trabalhos quantificaremos o risco através da seguinte formula:

Risco Total = peso1 * risco do contexto + peso2 * risco confidencialidade, integridade e disponibilidade + peso3 * risco considerando o historic

Os pesos serão carregados através do banco de dados, oferecendo uma maior flexibilidade para a estrutura.

Para incluir o RadAc e possibilitar o cálculo do risco, incluiremos um novo componente ao Heras-af:

Dynamic Decision Point: Responsável por analisar e quantificar o risco, este componente diretamente ligado ao PDP para fornecer uma resposta o mais rápido possível.

Incluindo o novo componente ao Heras-af, a sua arquitetura ficará conforme a figura abaixo.

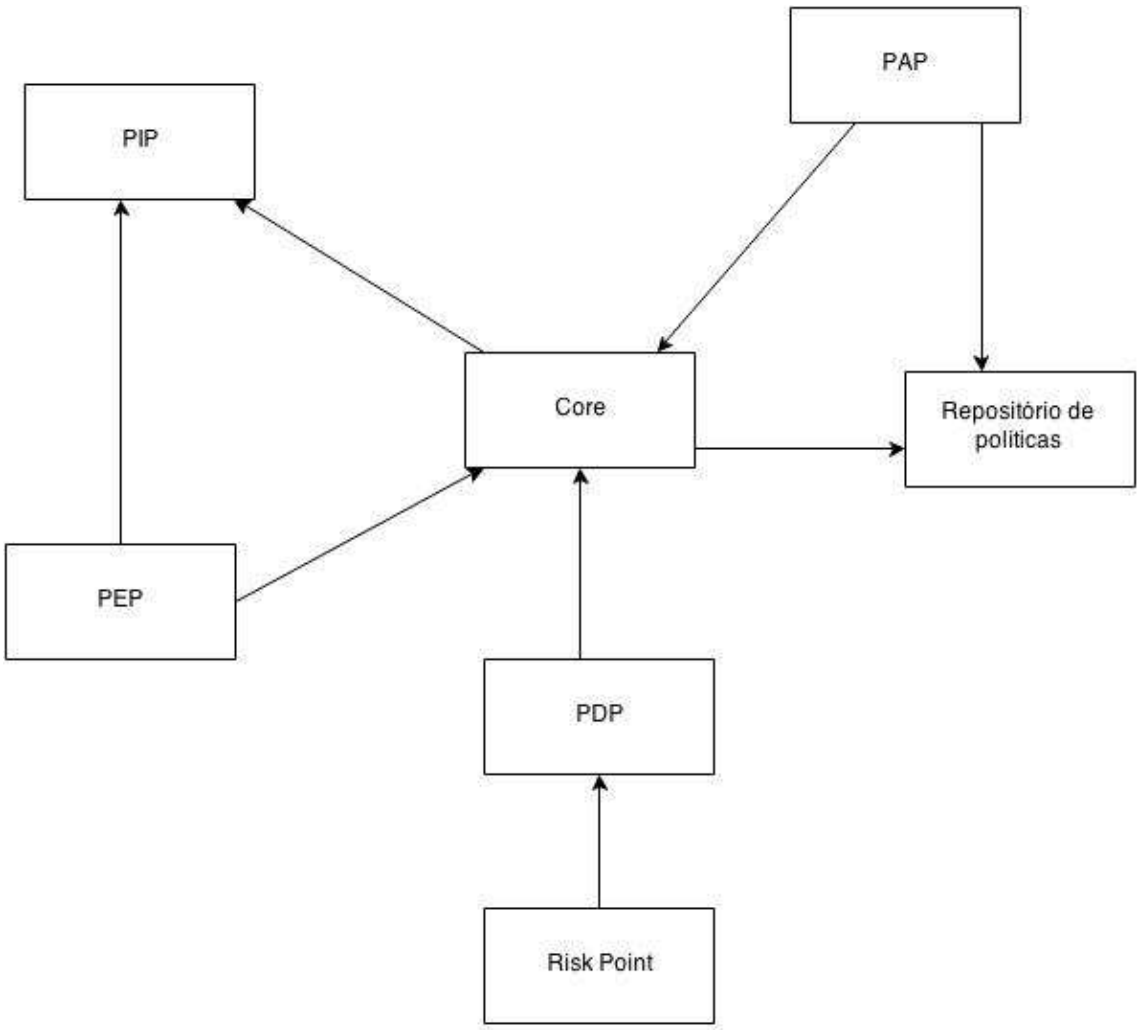

Figura 4. Proposta de extensão do XACML

5. RESULTADOS

Uma aplicação deve ser feita para demonstrar os resultados. Para tanto, foi necessário montar uma estrutura para desenvolver a aplicação.

