Um Modelo Pragmático de Separação de Responsabilidades para o Controle de Acesso Baseado em Papéis

Bruno C. B. Figueiredo[1, 2], Gustavo H. M. B. Motta[1]

[1]Programa de Pós-Graduação em Informática – Universidade Federal da Paraíba
58059-900 João Pessoa – PB

[2]Unimed João Pessoa – Marechal Deodoro, 420
58040-140 João Pessoa – PB

bruno@unimedjp.com.br, gustavo@di.ufpb.br

Abstract. *The separation of duties (SD) is a security principle accepted in the appliance of policies for reduction of conflict of interests. This work proposes a pragmatic model of SD (Pragma SD) for the role based access control model (RBAC), which concerns about actual situations on the regular work of companies. In Pragma SD, the association between roles and users and the role hierarchy are orthogonal to the SD policies. So, the SD doesn't affect the administration of the relations between users, roles and permissions, like is noted in the SD of the RBAC model. By this way, a user can have roles where there are conflicts of interests, just being prohibited to execute the tasks where the conflicts exist.*

Resumo. *A separação de responsabilidades (SR) é um princípio de segurança aceito na aplicação de políticas de redução de conflitos de interesses. Este trabalho propõe um modelo pragmático de SR (Pragma SR) para o padrão de controle de acesso baseado em papéis (CABP), que leva em conta questões práticas, relativas ao funcionamento das organizações. No Pragma SR, a associação entre usuários e papéis e a estruturação de papéis em hierarquias são ortogonais às políticas de SR. Portanto, a SR não afeta a administração das relações entre usuários, papéis e permissões, como na SR do padrão de CABP. Assim, um mesmo usuário pode ter papéis onde haja conflitos de interesses, sendo apenas proibido executar as tarefas que levem a tais conflitos.*

1 Introdução

A redução das situações de conflitos de interesses é um importante problema das organizações contemporâneas. O crescimento vertiginoso do uso de sistemas de informação nos últimos 20 anos vem permitindo a excessiva concentração de poderes em usuários individuais. Isso pode levar a situações de conflitos de interesses, nas quais uma mesma pessoa tem o poder de auditar os próprios atos.

A separação de responsabilidades (SR) visa assegurar que fraudes ou danos acidentais não ocorram como conseqüência da demasiada concentração de poder numa única pessoa, mas eventualmente como resultado de conluios. Embora seja um princípio de segurança tradicionalmente aplicado em organizações comerciais, industriais e de governo para combater o problema acima (Clark e Wilson, 1987; Brewer e Nash, 1989), ela não tem sido utilizada de forma efetiva para controlar o acesso através de sistemas de informação, em especial a dados financeiros (IBM Corporation, 2004).

Há uma forte demanda das organizações para utilização de tecnologias de controle de acesso que suportem políticas de SR de forma eficaz e prática, particularmente

nos EUA, motivada pelo *Sarbanes-Oxley Act of* 2002 (EUA, 2002)[1]. Segundo o *National Institute for Standards and Technology* (NIST, 2005), o padrão de controle de acesso baseado em papéis (ANSI/INCITS, 2004) é uma tecnologia chave para atender essa lei, visto que foi projetado para suprir as necessidades de controle de acesso de ambientes corporativos, incluindo aquelas relativas à separação de responsabilidades.

O controle de acesso baseado em papéis (CABP) regula o acesso dos usuários para executar operações em objetos (aplicações) com base nos papéis que eles exercem numa organização. Os papéis denotam funções que descrevem a autoridade e a responsabilidade concedidas aos usuários (Ferraiolo *et al.* 2003). O princípio da separação de responsabilidades é suportado no CABP através do particionamento compulsório da responsabilidade e da autoridade para realizar tarefas envolvendo conflitos de interesses por vários papéis mutuamente exclusivos, podendo a SR ser estática ou dinâmica. A inclusão desse modelo de SR no padrão de CABP baseou-se nos seguintes critérios (Ferraiolo *et al.* 2003): as características do modelo deveriam estar bem compreendidas e ser bem representadas na literatura sobre o CABP; e o modelo deveria ser viável, no sentido de que deveria haver pelo menos um exemplo de implementação comercial ou de referência. Observamos entretanto que, embora conceitualmente simples, o modelo de SR do CABP é pouco prático, tendo sua aplicação efetiva dificultada por não considerar a realidade de funcionamento das organizações.

Por exemplo, na SR estática (SRE), se um indivíduo possui o papel de *Comprador*, ele não pode ter o papel de *Auditor de Compras* e vice-versa, de modo a evitar que ele possa auditar os próprios atos. É uma solução eficaz, porém sua rigidez a torna impraticável em organizações de pequeno ou até mesmo de médio porte. Ter-se-ia que ter pelo menos dois funcionários exclusivos: um comprador e um outro, auditor. Ora, em geral, o volume de trabalho de um comprador é significativamente maior que o de um auditor. Somente se justificaria um profissional dedicado exclusivamente às tarefas de auditagem em organizações com um grande setor de compras. Agora, no segmento das organizações de planos de saúde, mesmo as de maior porte, é fato que um mesmo indivíduo atue como médico solicitante ou prestador de serviços e também exerça a função de médico auditor, sendo-lhe proibido apenas a auditagem dos próprios atos.

Já a SR dinâmica (SRD) é mais flexível que a SRE, pois permite que um mesmo indivíduo esteja associado a dois ou mais papéis conflitantes. O que se impede é que o indivíduo tenha os privilégios desses papéis simultaneamente. O problema é que a SRD só é eficaz em situações nas quais os conflitos de interesses emergem pelo exercício simultâneo privilégios, mas é ineficaz para os casos de conflitos motivados pelo exercício sucessivo de privilégios. Ou seja, a SRD não é uma alternativa à SRE e portanto, não resolve o problema visto anteriormente. Em um dado momento, um indivíduo exercendo apenas o papel de *Comprador* solicita uma compra (fraudulenta) e, em outro momento, atuando apenas no papel de *Auditor de Compras*, valida a solicitação. Em nenhum momento os poderes dos papéis *Comprador* e *Auditor de Compras* são exercidos simultaneamente, graças à SRD, mas ainda assim a fraude pôde ser cometida.

