Verificação de Conflitos na Gerência de Políticas de Controle de Acesso.

Paulo Ricardo B. Dutra Lima[1], Raul Ceretta Nunes[1], Gerson Antunes Soares[1], Roseclea Duarte Medina[1]

Universidade Federal de Santa Maria, PPGEP/DELC/CT
Laboratório de Tolerância a Falhas - GMicro
Santa Maria, Brasil, 97105-900

{paulor,gerson,ceretta,rose}@inf.ufsm.br

Resumo. *Em sistemas de controle de acesso a criação de políticas exige a verificação de conflitos. Este trabalho propõe algoritmos que gerenciam automaticamente, em tempo de criação das políticas, o controle de alguns tipos de conflitos, tais como conflitos de interesse. A idéia principal é de possibilitar que o processo de geração e edição políticas ocorra de maneira facilitada e sem conflitos.*

1. Introdução

A disponibilização de informações privadas em redes de computadores, tal como o que ocorre em sistemas ERP e de prontuário eletrônico de paciente, exige esforço para manter a privacidade dos usuários, a confidencialidade dos dados e a integridade da informação. Por isto a segurança costuma ser mantida com métodos fortes de controle de acesso e Políticas de Controle de Acesso (PCA) bem elaboradas. O enfoque deste trabalho é a área da saúde, mas os resultados podem ser aplicado em outras áreas.

Uma PCA define diretrizes de alto nível que determinam como o acesso é controlado e como as decisões de autorização de acesso são estabelecidas (Ferraiolo et al 2001). Em outras palavras, estabelecem como um objeto deve ser acessado em um meio. Deste modo, um sistema de informação pode prover maior segurança na medida em que satisfaz políticas de acesso aos seus dados. Por exemplo, se numa empresa os dados confidencias dos clientes forem regulados por uma política que determina que somente os gerentes podem ter acesso a estes dados e os demais funcionários não, ao garantir este controle de acesso pode-se dizer que o sistema está minimizando as chances de ocorrência de uma falha de segurança.

Na prática, uma PCA geralmente é definida por um administrador de redes ou uma pessoa que tenha um bom conhecimento técnico, pois as políticas são especificadas normalmente em linguagens de marcação como, por exemplo, XACML (Extensible Access Control Markup Language) (Oásis 2003), PONDER (Damianou 2001) e SPSL (Condell 2000), exigindo assim o conhecimento da linguagem.

No Middleware de Autenticação e Controle de Acesso - MACA (Motta 2003), utilizado para controlar o acesso ao PEP do Instituto do Coração do Hospital das Clínicas da Faculdade de Medicina da Universidade de São Paulo (INCOR), para gerar uma nova política existe um formulário de solicitação de acesso, que o usuário interessado deve preencher a mão, onde descreve-se os recursos que ele quer ter direito de utilizar. Esta requisição é então enviada para um administrador de redes, que posteriormente adiciona uma nova PCA que possibilita o acesso aquele usuário.

Neste fluxo pode-se identificar alguns problemas como: 1) o responsável por gerar a política não é o mesmo interessado nela, o que gera um atraso no processo de emissão das política; 2) o criador da política deve ser um especialista, para poder descrevê-la em linguagem técnica e para poder avaliar se a nova política não conflita com as já geradas.

Neste trabalho propõe-se algoritmos para detecção de conflitos em tempo de geração ou edição de PCAs, o que facilita a manipulação de políticas por pessoal não técnico, pois elimina a necessidade do criador de políticas gerenciar conflitos.

O artigo está estruturado como segue. A seção 2 descreve os trabalhos relacionados. A seção 3 relata sobre gerenciamento de políticas. A seção 4 apresenta os conflitos nas políticas. A seção 5 trata sobre os algoritmos para controle de conflitos e a seção 6 tece as considerações finais.