A estrutura utilizou a linguagem de programação Java na versão 8. Para facilitar o desenvolvimento e agregar funcionalidades, sem a necessidade de desenvolve-las, o Spring 4 foi utilizado. Dentre estas funcionalidades, podemos citar o controle

de transação, que é feito na camada de persistência; a possível internacionalização de mensagens, permitindo tornar a aplicação multi-idioma; injeção de dependências; a divisão de camadas do sistema.Para a parte de visualização foi usado JSP 2.1, HTML 5 e CSS 3.

A divisão de camadas, conforme figura 5, começa na comunicação do browser com o controlador. O usuário clica ou digita uma URL (*Uniforme Resource Locator*) e a requisição é passada para o controlador. O controlador é responsável por chamar o método que realiza a regra de negócio daquela URL (*Uniforme Resource Locator*) e envia a página HTML (*HyperText Markup Language*) para o Browser. A camada de regra de negócio implementa as regras relativas ao domínio da aplicação.

A camada de entidade representa as abstrações feitas do mundo real que têm relação com a aplicação. A camada de persistência separa a entidade das regras de acesso ao banco de dados, como conexão com o banco ou executar comandos SQL (Structured Query Language).

Por fim, o banco de dados salva as entidades.

Figura 5. Estrutura da aplicação

Para tanto, visando viabilizar a distribuiçãodo XACML (PED e PDP) conforme figura 6, três máquinas virtuais foram criadas: Aplicação, PEP e PDP.

A aplicação hospedará um sistema para demonstrar o funcionamento deste trabalho. A visão geral é ilustrada na figura 6 e melhor descrita nos próximos três parágrafos.

A aplicação terá como cenário o armazenamento de documentos eletrônicos, onde será possível listar os documentos e visualizar o mesmo,caso tenha permissão. Ao tentar visualizar um documento, a aplicação deverá solicitar ao controle de acesso se

usuário tem privilégios para acessá-lo. A tela principal da aplicação pode ser no apêndice D.

O PEP é responsável por transformar a requisição da aplicação em uma requisição XACML, enviar para o PDP e devolver uma resposta para a aplicação.

O PDP verifica, através de um atributo em uma tabela do banco de dados, se desejamos levar em consideraçãoo RadAc na decisão. Se não desejarmos, o PDP segue o fluxo normal do XACML. Se desejamos levar em consideração o RadAc, o PDP chama o DDP (*Dynamic Decision Point*) que segue o fluxo. Após seguir o fluxo, o resultado da política do XACML e do RadAc passam pelo algoritmo de combinação que possibilita agregar os resultados e transformar no resultado final.

