Gerenciamento Baseado em Modelos da Configuração de Sistemas de Segurança em Redes de Larga Escala

João Porto de Albuquerque[1,2]*, Holger Isenberg[2], Heiko Krumm[2], Paulo Lício de Geus[1]

[1]Instituto de Computação – Universidade Estadual de Campinas
13083-970 Campinas/SP Brazil
{jporto, paulo}@ic.unicamp.br

[2]FB Informatik – University of Dortmund
44221 Dortmund Germany
{Joao.Porto, Heiko.Krumm}@udo.edu

Abstract. *The security mechanisms employed in today's networked environments are increasingly complex and their configuration management has an important role for the protection of these environments. Especially in large scale networks, security administrators are faced with the challenge of designing, deploying, maintaining and monitoring a huge number of mechanisms, most of which have complicated and heterogeneous configuration syntaxes. A consequence of this is that configuration errors are a frequent cause of security vulnerabilities. This work offers a management process for the configuration of network security systems that is built upon the model-based management approach. We present a modelling technique that uniformly handles different types of mechanisms and a supporting graphical editor for the design of the system. The editor incorporates focus and context concepts in order to improve the visualization and the navigation of large models.*

Resumo. *Os mecanismos de segurança empregados em ambientes de redes atuais são de crescente complexidade e o gerenciamento de suas configurações adquire, portanto, um papel fundamental para proteção desses ambientes. Particularmente em redes de computadores de larga escala, os administradores de segurança vêem-se confrontados com o desafio de projetar, implementar, manter e monitorar um elevado número de mecanismos, os quais em sua maioria possuem sintaxes de configuração heterogênea e complicada. Uma conseqüência dessa situação é que erros de configuração são causas freqüentes de vulnerabilidades de segurança. O presente trabalho oferece uma sistemática para o gerenciamento da configuração de sistemas de segurança de redes, construída sobre a abordagem de gerenciamento baseado em modelos. Apresentamos aqui uma técnica de modelagem que trata uniformemente diferentes tipos de mecanismos e é apoiada por um editor gráfico para o projeto do sistema. O editor incorpora conceitos de foco e contexto para facilitar a visualização e navegação de grandes modelos.*

1. Introdução

Diversas tecnologias e tipos de mecanismos de segurança são empregados nos ambientes de rede atuais, com o intuito de oferecer proteção contra ataques através da rede. Assim, o administrador da segurança se vê confrontado com a complexa tarefa de ajustar

*Financiado pelo Serviço Alemão de Intercâmbio Acadêmico (DAAD).

corretamente as configurações desses diferentes mecanismos, para que assegurem o cumprimento das políticas de sua organização. Como novas vulnerabilidades e técnicas de invasão são descobertas diariamente, os mecanismos necessitam, ainda, ser continuamente reconfigurados para acompanhar a evolução das ameaças.

Enquanto que um progresso significativo foi alcançado na melhoria da tecnologia de segurança de redes nos últimos anos, apenas uma modesta atenção tem sido dada à sua interface de configuração. Há, de fato, alguns produtos com uma interface gráfica um pouco mais palatável. Estes restringem-se, entretanto, a tipos particulares de mecanismos, não mitigando a árdua tarefa da configuração da segurança como um todo. Na prática, o administrador da segurança se vê obrigado a lidar com sintaxes de configuração complicadas e heterogêneas, a maioria das quais não é intuitiva e, em alguns casos, até mesmo induz ao erro. A situação é especialmente dramática em ambientes de larga escala, nos quais se torna muito difícil obter uma visão global dos numerosos mecanismos de segurança empregados, os quais têm de ser postos em harmônica cooperação. Nesse contexto, um único desajuste entre dois quaisquer mecanismos pode deixar vulnerável todo o sistema.

Tendo em conta o cenário delineado, não deveria surpreender a conclusão do paradigmático trabalho de Anderson: "a maioria das falhas de segurança se deve a erros de implementação e gerenciamento"[Anderson, 1994]. Além disso, a situação não parece ter evoluído satisfatoriamente, pois os mesmos resultados foram confirmados aproximadamente dez anos depois num recente estudo sobre três serviços de grande porte na Internet [Oppenheimer et al., 2003]. Este estudo conclui que os erros de configuração constituem a maior categoria dos erros de operação – estes sendo, mesmo, a causa mais freqüente de falha em dois dos três serviços analisados.

Por conseguinte, abordagens que ofereçam abstração, integração e ferramentas de suporte ao gerenciamento da configuração de mecanismos de segurança são fundamentais para tornar o processo de configuração menos sujeito a erros e mais efetivo. Dentro desse processo, quatro tarefas básicas podem ser distinguidas do ponto de vista do administrador: i) o projeto do sistema de segurança, incluindo a definição das tecnologias e mecanismos a serem empregados, assim como o posicionamento dos diferentes componentes dentro da rede; ii) a implantação da configuração projetada no sistema real; iii) a manutenção da configuração, possibilitando a introdução de mudanças, de forma a obter adaptabilidade em face a novos requisitos; iv) monitoramento do sistema durante a operação, assegurando um funcionamento compatível com o comportamento esperado.

O presente trabalho abrange as três primeiras etapas do gerenciamento da configuração. Para apoiar a fase de projeto, empregamos uma técnica de modelagem que viabiliza o projeto do sistema de segurança a ser gerenciado de forma modular, mediante um modelo orientado a objetos [Porto de Albuquerque et al., 2005a]. Esse modelo é segmentado em unidades lógicas (denominadas *Abstract Subsystems*, ou subsistemas abstratos) que abarcam um grupo de mecanismos de segurança e outras entidades relevantes do sistema, oferecendo, também, uma representação mais abstrata deles. Dessa forma, o administrador do sistema pode projetar um sistema de segurança – incluindo seus diferentes tipos de mecanismos e suas relações mútuas – por meio de uma técnica de modelagem abstrata e uniforme.

Uma ferramenta de suporte apóia a modelagem, provendo um editor gráfico de modelos. Esse editor incorpora conceitos de *foco e contexto* – que são originários da pesquisa em visualização da informação [Card et al., 1999] –, através das técnicas *visão olho-de-peixe* (*fisheye-view*) [Furnas, 1986] e *zoom semântico* (*semantic zooming*) [Köth and Minas, 2002, Musial and Jacobs, 2003]. Além disso, nosso trabalho se utiliza

das abordagens de hierarquias de política [Moffett and Sloman, 1993] e gerenciamento baseado em modelos [Lück et al., 2002] para assistir às fases de implantação e manutenção supra-citadas. Assim, um modelo do sistema organizado em diferentes níveis de abstração proporciona uma modelagem assistida passo-a-passo pela ferramenta, culminando em um refinamento automático das políticas até a geração de parâmetros de configuração de baixo nível para os mecanismos. Logo, enquanto que este refinamento automático contempla a fase de implantação da configuração, o suporte à fase de manutenção é provido pela possibilidade de edição dos modelos e repetição do processo de geração automática.