2. Trabalhos Relacionados

A ferramenta para gerência de políticas do Globus é uma aplicação Java onde um administrador de redes pode navegar entre os elementos das políticas selecionadas e especificar atributos presentes nas políticas como: objetos, usuários e ações (Lorch, Dennis e Shah 2003). Para manter a integridade, proteção e não repúdio, depois de gerar uma PCA a ferramenta utiliza uma assinatura digital antes do armazenamento e distribuição da política, possibilitando a criptografia da PCA gerada. A ferramenta utiliza a biblioteca Sun XACML (Sun 2007), que fornece suporte para a construção de PCAs e também para requisições e respostas de acesso, entretanto não realiza verificação de conflitos nas PCAs, o que é um importante fator em gerenciamento de políticas.

Na arquitetura LARGES-Seg (Ferreira, Santos e Júnior 2004), que também utiliza a linguagem XACML para expressar as políticas definidas por administradores de redes, a ferramenta de gerenciamento de PCAs trabalha com configuração de VPNs e restrições de horários. Na sua interface, são informados os seguintes campos para a geração/edição de PCAs: servidor de origem e destino, utilização ou não de criptografia, horário permitido de acesso, e se o usuário é do tipo cliente ou funcionário. Estes dados servem para fornecer segurança no estabelecido de VPNs, mas a verificação de conflitos nas PCAs também não é tratada.

Damianou (2001) propõe um framework para o gerenciamento de PCAs e uma linguagem de especificação de políticas. O framework possibilita a administração de recursos, pessoas e também conta com uma distribuição automatizada de políticas, além de possibilitar a gerência de domínios, políticas e perfis, sendo inclusive realizada a verificação de conflitos em PCAs. Porém, os conflitos verificados são os de Interesse e os por regras negativas, não sendo implementadas outras modalidades de conflitos como, por exemplo, o de redundância. Além disto, não considera-se o conhecimento do usuário, pois assume-se que na sua maioria são administradores de redes, que detém um bom conhecimento técnico podendo realizar algumas mudanças de forma manual caso algum conflito ocorra.

Observa-se que os trabalhos citados são destinados para administradores de redes, logo a interface destes sistemas é planejada para este grupo de usuários. Além disso, observa-se que a maioria dos trabalhos não executa a verificação de conflitos. Por outro lado, este trabalho busca características de simplicidade aliada a algoritmos que

realizam a verificação de conflitos de PCAs de forma automática, possibilitando a geração de políticas tanto positivas como negativas sem que ocorram conflitos.

3. Gerenciamento de Políticas

O modelo de gestão de redes com a utilização de políticas traduz as diretrizes das empresas (políticas de alto nível) para o mundo dos elementos de rede (políticas de baixo nível), permitindo que estas políticas sejam difundidas mais fáceis e rapidamente, sem a necessidade de gerência direta em equipamentos e arquivos, possibilitando desta forma, a reutilização de conhecimentos e processos. Após o entendimento da política global, o usuário responsável deverá utilizar a ferramenta de gerência para a definição e validação das políticas, que serão armazenadas em um repositório e aplicadas automaticamente aos equipamentos e arquivos (Ferreira, Santos e Júnior 2002).

A gerência baseada em PCA, tem tornado-se recentemente uma solução muito empregada e prometedora para controlar as redes das empresas e sistemas distribuídos. Tais sistemas são dirigidos pelas necessidades dos negócios, que requerem soluções de gerencia auto-adaptáveis e que mudam dinamicamente o comportamento do sistema (Damianou 2001).

O gerenciamento de PCA, envolve a verificação de conflitos que possam ser levantados quando uma política é gerada. O gerenciador de PCA deve verificar diversos tipos de conflitos que podem ocorrer tanto a nível de sistema operacional como em mecanismos de controle de acesso, com um número significativo de PCAs, os quais devem estar de acordo com a especificação realizada pelo administrador de redes.

4. Conflitos em Políticas

Um conflito em uma política pode aparecer quando múltiplos eventos ou condições conflitantes atuam (Shankar 2005), tornando-se difícil para o sistema decidir que ação executar em um determinado momento. Dursun e Orencik (2004) fundamentam que um provável conflito pode ocorrer pela sobreposição de sujeitos entre duas ou mais políticas. Logo caso não exista sujeitos em comum entre duas políticas não é provável que ocorra um determinado conflito.