Contrapondo as duas formas de separação de responsabilidades no CABP, podemos afirmar que a SRE é eficaz para reduzir as situações de conflitos de interesses, porém sua rigidez a distancia da realidade das organizações, tornando-a pouco útil na prática. Por outro lado, a flexibilidade da SRD não é verdadeiramente uma alternativa à

[1] Lei que estabelece normas para melhorar a acurácia e a confiabilidade das informações para os investidores, determinando um estrito controle das atividades internas às corporações, dentre outras prescrições.

rigidez da SRE, pois não tem o mesmo poder desta. De fato, as situações nas quais a SRD é aplicável são limitadas, quando comparada com a SRE, como no caso dos conflitos de interesses entre os papéis de operador e supervisor de caixa, descrito por Ferraiolo e colaboradores (2003) e Motta e Furuie (2004).

O objetivo deste trabalho é propor um modelo pragmático para separação de responsabilidades (PRAGMA SR) no padrão de controle de acesso baseado em papéis. Pragmático porque considera questões de ordem prática, relativas ao funcionamento das organizações. O PRAGMA SR permite que um mesmo usuário seja associado e ative papéis com conflitos de interesses. Apenas é proibido que esse usuário execute nas aplicações as tarefas sensíveis que levam a tais conflitos. Isso é possível porque no PRAGMA SR as associações entre usuários, papéis e permissões e a estruturação de papéis em hierarquias são ortogonais à configuração de políticas de SR. Ou seja, a definição de políticas de SR não afeta a administração das relações entre usuários, papéis e permissões, como na SR presente no modelo de CABP. Desse modo, o PRAGMA SR combina a flexibilidade da SR dinâmica do CABP com a robustez da SR estática.

O restante do trabalho está organizado da seguinte forma. A seção 2 introduz o modelo de referência para o CABP do NIST, que é estendido no nosso modelo, descrito na seção 3. A seção 4 apresenta situações práticas de aplicação do PRAGMA SR, onde se observa de forma mais clara as suas características. A seção 5 discute as contribuições do PRAGMA SR frente à separação de responsabilidades do padrão de CABP, bem como frente a outros trabalhos relacionados. Por fim, a seção 6 traz a conclusão do trabalho.

2 Modelo de Referência do Controle de Acesso Baseado em Papéis

O modelo de referência do padrão de CABP é um instrumento conceitual abstrato que trata das relações, das restrições e das funções administrativas entre entidades constantes do modelo, como usuários, papéis, sessões e autorizações (Figura 1) (Ferraiolo *et al*, 2003). Está organizado em três partes: o *CABP Básico*, *CABP Hierárquico* e o *CABP Restrito* (especifica as restrições de SR estática e dinâmica). Ainda apresentamos a especificação da função de autorização de acesso do CABP. Utilizaremos uma notação baseada naquela usada por Ferraiolo *et al.* (2001)[2] para a descrição formal dos modelos CABP e PRAGMA SR.

2.1 *CABP Básico*

O *CABP Básico* estabelece os requisitos essenciais de um modelo de CABP. Especifica as relações usuário-papel (UA) e papel autorização (PA) (Figura 1), define funções básicas e introduz o conceito de sessão, onde um usuário poderá ativar ou desativar um ou mais papéis. Segue abaixo a especificação das entidades e de seus relacionamentos.

- *Roles*, *Users*, *OBS* e *OPS* são, respectivamente, conjuntos de papéis, de usuários (humanos), de objetos protegidos e de operações associadas aos objetos protegidos. *Grosso modo*, os objetos representam aplicações que o usuário utiliza no desempenho de suas atividades. Em geral, operações representam a funcionalidade da aplicação disponível para os usuários;
- $PRMS \subseteq 2^{(OPS \times OBS)}$. Conjunto de permissões. Uma permissão (ou autorização) é um consentimento para executar uma operação num objeto de uma aplicação;

[2] Para melhor entendimento da notação empregada temos que: 2^c = *powerset* de C, ou seja, o conjunto de todos os subconjuntos de C.

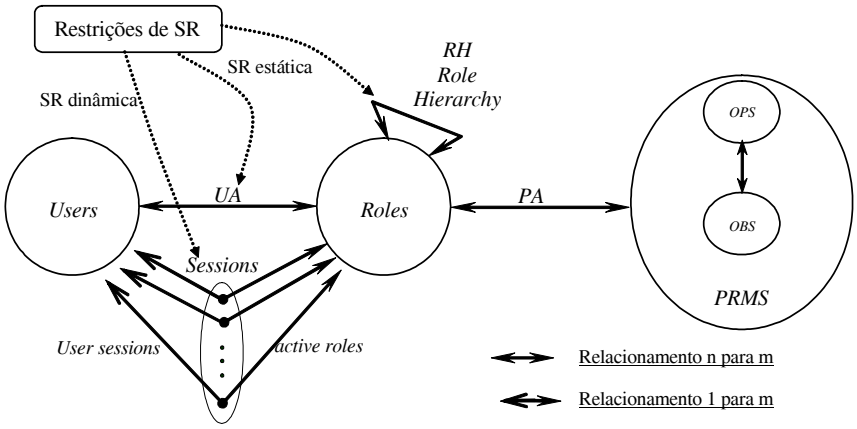

Figura 1 – Esquema do modelo de referência do padrão de controle de acesso baseado em papéis

- $UA \subseteq Users \times Roles$. Conjunto estabelecendo as associações muitos-para-muitos entre usuários e papéis;
- $PA \subseteq Roles \times PRMS$. Conjunto estabelecendo as associações muitos-para-muitos entre papéis e permissões;
- $Permissões\_associadas(r: Roles) \rightarrow 2^{PRMS}$. Mapeamento de um papel r para o conjunto de permissões associadas. Formalmente, $Permissões\_associadas(r) = \{p \in PRMS \mid (r, p) \in PA\}$;
- *Sessions*. Conjunto de sessões, onde cada uma representa o momento que o usuário conecta-se a um sistema e tem seu acesso controlado pelo CABP;
- $Usuário\_da\_sessão(s: Sessions) \rightarrow Users$. Mapeia a sessão s para o usuário que a iniciou;
- $Papéis\_da\_sessão(s: Sessions) \rightarrow 2^{Roles}$. Mapeia a sessão s para o conjunto dos papéis ativados na sessão por um usuário, que é um subconjunto dos papéis que o usuário tem associados. Num dado momento, o usuário só tem disponíveis as permissões associadas aos papéis que estão ativados. Formalmente, $Papéis\_da\_sessão\ (s) \subseteq \{r \in Roles \mid (Usuário\_da\_sessão(s), r) \in UA\}$.