O restante deste trabalho está organizado da seguinte forma: a Seção 2 apresenta os principais elementos de nossa técnica de modelagem, e a Seção 3 descreve as técnicas de foco e contexto incorporadas na ferramenta de suporte. Posteriormente, um caso de estudo que exemplifica a aplicação prática da abordagem em um típico ambiente de larga escala é apresentado na Seção 4. Na Seção 5 são discutidos trabalhos correlatos e, finalmente, a Seção 6 oferece conclusões para o presente trabalho.

2. Técnica de Modelagem

Nossa técnica de modelagem se fundamenta na abordagem de Gerenciamento Baseado em Modelos (*Model-based Management*, MBM) [Lück et al., 2002], que emprega um modelo do sistema estruturado em três camadas, como ilustrado na Fig. 1. As linhas tracejadas horizontais da figura delimitam os níveis de abstração do modelo: *Roles & Objects* (RO), *Subjects & Resources* (SR), and *Diagram of Abstract Subsystems* (DAS). Cada um desses níveis é um refinamento do seu superior no sentido de uma hierarquia de políticas [Moffett and Sloman, 1993], isto é, passando-se de um certo nível a seu inferior, a visão mais abstrata contida no nível superior é complementada por uma representação de nível de abstração mais baixo, a qual é mais detalhada e mais próxima ao sistema real. Já em relação à subdivisão vertical, ela separa o modelo do sistema a ser gerenciado (à esquerda) das políticas que o devem regular (à direita).

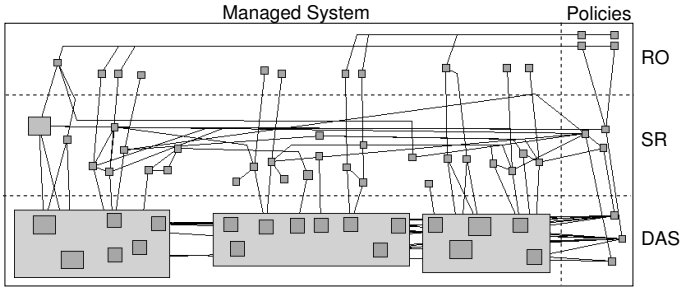

Figura 1: Visão geral do modelo

Como o nível mais inferior do modelo (DAS) é o foco do presente trabalho, ele será explicado detalhadamente na próxima seção. Os dois níveis superiores (RO e SR), por sua vez, foram adotados de trabalhos anteriores em MBM, e são, portanto, apresentados brevemente a seguir.

O nível RO utiliza conceitos do controle de acesso baseado em papéis (*Role-Based Access Control*, RBAC) [Sandhu et al., 1996]. As principais classes desse nível são: *Roles*, modelando papéis organizacionais nos quais atuam as pessoas que trabalham no ambiente modelado; *Objects*, representando os objetos que devem ser sujeitos ao controle de acesso; e *AccessModes*, que mapeiam os modos de acesso aos objetos. A classe *AccessPermission* expressa uma política de segurança, permitindo que alguém desempenhando um papel (*Role*) acesse um objeto (*Object*) segundo um particular modo de acesso (*AccessMode*).

O segundo nível (SR na Fig. 1) apresenta uma visão do sistema definida com base nos serviços que o mesmo deverá prover. Objetos desse nível representam: (a) pessoas que trabalham no ambiente modelado (classe *User*); (b) sujeitos (ou sessões) que agem a comando dos usuários, executando suas requisições (*SubjectType*); (c) serviços na rede que são utilizados para acessar recursos (*Service*); (d) dependências entre serviços (*ServiceDependency*); (e) recursos da rede (*Resources*) que são utilizados por serviços. A classe *ServicePermission* define políticas de autorização nesse nível, representando a permissão para que um sujeito, agindo no interesse de um usuário, acesse um recurso através do uso de um serviço.

2.1. Diagrama de Subsistemas Abstratos

O principal objetivo do *Diagrama de Subsistemas Abstratos* (DAS, na sigla em inglês) é descrever a *estrutura geral* do sistema a ser gerenciado de forma modular; i.e. evidenciando os principais blocos de construção do sistema e as interconexões entre os mesmos. Assim sendo, um DAS é definido formalmente como um grafo composto de *Subsistemas Abstratos* (AS, na sigla em inglês) como nós e com arestas que representam a possibilidade de comunicação bidirecional entre ASs. Um AS, por sua vez, contém uma visão abstrata de um certo segmento do sistema, ou seja, uma representação de mais alto nível de um certo grupo de componentes. Essa representação se constitui dos seguintes tipos de elementos:

Actors: (atores) grupos de entidades que tem um comportamento ativo no sistema, i.e. iniciam conexões e executam operações de acordo com as políticas de obrigação do sistema;

Mediators: (mediadores) elementos que intermediam comunicações, recebendo requisições, inspecionando tráfego e/ou transformando o fluxo de dados; eles também podem executar operações obrigatórias, como o registro de informação a respeito de fluxos de dados;

Targets: (alvos) elementos passivos; eles contêm informações relevantes, as quais são acessadas por *Actors*;

Connectors: (conectores) representam as interfaces de comunicação de entre dois ASs, possibilitando o fluxo de informações de um AS para outro.

Cada elemento dos tipos *Actors*, *Mediators* ou *Targets* representa, portanto, um grupo de mecanismos que tem um comportamento relevante para uma visão global do sistema. Os *Connectors*, por sua vez, estão relacionados às interfaces físicas de um AS. Dessa forma, um DAS permite a apreciação da estrutura do sistema *vis-à-vis* as políticas executáveis que a ele se aplicam, tornando explícita a distribuição dos diferentes participantes dessas políticas ao longo da arquitetura do sistema[1].

Com o intuito de modelar as próprias políticas de segurança, um outro tipo de objetos estará também presente em um DAS: *ATPathPermissions* (ATPP). Uma ATTP se relaciona a um caminho (p) em um DAS que parte de um *Actor* (A) e chega a um *Target* (T), e que possivelmente contém *Mediators* e *Connectors* nesse trajeto. Ela expressa, assim, uma permissão para que p seja usado por A com o intuito de acessar T – modelando, portanto, uma *política de autorização*, ou política de controle de acesso. No entanto, os objetos ATTP em um modelo não serão definidos pelo projetista do sistema, mas derivados automaticamente pelo sistema, conforme o processo descrito adiante na Seção 4.4.