Existem trabalhos que tratam conflitos relacionados a controle de acesso como, por exemplo, os embasados nos modelos do RBAC (Role Based Access Control) (Ferraiolo et al. 2001), e existem projetos os quais são direcionados a conflitos em sistemas distribuídos como em, (Moffet e Sloman 1993) e (Lupu e Sloman 1999).

4.1 Classificação de Conflitos

A classificação em conflitos de sistema operacional e de controle de acesso justifica-se pelo fato que o sistema proposto será dado enfoque nesses dois grupos de conflitos, visto a necessidade de implementação de algoritmos para controle de conflitos nessas duas modalidades.

A classificação de conflitos de SO é referenciada segundo (Moffet e Sloman 1993), em: conflitos de prioridades, deveres, interesses, múltiplos gerentes, auto - gerencia e por regras negativas. Sendo conflitos de redundância referenciado por (Charalambides et al. 2006), onde é realizada sua especificação teórica.

Conflitos em mecanismos de controle de acesso são fundamentados em dois tipos: Conflitos por regras negativas e conflitos de interesse. Estas modalidades de conflitos são propostas no modelo padrão RBAC proposto por NIST, e são aplicadas neste projeto adotando metodologias diferentes das propostas visando a melhor adequação no modelo como pode-se observar na seção abaixo.

5. Algoritmos Propostos

Nas seções 5.1 e 5.2 serão dados enfoques em conflitos e algoritmos propostos levando-se em conta mecanismos de controle de acesso. Na seção 5.3 o foco será conflitos em SO. Tanto conflitos em SO como em controle de acesso possuem várias modalidades e classificações, sendo que a escolha da implementação dos conflitos selecionados baseou-se em estudos de todas as modalidades de conflitos já propostas como em (Moffet e Sloman 1993).

5.1 Conflitos por Regras Negativas

Sistemas Gerenciadores de Políticas de Controle de Acesso dão suporte a utilização de regras positivas e negativas em PCAs, sendo que é possível regrar a utilização do sistema gerando políticas para negação de um usuário (DUA - Denial User Assignment) e para negação de um determinado perfil (DRA - Denial Role Assignment).

Com a finalidade de evitar que conflitos por PCAs com regras negativas venham a ocorrer, é proposto o Algoritmo de Políticas Negativas da figura 1, que além de impedir que uma política venha a ser gerada de forma conflitante, também indica que política está conflitando com a que está sendo gerada. No algoritmo P é a política sendo gerada e PR o repositório de políticas, o qual contém políticas positivas (PAs) e negativas (DUAs e DRAs).

```
Algoritmo1 Controle de Políticas Negativas
Entrada: política sendo gerada P, repositório de políticas PR contendo PAs, DUAs e DRAs
Saída: 1) Mensagem de conflito
1  For each PA ∈ PR
2        If (PA.R = P.R And PA.UN = P.UN And PA.T ∩ P.T ≠ Ø)And
3           (PA.US = P.US And PA.UN = P.UN And PA.T ∩ P.T ≠ Ø)
4        then  (P conflita com PA)
5        else Message  PR ← P
6        EndIf
7  EndFor
8  End Algoritmo1
```

Figura 1 – Conflitos Regras Negativas

Este algoritmo atua sempre que uma PCA for gerada tanto com regras positivas como negativas, ou seja, se uma política positiva for gerada o algoritmo verifica se há alguma política com regras negativas para um mesmo usuário ou perfil, caso afirmativo a política não é gerada.