2.2 *CABP Hierárquico*

Incorpora ao *CABP básico* requisitos para permitir a hierarquia de papéis (Figura 2). Matematicamente, uma hierarquia é uma ordem parcial que define uma relação de responsabilidades entre papéis. Em geral, papéis com maior responsabilidade, mais específicos (e.g., papel *Comprador*), adquirem as autorizações de papéis com menor responsabilidade, mais genéricos (e.g., papel *Administrador*). Logo, um comprador tem pelo menos os mesmos poderes que um administrador. Por outro lado, papéis com menor responsabilidade incorporam os usuários dos papéis com maior responsabilidade. Por exemplo, todo comprador é um administrador. Segue a definição da hierarquia de papéis (relação *RH* na Figura 1).

- $RH \subseteq Roles \times Roles$. Relação de ordem parcial sobre o conjunto de papéis, denominada de herança, indicada por $\succeq$, no qual, $p_1 \succeq p_2$ somente se todas as permissões de p_2 forem também permissões de p_1, e todos os usuários de p_1 forem

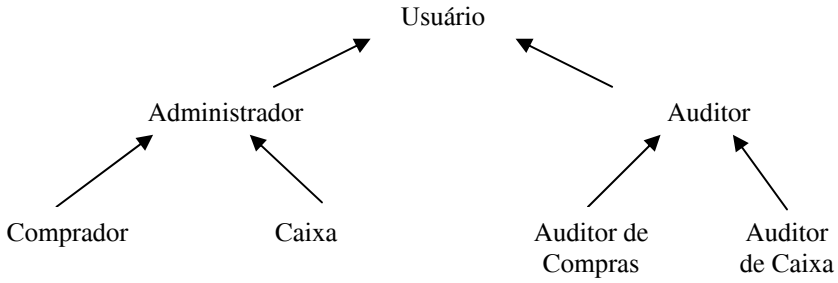

Figura 2 – Ilustração de uma hierarquia de papéis

também usuários de p_2. Formalmente, $p_1 \succeq p_2 \Rightarrow Permissões\_associadas\_rh(p_2) \subseteq Permissões\_associadas\_rh(p_1) \wedge Usuários\_associados\_rh(p_1) \subseteq Usuários\_associados\_rh(p_2)$;

- $Permissões\_associadas\_rh(r: Roles) \rightarrow 2^{PRMS}$. Mapeamento de um papel r para um conjunto de permissões direta ou indiretamente associadas a esse papel. Formalmente, $Permissões\_associadas\_rh(r) = \{p \in PRMS \mid r \succeq r' \wedge (r', p) \in PA\}$;
- $Usuários\_associados\_rh(r: Roles) \rightarrow 2^{Users}$. Mapeamento de um papel para um conjunto de usuários associados ao mesmo. Formalmente $Usuários\_associados\_rh(r) = \{u \in Users \mid r' \succeq r \wedge (u, r') \in UA\}$.

2.3 *CABP Restrito*

O *CABP Restrito* estende o modelo hierárquico para tratar a questão dos conflitos de interesses. Inclui relações de SR para impor restrições que ajudem a evitar que um usuário seja autorizado a executar operações associadas a papéis com conflitos de interesses. Nele, a SR pode ser estática ou dinâmica.

2.3.1 Separação de Responsabilidades Estática

Atua na relação usuário-papel (*UA*) e na hierarquia de papéis (*RH*) (Figura 1).

- $SRE \subseteq (2^{Roles} \times \mathbb{N})$. Conjunto de pares (pc, n), onde pc é um conjunto de papéis conflitantes e n é um número natural ≥ 2. Estabelece-se a propriedade de que nenhum usuário poderá associar-se a n ou mais papéis de pc, para todo $(pc, n) \in SRE$. Formalmente, $\forall (pc, n) \in SRE, \forall t \subseteq pc$:

$$|t| \geq n \Rightarrow \bigcap_{p \in t} Usuários\_associados\_rh(p) = \varnothing.$$

Tomemos como exemplo o conflito entre os papéis *Comprador* e *Auditor de Compras*. Instanciando essa situação para SR estática, temos: ({*Comprador*, *Auditor de Compras*}, 2) ∈ *SRE*, havendo as seguintes autorizações na relação *PA*: (*Comprador*, (*solicitaCompra*, *SI*)) e (*Auditor de Compras*, (*validaCompra*, *SI*)). *SI* representa um sistema de informação de uma organização. De acordo com a propriedade estabelecida para SRE, um mesmo usuário associado ao papel *Comprador* não poderá estar associado ao papel *Auditor de Compras* e vice-versa. Ou seja, a intersecção dos usuários de ambos os papéis deve ser o conjunto vazio. No exemplo anterior, um mesmo indivíduo não poderia, como comprador, solicitar compras fraudulentas e, como auditor de compras, considerá-las válidas. Note-se que a função *Usuários_associados_rh* leva em con-

sideração a hierarquia de papéis. Havendo pelo menos um usuário distinto associado aos papéis *Comprador* e *Auditor de Compras*, não seria possível criar um papel descendente de ambos, por exemplo, o papel *Comprador Auditor*, pois isto implicaria em aqueles papéis terem usuários em comum, violando a propriedade da SRE.

2.3.2 Separação de Responsabilidades Dinâmica

A SR dinâmica atua durante a ativação de papéis numa sessão de um usuário (Figura 1).