Adicionalmente, cada AS em um DAS também estará associado a uma visão detalhada dos mecanismos do sistema real. Essa visão expandida incorpora objetos que

[1]Para uma explanação detalhada da modelagem com subsistemas abstratos ver [Porto de Albuquerque et al., 2005a].

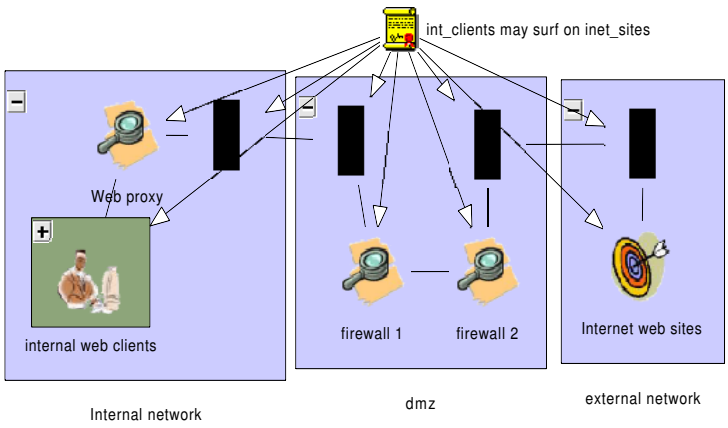

Figura 2: Exemplo de DAS

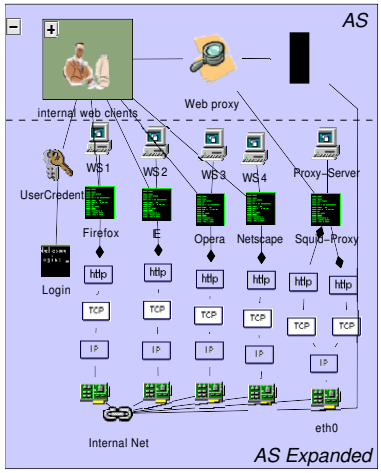

Figura 3: AS Expandido

representam máquinas, processos, protocolos e interfaces de rede do sistema (essa visão detalhada relaciona-se com o nível *Process & Hosts* em trabalhos anteriores do gerenciamento baseado em modelos, ver a respeito [Lück et al., 2002]).

2.1.1. Exemplo de DAS

O exemplo exibido na Fig. 2 corresponde a um ambiente de rede simples, o qual contém três ASs: "internal network" (rede interna), "dmz" (zona desmilitarizada, no acrônimo em inglês) e "external network" (rede externa). No AS "internal network", o objeto "internal web clients" é um *Actor*, representando um grupo de processos que são autorizados a acessar aqueles processos que são mapeados pelo *Target* "internet web sites" (no AS "external network") através do *Mediator* "Web proxy". Essa autorização é modelada pela ATPP "int_clients may surf on inet_sites" e suas conexões com os objetos anteriores. Os *Mediators* "firewall 1" e "firewall 2", por sua vez, correspondem a processos que inspecionam e controlam os fluxos de comunicação que os atravessam.

A Fig. 3 mostra ambas representações – abstrata e expandida – para o AS "internal network" (mais à esquerda na Fig. 2). Cada objeto na visão abstrata do AS (topo) está conectado aos elementos do modelo que representam as entidades correspondentes no sistema real; por exemplo, o *Actor* "internal web clients" está relacionado a seus respectivos processos de navegadores web.

Essa representação dupla de um AS proporciona ao projetista um modelo flexível do sistema, que oferece não só uma descrição mais abstrata, concisa e compreensível da estrutura do sistema, mas também uma visão detalhada de seus mecanismos reais. Ela também constitui a base para o processo de geração automática de parâmetros de configuração (explicado adiante na Seção 4.4) e para a aplicação das técnicas de visualização e navegação apresentadas na próxima seção.

3. Foco e Contexto

A expressão *foco e contexto* (*focus and context*) se refere a técnicas que possibilitam ao usuário centralizar sua visão em uma parte do modelo que é exibida em detalhes (foco), percebendo, ainda, simultaneamente, áreas periféricas de forma menos detalhada (contexto). De acordo com [Card et al., 1999], essas técnicas fundamentam-se em três premissas básicas: a) o usuário necessita ao mesmo tempo de uma visão geral e uma representação detalhada; b) podem haver requisitos diferentes para a informação na área

detalhada em relação à visão geral; c) ambos tipos de informação podem ser combinados em uma única visão. A principal vantagem da utilização dessas técnicas é a melhoria da eficiência espaço-tempo para o usuário, isto é, a informação exibida por unidade de área da tela é mais útil e, conseqüentemente, o tempo requerido para encontrar um item de interesse é reduzido, já que é mais provável que esse item esteja na tela.

Nós adotamos o conceito de foco e contexto segundo dois diferentes métodos. O primeiro deles se aplica à estrutura de elementos compostos de um modelo e denomina-se *zoom semântico* (*semantic zooming*). O segundo método se baseia em uma projeção gráfica específica do modelo no espaço euclidiano bidimensional, a qual se dá o nome de *visão olho-de-peixe* (*fisheye view*). As seções que se seguem descrevem respectivamente cada uma dessas técnicas.

3.1. Zoom Semântico

O conceito de zoom semântico [Musial and Jacobs, 2003, Köth and Minas, 2002] se baseia na habilidade de exibir objetos de um modelo em diferentes níveis de abstração, dependendo de sua distância em relação ao foco. Assim, os objetos localizados dentro da região focalizada de um diagrama são exibidos em sua forma mais detalhada (menor abstração), enquanto que os objetos localizados nas regiões mais periféricas são visualizados da maneira mais simplificada (maior abstração). As regiões entre esses dois extremos são representadas com níveis intermediários de detalhes. Dessa forma, a informação apresentada é reduzida seletivamente pelo ajuste do nível de detalhe em cada região em função do interesse do usuário naquela área – um princípio básico das técnicas de foco e contexto. No caso particular do zoom semântico, contudo, os diferentes níveis de detalhe empregados não se relacionam a propriedades gráficas dos objetos, mas sim ao tipo de informação que eles carregam; i.e. à sua *semântica*.