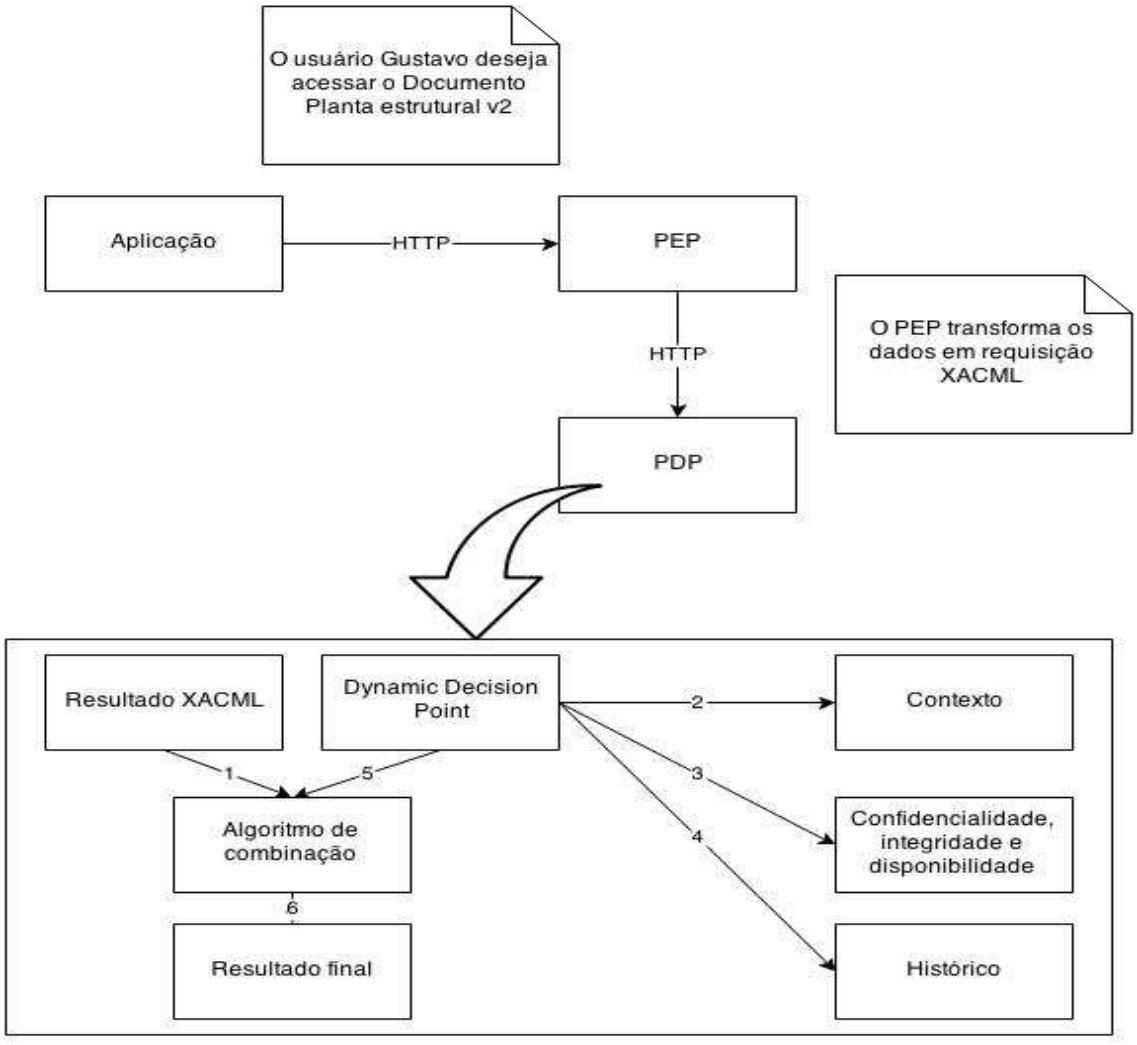

Figura 6. Visão geral da proposta

Para implantar o RadAc no XACML foi criado o DDP (*Dynamic Decision Point*), responsável por chamar o RadAc e agregar o resultado do XACML com o do RadAc.

Para utilizar o DDP é necessário fazer três configurações. Na primeira deve-se ativar ou desativar a possibilidade de calcular o risco. É possível definir o controle de acesso apenas com o XACML ou utilizar o XACML e o RadAc. É possível definir através do banco de dados na tabela “config”; A segunda é escolherqual o algoritmo de combinação utilizar; Na última configuração são escolhidos os valores de cada peso para calcular o risco. Os valores são definidos na tabela “peso”.

Regra	XACML	Risco	Decisão final
Deny overrides	DENY	PERMIT	DENY
Permit overrides	DENY	PERMIT	PERMIT
ABAC Precedence	DENY	PERMIT	DENY
Risk Precedence	DENY	PERMIT	PERMIT

Figura 7. Algoritmo de combinação de Santos et al., (2013)

5.1. CASO DE USO

Para criar o caso de uso, criamos uma política que adeque-se ao cenário proposta. Esta política pode ser vista na figura 8. A política permite acesso ao sujeito “Gustavo”, no recurso “Documento Estrutural”, poder fazer as ações de “Visualizar” e “Listar”. Para a requisição, utilizaremos a requisição que está na figura 9. Portanto, a resposta do XACML será “*Permit*”.

```
<Rule
      RuleId="urn:oasis:names:tc:xacml:2.0:conformance-test:IIA1:rule"
      Effect="Permit">
    <Description>
    </Description>
    <Target>
        <Subjects>
            <Subject>
                <SubjectMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                          DataType="http://www.w3.org/2001/XMLSchema#string">Gustavo</AttributeValue>
                    <SubjectAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
                          DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </SubjectMatch>
            </Subject>
        </Subjects>
        <Resources>
            <Resource>
                <ResourceMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                          DataType="http://www.w3.org/2001/XMLSchema#string">Documento Estrutural</AttributeValue>
                    <ResourceAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
                          DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ResourceMatch>
            </Resource>
        </Resources>
        <Actions>
            <Action>
                <ActionMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                          DataType="http://www.w3.org/2001/XMLSchema#string">Visualizar</AttributeValue>
                    <ActionAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
                          DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
            <Action>
                <ActionMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                          DataType="http://www.w3.org/2001/XMLSchema#string">Listar</AttributeValue>
                    <ActionAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
                          DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
        </Actions>
    </Target>
</Rule>
```