O laço condicional nas linhas 2 e 3, realizam as seguintes comparações: se o perfil da PCA que está sendo gerada é o mesmo presente no repositório das políticas, referenciadas pelas propriedades P.R e PA.R respectivamente. A segunda comparação realizada é dos elementos: P.UN e PA.UN, sendo da mesma forma realizada a comparação, só que neste caso a propriedade verificada é a unidade (setor) da política que está sendo gerada, verificando assim se é a mesma presente no repositório das

PCAs, por fim é verificada se há intersecção temporal entre as duas políticas. Se estas condições forem validadas uma mensagem de conflito é apresentada apontando que política está conflitando. Se pelo menos uma comparação não for validada a política é gerada e enviada para o repositório das políticas.

A mesma verificação que é realizada para perfis é realizada para usuários, visto que em sistemas gerenciadores de PCA pode-se negar usuários e perfis, logo o algoritmo deve ser flexível o suficiente para suportar estas duas funcionalidades. Na linha 3 a propriedade US é referenciada com sendo o usuário do sistema.

5.2 Conflitos de Interesse

Conflitos de interesse em um sistema do controle de acesso, podem acontecer quando um usuário associa-se a um ou mais perfis conflitantes, ganhando assim permissões que podem comprometer a segurança do sistema. Um caminho para evitar este conflito é pela Separação de Deveres Estática (SSD) (Ferraiolo et al. 2001), que age no relacionamento entre usuários e perfis.

Em paralelo aos algoritmos, existe um arquivo de conflitos onde são armazenados os papéis e setores conflitantes, os quais são informados pelo usuário. Este arquivo é acionado sempre que uma nova PCA venha a ser gerada, com a finalidade de verificar se o perfil e setor da política que esta sendo gerada conflita com as definidas no arquivo de conflitos. Na Figura 2, pode-se observar o algoritmo que trata desta funcionalidade.

Algoritmo2 Controle de Conflitos de Interesse
Entrada: Política sendo gerada P
Saída: 1) Mensagem de conflito

```
1  For each UA1 ∈ UA
2      If (SSD1.R1 = P.R And SSD1.R2 = P.R) And
3          (UA2.US = P.US)
4          then  (P conflita com PA)
5          else Message  PR ← P
6          EndIf
7  EndFor
8  End Algoritmo1
```

Figura 2 – Conflitos de Interesse

Sendo o conjunto de perfil e unidade conflitante representados pela propriedade SSD.R, os quais são armazenados no arquivo de conflitos, este algoritmo recebe como entrada a política a ser gerada, onde é possível definir que papéis um usuário terá. É fornecido como saída uma mensagem de conflito caso este venha a ocorrer. Define-se, US como o usuário do sistema, SSD.R1 e SSD.R2 realizam o armazenamento dos perfis conflitantes junto com suas unidades hospitalares, presentes no arquivo de conflitos. O laço condicional nas linhas 2 e 3, realizam as seguintes comparações: verifica se a PCA que está sendo gerada, contém os mesmos perfis e unidades dos referenciados na propriedade SSD1.R1, este especificado no arquivo de conflitos, posteriormente verifica-se se o perfil 2 e setor 2 representados pela propriedade SSD.R2, são os mesmos da PCA que está sendo gerada. Após a comparação dos perfis e setores deve-se verificar o usuário, ou seja para assegurar que o perfil que está sendo verificado é o do referido usuário.

Este controle é definido por exemplo, para evitar que um usuário que está gerando PCAs venha a atribuir perfis conflitantes. Para exemplificar esta situação vamos supor o seguinte caso: um Médico responsável está gerando uma PCA para definir que o usuário Pedro, o qual já tem atribuído o perfil de Enfermeiro na Cardiologia venha a ter o perfil de Médico no mesmo setor hospitalar. Deve-se evitar que isto ocorra pois não deve ser permitido que um usuário venha a ter 2 perfis conflitantes em um mesmo setor, como pode-se observar na Figura 3.