Em nosso contexto há duas classes de objetos compostos, para as quais se aplica o zoom semântico: *typed folders* (pastas tipadas) e *Abstract Subsystems* (ASs). Um *typed folder* é utilizado para agregar um grupo de objetos da mesma classe (ou tipo), com o intuito de aumentar a concisão da representação. Por outro lado, um AS contém objetos de várias classes (Seção 2.1) e pode também abarcar *typed folders*. Assim, em ambos os casos, o nível de detalhes da representação gráfica pode ser alterado mediante a exibição seletiva de objetos internos.

A situação mais simples ocorre com *typed folders*, para os quais duas representações diferentes são possíveis: pasta fechada (todos os objetos internos são escondidos) e aberta. Quanto aos ASs, três diferentes níveis de abstração são utilizados: i) uma representação totalmente detalhada que inclui todos os objetos internos, ou seja, ambas as visões abstrata e expandida (como na Fig. 3); ii) uma representação abrangendo apenas os objetos da visão abstrata (como na Fig. 2); e iii) uma visão “fechada", na qual todos os objetos internos não são exibidos.

3.2. Visão Olho-de-peixe

O termo *visão olho-de-peixe* refere-se ao tipo de projeção criado por uma lente olho-de-peixe usada na fotografia. Esse tipo de lente abrange um campo de visão de 180° e é não-corrigida. Como resultado, obtém-se uma ampliação ótica dos objetos perto do centro em relação aos das bordas. Essa característica emula a percepção visual humana, a qual pelo efeito dos movimentos dos olhos tem uma área de foco nítida e uma gradual perda de resolução visual na direção das regiões periféricas. Uma visão olho-de-peixe combina, portanto, uma visão geral da imagem completa com uma degradação progressiva de detalhes que aumenta com a distância a partir do foco – sendo, portanto, adequada para implementar o conceito de foco e contexto. Em oposição ao zoom semântico, a visão

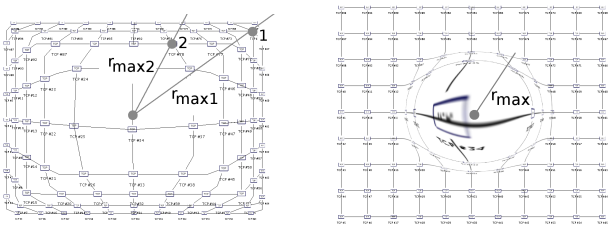

Figura 4: Visão olho-de-peixe com raio variável e fixo

olho-de-peixe manipula o *tamanho gráfico* com que são exibidos os objetos, de forma a alterar a quantidade de informação apresentada.

Uma primeira formalização da visão olho-de-peixe para a visualização de dados é oferecida por [Furnas, 1986], enquanto que uma aplicação prática na visualização de grafos é apresentada em [Sarkar and Brown, 1992]. Esse último trabalho traz o benefício de prover uma visão auto-adaptativa com um raio variável (r_{max}) para a área focalizada a ser ampliada. Em comparação com uma visão que usa raio fixo, a abordagem variável define o raio dinamicamente como a distância entre o ponto focal e as bordas da imagem. A Figura 4 ilustra a vantagem do uso da técnica de raio variável (à esquerda): esta produz uma maior região de foco do que a visão com raio fixo (à direita).

Por essa razão, empregamos uma abordagem de raio variável na qual o ponto focal pode, ainda, ser deslocado livremente pelo usuário ao longo do modelo. Dessa forma, objetos dentro da área focalizada são exibidos com uma escala maior, enquanto que os outros tornam-se gradualmente menores a medida que se aproximam das bordas do modelo. O projetista trabalhando num modelo pode, com isso, permanecer consciente do contexto no qual trabalha, ao mesmo tempo em que detalhes desnecessários são suprimidos através de miniaturização ótica.

A função de transformação que usamos (obtida de [Sarkar and Brown, 1992]) é definida com base na distância r entre o centro da visão e a coordenada que deverá ser projetada, de acordo com a fórmula:

$$f(r) = r_{max} \frac{(v+1)\frac{r}{r_{max}}}{\frac{r}{r_{max}} + 1} \tag{1}$$

Nessa expressão, o parâmetro de distorção $v \geq 0$ controla a ampliação da área focalizada em relação a suas circunvizinhanças. O raio variável r_{max} é o tamanho da extensão do raio r até as bordas da área visível.

4. Estudo de Caso

Para analisar a aplicabilidade prática dos conceitos previamente descritos, oferecemos nesta seção um paradigmático caso de estudo. O cenário considerado consiste em uma rede de uma grande corporação, a qual é composta por um escritório principal e uma filial, ambos conectados à Internet. Nosso objetivo principal, portanto, é auxiliar o administrador da segurança nas tarefas de projetar e implantar a configuração para os mecanismos de segurança que são necessários para habilitar e controlar os serviços de correio eletrônico e acesso à *web* pelos empregados da companhia.

As políticas de segurança de alto nível para esse ambiente são assim formuladas:

P1: Os empregados podem surfar na web a partir dos computadores do escritório principal e da filial;

P2: Usuários na Internet podem enviar e-mail para endereços internos da companhia.

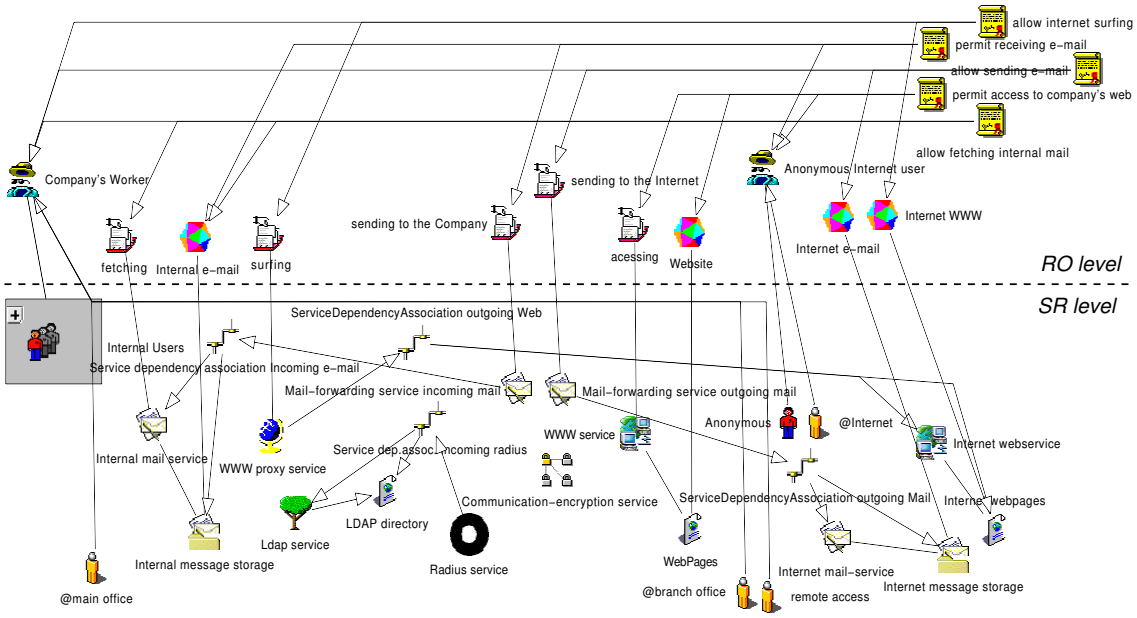

Figura 5: Modelo dos níveis RO e SR

P3: Os empregados podem enviar e-mails para endereços internos ou externos;
P4: Usuários na Internet podem acessar o servidor web corporativo;
P5: Os empregados podem ler seus e-mails corporativos tanto a partir do escritório principal, como também da filial e de casa;