- $SRD \subseteq (2^{Roles} \times \mathbb{N})$. Conjunto de pares (pc, n), onde pc é um conjunto de papéis conflitantes e n é um número natural ≥ 2. Estabelece-se a propriedade de que nenhum usuário poderá ativar n ou mais papéis de pc, para todo $(pc, n) \in SRD$. Formalmente, $\forall\ (pc, n) \in SRD, \forall\ spc \in 2^{Roles}, \forall\ s \in sessions: |pc| \geq n \wedge spc \subseteq pc \wedge spc \subseteq papéis\_da\_sessão(s) \Rightarrow |spc| < n$.

Um usuário, em um dado momento, tem o poder concedido pelas permissões associadas aos papéis ativos de suas sessões. Logo, restringir a ativação simultânea de papéis pela relação *SRD* e pela propriedade anterior resulta na limitação dos poderes do usuário. Em geral, ele não terá, simultaneamente, a combinação dos poderes dos papéis a que está associado. Para ilustrar, tomemos o exemplo do comprador e do auditor de compras. Nessa instanciação, temos o par ({*Comprador*, *Auditor de Compras*}, 2) ∈ *SRD*, sendo as seguintes permissões pertencentes a *PA*: (*Comprador*, (*solicitaCompra*, *SI*)) e (*Auditor de Compras*, (*validaCompra*, *SI*)). Aqui, o usuário com o papel *Comprador* ativo poderá solicitar a compra de produtos, mas terá de encerrar sua sessão para poder ativar o papel *Auditor de Compras*. Para a SRD ser efetiva, quando do encerramento da sessão, todas as compras solicitadas e não validadas terão de ser canceladas, para impedir que o comprador as valide (as próprias solicitações) após a ativação do papel *Auditor de Compras*. Com o papel *Auditor de Compras*, ele poderá validar as solicitações de outros usuários que estejam com o papel *Comprador* ativado, no entanto, não poderá fazer solicitações.

Nota-se claramente a maior flexibilidade da SR dinâmica em contraposição à SR estática. Nessa, o controle desejado se dá pela impossibilidade de se atribuir a um mesmo usuário os papéis *Comprador* e *Auditor de Compras*, que têm poderes que podem levar à conflitos de interesses. Já na SR dinâmica, há a possibilidade da atribuição a um mesmo usuário de papéis conflituosos, porém com a limitação de poderes pela restrição da ativação simultânea de papéis com conflitos de interesses pelo usuário.

2.4 Autorização de Acesso

O modelo de referência do CABP especifica uma função chamada *Check_Access*, que retorna um booleano indicando se um usuário, mediante sua sessão, está ou não autorizado para executar uma operação de um objeto. Um usuário só poderá executar alguma operação se for autorizado de acordo com o especificado nessa função. Formalmente,

$$Check\_access\ (s: Sessions; op: OPS; ob: OBS) \rightarrow Boolean.$$
$$Check\_access\ (s, op, ob) \Leftrightarrow (\exists\ r \in Roles : r \in Papéis\_da\_sessão(s) \wedge (r, (op, ob)) \in PA)$$

A função *Check_access* autoriza o acesso de um usuário para executar uma operação *op* sobre um objeto *ob* se e somente se existir um papel *r* ativo em sua sessão *s* e o par operação-objeto é uma permissão associada ao papel na relação *PA*. Observa-se que os papéis ativos na sessão se sujeitam às restrições da SR dinâmica (ver subseção 2.3).

3 O Modelo PRAGMA SR

O modelo PRAGMA SR é uma extensão do padrão de CABP (básico e hierárquico). Nele, a separação de responsabilidades não é imposta por restrições à associação entre usuários e papéis, como na SR estática do padrão de CABP, nem mediante restrições para ativação simultânea de papéis, como na SR dinâmica. O PRAGMA SR baseia-se no fato da SR ser inerentemente uma política de acesso orientada a aplicação, que deve ser projetada considerando-se a partição das tarefas que levem a conflitos de interesses ou a erros (Gligor *et al*., 1998). Com isso, as restrições de SR são estabelecidas no PRAGMA SR por meio de relações entre operações (e respectivos objetos) que possam levar a situações de conflitos de interesses (ou a erros) se estiverem disponíveis para uma mesma pessoa. Qualquer tentativa por parte de um usuário em executar uma operação que tenha relação de conflito de interesses com outra operação que ele disponha através de seus papéis não é autorizada.

Por exemplo, poderíamos estabelecer que as operações *solicitaCompra* e *validaCompra* são operações conflitantes. Considerando que um mesmo usuário estivesse associado aos papéis *Comprador* e *Auditor de Compras*, como foram estabelecidos conflitos entre essas operações, as tentativas do usuário em executar pelo menos uma delas não seriam autorizadas. Ter ambos os papéis associados a uma mesma pessoa resultaria em ela ter menos poderes que a combinação dos poderes desses papéis associados a usuários distintos. No PRAGMA SR, também podemos definir a SR levando em conta o histórico das operações realizadas pelo usuário. Assim, um usuário com ambos os papéis poderia autorizar uma compra, desde que não fosse para uma solicitação que ele tenha efetuado.

3.1 Descrição Formal

Por se tratar de uma extensão ao modelo de referência do CABP, vamos considerar disponíveis no PRAGMA SR as definições presentes nas subseções 2.1 e 2.2. Nas definições a seguir, apresentamos apenas os acréscimos ou modificações ao modelo original.

- $OPC \subseteq \{(ps, h, n) \mid ps \subseteq (OPS \times OBS) \wedge h \in Boolean \wedge n \in \mathbb{N} \wedge n \geq 2\}$. *OPC* é o conjunto que relaciona as operações (e respectivos objetos) que podem levar a situações de conflitos de interesses. Cada elemento de *OPC* é composto de três valores: o primeiro é um subconjunto de $OPS \times OBS$, indicando as operações que têm restrições de separação de responsabilidade; o segundo é um valor booleano que indica se a SR levará em consideração o histórico das ações dos usuários; o terceiro valor é um número natural n, com $(n - 1)$ indicando quantas operações pertencentes a *ps* um usuário poderá executar sem violar a SR;
- *ID* é o conjunto itens de dados sensíveis acessados pelos usuários através das operações executadas a partir de um objeto (aplicação). Tipicamente, corresponde a dados de auditoria;
- $HA \subseteq Users \times (OPS \times OBS) \times ID$. *HA* é o conjunto com o histórico dos acessos de execução das operações (e respectivos objetos) realizadas pelos usuários sobre itens de dados;
- *Usuário_tem_operação_conflitante*(*u*: *Users*; *op*: *OPS*; *ob*: *OBS*; *id*: *ID*) → *Boolean*. Função que indica se o usuário *u* tem restrição de SR para executar a operação *op* do objeto *ob* sobre o item de dado *id*. Formalmente,