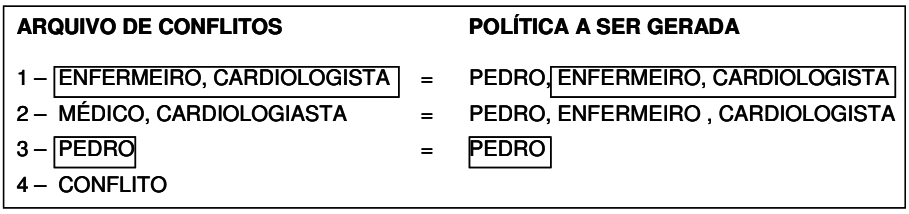

Figura 3 – Exemplo - Conflitos de Interesse

Seguindo ainda o exemplo, pode-se observar o comportamento do algoritmo, que na primeira linha verifica se o perfil de Enfermeiro e o setor de Cardiologia que estão sendo gerados são os mesmos do referenciado no arquivo de conflitos. Posteriormente, na linha 2 é verificado o perfil que o usuário está querendo atribuir. Por fim, na linha 3 é realizado um controle para ver se o usuário é ele mesmo. Se todas estas regras forem verificadas, uma condição conflitante é levantada e a política não é gerada.

5.3 Conflitos de Redundância

Esta modalidade de conflito é referenciada como conflito de SO onde conflitos de redundância podem ocorrer por PCAs duplicadas ou por políticas com inconsistências de ações. Se duas políticas são caracterizadas como mesmo sujeito, objeto e ações pode-se dizer que está duplicada e ocorre uma inconsistência, logo não deve existir (Charalambides et al. 2006). Esta modalidade de conflito pode ocasionar inconsistências e um acúmulo de PCA desnecessárias podendo assim fazer com que o mecanismo de controle de acesso que seleciona a política, não funcione de forma correta pois o mesmo seleciona a primeira PCA que encontra, desta forma podendo selecionar no repositório de políticas uma que não seja adequada para uma determinada solicitação.

Para prevenir esta modalidade de conflito, o algoritmo da Figura 4 controla o aparecimento desta modalidade de conflito não deixando gerar PCAs duplicadas. A especificação utilizada neste algoritmo baseia-se na de SO visto que não pode-se realizar duas especificações com fundamentações diferentes com os mesmos elementos.

No algoritmo apresentado na Figura 4, nas etapas iniciais do algoritmo assemelha-se a dos apresentados nas seções anteriores, tendo como entrada a PCA a ser gerada referenciada pela propriedade P logo pode-se observar a existência de uma laço condicional na linha 2 no elemento onde deve-se observar a comparação de todos os elementos da PCA que estão sendo geradas com as presentes no repositório de políticas que já foram geradas. Sendo a propriedade “N” o usuário do sistema, “O” o objeto a ser protegido, “P” o perfil, “T” a propriedade relacionada ao setor e “B” a ação que este objeto pode ter. Caso essa verificação se confirme a política não é gerada e uma mensagem de conflito é mostrada, posteriormente indicando qual a PCA está conflitando.

```
Algoritmo3 Controle de Conflitos de Redundância
Entrada: política sendo gerada P, repositório de políticas PR
Saída: 1) Mensagem de conflito
1  For each PA ⊂ PR
2        If (PA.N = P.N And PA.O = P.O And PA.P = P.P) And
3           (PA.T = P.T And PA.B = P.B)
4        then  (P conflita com PA)
5        else Message  PR ← P
6        EndIf
7  EndFor
8  End Algoritmo1
```

Figura 4 – Algoritmo de Controle de Conflitos de Redundância

6. Conclusão

Neste artigo, foram propostos algoritmos para o tratamento de conflitos em PCA, o qual é uma forma de aumentar a segurança na gerência das políticas, possibilitando assim que uma PCA venha a ser gerada de forma correta, ou seja, sem que venha causar conflitos com as já existentes no repositório das políticas. Os algoritmos definidos neste artigo possibilitam uma maior segurança no gerenciamento de políticas de controle de acesso, evitando assim que um conflito ocorra antes da geração da política.

Os algoritmos apresentados neste artigo foram implementados no SGPCA (Sistema Gerenciador de Políticas de Controle de Acesso) projeto sendo desenvolvido em conjunto, os quais deram como resultado uma maior segurança na geração das políticas, não deixando que uma PCA conflitante venha a ser gerada e possibilitando que o usuário saiba que política está conflitando, para posterior alteração.