Tendo como ponto de partida essas políticas abstratas e o cenário previamente descrito, aplicamos nossa técnica de modelagem segundo um processo passo-a-passo para o projeto da configuração, passando pelos diferentes níveis de abstração de nosso modelo. Esse processo desenvolve-se primeiramente ao longo das três etapas de modelagem discutidas respectivamente nas seções 4.1, 4.2 e 4.3, culminando, então, na derivação automática dos parâmetros de configuração descrita na Seção 4.4. Adicionalmente, a Seção 4.5 explora questões relativas à edição, navegação e visualização ao longo do processo anterior, analisando as vantagens obtidas pelo uso das técnicas descritas na Seção 3.

4.1. Modelagem do nível RO

Como o nível mais alto de nosso modelo se baseia em conceitos do RBAC (Seção 2), o projetista deve iniciar o desenvolvimento com o mapeamento das políticas abstratas, expressas em linguagem natural, para a sintaxe mais formal do RBAC. Logo, cada uma das cinco sentenças da seção anterior devem ser expressas por objetos dos tipos *Roles*, *Objects*, *AccessModes* e *AccessPermissions*, e suas interrelações.

O topo da Fig. 5 mostra o modelo resultante no nível RO para o cenário considerado. Os objetos básicos são: as *Roles* "Company's Worker" (empregados) e "Anonymous Internet User" (usuário anônimo na Internet), e os *Objects* "Internal e-mail", "Website", "Internet e-mail" e "Internet WWW". Esses objetos são associados a *AccesModes* através de cinco *AccessPermissions* (acima à direta da Fig. 5), cada uma das quais correspondendo a uma das sentenças de política da seção anterior. Assim, por exemplo, a *AccessPermission* "allow Internet surfing" (permitir surfar na Internet) modela a sentença **P1**, ao associar o papel "Company's Worker" a "surfing" e "Internet WWW". As outras políticas são analogamente modeladas pelas outras *AccessPermissions*.

4.2. Modelagem de Usuários, Serviços e Recursos

A segunda fase no processo de projeto consiste na definição dos serviços que o sistema deve prover, dos recursos empregados e dos usuários que podem utilizá-los. Em rela-

ção à modelagem de usuários, para cada *Role* do nível RO devem ser definidos objetos relacionados das classes *User* e *SubjectType*. Essas duas classes adicionam informação sobre cada usuário que pode atuar em um certo papel e os possíveis sujeitos (*subjects*) ou sessões de que ele se pode valer (ambos os termos usuários e sujeitos referem-se aos conceitos homônimos da terminologia RBAC).

No exemplo, o *User* "Anonymous" e o *SubjectType* "@Internet" são associados ao papel "Anonymous Internet User". Para o papel "Company's Worker", diversos objetos *User* estão agrupados no *TypedFolder* "Internal Users" (Seção 3.1), e três *SubjectTypes* são definidos: "@main office", "@branch office" and "@remote access". Esses objetos mapeiam os três tipos de sessão que podem ser estabelecidas por um empregado no cenário considerado, dependendo de sua localização física.

Em relação à modelagem de serviços e recursos, um objeto *Service* será definido, basicamente, para cada *AccessMode* no nível RO, enquanto que cada *Object* neste nível será mapeado para um *Resource*. Nos casos em que mais de um serviço é necessário para prover acesso a um recurso, esse fato será expresso por um objeto *ServiceDependency*. Isso é o que ocorre no nosso modelo, por exemplo, com o *AccessMode* "sending to the company" (enviar para a companhia), o qual está relacionado ao *Service* "Mail-forwarding service incoming e-mail" (serviço para encaminhamento de e-mails entrantes). Este serviço precisa do "Internal mail service" (serviço de e-mails interno) para que possa prover acesso ao recurso "Internal message store" (depósito de mensagens interno) – o qual está, por sua vez, relacionado ao *Object* "Internal e-mail".

Além disso, serviços e recursos podem também ser definidos para representar componentes mais técnicos do sistema que não são modelados no nível RO. No exemplo, os serviços "Communication-encryption service", "Radius service" e "LDAP service" pertencem a essa categoria.

4.3. Modelagem de Subsistemas Abstratos

Para produzir um DAS, o projetista deve iniciar com a identificação dos principais segmentos no quais se subdivide o sistema. Considerando nosso exemplo e tendo também em conta trabalhos consolidados sobre técnicas de segurança (ver [Zwicky et al., 2000], por exemplo), uma subdivisão estrutural em cinco blocos pode ser definida: a rede interna, a rede desmilitarizada (dmz, no acrônimo em inglês), a rede da filial, os pontos de acesso remoto (representando as residências dos usuários) e a rede externa (a Internet). Assim sendo, o DAS para esse exemplo possui um AS para cada um desses segmentos.

Na seqüência, o projetista deve definir, para cada subsistema, os mecanismos de segurança e os outros elementos de rede relevantes em face às políticas de segurança; i.e. deve modelar a visão expandida de cada AS (Seção 2.1). As classes de objetos que devem ser consideradas são: os computadores do sistema (*hosts*); os processos que tomam parte na comunicação correspondente às políticas e os respectivos protocolos que utilizam; as interfaces e segmentos de rede; e as credenciais de usuários, como nomes de login e certificados. Essas classes básicas são usadas para definir tanto os componentes do sistema como os mecanismos de segurança empregados para seu controle. Para estes, classes de processo especiais são também definidas para representar cada tipo específico de mecanismo contemplado.