Usuário_tem_operação_conflitante $(u, op, ob, id) \Leftrightarrow$
$(\exists\, (ps, h, n) \in OPC \mid (op, ob) \in ps$:
$\neg h \Rightarrow |Permissões\_usuário(u) \cap ps| \geq n$
$h \Rightarrow |(Operações\_executadas\_usuário(u, id) \cup \{(op, ob)\}) \cap ps| \geq n)$

O usuário u terá restrições de SR para executar a operação op do objeto ob sobre o item de dado id se e somente se houver algum conflito para o par (op, ob) em OPC e pelo menos uma das seguintes condições valham: (a) o histórico de operações executadas pelo usuário é ignorado ($\neg h$) e ele tem associada uma quantidade de operações conflitantes com op superior ou igual a n; (b) o histórico de operações é considerado (h) e as operações de objetos já executadas pelo usuário u sobre o item de dado id, considerando a operação que ele tenta executar, conflita com uma quantidade de operações superior ou igual a n.

- *Permissões_usuário*(u: *Users*) $\rightarrow 2^{PRMS}$. Função que devolve o conjunto de todas as permissões associadas indiretamente a um usuário u através de seus papéis, considerando a hierarquia de papéis. Formalmente,

  Permissões_usuário $(u) = \{p \in PRMS \mid p \in Permissões\_associadas\_rh(r) \wedge (u, r) \in UA\}$

- *Operações_executadas_usuário*(u: *Users*; id: *ID*) $\rightarrow 2^{(OPS \times OBS)}$. Função que devolve o conjunto de todas operações executadas pelo usuário u sobre um item de dado id, disponíveis no histórico de operações executadas. Formalmente,

  Operações_executadas_usuário $(u, id) = \{(op, ob) \in OPS \times OBS \mid (u, (op, ob), id) \in HA\}$

Modificamos a função *Check_Access* do modelo de referência do padrão de CABP para incluir um restrição adicional: o usuário da sessão somente poderá executar uma operação sobre um item de dado se ela não tem restrições de separação de responsabilidade, ou seja, não tem conflitos com outras operações. Formalmente,

Check_access(s: *Sessions*; op: *OPS*; ob: *OBS*; id: *ID*) $\rightarrow$ *Boolean*.
Check_access $(s, op, ob, id) \Leftrightarrow$
$(\exists\, r \in Roles : r \in Papéis\_da\_sessão(s) \wedge (r, (op, ob)) \in PA \wedge$
$\neg Usuário\_tem\_operação\_conflitante\ (Usuário\_da\_sessão(s), op, ob, id))$

A versão da função *Check_access* do PRAGMA SR autoriza o acesso de um usuário para executar uma operação op de um objeto ob sobre um item de dado id se e somente se existir um papel r ativo em sua sessão s e o par operação-objeto é uma permissão do papel na relação PA e não há conflitos de SR para executar esse par.

4. Aplicação do modelo PRAGMA SR

Aplicaremos agora o PRAGMA SR a uma situação prática para melhor ilustrar suas características. Consideramos uma organização onde se tem a figura do comprador e do auditor de compras. Ao comprador cabe gerenciar (criar, alterar e excluir) e efetuar solicitações de compras. Já ao auditor cabe validar as solicitações, autorizando as compras. O sistema de informação (*SI*) da organização só confirma compras de solicitações validadas. Caso uma solicitação de compra validada seja alterada, ela deverá ser revalidada para que a compra seja efetuada. É intuitivo que as funções de gerência e de validação de solicitações de compras possam levar a situações de conflitos de interesses, se reali-

zadas por uma mesma pessoa. A solução é impor uma política de SR para evitar que uma mesma pessoa possa gerenciar e validar solicitações de compras.

Em nosso problema, consideramos a existência dos papéis *Comprador* e *Auditor de Compras*, havendo as seguintes autorizações na relação *PA*: (*Comprador*, (*gerenciaSolicitaçãoCompra*, *SI*)), (*Comprador*, (*lêSolicitaçãoCompra*, *SI*)) (*Comprador*, (*efetuaCompra*, *SI*)), (*Auditor de Compras*, (*validaSolicitaçãoCompra*, *SI*)), (*Auditor de Compras*, (*lêSolicitaçãoCompra*, *SI*)). A solução para o problema do conflito de interesses no PRAGMA SR pode levar em conta (ou não) o histórico das ações do usuário.

Para a solução sem histórico, basta incluir na relação *OPC* a tripla a seguir, relacionando as operações conflitantes, em nosso caso, as operações *validaSolicitaçãoCompra* e *gerenciaSolicitaçãoCompra*:

({(*validaSolicitaçãoCompra*, *SI*), (*gerenciaSolicitaçãoCompra*, *SI*)}, false, 2)

Supomos agora que um mesmo usuário *u* esteja associado aos papéis *Comprador* e *Auditor de Compras*. Quando ele tenta obter uma autorização com *Check_access* para executar a operação *validaSolicitaçãoCompra*, o acesso é negado, pois a função *Usuário_tem_operação_conflitante* retorna o valor true. Isso porque existe uma tripla (*ps*, *h*, *n*) em *OPC*, na qual

(*validaSolicitaçãoCompra*, *SI*) ∈ *ps*. Tendo

Permissões_usuário(*u*) = {(*validaSolicitaçãoCompra*, *SI*),
(*gerenciaSolicitaçãoCompra*, *SI*), (*efetuaCompra*, *SI*), (*lêSolicitaçãoCompra*, *SI*)},

sabemos que

|*Permissões_usuário*(*u*) ∩ {(*gerenciaSolicitaçãoCompra*, *SI*),
(*validaSolicitaçãoCompra*, *SI*)}| ≥ 2.