Este projeto encontra-se ainda em fase de desenvolvimento e para finalização necessita-se realizar testes tanto com mecanismos de controle de acesso como, em sua interface propriamente dita. O SGPCA deve ser aplicado junto ao HUSM para a definição das políticas que serão aplicadas no hospital em seus respectivos setores junto com suas Hierarquias de Perfis de Usuários. Este modelo será incorporado ao CIBAC (Soares, Nunes e Amaral 2006) (Modelo de Controle de Acesso Baseado em Informações Contextuais), projeto também sendo desenvolvido em parceria com o HUSM.

7. Referências

CFM – Conselho Federal de Medicina (2002). Resolução 1.639/2002 do Conselho Federal de Medicina do Brasil.

Charalambides, M.; Flegkas, P.; Pavlou, G.; Rubio-Loyola, J.; Bandara, A.K.; Lupu, E.C.; Russo, A.; Sloman, M.; Dulay, N. (2006) "Dynamic Policy Analysis and Conflict Resolution for DiffServ Quality of Service Management". IEEE/IFIP NOMS. p.294-304.

Condell, M. (2000) "Security Policy Specification Language", Internet Draft, draft-ietf-ipsp-spsl-00.txt.

Damianou, N. (2001) "The Ponder Specification Language", Workshop on Policies for Distributed Systems ans Networks (POLICY 2001), Bristol, UK.

Dursun, T. e Orencik, P. (2004) “Police Distributed Conflict Detection Architecture”. IEEE Communications Society.

Ferraiolo, D. F.; Sandhu, R. S.; Gavrila, S. I.; Kuhn, D. R.; Chandramouli, R. (2001). “Proposed NIST standard for role-based access control”. Information and System Security, 4(3):224–274.

Ferreira, F.J.H.S.; Santos, A.R.; Junior, C.J. (2004) “LARCES_Seg - Uma Arquitetura para Gerenciamento de VPNs Baseada em Políticas utilizando XACML”. I Workshop de Computação da Região Sul (WORKCOMP-SUL), Florianópolis, Brasil.

Lorch M; Dennis K. D.; Shah S.An (2003) “XACML-based Policy Management and Authorization Service for Globus Resources”. Proc. of the Fourth IEEE International Workshop on Grid Computing (GRID’03).

Lupu E. e Sloman, M. (1997) “Conflict Analysis for Management Policies”. Proc. of the 5th International Symposium on Integrated Network Management IM’97 (ISINM), San-Diego, U.S.A., Chapman&Hall.

Moffet, J. D e Sloman, M. S. (1993) “Policy Conflict Analysis in Distributed System Management”. Journal of Organizational Computing, 4(1), p. 1-22.

Motta, G. H. M. B. (2003) “Um modelo de autorização contextual para o controle de acesso ao Prontuário Eletrônico do Paciente em Ambientes Abertos e Distribuídos”. Escola Politécnica da Universidade de São Paulo, São Paulo-SP, Brasil. (Tese de doutorado)

Oasis. (2003) Organization for the Advancement of Structured Information Standards “Extensible Access Control Markup Language (XACML) Specification Set v1.0”,. Oasis XACML TC.

Sandhu, R. S. e Samarati, P. (1994) “Access Control: principles and practice”. IEEE Communications Magazine, p. 40-48.

Shankar, S. C. (2005) “An ECA-P Policy-based Framework for Managing Ubiquitous Computing Environments”. Proc. of the Second IEEE International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous’05).

Soares, G. A; Nunes R.C.; Amaral, É M. H. do. (2006) “Um Modelo de Controle de Acesso Baseado em Contexto para Autorizações a Informações Médicas”. In: XXXII Conferência Latino-Americana de Informática, Santiago do Chile.

Sun Microsystem, XACML Implementation. http://sunxacml.sourceforge.net/