A parte inferior da Fig. 6 mostra a visão expandida do AS "internal network". Ela contém objetos que representam os processos rodando em sete estações de trabalho, um servidor de e-mails, um *web proxy* e um servidor de diretórios LDAP. Cada um desses processos está conectado através de um pilha de protocolos adequada – modelada por um série de objetos de protocolo interconectados – a sua respectiva interface de rede.

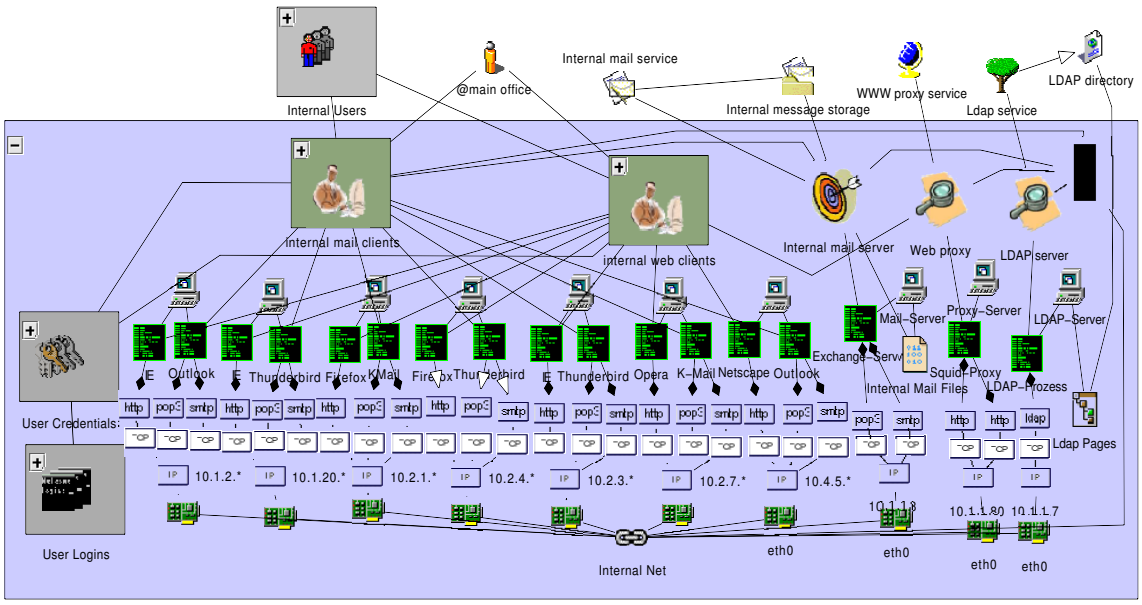

Figura 6: Extrato de um DAS e sua relação com o nível SR

Posteriormente, a visão abstrata de cada AS deve ser definida através da criação de objetos para *Actors*, *Mediators*, *Targets* e *Connectors*, e do estabelecimento de associações desses com os objetos do nível SR. Para tanto, o comportamento dos elementos da visão detalhada, previamente definida, será classificado de acordo com as referidas classes (uma abordagem detalhada desse mapeamento pode ser obtida em [Porto de Albuquerque et al., 2005a]).

No exemplo, os *Actors* "internal mail clients" e "internal web clients" (Fig. 6) são criados no AS "internal network" para mapear os processos desse subsistema com comportamento ativo. Eles também são conectados aos objetos no nível SR que representam o mesmo comportamento: "Internal Users" e "@main office". Por outro lado, os *Mediators* "Web proxy" e "LDAP Server" são criados para refletir sua função mediadora, ou de apoio; eles são também conectados aos processos correspondentes na visão expandida e aos serviços respectivos do nível SR.

Procedendo de forma análoga para todos os ASs restantes de nosso exemplo, um DAS completo é obtido e, conseqüentemente, encerra-se a fase de projeto do sistema.

4.4. Refinamento de Políticas e Geração da Configuração

Depois da definição de todos os níveis de abstração do modelo, o administrador da segurança poderá, então, utilizar-se de nossa ferramenta de suporte na implantação dos parâmetros de configuração para os mecanismos de segurança modelados. Isso ocorre mediante a construção automática de uma hierarquia de políticas, a qual toma como ponto de partida a política de segurança de alto nível definida pelo projetista no nível RO (Seção 4.1) e deriva políticas para os níveis inferiores do modelo (SR e DAS). Em cada passo desse processo – o qual recebe o nome de refinamento (ou transformação) de políticas – a análise dos objetos do sistema, das associações entre eles, e das políticas de um certo nível possibilita a geração de um conjunto de políticas para o nível imediatamente inferior, com base, ainda, nos objetos desse nível mais inferior e nas relações destes com as entidades do nível superior.

Dessa forma, a ferramenta primeiro refina cada uma das *AccessPermissions* modeladas (no nível RO) em uma ou mais *ServicePermissions* no nível SR. Posteriormente, um conjunto de objetos *ATPathPermission* (Seção 2) é gerado a partir das *ServicePermissions*. O processo de refinamento prossegue com a derivação de *ProtocolPermissions* das

ATPathPermissions. Cada *ProtocolPermission* relaciona-se com um conjunto de objetos na representação detalhada de um AS, de modo a denotar que um processo iniciador, ao qual se associa uma credencial de usuário, está autorizado a se comunicar – por meio de uma entidade de protocolo local e um protocolo remoto – com um processo servidor, com o intuito de acessar um certo recurso físico (uma explanação pormenorizada do processo de refinamento está fora do escopo do presente trabalho e pode ser encontrada em [Porto de Albuquerque et al., 2005b]).

Finalmente, como última etapa da implantação da configuração, uma série de módulos *back-end* são executados, cada um dos quais correspondente a um produto que implementa um serviço de segurança específico (e.g. Kerberos, FreeS/WAN, Linux IP tables etc.). Esses módulos avaliam, então, as *ProtocolPermissions* e a visão expandida dos ASs, gerando os arquivos de configuração adequados para cada produto. Mais detalhes sobre esse processo podem ser obtidos em [Lück et al., 2002].

4.5. Edição, Navegação e Visualização de Modelos

Para que se possa editar partes específicas de grandes modelos (como o que foi utilizado para nosso estudo de caso), o usuário necessita utilizar técnicas para ampliação de extratos do modelo, ou seja, técnicas de zoom. Entretanto, com o método padrão de zoom, que se baseia na ampliação linear de um pedaço do modelo de tamanho fixo, a navegação e visualização se tornam problemáticas. Nesse ponto, os benefícios proporcionados pela visão olho-de-peixe podem ser comprovados pela consideração de uma simples tarefa de desenho do sistema. Primeiramente, consideramos o que ocorre quando se usa zoom tradicional, comparando, posteriormente, com a situação quando do emprego da visão olho-de-peixe.