Então, podemos concluir que o usuário **tem** permissões conflitantes com a operação *validaSolicitaçãoCompra*. O mesmo ocorre com a operação *gerenciaSolicitaçãoCompra*, ficando o usuário proibido de executá-la. Agora, se o usuário fosse executar a operação *efetuaCompra*, a sua execução seria autorizada, pois não existe uma tripla (*ps*, *h*, *n*) em *OPC*, na qual

(*efetuaCompra*, *SI*) ∈ *ps*.

Isso é condizente com a SR, pois o usuário só poderá efetuar compras cujas solicitações foram validadas e gerenciadas por outras pessoas, mas não por ele próprio, visto que não tem poderes para executar nenhuma das duas operações.

Caso o usuário estivesse associado apenas ao papel *Comprador*, ele poderia executar normalmente a operação *gerenciaSolicitaçãoCompra*, pois a função *Usuário_tem_operação_conflitante* retornaria o valor false. Isso porque existe uma tripla (*ps*, *h*, *n*) em *OPC*, na qual

(*gerenciaSolicitaçãoCompra*, *SI*) ∈ *ps*. Tendo agora

Permissões_usuário(*u*) = {(*gerenciaSolicitaçãoCompra*, *SI*),
(*efetuaCompra*, *SI*), (*lêSolicitaçãoCompra*, *SI*)},

sabemos que

|*Permissões_usuário*(*u*) ∩ {(*validaSolicitaçãoCompra*, *SI*),
(*gerenciaSolicitaçãoCompra*, *SI*)}| < 2.

Então, podemos concluir que o usuário **não** teria permissões conflitantes com a operação *gerenciaSolicitaçãoCompra*.

Para a solução com histórico, basta alterar a tripla em *OPC*, do exemplo anterior, para a seguinte configuração:

({(*validaSolicitaçãoCompra*, *SI*), (*gerenciaSolicitaçãoCompra*, *SI*)}, true, 2)

Supomos novamente que um mesmo usuário *u* esteja associado aos papéis *Comprador* e *Auditor de Compras*. Esse usuário ainda não registra em seu histórico a execução das operações *validaSolicitaçãoCompra* ou *gerenciaSolicitaçãoCompra* para nenhum item de dado. Quando ele tenta obter uma autorização com *Check_access* para executar a operação *gerenciaSolicitaçãoCompra* sobre o item de dado *id*, o acesso é permitido, pois a função *Usuário_tem_operação_conflitante* retorna o valor false. Isso porque existe uma tripla (*ps*, *h*, *n*) em *OPC*, na qual

(*gerenciaSolicitaçãoCompra*, *SI*) $\in$ *ps*. Tendo

Operações_executadas_usuário(*u*, *id*) = $\emptyset$, sabemos que

$$|(\textit{Operações\_executadas\_usuário}(u, id) \cup \{(\textit{gerenciaSolicitaçãoCompra}, SI)\}) \cap \{(\textit{validaSolicitaçãoCompra}, SI), (\textit{gerenciaSolicitaçãoCompra}, SI)\}| < 2.$$

Então, podemos concluir que o usuário **não** tem permissões conflitantes com a operação *gerenciaSolicitaçãoCompra* sobre *id*. Agora, supomos que o usuário *u* tenta executar a operação *validaSolicitaçãoCompra* sobre o mesmo item de dado *id*. A execução da operação não é autorizada, pois existe uma tripla (*ps*, *h*, *n*) em *OPC*, na qual

(*validaSolicitaçãoCompra*, *SI*) $\in$ *ps*. Tendo agora

HA = {(*u*, (*gerenciaSolicitaçãoCompra*, *SI*), *id*)} e

Operações_executadas_usuário(*u*, *id*) = {(*gerenciaSolicitaçãoCompra*, *SI*)},

sabemos que

$$|(\textit{Operações\_executadas\_usuário}(u, id) \cup \{(\textit{validaSolicitaçãoCompra}, SI)\}) \cap \{(\textit{validaSolicitaçãoCompra}, SI), (\textit{gerenciaSolicitaçãoCompra}, SI)\}| \geq 2.$$

Então, podemos concluir que o usuário **tem** permissões conflitantes com a operação *validaSolicitaçãoCompra*. Nota-se que, inicialmente, quando o histórico do usuário era vazio, ele poderia executar tanto a operação *validaSolicitaçãoCompra* quanto a operação *gerenciaSolicitaçãoCompra*. Uma vez executada uma delas sobre um item de dado, a outra fica indisponível para este mesmo item de dado.

Supomos agora que o usuário *u* vai validar uma solicitação sobre um item de dado *id´*, distinto de *id*, que foi gerenciada por um outro usuário, *u´*. O usuário *u* é autorizado a validar a solicitação, pois a função *Usuário_tem_operação_conflitante* retorna o valor false. Isso porque existe uma tripla (*ps*, *h*, *n*) em *OPC*, na qual

(*validaSolicitaçãoCompra*, *SI*) $\in$ *ps*. Tendo agora

HA =
{(*u*, (*gerenciaSolicitaçãoCompra*, *SI*), *id*), (*u*, (*validaSolicitaçãoCompra*, *SI*), *id´´*),
(*u´*, (*gerenciaSolicitaçãoCompra*, *SI*), *id´*), (*u´*, (*validaSolicitaçãoCompra*, *SI*), *id*)} e

Operações_executadas_usuário(u, id') = ∅, sabemos que

$$|(\textit{Operações\_executadas\_usuário}(u, id') \cup \{(\textit{validaSolicitaçãoCompra}, SI)\}) \cap \{(\textit{validaSolicitaçãoCompra}, SI), (\textit{gerenciaSolicitaçãoCompra}, SI)\}| < 2.$$

Então, podemos concluir que o usuário **não** tem permissões conflitantes com a operação *validaSolicitaçãoCompra* sobre o item de dado *id´*.

5. Discussão

O modelo PRAGMA SR é discutido sob duas perspectivas. A primeira procura defender as suas contribuições frente à separação de responsabilidades estática e dinâmica do padrão de CABP. A segunda analisa as contribuições do PRAGMA SR frente a outros trabalhos relacionados.