Figura 8. Política XACML

```
<Request
      xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:context:schema:os
        access_control-xacml-2.0-context-schema-os.xsd">
    <Subject>
        <Attribute
              AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
              DataType="http://www.w3.org/2001/XMLSchema#string">
            <AttributeValue>Gustavo</AttributeValue>
        </Attribute>
    </Subject>
    <Resource>
        <Attribute
              AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
              DataType="http://www.w3.org/2001/XMLSchema#string">
            <AttributeValue>Documento Estrutural</AttributeValue>
        </Attribute>
    </Resource>
    <Action>
        <Attribute
              AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
              DataType="http://www.w3.org/2001/XMLSchema#string">
            <AttributeValue>Visualizar</AttributeValue>
        </Attribute>
    </Action>
    <Environment/>
</Request>
```

Figura 9. Requisição XACML

No DDP configuramos para calcular o risco e utilizar o algoritmo de combinação “*Deny Overrides*”. Os pesos foram 50%

para o contexto, 30% para confidencialidade, integridade e disponibilidade e 20% para o histórico. Portanto, teremos a fórmula conforme abaixo.

Risco Total = 50% * risco do contexto + 30% * risco confidencialidade, integridade e disponibilidade + 20% * risco considerando o histórico

Para completar a fórmula, devemos preencher os outros três riscos. Para calcular o risco do contexto foi utilizado os dados conforme a figura 10.

Fator de Risco	Peso	Atributo	Valor
Characteristics of Requester	16,666670,		
Role	2,777780,	Teantheader	7
Rank	2,777778,	E3	8
Clearance Level	2,777778,	Secret	4
Access Level	2,777778,	No	10
Previous Violations	2,777778,	No	0
Education Level	2,777778,	Phd	2
Characteristics of IT Components	16,666670,		
Machine Type	2,380952,	PDA	10
Application	2,380952,	Database	4
Connection Type	2,380952,	Wireless	8
Authentication Type	2,380952,	User/Password	7
Network	2,380952,	Internet	9
QoP/Encryption Level	2,380952,	WEP	5
Distance from requester to source	2,380952,	10000km	8
Heuristics	16,666670,		
Risk Knowledge	8,333333,	None	10
Trust Level	8,333333,	Low	9
Situational Factors	16,666670,		
Specific Mission Role	3,333333,	SuportTeam	2
Time Sensitivity of Information	3,333333,	Needed Now	2
Transaction Type	3,333333,	Query	2
Auditable or Non-auditable	3,333333,	Auditable	2
Audience Size	3,333333,	SinglePerson	2
Environmental Factors	16,666670,		
Current Location	8,333333,	Unknown	10
Operational Environment Threat Level	8,333333,	Severe	10
Characteristics of Information Requested	16,666670,		
Classification Level	3,333333,	TopSecret	10
Encryption Level	3,333333,	PHI	1
Network Classification Level	3,333333,	TWICS	9
Permission Level	3,333333,	ReadOnly	9
Perishable/Non-Perishable	3,333333,	NonPerishable	9

Figura 10. Risco do contexto

Após fazer os cálculos, o valor encontrado é 701.O risco relativo a confidencialidade, integridade e disponibilidade será considerado 250. O risco relativo ao histórico, consideraremos 600. Portanto, o risco total será conforme abaixo.