A tarefa consiste em conectar um objeto na área do modelo que se está correntemente editando a outro, localizado numa região oposta dentro de um grande modelo. Nesse caso, "grande modelo" significa que ele não cabe na tela quando exibido em uma escala que viabilize sua edição. Utilizando o método de zoom tradicional, o usuário necessita, para tanto, efetuar os seguintes passos: 1) reduzir a escala (*zoom out*), de forma a poder visualizar o modelo inteiro; 2) estimar as localizações dos objetos alvo e origem e o ângulo da aresta necessária para conectá-los; 3) ampliar a área em torno do objeto origem, de forma a poder visualizá-lo precisamente; 4) selecionar o objeto origem; 5) clicar e arrastar uma nova aresta a partir da origem na direção do ângulo previamente estimado (a região ampliada do modelo se move automaticamente, seguindo o mouse); 6) parar de arrastar, uma vez que o alvo se tornar visível na tela; 7) soltar a aresta sobre o objeto alvo, marcando-o como ponto final.

Já com o auxílio da visão olho-de-peixe, apenas os seguintes passos são necessários: 1) ativar o modo visão olho-de-peixe; 2) selecionar o objeto origem, o qual é exibido dentro da área de foco ampliada (marcado como *source* na Fig. 7); 3) clicar e arrastar uma nova aresta a partir da origem na direção do alvo, cuja localização pode ser simultaneamente visualizada dentro da área periférica miniaturizada; (a área de foco ampliada segue o mouse durante esse processo, de maneira que o alvo poderá ser, enfim, visto em detalhes; 4) parar de arrastar, uma vez que o alvo se tornar visível na tela; 5) soltar a aresta sobre o objeto alvo, marcando-o como ponto final (a área ao redor do alvo está ampliada, como mostrado na Fig. 7).

Logo, o emprego da visão olho-de-peixe reduz o número de passos necessários, e proporciona uma redução ainda maior no tempo para realizar a tarefa, pois a mudança abrupta entre diferentes escalas da técnica tradicional requer um tempo extra para que o usuário se oriente. Além disso, a forma de utilização é imediatamente intuitiva, já que o usuário nunca perde de vista o contexto maior em que está trabalhando. A técnica

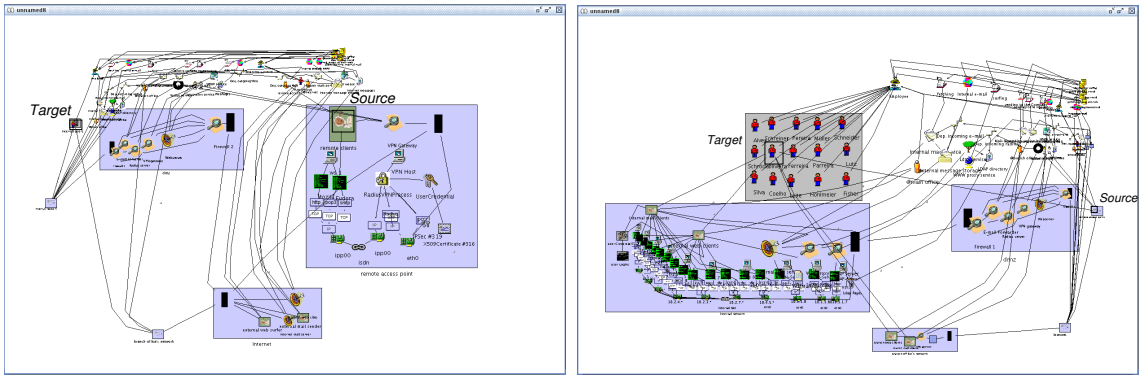

Figura 7: Visão olho-de-peixe com o foco nos objetos origem e alvo

não-linear de miniaturização gradual da visão olho-de-peixe parece, portanto, "natural" em contraste com os extratos ampliados linearmente e alijados de suas vizinhanças do método tradicional.

4.5.1. Combinando Visão Olho-de-peixe e Zoom Semântico

Como as duas técnicas introduzidas na Seção 3 operam em componentes ortogonais – nomeadamente, a visão olho-de-peixe na parte gráfica e o zoom semântico na estrutura do modelo – eles podem ser combinadas. Implementamos essa combinação utilizando o fator de ajuste resultante da função de transformação olho-de-peixe (i.e. da Equação 1 da Seção 3.2) para ajustar o nível de abstração que em que são exibidos os componentes compostos de um modelo (i.e. os ASs e os *typed folders*).

Como resultado, a medida que um objeto está localizado a uma maior distância do foco, esse objeto é apresentado em uma visão gradualmente mais abstrata – a qual possui *per se* uma representação gráfica menor – e, ainda, adicionalmente miniaturizado pela função olho-de-peixe. Por conseguinte, uma maior área em foco é possibilitada sem perder, no entanto, o contexto, otimizando, assim, o uso do espaço de tela.

A Fig. 7 apresenta o efeito desse uso combinado. À esquerda, o AS "remote access point" (que contém o objeto "source") detém o foco: ele é, portanto, exibido em uma escala maior, com todos os seus detalhes, possibilitando a edição de seus elementos internos. Os ASs "dmz" e "Internet" pertencem a um contexto próximo e são exibidos em sua representação abstrata, com tamanho gradualmente menor, enquanto que os ASs remanescentes são exibidos em uma visão "fechada" e miniaturizada, economizando espaço na tela, possibilitando, ainda assim, que o usuário perceba a sua existência.

5. Trabalhos Correlatos

Há diversas abordagens que empregam técnicas de foco e contexto no sentido de melhorar a usabilidade de editores gráficos genéricos, incluindo, por exemplo, uma recente aplicação à UML em [Musial and Jacobs, 2003] e a técnica mais genérica proposta em [Köth and Minas, 2002]. Contudo, até onde pudemos verificar, essas técnicas não tinham ainda sido aplicadas no contexto de visualização e navegação para o projeto de sistema de segurança de redes.

Em um contexto mais amplo, em [Damianou et al., 2002] apresenta-se um conjunto de ferramentas para especificação, implantação e gerenciamento de políticas especificadas na linguagem Ponder. Essa linguagem oferece suporte à definição de políticas

de gerenciamento e segurança e se baseia em domínios, os quais são estruturas hierárquicas usadas para agrupar objetos gerenciados. O protótipo descrito inclui também um navegador de domínios que utiliza visão olho-de-peixe para lidar com grandes estruturas. Por centrar sua abordagem em políticas, esse trabalho não oferece, entretanto, uma representação da arquitetura do sistema a ser gerenciado, tornando difícil para o administrador o estabelecimento da relação entre as políticas definidas e o seu modelo mental do sistema. Um trabalho posterior dos mesmos autores [Lymberopoulos et al., 2004] apresenta uma abordagem para implementação e validação de políticas Ponder para *Differentiated Services*, a qual usa o padrão DMTF CIM (*Common Information Model*) para modelar elementos de rede. O CIM se concentra na modelagem de informações de gerenciamento (e.g. capacidades e estados de dispositivos), enquanto que nosso modelo representa toda a estrutura relevante do sistema em conjunto com os componentes gerenciados.