5.1 O PRAGMA SR e o Padrão de CABP

A separação de responsabilidades proposta no PRAGMA SR combina a flexibilidade da SR dinâmica com a robustez da SR estática do padrão de CABP. É flexível porque não impõe restrições às associações entre usuários e papéis e entre papéis e autorizações, tampouco para a ativação de papéis. Isso facilita a administração do CABP nas organizações contemporâneas, caracterizadas pela grande variedade de funções exercidas por um usuário ao longo do tempo. Já a rigidez da SR estática para atribuição de usuários a papéis não condiz com a realidade de muitas empresas. Nos casos de empresas de pequeno ou mesmo de médio porte, por exemplo, ter-se-ia que contratar um funcionário exclusivo para auditoria e outro para ser comprador. Ora, certamente o volume de trabalho de um comprador é significativamente maior que o de um auditor. Pelo comprador passam as tarefas de cotar preços, negociar qualidade, prazos de pagamento e de entrega, receber e visitar fornecedores, entre outras. Porém o trabalho de auditagem, embora de grande importância, resume-se a verificar a legalidade do processo de compra, o que demanda um tempo muito menor. É fácil então concluir que, para se ter um profissional completamente dedicado às tarefas de auditagem numa empresa, a mesma teria de ter um grande setor de compras com, provavelmente, vários funcionários realizando as tarefas inerentes à função de comprador. Ou seja, a situação descrita acima só se aplicaria bem para grandes empresas onde o volume de compras justifique um profissional com a função de auditor de compras. Ainda assim, no setor de planos de saúde, mesmo os de grande porte, é comum que um médico solicitante também atue como médico auditor, sendo-lhe proibida apenas a auditoria das suas próprias solicitações.

Por outro lado, a SR no modelo PRAGMA SR é tratada de forma direta e prática, levando-se em conta apenas as operações que podem levar a situações de conflitos de interesses. A configuração do modelo é simples, bastando que se incluam tuplas no conjunto *OPC* relacionando operações que levam a conflitos. O controle para bloqueio das operações conflituosas é realizado no momento da solicitação de acesso. Ademais, o uso da opção de histórico de acessos permite impor a SR de uma forma ainda mais flexível, aliando eficácia com utilização eficiente dos recursos humanos disponíveis. Por exemplo, numa organização, compradores poderiam ser auditores de compras e vice-versa, porém, a SR com histórico impediria que um usuário auditasse as próprias compras.

O PRAGMA SR, por tratar a SR de forma ortogonal às relações do CABP, tem sua administração facilitada, possibilitando verificações de erros de atribuições de permissões indevidas através da verificação da configuração dos usuários, papéis e permissões com as restrições de separação de responsabilidades configuradas no PRAGMA SR.

Dessa forma, a detecção automática de possíveis erros de atribuição de permissões conflitantes torna-se viável, contribuindo para a segurança do mecanismo de controle de acesso.

A despeito das considerações anteriores, ainda é necessário efetuar a implementação do modelo para investigar a sua escalabilidade e a sua aplicabilidade prática, em particular, na SR com histórico. Em adição, é preciso avaliar o quão difícil seria aplicar o modelo numa grande organização, com variadas situações de conflitos de interesses, diferentes do caso de uso utilizado neste trabalho.

5.2 O PRAGMA SR e os Trabalhos Relacionados

O modelo PRAGMA SR traz a contribuição de separar a configuração e a administração das relações do CABP da configuração e da administração das relações das políticas de separação de responsabilidades, colocando-as em planos ortogonais. Por exemplo, no PRAGMA SR pode-se alterar a hierarquia de papéis de forma independente das políticas de SR eventualmente configuradas e vice-versa. Baseamo-nos no fato da SR ser inerentemente uma política de acesso orientada a aplicação, que deve ser projetada considerando-se a partição das tarefas que levem a conflitos de interesses ou a erros (Gligor *et al*., 1998). Em geral, a SR dos modelos de CABP propostos na literatura são dependentes das relações usuário-papel ou papel-permissão e da hierarquia de papéis.

Um dos primeiros trabalhos a investigar a SR com profundidade no CABP foi realizado por Kuhn (1997). O modelo proposto por Kuhn usa papéis mutuamente exclusivos para especificar políticas de SR. No trabalho são introduzidas as formas estática e dinâmica de separação de responsabilidades, que vieram a compor o padrão de CABP (ANSI/INCITS, 2004), já apresentadas na subseção 2.3 e discutidas na subseção 5.1.

Ahn e Sandhu (2000) especificam uma linguagem baseada em restrições (*Constraints*) –RCL 2000–, para o controle de acesso baseado em papéis. A especificação das políticas de SR, realizada por meio da escrita de *constraints*, dificulta seu uso na prática, dada a complexidade da linguagem e da necessidade de verificação de propriedades de SR. Ademais, não houve uma preocupação explícita sua aplicação prática, como no PRAGMA SR.

A forma como a separação de responsabilidades foi introduzida no PRAGMA SR faz com que o mesmo se diferencie de trabalhos como o de Oh e Park (2003), que propõe um modelo de CABP baseado em *tasks* e de *WorkFlows* (T_RBAC). Nele, as permissões são agrupadas em *tasks* e as mesmas são classificadas segundo suas características de acesso às informações. Nesse trabalho, são inseridas noções de controle de regra de negócio, adaptando o modelo de CABP à realidade operacional das empresas. Porém a implementação das regras de negócio (*Bussiness Rules*) se dá através de *Constraints*, sem que haja uma forma específica para especificação de políticas de separação de responsabilidades, ou seja, há uma interposição de funções onde a regra do negócio se confunde com a própria política de separação de responsabilidades, o que prejudica o trabalho nesse aspecto, já que não há a independência na implementação da política de separação de responsabilidades, como encontramos no PRAGMA SR.

Em Moon *et al*. (2004), encontramos *Constraints* que resolvem problemas de SRE do CABP quanto à atribuição indevida de permissões a papéis, esses problemas são causados normalmente pela herança de permissões entre papéis sênior e júnior. Não há definição, entretanto, de um novo modelo para tratar a questão da SR, como no PRAGMA SR. Na verdade o trabalho se limita a solucionar falhas existentes no modelo

RBAC96 (Sandhu *et al.*, 1996) através da inserção de *constraints*. Logo, não encontramos uma separação entre as políticas de SR e as relações do modelo de CABP, nem tampouco eficiência do uso dos recursos humanos, ambos encontrados no PRAGMA SR.