Risco Total = 50% * 701 + 30% * 250 + 20% * 600

Risco Total = 5455

Portanto, o risco de segurança é de 54. Para os calculos do RadAc, são necessários cinco atributos. Neste caso de uso, os valores para os atributos serão:

- Risco aceitável: 60
- Política requer verificação da necessidade operacional: Sim
- Risco de segurança: 54
- Necessidade operacional: 60
- Política permite que a necessidade operacional sobrescreva o risco de segurança: Sim

O RadAc verifica se o risco aceitável é maior que o risco de segurança, ou seja 60 é maior que 54. O segundo passo é verificar a política requer a verificação da necessidade operacional, neste caso sim. Portanto, o terceiro passo compara a necessidade operacional com o risco de segurança, neste caso a comparação é entre 60 e 52. Logo, a avaliação do RadAc resulta em "*Permit*".

Com as duas respostas, do XACML e RadAc, o DDP utiliza o algoritmo de combinação configurado que, neste caso, é o "*Deny Overrides*". Portanto, o resultado final será "*Permit*", ou seja, concederá acesso ao usuário.

6. CONCLUSÃO E TRABALHOS FUTUROS

Neste trabalho foi apresentado uma estrutura flexível, em conjunto com uma extensão do XACML, para avaliação de políticas de controle de acesso utilizando o RadAc.

Uma vez que o RadAc é um modelo complexo, torna-se oneroso e confuso o desenvolvimento de um protótipo em virtude dos diversos fatores associados ao seu fluxo. Contudo, este trabalho é um esforço para o desenvolvimento de um controle de acesso utilizando o RadAc.

Para utlização do RadAc, o risco de segurança do modelo foi calculado através do risco de contexto, confidencialidade, integridade e disponibilidade e histórico do contexto. Já o cenário desenvolvido foi o controle de documentos, onde o acesso a documentos é feito através de uma aplicação Web.

Ao final da implementação, os objetivos foram atingidos. Foi possível a criação da estrutura, a extensão do XACML, incluindo o

RadAc, o cálculo do risco, a criação de um cenário a o desenvolvimento de uma aplicação de exemplo. Com a aplicação desenvolvida, foi possível realizar medições de desempenho, verificando o tempo de resposta da aplicação. Com a adição da estrutura neste trabalho, o tempo de execução, como esperado, foi maior, mas com desempenho aceitável. A utilização do algoritmo de combinação permite uma maneira customizável para a agregação entre os valores do XACML e do RadAc, de forma simples e transparente.

Com a aplicação, é possível criar diversos estudos relativos ao controle de acesso dinâmico utilizando o RadAc. Podendo verificar a efetividade do modelo de controle de acesso, assim como testes relativos ao cálculo de risco e seus fatores. Como a quantidade de fatores associados ao risco é grande, faz-se necessário uma maneira para distribuir os pesos para os fatores não conhecidos. Existem diversas maneiras, mas, por questão de tempo, não foi possível implementar.

A principal contribuição deste trabalho está no fomento ao RadAc e sua evolução, pois até o momento um número pequeno de implementações surgiram no meio acadêmico, e dentre estas, poucas possuem flexibilidade e possibilidade de extendes para outros cenários.

Durante este trabalho, foram encontradas possibilidades para desenvolver trabalhos futuros. Dentre estes, podemos citar:

- Estudo da necessidade operacional e a definição de uma maneira de calculá-la de forma quantitativa;
- A utilização de métricas e fatores adequadas conforme o escopo da aplicação;
- A definição de fatores em tempo de execução.
- Utilizar técnicas de inteligência artificial para melhorar os fatores;
- Melhorar o desempenho;
- Realização de testes pertinentes ao cálculo do risco, visando refinar ainda mais o cálculo.

REFERENCES

[SANTOS, D. R.; WESTPHALL, C.M.; WESTPHALL, C.B.] Risk-based Dynamic Access Control for a Highly Scalable Cloud Federation". In: SECURWARE 2013 - The Seventh International Conference on Emerging Security Information, Systems and Technologies, 7th edition, Barcelona. Proceedings IARIA: XPS Press, 2013.

[MCGRAW, R.] Risk Adaptive Access Control. 2009. National Security Agency.

[OASIS] About Us. Disponível em: <https://www.oasis-open.org/org>. Acessado em 12 maio 2011.

[BRITTON, D.; BROWN, I.] A security risk measurement for the RAdACmodel. [S.l.: s.n.], 2007.

[SARIPALLI, P.; WALTERS, B.] QUIRC: A Quantitative Impact and Risk Assessment Framework for Cloud Security. In: 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD), 3 rd edition, USA. Proceedings... IEEE: IEEE Press, 2010. pp. 280-288. doi: 10.1109/CLOUD.2010.22.