A ferramenta gráfica Firmato [Bartal et al., 2004] parece ser a abordagem mais próxima da nossa, pois ela suporta o desenho interativo de políticas através de diagramas e deriva automaticamente as correspondentes configurações para mecanismos. Entretanto, como os níveis de abstração da definição das políticas e dos parâmetros de configuração são relativamente próximos, o suporte oferecido restringe-se a um nível de abstração que se aproxima dos próprios mecanismos, perdendo generalidade.

6. Conclusão

O presente trabalho apresenta uma abordagem para o gerenciamento da configuração de sistemas de segurança de redes de grande porte que se fundamenta em trabalhos anteriores do Gerenciamento Baseado em Modelos [Lück et al., 2002] e do Diagrama de Subsistemas Abstratos [Porto de Albuquerque et al., 2005a, Porto de Albuquerque et al., 2005b]. Estendemos esses trabalhos, apresentando um processo de projeto de sistema centrado no administrador e acompanhando a aplicação dos quatro passos desse processo em um cenário de exemplo. Além disso, integramos técnicas de foco e contexto na ferramenta-protótipo e analisamos questões relacionadas a essa integração.

O processo de projeto da configuração permite uma modelagem gradual e assistida a partir de um esquema abstrato até uma representação mais próxima do sistema real. Durante essa modelagem, as técnicas de foco e contexto empregadas beneficiam a navegação e visualização de grandes modelos, permitindo que o projetista defina em detalhes uma certa parte do modelo sem perder de vista o sistema como um todo, ou seja, mantendo-se consciente do contexto maior em que está trabalhando. Ao término do desenho do sistema, a ferramenta executa, então, um processo de refinamento que produz automaticamente arquivos de configuração para os mecanismos de segurança. Dessa forma, esperamos que o uso de nossa metodologia contribua para tornar o gerenciamento da configuração de sistemas de segurança mais eficiente e próximo do administrador, o qual pode, assim, definir as regras que regulam o comportamento do sistema em um nível de abstração mais inteligível, sem ter que se preocupar com detalhes de configurações de baixo nível.

Trabalhos futuros poderiam explorar a representação de políticas dos níveis inferiores do modelo, de forma a melhorar sua manipulação, possivelmente adaptando as técnicas empregadas no presente trabalho. Outra direção de pesquisa interessante nos parece ser a aplicação de testes de usabilidade para avaliação da ferramenta-protótipo.

Referências

Anderson, R. J. (1994). Why cryptosystems fail. *Communications of ACM*, 37(11):32–40.

Bartal, Y., Mayer, A. J., Nissim, K., and Wool, A. (2004). Firmato: A novel firewall management toolkit. *ACM Transactions on Computer Systems*, 22(4):381–420.

Card, S. K., Mackinlay, J. D., and Shneiderman, B., editors (1999). *Readings in Information Visualization: Using Vision to Think*. Series in Interactive Technologies. Morgan Kaufmann Publishers, San Francisco, CA.

Damianou, N., Dulay, N., Lupu, E., Sloman, M., and Tonouchi, T. (2002). Tools for domain-based policy management of distributed systems. In *IEEE/IFIP Network Operations and Management Symposium (NOMS2002)*, pages 213–218, Florence, Italy.

Furnas, G. W. (1986). Generalized fisheye views. In *Proceedings of ACM CHI'86 Conference on Human Factors in Computing Systems*, Visualizing Complex Information Spaces, pages 16–23.

Köth, O. and Minas, M. (2002). Structure, abstraction, and direct manipulation in diagram editors. In *Diagrammatic Representation and Inference, Second International Conference (Diagrams 2002)*, volume 2317 of *Lecture Notes in Computer Science*, Callaway Gardens, GA, USA. Springer.

Lymberopoulos, L., Lupu, E., and Sloman, M. (2004). Ponder policy implementation and validation in a CIM and differentiated services framework. In *IFIP/IEEE Network Operations and Management Symposium (NOMS 2004)*, Seoul, Korea.

Lück, I., Vögel, S., and Krumm, H. (2002). Model-based configuration of VPNs. In *Proc. 8th IEEE/IFIP Network Operations and Management Symposium NOMS 2002*, pages 589–602, Florence, Italy. IEEE.

Moffett, J. D. and Sloman, M. S. (1993). Policy hierarchies for distributed system management. *IEEE JSAC Special Issue on Network Management*, 11(9).

Musial, B. and Jacobs, T. (2003). Application of focus + context to UML. In *Australian Symposium on Information Visualisation, (invis.au'03)*, volume 24 of *Conferences in Research and Practice in Information Technology*, pages 75–80, Adelaide, Australia. ACS.

Oppenheimer, D., Ganapathi, A., and Patterson, D. (2003). Why do internet services fail, and what can be done about it. In *4th USENIX Symposium on Internet Technologies and Systems (USITS'03)*.

Porto de Albuquerque, J., Krumm, H., and de Geus, P. L. (2005a). On scalability and modularisation in the modelling of security systems. In *10th European Symposium on Research in Computer Security (ESORICS 05)*, volume 3679 of *LNCS*, pages 287–304, Heidelberg, Germany. Springer Verlag.

Porto de Albuquerque, J., Krumm, H., and de Geus, P. L. (2005b). Policy modeling and refinement for network security systems. In *Sixth IEEE International Workshop on Policies for Distributed Systems and Networks*, pages 24–33, Stockholm, Sweden.

Sandhu, R. S., Coyne, E. J., Feinstein, H. L., and Youman, C. E. (1996). Role-based access control models. *IEEE Computer*, 29(2):38–47.

Sarkar, M. and Brown, M. H. (1992). Graphical fisheye views of graphs. In *Proceedings of ACM CHI'92 Conference on Human Factors in Computing Systems*, Visualizing Objects, Graphs, and Video, pages 83–91.

Zwicky, E. D., Cooper, S., and Chapman, D. B. (2000). *Building Internet Firewalls*. O'Reilly and Associates, Sebastopol, CA, 2nd edition.