Encontramos uma proposta semelhante ao PRAGMA SR em Motta e Furuie (2004), que propõem um modelo de SR orientado a aplicação, contextual, onde também há uma verificação no momento da tentativa de uso de cada operação, onde se determina se a mesma pode ou não ser executada sobre um objeto. Porém, no modelo, o tratamento da SR depende diretamente dos papéis existentes, podendo haver a necessidade de reconfigurações na política de SR sempre que um novo papel é incluído. Logo, temos uma indesejada dependência dos papéis, o que não ocorre no PRAGMA SR.

Encontramos em Giogini *et al.* (2006) um estudo sobre a detecção de conflitos de interesses. Nele temos uma explanação de possíveis conflitos de interesses e ações que, se tomadas, poderão detectá-los, porém a SR para tratar tais conflitos é feita através de *constraints.* Ou seja, há uma dependência direta da aplicação quando do tratamento da SR, o que nos leva aos problemas já comentados e torna clara, mais uma vez, a contribuição do PRAGMA SR.

Vemos, através dessa análise, que o PRAGMA SR define formalmente o tratamento dos conflitos de interesses através de um modelo para especificação de SR entre permissões. Proporcionando uma maior flexibilidade, ortogonalidade e usabilidade ao modelo, como comentado na seção 5.1. Essas características não foram encontradas em outros trabalhos, que restringem, em geral, seu tratamento de SR à elaboração de *constraints* dependentes da configuração das relações do modelo de CABP. Nesses pontos, ressaltamos as contribuições de nosso trabalho.

6. Conclusão

Foi apresentado o PRAGMA SR, um modelo que estende as funcionalidades do padrão de CABP, tornado-o mais flexível e prático para especificação de políticas de SR. Permite a ativação simultânea de papéis com conflitos de interesses entre si, por um mesmo usuário, numa mesma sessão. É dada ênfase na questão da existência de conflitos de interesses nas permissões associadas a esses papéis, mas não há restrições quanto à administração de permissões conflitantes em papéis associados a um mesmo usuário, deixando-se bloqueio do uso dessas permissões para o momento do uso das mesmas.

Um ponto importante é a verificação do histórico de uso dos dados, permitindo que, caso um dado tenha sido acessado através de uma permissão, não possa mais ser acessado pelo mesmo usuário utilizando uma outra permissão que conflite com essa. Dessa forma, a SR pode ser realizada de forma mais precisa, evitando que restrições desnecessárias sejam impostas para os usuários.

Referências

Ahn, G. e Sandhu R., (2000) "Role-Based Authorization Constrains Specification", *ACM Transactions on Information and System Security*, vol. 3, n. 4, p. 207-226, November.

ANSI/INCITS 359-2004. (2004) *Information Technology: Role Based Access Control. InterNational Committee for Information Technology Standards*, 56 p. February.

Brewer, D. F. C. e Nash, M. J. (1989) "The Chinese Wall Security Policy". In: *IEEE*

Symposium on Security and Policy, **Proceedings**... p. 206-214.

Clark, D. e Wilson, D. (1987) "A Comparison of Commercial and Military Computer Security Policies". In: *IEEE Symposium on Security and Policy*, **Proceedings**... p. 184-194.

EUA. Public Law 107-204 (2002). *Sarbanes-Oxley Act of 2002*. Washignton, DC: The Library of Congress. Disponível em: <http://thomas.loc.gov/cgi-bin/bdquery/z?d107:HR03763:>. Acesso em: 23 mar. 2007.

Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn, D. R. e Chandramouli, R. (2001) "Proposed NIST Standard for Role-Based Access Control", *ACM Transactions on Information and System Security*, v. 4, n. 3, p. 224-274, August.

Ferraiolo, D. F., Kuhn, D. R. e Chandramouli, R. *Role-Based Access Control*. Boston: Artech House, 2003.

Giorgini, P., Massacci, F., Mylopoulos, J. e Zannone N. (2006) "Detecting Conflicts of Interest", In 14th *IEEE International Requirements Engineering Conference* (RE´06), **Proceedings…** p. 315 – 318.

Gligor, V. D., Gavrila, S. I. e Ferraiolo, D. (1998) "On the Formal Definition of Separation of Duty Policies and Their Composition", In *IEEE Symposium on Security and Privacy*, **Proceedings…** p. 172 – 183.

IBM Corporation. (2004) *Addressing the Key Implications of Sarbanes-Oxley*. September. Disponível em: <http://www.bizforum.org/whitepapers/ibm.htm>. Acesso em: 23 mar. 2007.

Kuhn, D. R. (1997) "Mutual Exclusion of Roles as a Means of Implementing Separation of Duty in Role-Based Access Control Systems", In: *Second ACM Workshop on Role-Based Access Control*, **Proceedings…** p. 23-30. .

Moon C., Park D, Park S. e Baik D. (2004) "Symmetric RBAC Model that Takes the Separation of Duty and Role Hierarchies into Consideration", *Computers & Security*, v. 23, p. 126-136.

Motta, G. H. M. B. e Furuie, S. S. (2004) "Separação de Responsabilidades Orientada a Aplicação no MACA: um Modelo de Autorização Contextual para o Controle de Acesso Baseado em Papéis", In: *VI Simpósio de Segurança em Informática*, **Anais**... 11 p.

NIST. *RBAC and Sarbanes-Oxley Compliance*. (2005) Sítio do NIST que disponibiliza informações sobre o padrão RBAC (Role-Based Access Control). Disponível em: < http://csrc.nist.gov/rbac/sarbanes-oxley-compliance.html>. Acesso em: 23 março. 2007.

Oh, S. e Park, S. (2003) "Task-Role-Based Access Control Model", *Information Systems*, v. 28, p. 533-562.

Sandhu, R. S., Coyne, E. J., Feinstein, H. L. e Youman, C. E. (1996) "Role-Based Access Control Models", *Computer*, p. 38-47, February.